**Control and Cybernetics**

vol. **31** (2002) No. 2

# Neuro-fuzzy TSK network for approximation of static and dynamic functions

by

**Tran Hoai Linh**[1] **and Stanisław Osowski**[1,2]

[1] Warsaw University of Technology, 00-661 Warsaw, pl. Politechniki 1

[2] Military University of Technology, Warsaw, email: sto@iem.pw.edu.pl

**Abstract:** The paper presents the neuro-fuzzy network in application to the approximation of the static and dynamic functions. The network implements the Takagi–Sugeno inference rules. The learning algorithm is based on the hybrid approach, splitting the learning phase into two stages: the adaptation of the linear output weights using the SVD algorithm and the conventional steepest descent backpropagation rule in application to the adaptation of the nonlinear parameters of the membership functions. The new approach to the generation of the inference rules, based on the fuzzy self-organization is proposed and the algorithm of automatic determination of the number of these rules has been also implemented. The method has been applied for the off-line modelling of static nonlinear relations and on-line simulation of the dynamic systems.

**Keywords:** neurofuzzy networks, learning algorithms, approximation

## 1. Introduction

In recent years several authors have used fuzzy models instead of the multilayer perceptron for approximation purposes (Duch et al., 2000, Ishibuchi, 1997, Jang et al., 1997, Lin et al., 1995, Takagi and Sugeno, 1985, Guillaume, 2001). Two groups of approaches can be recognized. One group consists of neural networks with fuzzy weights and fuzzy activation functions, each processing crisp signals (Ishibuchi,1997). In the second group, the input signals are fuzzified in the first or second layer, but the neural network weights are not fuzzy (Jang et al., 1997, Lin et al., 1995, Wang, 1994, Guillaume, 2001).

This paper belongs to the second group. We present the fuzzy neural network structure based on the Takagi–Sugeno linear inference rule, from to the ANFIS family (Jang et al., 1997). It is a multilayer (typically five-layers) network, in which there is a strict division between the nonlinear parameters of the fuzzy

split the learning algorithm into two stages: the adaptation of the linear output weights in the first stage and then learning of the nonlinear parameters using backpropagation algorithm. The adaptation of the linear weights is done using the SVD approach. At this stage the nonlinear parameters are fixed. In the second phase the output weights are constant and the nonlinear parameters are adjusted using steepest descent optimization rule and the backpropagation strategy.

The important problem in the learning of this network is the generation of rules. In this paper we propose application of self-organization for creation of rules. The fuzzy Gustafson-Kessel (GK) self-organizing algorithm splits the data into overlapping clusters. Each cluster is associated with the inference rule, whose center is placed in the middle of the cluster. The number of clusters is generated automatically using specially adopted measures of quality.

The proposed solution has been tested on different examples of approximation tasks. The investigations include the static and dynamic systems. The static system is understood as the algebraic nonlinear relationship between input and output signals. In such systems the output response to the input does not change with time. It means that the output always has the same instantaneous relationship with the input. The description of such system does not need the use of differential or difference equations. On the other hand the dynamic systems have a response to an input that is not instantaneously proportional to the input or disturbance and that it may continue after the input is held constant. The description of dynamic system is always associated with the use of differential or difference equations.

## 2. Fuzzy systems

The classical set is a set with a crisp boundary. The variable either belongs or does not belong to the set. In contrast to a classical crisp set, a fuzzy set is defined without crisp boundary, where the transition between "belonging to a set" and "not belong to a set" is gradual and this transition is characterized by the membership functions in the range $[0, 1]$ that give fuzzy sets flexibility in modelling (Zimmerman, 1985, Yager and Filev, 1995).

The membership may be described either in a discrete form as a set of membership values or as a continuous function valid in some ranges of values of the variable $x$. The most popular types of membership functions are the triangle, the trapezoidal, the gaussian and the bell functions. We will use here the generalized description of the bell function, given in the form (see Jang et al., 1997, Osowski, 2000)

$$\mu_F(x) = \frac{1}{1 + \left(\frac{x-c}{\sigma}\right)^{2b}}. \tag{1}$$

The shape of this function is controlled by three parameters: the center $c$, the

gaussian membership function, at $b \cong 0.6$ it is a triangle and at $b > 3$ the function is transformed to the trapezoidal shape.

The most popular solution of the fuzzy networks is based on the so called fuzzy inference system, fuzzy *if – then* rules and fuzzy reasoning. Such fuzzy inference system implements a nonlinear mapping from the input space to the output space. This mapping is accomplished by a number of fuzzy *if– then* rules, each of which describes the local behavior of the mapping, like it is done in radial basis function networks (Jang et al., 1997, Osowski, 2000). The antecedent of the rule defines the fuzzy region in the input space, while the consequent specifies the output of the fuzzy region.

There are different solutions of fuzzy inference systems. The most known include the Mamdani fuzzy model, the Tsukamoto and the Takagi–Sugeno–Kang (TSK) ones, defined by Takagi and Sugeno in the paper (Takagi and Sugeno, 1985). In this paper we will consider only the TSK model. A typical fuzzy rule in this model has the form

$$\text{if } x_1 \text{ is } A_1 \text{ and } x_2 \text{ is } A_2 \cdots \text{ and } x_N \text{ is } A_N \text{ then } y = f(\mathbf{x}) \tag{2}$$

where $\mathbf{x} = [x_1, x_2, \cdots, x_N]^T$, $A_1, A_2, \ldots A_N$ are fuzzy sets in the antecedent, while $y$ is a crisp function in the consequent. The function $y = f(\mathbf{x})$ is a polynomial in the input variables $x_1, x_2, \ldots, x_N$. We will apply here the linear form of this function. The aggregated values of the membership function for the vector $\mathbf{x}$ may be assumed either in the form of *MIN* operator or in a product form. For $M$ fuzzy rules of the form (2) we have $M$ such membership functions $\mu_1, \mu_2, \ldots, \mu_M$. We assume that each antecedent is followed by the consequent of the following linear form

$$y_i = p_{i0} + \sum_{j=1}^{N} p_{ij} x_j \tag{3}$$

where $p_{ij}$ are the adjustable coefficients, for $i = 1, 2, \ldots, M$, and $j = 1, 2, \ldots, N$.

