Grzegorz DUDEK

Czestochowa University of Technology

Neuro-fuzzy approach to the next day load curve forecasting

***Abstract**. An adaptive neuro-fuzzy inference system ANFIS is used to the short-term load forecasting. ANFIS combines the comprehensibility of fuzzy rules and the adaptability and self-learning algorithms of neural networks. The model maps the input pattern of the sequence of the historical hourly load time series to the component of the next sequence. Input space is divided on fuzzy sets by fuzzy c-means clustering. The most informative input variables are determined using deterministic variable selection algorithms. Individual models are constructed for each day type and hour of the day. The method is applied to several load forecasting problems.*

***Streszczenie.** Adaptacyjny neuronowo-rozmyty system wnioskujący ANFIS zastosowano do prognozowania krótkoterminowego obciążeń systemów elektroenergetycznych. ANFIS łączy czytelność reguł rozmytych i adaptacyjność samouczących się sieci neuronowych. Model odwzorowuje obraz wejściowy sekwencji historycznego godzinowego szeregu czasowego obciążeń na składową obrazu następnej sekwencji. Przestrzeń wejściowa jest dzielona na zbiory rozmyte przy użyciu rozmytej metody c-średnich. Zmienne wejściowe niosące najwięcej informacji wyznaczane są za pomocą deterministycznych algorytmów selekcji zmiennych. Odrębne modele są tworzone dla każdego typu dnia i godziny doby. Metodę zastosowano do kilku problemów prognozowania obciążeń. (**Sieć neuronowo-rozmyta do prognozowania dobowej krzywej obciążenia z jednodobowym wyprzedzeniem**).*

Keywords: short-term load forecasting, neuro-fuzzy network, adaptive-network-based fuzzy inference system.
Słowa kluczowe: prognozowanie krótkoterminowe obciążeń, sieć neuronowo-rozmyta, adaptacyjny sieciowy rozmyty system wnioskowania.

Introduction

In many function approximation tasks, especially nonlinear ones, an artificial neural network (ANN) is used (multilayer perceptron) due to its universal approximation property. The approximation function depends on the network structure, internal connections and parameters (weights). The model obtained with neural network is not understandable in terms of physical parameters (black box model) and it is impossible to interpret the result in terms of natural language. On the other hand, the fuzzy rule base consists of if-then statements that are almost natural language, but it cannot learn the rules itself. To obtain a set of if-then rules two approaches are used. First, transforming human expert knowledge and experience, and second, automatic generation of the rules. The second method is intensively investigated. The fusion of neural networks and fuzzy logic in neuro-fuzzy models achieves readability and learning ability (extracting rules from data) at once.

The steps of fuzzy reasoning (inference operations upon fuzzy if-then rules) performed by fuzzy inference systems (FIS) are [1]:

1. Compare the input variables with the input membership functions on the premise part to obtain the membership values (or compatibility measures) of each linguistic label. This step is often called fuzzification.
2. Combine (through a specific *t*-norm operator, usually multiplication or min) the membership values on the premise part to get firing strength (weight) of each rule.
3. Generate the qualified consequent (either fuzzy or crisp) of each rule depending on the firing strength.
4. Aggregate the qualified consequents to produce a crisp output. This step is called defuzzification.

The parameters of the membership functions are tuned during training. A common way to apply a learning algorithm to a fuzzy system is to represent it in a special ANN like architecture. However the conventional ANN learning algorithms (gradient descent) cannot be applied directly to such a system as the functions used in the inference process are usually non differentiable. This problem can be tackled by using differentiable functions in the inference system or by not using the standard neural learning algorithm. There are several approaches to integrate ANN and FIS. Some of the major woks in this area are [2]: GARIC, FALCON, ANFIS, NEFCON, FUN, SONFIN, FINEST, EFuNN, dmEFuNN, ANNBFIS [3], evolutionary design of neuro-fuzzy systems, and many others.

Neuro-fuzzy approaches have been widely applied to the short-term load forecasting (STLF) [4-6]. In this article the ANFIS model to STLF is presented.

Load forecasting model based on ANFIS

One of the most popular neuro-fuzzy model is ANFIS (Adaptive-Network-Based Fuzzy Inference System) proposed by Jang [1]. ANFIS architecture is functionally equivalent to a Sugeno type fuzzy rule base and, under certain constraints, it is also equivalent to a radial basis function network.

