Applied Soft Computing 36 (2015) 422–441

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Pattern similarity-based methods for short-term load forecasting – Part 2: Models

Grzegorz Dudek*

Department of Electrical Engineering, Czestochowa University of Technology, Al. Armii Krajowej 17, 42-200 Czestochowa, Poland

ARTICLE INFO

Article history:
Received 9 December 2014
Received in revised form 16 June 2015
Accepted 27 July 2015
Available online 5 August 2015

Keywords:
Artificial immune systems
Nonparametric regression
Short-term load forecasting
Similarity-based methods

ABSTRACT

Models for the short-term load forecasting based on the similarity of patterns of seasonal cycles are presented. They include: kernel estimation-based model, nearest neighbor estimation-based models and pattern clustering-based models such as classical clustering methods and new artificial immune systems. The problem of construction of the pattern similarity-based forecasting models and the elements and procedures of the model space are characterized. Details of the model learning and optimization using deterministic and stochastic methods such as evolutionary algorithms and tournament searching are described. Sensitivities of the models to changes in parameter values and their robustness to noisy and missing data are examined. The comparative studies with other popular forecasting methods such as ARIMA, exponential smoothing and neural networks are performed. The advantages of the proposed models are their simplicity and a small number of parameters to be estimated, which implies simple optimization procedures. The models can successfully deal with missing data. The increased number of the model outputs does not complicate their structure. The local nature of the models leads to their simplification and accuracy improvement. The proposed models are strong competitors for other popular univariate methods, which was confirmed in the simulation studies.

1. Introduction

The importance of the short-term load forecasting (STLF) in the power system control, scheduling and security translates into a large number of forecasting models. In the last few decades various forecasting methods have been proposed. They can be generally divided into conventional and unconventional methods. Conventional STLF models use regression methods, smoothing techniques and statistical analysis. Regression methods, linear and nonlinear, parametric or nonparametric, are usually applied to model the relationship between load consumption and other factors (weather, day type, customer class). Examples of semi-parametric additive models were recently presented in [1], whilst the nonparametric model using kernel estimators was presented in [2].

Gross and Galiana in their review paper [3] consider two basic conventional STLF models: time-of-day models and dynamic models. The former defines the load as a linear combination of a finite number of explicit time function, usually sinusoids with a period of 24 or 168 h. The latter take into account the most recent behavior of the time series and also exogenous variables and random component. Dynamic models are of two basic types: autoregressive moving average (ARMA) and state-space models. These approaches are used successfully up to today. Some examples such as ARIMA, exponential smoothing and the structural time series models are presented in the first part of this work [4]. Nowadays the conventional methods are often hybridized with new computational intelligence methods. As an example a new self-organizing model of fuzzy autoregressive moving average with exogenous input variables proposed in [5] can be given. In this approach a combined use of heuristics and evolutionary programming scheme is relied on to solve the problem of determining optimal number of input variables, best partition of fuzzy spaces and associated fuzzy membership functions. Good overview of the autoregressive moving average and other statistical approaches to modeling and forecasting electricity loads and prices can be found in [6]. Some conventional approaches to load forecasting such as static and dynamic state estimation are described in book [7].

The rapid development of computational intelligence observed in recent years has brought new methods of STLF. They are based on artificial neural networks (ANNs), fuzzy logic and expert systems. Also intelligent searching methods, such as evolutionary algorithms and swarm intelligence, are often applied to optimize the STLF models.

The multilayer perceptron (MLP), ANN which is most often applied in load forecasting, is an attractive tool to modeling of nonlinear problems due to their universal approximation property.

* Tel.: +48 343250896; fax: +48 343250803.
E-mail address: dudek@el.pcz.czest.pl

http://dx.doi.org/10.1016/j.asoc.2015.07.035
1568-4946/

Dudek G.: *Pattern Similarity-based Methods for Short-term Load Forecasting – Part 2: Models*. Applied Soft Computing, vol. 36, pp. 422-441, 2015.

http://dx.doi.org/10.1016/j.asoc.2015.07.035
http://www.gdudek.el.pcz.pl/files/SBFM_Models_15.pdf

Pattern Similarity-based Methods for Short-Term Load Forecasting – Part 2: Models

Grzegorz Dudek

***Abstract*—Models for the short-term load forecasting based on the similarity of patterns of seasonal cycles are presented. They include: kernel estimation-based model, nearest neighbor estimation-based models and pattern clustering-based models such as classical clustering methods and new artificial immune systems. The problem of construction of the pattern similarity-based forecasting models and the elements and procedures of the model space are characterized. Details of the model learning and optimization using deterministic and stochastic methods such as evolutionary algorithms and tournament searching are described. Sensitivities of the models to changes in parameter values and their robustness to noisy and missing data are examined. The comparative studies with other popular forecasting methods such as ARIMA, exponential smoothing and neural networks are performed. The advantages of the proposed models are their simplicity and a small number of parameters to be estimated, which implies simple optimization procedures. The models can successfully deal with missing data. The increased number of the model outputs does not complicate their structure. The local nature of the models leads to their simplification and accuracy improvement. The proposed models are strong competitors for other popular univariate methods, which was confirmed in the simulation studies.**

***Index Terms*—Artificial Immune Systems, Nonparametric Regression, Short-Term Load Forecasting, Similarity-based Methods**

I. Introduction

THE importance of the short-term load forecasting (STLF) in the power system control, scheduling and security translates into a large number of forecasting models. In the last few decades various forecasting methods have been proposed. They can be generally divided into conventional and unconventional methods. Conventional STLF models use regression methods, smoothing techniques and statistical analysis. Regression methods, linear and nonlinear, parametric or nonparametric, are usually applied to model the relationship between load consumption and other factors (weather, day type, customer class). Examples of semi-parametric additive models were recently presented in [1], whilst the nonparametric model using kernel estimators was presented in [2].

The study was supported by the Research Project N N516 415338 financed by the Polish Ministry of Science and Higher Education.

G. Dudek is with the Department of Electrical Engineering, Czestochowa University of Technology, 42-200 Czestochowa, Al. Armii Krajowej 17, Poland (e-mail: dudek@el.pcz.czest.pl).

Gross and Galiana in their review paper [3] consider two basic conventional STLF models: time-of-day models and dynamic models. The former defines the load as a linear combination of a finite number of explicit time function, usually sinusoids with a period of 24 or 168 h. The latter take into account the most recent behavior of the time series and also exogenous variables and random component. Dynamic models are of two basic types: autoregressive moving average (ARMA) and state-space models. These approaches are used successfully up to today. Some examples such as ARIMA, exponential smoothing and the structural time series models are presented in the first part of this work [4]. Nowadays the conventional methods are often hybridized with new computational intelligence methods. As an example a new self-organizing model of fuzzy autoregressive moving average with exogenous input variables proposed in [5] can be given. In this approach a combined use of heuristics and evolutionary programming scheme is relied on to solve the problem of determining optimal number of input variables, best partition of fuzzy spaces and associated fuzzy membership functions. Good overview of the autoregressive moving average and other statistical approaches to modeling and forecasting electricity loads and prices can be found in [6]. Some conventional approaches to load forecasting such as static and dynamic state estimation are described in book [7].

The rapid development of computational intelligence observed in recent years has brought new methods of STLF. They are based on artificial neural networks (ANNs), fuzzy logic and expert systems. Also intelligent searching methods, such as evolutionary algorithms and swarm intelligence, are often applied to optimize the STLF models.

The multilayer perceptron (MLP), ANN which is most often applied in load forecasting, is an attractive tool to modeling of nonlinear problems due to their universal approximation property. Its other useful properties are: massive parallelism among a large number of simple units, learning capabilities, robustness in the presence of noise, and fault tolerance. Many forecasting models based on the MLP is used in practice by electric companies. An example would be ANNSTF system, which uses more than 40 companies from the U.S. and Canada [8]. Examples of some new publications on the use of MLP in STLF are: [9], where the complexity of MLP applied to STLF problems has been controlled by the Bayesian approach, [10], where a new hybrid forecasting method composed of wavelet transform, MLP and evolutionary algorithm is proposed, [11], where a generic framework that combines similar day selection, wavelet decomposition, and MLP is presented, [12], where MLP is

combined with wavelet transform and particle swarm optimization, [13], where an approach of MLP with rough sets for complicated STLF with dynamic and non-linear factors is proposed, and [14], where the neural model generates the prediction intervals.

A radial basis function (RBF) network is an alternative to MLP in STLF. The RBF network approximates the target function by a linear combination of radial functions (usually Gaussian), which nonlinearly transform the input data. The learning algorithms for RBF are simpler than for MLP. The RBF network has a property of universal approximation. Some new publications concerning the STLF models based on the RBF network are: [15], where RBF is combined with fuzzy inference system and genetic algorithm, [16], where a model to STLF is established by combining the RBF network with the adaptive neural fuzzy inference system and [17], where RBF is combined with the wavelet transform.

A self-organizing feature map (SOFM) is an another ANN used in STLF. This network is trained using unsupervised competitive learning to produce a low-dimensional representation of the input pattern space. The input patterns are grouped and represented by neurons. Some examples of STLF models using SOFM are: [18], where a hierarchical model composed of two SOFM is presented, [19], where an adaptive two-stage hybrid network with SOFM and support vector machine is proposed, [20], where SOFM is combined with MLP and a flexible smooth transition autoregressive model, and [21], where nonlinear model based on SOFM and predictors determined using curvilinear component analysis is described.

Many other types of neural networks have been used for STLF including: recurrent networks, generalized regression ANN [22], probabilistic ANN, adaptive resonance theory ANN, functional link network and counterpropagation ANN. The survey of ANN applications to STLF can be found in [23] and [24].

Fuzzy logic allows to take into account imprecise, incomplete and ambiguous information in the STLF models. Fuzzy models are often simpler and more accurate than standard statistical models and allow to enter input information by rules formulated verbally by experts. The advantage of fuzzy inference systems is that they describe the behavior of complex systems by using linguistic expressions, mimicking the action of man. The fuzzy rule base consists of if-then statements that are almost natural language. To obtain a set of if-then rules two approaches are used. First, transforming human expert knowledge and experience, and second, automatically generating the rules from examples. The fusion of neural networks and fuzzy logic in neuro-fuzzy models achieves readability and learning ability (extracting rules from data) at once. The fuzzy inference mechanism leads to a nonlinear global model, which is an interpolation of local models implemented in the individual rules. The fuzzy STLF models are based on: fuzzy interpolation [25], fuzzy linear regression [26], fuzzy C-regression [27], Takagi-Sugeno-Kang model [28], fuzzy inductive reasoning [29] and neuro-fuzzy networks. The main advantages of the latter hybrid approach are: the ability to respond accurately to unexpected changes in the input variables, the ability to learn from experience, and the ability to synthesize new relationships between the load demand and the input variables. Examples of such STLF models are: [30], where the neuro-fuzzy system is used to adjust the results of load forecasting obtained by RBF network, [31], where two neuro-fuzzy networks are proposed: a wavelet fuzzy neural network using the fuzzified wavelet features as the inputs and fuzzy neural network employing the Choquet integral as the outputs, [32], where an efficient adaptive fuzzy neural network is proposed which can reduce its complexity removing the unneeded hidden units, [33], where an integrated approach which combines a self-organizing fuzzy neural network learning method with a bilevel optimization method, [34] where a neuro-fuzzy system working on the seasonal cycle patterns is proposed, and [35], where fuzzy logic is combined with wavelet transform and neural network.

Another useful tools for STLF are: support vector machines [36], [37], [38], clustering methods [19], [20], [39] and ensembles of models [40], [41], [42]. An interesting approach which can be classified as the similarity-based one is presented in [43]. It uses the clustering of the normalized daily curves for definition new inputs: sequences of the group labels for the successive days. The sequences are paired with load curves of the next days. The forecast is composed from the daily curves paired with the sequences from the history which are the same as the current sequence. Another interesting similarity-based model for STLF is described in [44]. The forecast is calculated as the a weighted average of past daily load segments, the shape of which is similar to the expected shape of the load segment to be predicted.

It is noteworthy that many of the models developed in recent years are hybrid solutions (most papers concerning STLF published in IEEE Transaction on Power Systems in the last 10 years relate to just such models). These approaches combine data preprocessing methods (e.g. wavelet transform) with approximation models (such as neural and neuro-fuzzy networks) and methods of optimization and learning of these models (e.g. evolutionary and swarm algorithms). Sometimes forecast is adjusted depending on additional factors affecting the load demand and not included in the basic model.

This paper presents the univariate STLF models based on similarities between patterns of the daily cycles of the load time series. The principles of the models were described in the first part of this work [4]. The main advantage of the pattern similarity-based forecasting models (PSBFMs) is their simplicity: they have a clear structure and comprehensible principles of operation. The number of parameters is low and the optimization and learning procedures are fast.

The remainder of this paper is divided into seven sections. The problems of construction of the PSBFMs in Section II are presented. In Section III–V the STLF models based on pattern similarity including non-parametric regression methods and clustering methods are presented. In Section VI we analyze the proposed forecasting methods and we compare the results to other STLF methods: ARIMA, exponential smoothing and

MLP. An overview of the work is given in Section VII.

II. Construction of PSBFMs

The forecasting models considered in this work are memory-based inductive approximation models which learn the relevant relationships between variables on the basis of observed instances. Instances (examples or samples) are pairs of x- and y-patterns extracted from the load time series using the functions f_x and f_y (see Section III in [4]). Instances form the series $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$, $i = 1, 2, \ldots, N$, $\mathbf{x}_i \in X = \mathbb{R}^n$, $\mathbf{y}_i \in Y = \mathbb{R}^n$, where X and Y are domain and codomain, respectively. The goal is to approximate the vector-valued function $g : X \to Y$. We can decompose this problem by treating the vector-valued function as vectors of scalar-valued functions and approximating these functions separately: $g_t : X \to Y_t = \mathbb{R}$, $t = 1, 2, \ldots, n$. We expect that the model generates approximating function $f(\mathbf{x},\phi)$, $\phi \in \Phi$, where Φ is a set of acceptable values of parameters ϕ, which approximates accurately the unknown (seen only by the examples) target function g or g_t. In the proposed PSBFMs the regression function has the nonparametric form $m(\mathbf{x})$ (see (5)).

The optimal model is selected during searching the model space, which is a combination of the following elements and procedures:

$$M = \{f_x(z,\psi), f_y(z,\psi), s(\mathbf{x}_a,\mathbf{x}_b), sel(\mathbf{x}), \mathcal{L}, L(\mathcal{L}), w(\mathbf{x}_a,\mathbf{x}_b), m(\mathbf{x}), Q(M), O(M)\}, \quad (1)$$

where:

$f_x(z,\psi)$, $f_y(z,\psi)$ – functions which map the original time series elements $\{z_t\}$ into x- and y-patterns, respectively,

$s(\mathbf{x}_a,\mathbf{x}_b)$ – the pattern similarity function,

$sel(\mathbf{x})$ – the feature selection procedure,

$\mathcal{L}$ – the training sample to estimation of the model parameters,

$L(\mathcal{L})$ – the learning mode, which determines how the elements of the learning sample are used during learning to estimate the generalization error (e.g. cross-validation, bootstrap),

$w(\mathbf{x}_a,\mathbf{x}_b)$ – the weighting function, which gives the weights to patterns according to their similarities to the query pattern,

$m(\mathbf{x})$ – the nonparametric regression function,

$Q(M)$ – the measure of the model quality,

$O(M)$ – the optimization procedures.

The functions defining patterns and the similarity measures are described in Section III of the first part of this work [4].

The aim of the feature selection is to reduce the dimensionality of the x-pattern vector by elimination of irrelevant, redundant and unpredictive components. The x-pattern composed with selected features should ensures the best quality of the learning model. The dimensionality reduction is related to the curse of dimensionality problem. It concerns especially similarity-based methods, where we infer about the target function based on the neighborhood of the query pattern. There are many manifestation of this problem [45]. For example in high dimensions data points are closer to the boundary of the sample space than to any other data point, so the prediction is much more difficult. It requires extrapolation from neighboring sample points rather that interpolate between them. Another problem is that in high dimensions the training samples sparsely populate the input space. Their density is proportional to $N^{1/n}$. The distance between the closest points increases, and the distances between all pairs of points are similar. The function complexity can grow exponentially with the dimension, and if we want to estimate the function with high accuracy we need the size of training set to grow exponentially as well. In STLF the x-pattern size n is 24, 48 or 96 for hourly, half-hourly or quarter-hourly load time series, respectively. Meanwhile, the length of the time series is limited to the period of several years, which gives the size of the learning sample N of hundreds or thousands instances. This is insufficient. Since the acquisition of additional samples is unreal, the solution is to reduce the dimension of vector $\mathbf{x}$. But it should be noted that the components of x-patterns are highly correlated which means that in the input space there are regions with greater density and regions which are empty. This reduces unfavorable phenomenon of the curse of dimensionality and enables to approximate the function locally in the denser regions with greater accuracy.