When we apply the fuzzy singleton, the generalized bell membership function described by equation (1), and the algebraic product aggregation of the input variables, given the existence of $M$ rules the neuro-fuzzy TSK system output signal $y(\mathbf{x})$ excitation by the vector $\mathbf{x}$ is described by the equation (Jang et al., 1997, Osowski, 2000)

$$y(\mathbf{x}) = \frac{1}{\sum_{r=1}^{M} [\Pi_{j=1}^{N} \mu_r(x_j)]} \sum_{k=1}^{M} \left( [\Pi_{j=1}^{N} \mu_k(x_j)] \left[ p_{k0} + \sum_{j=1}^{N} p_{kj} x_j \right] \right). \tag{4}$$

The adjusted parameters of the system are the nonlinear parameters ($c_j^{(k)}$, $\sigma_j^{(k)}$, $b_j^{(k)}$) for $j = 1, 2, \ldots, N$, and $k = 1, 2, \ldots, M$, of the fuzzifier functions and the linear parameters (weights $p_{kj}$) of TSK functions. Contrary to the Mamdani fuzzy inference system (Yager and Filev, 1995) the TSK model generates the crisp output value instead of a fuzzy one. Thanks to this the structure of the network is simplified, since the defuzzifier is not necessary.

## 3. Neuro-fuzzy network structure

The TSK fuzzy inference systems described by equation (4) can be easily implemented in the form of the so called neuro-fuzzy network structure. Fig. 1 presents the five-layer structure of the neuro-fuzzy network, realizing the TSK model of the fuzzy system (Jang et al., 1997, Mitra and Hayashi, 2000). It

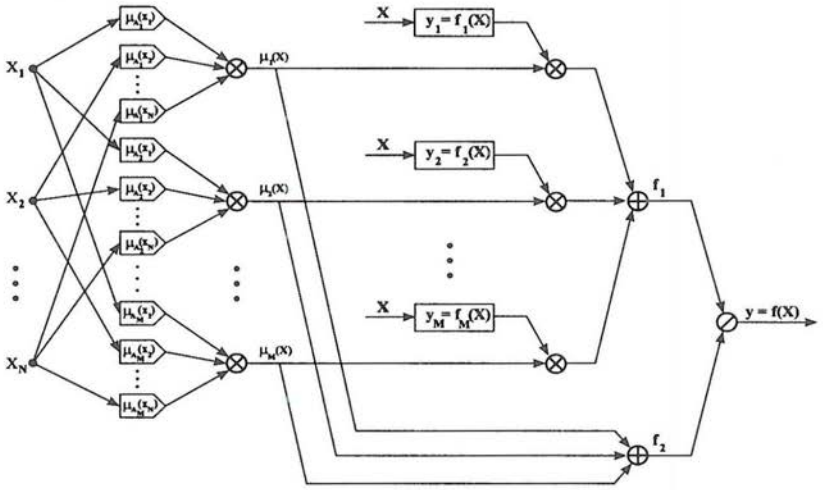


Figure 1. The structure of neuro-fuzzy TSK model of fuzzy system

is assumed that the functions $y_i$, $y_i = f_i(\mathbf{x})$ are linear of the form $f_i(\mathbf{x}) = p_{i0} + \sum_{j=1}^{N} p_{ij}x_j$. The adaptable parameters of the network are the variables of the membership functions $c_j^{(k)}$, $\sigma_j^{(k)}$, $b_j^{(k)}$ for $j = 1, 2, \ldots, N$, $k = 1, 2, \ldots, M$ and the coefficients (linear weights) $p_{ij}$ for $i = 1, 2, \ldots, M$ and $j = 0, 1, 2, \ldots, N$ of the linear Takagi–Sugeno functions.

The network on Fig. 1 has a multilayer form. The first layer performs the fuzzification according to the membership function $\mu_k(x_j)$, described by the relation (1). The second layer aggregates the fuzzified results of the individual scalar functions of every variable and determines the membership function of the whole vector $\mathbf{x}$. This is the product type aggregation. Each node of this layer represents the firing strength of a rule. The third layer calculates the aggregated signal of the fuzzy inference for each inference rule. The output signal of each unit of this layer is the product of the firing strength of the rule and the consequent membership value. The fourth layer calculates only the sum of the signals of the second and the third layers of the network. The final fifth

many output neurons as needed in a fashion similar to the case of one output. The output neuron computes the overall output signal according to relation (4). Thus we have constructed the neuro-fuzzy network that is functionally equivalent to the Takagi-Sugeno fuzzy model. Only the first and third layers are parametric. The parameters of the first layer are associated with the nonlinear functions and the weights $p_{ij}$ of the third layer are linear.

## 4. Hybrid learning algorithm

The learning of the neuro-fuzzy network, that is, the adaptation of the parameters of the first ($c_j^{(k)}$, $\sigma_j^{(k)}$, $b_j^{(k)}$) and the third ($p_{ij}$) layers of the network can be done either in supervised or self-organizing mode. For the purpose of approximation more efficient and straighforward is the supervised one.