ANFIS is a multi-input, single output quasi-nonlinear model consisting of a set of linguistic if-then rules. The ANFIS architecture in application to the next day hourly load curve forecasting is shown in Fig. 1. Squares indicate adaptive nodes, whereas circles indicate fixed nodes (without parameters).

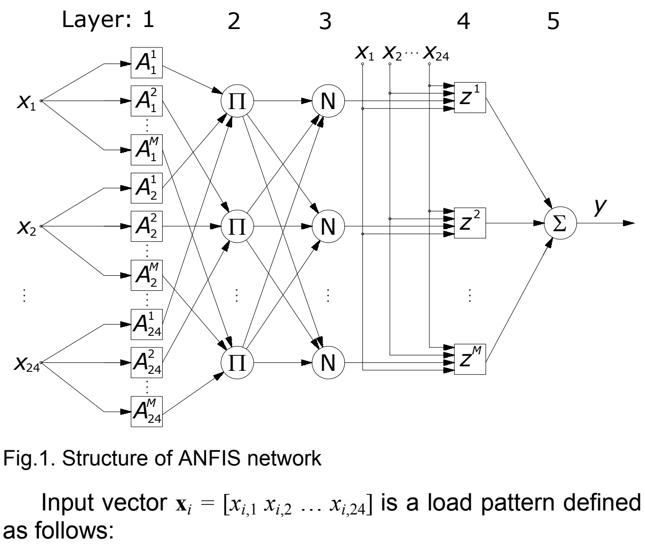

Fig.1. Structure of ANFIS network

Input vector $\mathbf{x}_i = [x_{i,1}\ x_{i,2}\ \ldots\ x_{i,24}]$ is a load pattern defined as follows:

$$x_{i,h} = \frac{L_{i,h} - \overline{L}_i}{\sqrt{\sum_{l=1}^{24} (L_{i,l} - \overline{L}_i)^2}}, \tag{1}$$

and an output is the component of the forecast pattern $\mathbf{y}_i = [y_{i,1}\, y_{i,2} \dots y_{i,24}]$ defined as follows:

$$y_{i,h} = \frac{L_{i+\tau,h} - \overline{L}_i}{\sqrt{\sum_{l=1}^{24}(L_{i+\tau,l} - \overline{L}_i)^2}}, \tag{2}$$

where: i – the day number; h = 1, 2, …, 24 – the hour of the day; τ – the forecast horizon, here τ = 1; $L_{i,h}$ – the load at hour h of day i; $\overline{L}_i$ – the mean load of day i.

Definition (1) expresses normalization of the original load vectors $\mathbf{L}_i$. After normalization they have the unity length, zero mean and the same variance.

Forecast patterns (2) are analogous to input patterns (1), but they are encoded using the current loads determined from the nearest past of the process history, what enables decoding of the forecasted vector $\mathbf{L}_{i+\tau}$ after the forecast of pattern $\mathbf{y}$ is determined.

Each layer of the ANFIS network consists of the nodes which functions are described below [1].

Layer 1. A node realises the membership function, which here is of the form of the Gaussian function:

$$\mu_{A_k^m}(x_k) = \exp\left[-\left(\frac{x_k - c_k^m}{\sigma_k^m}\right)^2\right], \tag{3}$$

where m = 1, 2, …, M – the fuzzy set number; k = 1, 2, …, 24 – the input variable number; A_k^m – the fuzzy set describing linguistically the input component x_k; c_k^m and σ_k^m – parameters (center and spread, respectively). These parameters are referred to as premise parameters.

The node output is the grade of membership to which the given input x_k satisfies the quantifier A_k^m. The number of nodes is determined by the number of linguistic labels M (here the number of clusters on which the input space is divided; this number is assumed a priori) and the number of input variables.

Layer 2. In this layer the firing strength of the mth rule α^m is calculated. The node function is the product t-norm:

$$\alpha^m = \prod_{k=1}^{24} \mu_{A_k^m}(x_k). \tag{4}$$

The closer the input vector components to the centres of the membership functions appearing in the mth rule, the higher the membership degrees $\mu_{A_k^m}$, and in consequence the higher the firing strength α^m.

Layer 3. Every node in this layer calculates the ratio of the mth rule firing strength relative to the sum of rule firing strengths:

$$\overline{\alpha}^m = \frac{\alpha^m}{\sum_{j=1}^{M}\alpha^j}. \tag{5}$$

The result is normalised firing strength.

