CHAIN MODEL AS A TOOL FOR IMPROVING QUALITY OF A RULE BASE OF A FUZZY MODEL

<excluded>

<excluded>
<excluded>

<excluded>

Abstract

One of the most popular approaches taken in the process of automatic creation of a rule base of a fuzzy model is a classic grid partitioning of an input space of the analyzed system. The basic feature of this approach is that a rule base created on the basis of this grid contains a lot of rules which can be eliminated from the model without a loss in a model precision. The unnecessary rules exist in the model because the grid is applied over the whole input space, regardless of the distribution of data points. Hence, in case of an unequal data distribution, some parts of the input space of the analyzed system are not covered by data points, which means that some rules are unsupported by any data point. Of course rules not supported by data points should not take part in the inference process because they can produce incorrect results in future model application. However, when these rules are eliminated from the rule base, a fuzzy model becomes an incomplete one. The theory states that an incomplete model either should not be used in practice at all or should be used only in some, very precisely defined, regions of an input space. In fact, there is one more possibility. The model completeness can be extorted by restoring some of previously discarded rules – of course after necessary changes in their conclusions. The aim of this article is to present an approach to chain systems modeling which can be used for: eliminating rules, calculating new conclusions for some of them and adding them again to the model.

Keywords: fuzzy model, model completeness, rules elimination, chain model, chain system.

1 Introduction

When a fuzzy model is created automatically on the basis of a data set describing dependences existing in the analyzed system, some general assumptions about the model architecture have to be made. One of them regards the partitioning of the input space of the analyzed system. Although, there are many different possibilities to deal with this task (e.g. fuzzy grid, fuzzy boxes, fuzzy clusters, fuzzy k-d tree [2]) only two of them are used in majority of practical applications – grid and cluster partitioning. Both approaches have their own benefits and drawbacks and suits better for other types of systems and for other aims. The main reason for using the first approach – cluster partitioning – is the possibility of finding only the most significant rules of the analyzed system behavior, that is rules situated in an input space in regions containing the vast majority of data [1, 3]. On the other hand, the main drawback of this approach is that the fuzzy sets are used rather by individual than by all rules. Since, it is impossible to ensure that the local membership functions have any semantic values, a linguistic interpretation of the rules is very difficult [4].

While cluster partitioning produces the local fuzzy sets, fuzzy sets obtained when grid partitioning is applied are global ones. That means they are not assigned to individual rules but are used by many rules from the whole universe. This enables their linguistic interpretation. Hence, when it is important to create a fuzzy model of high linguistic interpretability, the grid partitioning of an input space should be used.

The main drawback of a fuzzy model utilizing the grid partitioning is that it is very prone to the *curse of dimensionality* problem [1] – it is a problem of exponential growth of number of rules in relation to growth of number of input variables and number of membership functions. The *curse of dimensionality* problem is a very serious one because it significantly decreases the readability of the model. Fortunately, it can be overcome in case of most real systems. The fact is that most rule bases created on the basis of the grid partitioning of an input space contain a lot of unnecessary rules, it is rules which can be eliminated from them without the loss in the model precision. The unnecessary rules exist in the model because the grid is applied over the whole input space, regardless of the distribution of data points. Hence, in case of an irregular data distribution, some parts of the input space of the analyzed system are not covered by data points, which means that some rules are unsupported by any data point. Conclusions of these rules take in the process of model parameters estimation incidental values – in most cases values which were assigned to them at the beginning of the estimation process. Of course rules not supported by data points should not take part in the inference process because they can produce incorrect results in future model application. Therefore, these rules should be eliminated from the model rule base. This, however, will lead to an incomplete rule base.

The classic inference methods can be used only when the rule base of a fuzzy model is a dense one, it is when the input space is completely covered by rule premises. When this condition is not fulfilled it can happen that for certain observation no rule is fired and the inference mechanism fails [5]. Therefore, when the rule base is an incomplete one, it is necessary either to apply other inference method or to extort the model completeness by restoring some of previously discarded rules. While the approach for dealing with the first issue was proposed by Koczy and Hirota in [6], an approach which can be used to restore rules to a fuzzy model will be proposed in the paper.

The paper discusses three issues which should be taken into account when the quality of a fuzzy rule based is to be improved:

- first - how to find out which rules are really unnecessary ones and should be eliminated from the rule base,
- second – how to find out which rules should be restoring to the rule base in order to extort its completeness,
- third – how to calculate conclusions of restored rules.

