Evolving Systems (2012) 3:81–93
DOI 10.1007/s12530-011-9043-0

ORIGINAL PAPER

Introducing evolving Takagi–Sugeno method based on local least squares support vector machine models

Mohammad Komijani · Caro Lucas · Babak Nadjar Araabi · Ahmad Kalhor

Received: 4 May 2011 / Accepted: 29 November 2011 / Published online: 17 December 2011

Abstract In this study, an efficient local online identification method based on the evolving Takagi–Sugeno least square support vector machine (eTS-LS-SVM) for nonlinear time series prediction is introduced. As an innovation, this paper has applied the nonlinear models, i.e. local LS-SVM models, as the consequence parts of the fuzzy rules, instead of the linear models used in the conventional evolving TS fuzzy models. In each step, the proposed learning approach includes two phases. The fuzzy rules (rule premise) are first created and updated adaptively based on a sequential clustering technique to obtain the structure of TS model. Then, the parameters of each local LS-SVM model (rule consequence) are recursively updated by deriving a new recursive algorithm (a local decremental and incremental procedure) to minimize the local modelling error and trace the process's dynamics. Besides, a new learning algorithm based on the recursive gradient-based method is used to adaptively update the meta-parameters of the LS-SVM models. Comparison of the suggested method with some of the previous approaches based on the online prediction of the nonlinear time series has shown that the introduced identification algorithm has a proper performance in terms of learning and generalization abilities while having a lower redundancy.

Keywords Evolving Takagi–Sugeno · Least square support vector machine · Time series prediction

M. Komijani (✉) · C. Lucas · B. N. Araabi · A. Kalhor
Control and Intelligent Processing Center of Excellence, School of Electrical and Computer Engineering, University of Tehran, Tehran 14395-515, Iran
e-mail: m.komijani@gmail.com

C. Lucas
e-mail: lucas@ipm.ir

B. N. Araabi
e-mail: araabi@ut.ac.ir

A. Kalhor
e-mail: akalhor@ut.ac.ir

1 Introduction

The evolving fuzzy rule-based systems have been in the focus of attention since the beginning of the last decade. Direct learning of fuzzy rule-based systems is an approach which employs both online clustering and supervised learning methods to recursively update the structure and parameters of Takagi–Sugeno (TS) model (Kasabov and Song 2002; Lughofer and Klement 2005; Angelov et al. 2008).

Taking the possibility of merging or splitting of fuzzy sets and rules into account is a helpful idea for keeping rule bases compact and interpretable (Angelov and Zhou 2006; Angelov and Filev 2006; Lughofer et al. 2011). More details are provided in Ramos et al. (2010). In recent years, several incremental clustering-based methods with the main focus on the discovery of inherent model structure and premise variables have been reported (Martinez et al. 2008; Mirmomeni et al. 2011). In this context, indirect partitioning of data space (Kalhor et al. 2009) and recursive Gath–Geva clustering (Soleimani-B et al. 2010) are two considerable efforts to handle more sophisticated behaviours of nonlinear systems with less modelling error as well as fewer number of local models. In general, these studies have employed TS model using recursive parameter estimation to solve a complex modeling problem by decomposing it into a number of simpler sub-problems (Pouzols and Lendasse 2010; Dovžan and Škrjanc 2011). The conventional evolving TS fuzzy models have often

used an online clustering method to update the premise part parameters, and then have used local linear models in the consequence part to approximate subspaces caused by those clustering techniques (Lin et al. 2011). However, fitting the local linear models to the created local regions which are not necessarily linear may lead to undesirable modelling error. To solve this problem, some papers have sought to increase the degree of granularity versus the degree of nonlinearity in the model. The extraction of the correct granularity may cause piecewise linear model to approximate a nonlinear dependency quite well; however, this requires the use of much more local models, and hence more parameters are needed (Kasabov and Song 2002; Lughofer 2008; Angelov and Zhou 2006; Yamauchi 2010; Kim et al. 2007). In this context, a good solution is to employ local nonlinear models considering the possibility of recursive learning.

Nonlinear system identification based on support vector machine (SVM) (Vapnik 1995) has received increasing attention in pattern recognition (e.g. classification) and function approximation (e.g. regression) problems due to its high generalization performance and ability to derive the optimal network structure (Smola and Schölkopf 2004). Least squares support vector machine (LS-SVM) is a modified version of SVMs which considers equality constraint instead of inequalities as in the classical SVM approach. It holds the integrity of the traditional SVM and greatly simplifies the problem by direct solving of a set of linear equations instead of quadratic programming (QP) problem (Suykens and Vandewalle 1999). Despite this prominent property, LS-SVM's solution has some main drawbacks such as lost of sparseness and less robust estimation, compared to the classical SVM. These two shortcomings can be solved by omitting relatively small amount of the least meaningful support values (Suykens et al. 2002), and trying to give each data point its proper amount of influence in parameter estimation (Leuven et al. 2000), respectively.

However, most existing algorithms for the SVM and LS-SVM require that training samples be delivered in a single batch; i.e. they are offline algorithms and do not fit the real world applications such as online system identification and control problems. More recently, much effort has been made to propose recursive algorithms for LS-SVM (Diehl and Cauwenberghs 2003; Engel et al. 2004; Liu et al. 2009). However, the main drawback of these recursive algorithms is the lack of sparsity implying that as the online process goes forward, all the samples enter to the identification stage, and thus an excessive computation time occurs during the learning. In order to obtain a parsimonious model, several tricks have been proposed. In Chi and Ersoy (2003), using some matrix update equations, an adaptive learning has been suggested in which the size of training data remains fixed by removing the oldest data sample as the new one is available. Another idea is to remove the least important sample corresponding to the least valuable support vector at each sampling period (Tang et al. 2006; Li et al. 2007).

de Kruif and de Vries (2003) and An et al. (2007) have proposed an alternative procedure to eliminate the useless data point using fast leave-one-out (FLOO) criterion which implies the smallest approximation error. In Liu et al. (2010), a sparsification strategy using prediction error criterion and FLOO criterion has been developed to restrict the model complexity for online identification of multi-input multi-output (MIMO) systems. Aiming at online modeling of batch processes, Liu et al. (2007) has presented a local LS-SVM (LLS-SVM) that selects the neighbouring data of new input into the training set. For continuous processes, the global LS-SVM always gives good prediction for the inputs located in the neighbourhood of dense samples but incapable for those in the sparse part. Keeping this fact into mind, Li et al. (2010) has recommended that only the samples similar to testing sample should be selected for training to give accurate prediction.

Cheng and Juang (2011) have proposed an incremental support vector machine-trained TS-type fuzzy classifier (ISVM-FC) in which the structure of TS model are determined by an incremental rule generation approach and the consequent parameters are estimated within one identification process namely global linear SVM. Therefore, with the global approach, the generated local models are not necessarily the local linearization of the nonlinear system and the model cannot be locally interpreted (Abonyi and Babuska 2000).