The measure of the space filling by a set of random points can be a fractal dimension. Among many different types of fractal dimensions we choose the correlation dimension [46], which is based on the correlation integral defined according to:

$$C(r) = \lim_{N\to\infty} \frac{1}{N^2} \sum_{i,j=1}^{N} H\left(r - d(\mathbf{x}_i, \mathbf{x}_j)\right), \quad (2)$$

where $H(.)$ is the Heaviside step function, r is the radius and $d(.,.)$ is the distance between two points.

$C(r)$ is proportional to the total number of pairs of points closer then r to each other. For small r the correlation integral grows like a power: $C(r) \sim r^v$, and v is interpreted as the correlation dimension. If the number of points is sufficiently large a log-log graph of the correlation integral versus r will yield an estimate of v. For the hourly load time series of the Polish power system the correlation dimension was 2.01 (X3.1 patterns were used – see Section III in [4]), whilst for the random points distributed uniformly in the same region v = 7.59.

Another simple measure of the space filling is proposed: the length of the transition path at all points, wherein each point is visited once and the next step is performed to the unvisited nearest neighbor. This path was three times longer for points uniformly filling the space than for points representing x-patterns for the Polish power system regardless of the starting point.

The shorter path and smaller correlation dimension indicate that the intrinsic dimension of the set of x-pattern is less than 24. Feature selection procedure can reduce dimensionality as

well as feature extraction methods such as principal component analysis. In the experimental part of this work the genetic algorithms and tournament searching method [47] are used as wrappers to the feature selection.

Next element of model (1) is the learning sample. In [4] it was shown that the properties and performance of PSBFMs are better when the learning sample contains patterns representing the same day of a week as the query pattern. In the proposed approach for each forecasting task, i.e. forecasting the daily load curve of the day $i+\tau$ or, after decomposition, load at the time t of this day, the individual model is learned and optimized. This allows to fine-tune the model to the specifics of this task. The learning sample in this case includes pairs of patterns representing the same days of a week from the history as the query instance $(\mathbf{x}_i, \mathbf{y}_i)$: $L = \{(\mathbf{x}_j, \mathbf{y}_j)\}$, $j = i–7q, i–7(q–1), \ldots, i–7$, where $q = \lfloor (i–1)/7 \rfloor$. The outliers are removed from the training set.

To estimate the generalization error in the learning and optimization processes the leave-one-out cross-validation (LOO) is used. This procedure can be applied in two local versions. In the first case (LOO-v1) the validation samples are chosen one by one from the set of nearest neighbors of the query pattern. We do not need to learn the model for each training sample but only for some samples from the neighborhood of query x-pattern. Thus we gain savings in computation time and more accurate fitting of the model in the neighborhood of the query pattern. In the second local version of leave-one-out (LOO-v2) we determine the error for each training point (global LOO) and we estimate the generalization error averaging these errors with the weights v dependent on the distance between training points and the query point:

$$WMAPE_{val} = \frac{\sum_{j=1}^{N} v_i APE_{val,j}}{\sum_{j=1}^{N} v_j}, \tag{3}$$

where

$$v_j = \exp\left(-\left(\frac{d(\mathbf{x}^*, \mathbf{x}_j)}{\sigma}\right)^2\right), \tag{4}$$

APE is the absolute percentage error, $\mathbf{x}^*$ is the query pattern, $d(.,.)$ is the distance function and σ controls the width of the Gaussian function (4).
Equation (3) expresses the weighted mean absolute percentage error. (MAPE is traditionally used as an error measure in STLF.)

The regression function $m(\mathbf{x})$ in (1) has the nonparametric form:

$$m(\mathbf{x}) = \sum_{j=1}^{N} w(\mathbf{x}, \mathbf{x}_j)\mathbf{y}_j . \tag{5}$$

The weighting function $w(\mathbf{x},\mathbf{x}_j)$ is dependent on the similarity or distance between patterns $\mathbf{x}$ and $\mathbf{x}_j$. Usually its value decreases monotonically with the distance and

$$\sum_{j=1}^{N} w(\mathbf{x}, \mathbf{x}_j) = 1. \tag{6}$$

Forms of the weighting function are described in the next sections.

The quality measures of the model $Q(M)$ are mainly based on the error which is minimized in the training process (MAPE here). They can also include a component related to the model complexity. Typical examples of such a measure are the Akaike or Bayesian information criterions.

The goal of the model optimization process is to find its structure and parameter values to get the minimum of the objective function measuring the model quality. The optimization procedures $O(M)$ are dependent on the model parameters and the objective function character. Most preferably is to optimize the model in the space of all parameters, but usually it is unrealistic because of the different types of these parameters (continuous, integer, binary, enumeration), the huge size of the space and multimodality of the objective function. The solution is the decomposition of the optimization problem into subproblems, each of which includes some subset of the parameters. These subproblems are solved alternately or one by one. This approach leads to the local optimum, rarely to the global one.

The proposed PSBFMs have few parameters, which is their great advantage. To optimization of these models the exhaustive search method is even possible after discretization of the continuous parameters. In the experimental part of this work evolutionary algorithms and tournament searching methods are used as well. Their advantage is the ability to simultaneously optimization of the parameters of different types (continuous and discrete) and the global optimization property. The implementation details of this algorithms are described later in this work.

III. Kernel Estimation-based Model

The kernel methods are characterized by flexibility in the estimation of the regression function $m(.)$. This flexibility is due to the local nature of fitting of the simple regression models. The most popular estimator from this group is the Nadaraya-Watson estimator (N-WE):

$$m(x) = \frac{\sum_{j=1}^{N} K\left(\frac{x - x_j}{h}\right) y_j}{\sum_{j=1}^{N} K\left(\frac{x - x_j}{h}\right)}, \tag{7}$$

where $K(.,.)$ is a kernel function and h is a bandwidth .
When we put vector $\mathbf{y}_j$ in (7) instead of scalar y_j we get a vector valued function like (8). In the experimental part of this

work we use both: scalar y_j getting MISO model and vector $\mathbf{y}_j$ getting MIMO model.

For multidimensional input variables the kernels are expressed using a multidimensional product kernel function. In this case the estimator is defined as:

$$m(\mathbf{x}) = \frac{\sum_{j=1}^{N}\prod_{t=1}^{n} K\left(\frac{x_t - x_{j,t}}{h_t}\right)\mathbf{y}_j}{\sum_{j=1}^{N}\prod_{t=1}^{n} K\left(\frac{x_t - x_{j,t}}{h_t}\right)}. \quad (8)$$

The selection of the kernel function form is not as important as the selection of its bandwidth. We choose a normal kernel and the estimator is now of the form:

$$m(\mathbf{x}) = \frac{\sum_{j=1}^{N}\exp\left(-\sum_{t=1}^{n}\frac{(x_t - x_{j,t})^2}{2h_t^2}\right)\mathbf{y}_j}{\sum_{j=1}^{N}\exp\left(-\sum_{t=1}^{n}\frac{(x_t - x_{j,t})^2}{2h_t^2}\right)}. \quad (9)$$

Estimator (8) is a linear combination of vectors $\mathbf{y}_j$ (or scalars $y_{j,t}$ in scalar-valued version of the model) weighted by the normalized kernel functions (to satisfy constraint (6)) which nonlinearly map the distance between patterns $\mathbf{x}$ and $\mathbf{x}_j$. The greater the distance the lower the weight. The distance is parameterized by the bandwidths. The parameter h_t strengthens or weakens the share of the t-th component of $\mathbf{x}$ in the distance. This is an analogy to the weighted feature selection where weights are not binary but continuous. The bandwidth values decide about the bias-variance tradeoff of the estimator. Too small bandwidth values result in undersmoothing, whereas too large values result in oversmoothing. Thus the selection of the bandwidth values is a key problem. The simplest way is to adopt the h values from the formula proposed by Scott [48] for the normal product density estimators:

$$h_t^S = \hat{\sigma}_t N^{-\frac{1}{n+4}}, \quad (10)$$

where $\hat{\sigma}_t$ is the standard deviation of the t-th component of $\mathbf{x}$ estimated from the learning sample.

The next step is to search the neighborhood of the point $\mathbf{h}^S = [\,h_1^S \;\; h_2^S \;\; \ldots \;\; h_n^S\,]$ to adjust the bandwidths to our problem. The simplest method is the iteration process where the $\mathbf{h}$ vectors are generated according to the scheme:

$$\mathbf{h}_l = a_l \mathbf{h}^S, \quad l = 1, 2, ..., \quad (11)$$

where $a_l = a_0 + \Delta(l-1)$, $a_0 \in \mathbb{R}^+ \le 1$, Δ is the step defining the density of search.

The final value of l results from the stop criterion such as L iterations without improvement in results. This grid method (GM) is sub-optimal and searches sets of discrete values of the components of $\mathbf{h}$. The multi-dimensional optimization problem is here replaced with a simple one-dimensional optimization problem (searching of a value instead of h_1, h_2, ..., h_n).

To the individual, independent tuning of each bandwidth the evolutionary algorithm (EA) and the tournament searching (TS) are used. In EA the vectors $\mathbf{h}$ are individuals. The population of individuals is initialized by the Scott's rule (10). The evolutionary process consists of mutation, recombination and selection. The mutation operator adds to each component of $\mathbf{h}$ the random disturbance ξ from the normal distribution with mean zero and standard deviation σ:

$$h_t' = h_t + \xi_t, \quad t = 1, 2, ..., n, \quad (12)$$

The standard deviation σ determines the mutation range (diversity of mutants). It is assumed that $\sigma_t = w_\sigma h_t^S$, where $w_\sigma = \text{const} \in \mathbb{R}^+$. Thus the mutation range in the t-th direction is dependent on the initial value of h_t, i.e. on the variance of x_t.

The arithmetic recombination (intermediate) [49] was applied. This operator produces two new individuals by taking two linear combinations of the parent individuals which are selected by random:

$$h_{a,t}' = h_{a,t} + c(h_{b,t} - h_{a,t}), \quad (13)$$

$$h_{b,t}' = h_{b,t} + c(h_{a,t} - h_{b,t}), \quad (14)$$

where $c \sim U(0,1)$, $t = 1, 2, \ldots, n$.

As a selection operator the tournament selection was applied [49]. The tournament size T_s determines the selection pressure. To save the best solution the elitist strategy was used: the best individual in the population is copied to the next population.

The EA parameters are: the population size, the number of generations, the tournament size, probability of mutation and probability of recombination.

The TS method has been proposed in [47] to feature selection problem as an alternative to the more complex combinatorial optimization algorithms such as genetic algorithm and simulated annealing. In application to the continuous optimization problem of estimation of the bandwidth values it is redefined and labeled as TSc. The TSc explores the solution space starting from $\mathbf{h}^S$ determined by the Scott's rule and generating new solutions by perturbing it. When the set of new l candidate solutions $\{\mathbf{h}_1 \; \mathbf{h}_2 \; \ldots \; \mathbf{h}_l\}$ is generated (l is called the tournament size), their costs are calculated using the learning model. The best candidate solution (the tournament winner), with the lowest value of the cost function is selected and it replaces the parent solution, even in case it is worse than the parent solution (this prevents getting stuck in local minima). The only operator is the move operator which is identical to the mutation operator (12). The standard deviation of mutation σ and the tournament size l

decide about the exploration/exploitation properties of the algorithm. If the tournament size is equal to 1, this procedure comes down to the random walk. On the other hand, when l increases the neighborhood of the parent solution is sampled densely (local searching) and this method becomes more greedy.

The TS method in binary version (TSb) [47] was applied to the selection of the x-pattern components. The solution is represented by a binary vector composed of bits corresponding to n components of $\mathbf{x}$: $\mathbf{b} = [b_1, b_2, \ldots, b_n]$. The bit value indicates whether the component is selected (1) or not (0). The starting solution is initialized by random. The move operator generates $l \in \{1, 2, \ldots, n\}$ candidate solutions by switching the value of the randomly chosen bit (different for each candidate solution) of the parent solution. For $l = 1$ we get a random walk, and for $l = n$ we get a hill climbing procedure. The former has a global search property, the latter is the local deterministic search method. The tournament size decides about the exploration/exploitation properties, as in the case of TSc. The best candidate solution replaces the parent solution in the next generation.

To the component selection the genetic algorithm (GA) and two deterministic suboptimal methods: sequential forward selection (SFS) and sequential backward selection (SBS) [50] are also used. The solution representation in all these algorithms was the binary vector $\mathbf{b}$, the same as in TSb. The GA consists of the bit-flip mutation, one-point crossover and tournament selection.

Results of the bandwidth optimization and selection of the x-pattern components are obviously dependent on each other. For simultaneous optimization of the model in these two spaces the algorithm based on TS is proposed (labeled as TScb). The algorithm processes two connected vectors: $\mathbf{b}$ encoding binary the selected components, and $\mathbf{h}$ encoding the bandwidths. The vectors $\mathbf{b}$ and $\mathbf{h}$ are initialized as in TSc and TSb, respectively. There are two types of the move operator: one for the $\mathbf{b}$ vector and second for the $\mathbf{h}$ vector. The former is the same as in TSb. The latter has form (12), wherein only these components of $\mathbf{h}$ are modified which correspond to 1s in the paired $\mathbf{b}$ vector. The tournament size is $l \in \{1, 2, \ldots, n\}$. The best candidate solution becomes the parent solution in the next iteration.

IV. Nearest Neighbor Estimation-based Models

The nearest neighbor estimate $m(\mathbf{x})$ is defined as the weighted average of the response variables in a varying neighborhood of $\mathbf{x}$. This neighborhood is defined through those x-patterns which are among the k nearest neighbors of the query pattern. The value of k determine the number of pattern from which the regression function is constructed (these patterns are called the construction patterns). If the response and explanatory variables are vectors the k-NN estimator is defined as:

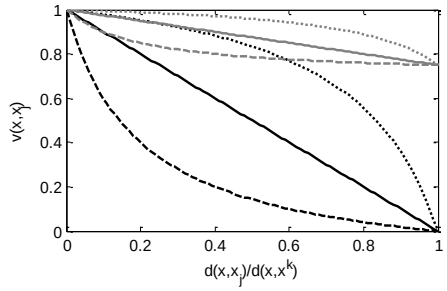

Fig. 1. The weighting function (16) for $\gamma = 0$ - solid line, $\gamma = 5$ - dashed line, $\gamma = -0.8$ - dotted line, $p = 1$ - black, $p = 0.25$ - gray.

$$m(\mathbf{x}) = \frac{\sum_{j=1}^{N} v(\mathbf{x}, \mathbf{x}_j)\mathbf{y}_j}{\sum_{j=1}^{N} v(\mathbf{x}, \mathbf{x}_j)}, \tag{15}$$

where $v(\mathbf{x},\mathbf{x}_j)$ is the weighting function of the form:

$$v(\mathbf{x}, \mathbf{x}_j) = p\left(\frac{1 - \dfrac{d(\mathbf{x}, \mathbf{x}_j)}{d(\mathbf{x}, \mathbf{x}^k)}}{1 + \gamma \dfrac{d(\mathbf{x}, \mathbf{x}_j)}{d(\mathbf{x}, \mathbf{x}^k)}} - 1\right) + 1, \tag{16}$$

$j \in \Theta_k(\mathbf{x})$, $\Theta_k(\mathbf{x})$ is the set of the k nearest neighbors of $\mathbf{x}$, $\mathbf{x}^k$ is the k-th nearest neighbor of $\mathbf{x}$, $p \in [0, 1]$ is a parameter that controls the degree of differentiation of weights, $\gamma \geq -1$ is a parameter that controls the convexity of the function.

For $p = 1$ the weights are the most diverse, for $p = 0$ all weights are the same, equal to 1. When $\gamma = 0$ the weighting function (16) is linear, when $\gamma > 0$ it decreases more rapidly than a linear function, and when $\gamma < 0$ it decreases slower than a linear function. The weighting function is shown in Fig. 1.