Theoretically the solution of the first problem is simple – it should be enough to reveal which rules are not supported by any data point. In practice, however, such straightforward approach can result in leaving in a rule base rules which conclusions were calculated on the basis of remote data points (outliers). Assuming that outlying data points does not have any connection with the general law of the analyzed system, rules based on them should be also eliminated from the rule base. In order to deal with this task, it is to discard all unnecessary rules, a region containing the vast majority of data points used in the model parameters estimation process should be determined.

Addressing the second issue, it should be underlined that a fuzzy model can be applied in practical applications only in this region of the whole system domain which is covered by data points used in the estimation process. Hence, the model completeness can be extorted only inside this region which means that only rules which antecedents are situated inside this region can be restored to the rule base.

As it can be noticed, in order to deal with two first of the aforementioned issues, an interpolation region of a fuzzy model should be determined. In order to deal with this task some classic method can be applied, e.g. hypercube or convex hull [7]. Unfortunately, the classic methods of determining the interpolation region work properly only in case of systems of a surface data distribution. When a data distribution of a system is a non-surface but a chain one these methods are unable to produce a correct interpolation region of a fuzzy model.

The aim of this paper is to present an approach which can be used for determining the interpolation region of a fuzzy model of a system of a chain data distribution. The paper presents also the algorithms for both - eliminating and restoring rules to and from a rule base of a fuzzy model.

The paper is organized as follows. Section II provides general information on systems of a chain data distribution and their models. Section III gives an overview of methods of determining the interpolation region of a surface model and presents an approach to deal with this task in case of systems of a chain nature, first proposed by Rejer and Mikołajczyk in [8]. Section IV describes algorithms for eliminating and restoring rules from and to a rule base of a fuzzy model based on a chain model of the analyzed system. Finally, Section V presents the practical application of the discussed algorithms in a neuro-fuzzy model of a real economic problem – a problem of unemployment in Poland in years 1992-1999.

2 System of a chain data distribution

2.1 Chain system

The main feature of a multi-dimensional system of a chain data distribution is that its decomposition to one-dimensional subsystems (describing the behavior of each system variable in regard to parameter *t* - indicating the approximated data sequence) gives a set of tight chain dependences. Therefore, in order to verify whether a system is of a chain profile, the reverse analysis should be performed. Tight chain dependencies, visible on two-dimensional graphs presenting the behavior of all system variables in regard to *t* parameter, will indicate the chain profile of the system in the whole multi-dimensional space.

An important fact here is that in order to regard a system as a chain one all of its input variables should have a chain data distribution. Even one variable of a non-chain data distribution indicates that the whole system has a surface, not chain, nature. Fig. 1 and 2 present two sets of graphs describing the same real economic factor – unemployment rate in years 1992-1999 but in respect to different variables. Each graph shows the behavior of one system variable in regard to *t* parameter. Since the whole system is an example of time series, *t* parameter can be interpreted as a time variable. The first figure (Fig. 1) presents the dependency between *unemployment rate* and *money supply* and *number of inhabitants* and the second one (Fig. 2) presents the dependency between *unemployment rate* and *money supply* and *export*.

As it can be noticed in Fig. 1a,b,c all system variables are characterized by very tight chain data distributions. This indicates that also the whole system has a chain nature (Fig. 1d). On the other hand, only two out of three variables from the second figure have a chain data distribution (Fig. 2b,c), the third one has a surface characteristic (Fig. 2b). This means that the whole system cannot be regarded as a chain one (Fig 2d).

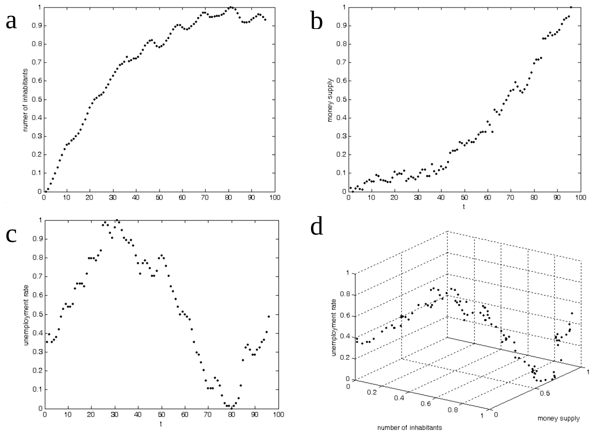

Fig. 1 System of unemployment rate; a, b, c) dependences between system variables and *t* parameter, b) input-output dependency