This paper has developed an effective method, called the evolving Takagi–Sugeno LS-SVM (eTS-LS-SVM), in which local LS-SVMs are used as a piecewise nonlinear estimator in the consequence part of TS model. Adopting this method, the performance of the recursive global LS-SVM algorithms in prediction of sparse continuous data set is improved and the said shortcoming of the conventional evolving TS fuzzy systems can be compensated. Efficient sequential clustering method with the possibility of splitting or merging of fuzzy rules is used to adjust the the rule premise parameters of validity functions. Deriving a novel recursive algorithm (a local decremental and incremental procedure), the parameters in each local LS-SVM model (rule-consequence) are recursively updated to minimize the local modelling error. Furthermore, as another contribution, a new learning algorithm based on the recursive gradient-based method is employed to adaptively update the meta-parameters of the LS-SVM models. The suggested method has been compared with some previous approaches to the prediction of nonlinear time series (Mackey–Glass time series and sunspot number time series), and the results have shown that our identification algorithm has a comparatively superior performance in

terms of generalization ability with less redundancy. Short term load forecasting has also been applied as another case study to show that a local LS-SVM may produce a better result with less prediction error.

The remainder of the paper is organized as follows: in Sect. 2, main aspects of TS-LS-SVM model are briefly illustrated. Sections 3 and 4 are devoted to describing the recursive learning for tuning the parameters of the proposed model. The suggested method is implemented on some case studies in Sect. 5. Finally, the concluding remarks are provided in Sect. 6.

2 Structure of Takagi–Sugeno-LS-SVM model

Fuzzy modelling is a powerful and practical tool for modelling and control of the nonlinear systems and complex processes. The method of fuzzy reasoning introduced by Takagi and Sugeno has become one of the major topics in the field of fuzzy control and modeling. The basic concept of TS modeling is to decompose input data space into fuzzy regions and to approximate the system in each region by a simple function. Learning method of TS models is based on the idea of structure and parameter identification. Structure identification includes determination of fuzzy rules and partitioning of input/output space, and parameter estimation involves the calculation of consequence parameters of the rules. TS fuzzy models can provide an effective solution to the nonlinear function approximation problem. Assume M fuzzy rules exist, $x \in R^n$ is the input variable, and $y_i \in R$ is the output. A typical TS fuzzy rule has the following form: R^i : IF $(x_1$ *is* $\phi_1^i)$ *and* $\cdots$ *and* $(x_n$ *is* $\phi_n^i)$, Then $y_i = f_i$.

In this paper, a new Takagi–Sugeno (TS) fuzzy model with LS-SVM as the local nonlinear model in the output layer is presented, which can approximate a class of nonlinear dynamical systems. Each local LS-SVM model has the following form (Smola and Schölkopf 2004):

$$\hat{y}^i = \frac{1}{\gamma}\sum_{k=1}^{t} \alpha_k^i K(x, x_k) + b^i \tag{1}$$

where t is the number of training set, $K(.,.)$ is the kernel function, and α^i and b^i are the consequence parameters.

Output of the model can be simply calculated as the weighted sum of the outputs of local models:

$$\hat{y} = \sum_{i=1}^{M} \hat{y}^i \Phi^i(x) \tag{2}$$

where $\Phi^i(x)$ denotes the ith validity function which is used only for partitioning of input space, whereas linear combination of kernel functions, $\hat{y}^i$, is used for nonlinear transform (i.e. capturing the nonlinear behaviour). The architecture of TS-LS-SVM model is depicted in Fig. 1.

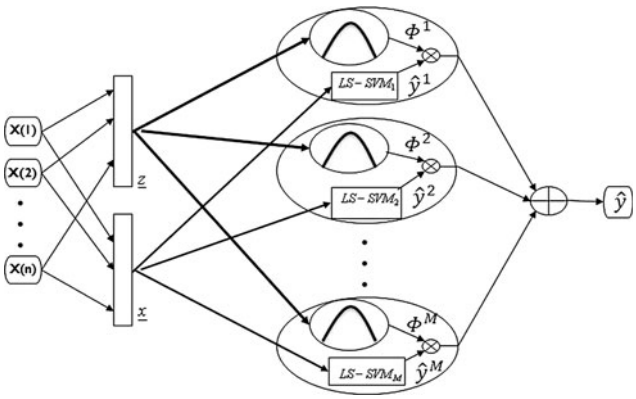

Fig. 1 Structure of TS-LS-SVM

For n-dimensional input space, the separable fuzzy basis as the tensor product of n single basis functions is utilized to decompose the signal defined by

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

The validity functions are considered as normalized Gaussian basis functions; normalization is necessary for a proper interpretation of validity functions:

$$\Phi^i(x) = \frac{\phi^i(x)}{\sum_{l=1}^{M} \phi^l(x)} \tag{4}$$

$$\phi^i(x) = \exp(-0.5(x-\mu^i)^T\,(C^i)^{-1}(x-\mu^i)) \tag{5}$$

where μ^i and C^i are the center and covariance matrix related to the ith validity function, respectively. The parameters of nonlinear hidden layer are the same as the parameters of validity functions (namely, center (μ^i) and covariance matrix (C^i), in which i stands for the ith validity function). Optimization or learning methods are generally used to adjust the rule premise parameters (μ^is and C^is) and the rule consequent parameters (α^is and b^is) of the TS model.

3 Structure identification using sequential clustering

In order to partition the input space as is evident from (5), every cluster can be defined by utilizing its center and covariance matrix. Therefore, recursive tuning of centers and covariance matrices values is considered in an incremental clustering method, as described in the following stages.

3.1 Cluster creation

Two non-recursive formulae for estimation of the centre (μ^i) and the covariance matrix (C^i) for the ith cluster with n^i data points are defined as below (Duda et al. 2001):

$$\mu^i = \frac{1}{n^i}\sum_{k=1}^{n^i} x_k \tag{6}$$

$$C^i = \frac{1}{n^i}\sum_{k=1}^{n^i} (x_k - \mu^i)(x_k - \mu^i)^T \tag{7}$$

Alternative recursive techniques for calculation of μ^i and C^i based on the successive addition of new samples are introduced by Duda et al. (2001):

$$\mu^i_{k+1} = \mu^i_k + \frac{1}{n^i_{k+1}+1}(x_{k+1} - \mu^i_k) \tag{8}$$

$$C^i_{k+1} = \frac{n^i_k}{n^i_k+1}C^i_k + \frac{n^i_k}{(n^i_k+1)^2}(x_{k+1} - \mu^i_k)(x_{k+1} - \mu^i_k)^T \tag{9}$$

where the $(k+1)$ and k stand for the new and previous values of the parameters, respectively, and n^i is the number of data samples belonging to the ith cluster. It should be noted that the non-recursive calculation of $(C^i)^{-1}$ (the inverse of the covariance matrix) in 5 may require computations by standard matrix methods. An alternative recursive method for $(C^i)^{-1}$ computation employed in this study is called Sherman-Morrison-Woodbury matrix identity that is given by Duda et al. (2001):

$$(A + xy^T)^{-1} = A^{-1} - \frac{A^{-1}xy^TA^{-1}}{1 + y^TA^{-1}x} \tag{10}$$

By inserting (9) into (10), then:

$$(C^i_{k+1})^{-1} = \frac{n^i_k+1}{n^i_k}\left[(C^i_k)^{-1} - \frac{(C^i_k)^{-1}(x_{k+1} - \mu^i_k)(x_{k+1} - \mu^i_k)^T(C^i_k)^{-1}}{(n^i_k+1) + (x_{k+1} - \mu^i_k)^T(C^i_k)^{-1}(x_{k+1} - \mu^i_k)}\right] \tag{11}$$

It is clear that (11) reveals less computational complexity than the direct matrix inversion.