The number of the nearest neighbors k is a parameter controlling the degree of smoothing. It performs a similar function to the bandwidths in the N-WE. When $k = 1$, the regression function is a step function exactly fitted to the learning points. Increasing k leads to smoothing the regression function, which implies an increase in the bias and the reduction in variance of the model. The k-NN estimator gives the regression function, which is less smooth than in the case of the Gaussian kernel estimation. It is discontinuous: in the points where the set of the nearest neighbors is modified the jumps on the function graph appear.

In the N-WE the kernel functions are stretched over each learning point. This gives the opportunity to decide about the influence of individual points on the shape of the regression function. In the nearest neighbor estimators the weighting function is one, stretched over the query point. Thus there is no possibility of such a flexible control of the impact of the individual points on the regression curve. Moreover the number of the construction points are limited to k.

In the above-described approach the neighborhood of the

query point include the k nearest neighbors. In [51] a fuzzy membership of the learning points to the neighborhood of the query point was introduced. In this case, each learning point belongs to this neighborhood but with a different degree. The number of the construction points is equal to the learning sample size N. The weighting function has a form of the membership function, e.g.:

$$\mu(\mathbf{x},\mathbf{x}_j) = \exp\left(-\left(\frac{d(\mathbf{x},\mathbf{x}_j)}{\sigma}\right)^{\alpha}\right), \tag{17}$$

or

$$\mu(\mathbf{x},\mathbf{x}_j) = \left(1+\left(\frac{d(\mathbf{x},\mathbf{x}_j)}{\sigma}\right)^{\alpha}\right)^{-1}, \tag{18}$$

where α and σ are parameters controlling the shape of the function (see Fig. 2).

These functions have a maximum in $d(\mathbf{x}, \mathbf{x}_j) = 0$. Membership function (17) is a Gaussian-type function whereas (18) is a Cauchy-type function with fatter tail than the Gaussian function, which provides a greater influence of the more distant points on the regression curve.

The forecasting models based on nearest neighbor estimators are characterized by a small number of parameters. There are only three parameters in the k-NN model: k, p and γ. The values of these parameters can be estimated in a grid search procedure for $k = 1, 2, \ldots, k_{\max}$, $p = 0, \Delta, 2\Delta, \ldots, 1$, $\gamma = \gamma_1, \gamma_2, \ldots, \gamma_m$. In fuzzy neighborhood models (FNMs) there are two parameters: α and σ, which can be estimated in the same way. In [51] to the estimation of these parameters two local optimization methods were used: the Nelder–Mead simplex method and quasi-Newton method. To increase the probability of finding global minima the multistart was used.

The components of the vectors **x** can be selected using the same methods as for the N-WE. In [51] to component selection for the FNM the weighted feature selection was applied, where the importance of the components were expressed using not binary weights but continuous ones from the range of [0, 1]. The distance measure in (17) and (18) is based on the components of **x** multiplied by the corresponding weights. To estimation of the weights two methods were used: the (μ, λ) evolution strategy and the evolutionary algorithm with the continuous representation, Gaussian mutation, uniform crossover and tournament selection.

For the simultaneous search of the feature space and the width parameter space (k or σ) a combination of the TSb and the grid search is proposed. Each solution **b** representing the selected components of **x** is evaluated for each value of the k or σ from the assumed range (σ is discretized). The best score and the width parameter value at which it was achieved are assigned to the solution **b**. The solutions are modified using the move operator for TSb described above. The best solution among l candidate solutions replaces the parent solution. Instead of TSb other feature selection method can be applied in combination with the grid search. In Section VIA the SFS and SBS methods are used as well.

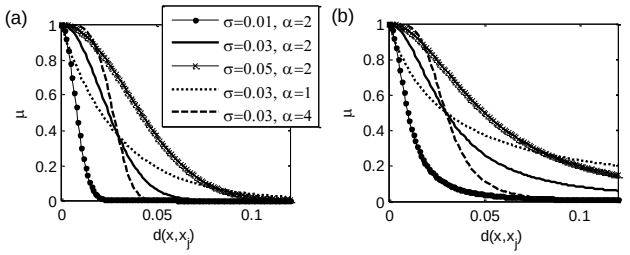

Fig. 2. The membership functions: Gaussian functions (a), Cauchy functions (b).

V. Pattern Clustering-based Models

The aim of clustering is to extract clusters of patterns representing similar shapes of the load curves. Grouping patterns allows to decrease in the number of construction patterns, which now represent the clusters of original patterns. This can lead to the reduction of the impact of errors affecting the data on the estimator accuracy, and improvements in generalization. Two forecasting procedures based on pattern clustering and two new approaches based on the artificial immune systems are described below.

A. Forecasting Procedures

In the first forecasting procedure (FP1) the paired vectors **x** and **y** are concatenated and form vector $\mathbf{u} = [\mathbf{x}^T\ \mathbf{y}^T]^T$. When we forecast the daily load curve for the day type s (Monday, …, Sunday), the vectors **u** that include y-patterns for day s are selected and grouped. After the clustering phase, each cluster C is represented by a single vector **m** (prototype of a cluster), which has two parts corresponding to x- and y-patterns: $\mathbf{m}_x$ and $\mathbf{m}_y$. The prototype vector **m** is a point located inside the cluster. Its position depends on the clustering method. In the forecasting phase the query x-pattern is presented and it is assigned to the cluster i^* represented by the closest prototype to the query pattern:

$$i^* = \arg\min_{i=1,2,\ldots K} d(\mathbf{x},\mathbf{m}_{x,i}), \tag{19}$$

where K is the number of clusters and $\mathbf{m}_{x,i}$ is the x-part of the i-th cluster prototype.

The y-part of the closest cluster prototype is the estimator $m(\mathbf{x})$:

$$m(\mathbf{x}) = \mathbf{m}_{y,i^*}. \tag{20}$$

The forecasted y-pattern is the mean of y-patterns forming the nearest cluster. The number of clusters K is predefined or adjusted during the learning phase. If $K = 1$ this method comes down to the k-NN method with $k = 1$ and $v(\mathbf{x}, \mathbf{x}_j) = \text{const}$. In this case the variance of the estimator is the highest and its bias is the lowest. Increasing of K causes the increasing in bias and decreasing in variance. Thus K should be chosen carefully to ensure a compromise between bias and variance.

In the second forecasting procedure (FP2) inspired by [21]

patterns **x** and **y** are grouped independently into K and L clusters, respectively. In the forecasting task for day type s the subset of the learning set is selected containing only these pairs (**x**, **y**) which include y-patterns representing day of type s. Patterns from this subset are grouped and two populations of clusters are created: C_x and C_y represented by the prototypes $\mathbf{m}_x$ and $\mathbf{m}_y$, respectively. After grouping the successive pairs (**x**, **y**) from the learning subset are presented, and the empirical conditional probabilities $P(C_{y,l}|C_{x,k})$ that the forecast pattern **y** belongs to cluster $C_{y,l}$, when the corresponding pattern **x** belongs to cluster $C_{x,k}$, are estimated. In the forecasting phase the x-pattern is assigned to the group $C_{x,i*}$. The forecasted y-pattern paired with it is determined from the prototypes $\mathbf{m}_y$ weighted by the conditional probabilities $P(C_{y,l}|C_{x,i*})$:

$$m(\mathbf{x}) = \frac{\sum_{l=1}^{L} P(C_{y,l} \mid C_{x,i*})\mathbf{m}_{y,l}}{\sum_{l=1}^{L} P(C_{y,l} \mid C_{x,i*})}, \qquad (21)$$

where $\mathbf{m}_{y,l}$ is the prototype of the cluster $C_{y,l}$.
The prototypes of these y-clusters have the largest share in mean (21), which probability of occurrence after observing the cluster $C_{x,i*}$ including the query pattern is the highest. The number of clusters determine the bias and variance of the model as in FP1.

The cluster prototypes determined in FP1 and FP2 are potential construction patterns. Note that in FP2 prototypes $\mathbf{m}_x$ representing periods preceding the forecasted periods and prototypes $\mathbf{m}_y$ representing forecast periods are not paired as it was assumed for patterns **x** and **y** so far, but connected using conditional probabilities.

The forecasting procedures are summarized in Algorithms 1 and 2.

The clustering method applied to FP1 and FP2 should return the prototype vectors **m** which represent groups of patterns in U, X or Y spaces. Many popular clustering methods can be used, e.g. k-means in crisp and fuzzy variants [39], self organizing maps and neural gas [52]. These algorithms belong to the sequential clustering or partitioning ones, where the measure of similarity between patterns and clusters are based on the distance measure $d(\mathbf{x},\mathbf{m})$. The goal is to partition N data points into K disjoint groups so as to minimize the within-cluster sum-of-squares criterion. In the next sections two new methods dedicated to STLF and based on the artificial immune systems are described. These methods use the forecasting error in the grouping phase. This distinguishes these methods from the mentioned above ones, and enables to adjust the prototype positions so as to minimize the forecasting error.

B. Artificial Immune System AIS1

AIS1 was proposed in [53] and operates according to FP1. The concatenated patterns **x** and **y** are represented by antigens (AGs) with epitopes **u**. Antigens are recognized by antibodies (ABs) which play a role of clusters. Epitopes correspond to antibody paratopes (cluster prototypes) which are constructed analogously to the epitopes: $\mathbf{v} = [\mathbf{p}^T\ \mathbf{q}^T]^T$, where $\mathbf{p} \in X = \mathbb{R}^n$ corresponds to x-patterns, and $\mathbf{q} \in Y = \mathbb{R}^n$ corresponds to y-patterns. Unlike epitopes paratopes are modified during training. The paratope and epitope structure in Fig. 3 is shown.

1. Concatenation of the paired x- and y-patterns in the pattern **z**.
2. Grouping of patterns **z**.
3. Presentation of the query pattern **x** and assigning it to the nearest group (19).
4. Reconstruction of the y-pattern paired with the query pattern based on the y-part of the nearest cluster prototype $\mathbf{m}_y$ (20).

Algorithm 1: The first forecasting procedure (FP1).

1. Independent grouping of patterns **x** and **y**.
2. Estimation of the conditional probabilities $P(C_{y,l}|C_{x,k})$.
3. Presentation of the query pattern **x** and assigning it to the nearest group (19).
4. Reconstruction of the y-pattern paired with the query pattern based on the cluster prototypes $\mathbf{m}_y$ and probabilities $P(C_{y,l}|C_{x,k})$ (21).

Algorithm 2: The second forecasting procedure (FP2).

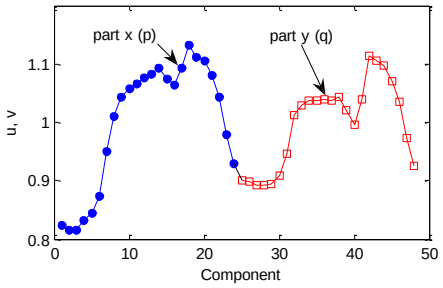

Fig. 3. The antigen and antibody structure in AIS1.

AB have the recognition regions or receptive fields represented by the n-dimensional hyperballs of radius $r > 0$ with centers in the points **p**. The radius r is called the cross-reactivity threshold. It is unchanging, fixed a priori and the same for each AB. The k-th AB can be seen as a pair ($\mathbf{v}_k$, r). The cluster represented by an AB includes those AGs, whose x-epitopes demonstrate the affinity for the p-paratope of this AB. The affinity depends on the distance between vectors **x** and **p** as well as the cross-reactivity threshold.

In the training phase the immune memory is formed (i.e. the population of ABs) which represents a set of clusters covering the population of the learning AGs. The quality of this memory is measured using an average forecast error.

In the forecasting phase an incomplete AG is presented having only the x-epitope (query pattern). It is recognized by a set of ABs from the immune memory which demonstrate the affinity for this AG. We can infer about the y-epitope of this AG on the basis of the q-paratopes of the activated ABs and so reconstruct it.

The steps of AIS1 are presented in Algorithm 3 and described in detail below.

Step 1. The training AG population contains AGs with y-epitopes representing historical daily curves of the same day type as the forecasted day.

Step 2. The AB paratopes are created by copying the epitopes of the training AGs: $\mathbf{v}_k = \mathbf{u}_k$, $k = 1, 2, \ldots, N$. The

starting AB population has the same size as the training AG population. The cross-reactivity threshold r is initiated by a constant.

Step 3 and 5.2. The affinity of the k-th AB for the j-th AG depends on the distance between their paratope and epitope:

$$a(\mathbf{p}_k, \mathbf{x}_j) = \begin{cases} 0, & \text{if } d(\mathbf{p}_k, \mathbf{x}_j) > r \\ 1 - \dfrac{d(\mathbf{p}_k, \mathbf{x}_j)}{r}, & \text{otherwise} \end{cases} \tag{22}$$

where $a(\mathbf{p}_k, \mathbf{x}_i) \in [0, 1]$.
It is assumed that if $a(\mathbf{p}_k, \mathbf{x}_j) > 0$ then the j-th AG is recognized by the k-th AB or the k-th AB is activated by the j-th AG. Affinity $a(\mathbf{p}_k, \mathbf{x}_j)$ informs about the degree of membership of the j-th AG to the cluster represented by the k-th AB. The affinity is maximal when $\mathbf{p}_k = \mathbf{x}_j$.

Step 4 and 5.3. For each AB the set Ψ of AGs lying in its recognition region is determined (i.e. AGs having the nonzero affinity for this AB). For each AG from the set Ψ the forecast of the load curve encoded in the AG y-epitope on the basis of the AB q-paratope is determined and its error is calculated:

$$\delta_{k,j} = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{z_{j+\tau,t} - f_y^{-1}(q_{k,t}, \varphi_j)}{z_{j+\tau,t}} \right| \tag{23}$$

where: $j \in \Psi_k$, Ψ_k is the set of AGs lying in the recognition region of the k-th AB, $z_{j+\tau,t}$ is the t-th time series element in the forecast period $i+\tau$ (load) encoded in the y-epitope of the j-th AG: $y_{j,t} = f_y(z_{j+\tau,t}, \varphi_j)$, $f_y^{-1}(q_{k,t}, \varphi_j)$ is the inverse function for y-patterns (see Section III in [4]) which returns the forecast of $z_{j+\tau,t}$ on the basis of the k-th AB q-paratope and the variables φ determined for the j-th AG epitope.

The evaluation measure of AB is the average forecast error for all AGs lying in its recognition region:

$$\bar{\delta}_k = \frac{1}{|\Psi_k|} \sum_{j \in \Psi_k} \delta_{k,j} \,. \tag{24}$$

Step 5. In the clonal selection loop successive populations of ABs are generated, which forecast the load curves encoded in the AG y-epitopes with the decreasing error. This loop include AB clonning, hypermutation, evaluation and the clonal selection. The stop condition is: there is no decreasing of the average forecast error in S successive iterations.

Step 5.1.1. AB secrets as many clones as many AGs are in its recognition region. Thus in the dense AG clusters more clones are generated.

Step 5.1.2. The goal of the hypermutation is to modificate the AB paratopes to maximize their recognition and forecasting abilities. For a certain parent AB secreting clones the hypermutation results in a shift of each clone towards different AG lying in the receptive field of this AB. The greater error (23) for the j-th AG results in the greater shift of the clone towards this AG. New paratope of the j-th clone generated from the k-th AB after hypermutation is:

```
Training (immune memory creation)
1. Loading of the training population of antigens.
2. Generation of the initial antibody population.
3. Calculation of affinity of antibodies for
   antigens.
4. Evaluation of antibodies.
5. Do until the stop criterion is reached (clonal
   selection loop).
   5.1. Do for each antibody.
        5.1.1. Clonning.
        5.1.2. Clonal hypermutation.
   5.2. Calculation of affinity of clones for
        antigens.
   5.3. Evaluation of clones.
   5.4. Clonal selection.
Test
6. Antigen presentation with x-epitope and
   detection of the activated antibodies.
7. Reconstruction of the y-epitope using q-
   paratopes of the activated antibodies.
```

Algorithm 3: Artificial immune system AIS1.