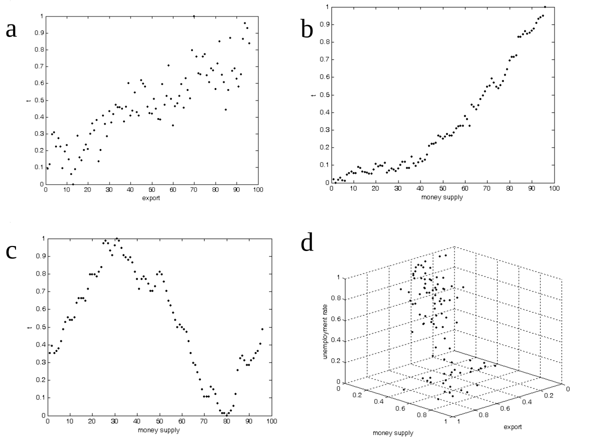

Fig. 2 System of unemployment rate; a, b, c) dependences between system variables and *t* parameter, b) input-output dependency

It should be noted here that while the conclusion of the surface nature of the analyzed system can be propagated onto the whole space of system variables, the conclusion of the system chain nature is valid only in a space limited to analyzed variables. It is due to the fact that it is impossible to analyze the system output variable in regard to all variables influencing it. That means that always variables can exist which have not been considered in the analysis and which can have a surface nature.

2.2 Chain model

The main benefit of the chain system is that it can be described not only by a surface model but also by a parametric curve [9]. The main idea of the parametric curve modeling method is to build a set of two-dimensional models, where each model describes the behavior of one variable (input or output) in regard to the known parameter *t*. These two-dimensional models can be created with many different mathematic techniques e.g. non-linear neural networks, polynomial regression, splines, etc. Two-dimensional models, built with one of the mentioned techniques, are then assembled together in order to create a multi-dimensional model describing the input-output mapping in the whole space [12]:

$$\begin{cases} x_1 = f_1(t) \\ x_2 = f_2(t) \\ \dots\dots\dots\dots \\ x_k = f_k(t) \\ y = f_{k+1}(t) \end{cases}, \qquad (1)$$

where: $x_1 \ldots x_k$ – input variables, *y* – output variable, *t* - parameter indicating the approximated data sequence.

3 Interpolation region of a fuzzy model

Interpolation and extrapolation are terms used when a new data point is constructed on the basis of a discrete set of known data points. The term interpolation is used when this new data point is situated in a range of known data points and the term extrapolation is used when the new data point is situated outside this range. Both terms can be referred to the modeling process. In this sense interpolation means calculating a model output for input values situated inside the region of data points used in the model parameters estimation process and extrapolation – calculating a model output for input values situated outside this region. While a properly trained neuro-fuzzy model should give correct results in an interpolation case, a question is whether it can be successfully used in an extrapolation case.

According to Niederlinski "there seems to be no engineering justification whatever for extrapolating any model, be it polynomial or be it neural, beyond the region of fit used in the identification experiment. On the contrary, there are plenty of counterexamples showing that systems described by models established for some region of fit may break down when driven beyond this region" [11]. Niederlinski words refer also to fuzzy model utilizing the grid partitioning of an input space. It is due to the fact that this model produces the same type of surface as a neural and polynomial model – it is a surface spread over the whole input space.

Since the application of a fuzzy model is justified only in case of data points located in a range of training data (an interpolation case), a very important issue is to determine properly the interpolation region of a fuzzy model. While determining this region for a two-dimensional model is a relatively easy task (it can be done on the basis of visual analysis of two-dimensional graphs), it can be very hard in a multi-dimensional case.

3.1 Hypercube and convex hull

The most popular approach used to determine a model interpolation region is to build a hypercube covering all data points used in a model parameters estimation process. The hypercube edges are established on the basis of minimal and maximal values of succeeding input variables. Fig. 3a presents an example of a three-dimensional hypercube built over a given data set of one thousand points.

The approach based on the hypercube is very easy to implement, however, it generates a very broad interpolation region, only partially covered by data points. A more strict approach to establish borders of the model interpolation region is to build a convex hull spread over all data points in a multi-dimensional input space.

A convex hull of a set of points *S* in *n* dimensions is the intersection of all convex sets containing *S*. For *N* points $p_1 \ldots p_n$, the convex hull *C* is given by [12]:

$$C \equiv \left\{ \sum_{j=1}^{N} \lambda_j p_j : \lambda_j \geq 0 \ \ for\ all\ j,\ and \sum_{j=1}^{N} \lambda_j = 1 \right\} \qquad (2)$$

An example of a convex hull built over the same set of one thousand points is presented in Fig. 3b. There are a lot methods which can be applied in the process of building the convex hull of a given data set e.g. Graham algorithm [13], divide and conquer algorithm [14], greedy algorithm [15], Beneath-Beyond algorithm [16], etc.

