TSK Fuzzy Modeling with Nonlinear Consequences

Jacek Kabziński, Jaroslaw Kacerka

TSK Fuzzy Modeling with Nonlinear Consequences

Jacek Kabziński, Jarosław Kacerka

Institute of Automatic Control, Lodz University of Technology
{jacek.kabzinski , jaroslaw.kacerka}@p.lodz.pl

Abstract. We propose to generalize TSK fuzzy model applying nonlinear functions in the rule consequences. We provide the model description and parameterization and discus the problem of model training and we recommend PSO for tuning parameters in membership functions and in nonlinear part of a rule consequence. We also propose some more or less formalized approach to nonlinear consequence selection and construction. Several examples demonstrate the main features of the proposed fuzzy models. The proposed approach reduces the average obtained model Root Mean Square Error (RMSE) with regard to the linear fuzzy model, as well that it allows to reduce the model complexity preserving the desired accuracy.

Keywords: fuzzy modeling, TSK fuzzy model, fuzzy model training.

1 Introduction

Among various fuzzy modeling techniques the approach proposed by Takagi, Sugeno and Kang [1, 2], so called TSK model, remains one of the most popular and effective. The standard TSK model employs affine functions in consequences of fuzzy rules. So the idea of modeling is based on local linear models dominating if the rule activation strength is high. The model possesses the universal approximation property [3] but in many practical problems the number of rules to achieve the desired accuracy is really high. Motivated by many experiences from modeling mechatronic systems (see example 2 below) we claim that it is possible to propose local nonlinear models in many practical applications. Having some knowledge on the nature of the modeled phenomenon we may foresee nonlinear functions that should appear in the model, even if we are not sure about the exact parameters of these functions.

In this contribution we propose to generalize TSK model by accepting nonlinear functions of model inputs in fuzzy rule consequences. We propose a formal model description and parameterization. Next we consider the problem of model training from the numerical data, which is more difficult than in the standard case. We also propose some formal methods to develop nonlinear consequences. The aim of the proposed modification is to reduce the model complexity preserving the desired accuracy, and we demonstrate on several examples that that it really happens.

The idea of using nonlinear functions in fuzzy rule consequences was already mentioned in literature, but only particular cases were considered. In [4] switched Takagi-Sugeno models with an affine nonlinear consequent part are used to control switched

nonlinear dynamical systems and in [5] it is proved that under some special assumptions such fuzzy model can approximate a particular class of nonlinear functions, nonlinear dynamic systems and nonlinear control systems.

2 Fuzzy TSK System with Nonlinear Consequences

We consider a fuzzy, single output, multi input model given by the rules

$$R_i: IF\ (x_1 is\ \mu_{i1}) and\ \ldots\ and\ (x_n is\ \mu_{in})\ THEN\ y\ is\ y_i = f_i(x_1, \ldots, x_n), \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)^T$ is the vector of inputs, $i = 1, \ldots, R$ is the rule number, μ_{ij} are membership functions, each defined by three parameters $\bar{\mu}_{ij}, \bar{\bar{\mu}}_{ij}, \breve{\mu}_{ij}$, and y_i are consequences

$$f_i(x) = f_i(x_1, x_2, \ldots, x_n) = a_{i,1}\varphi_{i,1}(x, p_{i,1}) + a_{i,2}\varphi_{i,2}(x, p_{i,2}) + \cdots + a_{i,m_i}\varphi_{i,m_i}(x, p_{i,m_i}). \tag{2}$$

Parameters of the consequences are organized as follows: $p_{i,j}$ is a $s_{i,j}$ –dimensional vector parameterizing nonlinear function $\varphi_{i,j}$ while $a_{i,j}$ are scalars. For each rule we define its activation level as

$$\alpha_i(x) = \prod_{j=1}^{n} \mu_{ij}(x_j) \tag{3}$$

and denote

$$\sigma(x) = \sum_{i=1}^{R} \alpha_i(x) \tag{4}$$

The model output is calculated from

$$y(x) = \frac{1}{\sum_{i=1}^{R} \alpha_i(x)} \sum_{i=1}^{R} [\alpha_i(x) f_i(x)]. \tag{5}$$