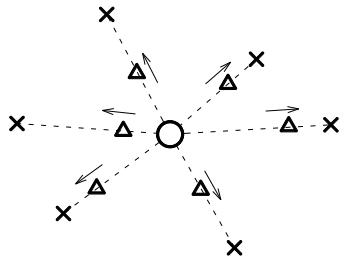

Fig. 4. Hypermutation: shifts of the clones (**Δ**) towards antigens (**×**) lying in the receptive field of the parent antibody (**O**).

$$\mathbf{v}_k^j = \mathbf{v}_k + \eta_k^j (\mathbf{u}_j - \mathbf{v}_k) \,, \tag{25}$$

where: $j \in \Psi_k$, $\mathbf{v}_k^j$ is the paratope of the clone secreted by the k-th AB and shifted towards the j-th AG, $\eta_k^j \in [0, 1)$ is the shift coefficient calculated from the sigmoid function:

$$\eta_k^j = \frac{2}{1 + \exp(-\beta \delta_{k,j} \mid \xi_{k,j} \mid)} - 1 \,, \tag{26}$$

$\beta > 0$ is the slope parameter and $\xi_{k,j} \sim N(1, \sigma)$.

The value of the shift coefficient depends on the error $\delta_{k,j}$ and takes higher values for larger $\delta_{k,j}$ as well as β. The random perturbation $\xi_{k,j}$ with intensity regulated by σ introduces a disturbance of the shift coefficient to prevent stagnation of the learning process due to trapping into the local minima of the error function.

The shifts of clones are illustrated in Fig. 4. The clones are shifted in the paratope space from their initial position (at the parent AB) towards the AGs. As we can see the hypermutation generates new clones inside the region of the cluster

represented by the parent AB. Clones do not exceed the receptive fields of the parent AB.

Step 5.4. For each training AG the set Θ of ABs activated by this AG is determined (this is a subset of the set consisting of the parent ABs and all clones generated from them in the current iteration of the clonal selection loop). The AB with the best score (24) is selected from the set Θ and become one of the parent AB in the next iteration. This clonal selection process is repeated for each AG. The maximum number of ABs in the next population is thus equal to the number of AGs, but the actual number of ABs is usually smaller, since the same AB can be selected by several AGs. The AB number depends on the cross-reactivity threshold *r*. The larger *r* implies less ABs.

Step 6 and 7. ABs contained in the immune memory have paratopes formed during training representing the clusters of AGs in the best manner in terms of the forecasting ability. In the forecast phase the AG is presented having only x-epitope. The set Θ of activated ABs is determined. The q-paratopes of these ABs store information about y-epitopes of the training AGs which x-epitopes were classified to the same clusters. The y-epitope of the query AG is reconstructed from these q-paratopes. The regression function is of the form:

$$m(\mathbf{x}) = \frac{\sum_{k\in\Theta} a(\mathbf{p}_k, \mathbf{x})\mathbf{q}_k}{\sum_{k\in\Theta} a(\mathbf{p}_k, \mathbf{x})}. \quad (27)$$

AB showing a greater affinity for the query AG have a greater impact on the reconstruction of its y-epitope. When AG is not recognized by any AB, it means that the x-epitope represents a new load curve which is dissimilar to those contained in the training set and represented by x-epitopes of AGs.

Discussion. In the immune memory creation process the average forecast error for all ABs is minimized:

$$\frac{1}{K}\sum_{k=1}^{K}\bar{\delta}_k = \\ = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{|\Psi_k|}\sum_{j\in\Psi_k}\frac{100}{n}\sum_{t=1}^{n}\left|\frac{z_{j+\tau,t} - f_y^{-1}(q_{k,t},\varphi_j)}{z_{j+\tau,t}}\right| \to \min. \quad (28)$$

where the number of ABs (*K*) is determined adaptively during training.

Subsequent populations of ABs generated in the clonal selection loop represent the population of AG with lower error. The final population of ABs optimized in this process is the immune memory. This population corresponds to the set of the overlapping clusters in *U* space. These clusters are composed of the spherical subclusters in the subspace *X*, and subclusters in the subspace *Y*. The subcluster size in *X* is limited by the radius *r*. A subcluster in *Y* is understood as a set of y-epitopes of these AGs, which are assigned to the same subcluster in the subspace *X*. This is illustrated in Fig. 5. The immune memory is complete, i.e. it covers all training AGs.

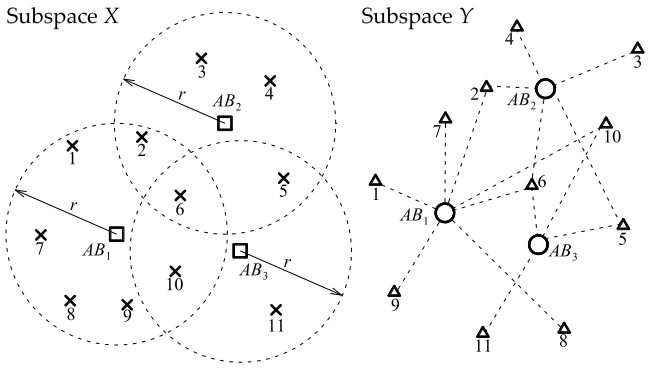

Fig. 5. Subclusters in the subspace *X* represented by the AB recognition regions and corresponding subclusters in subspace *Y*, where: □ are p-paratopes, **O** are q-paratopes, × are x-epitopes and Δ are y-epitopes.

The number of clusters (ABs) results from the cross-reactivity threshold and their compactness in the subspace *Y*. The prototypes of the subclusters in *X* and *Y* are the p-paratope and q-paratope, respectively. These paratopes are shaped simultaneously by the hypermutation operator.

The model parameters are: the cross-reactivity threshold *r*, the slope parameter of the sigmoid function β, the width parameter of the normal distribution σ regulating intensity of the random perturbation of the shift coefficient η, and the number of iterations *S* determining the stop condition of the clonal selection loop.

The large value of the cross-reactivity threshold implies the larger numbers of AGs in the reception fields of ABs and the larger sets Θ of activated ABs. In this case the forecast is calculated by averaging more q-paratopes. This implies an increase in the model bias and decrease in its variance. The model is less sensitive to noise in the training data, but also less accurate. Decreasing of the *r* value has the opposite effect. It also reduces the detection ability of new AGs.

In comparison to the above-mentioned methods of data clustering, AIS1 during clustering uses the forecast errors. This leads to such cluster positions in space that minimize the forecast error. More difficult regions in subspace *X* are covered by more ABs, which allows these regions to be represented more accurately. The number of groups is adaptively adjusted depending on the data arrangement in space, which is an additional advantage.

C. Artificial Immune System AIS2

AIS2 operating according to FP2 includes the immune memory consisting of two populations of ABs. The population of ABs of type x (ABx) recognize AGs representing the x-patterns (AGx), whilst the population of ABs of type y (ABy) recognize AGs representing the y-patterns (AGy). Patterns **x** are the epitopes of AGxs and paratopes of ABxs, and patterns **y** are the epitopes of AGys and paratopes of ABys. Epitopes and paratopes are fixed. ABx has the cross-reactivity threshold *r* defining the recognition region or receptive field (*n*-dimensional hyperball) of radius *r* with center in the point **x**. Similarly, ABy has a receptive field of radius *s* with center in the point **y**. Radii *r* and *s* are adjusted individually during

training, so that AB covers AGs which epitopes are similar to the AB paratope. AG can activate or stimulate many ABs of the same type (x or y). Stimulation occurs when the AG is in the receptive field of AB. The strength of the stimulation (affinity) is dependent on the distance between an epitope and a paratope. AB represents a cluster of similar AGs in the feature space X or Y. The k-th ABx can be seen as a pair ($\mathbf{p}_k$, r_k), where $\mathbf{p}_k = \mathbf{x}_k$ is a paratope recognizing and representing AGx epitopes, and the k-th ABy can be seen as a pair ($\mathbf{q}_k$, s_k), where $\mathbf{q}_k = \mathbf{y}_k$ is a paratope recognizing and representing AGy epitopes. Number of AGs and ABs of both types is equal to the number of learning patterns. Sizes of the recognition regions of ABs depend on the data distribution in the spaces X and Y.

After the two populations of the immune memory have been created, the empirical conditional probabilities $P(ABy_k \mid ABx_j)$, j, $k = 1, 2, \ldots, N$, that the i-th AGy stimulates the k-th ABy, when the corresponding i-th AGx stimulates the j-th ABx, are determined on the training population of AGs. These probabilities are used to determine the forecast pattern $\mathbf{y}$ paired with the query pattern $\mathbf{x}$, as well as the affinities.

The AIS2 is presented in Algorithm 4 and described in detail below. This is a modified version of the artificial immune system for forecasting seasonal time series proposed in [54] and [55].

Step 1. The training AGy population contains AGs representing historical daily curves of the same day type as the forecasted day (y-patterns) and the training AGx population contains AGs representing corresponding daily curves preceding the daily curves encoded in AGy (x-patterns).

Step 2. The paratopes of ABs of both types are created by copying the epitopes of the training AGs: $\mathbf{p}_k = \mathbf{x}_k$, $\mathbf{q}_k = \mathbf{y}_k$, $k = 1, 2, \ldots, N$. The number of AGs and ABs of both types is fixed and is the same as the number of learning patterns N. The cross-reactivity thresholds r and s do not require initialization.

Step 3. The recognition region of the k-th ABx should be as large as possible and cover only the AGxs that satisfy two conditions:

(i) their epitops $\mathbf{x}$ are similar to the paratope $\mathbf{p}_k$, and

(ii) the AGy paired with them have epitopes $\mathbf{y}$ similar to the k-th ABy paratope – $\mathbf{q}_k$.

A measure of similarity between the j-th AGx and the k-th ABx is the distance between their epitope and paratope $d(\mathbf{p}_k, \mathbf{x}_j)$. Similarity between the j-th AGy and k-th ABy, mentioned in (ii) is measured using the forecast error of the daily load curve encoded in the paratope of the k-th ABy. This curve is forecasted using the epitope of the j-th AGy. The forecast error is:

$$\delta_{k,j} = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{z_{k+\tau,t} - f_y^{-1}(y_{j,t},\varphi_k)}{z_{k+\tau,t}}\right|, \tag{29}$$

where: $z_{k+\tau,t}$ is the t-th time series element in the forecast period $i+\tau$ (load) encoded in the paratope of the k-th ABy:

```
Training (immune memory creation)
1. Loading of the training population of x- and y-
   antigens.
2. Generation of the initial x- and y-antibody
   population.
3. Determination of the cross-reactivity thresholds
   of x-antibodies.
4. Determination of the cross-reactivity thresholds
   of y-antibodies.
5. Determination of the empirical conditional
   probabilities P(ABy_k|ABx_j).
Test
6. X-antigen presentation and detection of the
   activated x-antibodies.
7. Reconstruction of the y-antigen epitope using y-
   antibodies, P(ABy_k|ABx_j) and affinities.
```

Algorithm 4: Artificial immune system AIS2.

$q_{k,t} = f_y(z_{k+\tau,t},\varphi_k)$, $f_y^{-1}(y_{j,t},\varphi_k)$ is the inverse function for y-patterns returning the forecast of $z_{k+\tau,t}$ using the epitope of the j-th AGy and the variables φ determined for the k-th ABy epitope.

If the condition $\delta_{k,j} \le \delta$ is satisfied, where δ is the error threshold value, it is assumed that the j-th AGy is similar to the k-th ABy, and it is classified to class 1 as well as the j-th AGx, paired with it. When the above condition is not met the j-th pair (AGx, AGy) is classified to class 2. Class 1 indicates the high similarity between ABy and AGy. The classification procedure is performed for each ABx. As a result, the pairs of AGs are split into two classes for each ABx.

The cross-reactivity threshold of the k-th ABx is defined as follows:

$$r_k = d(\mathbf{p}_k,\mathbf{x}_A) + c[d(\mathbf{p}_k,\mathbf{x}_B) - d(\mathbf{p}_k,\mathbf{x}_A)], \tag{30}$$

where B denotes the nearest AGx of class 2 to the k-th ABx, and A denotes the furthest AGx of class 1 satisfying the condition $d(\mathbf{p}_k, \mathbf{x}_A) < d(\mathbf{p}_k, \mathbf{x}_B)$. The parameter $c \in [0, 1)$ allows to adjust the cross-reactivity threshold value from $r_{k\min} = d(\mathbf{p}_k, \mathbf{x}_A)$ to $r_{k\max} = d(\mathbf{p}_k, \mathbf{x}_B)$.

The reception field of the k-th ABx covers only these AGxs which are located in its geometrical neighborhood and are characterized by similarity of AGys paired with them to the k-th ABy. This is illustrated in Fig. 6, where the reception field of ABx_a covers AGx_a, AGx_b, AGx_c, and AGx_A, because they are near the ABx_a in X space and paired with them AGy_a, AGy_b, AGy_c, and AGy_A are similar to ABy_a. AGx_B is outside the ABx_a reception field because AGy_B is not similar to ABy_a (to big error (29) for AGy_B). AGy_d and AGy_e are also outside the ABx_a reception field because the distance between them and ABx_a is greater than the distance between ABx_a than AGx_B (AGx_B cannot be included in the reception field of ABx_a).

Step 4. The recognition region of the k-th ABy contains AGy which epitopes are similar to the paratope of this ABy. A measure of the similarity is error (29). The classification of the AG pairs carried out in step 3 classify to class 1 those AGys which are similar to the k-th ABy.

The cross-reactivity threshold of the k-th ABy is determined analogously to the threshold of k-th ABx:

$$s_k = d(\mathbf{q}_k, \mathbf{y}_A) + b[d(\mathbf{q}_k, \mathbf{y}_B) - d(\mathbf{q}_k, \mathbf{y}_A)], \quad (31)$$

where B denotes the nearest AGy of class 2 to the k-th ABy, and A denotes the furthest AGy of class 1 satisfying the condition $d(\mathbf{q}_k, \mathbf{y}_A) < d(\mathbf{q}_k, \mathbf{y}_B)$. The parameter $b \in [0, 1)$ has the same function as the parameter c in (30).

In the recognition region of the k-th ABy there are AGys paired with AGxs lying in the recognition region of the k-th ABx, but in this region there can be also AGys paired with AGxs laying outside the recognition region of the k-th ABx if for these AGys the following condition is satisfied: $d(\mathbf{q}_k, \mathbf{y}) < d(\mathbf{q}_k, \mathbf{y}_B)$ (see Fig. 6).

Step 5. When both populations of the immune memory are created, the successive pairs of learning AGs are presented: (AGx_l, AGy_l), $l = 1, 2, \ldots, N$. For each pair the sets of stimulated ABx and ABy are determined and the conditional probabilities are estimated: $P(ABy_j|ABx_i) = L_{j,i}/N$, where $L_{j,i}$ is the number of simultaneous stimulations of the j-th ABy and i-th ABx by the paired AGs.

Step 6 and 7. In the forecast procedure new AGx, representing the query pattern $\mathbf{x}$, is presented to the immune memory. Let Θ be a set of ABx stimulated by this AGx. The forecasted pattern $\mathbf{y}$ corresponding to the query pattern is estimated using regression function:

$$m(\mathbf{x}) = \sum_{j=1}^{N} w_j(\mathbf{x})\mathbf{q}_j, \quad (32)$$

where

$$w_j(\mathbf{x}) = \frac{\sum_{i\in\Theta} P(ABy_j \mid ABx_i)a(\mathbf{p}_i, \mathbf{x})}{\sum_{k=1}^{N}\sum_{i\in\Theta} P(ABy_k \mid ABx_i)a(\mathbf{p}_i, \mathbf{x})}, \quad (33)$$

$a(\mathbf{p}_i, \mathbf{x}) \in [0, 1]$ is the affinity informing about the membership degree of the query AGx to the cluster represented by the i-th ABx defined as:

$$a(\mathbf{p}_i, \mathbf{x}) = \begin{cases} 0, & \text{if } d(\mathbf{p}_i, \mathbf{x}) > r_i \text{ or } r_i = 0 \\ 1 - \dfrac{d(\mathbf{p}_i, \mathbf{x})}{r_i}, & \text{otherwise} \end{cases}. \quad (34)$$

Thus the forecast is the weighted average of ABy paratopes. Weights express the sums of products of affinities of the stimulated ABx for the query AGx and probabilities $P(ABy_j|ABx_i)$.

Discussion. The ABs in AIS2 represent clusters in X and Y spaces. These clusters have a spherical shape, may overlap and are limited by the cross-reactivity thresholds, as in AIS1, but these thresholds are not the same. They are determined individually for each AB. The cross-reactivity thresholds of ABx determining the size of the groups in X, are adjusted to the training data so that the clusters in X correspond to tight clusters in Y. Thresholds of ABy are adjusted so that the ABy receptive field covers a tight cluster in Y. The compactness of this cluster is measured with the forecast error of the load curve encoded in the ABy paratope. This forecast is determined using AGy belonging to the same cluster.