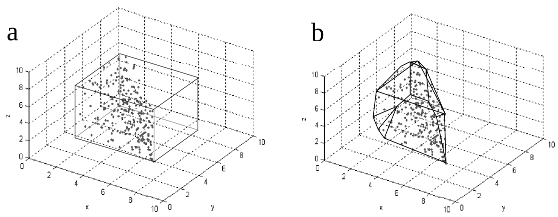

Fig. 3 Hypercube (a) and convex hull (b) in a three-dimensional space

The approach based on the convex hull generates a much more narrow interpolation region of the analyzed model than the approach based on the hypercube. However, when systems of a chain data distribution are analyzed, this region is still too large. This is due to the fact that the data distribution in chain systems is very often a non-convex one, which means that these systems should not be described by the convex hull. Fig. 4 presents a hypercube and a convex hull built over a data set given in Fig. 1.

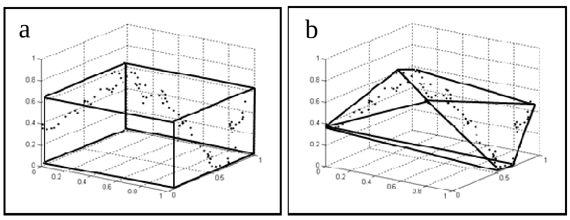

Fig. 4 Hypercube (a) and convex hull (b) of a data set given in Fig. 1

3.2 Hypertube

In order to determine the interpolation region of a model built over a data set of a chain data distribution, an approach reflecting the chain characteristic of this distribution should be used. The best tool which allows to deal with this task is a chain model – it is a parametric curve built over a given data set.

The parametric curve can be used in the process of establishing the interpolation region of a fuzzy model because it shows the very center of the data distribution in a multi-dimensional space. Obviously, the knowledge of the center of data points is not sufficient to establish the model interpolation region. The second point which has to be addressed is a width of this region in a multi-dimensional space. Assuming that the width of this region is the same in each direction, it can be calculated as a radius of a hypertube surrounding the chain model in a multi-dimensional input space.

Theoretically, the distance between the most remote data point and the model seems to be appropriate for establishing the radius of the hypertube. In fact, however, this measure is a proper one only when artificially generated data sets of normal distribution are considered. In case of real systems of an unknown data distribution, the distance between the most remote data point and the model cannot be used as the hypertube radius because of the outliers problem, often met in real systems [13]. Hence, instead of the greatest distance, the distances between all training data points and the chain model, calculated in a multi-dimensional input space should be considered.

Taking into account above considerations, the general equation of the hypertube radius can be formulated on the basis of three first quartiles of the absolute distances between training data and the chain model:

$$R_h = Q3 + \min(Q1, Q2 - Q1, Q3 - Q2), \qquad (3)$$

where: *Q1* - first quartile indicating the region surrounding the chain model covered by 25% of data points, *Q2* - second quartile indicating the region surrounding the chain model covered by 50% of data points, *Q3* – third quartile indicating the region surrounding the chain model covered by 75% of data points, R_h - radius of the hypertube surrounding the chain model in a multi-dimensional input space.

Formula (3) is valid for both artificial and real systems because the statement: *min(Q1, Q2-Q1, Q3-Q2)* produces a value which covers the whole input space in case of artificially generated data sets and preserves the outliers problem in case of real systems. Of course formula (3) is a general one. When the characteristic of the modeling system is well known and it is possible to determine the rough amount of outliers in the data set, this equation can be adopted to the system characteristic by applying values of other percentiles of the absolute distances between training data and the chain model.