If we represent the vector of linear combination parameters as

$$a = \left[a_{1,1}, \ldots, a_{1,m_1}, a_{2,1}, \ldots, a_{2,m_2}, \ldots, a_{R,1}, \ldots, a_{R,m_R}\right]^T \tag{6}$$

or

$$\bar{a}_i = \left[a_{i,1}, \ldots, a_{i,m_i}\right]^T,\ a = \begin{bmatrix} \bar{a}_1 \\ \vdots \\ \bar{a}_R \end{bmatrix} \tag{7}$$

and we refer to the vector of activated consequences:

$$v_i(x) = \alpha_i(x)\left[\varphi_{i,1}(x, p_{i,1}), \ldots, \varphi_{i,m_i}(x, p_{i,m_i})\right], v(x) = [v_1(x), \ldots, v_R(x)], \tag{8}$$

we get a short description of the system output:

$$y(x) = \frac{1}{\sigma(x)} v(x) a, \tag{9}$$

which stresses that the model is linearly parameterized by a. This important feature will be utilized during model training and makes such model attractive in adaptive control applications.
If the same set of functions appears in each of the consequences, i.e.

$$m_1 = m_2 = \cdots = m_R = m, \varphi_{i,j}(x, p_{i,j}) = \varphi_j(x, p_j), j = 1,2, \dots, m \tag{10}$$

we may describe the model output as

$$y(x) = \frac{1}{\sigma(x)}[\varphi_1(x, p_1), \quad \dots \quad , \varphi_m(x, p_m)][\bar{a}_1, \quad \dots \quad , \bar{a}_m]\begin{bmatrix}\alpha_1(x)\\ \vdots \\ \alpha_R(x)\end{bmatrix} \tag{11}$$

and after accepting notation

$$\begin{gathered} z^T = [\varphi_1(x, p_1) \quad \varphi_2(x, p_2) \quad \cdots \quad \varphi_m(x, p_m)], \hat{A} = \begin{bmatrix} a_{1,1} & \cdots & a_{R,1} \\ \vdots & \ddots & \vdots \\ a_{1,m} & \cdots & a_{R,m} \end{bmatrix} \\ \xi = \begin{bmatrix}\alpha_1(x)\\ \vdots \\ \alpha_R(x)\end{bmatrix}\frac{1}{\sigma(x)} \end{gathered} \tag{12}$$

we have another short output description

$$y(x) = z^T \hat{A} \xi. \tag{13}$$

Similarly to the classical TSK model with linear consequences our model may be represented as a multi-layer neural network. In the first layer actual values of the membership functions are calculated, in the second the activation level for the each rule is calculated, in the third layer normalized activation strength $\bar{\alpha}_i(x) = \frac{\alpha_i(x)}{\sigma(x)}$ of each rule is computed, in the fourth products $\bar{\alpha}_i f_i$ are obtained and the final node performs the summation $y = \sum_i \bar{\alpha}_i f_i$.

3 Model Training

The task to construct a TSK fuzzy model can be divided into two steps: first we have to design the model structure – recognize relevant inputs, decide about membership functions, plan the rules and the consequences etc. While the structure is established we optimize the system parameters based on some numerical data representing the desired behavior of the model – this stage of modeling is usually called model training or tuning.
In case of the proposed model with nonlinear consequences the model structure selection may be performed using any standard approach as placing membership functions on a grid, by clustering, or others [6]. The problem of nonlinear consequences selection is discussed separately. Here we concentrate on the model training methods.

The proposed model parameters may be divided into three sets: parameters of membership functions (parameters μ), parameters that appear nonlinearly in the consequences (parameters p) and parameters that are coefficients of linear combination in consequences (parameters a). The training data is collected in a M-dimensional target output vector $\bar{Y}$ and each entry $\bar{y}^i$ corresponds to an input vector x^i while the model output for the same input is y^i. The aim of training is to find model parameters minimizing the RMSE

$$J = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\bar{y}^i - y^i)^2} \to min. \tag{14}$$

Calculation of optimal parameters a, for given parameters μ and p, is trivial since according to (9) a is the solution in the least squares sense to system of equations

$$\bar{Y} = \Phi a, \quad \Phi = \begin{bmatrix} \frac{1}{\sigma(x^1)} v(x^1) \\ \vdots \\ \frac{1}{\sigma(x^M)} v(x^{M1}) \end{bmatrix}, \quad a = \Phi \backslash \bar{Y}. \tag{15}$$

Unfortunately the target function J in (14) is heavily nonlinear function of parameters μ and p. It may be nonsmooth and possess many local minima. So we propose to apply evolutionary optimization to find optimal model parameters. We have experimented with many evolutional optimization algorithms and finally we recommend PSO as well suitable for the problem. We use a well-known version as described in [8]. Parameters [μ , p] are coded as particles X. Each calculation of the fitting function for the given [μ , p] is done as follows: calculate Φ and a from (15), next calculate J from (14).