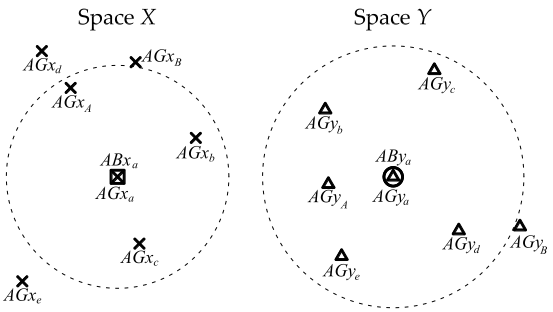

Fig. 6. A cluster in the space X represented by the ABx recognition region and corresponding cluster in the space Y represented by the ABy recognition region, where: □ is the paratop **p**, O is the paratope **q**, × are epitopes **x** and Δ are epitopes **y**.

The number of groups is equal to the number of learning patterns, and locations of the cluster prototypes in X and Y spaces (x- and y-paratopes) are fixed and the same as the locations of training patterns **x** and **y**, respectively. Each AGx (AGy) from the training set is covered by at least one ABx (ABy). AGx located in dense clusters may be covered by many ABx, especially when paired with them AGy are not outliers. AGx paired with an outlier AGy is covered by only one ABx, specialized to recognize this AGx.

The way of forming clusters in X makes their sizes dependent on the dispersion of y-patterns paired with x-patterns belonging to these clusters. To the cluster represented by ABx_k the pattern $\mathbf{x}_j$ is added (increasing the cross-reactivity threshold r_k), if the pattern $\mathbf{y}_j$ is sufficiently similar to the paratope of ABy_k. The pattern is sufficiently similar if it can forecast the paratope $\mathbf{q}_k$ with an error not greater than the threshold value δ. Such a clustering procedure ensures that the forecast error for the training patterns is not greater than the threshold error.

The relationship between ABx and ABy are expressed in probabilities $P(ABy_j|ABx_i)$. The regression function is defined using these probabilities and affinities ABx for AGx. The share of the paratope $\mathbf{q}_j$ in forming of the regression curve depends on the similarity between paratope $\mathbf{p}_j$ paired with $\mathbf{q}_j$ and the query pattern $\mathbf{x}$. This paratope $\mathbf{q}_j$ has the larger share which is paired with ABx showing greater affinity for the query AGx, and for which a greater probability of simultaneous occurrence of the cluster they represent and the clusters to which the query AGx was assigned is observed.

The learning procedure is deterministic and requires only one pass of the training data. The deterministic nature of the model means stable responses and short training time. AIS2 has three parameters: threshold error δ and parameters b and c adjusting the cross-reactivity thresholds. Increasing the values of these parameters implies an increase in cluster sizes. This gives higher bias and smaller variance of the model. During the forming of clusters the information about the forecast error

is used which distinguishes this forecasting model from others based on the classical clustering methods.

VI. Evaluation of PSBFMs

In this section we illustrate the proposed PSBFMs on examples and we analyze their performance and features. In the first example we train and optimize models for tasks of hourly load forecasting with one day horizon for the Polish power system. Then we study the sensitivities of the models to changes in parameter values and model robustness to noisy and missing data. We compare our models with other popular STLF models such as: ARIMA, exponential smoothing and neural network in the forecasting tasks on several load time series and forecasting horizons up to 7 days. Finally the computational complexity analysis of PSBFMs is carried out.

In these studies we use X3.1 and Y3.1 pattern definitions (see Section III in [4]) and Euclidean metric as a measure of distance between patterns.

A. Training and Optimization of PSBFMs

The task is to forecast the hourly load of the Polish power system at hour t = 1, 6, 12, 18, 24 for the next day (τ = 1). We use the N-WE in MISO version and other PSBFMs described above in MIMO versions. The time series is from the period 2002-2004 (see Fig. 1 in [4]; these data can be downloaded from the website http://gdudek.el.pcz.pl/varia/stlf-data). The test samples are from January 2004 (without untypical 1 January) and July 2004.

In the N-WE the bandwidth values were estimated using Scott's rule (10), GM (11), EA and TSc. The parameters of these methods were:

- GM: $a_0 = 0.1$, $\Delta = 0.05$, $L = 20$,
- EA: population size = 30, number of generations M = 100, tournament size T_s = 2, crossover probability = 0.9, mutation probability of the individual = 1, $w_\sigma = 0.1$,
- TSc: number of iterations M = 100, l = 30, $w_\sigma = 0.1$.

These parameters were adjusted in the preliminary tests. The stop criterion in EA and TSc was: there is no improvement in results in 0.25M successive iterations.

In Table I errors for validation (global LOO) and test samples are presented. The optimization of the bandwidths results in the validation error reduction but it did not bring the expected effect on the test samples. Using the local versions of LOO: LOO-v1 and LOO-v2, we did not improve results on the test samples as well.

In the next experiment we select the components of the x-patterns using SFS, SBS, GA and TSb. The parameters of GA and TSb determined on the basis of preliminary tests were:

- GA: population size = 8, number of generations M = 100, tournament size T_s = 2, crossover probability = 0.9, mutation probability = 0.05,
- TSb: number of iterations M = 100, l = 8.

The bandwidths were determined using the Scott's rule. The errors in Table II are presented. Fig. 7 shows how often the components were selected as inputs when using different feature selection methods.

TABLE I
Forecast Errors for N-WE Model using Different Methods of Estimation of Bandwidths

Method	January		July		Average	
	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$
Scott's rule	1.62	1.20	1.54	0.92	1.58	1.05
GM	1.58	1.21	1.51	0.96	1.55	1.09
EA	1.32	1.36	1.28	0.90	1.30	1.13
TSc	1.30	1.23	1.25	0.93	1.28	1.08

TABLE II
Forecast Errors for N-WE Model using Different Methods of X Component Selection

Method	January		July		Average	
	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$
SFS	1.37	1.25	1.32	0.90	1.34	1.07
SBS	1.37	1.20	1.35	0.90	1.36	1.05
GA	1.38	1.17	1.34	0.90	1.36	1.03
TSb	1.34	1.17	1.30	0.90	1.32	1.03
TScb	1.25	1.20	1.21	0.86	1.23	1.03

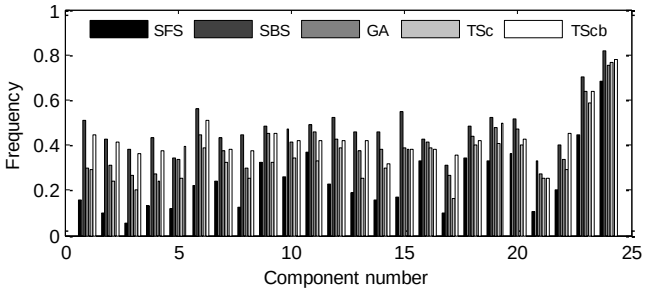

Fig. 7. The frequencies of the component selection in the N-WE model.

The results of the combined optimization of the bandwidth values and selection of the x-pattern components using TScb are shown in Table II and Fig. 7.

From Table II it can be seen that the validation error was reduced compared to the case without selection but the test errors are statistically indistinguishable (Wilcoxon signed-rank test was used). This can be caused by the insufficient information about the target function included in the learning sample. Note that the number of the learning points is only about one hundred and their size is up to 24, so they are sparsely distributed in the space. In addition, points are distorted by noise. Thus the target function in the neighborhoods of the test points is poorly represented by the learning points.

The average reduction in the number of components was as follows: for SFS - 76%, for SBS - 52% for GA - 60%, for TSb - 67% and for TScb - 57%. This means that rejecting more than half of the x-pattern components should not adversely affect the accuracy of the model. The most information about the forecast is included in the last components, i.e. the system loads at hours 23 and 24 (see Fig. 7).

In conclusion it should be noted that the most accurate model based on the Nadaraya-Watson estimator was obtained when the smoothing parameters were calculated using Scott's rule. Any attempt to optimize the model for the analyzed forecasting tasks did not bring a statistically significant improvement of the accuracy on the test sample.

In the *k*-NN model the number of nearest neighbors *k*, and parameters p and γ were optimized using the grid search method. It was assumed: $k = 1, 2, \ldots, 50$, $p = 0, 0.25, \ldots, 1$ and $\gamma \in \{0, -0.8, 5\}$. The model performance was evaluated in the global LOO procedure. The validation errors reached lower values for higher values of p. The validation MAPE at $p = 1$ were: 1.58 for $\gamma = 0$ (the linear model), 1.60 for $\gamma = -0, 8$, and 1.57 for $\gamma = 5$. When using the same nonzero weights for each construction pattern in (15) the validation MAPE was 1.65. When $p = 1$ and $\gamma = 0$ the weighting function is linear of the form: $v(\mathbf{x},\mathbf{x}_j) = 1 - d(\mathbf{x}, \mathbf{x}_j)/d(\mathbf{x}, \mathbf{x}^k)$. In Table III the errors for this model are presented. The optimal values of the nearest neighbors ranged from 4 to 17.

The extensive studies of FNM reported in [51] showed that this model is not very sensitive to parameter α in (17) and (18). So in our study it was assumed $\alpha = 2$. The parameter σ was changed according to the schemes:

(i) $\sigma = b \cdot d_{med}$, where d_{med} is a median of distances between x-patterns in the training set, $b = 0.02, 0.04, \ldots, 1$,

(ii) $\sigma = d(\mathbf{x}, \mathbf{x}^k)$, $k = 1, 2, \ldots, 50$.

The lower errors were achieved when we used scheme (i). In all forecasting tasks the Cauchy function (18) gave higher errors than the Gaussian function (17). The average validation errors were: 1.55 for the Gaussian function and variant (i), 2.04 for the Cauchy function and variant (i), 1.63 for the Gaussian function and variant (ii), and 2.15 for the Cauchy function and variant (ii). The optimal value of σ in the model with the Gaussian function does not exceed $0.32d_{med}$ at its modal value of $0.20d_{med}$. The errors for FNM with the Gaussian membership function and σ determined using (i) in Table III are shown. The optimal σ values ranged from 0.080 to 0.161.

From Table III it can be seen that FNM outperforms *k*-NN model. In further studies we simultaneously select components of **x** and optimize the width parameter σ in FNM using a combination of the feature selection algorithm and the grid search (GS). As a feature selection algorithm SFS, SBS and TSb are applied. The tournament size in TSb $l = 8$ and the number of iterations $M = 100$. The errors in Table IV are shown. As can be seen the validation and test errors were reduced, but the test error reduction is statistically indistinguishable (Wilcoxon test was used). The selected components of x-patterns for each forecasting task in Fig. 8 are presented. The average number of x-pattern components was reduced as follows: SFS+GS – 66%, SBS+GS – 49% and TSb+GS – 59%. Components 18, 23 and 24 were most often selected.

The forecasting models based on the clustering methods were examined under the grant [56]. As the clustering methods *k*-means in crisp and fuzzy variants, agglomerative hierarchical clustering, self organizing maps and neural gas were used. The models were tested on 8 electric load time series and 4 energy price time series. The forecasting model using FP2 and the crisp *k*-means clustering turned out to be the best one. So we limit further studies to the models based on the crisp *k*-means. The FP1 and FP2 are both examined.

TABLE III
FORECAST ERRORS AND THEIR INTERQUARTILE RANGES FOR *K*-NN AND FNM

Method	January		July		Average	
	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}
k-NN	1.47	1.12	0.99	1.01	1.23	1.14
FNM	1.22	1.30	0.96	0.89	1.08	1.06

TABLE IV
FORECAST ERRORS FOR FNM USING COMBINED METHODS OF COMPONENT SELECTION AND ESTIMATION OF σ

Method	January		July		Average	
	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$
SFS + GS	1.46	1.13	1.42	0.98	1.44	1.05
SBS + GS	1.44	1.17	1.41	0.92	1.42	1.04
TSb + GS	1.44	1.19	1.41	0.93	1.42	1.06

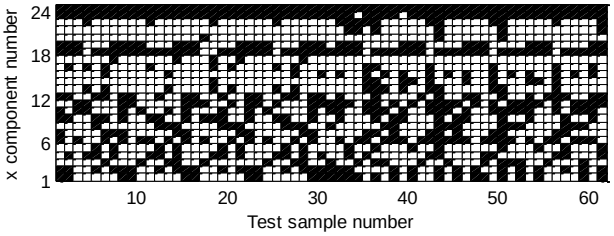

Fig. 8. The components of x-patterns selected using combined algorithm TSb + GS for FNM (black elements).

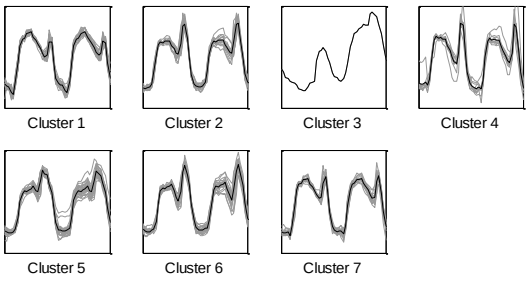

Fig. 9. The clusters created by FP1+*k*-means model for the forecasting task of July 1, 2004 (black lines are the cluster prototypes, gray lines are the patterns **u** assigned to the clusters).

In the model using FP1 and *k*-means (FP1+*k*-means) the only parameter is the number of groups *K*. Each group is represented by a prototype vector **m**, which is a mean of u-patterns belonging to this group. The *K* value was changed from 1 to 40. Models were evaluated in LOO-v1 procedure. The errors for optimal values of *K* in Table V are shown. The clusters created for the forecasting task of July 1, 2004 in Fig. 9 are shown. The query pattern in this case was assigned to cluster 1 (Fig. 10). Cluster 3 includes only one outlier pattern representing the daily curves of 1 and 2 January 2003.

A similar test procedure was carried out for FP2 and *k*-means (FP2+*k*-means). The number of clusters *K* and *L* were changed from 1 to 40 and the GS was used to find their optimal values using LOO-v1 procedure to model evaluation. Errors in Table V are shown. For the forecasting task of July 1, 2004 the lowest errors were achieved for $K = 25$ and $L = 29$. The clusters created for this task in Fig. 11 are presented, the

TABLE V
FORECAST ERRORS AND THEIR INTERQUARTILE RANGES FOR *K*-MEANS-BASED MODELS

Method	January		July		Average	
	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}
FP1+*k*-means	1.85	1.37	1.06	0.89	1.45	1.11
FP2+*k*-means	1.52	1.11	1.07	0.88	1.29	0.95

TABLE VI
FORECAST ERRORS AND THEIR INTERQUARTILE RANGES FOR ARTIFICIAL IMMUNE SYSTEMS-BASED MODELS

Method	January		July		Average	
	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}
AIS1	1.40	1.27	0.99	0.97	1.19	1.06
AIS2	1.32	1.39	1.01	0.81	1.16	1.11

validation errors depending on K and L in Fig. 12a are presented and the probabilities $P(C_{y,l}|C_{x,k})$ in Fig. 12b are presented. The query pattern was assigned to cluster C_x number 20 and the forecast pattern **y** was reconstructed from the prototypes $\mathbf{m}_y$ of clysters 1, 18, 19 and 22. The probabilities $P(C_{y,l}|C_{x,k=20})$ for these clusters were: 1/*N*, 4/*N*, 2/*N* and 1/*N*, respectively. The forecasted y-pattern in Fig. 13 is shown.

In the first stage of AIS1 optimization the cross-reactivity threshold was changed according to $r = \Delta d$, where d is the average distance between each training AG and AB from the initial population and $\Delta = 0.1, 0.15, \ldots, 1.0$. Other parameters were kept constant: $\beta = 0.2$, $\sigma = 0.1$ and $S = 10$. In the case when the query AG was not recognized by any AB, the nearest AB was selected and its q-paratope was taken as the predicted y-epitope of the query AG. The lowest validation errors $MAPE_{val} = 1.27$ were obtained for $\Delta = 0.3$. In the next stages other parameters were changed according to the schemes:

(i) $\beta = 0.05, 0.10, \ldots, 1.00$, at $\Delta = 0.3$ and $\sigma = 0.1$,

(ii) $\sigma = 0.025, 0.050, \ldots, 0.200$, at $\Delta = 0{,}3$ and $\beta = 0{,}2$.