Layer 4. In this layer the conclusion of each rule is determined according to the Takagi-Sugeno-Kang method, where the output membership functions are either linear or constant (first or zeroth order Sugeno-type systems). The output level of each rule is weighted by the firing strength. The node function for the first order system is of the form:

$$z^m = \overline{\alpha}^m\left(\sum_{k=1}^{24} a_k^m x_k + b^m\right), \tag{6}$$

where a_k^m and b^m are parameters referred to as consequent parameters.

Layer 5. The single node in this layer computes the overall output as the summation of all incoming signals:

$$y = \sum_{m=1}^{M} z^m = \frac{\sum_{m=1}^{M}\alpha^m\left(\sum_{k=1}^{24} a_k^m x_k + b^m\right)}{\sum_{j=1}^{M}\alpha^j}. \tag{7}$$

The output of each rule is a linear combination of input variables and the final ANFIS output is the weighted average of each rule output. Because the weights (firing strengths of rules) are dependent on input variables in nonlinear way the final output is nonlinear.

The rule base has the form:

$$\begin{aligned}
&\text{If } x_1 \text{ is } A_1^1 \text{ and...and } x_{24} \text{ is } A_{24}^1 \text{ then } z^1 = \sum_{k=1}^{24} a_k^1 x_k + b^1\\
&\text{If } x_1 \text{ is } A_1^2 \text{ and...and } x_{24} \text{ is } A_{24}^2 \text{ then } z^2 = \sum_{k=1}^{24} a_k^2 x_k + b^2\\
&\dots\\
&\text{If } x_1 \text{ is } A_1^M \text{ and...and } x_{24} \text{ is } A_{24}^M \text{ then } z^M = \sum_{k=1}^{24} a_k^M x_k + b^M
\end{aligned} \tag{8}$$

The left hand side of each rule defines a fuzzy region (cluster) for the linear model on the right hand side. The inference mechanism interpolates smoothly between each local model to provide a global model.

Before the start the ANFIS training an initial model must be specified. It can be achieved by applying clustering method on the input data. The number of clusters M equals to the number of membership functions assigned to each input variable can be chosen empirically by trial and error method. Fuzzy c-means clustering is often used to estimate initial positions of the input membership functions.

The neuro-adaptive learning method works similarly to that of neural networks. The backpropagation can be used for membership function parameter estimation [1]. The error measure is defined by the sum of the squared difference between actual and desired outputs.

The number of parameters of the described above forecasting model is equal to $M(3{\cdot}24+1)$ (each rule contains 73 parameters: 24 centers c_k^m, 24 spreads σ_k^m and 25 coefficients of the linear functions in the rule conclusions a_k^m and b^m). Because of many input variables, the number of parameters often exceeds the number of training patterns, what causes overfitting and poor predictive performance. In order to reduce the dimension of the input patterns and simplify the model, the variable subset selection procedures are used. The sequential forward selection (SFS) [7], based on the simple greedy deterministic heuristics, starts with an empty variable subset and adds one new variable to the current set of selected variables in each step. To determine which variable to add, the algorithm tentatively adds to the candidate variable subset one variable that is not already selected and tests the accuracy of a forecasting model built on the tentative variable subset. The variable that results in the highest accuracy is definitely added to the variable subset. The process stops after an iteration where no variable additions result in an improvement in accuracy. Similar method is the sequential backward selection (SBS) which starts with all the possible variables and discards one at the time.

Another way to reduce the number of parameters is decreasing the number of clusters M and the number of fuzzy rules in consequence.

Application examples

This section presents the simulation results of the proposed forecasting model. Five practical problems of the next day load curve forecasting are considered. Data are described in Table 1.

Table 1. Description of data used in experiments

Data symbol	Data description
A	Time series of the hourly loads of the Polish power system from the period 2002-2006, mean load of the system ~16 GW
B	Time series of the hourly loads of the Polish power system from the period 1997-2000, mean load of the system ~15,5 GW
C	Time series of the hourly loads of the local power system from the period July 2001-January 2003, mean load of the system ~1,2 GW
D	Time series of the hourly loads of the local power system from the period June 1998-July 2002, mean load of the system ~300 MW
E	Time series of the hourly load demands of the chemical plant from the period 1999-2001, mean load demand of the plant ~80 MW

For each day type and each hour of the day a separate neuro-fuzzy model was generated. MAPE (Mean Absolute Percentage Error) was used as a model quality criterion.