By the first step, M is assumed to be the number of existing clusters. Initial value of M is 0 since there is no cluster. In the next step, first cluster is created with $\mu^1 = x_1$ and $C^1 = \sigma_0 \times I_{n\times n}$ where σ denotes initial deviations for the new cluster. A suitable fuzzy set width, σ_0, is assigned according to the input domain, so that a fuzzy set covers a suitable input region. A too large value of σ_0 generates a fuzzy set that covers almost the entire input range and the generated fuzzy sets are highly overlapped. On the contrary, a too low value of σ_0 generates fuzzy sets with almost no overlap.

3.2 Cluster merging

At the previous stage, an incremental clustering algorithm is considered to improve the discovery of local structures. However, its dependency on the order of data is clearly evident. Furthermore, to guarantee the generalization property, priority order is given to a model which has a smaller number of clusters and consequently less parameters. This goal can be achieved by merging the two closest clusters (mth and nth clusters) into one. For merging process, similarity criterion should be defined and the clusters that can be merged according to that similarity criterion should be found. We first define the membership degree for a given n-dimensional input vector (x) by Kim et al. (2007):

$$\phi^l(x) = \exp(-0.5(x - \mu^l)^T\,(C^l)^{-1}(x - \mu^l)) \tag{12}$$

where μ^l and C^l are the center and covariance matrix related to the lth cluster, respectively. For each two separate clusters, a relation to get the similarity criterion between the mth and nth clusters can be defined by:

$$sim^{mn} = \sqrt{\phi^m(\mu^n)\phi^n(\mu^m)} \tag{13}$$

The reason why choosing the similarity criterion in (13) is that, as the two clusters become more close to each other, the value of the similarity measure increases which further helps in finding potential candidates for merging process.

These two clusters are merged into one cluster if

$$sim^{mn} \geq \rho \tag{14}$$

where ρ is a problem-dependent parameter. The parameters of the two eligible clusters for merging are calculated as follows (Soleimani-B et al. 2010):

$$\mu^l = \frac{n^m\mu^m + n^n\mu^n}{n^l}, \quad n^l = n^m + n^n \tag{15}$$

$$C^l = \frac{n^mC^m + n^nC^n}{n^l} + \frac{n^mn^n}{(n^l)^2}(\mu^m - \mu^n)(\mu^m - \mu^n)^T. \tag{16}$$

3.3 Procedures of cluster partitioning algorithm

Now, stages of the sequential cluster partitioning algorithm can be represented in Table 1.

4 Recursive local least squares support vector machines

4.1 Least squares support vector machine

Given training set $\{x_k, y_k\}_{k=1}^t$ where t is number of training data set, $x \in R^n$ is the regression vector and $y \in R$ is the output vector, we can take the following form in feature space by use of the nonlinear function $\theta(x)$ (Suykens and Vandewalle 1999):

$$y(x) = w^T\theta(x) + b \quad \text{with} \quad w \in Z^n, \quad b \in R \tag{17}$$

Table 1 Stages of the sequential cluster partitioning algorithm

1. Initiate $\mu^1 = x_1$, $C^1 = \sigma_0 \times I_{n\times n}$ and $M = 1$.
2. Create a new cluster for current data sample or assign it to a proper existing cluster. Cluster suitability is determined by membership degree. Therefore,
   $\overset{\wedge}{\phi} = \max_{i=1}^{M} \phi^i(x)$ and $\hat{i} = \arg\max_{i=1}^{M} \phi^i(x)$
   (a) If $\overset{\wedge}{\phi} \leq \tau$, create a new cluster similar to the first stage, and set $(M+1) \rightarrow M$. A good idea for choosing a proper value for the parameter τ has been considered in (Kim et al, 2007).
   (b) If $\overset{\wedge}{\phi} \geq \tau$, assign the current data sample to the $(\hat{i})^{th}$ cluster. The center and covariance of this cluster are updated according to (8) and (9). *
3. Compute sim^{mn} for:
   (a) $m = M$ and $n = 1,2,\ldots,(M-1)$ if in previous stage $\overset{\wedge}{\phi} \leq \tau$
   (b) $m = \hat{i}$ and $n = 1,2,\ldots,M; n \neq \hat{m}$ if in previous stage $\overset{\wedge}{\phi} \geq \tau$
   it means that only merged clusters or new added one is a candidate for merging.
4. If (14) is satisfied, then the m^{th} and the n^{th} clusters are merged into one $(\overset{*}{i})$ and set $(M-1) \rightarrow M$, again compute sim^{mn} for $m = \overset{*}{i}$ and $n = 1,2,\ldots,M; n \neq \overset{*}{i}$; return to the beginning of stage 4.
   Else, go to the next step.
5. If there is more data point available, repeat the process from step 2.

* However, regarding this step, the algorithm may be prone to single outliers. To avoid this problem, the preprocessing methods, as well as the methods used in Mirmomeni et al. (2011) and Angelov and Filev (2004), can be considered. These methods practically exclude the chance of an outlier to become a cluster center and to influence the output of the model

In LS-SVM for function estimation, the optimization problem using the Tihonov regularization is formulated to get solution for (17) (Suykens and Vandewalle 1999):

$$\min_{w,e}\left\{J(w,e) = \frac{\gamma}{2}w^T w + \frac{1}{2}\sum_{k=1}^{t} e_k^2\right\}$$
$$\text{s.t. } y_k = w^T\theta(x_k) + b + e_k \quad k = 1,2,\ldots,t, \quad \gamma \succ 0 \tag{18}$$

where e_k is the approximation error between actual output and predictive output of the kth data, $\gamma > 0$ is regularization parameter which determines the trade-off between structural risk minimization (SRM) and empirical risk minimization (ERM), and $\theta(.)$ is a nonlinear mapping which maps the input data into a high-dimensional feature space.