Using schemes (i) and (ii) it was observed that the AIS1 model showed low sensitivity to changes in parameters β and σ. The validation error in these cases varied between 1.24 and 1.27. It was assumed that the best values of parameters are: $\Delta = 0.3$, $\beta = 0.2$, $\sigma = 0.1$ and $S = 10$. Errors for this parameter values in Table VI are presented. The size of the immune memory (number of clusters) was changing in 30 training sessions from 52 to 84 and the number of iterations of the clonal selection loop varied from 16 to 57 (the average value was 27). In Fig. 14 the set Θ of activated ABs in the forecasting task of July 1, 2004 and the reconstructed forecast pattern are shown.

In the first stage of AIS2 optimization we change $\delta = 1.00, 1.25, \ldots, 3.00$ keeping other parameters constant: $b = c = 1$. At lower values of δ many validation AGs were unrecognized (up to 18%). The value of $\delta = 2.00$ ensures recognition of 98% AGs. Increasing δ above this value leads to an increase in the validation error. In the second stage of the study the values of b and c were reduced ($b = c = 0.8, 0.6, \ldots 0.0$) at a fixed value of $\delta = 2$. The validation error in these cases remained at a similar level, but the number of unrecognized AGs

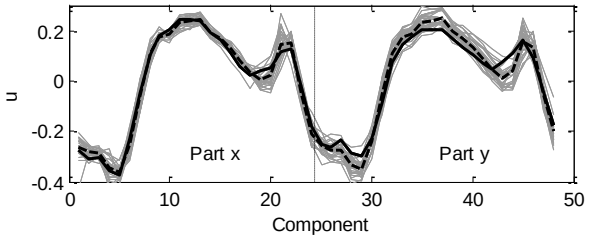

Fig. 10. The test pattern **u** (continuous black line) assigned to cluster 1 (dashed line is the cluster prototype) in the forecasting task of July 1, 2004; FP1+*k*-means model.

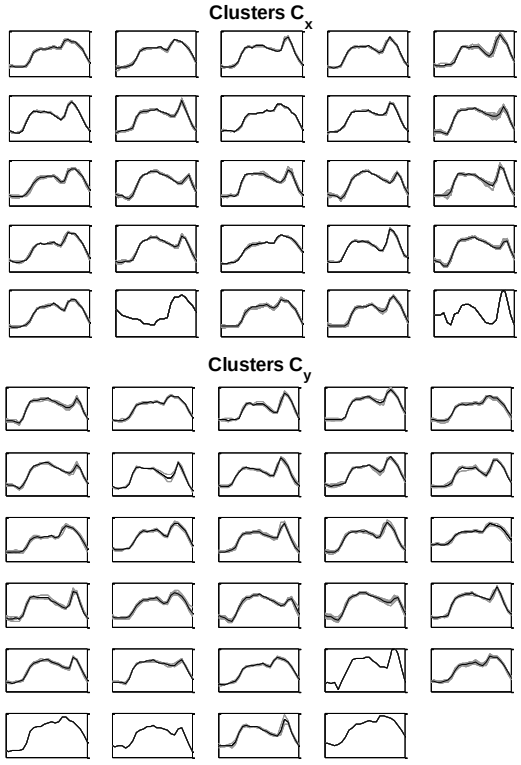

Fig. 11. The clusters created by FP2+*k*-means model for the forecasting task of July 1, 2004.

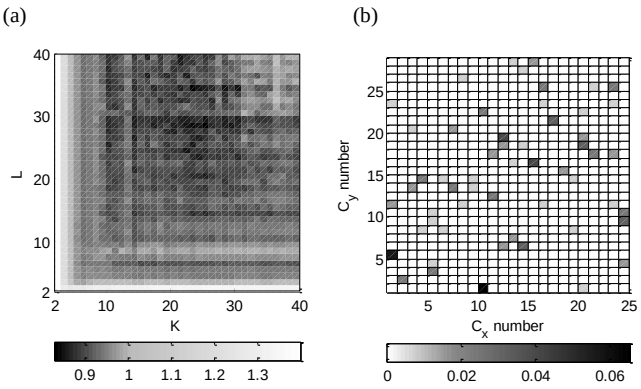

Fig. 12. The validation errors depending on the number of clusters in FP2+*k*-means model (a) and the probabilities $P(C_{y,l}|C_{x,k})$ (b) for the forecasting task of July 1, 2004.

increased. The forecast errors at $\delta = 2$, $b = c = 1$ in Table VI are shown. Fig. 15a shows the empirical conditional probabilities $P(ABy_j|ABx_i)$ estimated on the training set in the forecasting task of July 1, 2004. From this figure it can be seen a specific pattern showing simultaneously activated ABs in both populations of immune memory. These are ABs representing load curves of days lying in the same periods of the years. The weights (33) of activated ABs for this forecasting task in Fig. 15b are shown and in Fig. 16 the activated ABys and the reconstructed y-pattern are presented.

It is worth noting that all clustering methods except AIS2 are stochastic and unstable. They can give different results for the same input.

B. Sensitivity and Robustness of PSBFMs

In this section we analyze the sensitivity of PSBFMs to changes in parameter values and their robustness to noisy and missing data. For nondeterministic models results presented in tables and figures are averaged over 30 training sessions.

The sensitivity measure to changes in parameter value is defined as follows:

$$S_p = \frac{MAPE_{tst\max} - MAPE_{tst\min}}{MAPE_{tst\min}} \cdot 100, \qquad (35)$$

where $MAPE_{tst\min}$ and $MAPE_{tst\max}$ are minimum and maximum test errors, respectively, when the value of the parameter p changes from $0.5p^*$ to $1.5p^*$, p^* is the value of p ensuring the minimal validation error.

Measure (35) informs about the relative percentage difference between the maximum and minimum errors $MAPE_{tst}$ when the parameter varies in a given range. This measure is calculated for each parameter keeping other parameters constant at their optimal values. In Table VII the values of S_p are presented.

In many cases the parameter value ensuring minimum validation error (p^*) is not the same as its value ensuring minimum test error (p'). The differences between test errors for p^* and p': $\Delta MAPE = MAPE_{tst}(p^*) - MAPE_{tst}(p')$ are shown in Table VII. $\Delta MAPE$ shows how much the forecast error increases when we estimate model parameters in the validation procedure like LOO.

The test error reached a minimum for the parameter value estimated in the validation procedure in two cases: for σ in FNM and δ in AIS2. The deviation $\Delta MAPE$ in no case exceed the value of 0.1. The sensitivity of the N-WE model to the width parameter was approximately twice higher than for FNM. The AIS1 model is the most sensitive to the parameter Δ determining the crossreactivity threshold ($S_p = 35.29\%$). The sensitivity of this model to other parameters is low (no more than 3.56%). Also AIS2 is highly sensitive to the crossreactivity threshold. The sensitivity to the number of clusters in the models using k-means is at the level of 11-16%. FNM shows the lowest sensitivity to changes in parameters among the proposed PSBFMs.

The noise in the load data arises from errors of measurements and load estimation. It is assumed that the

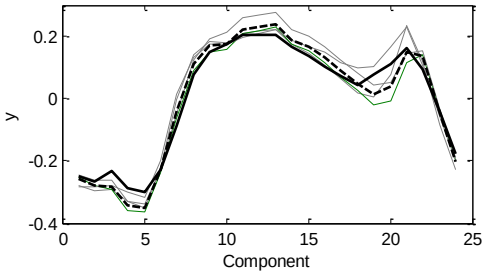

Fig. 13. The prototypes $\mathbf{m}_y$ of clusters of non-zero conditional probabilities $P(C_{y,l}|C_{x,k=20})$ (gray lines), the reconstructed forecast pattern (dashed line) and actual forecast pattern (continuous black line) for the forecasting task of July 1, 2004; FP2+k-means model.

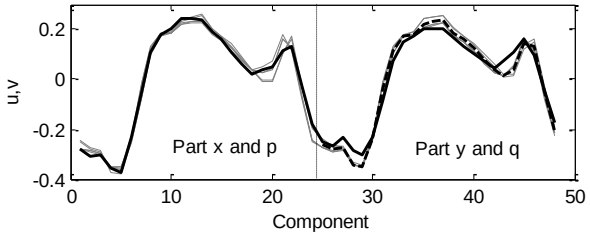

Fig. 14. The test pattern **u** (continuous black line) recognized by a set of ABs (gray lines) and reconstructed y-pattern (dashed line) in the forecasting task of July 1, 2004; AIS 1 model.

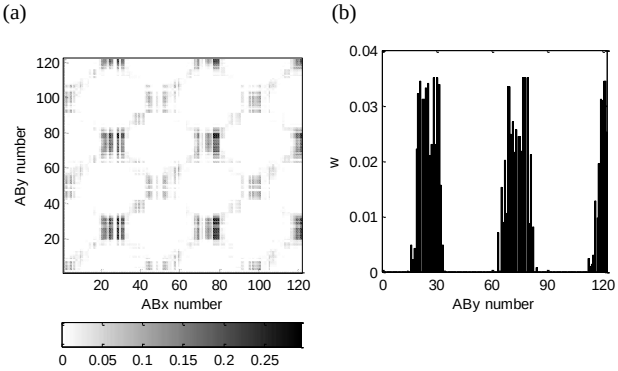

Fig. 15. The probabilities $P(ABy_j|ABx_i)$ (a) and weights of ABy (b) for the forecasting task of July 1, 2004; AIS2 model.

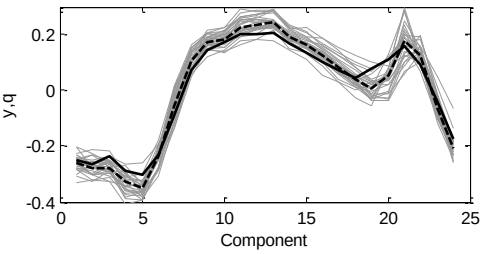

Fig. 16. The activated ABys (gray lines), the reconstructed forecast pattern (dashed line) and actual forecast pattern (continuous black line) for the forecasting task of July 1, 2004; AIS2 model.

components of the load vector **z** are disrupted by noise as follows:

$$z'_{i,t} = z_{i,t}\xi_{i,t}, \tag{36}$$

where $\xi_{i,t} \sim N(1,\sigma)$.

The models were learned and tested using noisy data. The noise intensity was controlled by the standard deviation of the normal distribution: $\sigma = (0, 0.1)$. This correspond to a share of noise in the data ($100|z'-z|/z$) from 0 to 8%. Results in Fig. 17 are shown.

A measure of sensitivity to the noise in data is defined as follows:

$$S_n = \frac{MAPE_{tst}(\sigma) - MAPE_{tst}(\sigma = 0)}{\sigma} \cdot 100, \tag{37}$$

where $MAPE_{tst}(\sigma)$ is the test error observed when $\sigma > 0$.
Measure (37) expresses the ratio of the change in forecast error due to the noisy data to the intensity of the noise. The mean values of S_n for all σ, which corresponds to the slope of the straight line approximating the characteristics presented in Fig. 17, were shown in Table VIII. In this table the increases in test errors $\Delta MAPE$ at $\sigma = 0.01$ and $\sigma = 0.1$ with respect to errors for noiseless data are also presented.

The noisy x-patterns have different positions in space in relation to the original x-patterns. It results in changes in the construction pattern weights in the formulas for the regression function estimators. For the small noise ($\sigma = 0.01$) the forecast errors increase slightly, from 0.12 for FP1+*k*-means to 0.56 for AIS2. Big noise causes a large increase in the forecast error (from 3-4 percentage points for *k*-means models up to 7-8 percentage points for AIS). S_n takes the lowest values for the models based on the *k*-means clustering (26.53% for FP2+*k*-means and 31.15% for FP1+*k*-means), and the largest ones for the models based on the immune systems (66.94% for AIS2 and 74.41% for AIS1).

In the robustness to missing data analysis it is assumed that *m* components the of vector **z*** corresponding to the query pattern **x*** are missing. These components are missing in **x*** as well, and moreover the values of other components can change when the x-pattern is defined using variables φ (see Table I in [4]), which are determined on the basis of many components of vector **z**, among which are the missing ones (e.g. pattern X3.1 is defined using mean value of **z** components). In the same way as for **x*** components of the training x-pattern are determined assuming that the *m* components of the corresponding vectors **z** are undefined. If as a result of the lack of components in **z*** the value of variables φ in function f_y change, the components of y-pattern change as well.

The x-patterns of reduced number of components which values may be different than the original ones are arranged in the space differently relative to each other than the original patterns. This has an impact on the distances between them, and thus, on the weights of construction patterns and

TABLE VII
SENSITIVITY OF PSBFMS TO CHANGES IN PARAMETER VALUES

Model	Parameter	S_p	$\Delta MAPE$
N-WE	h	17.14	<0.01
FNM	σ	8.07	0.00
FP1+*k*-means	K	15.71	0.08
FP2+*k*-means	K	11.56	0.04
	L	12.92	0.05
AIS1	Δ	35.29	0.04
	β	3.56	0.03
	σ	3.09	0.02
	S	3.36	0.02
AIS2	δ	29.23	0.00
	b,c	8.02	0.09

TABLE VIII
SENSITIVITY OF PSBFMS TO NOISY DATA

Model	S_n	$\Delta MAPE$ for $\sigma = 0.01$	$\Delta MAPE$ for $\sigma = 0.1$
N-WE	52.51	0.25	6.14
FNM	41.28	0.15	4.84
FP1+*k*-means	31.15	0.12	4.11
FP2+*k*-means	26.53	0.14	3.14
AIS1	74.41	0.56	7.92
AIS2	66.94	0.17	7.59

TABLE IX
SENSITIVITY OF PSBFMS TO MISSING DATA

Model	S_m for $m = 6$	S_m for $m = 12$	$\Delta MAPE$ for $m = 6$	$\Delta MAPE$ for $m = 12$
N-WE	10.98	28.01	0.03	0.14
FNM	6.97	17.27	0.02	0.09
FP1+*k*-means	8.42	14.19	0.02	0.07
FP2+*k*-means	9.20	18.61	0.02	0.09
AIS1	15.75	18.14	0.04	0.09
AIS2	16.42	22.20	0.04	0.11

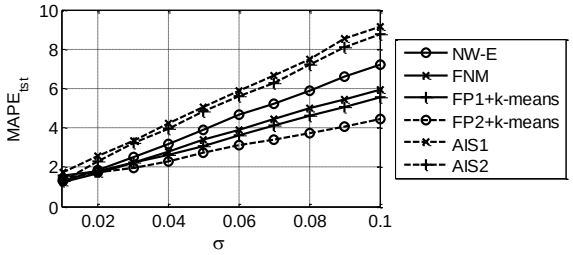

Fig. 17. The forecast errors for noisy data depending on the deviation σ.

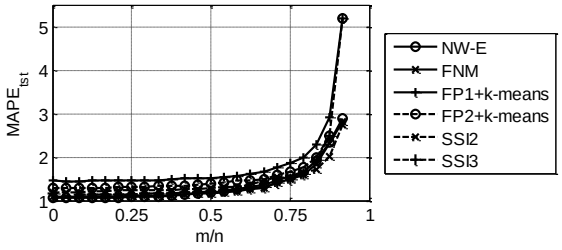

Fig. 18. The forecast errors depending on the relative number of missing components of x-patterns.

consequently on the forecast.

To examine the robustness of PSBFMs to missing data we remove m components of the vectors $\mathbf{z}^*$ and then we redefine training patterns $\mathbf{x}$ and $\mathbf{y}$. The models are learned and tested on the modified data. We change the number of missing components m = 1, 2, ..., 22. Their positions were determined by random. The forecast errors depending on the relative number of missing components in Fig. 18 are shown. Note that when the number of missing components is low the deterioration in the model accuracy is not observed. Errors begin to grow rapidly when m exceeds 16.

A measure of the model sensitivity to the missing components is defined as follows:

$$S_m = \frac{MAPE_{tst}(m) - MAPE_{tst}(m=0)}{m/n} \cdot 100, \qquad (38)$$

where $MAPE_{tst}(m)$ is the average test error observed at m missing components.

The values of S_m and the increases in test errors with respect to errors for noiseless data $\Delta MAPE$ at m = 6 and m = 12 in Table IX are presented. From this table it can be seen that the least sensitive models are: FNM and k-means based models, and the most sensitive models are AISs. The deterioration of the test error for m = 6 is not greater than 0.04 percentage points and for m = 12 is not greater than 0.14 percentage points.