Datasets were divided into two subsets – training one and test one. The first sequences of the time series (two thirds of the whole time series) were included in the training set and the latest sequences were included in the test set. The number of clusters and rules was assumed to be 2. Fuzzy c-means clustering was used to determine the initial fuzzy sets. System parameters (c_k^m, σ_k^m, a_k^m and b^m) were estimated using backpropagation method.

Table 2 contains forecasting errors for the test sets using ANFIS models with and without variable selection. For comparison, forecast using the simple nearest neighbour (NN) method, multilayer perceptron (MLP) and the model based on the fuzzy estimators (FE) [8, 9] were calculated. More results in [9] can be found, where many other models were tested (including models based on artificial immune systems, neural gas, self-organizing maps and k-NN estimators).

The NN method applies the following rule: the forecasted y-pattern paired with the input x-pattern is the same as the y-pattern paired with nearest neighbour of the input x-pattern found in the reference set.

The MLP model consisted of only one linear neuron and was trained using the Bayesian regularization. For each day type and hour of the day a separate net (24 inputs and 1 output) was created and trained. That simple net structure was one of the best comparing to other structures tested in [8] because of good generalization properties.

The forecast results for these methods are presented in Table 2. More detailed results for the test part of A time series are presented in Fig. 2.

The variable selection gives usually better results, especially SFS. But in some cases the test error was significantly higher using SFS than without variable selection (peaks observed on Fig. 2). The reason for this can be that the relation between selected input variables and output variable, which is determined on training set, is not valid on test set. The average reduction in the input numbers was from 24 to 5.4 using SFS and to 19.7 using SBS. For example the model for time series A, day type Wednesday and hour 12 has only 2 input variables selected by SFS (x_{12} and x_{23}) or 21 input variables selected by SBS (all excluding x_9, x_{18} and x_{19}). Other, especially stochastic variable selection methods (genetic algorithms, simulated annealing, tournament searching [10]), not susceptible to the local minima traps, can improve results.

Table 2. Forecast errors for the test sets

Data symbol	ANFIS			NN	MLP	FE
	Without selection	SFS	SBS			
A	2.26	2.17	2.28	1.94	2.02	1.76
B	2.46	2.42	2.43	2.55	2.24	2.14
C	6.12	4.91	6.15	5.12	4.89	4.08
D	3.97	3.97	3.89	3.98	3.71	3.63
E	9.77	8.63	9.40	9.18	8.32	8.24

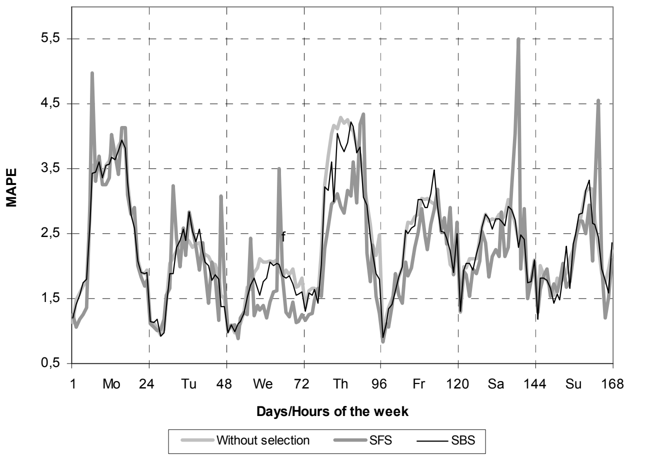

Fig.2. The forecast errors for time series A

In order to improve generalization properties (to prevent overfitting) the training was performed using validation data. When overfitting is detected (the validation error starts increasing while the training error is still decreasing) the training stops. Unfortunately the test error was not reduced after this procedure. This is because insufficient number of training points taking into account their dimension.

It can be interesting to know how stable the model is, i.e. whether the regression function does not change when we repeat training using the same learning data. To check this we train the model 30 times and calculate its bias and variance on the test set according to the equations:

(9) $$Bias = E[\hat{L}(M)] - L,$$

(10) $$Var = E[(\hat{L}(M) - E[\hat{L}(M)])^2],$$

where L and $\hat{L}$ are real and forecasted loads, respectively.