Summing up, the interpolation region of a fuzzy model can be determined as the intersection of a fuzzy model and the hypertube of the radius given by (3) in an input space. Hence, the interpolation region is a set of points $(x_1, \ldots x_n)$ satisfying the following set of equations:

$$\begin{cases} \frac{\partial x_1(t)}{\partial t}\cdot(x_1 - x_1(t)) + \ldots + \frac{\partial x_n(t)}{\partial t}\cdot(x_n - x_n(t)) = 0, \\ (x_1 - x_1(t))^2 + \ldots + (x_n - x_n(t))^2 = R_h^2 \end{cases} \qquad (4)$$

Fig. 5 presents a fuzzy model and a hypertube built over a data set given in Fig. 1

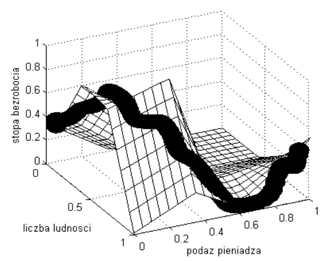

Fig. 5 Hypertube built over a data set given in Fig. 1

4 Eliminating and adding rules to and from a rule base

4.1 Algorithm for rules elimination

The basic idea of the proposed algorithm for rules elimination is to identify all rules overlapping the hypertube in the input space. It is of no importance whether a rule covers a large or a small part of a hypertube – it is essential to find out all appropriate rules. At the end of the search process chosen rules remain in the model and the rest of them is discarded.

In order to find out whether a rule overlaps a hypertube, the absolute distances between the boundaries of a rule hypercube and the chain model (which shows the center of the hypertube) in all input dimensions have to be calculated. If all distances are smaller than the hypertube radius (given by (3)), that means the rule overlaps the hypertube and should remain in the model. Obviously, only one of two possible hypercube boundaries (this situated closer to the hypertube) is taken into account in each input dimension.

A detailed algorithm for rule reduction can be described as follows:

1. The chain model is equally sampled in a large number of points.
2. For each sample and for each input dimension:

2.1. two new points are created – by adding and subtracting a hypertube radius to/from the sample,
2.2. the universe of membership functions is searched and two membership functions are chosen - these which supports contain one of two previously created points,
2.3. all membership functions situated between two previously established functions are chosen.
3. The whole universe of rules is searched and the rules which all premises contain any of the membership functions chosen for succeeding dimensions are selected.
4. Remaining rules are eliminated from the model rule base.

4.2 Algorithm for changing rules conclusions

After application of the algorithm presented in the previous section, the rule base of a fuzzy model contains all rules lying inside or at the borders of the hypertube. Since the algorithm does not check whether rules left in the model are covered by data points, some of these rules can have random conclusions – conclusions which were assigned to them at the beginning of the model parameters estimation process. Of course, rules of random conclusions should not take part in the inference process because they can produce incorrect results in future model application. Therefore, either these rules should be discarded from the model or their conclusions should be changed. Since discarding rules would lead to a sparse rule base, the second possibility – it is evaluating conclusions of rules not covered by data points – is proposed to apply.

The most straightforward approach to deal with mentioned issue is to calculate rules conclusions on the basis of conclusions of neighboring rules. Hence, in order to establish a conclusion of an individual rule, all rules which antecedents are situated next to the antecedents of the analyzed rule in the whole multi-dimensional input space have to be revealed. Then, the conclusions of all rules are gathered together and a conclusion of the analyzed rule is calculated as a simple or weighted average.

The solution mentioned above seems to be an easy and a quick one but in fact, when multi-input systems are under consideration, the task of finding neighboring rules of all rules which conclusions should be calculated becomes a very challenging one. Therefore, other approaches to deal with this task are needed.

In case of systems of a chain data distribution a chain model of the analyzed system can be used to calculate conclusions of rules unsupported by data points. Since the chain model presents the behavior of the whole system for succeeding values of *t* parameter, in order to calculate rule conclusion, cores of fuzzy sets used in rule premises should be projected onto this model.

A detailed algorithm for calculating conclusions of rules (left in the model after applying algorithm from Sect. 4.1) unsupported by data points consists of five following steps:

1. Find a rule unsupported by any data point.
2. Find a point in an input space which fully supports premises of the rule found in the first step (it is a point which succeeding coordinates are cores of fuzzy sets contained in succeeding premises of the rule).
3. Project this point onto the chain model (it is – find a point of the chain model which lies at the closest distance from the point which was found in the second step),
4. Calculate *t* value of the point lying on the chain model using formula (1).
5. Calculate the conclusion of the rule by introducing *t* value obtained in the fourth step to the two-dimensional equation describing the behavior of the output variable of the analyzed system in regard to the parameter *t*.

All five steps of the algorithm should be performed for each rule unsupported by data points, left in the model after applying algorithm from Sect. 4.1. That means that the number of iterations of the algorithm is equal to the number of rules which conclusions are to be calculated.