It is noteworthy that in many models (e.g. ARIMA, exponential smoothing, neural networks) the incomplete input information is a serious problem, and the missing data reconstruction is needed. PSBFMs successfully deal with missing data because the similarity measure, on the basis of which the weights of input patterns in the nonparametric regression are determined, can be calculated without some components of these patterns.

C. Comparative studies of PSBFMs with other models

We compare our PSBFMs with other popular models of STLF: ARIMA, exponential smoothing (ES) and MLP. The models were tested in STLF problems on four time series of electrical load:

- PL: time series of the hourly load of the Polish power system from the period of 2002–2004 (this time series was used in the experiments described above). The test sample includes data from 2004 with the exception of 13 untypical days (e.g. holidays),
- FR: time series of the half-hourly load of the French power system from the period of 2007–2009. The test sample includes data from 2009 except for 21 untypical days,
- GB: time series of the half-hourly load of the British power system from the period of 2007–2009. The test sample includes data from 2009 except for 18 untypical days,
- VC: time series of the half-hourly load of the power system of Victoria, Australia, from the period of 2006–2008. The test sample includes data from 2008 except for 12 untypical days.

TABLE X
BASIC PARAMETERS OF THE LOAD TIME SERIES

Time series	$\bar{z}$	c_d	c_w	c_a	χ^2	V	ρ	δ_n
PL	16.05	8.08	7.70	10.87	30967	0.42	0.67	3.43
FR	55.64	9.05	7.38	16.73	20905	0.35	0.63	5.05
GB	37.45	16.49	6.93	10.40	37074	0.46	0.86	3.52
VC	5.96	10.34	7.07	5.32	16367	0.30	0.49	4.88

These load time series are characterized in Table X, where:

- $\bar{z}$ is the mean load of the power system in GW,
- c_d, c_w and c_a are the daily, weekly and annual variation coefficients, respectively (see Section V in [4]),
- χ^2, V, ρ are the chi-square value, Cramer's contingency coefficient and Pearson's correlation coefficient, respectively, determined on the basis of the contingency table for $d(\mathbf{x}_i, \mathbf{x}_j)$ and $d(\mathbf{y}_i, \mathbf{y}_j)$, where i and j are indices of patterns representing the same type of the day of a week (see Section VI in [4]),
- δ_n is the forecast error (MAPE) for the naïve method, where the forecast rule is of the form: the forecasted daily curve is the same as seven days ago.

From Table X it can be seen that the biggest daily variation of load is for GB data and the biggest annual variation is for FR data. The value of χ^2 is above of its critical value (66.34 at α = 0.05) for each time series. This confirms a relationship between random variables $d(\mathbf{x}_i, \mathbf{x}_j)$ and $d(\mathbf{y}_i, \mathbf{y}_j)$ and justifies using PSBFMs. The values of V and ρ indicate significant, moderately strong and positive correlation between random variables, which is the strongest for GB data.

To simplify the forecasting problem for ARIMA and ES the time series were decomposed into n series, i.e. for each t a separate series was created. In this way a daily seasonality was removed. For the independent modeling of these series ARIMA$(p, d, q)\times(P, D, Q)_m$ model was used. To find the best ARIMA model for each time series we use a step-wise procedure for traversing the model space which is implemented in the **forecast** package for the **R** system for statistical computing [57]. This automatic procedure returns the model with the lowest Akaike Information Criterion (AIC) value.

The ES state space models [58] are classified into 30 types depending on how the seasonal, trend and error components are taken into account. These components can be expressed additively or multiplicatively, and the trend can be damped or not. The time series were modeled independently using an automated procedure implemented in the **forecast** package for the **R** system [57]. In this procedure the initial states of the level, growth and seasonal components are estimated as well as the smoothing and damping parameters. AIC was used for selecting the best model for a given time series.

ARIMA and ES parameters were estimated using 12-week time series fragments immediately preceding the forecasted daily period. Untypical days in these fragments were replaced with the days from the previous weeks.

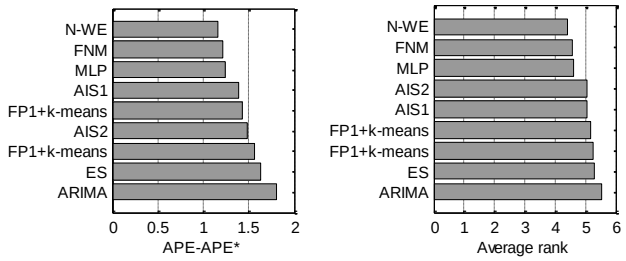

Fig. 19. Rankings of the forecasting models.

TABLE XI
FORECAST ERRORS AND THEIR INTERQUARTILE RANGES FOR THE EXAMINED MODELS

Model	PL		FR		GB		VC	
	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}	$MAPE_{tst}$	IQR_{tst}
N-WE	1.30	1.30	1.66	1.67	1.55	1.63	2.82	2.56
FNM	1.38	1.38	1.67	1.71	1.60	1.66	2.91	2.67
FP1+*k*-means	1.69	1.64	2.05	2.17	1.84	1.88	3.34	3.01
FP2+*k*-means	1.59	1.51	1.94	2.05	1.76	1.84	3.13	2.94
AIS1	1.50	1.50	1.93	1.95	1.77	1.84	3.04	2.75
AIS2	1.50	1.51	1.93	1.96	1.78	1.87	3.33	2.93
ARIMA	1.82	1.71	2.32	2.53	2.02	2.07	3.67	3.42
ES	1.66	1.57	2.10	2.29	1.85	1.84	3.52	3.35
MLP	1.44	1.41	1.64	1.70	1.65	1.70	2.92	2.69

The third reference model is built on MLP using patterns. The network learns the target function g_t mapping the input patterns **x** into the y-pattern components (MISO model). The MLP learns locally [59] using the training sample selected from the neighborhood of the query pattern **x***. By the neighborhood of **x*** we mean the set of its k nearest neighbors representing the same day of a week (k was assumed to be 12). For each forecasting task (forecast of system load at time t of day i) a separate MLP is learned. To prevent overfitting MLP is learned using Levenberg-Marquardt algorithm with Bayesian regularization [60]. Since the target function is modeled locally, using a small number of learning patterns, rather a simple form of this function should be expected, which implies small number of neurons. Based on the research reported in [59] the network composed of only one neuron with bipolar sigmoid activation function was chosen as an optimal architecture.

In Table XI errors for one day ahead forecasts are presented. All PSBFMs were used in the MIMO versions and were optimized and learned for each forecasting task. The lowest errors for all time series were achieved by N-WE, FNM and MLP. The Wilcoxon rank sum test with 5% significance level indicates the statistically significant difference between errors for these three models and other ones. The rankings of the forecasting models based on the average difference between model error (APE) and the smallest error (APE*) for the test sample and based on the average rank in the accuracy ranking for each test sample in Fig. 19 are shown. Note that the classical forecasting tools, ARIMA and ES occupy the last positions in both rankings.

The error distributions are illustrated in Fig. 20. In Fig. 21 the forecast errors are shown for horizons up to seven days ($\tau = 7$). Among PSBFMs the lowest errors for longer horizons are observed when using N-WE. In one day ahead STLF

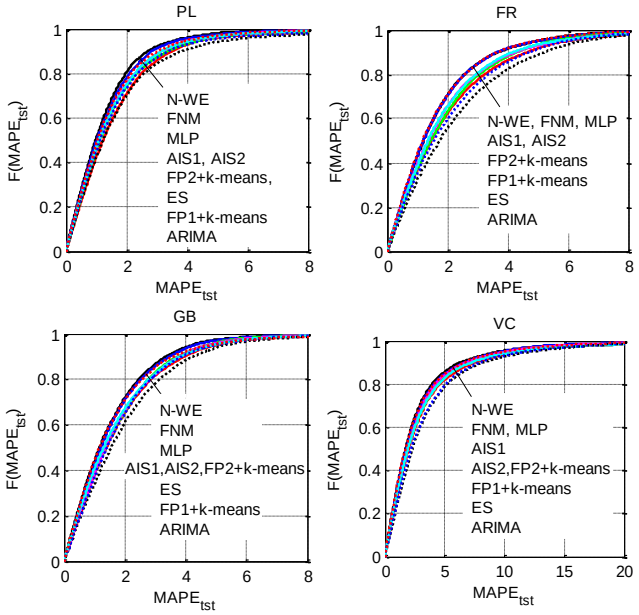

Fig. 20. Cumulative distribution functions of errors.

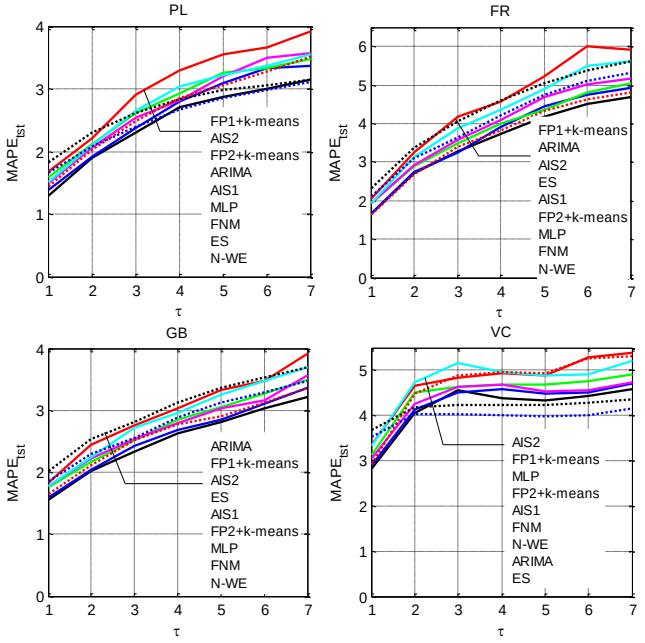

Fig. 21. Errors for different forecast horizons.

ARIMA and ES gave higher errors compared to other models, but they become more competitive for $\tau > 1$. For longer horizons errors achieved by the examined models approach errors achieved by the naive method.

Note that in the case of ARIMA and ES the model parameters are estimated on the basis of the time series fragment (12 weeks in our examples) directly preceding the forecasted fragment. In the case of PSBFMs it is assumed that the information about the forecast can be included in the historical data from a longer period. The construction patterns representing the daily cycles are selected from historical data using criterion based on the similarity to the query pattern. The regression function is constructed locally using these patterns. Similarly in the case of MLP locally learned on

TABLE XII
RUNTIME COMPLEXITY OF PSBFM ALGORITHMS

Model	Training	Test	Optimization
N-WE	-	$O(Nn)$	$O(NnMP_a)$
FNM	-	$O(Nn)$	$O(NnMP_\sigma)$
FP1+k-means	$O(NnKI)$	$O(nK)$	$O(Nn\sum_{i=1}^{P_K} K_i \sum_{j=1}^{M} I_{i,j})$
FP2+k-means	$O(Nn(KI_K+LI_L))$	$O(n(K+L))$	$O(Nn(\sum_{i=1}^{P_K} K_i \sum_{k=1}^{M} I_{Ki,k} + \sum_{i=1}^{P_L} L_i \sum_{k=1}^{M} I_{Li,k}) + M(NP_KP_L+n(\sum_{i=1}^{P_K} K_i + P_K \sum_{j=1}^{P_L} L_j))$
AIS1	$O(N^2n+Nn\sum_{i=1}^{I}\sum_{j=1}^{J_i} Z_{i,j}))$	$O(Vn)$	$O(N^2TMP_r + NT\sum_{o=1}^{M}(\sum_{m=1}^{P_r}\sum_{i=1}^{I_{m,o}}\sum_{j=1}^{J_{i,m,o}} Z_{i,j,m,o} + \sum_{l=1}^{P_\beta}\sum_{i=1}^{I_{l,o}}\sum_{j=1}^{J_{i,l,o}} Z_{i,j,l,o} + \sum_{k=1}^{P_\sigma}\sum_{i=1}^{I_{k,o}}\sum_{j=1}^{J_{i,k,o}} Z_{i,j,k,o}))$
AIS2	$O(N^2n+N^3)$	$O(N^2n)$	$O(M(P_\delta+P_b)(N^3+N^2n))$

patterns the training data are selected using criterion of similarity to the query pattern. Thus the construction or learning data are selected depending on the current input from its neighborhood.

D. Complexity analysis of PSBFMs

The runtime complexity of the PSBFM algorithms in Table XII is presented. We analyzed the complexity of training, test and optimization procedures. The validation procedure is LOO method with M validation samples (in the global LOO $M = N$). In the case of NW-E it is assumed that the model is optimized using GM. The grid search is also used in FNM to estimate the width parameter σ assuming a constant value of α. The AIS1 parameters were estimates according the scheme described in Section VIA: sequentially executed procedures for estimation of individual parameters with the other parameters constant. Similar sequential procedure was applied to the AIS2 parameter estimation δ and $b = c$. In the grid or sequential search the model is trained for P_x values of the x parameter. Thus the optimization complexity of algorithms is dependent on P_x.

In the stochastic models: FP1+k-means, FP2+k-means and AIS1 the time complexity is dependent on the unknown a priori factors: the number of loop iterations (I and J), the number of clones generated (Z) and the immune memory size (V). These factors are a function of the sample distribution in the space and the values of the algorithm parameters.

The complexity of the test procedures is small compared with the complexity of the training and optimization procedures. Test runtime depends linearly on the pattern size n and the number of learning samples N (N-WE, FNM), the number of clusters K and L (k-means based models) or the size of immune memory V (AIS1). The AIS2 test time is proportional to the square of the number of samples. In the deterministic algorithms N-WE and FNM the training phase is skipped. The optimization procedures for these models are considerably less time-consuming than for other ones. It is due to lack of the training, only one parameter to estimation in FNM and the simple one-dimensional optimization problem in N-WE (see (11)). These algorithms should be considered as the least complex ones among the proposed PSBFMs.

E. Best model selection

A method of model selection should balance goodness of fit with simplicity. Goodness of fit relates to the model prediction capability on independent test data. Usually to measure the goodness of fit mean squared error is used but we use MAPE because it is traditionally used in STLF literature and in practice. IQRs of MAPE presented in the above tables inform about dispersion of errors. Smaller IQR indicates more precise models (errors are centered around their mean value). The simplicity of the model is also important because more complex models tend to overfitting (the variance of the model is higher). The model complexity is generally measured by counting the number of parameters in the model. Some criteria of model selection, such as Akaike or Bayesian information criteria, deal with the trade-off between the goodness of fit and the model complexity.

The above mentioned criteria of model selection can be extended with additional ones, e.g.:

- sensitivity to changes in parameter values,
- robustness to noisy data,
- robustness to missing data,
- runtime complexity,
- clear structure which is understandable for people.

For the user of the forecasting model as important as the forecast accuracy is the clear structure of the model enables us to understand its principle of operation. This translates into a greater confidence in the forecast. Many popular STLF models have uninterpretable parameters and are too complex to understand. This applies to both classical models (e.g. ARIMA, exponential smoothing) and unconventional ones (e.g. neural networks, SVM). The proposed PSBFMs: NW-E, FNM, k-NN and k-means seem to be free of this drawback.

Among the proposed PSBFMs the best performance and properties have N-WE and FNM. These two models are the most accurate (see Fig. 19), the simplest and clear having the understandable principles of operation. The FNM has only one parameter to estimate, σ. N-WE has n parameters, h_t, but they are easy to estimate in the one-dimensional grid search. Small number of parameters results in the lowest runtime complexity of these both models (see Table XII).

VII. CONCLUSIONS AND FURTHER WORK

Due to the great importance of STLF in the daily operation of power systems and energy markets a variety of methods have been developed for this forecasting problem. In this work

similarity-based methods using patterns of seasonal cycles for STLF are described. The patterns enable us to simplify the problem of forecasting non-stationary time series with multiple seasonal cycles and trend. PSBFMs construct the regression curve aggregating the forecast patterns from the history with weights dependent on the similarity between input patterns paired with the forecast patterns. PSBFMs are characterized by simplicity. The number of parameters here is small, which implies a simple procedure of model optimization. Models with fewer number of parameters have better generalization properties.