In practical implementation we have applied the so called hybrid approach (Jang et al., 1997, Osowski, 2000). In this method we take into account that the network is linear in the parameters $p_{ij}$, thus we can identify these linear parameters by a linear least squares method based on singular value decomposition (SVD). At this stage we assume that all nonlinear parameters are fixed. This is the first run of the learning stage. In the second run we fix the linear parameters of the network and apply the steepest descent method for the estimation of the nonlinear parameters of the membership functions.

Thus, in hybrid learning each iteration is composed of a forward pass and a backward one. In the forward pass, after the input vector is presented, we calculate the node outputs in the network layers and on the basis of this the linear parameters $p_{ij}$ are adjusted using pseudoinverse on the basis of the SVD technique. After linear parameters are identified we can compute the error for training data pairs. In the backward pass the error signals propagate from the output end toward the input nodes; the gradient vector is calculated and the nonlinear parameters $c_j^{(k)}$, $\sigma_j^{(k)}$, $b_j^{(k)}$ updated by steepest descent method. The learning step of the nonlinear parameters update is adjusted using adaptive approach (Osowski, 2000). This process is repeated many times until there is a sufficient change of the values of the adapted parameters of the network. The details of the algorithm may be found for example in Jang et al. (1997), Osowski (2000).

In practical implementation of this type of learning the dominant stage of adaptation is the first one, adjusting the linear weights $p_{ij}$ in one step using SVD. The second step applying the steepest descent strategy of learning the nonlinear parameters is much less efficient. To get a balanced algorithm one use of SVD has been associated with 10 repetitions of the steepest descent algorithm.

The important advantage of the hybrid algorithm is splitting of the learning process into two independent stages: the adaptation of linear weights and the

to this the overall complexity of the algorithm has been decreased and at the same time the efficiency of learning increased.

## 5. Rule generation using fuzzy self-organization

One of the most important stages of the neuro-fuzzy TSK network generation is the establishment of the inference rules. The most often used is the so called grid method, in which the rules are defined as the combinations of the membership functions for each input variable. If we split the input variable range into limited number (say $n_i$ for $i = 1, 2, \ldots, N$) of membership functions, the combinations of them lead to many different inference rules. In general it is possible to create as many as $M = \Pi_{i=1}^{N} n_i$ rules in this way. Even at a small number of input variables and a moderate number of membership functions in each variable we get large number of inference rules. For example for a 10-input system, 3 membership functions each, the maximum possible number of rules is equal $M = 3^{10} = 59049$. The problem is that these combinations correspond in many cases to the regions of no data, hence a lot of them may be deleted.

We solve this problem by using fuzzy self-organization algorithm. This algorithm splits the data space into specified number of overlapping clusters. Each cluster may be associated with the specific rule of the center corresponding to the center of the appropriate cluster. In this way all rules correspond to the regions of the space containing the majority of data and thanks to this we avoid the problem of empty rules.

### 5.1. The Gustafson–Kessel algorithm

Assume that the vectors of data under clusterization are denoted by $\mathbf{x}_k$, where $\mathbf{x}_k \in R^N$ for $k = 1, 2, \ldots, p$. Let these vectors be partitioned into $c$ clusters, each represented by individual center vector $\mathbf{c}_i = [c_{i1}, c_{i2}, \cdots, c_{iN}]^T$. Denote by $\mathbf{U} \in R^{c \times p}$ the partition matrix of the elements $u_{ij}$ representing the membership degrees of the data vector $\mathbf{x}_j$ ($j = 1, 2, \ldots, p$) in the $i$th cluster ($i = 1, 2, \ldots, c$). The main task of fuzzy clustering is the partition of the data space in such a way that the following objective function $E$ is minimized (Jang et al., 1997, Osowski, 2000)

$$E = \sum_{i=1}^{c} \sum_{j=1}^{p} u_{ij}^m d^2(\mathbf{x}_j, \mathbf{c}_i) \tag{5}$$

subject to

$$\sum^{c} u_{ij} = 1 \tag{6}$$

for $j = 1, 2, \ldots, p$ and

$$0 \leq \sum_{j=1}^{p} u_{ij} \leq p \tag{7}$$

for $i = 1, 2, \ldots, c$. The parameter $m$ controls the fuzziness of the clusters (usually $m = 2$). The function $d(\mathbf{x}_j, \mathbf{c}_i)$ represents the distance between the data vector $\mathbf{x}_j$ and the center $\mathbf{c}_i$ of $i$th cluster.

One of the most efficient fuzzy clustering algorithms, able to take into account different shapes of clusters, is the GK algorithm, extending the $c$-means algorithm (Pal and Bezdek, 1995) by using the scaled metric norm for distance (Gustafson and Kessel, 1979). According to this algorithm for the given data vectors $\mathbf{x}_j$ choose the number of clusters $c$, the weighting coefficient $m$ and the termination tolerance $\varepsilon$. At the beginning the partition matrix $\mathbf{U}$ is initialized randomly in such a way that the appropriate conditions are satisfied. Then the following steps are iterated:

1. Determine the cluster prototype centers $\mathbf{c}_i$ for $i = 1, 2, \ldots, c$
$$\mathbf{c}_i = \frac{\sum_{j=1}^{p} u_{ij}^m \mathbf{x}_j}{\sum_{j=1}^{p} u_{ij}^m}. \tag{8}$$
2. Calculate the cluster covariance matrices $\mathbf{F}_i$ $(i = 1, 2, \ldots, c)$ according to
$$\mathbf{F}_i = \frac{\sum_{j=1}^{p} u_{ij}^m (\mathbf{x}_j - \mathbf{c}_i)(\mathbf{x}_j - \mathbf{c}_i)^T}{\sum_{j=1}^{p} u_{ij}^m}. \tag{9}$$
3. Compute the distances $d_{ij}^2$ $(i = 1, 2, \ldots, c$ and $j = 1, 2, \ldots, p)$ between the input vector $\mathbf{x}_j$ and cluster centers $\mathbf{c}_i$
$$d_{ij}^2 = (\mathbf{x}_j - \mathbf{c}_i)^T \sqrt[N]{det(\mathbf{F}_i)} \mathbf{F}_i^{-1} (\mathbf{x}_j - \mathbf{c}_i). \tag{10}$$
4. Update the partition matrix entries $u_{ij}$ $(i = 1, 2, \ldots, c$ and $j = 1, 2, \ldots, p)$ according to the rule
$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}}\right)^{2/(m-1)}}. \tag{11}$$
If $d_{ij} = 0$ for some $i = I$, take $u_{Ij} = 1$ and $u_{ij} = 0$ for $i \neq I$. Iterate until $\| \mathbf{U}^l - \mathbf{U}^{l-1} \| \leq \varepsilon$ for two succeding iterations.

### 5.2. Automatic determination of the number of clusters

The important problem in TSK network is the determination of the number of rules that should be used in the modelling of data. More rules means better representation of data, but at the same time also increase of complexity of the neural network and higher cost of data processing. Although more rules means better reproduction of learning data, too complex neural network may lead to the decrease of the generalization ability and the deterioration of the quality of network operation in the retrieval mode on the data not taking part in learning.

Therefore, the procedures for the automatic determination of the number of rules, specific for the actual distribution of data are required. In our solution

The control of the number of clusters has been solved here by applying the so called validity measures (Gath and Geva, 1989, Babuska and Verbruggen, 1998, Guillaume, 2001). These measures assess different parameters of clusters, on the basis of which the optimal number of clusters can be determined. We have applied four different validity measures in the solution of the problem. The first is the so called *fuzzy hypervolume measure* $V_h$

$$V_h = \sum_{i=1}^{c} \sqrt{det(\mathbf{F}_i)}. \tag{12}$$

A good partition is the one for which the hypervolume assumes a small value.

The second measure is the *average partition density* $D_A$ that is defined as

$$D_A = \frac{1}{c} \sum_{i=1}^{c} \frac{S_i}{\sqrt{det(\mathbf{F}_i)}} \tag{13}$$

where $S_i$ are calculated only for vectors $\mathbf{x}_k$ that lie within a hyperellipsoid, whose radii are the standard deviations of the cluster features, and are defined as

$$S_i = \sum_{k} u_{ik} \tag{14}$$

for such $k$, that $(\mathbf{x}_k - \mathbf{c}_i)^T \mathbf{F}_i^{-1} (\mathbf{x}_k - \mathbf{c}_i) < 1$. A good partition is achieved, when the average partition density $D_A$ assumes large value.

The next measure applied here is the *average within-cluster distance* $D_w$

$$D_w = \frac{1}{c} \sum_{i=1}^{c} \frac{\sum_{k=1}^{p} u_{ik}^m d_{ik}^2}{\sum_{k=1}^{p} u_{ik}^m} \tag{15}$$

A good choice of cluster is once again indicated by high value of $D_w$.

The last validity measure applied here is the *average cluster flatness* $t_A$. It has been defined (Babuska and Verbruggen, 1998) on the basis of the eigenvalues $\lambda$ of the cluster covariance matrices $\mathbf{F}_i$, arranged in a descending order, i.e., $\lambda_{i1} \geq \lambda_{i2} \geq \ldots \geq \lambda_{iN}$. In order to approximate the data by the hyperplane the clusters should be as flat as possible. It means that the flatness index of the cluster, defined as the ratio between the smallest and the largest eigenvalue of the matrix $\mathbf{F}_i$

$$t_i = \frac{\lambda_{in}}{\lambda_{i1}} \tag{16}$$

should take the smallest possible value for each cluster. The average measure of the flatness of clusters, called the average cluster flatness is defined as the mean of all the flatness indices

$$t_A = \frac{1}{c} \sum^{c} t_i. \tag{17}$$

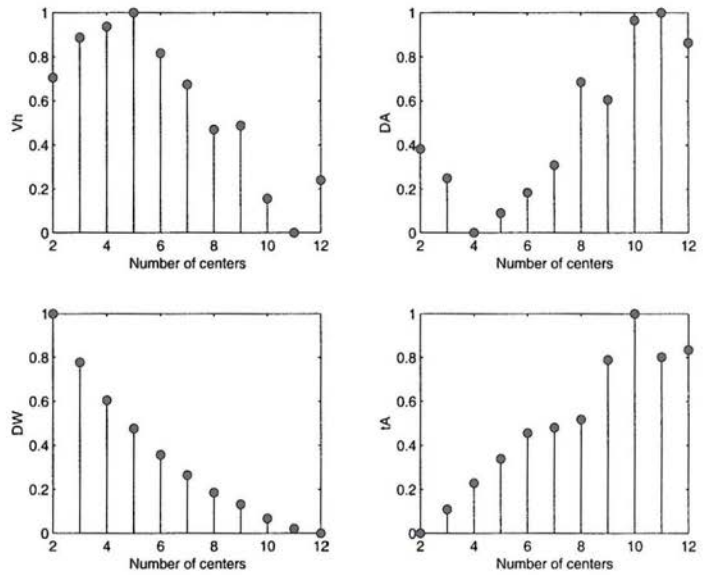


Figure 2. The shapes of four quality measures for the 2-D data distribution of the example

A good partition is indicated by a small value of $t_A$, prefernece being put on a small number of big flat clusters rather than a greater number of small ones.