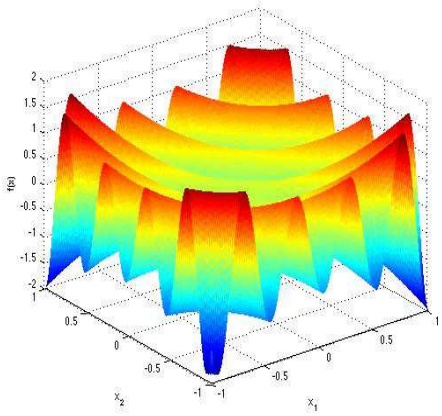

Fig. 1. Example function $f_1(x_1,x_2)$

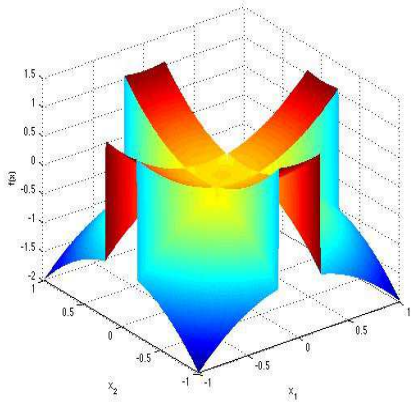

Fig. 2. Example function $f_2(x_1,x_2)$

4 Examples

The proposed model training approach was tested on several modeling problems.
Example 1. Let us consider the following nonlinear functions defined for $x_{1,}x_2 \in [-1,1]$ and plotted in fig. 1 and 2

$$f_1(x) = (-x_1^2 - x_2^2) \cdot \cos\big(13 \cdot (x_1 + x_2)\big), \tag{16}$$

$$f_2(x) = (-x_1^2 - x_2^2) \cdot sign(|x_1| - 0.2) \cdot sign(|x_2| - 0.2) \tag{17}$$

and assume that we know the general form of these functions as

$$f_1(x) = \alpha(x_1, x_2) \cdot \cos(p \cdot (x_1 + x_2)), \tag{18}$$

$$f_2(x) = \alpha(x_1, x_2) \cdot sign(|x_1| - p_1) \cdot sign(|x_2| - p_2). \tag{19}$$

This assumption motivates the following choice of consequences in rule number *i*

$$\mathrm{y}_i = a_{i0} + a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + a_{i3} \cdot \cos(p \cdot (x_1 + x_2)) \tag{20}$$

$$\mathrm{y}_i = a_{i0} + a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + a_{i3} \cdot sign(|x_1| - p_1) \cdot sign(|x_2| - p_2) \tag{21}$$

The fuzzy model structure was designed by selecting membership functions

$$\mu_{ij}(x_j) = \frac{1}{1+\left(\frac{x_j - c_{ij}}{a_{ij}}\right)^{2b_{ij}}} \tag{22}$$

initialized on an uniform grid. We have tested the performance of modeling by: standard TSK model with linear consequences trained by ANFIS (adaptive-network-based fuzzy inference system [9], the widely accepted standard for TSK systems training), TSK model with nonlinear consequences tuned by genetic algorithm (GA implemented as in Matlab Global Optimization Toolbox), TSK model with nonlinear consequences tuned by PSO. Results of modeling of function (14) in terms of average obtained model RMSE are presented in table 1. It may be observed, that modeling error for nonlinear models is lower regardless of model complexity, i.e. number of rules. PSO tuning results in a much more precise model – error is lower by 1 to 3 orders of magnitude than for GA tuned model.

Table 1. RMSE of linear model tuned by ANFIS and nonlinear models tuned by GA and PSO. Presented RMSE of PSO and GA tuned models is an average of 10 algorithm executions.

Rules	ANFIS	GA	PSO
4	0.5643	0.5369	0.0271
9	0.5611	0.5149	0.0017
16	0.5281	0.4962	5.5e-4
25	0.4845	0.1165	2.0e-5
36	0.4071	0.0523	3.7e-5
49	0.3266	0.0345	6.3e-5

Additional experiments demenstrated that both GA and PSO benefit from estimating the value of nonlinear p parameters prior to optimizations. Such estimation was performed by minimizing the value of the same RMSE objective with constant parameters of membership functions, i.e. membership functions were initialized as a uniform grid and remained unchanged – only p parameters were tuned by GA or PSO for a low number of iterations. During the main optimization the search space for p was limited to the neighborhood of the estimated value. Such two stage optimization results in further model improvement, e.g. for a 4 rule model the average RMSE is 0.029 for GA tuning and 0.006 for PSO.