The sum of $Bias^2$ and Var gives the mean squared error of the model (MSE). The zero variance indicates the most stable model.

It is observed that variance increases according to the number of clusters (an example on Fig. 3 is presented). Thus the lower number of clusters ensures more stable models. The instability of the model derives from the learning process specificity, where bacpropagation method is used. This method looks for the minimum of the error function in parameter space using the algorithm of gradient descent. It allows convergence only on local minima for error. Which minimum is reached depends on the starting values of the parameters.

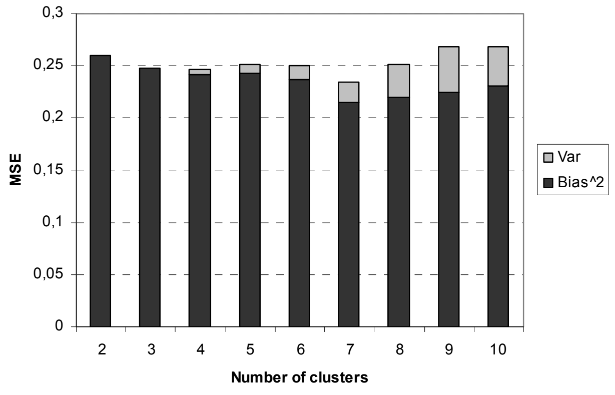

Fig.3. Bias and variance of the model (forecasts for time series A, day type Wednesday, hour 12)

Conclusions

The proposed load forecasting model has a complex structure and many parameters (degrees of freedom) – centres and spreads of input membership function and coefficients of the linear functions appearing in the rule conclusions (each rule contains 3*(number of variables)+1 parameters). It can cause overfitting in case of the limited number of training points. The selection of input variables and decreasing the number of rules are ways to reduce the parameter number and the model simplification.

The results have shown that despite the strong theoretical background, the STLF model based on the neuro-fuzzy network, in comparison to other STLF models (based on multilayer perceptron or fuzzy estimators of the regression function) gives higher errors for all of the application examples.

REFERENCES

[1] Jang J.-S.R., ANFIS: Adaptive-Network-Based Fuzzy Inference Systems. *IEEE Transactions on Systems, Man, and Cybernetics,* 23 (1993), n.3, 665-685

[2] Abraham A., Neuro Fuzzy Systems: State-of-the-Art Modeling Techniques, *Lecture Notes in Computer Science*, 2084 (2001), 269-276

[3] Łęski J., Czogała E., A New Artificial Neural Network Based Fuzzy Inference System with Moving Consequents in If-Then Rules, *BUSEFAL*, 71 (1997), 72-81

[4] Bakirtzis A. G. et al., Short Term Load Forecasting using Fuzzy Neural Networks, *IEEE Transactions Power Systems*, 10 (1995), 1518-1524

[5] Ling S.H. et al., Short-Term Electric Load Forecasting Based on a Neural Fuzzy Network, *IEEE Transactions on Industrial Electronics*, 50 (2003), n.6, 1305-1316

[6] Popławski T., Application of the Takagi-Sugeno (TS) Fuzzy Logic Model for Load Curves Prediction in the Local Power System, in: *IIIrd International Scientific Symposium Elektroenergetika* 2005, Stara Lesna Slovak Republic

[7] Theodoridis S., Koutroumbas K., Pattern Recognition, Elsevier Academic Press 2003

[8] Dudek G., Short-Term Load Forecasting using Fuzzy Clustering and Genetic Algorithms, *Final report of the Polish State Committee for Scientific Research founded grant no. 3T10B02329.* Dept. Elect. Eng., Częstochowa University of Technology 2006 (unpublished, in Polish)

[9] Dudek G., Similarity-Based Approaches to Short-Term Load Forecasting, in *Forecasting Models: Methods and Applications*, pp. 161-178, iConcept Press 2010

[10] Dudek G., Tournament Searching Method to Feature Selection Problem, in: Rutkowski L., Tadeusiewicz R., Zadeh L., Zurada J. (eds): *Lecture Notes in Artificial Intelligence*, Springer, Proceedings of the 10th International Conference on Artificial Intelligence and Soft Computing ICAISC 2010, (in print).

***Author**: Grzegorz Dudek PhD, Czestochowa University of Technology, Institute of Power Engineering, al. Armii Krajowej 17, 42-200 Czestochowa, Poland, E-mail: Dudek@el.pcz.czest.pl.*