The assumption that only conclusions of rules not supported by any data point should be changed is a general one. In case of some real systems it can occur that this assumption is too weak and not only rules unsupported by any data point but also rules supported by one, two or more data points should be taken into account in the process of changing conclusions. Obviously, in such case the first step of the algorithm should be changed. This, however, does not change the general idea of the proposed algorithm.

5 Case study

The practical application of the approaches described in Sect. 3 and 4 will be presented via a neuro-fuzzy model of a real economic system of an unemployment rate in Poland in years 1992-1999 (output variable - *unemployment rate*, input variables - *money supply* and *number of inhabitants*). Data for the case study were provided by Polish Statistic Department. To prepare data from the training set for estimating model parameters, all variables (input and output) were normalized to the interval <0, 1>. The basic model parameters were as follows:

- model type – Larsen model, given by (5) [18],
- input membership functions - asymmetrical triangular functions (5 functions per each input variable),
- output membership functions – 25 singleton functions,

- training algorithm – backpropagation algorithm with momentum rate,
- training time – 1000 epochs.

$$\bar{y} = \frac{\sum_{i=1}^{m} \bar{y}_i(\mu_{Bi}(\bar{y}_i)\prod_{j=1}^{s}\mu_{Aj}(\bar{x}_j))}{\sum_{i=1}^{m}(\mu_{Bi}(\bar{y}_i)\prod_{j=1}^{s}\mu_{Aj}(\bar{x}_j))} \quad (5)$$

where: $\bar{y}$ - output variable, $\bar{x}_j$- input variable j ($j=1,...,s$), $\bar{y}_i$- center of fuzzy set B_i, it is a point in which $\mu_{Bi}(\bar{y}_i)$ is maximal, $\mu_{Aj}(\bar{x}_j)$ - degree of activation of j premise of a rule which conclusion is equal to $\bar{y}_i$.

Figure 6 illustrates the model surface (fig. 6a) and its rule net in the input space (fig. 6b). The mean absolute model error (MAE), given by (6) was equal to 3.90%.

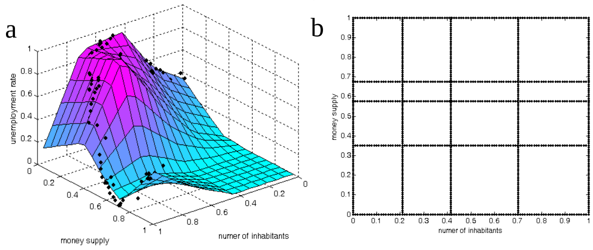

Fig. 6 Neuro-fuzzy model of unemployment rate: a) model surface, b) rule net

$$MAE = \frac{\sum_{k=1}^{n}\left|y_k^* - y_k\right|}{n} * 100\% , \quad (6)$$

where: y_k* - real values, y_k – theoretical values.

5.1 Chain model of an unemployment rate

The first step taken to improve the quality of the rule base of the analyzed fuzzy model was to build a chain model containing its input variables. In order to deal with this task two two-dimensional models describing the behavior of each input variable of the analyzed system in regard to *t* parameter were created. As a modeling tool neural networks of the following parameters were used [13]:

- flow of signals – one-way,
- architecture of connections between layers – all to all,
- hidden layers – 1 hidden layer with suitable number of sigmoid neurons (5 for variable *number of inhabitants*, 3 for variable *money supply*),
- output layer – 1 linear neuron,
- training method – backpropagation algorithm with momentum rate,
- training time – 20000 epochs.

By assembling together neural models built for both variables (Fig. 7a,b) the parametric curve model (chain model) was created (Fig. 7c).

Next step was to determine the interpolation region of the fuzzy model. In order to deal with this task, a radius of the hypertube covering the majority of data points was calculated according to formula (3). It was equal to 0.0387. Fig. 7d presents the hypertube surrounding the chain model in the input space over the rule net of the fuzzy model.

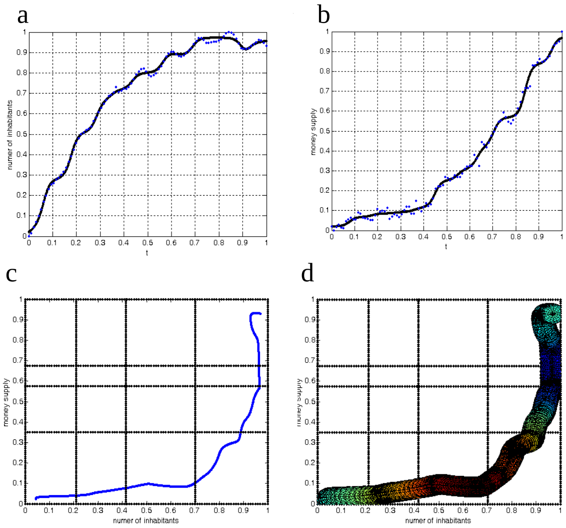

Fig. 7 Chain model of unemployment rate: a, b) two-dimensional time series models, c) three-dimensional parametric curve model, d) hypertube