By constructing the Lagrangian the problem is given by:

$$L(w,b,e,\alpha) = J(w,e) - \sum_{k=1}^{t}\alpha_k(w^T\theta(x_k) + b + e_k - y_k) \tag{19}$$

where $\alpha = [\alpha_1,\ldots,\alpha_t]^T$ are Lagrange multipliers and the optimality conditions are given as

$$\begin{cases} \frac{\partial L}{\partial w} = 0 \rightarrow w = \frac{1}{\gamma}\sum_{k=1}^{t}\alpha_k\theta(x_k) \\ \frac{\partial L}{\partial b} = 0 \rightarrow \sum_{k=1}^{t}\alpha_k = 0 \\ \frac{\partial L}{\partial e_k} = 0 \rightarrow \alpha_k = e_k \\ \frac{\partial L}{\partial \alpha_k} = 0 \rightarrow [w^T\theta(x_k) + b] - y_k + e_k = 0 \end{cases} \tag{20}$$

After elimination of the variables w and e_k, the following linear equations can be obtained:

$$\begin{bmatrix} b \\ \alpha \end{bmatrix} = P\begin{bmatrix} 0 \\ Y \end{bmatrix} = \Theta \tag{21}$$

$$P = \begin{bmatrix} 0 & \Gamma^T \\ \Gamma & U = (\Omega + I) \end{bmatrix}^{-1} \tag{22}$$

where $Y = [y_1,\ldots,y_t]^T, \Gamma = [1,\ldots,1]^T$, I is the $t \times t$ identity matrix and $\Omega_{m,n} = \theta(x_n)^T\theta(x_m)\gamma^{-1} = k(x_n,x_m)\gamma^{-1}$, $n,m = 1,\ldots,t$ and $k(x_n, x_m)$ is a kernel matrix.

The model of global LS-SVM can get the following form (Leuven et al. 2000):

$$y = \frac{1}{\gamma}\sum_{k=1}^{t}\alpha_k K(x,x_k) + b = \frac{1}{\gamma}\alpha^T K + b \tag{23}$$

where α and b are solutions of (21) and (22). There are several possibilities, as choices of kernel function. Radial basis function $(K(x_m,x_n) = \exp(-\|x_m - x_n\|^2/\sigma^2))$ as the most popular one, is chosen in this paper in the experiments.

4.2 Two-stage recursive parameter identification of local LS-SVM models

In this part, recursive updating of local LS-SVM parameters is derived from the decremental and incremental

algorithm. We implement an online adaptive selective kernel learning algorithm for local LS-SVM using a moving-window where training samples are selectively pruned and added.

Before more explanations about the decremental and incremental algorithm, the necessary parameters for implementation of the algorithm must be determined, that is, parameters of the first local model (at the first step), LS-SVM model parameters of the newly built cluster (at merging step), and LS-SVM model parameters of the cluster resulted from the merging step.

At the first step ($k = 1$) and for the first local model ($i = 1$), the first N data pairs $\{X_1^1, Y_1^1\} = \{(x_1^1(1), y_1^1(2)), \ldots, (x_1^1(N), y_1^1(N))\}$ are employed and $P_1^1 \in R^{N\times N}, \alpha_1^1 \in R^N$ and $b_1^1 \in R$ can be directly got from (21) and (22).

At the kth step, if a new cluster is to be built, our strategy has been to adopt its required parameters (i.e. $\left\{X_k^{(M+1)}, Y_k^{(M+1)}\right\}, P_k^{(M+1)}, \alpha_k^{(M+1)}$ and $b_k^{(M+1)}$) from the adjacent cluster. By substituting the center of the newly built cluster into (12) for $i = 1, \ldots, M$, the adjacent cluster can be identified as the cluster with the maximum membership degree. It should be noted that M signifies the number of existing clusters.

At the merging step, in which two close clusters are merged together, the LS-SVM model parameters for the resultant cluster can be extracted from either of the two clusters.

With regard to the above mention, in the decremental stage, the node with the least importance in any moving-windows training set in the local models is removed, and then the local models' parameters are updated. After that, in the incremental stage, the new incoming data sample is added to any moving-windows training set in the local models, and the final parameters are determined recursively.

At the decremental stage, the useless data pair (i.e. the $(j)^*$th node of any moving-windows training set in the local models) should be pruned from $\{X_k^i, Y_k^i\} = \{(x_k^i(1), y_k^i(2)), \ldots, (x_k^i(j^*), y_k^i(j^*)), \ldots, (x_k^i(N), y_k^i(N))\}$, $(i = 1, \ldots, M)$ using FLOO criterion. The FLOO criterion can be formulated as (Liu et al. 2010):

$$E^i(j) = \frac{\alpha_k^i(j)}{P_k^i(j+1, j+1)}, \quad \begin{matrix} j = 1, \ldots, N \\ i = 1, \ldots, M \end{matrix} \tag{24}$$

where $j^* = \arg Min_j E^i(j), N$ is moving-window's size and M is the number of clusters. This criterion will introduce the smallest error to the updated model when the $(j^i)^*$th node is deleted from the ith cluster (i.e. pruning $(x_k^i(j^*), y_k^i(j^*))$ from $\{X_k^i, Y_k^i\}$). It assures that:

$$\hat{y}^i(x_k) = \hat{y}_{pr}^i + E^i(j^*) \tag{25}$$

where $\hat{y}_{pr}^i$ is the output of ith LS-SVM model after the pruning step.

Then, updating rule for P_k^i can be formulated as (Liu et al. 2010):

$$P_{pr}^i = \bar{P}_k^i - \bar{p}_k^i (\bar{p}_k^i)^T / p_k^i \tag{26}$$

in which:

$$P_k^i = \begin{bmatrix} P_{11}^i & p_{1(j^*+1)}^i & P_{12}^i \\ \left(p_{1(j^*+1)}^i\right)^T & p^i & \left(p_{2(j^*+1)}^i\right)^T \\ \left(P_{12}^i\right)^T & p_{2(j^*+1)}^i & P_{22}^i \end{bmatrix} \tag{27}$$

$$\bar{P}_K^i = \begin{bmatrix} P_{11}^i & P_{12}^i \\ \left(P_{12}^i\right)^T & P_{22}^i \end{bmatrix} \tag{28}$$

$$\bar{p}_k^i = \begin{bmatrix} p_{1(j^*+1)}^i \\ p_{2(j^*+1)}^i \end{bmatrix} \tag{29}$$

where $\bar{P}_K^i$ stands for the matrix $\bar{P}_k^i$ after deleting the $(j+1)^*$th row and $(j+1)^*$th column, respectively, and $\bar{p}_k^i$ denotes the $(j+1)^*$th column of the matrix $\bar{P}_k^i$ after deleting its $(j+1)^*$th node. The pruning strategy for α^i is as follows:

$$\Theta_{pr}^i = P_{pr}^i \left[0\, y^i(1) \ldots y^i(N-1)\right]^T = \begin{bmatrix} b^i \\ \alpha^i \end{bmatrix} \tag{30}$$