PSBFMs can predict individual y-pattern components or an entire vector **y** (MISO or MIMO models, respectively). In the latter approach, the y-pattern may have any number of components, depending on the discretization of the time series. Increased number of model outputs usually complicates its structure, expanding the set of estimated parameters and make the learning more difficult. Examples for this can be neural networks or neuro-fuzzy systems. Other models such as ARIMA and exponential smoothing are only MISO-type. In the case of PSBFMs the number of outputs does not affect the number of parameters and their estimation method, which should be considered as another valuable property.

PSBFMs construct the forecast using nonparametric local regression. The local nature of the model leads to its simplification and accuracy improvement in the neighborhood of the query pattern. For a new query pattern a new local model is built. But building a new model is not very hard task due to the small number of parameters which are estimated in simple and fast procedures (in STLF practice time for preparation of forecast is sufficient to optimize the model).

The simulation studies have shown high accuracy of PSBFMs, especially for one day ahead forecasts. The proposed models are strong competitors for other popular univariate models such as the reference ones: ARIMA, exponential smoothing and multilayer perceptron. It is noteworthy that PSBFMs work correctly in the case of incomplete input information. The loss of even half of the components of the input pattern only slightly increases the forecast error. For other models, such as the above-mentioned reference ones, the lack of input variables is unacceptable.

Future work will focus on:

- taking into account additional input variables (exogenous) such as weather conditions. It can be done by defining "contexts" of the forecast patterns. A context expresses the curve or characteristics of factor correlated with the load, such as daily curve of atmospheric temperature. The weight of the construction pattern is strengthened depending on the similarity of its context to the context corresponding to the query pattern. Another idea is to construct a model that corrects forecasts generated by PSBFMs depending on the context similarity.
- using patterns in other forecasting models, e.g. based on neural and neuro-fuzzy networks, regression trees, random forests and multiple linear regression.
- committees of forecasting models. Aggregation of the results of the component models can reduce forecast error and strengthen stability of the final model.
- specialized forecasting models for an untypical days. The untypical days (public and religious holidays, days before and after holidays) are characterized by a specific load curve. This curve can be predicted on the basis of analogies to previous years when the untypical day occurs cyclically every year.
- introduction confidence degrees to the training data. Each training pattern is labeled with the confidence degree that expresses its representativeness. This reduces the impact of outliers on the forecast. In PSBFMs the confidence degrees can be additional weights of the construction y-patterns in the regression function.
- probabilistic forecasting using pattern similarity-based methods.

ACKNOWLEDGMENT

We are very grateful to James W. Taylor with the Saïd Business School, University of Oxford for supplying British and French data, and Shu Fan and Rob J. Hyndman with the Business and Economic Forecasting Unit, Monash University for supplying Australian data.

REFERENCES

[1] S. Fan, R.J. Hyndman, "Short-Term Load Forecasting Based on a Semi-Parametric Additive Model," *IEEE Trans. Power Systems*, vol. 27, no. 1, pp. 134-141, 2012.

[2] W. Charytoniuk, M.S. Chen, P. Van Olinda, "Nonparametric Regression Based Short-Term Load Forecasting," *IEEE Trans. Power Systems*, vol. 13, no. 3, pp. 725-730, 1998.

[3] G. Gross, F.D. Galiana, "Short-Term Load Forecasting," *Proc. IEEE*, vol. 75, no. 12, pp. 1558-1573, 1987.

[4] G. Dudek, "Pattern Similarity-based Methods for Short-Term Load Forecasting – Part 1: Principles," *Applied Soft Computing (in review).*

[5] H.T. Yang, C.M. Huang, "A New Short-Term Load Forecasting Approach using Self-Organizing Fuzzy ARMAX Models," *IEEE Trans. Power Systems*, vol. 18, np. 2, pp. 673-679, 2003.

[6] R. Weron, *Modeling and Forecasting Electricity Loads and Prices. A Statistical Approach.* Wiley, 2006.

[7] S.A. Soliman, A.M. Al-Kandari, *Electrical Load Forecasting: Modeling and Model Construction*. Butterworth-Heinemann, 2010

[8] A. Khotanzad, "Short-Term Load and Price Forecasting with Artificial Neural Networks," *The Electric Power Engineering Handbook*, L.L. Grigsby, ed., CRC Press, 2001.

[9] H.S. Hippert, J.W. Taylor, "An Evaluation of Bayesian Techniques for Controlling Model Complexity And Selecting Inputs in a Neural Network for Short-Term Load Forecasting," *Neural Networks*, vol. 23, pp. 386-395, 2010.

[10] N. Amjady, F. Keynia, "Short-Term Load Forecasting of Power Systems by Combination of Wavelet Transform And Neuro-Evolutionary Algorithm," *Energy*, vol. 34, pp. 46-57, 2009.

[11] Y. Chen, P.B. Luh, C. Guan, Y. Zhao, L.D. Michel, M.A. Coolbeth, P.B. Friedland, S.J. Rourke, "Short-Term Load Forecasting: Similar Day-Based Wavelet Neural Networks," *IEEE Trans. Power Systems*, vol. 25, no. 1, pp. 322-330, 2010.

[12] Z.A. Bashir, M.E. El-Hawary, "Applying Wavelets to Short-Term Load Forecasting Using PSO-Based Neural Networks," *IEEE Trans. Power Systems*, vol. 24, no. 1, pp. 20-27, 2009.

[13] Z. Xiao, S.-J. Ye, B. Zhong, C.X. Sun, "BP Neural Network with Rough Set for Short Term Load Forecasting," *Expert Systems with Applications*, vol. 36, pp. 273-279, 2009.

[14] Hao Quan, D. Srinivasan, A. Khosravi, "Short-Term Load and Wind Power Forecasting Using Neural Network-Based Prediction Intervals,"

IEEE Trans. Neural Networks and Learning Systems, vol.25, no.2, pp.303-315, 2014.

[15] H. Kebriaei, B.N. Araabi, A. Rahimi-Kian, "Short-Term Load Forecasting with a New Nonsymmetric Penalty Function," *IEEE Trans. Power Systems*, vol. 26, no. 4, pp. 1817-1825, 2011.

[16] Z. Yun, Z. Quan, S. Caixin, L. Shaolan, L. Yuming, S. Yang, "RBF Neural Network and ANFIS-Based Short-Term Load Forecasting Approach in Real-Time Price Environment," *IEEE Trans. Power Systems*, vol. 23, no. 3, pp. 853-858, 2008.

[17] S.J. Yao, Y.H. Song, L.Z. Zhang, X.Y. Cheng, "Wavelet Transform and Neural Networks for Short-Term Electrical Load Forecasting," *Energy Conversion & Management*, vol. 41, pp. 1975-1988, 2000.

[18] O.A.S. Carpinteiro, A.J.R. Reis, A.P. Alves da Silva, "A Hierarchical Neural Model in Short-Term Load Forecasting," *Applied Soft Computing*, vol. 4, pp. 405-412, 2004.

[19] S. Fan, L. Chen, "Short-Term Load Forecasting Based on an Adaptive Hybrid Method," *IEEE Trans. Power Systems*, vol. 21, no. 1, pp. 392-401, 2006.

[20] V. Yadav, D. Srinivasan, "A SOM-Based Hybrid Linear-Neural Model For Short-Term Load Forecasting," *Neurocomputing*, vol. 74, pp. 2874-2885, 2011.

[21] A. Lendasse, M. Verleysen, E. de Bodt, M. Cottrell, P. Gregoire, "Forecasting Time-Series by Kohonen Classification," *Proc. European Symposium on Artificial Neural Networks*, pp. 221–226, 1998.

[22] E.E. Elattar, J. Goulermas. Q.H. Wu,"Generalized Locally Weighted GMDH for Short Term Load Forecasting," *IEEE Trans. Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol.42, no.3, pp.345-356, 2012.

[23] H.S. Hippert, C.E. Pedreira, R.C. Souza, "Neural Networks for Short-Term Load Forecasting: A Review and Evaluation," *IEEE Trans. Power Systems*, vol. 16, no. 1, pp. 44-55, 2001.

[24] V.S. Kodogiannis, E.M. Anagnostakis, "Soft Computing Based Techniques for Short-Term Load Forecasting," *Fuzzy Sets and Systems*, vol. 128, pp. 413-426, 2002.

[25] T. Chen, M.-J.J. Wang, "Forecasting Methods Using Fuzzy Concepts," Fuzzy Sets and Systems, vol. 105, pp. 339-352, 1999.

[26] A.M. Al-Kandari, S.A. Soliman, M.E. El-Hawary, "Fuzzy Short-Term Electric Load Forecasting," *Electrical Power and Energy Systems*, vol. 26, pp. 111-122, 2004.

[27] S.E. Papadakis, J.B. Theocharis, A.G. Bakirtzis, "A Load Curve Based Fuzzy Modeling Technique for Short-Term Load Forecasting," *Fuzzy Sets and Systems*, vol. 135, pp. 279-303, 2003.

[28] P.A. Mastorocostas, J.B. Theocharis, V.S. Petridis, "A Constrained Orthogonal Least-Squares Method For Generating TSK Fuzzy Models: Application to Short-Term Load Forecasting," *Fuzzy Sets and Systems*, vol. 118, pp. 215-233, 2001.

[29] V.H. Hinojosa, A. Hoese, "Short-Term Load Forecasting Using Fuzzy Inductive Reasoning and Evolutionary Algorithms," *IEEE Trans. Power Systems*, vol. 25, no. 1, pp. 565-574, 2010.

[30] Z. Yun, Z. Quan, S. Caixin, L. Shaolan, L. Yuming, S. Yang, "RBF Neural Network and ANFIS-Based Short-Term Load Forecasting Approach in Real-Time Price Environment," *IEEE Trans. on Power Systems*, vol. 23, no. 3, pp. 853-858, 2008.

[31] M. Hanmandlu, B.K. Chauhan, "Load Forecasting Using Hybrid Models," *IEEE Trans. Power Systems*, vol. 26, no. 1, pp. 20-29, 2011.

[32] J. Du, M.J. Er, L. Rutkowski, "An Efficient Adaptive Fuzzy Neural Network (EAFNN) Approach for Short Term Load Forecasting," *Artificial Intelligence and Soft Computing*, Rutkowski L. at al., eds., ICAISC 2010, LNCS, vol. 6113, pp. 49-57, Springer, 2010.

[33] H. Mao, X.-J. Zeng, G. Leng, Y.-J. Zhai, J.A. Keane, "Short-Term and Midterm Load Forecasting Using a Bilevel Optimization Model," *IEEE Trans. Power Systems*, vol. 24, no. 2, pp. 1080-1090, 2009.

[34] G. Dudek, "Neuro-Fuzzy Approach to the Next Day Load Curve Forecasting," *Przegląd Elektrotechniczny (Electrical Review)*, vol. 87, no. 2, pp. 61-64, 2011.

[35] D.K. Chaturvedi, A.P. Sinha, O.P. Malik, "Short Term Load Forecast using Fuzzy Logic and Wavelet Transform Integrated Generalized Neural Network," *International Journal of Electrical Power & Energy Systems*, vol. 67, pp. 230-237, 2015.

[36] E. Ceperic, V. Ceperic, A. Baric, "A Strategy for Short-Term Load Forecasting by Support Vector Regression Machines," *IEEE Trans. Power Systems*, vol.28, no.4, pp.4356-4364, 2013.

[37] Q. Wu, "Power Load Forecasts Based on Hybrid PSO with Gaussian and Adaptive Mutation and Wv-SVM," *Expert Systems with Applications*, vol. 37, pp. 194-201, 2010.

[38] Y. Wang, Q. Xia, C. Kang, "Secondary Forecasting Based on Deviation Analysis for Short-Term Load Forecasting," *IEEE Trans. on Power Systems*, vol. 26, no. 2, pp. 500-507, 2011.

[39] G. Dudek, "Next Day Electric Load Curve Forecasting using k-Means Clustering," *Rynek Energii (Energy Market)*, vol. 92, no. 1, pp. 143-149, 2011.

[40] M. De Felice, "Short-Term Load Forecasting with Neural Network Ensembles: A Comparative Study," *IEEE Computational Intelligence Magazine*, vol. 6, no. 3, pp. 47-56, 2011.

[41] K. Siwek, S. Osowski, R. Szupiluk, "Ensemble Neural Network Approach for Accurate Load Forecasting in a Power System," *International Journal of Applied Mathematics and Computer Science*, vol. 19, no. 2, pp. 303-315, 2009.

[42] Rui Zhang, Zhao Yang Dong, Yan Xu, Ke Meng, Kit Po Wong, "Short-term load forecasting of Australian National Electricity Market by an ensemble model of extreme learning machine," *IET Generation, Transmission & Distribution*, vol.7, no.4, pp. 391-397, 2013.

[43] F. Martínez-Á□lvarez, A. Troncoso, J. C. Riquelme, J. S. Aguilar-Ruiz, "Energy Time Series Forecasting based on Pattern Sequence Similarity," *IEEE Trans. Knowledge and Data Engineering*, 23(8):1230-1243, 2011.

[44] E. Paparoditis, T. Sapatinas, "Short-Term Load Forecasting: The Similar Shape Functional Time-Series Predictor," *IEEE Trans. Power Systems*, vol.28, no.4, pp.3818-3825, 2013.

[45] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer, 2009.

[46] P. Grassberger, I. Procaccia, "Measuring the Strangeness of Strange Attractors," *Physica D: Nonlinear Phenomena* , vol. 9, no. 1-2, pp. 189-208, 1983

[47] G. Dudek, "Tournament Searching Method to Feature Selection Problem," *Artificial Intelligence and Soft Computing*, Rutkowski L. at al., eds., ICAISC 2010, LNCS, vol. 6114, pp. 437–444, Springer, 2010.

[48] D.W. Scott, *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley, 1992.

[49] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, 1996.

[50] S. Theodoridis, K. Koutroumbas, *Pattern Recognition*. Elsevier, 2009.

[51] G. Dudek, "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).

[52] G. Dudek, "A Comparison of the Neural Gas and Self Organizing Map Methods for Next Day Load Curve Forecasting," *Przegląd Elektrotechniczny (Electrical Review)*, vol. 85, no. 3, pp. 153–156, 2009.

[53] G. Dudek, "Artificial Immune System for Short-Term Electric Load Forecasting," *Artificial Intelligence and Soft Computing*, Rutkowski L. at al., eds., ICAISC 2008, LNCS, vol. 5097, pp. 1007–1017, Springer, 2008.

[54] G. Dudek, "Artificial Immune Clustering Algorithm to Forecasting Seasonal Time Series," *Computational Collective Intelligence, Technologies and Applications*, Jędrzejowicz P. et al., eds., ICCCI 2011, LNCS, vol. 6922, pp. 468-477, Springer, 2011.

[55] G. Dudek, "Artificial Immune System for Forecasting Time Series with Multiple Seasonal Cycles," *LNCS Transactions on Computational Collective Intelligence XI*, LNCS, vol. 8065, pp. 176-197, Springer, 2013.

[56] G. Dudek, "Short-Term Forecasting of Load and Energy Prices on the Balancing and Stock Markets using Clustering Methods," Final report of the Rector of Czestochowa University of Technology founded grant no. BW-3-301-204/2007/kier.bad., Dept. Elect. Eng., Częstochowa University of Technology, 2008 (unpublished, in Polish).

[57] R.J. Hyndman, Y. Khandakar, "Automatic Time Series Forecasting: The Forecast Package for R," *Journal of Statistical Software*, vol. 27, no. 3, pp. 1–22, 2008.

[58] R.J. Hyndman, A.B. Koehler, J.K. Ord, R.D. Snyder, *Forecasting with Exponential Smoothing: The State Space Approach*, Springer, 2008.

[59] G. Dudek, "Forecasting Time Series with Multiple Seasonal Cycles using Neural Networks with Local Learning," *Artificial Intelligence and Soft Computing*, Rutkowski L. at al., eds., ICAISC 2013, LNCS, vol. 7894, pp. 52-63, Springer, 2010.

[60] F.D. Foresee, M.T. Hagan, "Gauss-Newton Approximation to Bayesian Regularization," Proc. Inter. Joint Conference on Neural Networks, pp. 1930–1935, 1997.

Grzegorz Dudek received his PhD degree in electrical engineering from the Czestochowa University of Technology, Poland, in 2003 and habilitation degree in computer science from the Łódź University of Technology, Poland, in 2013. Currently, he is an associate professor at the Department of Electrical Engineering, Czestochowa University of Technology. His research interests include pattern recognition, machine learning, artificial intelligence, and their application in classification, regression, forecasting and optimization problems especially in the field of power systems.