In our solution we aim at simultaneous satisfaction of all these four quality measures, i.e., maximizing $D_A$ and $D_w$ and minimizing $V_h$ anf $t_A$. To get one quality measure we define one *global quality factor* $q$ in the following way

$$q = a_1 V_h - a_2 D_A - a_3 D_w + a_4 t_A \tag{18}$$

where $a_i$ for $i = 1, 2, 3, 4$ are positive scaling coefficients. The minimum of this measure indicates the optimal number of cluster centers. The procedure of adjusting the optimal number of clusters can be then stated as follows

- Set the maximum possible number $n$ of clusters that are taken into account in the clustering procedure.
- Perform the partition of the data using GK algorithm for number of clusters equal $i = 2, 3, \ldots, n$.
- Calculate the validity measures for all the acceptable possible numbers of clusters.
- The optimal number of clusters is the number $c$ ($c \leq n$) that corresponds to the deepest minimum of the global quality factor $q$. Usually there

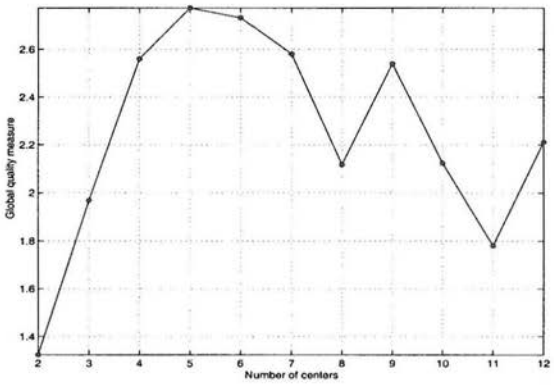


Figure 3. The global quality measure $q$ obtained for the 2-D data of the example

choose the one of the smallest value. The final criterion of selection is the quantization error $E_q$, that can be accepted in practical implementation of the algorithm.

We will illustrate this procedure on the example of 2D data distribution presented in Fig. 4 (the dots). The clusterization of the data has been performed

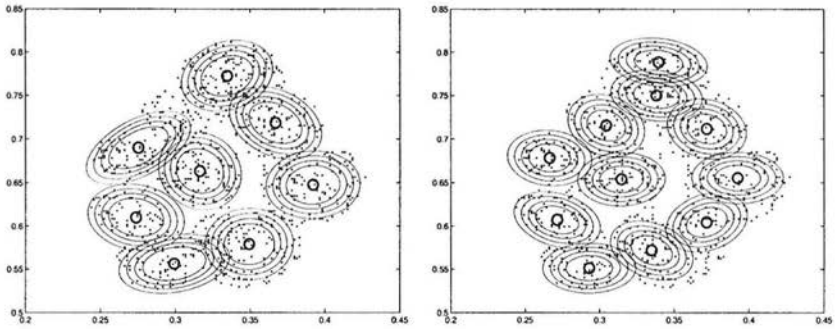

Figure 4. The distribution of data of the example and locations of the cluster centers with the equal membership lines for 8 centers (left) and 11 centers (right)

for the number of clusters changing from 2 to 12. Fig. 2 illustrates the change of $V_h$, $D_A$, $D_w$ and $t_A$ versus the number of clusters. The global quality factor $q$ was calculated with the scaling coefficients chosen in a way to get uniform influence of all factors. Fig. 3 presents its change versus the number of centers.

ond to 11 centers. Both solutions are acceptable, leading however to different quantization errors $E_q$. Fig. 4 presents the obtained locations of centers for both cases on the background of the data, represented by the dots (the same in both cases). The small circles point to the positions of centers obtained by GK algorithms. The ellipsoidal lines indicate the positions of data of equal membership degrees to the appropriate cluster (from 0.74 to 0.92). At 8 clusters the obtained quantization error $E_q$ was equal $E_q = 3.76 \times 10^{-4}$. At 11 clusters this error has been reduced to $E_q = 2.48 \times 10^{-4}$.

The clustering procedure described above has been applied for generating the inference rules for fuzzy TSK system. Each cluster center has been associated with the center of membership functions, forming the appropriate inference rule and the number of clusters indicates the number of applied rules. Note that cluster center is the $N$-dimensional vector, and each component of this vector points to the center in one dimension.

## 6. Results of numerical experiments

Different approximation tasks have been solved using the approach presented above and the results have confirmed good performance of it. The first task reported here is to approximate the static function of two variables given in the form

$$y(x_1, x_2) = \frac{sin(x_1) \cdot sin(x_2)}{x_1 \cdot x_2} \tag{19}$$

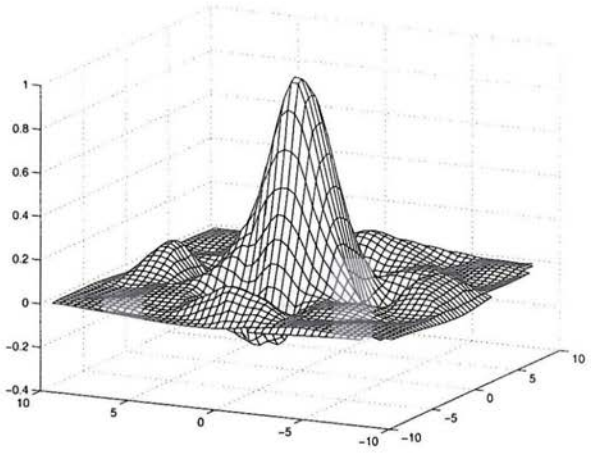


Figure 5. The destination function for approximation

in the input variable range $[-10, 10]$ for both $x_1$ and $x_2$. The shape of this function within the mentioned domain is shown in Fig. 5.