For incremental stage, let (x_{k+1}, y_{k+1}) be the new training sample pair, added to the $\{X_k^i, Y_k^i\}, i = 1, \ldots, M$ where M is the number of local models. The recursive updating of P_{k+1}^i is obtained as below (Li et al. 2007):

$$P_{k+1}^i = \begin{bmatrix} P_{pr}^i - \eta_{k+1}^i Z_{k+1}^i (Z_{k+1}^i)^T & \eta_{k+1}^i Z_{k+1}^i \\ \zeta_{k+1}^i \psi_{k+1}^i \left[\eta_{k+1}^i Z_{k+1}^i (Z_{k+1}^i)^T - P_{k+1}^i\right] & -\eta_{k+1}^i \end{bmatrix} \tag{31}$$

where $Y_k^i = [0 \quad y_k^i(1) \ldots y_k^i(N-1)]^T$ and

$$\psi_{K+1}^i = [1 \Omega_{1,N} \Omega_{2,N} \ldots \Omega_{(N-1),N}] \tag{32}$$

$$Z_{k+1}^i = P_{pr}^i (\psi_{k+1}^i)^T \tag{33}$$

$$\zeta_{k+1}^i = (\Omega_{N,N} + 1)^{-1} \tag{34}$$

$$\eta_{k+1}^i = \left(\psi_{k+1}^i Z_{k+1}^i - (\zeta_{k+1}^i)^{-1}\right)^{-1} \tag{35}$$

where in (32), we have: $\Omega_{l,N} = \theta(x_k^i(l))^T \theta(x_k^i(N)) \gamma^{-1}, (l = 1, \ldots, N-1)$ and $(x_k^i(N), y_k^i(N)) = (x_{k+1}, y_{k+1})$.

In this step to update Θ_{k+1}^i we cannot use an update rule something similar to (21) or (30), because we need an update rule which can give us the local estimation property. Therefore, we generalize the update rule by deriving a new recursive algorithm which contains global prediction error

between real output and output of the ith LS-SVM model (e^i_{k+1}):

$$\Theta^i_{k+1} = \begin{bmatrix} \Theta_{pr} + \eta^i_{k+1} Z^i_{k+1} e^i_{k+1} \\ -\eta^i_{k+1} e^i_{k+1} \end{bmatrix} = \begin{bmatrix} b^i_{k+1} \\ \alpha^i_{k+1} \end{bmatrix} \tag{36}$$

e^i_{k+1} is obtained as follows:

$$e^i_{k+1} = (y_{k+1} - \hat{y}^i_{pr}) = \left(y_{k+1} - (Z^i_{k+1})^T Y^i_k\right) \tag{37}$$

where $Y^i_k = [0 \quad y^i_k(1) \ldots y^i_k(N-1)]^T$. A detailed derivation of (36) is provided in the Appendix. Now, we offer a heuristic choice for e^i_{k+1} as the local approximation error:

$$\begin{aligned} e^i_{k+1} &= \Phi^i(x_{k+1})(y_{k+1} - \hat{y}^{pr}_i) \\ &= \Phi^i(x_{k+1})\left(y_{k+1} - (Z^i_{k+1})^T Y^i_k\right) \end{aligned} \tag{38}$$

Φ^i has been introduced previously. Regarding (1) and (36), it can be inferred that $e_{k+1}{}^i$ has a direct relation to $\hat{y}^i$.

Steps of decremental and incremental procedure of local LS-SVM algorithm at each period are illustrated in Table 2.

4.3 Adaptive tuning of regularization parameter and kernel parameters

The Gaussian kernel is one of the most common kernel functions. Regularization parameter and kernel parameter are to be chosen whenever a kernel function is adopted. According to Yongping Zhao (2009), the kernel parameter is more important than the regularization parameter, and has a decisive effect on the performance of the identified model. When a large value is selected for kernel parameter (σ), the elements of the kernel matrix show a more similar trend, and thus the model cannot describe a complex nonlinearity. Using a small value for σ, the elements in the kernel matrix tend to be more dissimilar, and thus the identified model may become over-fit (Liu et al. 2010).

To find a proper value for regularization parameter, it is worth noting that if the value of γ is set too large, it can obtain a more complex model; a too small value of γ can lead to a smoother model, which simply means that the model does not fit the learning data (Liu et al. 2010).

Therefore, with a prior knowledge of the data set, one can select a suitable pair of parameters (which generally, might not be optimal) in a valid and wide enough district using some trial-and-error experiments beforehand. The kernel parameter selection is still an open question in the machine learning area. There is no optimal parameter selection theory that is directed for the issue of online modeling (Liu et al. 2010; Yongping Zhao 2009).

Another contribution of this paper is developing a new recursive formulation for tuning a pair of parameters (i.e. regularization and kernel parameters). As indicated in Table 3, conventional recursive kernel learning-based methods find the optimal regularization and kernel parameters in a predefined manner (e.g. trial-and-error experiments or grid search), while an adaptive method is here innovated for tuning the said parameters (Table 4).

Assuming that the kth set of the rule consequent parameters and rule premise parameters are available from previous iteration, set of the kernel parameters can be calculated from a recursive gradient-based algorithm reported in Ngia et al. (1998). This algorithm is used for minimizing the local error criterion (38), and σ is the kernel parameters vector, which can be defined as:

$$\sigma^i_{k+1} = [\sigma^i_{k+1}(1), \ldots, \sigma^i_{k+1}(N)]^T \tag{39}$$

A recursive Gauss–Newton algorithm (Ngia et al. 1998) can be exploited for estimating (39) as follows:

$$\sigma^i_{k+1} = \sigma^i_k + \mathrm{G}^i_{k+1} J^i_{k+1} e^i_{k+1} \tag{40}$$

$$\mathrm{G}^i_{k+1} = \left[\mathrm{G}^i_k - \frac{\mathrm{G}^i_k J^i_{k+1} J^i_{k+1}{}^{\mathrm{T}} \mathrm{G}^i_k}{1 + J^i_{k+1}{}^{\mathrm{T}} \mathrm{G}^i_k J^i_{k+1}}\right] \tag{41}$$

$$\begin{aligned} J^i_{k+1} &= -\left[\frac{\partial e^i_{k+1}}{\partial \sigma^i_{k+1}(1)} \cdots \frac{\partial e^i_{k+1}}{\partial \sigma^i_{k+1}(N)}\right]^T \\ &= \Phi(x_{k+1})\gamma^{-1}\left[\alpha^i_k(1)\frac{\partial K^i_k(1)}{\partial \sigma^i_k(1)} \ldots \alpha^i_k(N)\frac{\partial K^i_k(N)}{\partial \sigma^i_k(N)}\right]^T \end{aligned} \tag{42}$$

where $\frac{\partial K^i_k(j)}{\partial \sigma^i_k(j)} = \frac{\|x_{k+1} - x^i_k(j)\|^2}{(\sigma^i_k(j))^3} \exp\left(-\frac{\|x_{k+1} - x^i_k(j)\|^2}{(\sigma^i_k(j))^2}\right)$, $(j = 1, \ldots, N)$ corresponds to the jth node and $(i = 1, \ldots, M)$