Example 2. The data presented in fig. 3 describes the q-axis flux component of a permanent magnet synchronous motor as function of current and rotor position. Acceptable fuzzy model with affine consequences requires at least N=11 rules. It is visible that for currents bigger than 1A we observe flux oscillations with rotor position. The frequency of these oscillations is 6 times bigger than the rotor angular speed. Hence we propose nonlinear consequences

$$y_i = a_{i1}x_1 + a_{i2}x_2 + a_{i0} + a_{i3}sin(6x_1) + a_{i4}cos(6x_1), i = 1 \dots N \qquad (23)$$

where x_1 is a rotor position and x_2 is q-axis current. With this it was possible to reduce the number of rules and the model with 4 rules only was equivalent to 11 rules model with linear consequences, as it is demonstrated in table 2. Presented error is calculated as RMSE for the whole set of collected data (80 000 points), while model training was based on points equal to local average of measurements over a grid of 900 points.

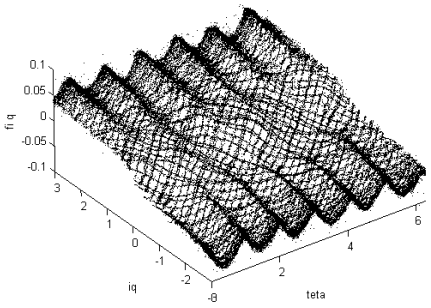

Fig. 3. The q-axis motor flux (Ψ_q [Wb]) as function of motor current (i_q [A]) and rotor position (θ [rad]). The data collected from the flux observer.

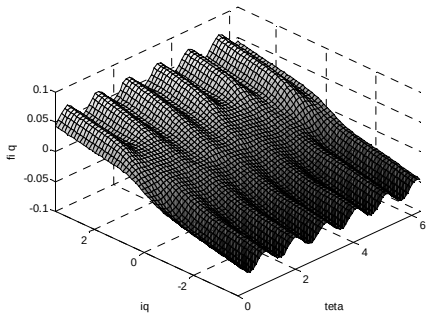

Fig. 4. Fuzzy model output corresponding to the data from fig. 3. The model with 4 rules and consequences (23) trained by PSO.

Table 2. Comparison of fuzzy models. The training was based on 900 local average values and test error was calculated using all 80 000 points from fig. 3.

	11 rules with affine consequences trained by ANFIS	4 rules with nonlinear consequences (23) trained by PSO
Training error	0.0056	0.0060
Test error	0.0079	0.0070

5 Nonlinear consequences construction

The obvious problem we have to solve constructing a TSK model with nonlinear consequences is how to select and parameterize nonlinear functions that appear in the consequences. Usually, in practical problems, the expert knowledge is sufficient to propose the consequences, as it was demonstrated in example 2. We may also propose a more formal approach.

Additive correction approach

If we may assume (from an expert knowledge for example) what kind of nonlinearities may be present in the system model we add these nonlinearities to the consequences. Finally we get the consequence for rule number *i*

$$y_i = a_{i0} + a_{i1}x_1 + \cdots + a_{in}x_n + a_i g(x,p). \tag{24}$$

The first part of (24) is a standard affine function as in classical TSK model and the last one is selected to cope with expected nonlinearities. This approach is especially useful if we expect discontinuities in the data – for example friction is discontinuous at zero velocity. Consequents similar to (24) were applied in all examples presented above and prove to be effective.

Constant coefficient approach

Let us consider a function $f: R^n \to R$ to be modelled and assume that it may be represented on a compact set *A* as

$$f(x) = \alpha(x) g(x,p). \tag{25}$$

Assume that $g(x,p)$ is known, although the actual values of constant parameters p may be unknown. Assume that *A* is covered by a grid defined by selection of points for each axis:

$$x_{k,0} < x_{k,1} < \cdots < x_{k,n_k}, k = 1, \dots, n. \tag{26}$$

We denote the middle points of the grid by

$$\bar{x}_{k,j} = \frac{x_{k,j} - x_{k,j-1}}{2}, \quad j = 1, \dots, n_k, \quad k = 1, \dots, n \tag{27}$$

and define the rules

$$R_{j_1,\dots,j_n}: IF\ \left(x_1\ is\ near\ \bar{x}_{1,j_1}\right)\ and\ \dots and\ \left(x_n\ is\ near\ \bar{x}_{n,j_n}\right)\ THEN\ y\ is$$
$$y_{j_1,\dots,j_n} = a_{j_1,\dots,j_n} g(x,p). \tag{28}$$