The first task to do was the determination of the optimal number of inference rules. We have solved it using the self-organization and the procedure based on the quality measures, described in the previous section. Fig. 6 presents the curve of global quality measure versus the number of clusters. There are three minima, corresponding to three possible suboptimal solutions (indicated by dots). For further study we have chosen the smallest possible minimum of the quality measure, corresponding to 12 clusters. This number of clusters means $M = 12$ inference rules for TSK network. The membership functions appearing in each rule are independent to each other, so the number of adapted parameters for 2-input network is equal 72. The number of linear parameters of TSK functions is equal 36. That makes together 108 parameters of the system.

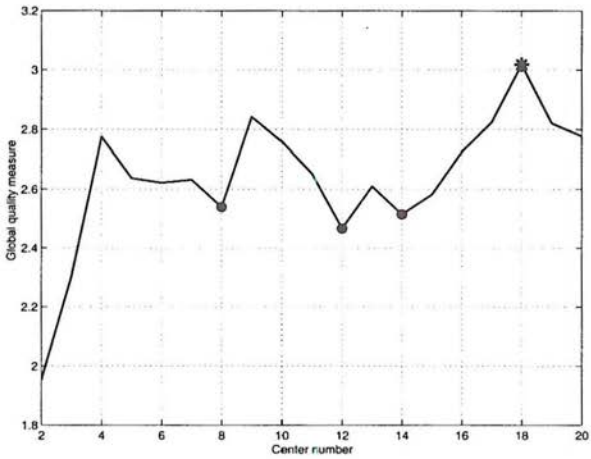


Figure 6. The global quality measure versus the number of clusters for the first example

The neuro-fuzzy network designed in this way has been tuned using the hybrid algorithm. Fig. 7 presents the changes of the parameters (centers, widths and exponent coefficients) of 6 randomly chosen membership functions in the adaptation process. As it can be seen the centers have moved only slightly from their initial positions obtained in the self-organization process. It means that the self-organization fulfills not only the role of determination of the number of rules but at the same time it is a quite precise pretuning. All other parameters not pretuned in the initial stage: the widths and the exponent coefficients have registered much higher changes specific to the problem. The hybrid learning process has reduced the error function[1] from the initial value of $E = 2.7856$ to

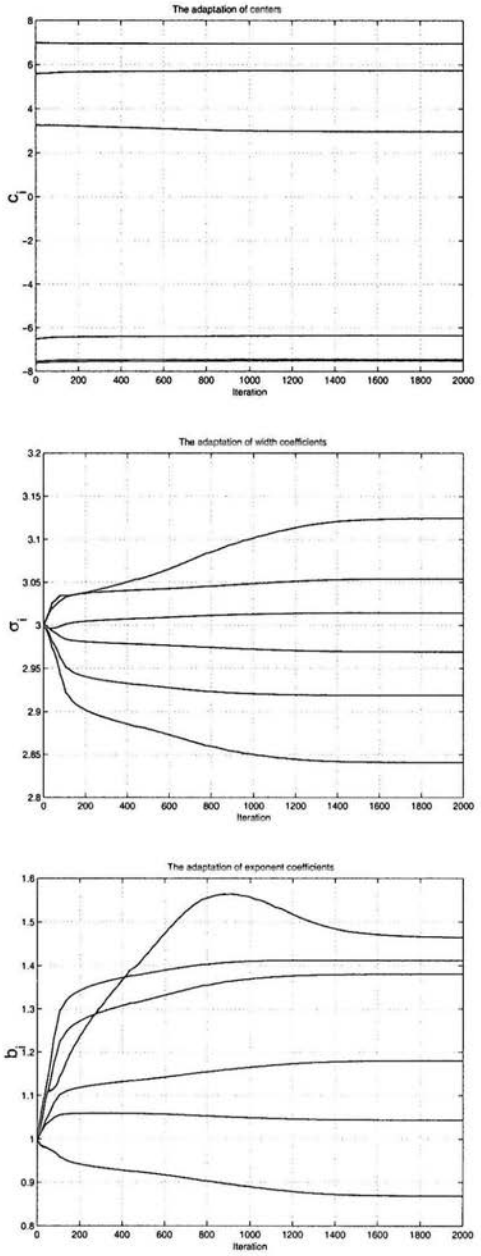


Figure 7. The adaptation of the parameters of six chosen membership functions by the hybrid algorithm: a) the centers, b) the widths, c) the exponent coefficients

the final value of $E = 0.5748$. At 200 learning samples it corresponds to the average of $E_s = 0.0029$ per sample.

After the learning stage, all parameters of the network have been frozen and the adapted network tested on the data, not having taken part in learning. Fig. 8 presents the results of testing. Fig. 8a depicts the actual approximated shape of the function and Fig. 8b – the error surface, that is the difference between the actual approximated values of $y$ and the destination surface. The total error value for 200 testing data points attained the value of $E = 0.7112$, i.e., only slightly more than for the learning data.

To check the quality of solution we have tried other numbers of rules: 8 (another local minimum of the quality function) and 18 (the point corresponding to the maximum of the quality function). Table 1 presents the comparison of the results of learning and testing TSK network.