Table 2 Steps of decremental and incremental procedure of local LS-SVM algorithm

1. Determine weight γ and size of training set N after selecting a proper kernel function. Compute P^1_1, α^1_1 and b^1_1 for the first N training set via (21) and (22)
2. For $k > 1$: eliminate j^*th sample pair from $\{X^i_k, Y^i_k\}$ (for $i = 1, \ldots, M$) using FLOO criterion (24) and determine P^i_{pr} and Θ^i_{pr} for each cluster using (26) and (30), respectively
3. Add the new training sample pair (x_{k+1}, y_{k+1}) to the $\{X^i_k, Y^i_k\}, i = 1, \ldots, M$, where M is number of local models and then, compute ψ^i_{k+1}, z^i_{k+1}, ζ^i_{k+1} and η^i_{k+1} via (32)–(35)
4. Determine local prediction error by (38) and apply (31) and (36) to calculate the updated P^i_{k+1} and Θ^i_{k+1}, respectively. Thus the updated regression model is obtained by (1)

Table 3 Comparison between online methods in the framework of recursive kernel learning

Methods	System	Finding optimal regularization parameter and kernel's parameter	Sparsification strategy	Nodes pruning	Type of estimation
OW-LSSVM (Tang et al. 2006)	SISO	Pre-defined	Moving window	Smallest Lagrange multiplier	Global
RR-LSSVR (Yongping Zhao 2009)	SISO	Grid search	Optimal reduced strategy	FLOO	Global
GPC (Li et al. 2007)	SISO	Pre-defined	Moving window	Smallest Lagrange multiplier	Global
SRKL (Liu et al. 2010)	MIMO	Pre-defined	Prediction error	FLOO	Global
LW-LSSVM (Liu et al. 2007)	SISO	Pre-defined	Moving window	Similarity measure	Local
eTS-LS-SVM (this study)	SISO	Adaptive	Moving window	FLOO	Local

Table 4 Comparison between online methods' in the framework of functional fuzzy systems

Methods	Premise parameter identification	Consequence model	Consequence parameter identification	No. of parameters	Accuracy	Speed
DENFIS (Kasabov and Song 2002)	Evolving clustering	Linear	RLS	Very high	High	Slow
FLEXFIS (Lughofer 2008)	VQ-INC-EXT clustering	Linear	RLS	Medium	Medium	Medium
eTS (Angelov and Zhou 2006)	Incremental clustering	Linear	RLS	High	High	Medium
eACM (Martinez et al. 2008)	Incremental clustering	Linear	RLS	Low	Low	Fast
IHFC (Kim et al. 2007)	IHFC	Linear	RLS	High	High	Slow
ANF-RSA (Mirmomeni et al. 2011)	Incremental clustering	Linear	RLS	Low	High	Fast
ENFM (Soleimani-B et al. 2010)	Recursive Gath–Geva clustering	Linear	RLS	Low	High	Fast
IFC (Kalhor et al. 2009)	Indirect fuzzy clustering	Linear	RLS	Low	High	Fast
AHLTNM (Kalhor et al. 2010)	Incremental clustering	Linear	RLS	Low	High	Fast
eTS-LS-SVM (fast) (this study)	Incremental clustering	Nonlinear	RLS-SVM	Low	High	Fast
eTS-LS-SVM (accurate) (this study)	Incremental clustering	Nonlinear	RLS-SVM	High	Very high	Slow

corresponds to the ith local model at the kth iteration of online learning.

Similarly, we can implement the above equations to find a sub-optimal regularization parameter during the learning process:

$$(\gamma^{-1})^i_{k+1} = (\gamma^{-1})^i_k + \overline{\mathrm{G}}^i_{k+1}\overline{J}^i_{k+1}e^i_{k+1} \tag{43}$$

$$\overline{\mathrm{G}}^i_{k+1} = \left[\overline{\mathrm{G}}^i_k - \frac{\overline{\mathrm{G}}^i_k\overline{J}^i_{k+1}(\overline{J}^i_{k+1})^{\mathrm{T}}\overline{\mathrm{G}}^i_k}{1+(\overline{J}^i_{k+1})^{\mathrm{T}}\overline{\mathrm{G}}^i_k\overline{J}^i_{k+1}}\right] \tag{44}$$

$$\begin{aligned}\overline{J}^i_{k+1} &= -\frac{\partial e^i_{k+1}}{\partial(\gamma^{-1})^i_{k+1}} \\ &= \Phi(x_{k+1})(\alpha^i_{k+1})^T K^i_{k+1}\end{aligned} \tag{45}$$

where $K^i_{k+1} = [\exp(-\frac{\|x_{k+1}-x^i_k(1)\|^2}{(\sigma^i_k(1))^2}),\ldots,\exp(-\frac{\|x_{k+1}-x^i_k(N)\|^2}{(\sigma^i_k(N))^2})]$ and e^i_{k+1} is the local approximation error which is introduced in (38).

Finally, the process of eTS-LS-SVM learning algorithm is depicted in Fig. 2.

5 Case study

In this section, in order to show the performance of the proposed method in online prediction of nonlinear time series, prediction of some nonlinear time series is considered.

5.1 Prediction of Mackey–Glass time series

Prediction of Mackey–Glass time series has recently been used as a benchmark problem for comparison between the efficiency of different online modeling approaches. It serves as a model for white blood cell production, and is defined by Soleimani-B et al. (2010):

$$\frac{dx(t)}{dt} = \beta x(t) + \frac{\alpha x(t)}{1+x^{10}(t-\tau)} \tag{46}$$

Time series is obtained using (46) while the parameters are assumed to be $\alpha = 0.2$, $\beta = -0.1$, $\tau = 17$ and $x_0 = 1.2$. The objective of this modeling process is to

predict values of $x(t + 85)$ from available input vectors $[x(t - 18), x(t - 12), x(t - 6), x(t)]$. A training data set of 3,000 data samples (from $t = 201$ to 3,200) is used for structure and parameter identification of the model. The constructed model is tested by 500 data points (from $t = 5{,}001$ to 5,500). In order to solve the considered problem, a covariance resetting method is applied as a rescue mechanism to ensure that the covariance matrix does not become rank deficient.

Figure 3 shows the performance of the eTS-LS-SVM algorithm in the 85-step ahead prediction of Mackey–Glass chaotic time series on the training data set. Besides, this figure shows the number of clusters (local models) during the operation of eTS-LS-SVM algorithm. As shown in Fig. 3, when the proposed method is applied, final structure of the model is maximally comprised of four local models. The suggested model is applied to the standard test case data set. Figure 4 shows the 85-step ahead prediction of Mackey–Glass time series on the test data.

The importance of the optimal number of support vectors cannot be neglected in a prediction problem. In practice, the proper window size for presented method depends on the trade-off between the accuracy of the model and the computation time (and storage resources). In this section, two window sizes (large and small) are used to show how the window size can affect the performance and computation time of the suggested method.