5.2 Eliminating rules

The interpolation region of the fuzzy model was established in order to find out which rules can be eliminated from the fuzzy model without loss in the model precision. The rule elimination was performed according to the algorithm presented in Sect. 4.1. Hence, first, the chain model was equally sampled in 1000 points. Then, the value of the hypertube radius was added and subtracted to and from all samples in all dimensions and appropriate set of border membership functions per each dimension was established. Numbers of chosen membership functions are presented in tab. 1.

Tab. 1 Numbers of border membership functions in both input dimensions

Number of inhabitants		Money supply	
1st border function	2nd border function	1st border function	2nd border function
1	5	1	2
4	5	1	5

Next, the set of chosen membership functions was expanded by adding (for each input dimension) membership functions situated between the functions from tab. 1. And finally, the universe of rules was searched and rules which premises (in all dimensions) contain any of the membership functions chosen for succeeding dimensions were determined. Fig. 8 presents the set of rules which were left in the model after applying the proposed algorithm and Tab. 2

presents the numbers of membership functions used in both rule premises in succeeding rules.

Tab. 2 Numbers of membership functions used in the premises of rules left in the two-input fuzzy model

Rule number	Premise 1 Number of inhabitants	Premise 2 Money supply
1	1	1
2	1	2
3	2	1
4	2	2
5	3	1
6	3	2
7	4	1
8	4	2
9	4	3
10	4	4
11	4	5
12	5	1
13	5	2
14	5	3
15	5	4
16	5	5

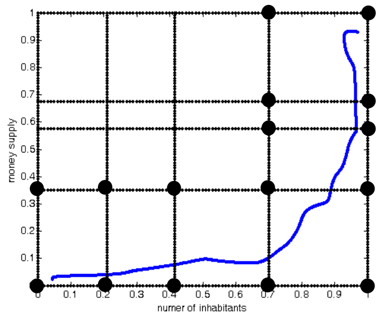

Fig. 8 Rules left in the rule base of the two-input fuzzy model

The application of the proposed method for rule reduction allowed to simplify the rule base of the unemployment rate model by eliminating 9 out of 25 rules. That means 36% of rules were identified as unnecessary ones and eliminated from the model. The MAE of the fuzzy model containing the reduced rule base was equal to 3.90%. Since this error value is exactly equal to the value of the original model, it can be said that the proposed method correctly identifies unnecessary rules.

5.3 Calculating rules conclusions

For further improvement of the rules base of the analyzed fuzzy model, the conclusions of rules unsupported by data points should be calculated. However, since data from the data set used in the survey covered all rules situated inside the interpolation region of the fuzzy model, rules conclusions recalculation was unnecessary.

Hence, in order to verify the practical applicability of the proposed approach for calculating conclusions of rules unsupported by data points, some data from the whole data set were removed and a second fuzzy model of the unemployment rate was created. This time the data set was composed of data points from the first and the last 36 months of the analyzed 96-months period. Remaining 24 data points (coming from the middle part of the set) were left for the verification process. Such supervised allotment of data set was necessary because in order to check the quality of the proposed approach a large (continuous) part of a fuzzy model had to be deprived of data points.

The experiment was composed of three steps. In the first step the limited set of data points was used to build two models of the analyzed system – a fuzzy model and a chain model. All parameters of both models were exactly the same as presented in Sect. 5 and 5.1, respectively. MAE of the fuzzy model calculated over the limited data set (72 data points) was equal to 3.71% and MAE of the same model calculated over the whole data set (96 data points - 24 of them not used in the model parameters estimation process) was equal to 5.68%.

In the second step of the experiment, the interpolation region of the fuzzy model was created (R_h=0.099) and 9 rules situated outside this region were eliminated. The eliminated rules were the same as in the model from Sect. 5.2.

Finally, in the last step, two rules situated inside the interpolation region, not-supported by any data point were discovered. The conclusions of these rules were calculated according to the algorithm presented in Sect. 4.2. MAE of the corrected fuzzy model calculated over the limited data set (72 data points) was of course equal to MAE of not-corrected fuzzy model (3.71%) and MAE of the same model calculated over the whole data set (96 data points) was equal to 4.6%.