Of course the initial guess for $a_{j_1,\dots,j_n}$ is $\left(\bar{x}_{j_1,\dots,j_n}\right)$, $\bar{x}_{j_1,\dots,j_n} = (\bar{x}_{1,j_1},\dots,\bar{x}_{n,j_n})$, or the average value of the data representing $\alpha(x)$ in the part of the grid around $\bar{x}_{j_1,\dots,j_n}$, if such information is available.

Linear approximation approach

Assuming that $\alpha(x)$ is sufficiently smooth, it may be approximated by

$$\begin{aligned}\alpha(x) &\approx \alpha\left(\bar{x}_{j_1,\dots,j_n}\right) + \sum_{i=1}^{n} \left.\frac{\partial \alpha}{\partial x_i}\right|_{\bar{x}_{i,j_i}} (x_i - \bar{x}_{i,j_i}) \\ &= \alpha\left(\bar{x}_{j_1,\dots,j_n}\right) - \sum_{i=1}^{n} \left.\frac{\partial \alpha}{\partial x_i}\right|_{\bar{x}_{i,j_i}} \bar{x}_{i,j_i} + \sum_{i=1}^{n} \left.\frac{\partial \alpha}{\partial x_i}\right|_{\bar{x}_{i,j_i}} x_i\end{aligned} \tag{29}$$

what motivates the rules

$$R_{j_1,\dots,j_n}: IF\ \left(x_1\ is\ near\ \bar{x}_{1,j_1}\right)\ and\ \dots and\ \left(x_n\ is\ near\ \bar{x}_{n,j_n}\right)\ THEN\ y\ is$$
$$y_{j_1,\dots,j_n} = a_{0,j_1,\dots,j_n} + a_{1,j_1,\dots,j_n} x_1 g(x,p) + \dots + a_{n,j_1,\dots,j_n} x_n g(x,p) \tag{30}$$

Example 3. Proposed methods of constructing rule consequences were applied to modeling of example function (16). The tested model consisted of 4 or 9 rules with consequences defined as:

- standard affine:

$$y_i = a_{i0} + a_{i1} x_1 + a_{i2} x_2 \tag{31}$$

- nonlinear - consisting of linear part and added nonlinear function:

$$y_i = a_{i0} + a_{i1} x_1 + a_{i2} x_2 + a_{i3} cos\left(p \cdot (x_1 + x_2)\right) \tag{32}$$

- nonlinear with constant coefficients:

$$y_i = a_i cos\left(p \cdot (x_1 + x_2)\right) \tag{33}$$

- nonlinear constructed according to the linear approximation approach:

$$y_i = a_{i0} + a_{i1} x_1 cos\left(p \cdot (x_1 + x_2)\right) + a_{i2} x_2 cos\left(p \cdot (x_1 + x_2)\right) \tag{34}$$

Model with linear consequences was trained by ANFIS, while nonlinear models were trained by PSO. The objective of training was to minimize the RMSE model error. Nonlinear models - gained much better accuracy than the linear model as is demon-

strated in table 3 and shown in figures 5 and 6 presenting modeling error of a sample linear and nonlinear model.
To illustrate the complexity of linear model required to match the accuracy of nonlinear models we continue constructing of linear models for as many as 144 rules trained by ANFIS. The resulting model's RMSE is similar to error of a 4-rule nonlinear model.

Table 3. RMSE of linear (ANFIS) and nonlinear models: PSO additive (eq. (32)), PSO constant (33), PSO lin.apx. (34) for 4 and 9 rules. Higher number of rules modeled for comparison for linear model only. Presented RMSE of PSO tuned models is an average of 10 algorithm executions.