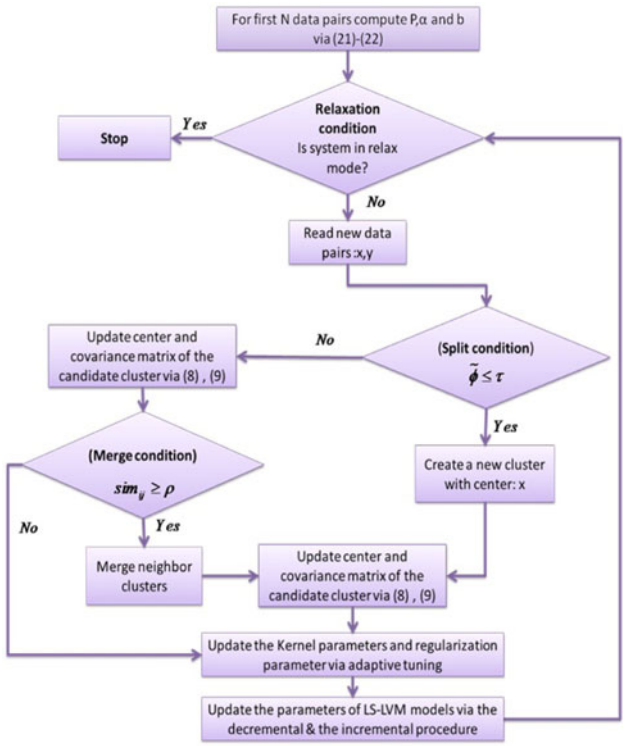

Fig. 2 Process of eTS-LS-SVM learning algorithm

Threshold values for creating new local models (τ) and for merging similar clusters (ρ) are set to 10^{-7} and 8×10^{-8}, respectively. We have used two sizes of support vectors as the moving window size. First, for fast eTS_LSSVM, $N = 4$, $\sigma_1^1 = 0.9$, $(\gamma_1^1)^{-1} = 1/2{,}800$ and $\sigma_0 = 0.004$ are used. Also, for accurate eTS_LSSVM, $N = 140$, $\sigma_1^1 = 0.175$, $(\gamma_1^1)^{-1} = 1/900$ and $\sigma_0 = 0.04$, are used.

The overall performance of the model is measured by non-dimensional error index (NDEI), which is defined as the root mean-square error divided by the standard deviation of the original time series (Kasabov and Song 2002).

The output NDEI for the test data using fast eTS-LS-SVM (with small window size) is 0.0811 and the corresponding execution time is 4.62(s). For accurate eTS-LS-SVM, which has 140 nodes, the output NDEI is equal to 0.0291 and the corresponding execution time is 30.24(s). This relies on fact that choosing the window size is the trade-off between sparseness and accuracy. Table 5 shows the number of evolving rules and values of NDEI obtained from different modeling methods. As indicated in Table 5, the superiority of our method over the other on-line methods in terms of having fewer number of local models and less modeling error (NDEI) can be observed.

5.2 Prediction of sunspot time series

Online prediction of sunspot time series is another concern of this section. Since solar activities have significant effects on the various aspects of life, such as climate, weather, ecosystem etc., different offline and online methods have attached great importance to the prediction of sunspot number as one of the most important features of solar activity (Mirmomeni et al. 2011; Kalhor et al. 2009;

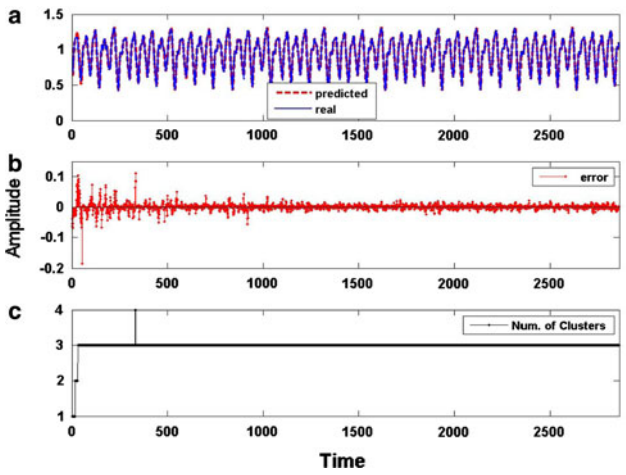

Fig. 3 85-Step ahead prediction of Mackey–Glass time series by the proposed method; **a** *solid* the real index values, *dashed line* prediction via eTS-LS-SVM algorithm; **b** 85-step ahead prediction error; **c** number of clusters

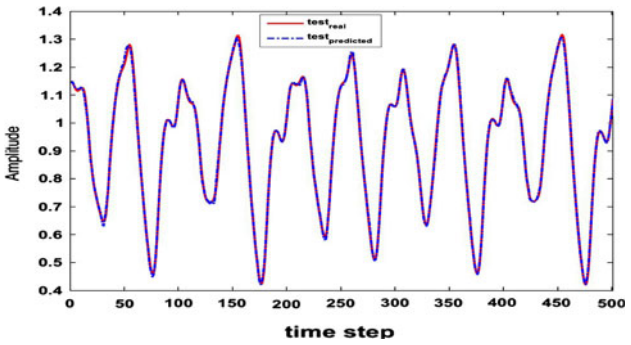

Fig. 4 85-Step ahead prediction of Mackey–Glass time series by the proposed method for test data

Soleimani-B et al. 2010). Due to the inherent complexity and nonlinear behaviour of the sunspot time series, prediction of sunspot number can well verify the proper performance of our proposed method. Time series data set is available from the Solar Influences Data Analysis Center (SIDC 2003). We have used a given data from 1900 until January of 1999 as train data to design a model for the 32-step ahead prediction on the sunspot time series. Furthermore, data points from January 1999 to May 2001 have been used for testing the efficiency of the trained model. The normalized mean squared error (NMSE) index is considered to compare the prediction results of our proposed method with some of the other approaches.

$$\text{NMSE} = \left(\frac{\sum_{i=1}^{Q}(y_i - \hat{y}^i)^2}{\sum_{i=1}^{Q}(y_i - \overline{y})^2}\right), \quad \overline{y} = \frac{\sum_{i=1}^{Q} y_i}{Q} \tag{47}$$

In this case study (i.e. prediction of sunspot time series), $x_k = [y_{k-1}, \ldots, y_{k-26}]$ and y_k are considered as input and output of the model, respectively. The threshold values of creating new local models (τ) and merging similar clusters (ρ) are set to be 10^{-6} , 7×10^{-7}, respectively, indicating that fuzzy clusters have a less common border. Also, similar to the previous case study, we have used two sizes of support vectors as moving window size. First, for the fast eTS_LSSVM, we have used $N = 20$, $\sigma_1^1 =$ 1,300, $(\gamma_1^1)^{-1} = 1/26{,}000$ and $\sigma_0 = 2{,}700$. For accurate eTS_LSSVM, $N = 150$, $\sigma_1^1 = 300$, $(\gamma_1^1)^{-1} = 1/5{,}000$ and $\sigma_0 = 500$ are used. Results are shown in Fig. 5. Moreover, Fig. 6 illustrates the performance of the accurate eTS-LSSVM for prediction of the test data set. It is clear from Table 6 that our online method has the advantage of fewer number of rules as well as lower prediction error (NMSE), compared to the other two methods mentioned in this table.