Table 1 Comparison of results of learning and testing for the different numbers of rules

| No of rules | Learning error | Testing error |
|---|---|---|
| 8 | 1.1249 | 1.3107 |
| 12 | 0.5748 | 0.7120 |
| 18 | 0.3569 | 0.9203 |

As it can be seen, increasing the number of rules decreases the learning error, but at the same time may increase the testing error (overlearning). Reducing the number of rules below the optimal one increases both: the learning and the testing errors. The table depicts the fact that 12 rules found in the self-organization procedure lead to the best results in the testing mode on the data not taking part in learning (the best generalization).

The network solution described here is also well suited for the on-line modelling of the dynamic, nonparametric processes. Our next example is concerned with the on-line learning of the TSK network, modelling the dynamic process described by the following equation (Narendra and Parthasarathy, 1990):

$$y(k+1) = 0.3y(k) + 0.6y(k-1) + 0.6sin(\pi u) + 0.3sin(0.3\pi u) + 0.1sin(5\pi u) \tag{20}$$

with $u = sin(2\pi k/250)$. To find out the optimal number of inference rules we have generated the limited number (i.e., 1000) of data governed by this equation and performed the clusterization, according to the procedure described in the previous sections. The quality measure test has indicated that the best number of clusters was equal 4, so four rules have been used in further experiments.

The next step was the on-line adaptation of the parameters of the TSK network, following the changes of the function. To make the problem more demanding, we changed twice the frequency of excitation. In this experiment the neuro-fuzzy network works in the adaptive mode, tuning all nonlinear pa-

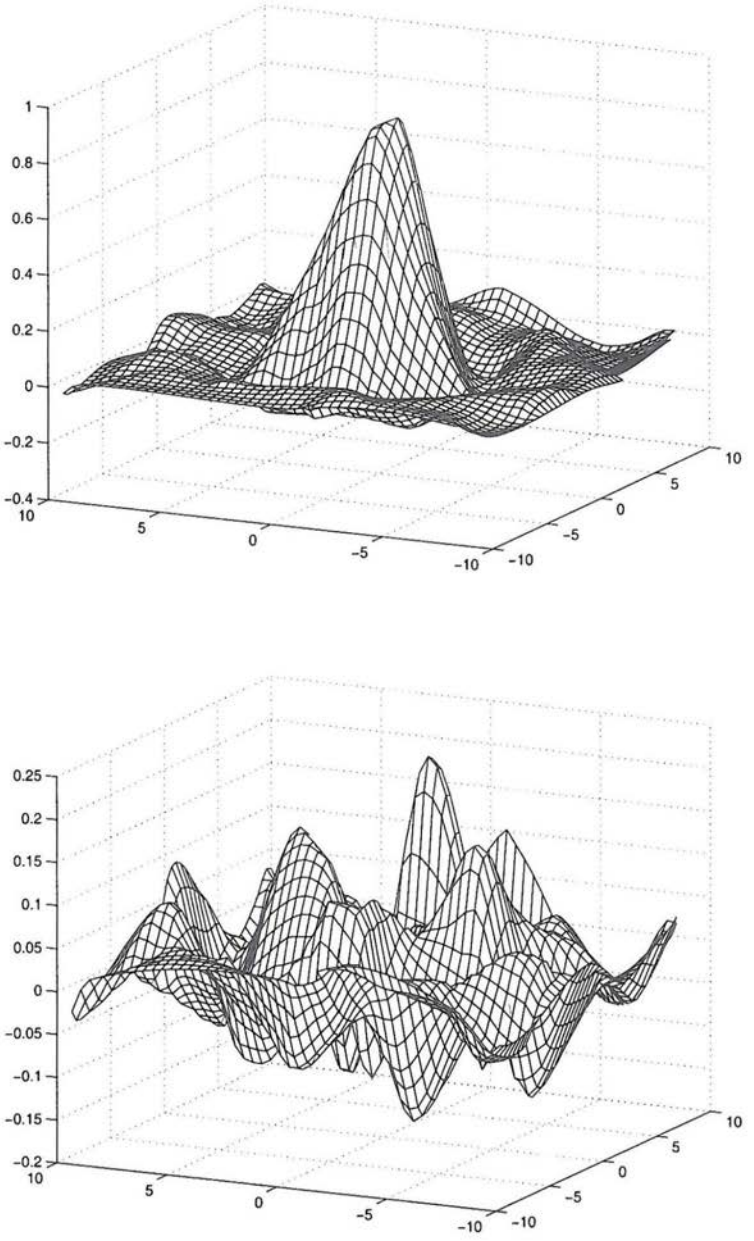


Figure 8. The results of approximation: a) the output 3D function, b) error of approximation at 12 inference rules

the linear parameters by using SVD, with the starting values obtained from the initialization. This time the SVD procedure has been performed on the matrix formed by the last 150 samples, replacing each time the oldest sample by the one just acquired.