Rules	ANFIS	PSO additive	PSO constant	PSO lin.apx.
4	0.5643	0.0271	0.0461	0.0024
9	0.5611	0.0017	0.0014	0.0003
81	0.1921	-	-	
100	0.0693	-	-	-
144	0.0254	-	-	-

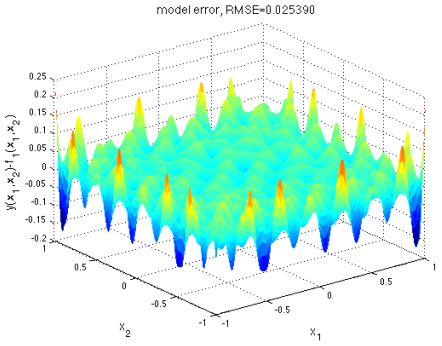

Fig. 5. Modeling error of function (16) for a 144-rule ANFIS trained model with linear consequents.

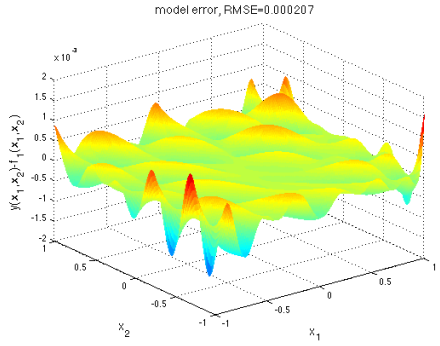

Fig. 6. Modeling error of function (16) for a 9-rule PSO trained model with nonlinear consequents given by eq. (34)

6 Conclusions

We propose to generalize TSK fuzzy model applying nonlinear functions in the rule consequences. We provide the model description and parameterization and discus the problem of model training and we recommend PSO for tuning parameters in membership functions and in nonlinear part of a rule consequence. We also propose some more or less formalized approach to nonlinear consequence selection and construction. Several examples demonstrate the main features of the proposed fuzzy models.

The proposed approach allows to reduce the model complexity preserving the desired accuracy. A model with smaller number of rules is simpler for real time application and easier interpretable. Nonlinear consequences enable modeling of function imperfectly approximated by standard TSK models, for example discontinuous functions [10]. As the constructed models are linear in some of parameters they may be efficiently utilized in adaptive control of dynamical systems, as it was proposed for example in [11]. It is also important that the proposed model may be described as a neural network and so NN-training techniques may be utilized for the model tuning.

7 References

1. Takagi, T., Sugeno, M.: Fuzzy Identification on Systems and its Applications to Modeling and Control. IEEE Trans. on Systems, Man, and Cybern., vol.15, pp.116-132 (1985)
2. Sugeno, M., Kang., G.: Structure Identification of Fuzzy Model. Fuzzy Sets and Systems, vol.28, 1986, pp.329-346 (1998)
3. Nguyen, H.T., Sugeno, M.: Fuzzy Systems: Modeling and Control, Springer (1998)
4. Chen, S-S., Chang, Y-C., Wu, J-L., Cheng, W-C., Su S-F.: Fuzzy Control Design for Switched Nonlinear Systems, Proc. of SICE Annual Conference, pp. 352 – 357 (2008)
5. Rajeshm, R., Kaimal, M.R.: T–S fuzzy model with nonlinear consequence and PDC controller for a class of nonlinear control systems, Applied Soft Computing 7, pp. 772–782, (2007)
6. Kabziński, J.: One-Dimensional Linear Local Prototypes for Effective Selection of Neuro-Fuzzy Sugeno Model Initial Structure, IFIP WG 12.5 International Conference "Artificial Intelligence Applications and Innovations". Larnaca, Cyprus Springer, IFIP Series Berlin, pp. 62 - 69 (2010)
7. Parsopoulos, K.E., Vrahatis, M.N.: UPSO: A Unified Particle Swarm Optimization Scheme, Lecture Series on Computer and Computational Sciences, Vol. 1, Proceedings of the International Conference of "Computational Methods in Sciences and Engineering" (ICCMSE 2004), pp. 868-873, VSP International Science Publishers, Zeist, The Netherlands (2004)
8. Shi, Y., Eberhart, R.: A modified particle swarm optimizer, Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on , pp.69,73, 4-9 May (1998)
9. Jang J.R.: ANFIS: Adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cybern. 23, pp. 665 – 684 (1993)
10. Jastrzebski M., A generalized Takagi-Sugeno-Kang model designed for approximation of discontinuous dependeces, Zeszyty Naukowe. Elektryka - Politechnika Łódzka, pp. 15-24, (2005)
11. Kabzinski,J.: Fuzzy Friction Modeling for Adaptive Control of Mechatronic Systems. Artificial Intelligence Applications and Innovations. IFIP Advances in Information and Communication Technology, vol. 381, pp.185-195, Springer, Berlin Heidelberg (2012)