The results of the experiment described above confirm the practical usefulness of the proposed method for calculating rules conclusions. The comparison of errors of fuzzy models with old and new values of two rules conclusions calculated on the basis of the whole data set (MAE equal to 5,68% and 4,6%, respectively), shows that after the application of the proposed method, the precision of the fuzzy model had been increased. That means the proposed method correctly calculated conclusions of rules unsupported by data points.

6 Conclusion

The aim of this article was to present methods which can be used for improving quality of a fuzzy model, mainly by reducing unnecessary rules from its rule base. The main advantages of the proposed methods are as follows:

- The reduction rate is a significant one (in presented application it was equal to 36%) which is very important not only when a model is used in its software version as a tool supporting a human but also when a model is a base for a hardware implementation.

- The reduced model has the same rate of precision as the non-reduced ones. That means the algorithm eliminates rules which are really unnecessary and do not take part in the inference process.
- The model obtained after applying the proposed methods is a continuous ones.

The methods were presented via a chain system. However, since both of them are based on the interpolation region of a fuzzy model, their main idea can be applied also to improve quality of a fuzzy model built for a system of a surface data distribution which interpolation region can be established with the classic techniques.

7 References

[1] Piegat A.: Fuzzy Modeling and Control. Physica-Verlag, New York (1999)

[2] Ojala T.: Neuro-fuzzy systems in control, Master of Science Thesis, Tampere University of Technology, Department of Electrical Engineering, Finland (1994)

[3] Pavia R. P., Dourado A.: Structure and parameter learning of neuro-fuzzy systems: A methodology and a comparative study, Journal of Intelligent & Fuzzy Systems No. 11 (2001)

[4] Xiong N.: Evolutionary learning of rule premises for fuzzy modelling, International Journal of Systems Science, Vol. 32, No. 9 (2001)

[5] Wong K. W., Tikk D., Gedeon, T. D., Kóczy L.: Fuzzy Rule Interpolation for Multidimensional Input Spaces With Applications: A Case Study, IEEE Transaction on Fuzzy Systems, Vo. 13, No. 6, December (2005)

[6] Kóczy L. T., Hirota K.: Approximate reasoning by linear rule interpolation and general approximation. Int. J. Approx. Reason., vol. 9 (1993)

[7] Klesk P.: The method of setting suitable extrapolation capabilities for neuro-fuzzy models of multidimensional systems, PhD Thesis, Technical University of Szczecin (2005)

[8] Rejer I, Mikołajczyk M.: A Hypertube as a Possible Interpolation Region of a Neural Model, Artificial Intelligence and Soft Computing ICAISC 2006 - Lectures Notes in Artificial Intelligence, Springer-Verlag, Berlin (2006)

[9] Piegat A., Rejer I., Mikołajczyk M.: Application of Neural Networks in Chain Curve Modelling. Artificial Intelligence and Soft Computing ICAISC 2006 - Lectures Notes in Artificial Intelligence, Springer-Verlag, Berlin (2006)

[10] Bronsztejn I.N., Siemiendiajew K.A., Musiol G., Mühlig H.: Modern Compendium of Mathematic, Polish Science Publisher Warsaw (2004)

[11] Niederlinski A.: Polynomial and Neural Input-Output Models for Control – a Comparison, MMAR'97, Poland (1997)

[12] Weisstein, E. W.: Convex hull, Published by Wolfram Research. From Math World, http://mathworld.wolfram.com/ConvexHull.html (2006)

[13] Masters T.: Practical Neural Networks Recipes in C++. Academic Press Inc. (1993)

[14] Graham, R.L.: An Efficient Algorithm for Determining the Convex Hull of a Finit Planar Set, Information Processing Letters 1, pp. 132-133 (1972)

[15] Preparate F. C., Hong S. J.: Convex Hulls of Finite Sets of Points in Two and Three Dimensions, Commun. ACM, vol. 20, no. 2, pp. 87–93 (1977)

[16] Klęsk P.: Algorithm for automatic definition of validated and nonvalidated region in multidimensional space, 10th International Conference on Advanced Computer Systems, (2003)

[17] Krivsky S., Lang B., Verified computation of HigherDimensional convex hulls and the solution of linear systems, Electronic Journal on Mathematics of Computation (2003)

[18] Rutkowska, D., Piliński, M., Rutkowski, L.: Neural networks, genetic algorithms and fuzzy systems, Warsaw: Scientific Publishing House Ltd. (1999)