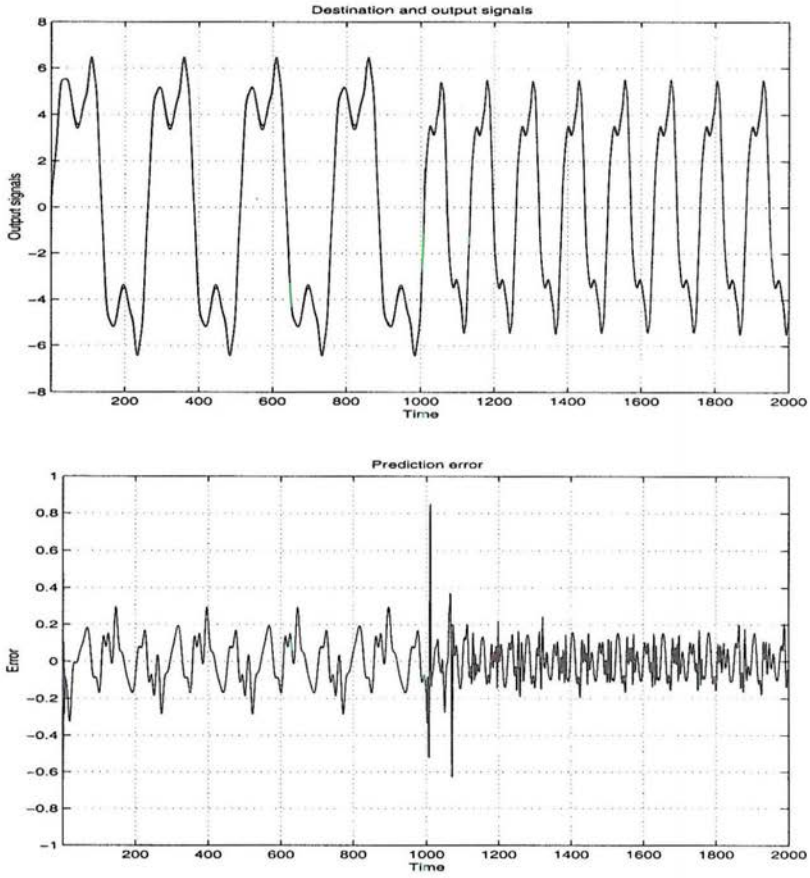


Figure 9. The results of modelling of the dynamic function

The results of experiment are depicted in Fig. 9. The upper figure presents the destination and the actual output signals, while the lower one – the adaptation error. The samples up to 1000 have been reproduced on the basis of off-line (batch) learning. The on-line learning has started from the sample number 1001. The change of frequency of excitation as well as the change of working mode has resulted in some distortions of adaptation and higher error of reproduction

imatelly 100 samples, the network has stabilized its operations and the error stayed on the lower level, slightly smaller than in the batch mode (the first 1000 samples).

This results have proved that the TSK network performs well in the on-line adaptation mode for the dynamic system modelling and may find practical application as the tracing system.

## 7. Conclusions

The paper has presented the neuro-fuzzy approach to the approximation of the static and dynamic processes. The multilayer network structure has been constructed and the parameter adaptation procedure, based on the hybrid approach, has been presented. The self-organization of the data has been proposed for establishing the optimal number of inference rules and pretuning the centers of the membership functions.

The important feature of this approach is an easy and flexible way of generating the inference rules. The self-organizing process combined with the quality measure is able to provide the optimal number of rules, leading to the best performance of the network in the retrieval mode. In contrast to the classical neural network modelling the neuro-fuzzy approach offers very efficient and fast adaptation. The examples presented in the paper have proved its efficiency in modelling both the static and the dynamic processes. The system based on neuro-fuzzy network is universal and relatively easy in practical implementation.

## References

BABUSKA, R. and VERBRUGGEN H.B. (1998) Constructing fuzzy models by product space clustering. In: H. Hellendoorn, D. Driankov, eds., *Fuzzy Model Identification.* Springer, Berlin, 53–90.

DUCH, W, KORBICZ, J., RUTKOWSKI, L. and TADEUSIEWICZ, R., eds. (2000) *Sieci neuronowe.* Akademicka Oficyna Wydawnicza, Warszawa.

GATH, I. and GEVA, A. (1989) Unsupervised optimal fuzzy clustering. *IEEE Trans. PAMI*, **7**, 773–781.

GUILLAUME, S. (2001) Designing fuzzy inference system from data: an interpretability-oriented review. *IEEE Trans. Fuzzy Syst.*, **9**, 426–443.

GUSTAFSON, D. and KESSEL, W. (1979) Fuzzy clustering with a fuzzy covariance matrix. *Proc. IEEE CDC*, San Diego, 761–766.

ISHIBUCHI, H. (1997) Development of fuzzy neural networks. In: W. Pedrycz, ed., *Fuzzy Modelling - Paradigm and Practice*, Dordrecht, 185–202.

JANG, J.R., SUN, C.T. and MIZUTANI E. (1997) *Neuro-fuzzy and soft computing.* Prentice Hall.

LIN, Y. and CUNNINGHAM III, G. (1995) A new approach to fuzzy neural system

MITRA, S. and HAYASHI, Y. (2000) Neuro-fuzzy rule generation: survey in soft computing framework. *IEEE Trans. Neural Networks*, **11**, 748–768.

NARENDRA, K.S. and PARTHASARATHY, K. (1990) Identification and control of dynamical systems using neural networks. *IEEE Trans. Neural Networks*, **1**, 4–27.

OSOWSKI S. (2000) *Sieci neuronowe do przetwarzania informacji.* Oficyna Wydawnicza, Warszawa.

PAL, N.R. and BEZDEK, J.C. (1995) On cluster validity for the fuzzy c-means model. *IEEE Trans. Fuzzy Systems*, **3**, 370–379.

TAKAGI, T. and SUGENO, M. (1985) Fuzzy identification of systems and its application to modelling and control. *IEEE Trans. SMC*, 116–132.

WANG, L.- X. (1994) *Adaptive Fuzzy Systems and Control. Design and stability analysis.* Prentice Hall, N. J.

YAGER, R. and FILEV, D. (1995) *Podstawy modelowania i sterowania rozmytego.* WNT, Warszawa.

ZIMMERMANN, H.J. (1985) *Fuzzy Set Theory and Its Applications.* Kluwer, Boston.