5.3 Short term load forecasting

Increasing growth in electricity energy consumption in recent years has made load forecasting an important issue. Numerous methods and algorithms have been presented for load forecasting in the literature. Accurate load forecasting can provide a better control and scheduling of the power systems, develop optimal energy planning, and enhance the power system planning. During the past years, several techniques have been proposed to deal with offline and online short term load forecasting problems (Pandian et al. 2006; Maia and Goncalves 2009). In this section, we aim to implement the proposed method in online forecasting of the electrical load consumption of Tehran in 2002 (courtesy of Iran Power Generation, Transmission and Distribution Management Company for the use of data). Used

Table 5 Comparison of eTS-LS-SVM with other evolving methods in prediction of Mackey–Glass time series

Methods	Number of rules	NDEI	NMSE	Number nodes (window-size)	Average execution time per sample[a] (s)	Plotting
DENFIS (Kasabov and Song 2002)	58	0.276	0.0763		–	
DENFIS (Kasabov and Song 2002)	883	0.042	0.0018		–	
FLEXFIS (Lughofer 2008)	69	0.206	0.0425		–	
FLEXFIS (Lughofer 2008)	89	0.157	0.0247		–	
eTS (Angelov and Zhou 2006)	113	0.095	0.0090		–	
eACM (Martinez et al. 2008)	25	0.223	0.0498		–	
IHFC (Kim et al. 2007)	104	0.0798	0.0064		–	
rGK (Dovžan and Škrjanc 2011)	100	0.1166	0.0207		–	
eF-OP-ELM (Pouzols and Lendasse 2010)	50	0.238	0.0463		0.11	
ENFM (Soleimani-B et al. 2010)	8	0.0652	0.0043			
AHLTNM (Kalhor et al. 2010)	10	0.0548	0.0030		0.011	
eTS-LS-SVM (fast) (this study)	4	0.0811	0.0066	4	0.009	
eTS-LS-SVM (accurate) (this study)	4	0.0291	0.0008	140	0.06	Figures 3 and 4

[a] The given execution times are measured by using a DELL-M1530 notebook

data set contained 1,000 hourly electric load time series: $y_k(t = 1, \ldots, 1{,}000)$. The proposed method is to perform one-step ahead prediction, i.e. prediction of next hour load where y_k is predicted based on x_k, which is defined as the last 24 values (a day) of time series:

$$x_k = [y_{k-1}, \ldots, y_{k-24}] \tag{48}$$

Data points of the last 14 days (336 h) of the considered data set are utilized to evaluate the efficiency of the trained models. Hourly electrical time series have drawn from continuous nonlinear time-varying system and as explained previously, the global LS-SVM models have some shortcoming facing such systems (Liu et al. 2007). So, we are to implement our local method to show its superior performance in dealing with such a problem. By changing initial deviation of new clusters one can control the number of rules in the proposed online learning procedure. Learning with only one rule corresponds with global LS-SVM and taking this assumption into account, we can represent a comparison between global LS-SVM and eTS-LS-SVM as local learning method.

Both global LS-SVM and eTS-LS-SVM methods are applied to test data and their accuracy in the load forecasting are evaluated based on the NMSE criterion.

The results of Table 7 show that the presented local LS-SVM outperforms the global LS-SVM (eTS-LS-SVM approach with one rule(cluster)) in terms of prediction error. If the number of local model increases excessively, then the model becomes over-parameterized. Resultantly, the prediction error goes up. It should be noted that, if the time series is nonlinear, then a TS fuzzy model with more local models should be used to trace the process dynamics. To obtain a more accurate model, pre-knowledge about the problem is required.

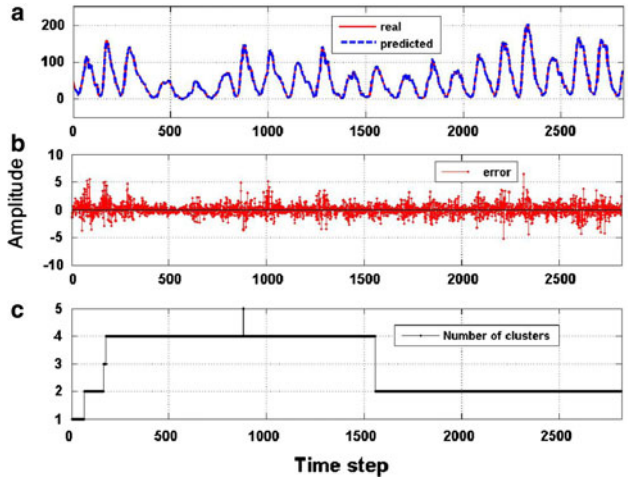

Fig. 5 Training of eTS-LSSVM (accurate) for sunspot time series prediction; **a** predicted and real data points; **b** prediction error; **c** number of local models

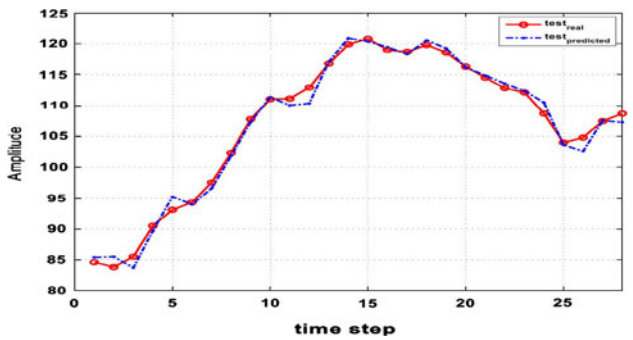

Fig. 6 The predicted and original test data points of sunspot time series

6 Conclusion

This paper has proposed a new approach for online prediction of time series based on the evolving Takagi–Sugeno least squares support vector machine (eTS-LS-SVM). An efficient incremental technique which consists of two phases: the split phase and the merge phase is used for structure identification of TS model, so that more compact set of rules can be provided and the rule interpretability is improved. The main contributions of our study are the use of LS-SVM models in the consequent part (local nonlinear models) instead of traditional local linear models, and developing a new recursive parameter adaptation technique for updating the parameters of the local models. Furthermore, in order to recursively update the meta-parameters of the LS-SVM models (regularization parameter and kernel

Table 6 Comparison of eTS-LS-SVM with other evolving methods in prediction of sunspot time series

Methods	No. of rules	NDEI	NMSE	No. of nodes (window size)	Average execution time per sample (Second)	Plotting
IFC (Kalhor et al. 2009)	6	0.1026	0.0109		0.77	
ENFM (Soleimani-B et al. 2010)	8	0.1603	0.0266			
eTS-LS-SVM (fast)	5	0.1393	0.0201	20	0.83	
eTS-LS-SVM (accurate)	5	0.0968	0.0097	150	1.6	Figures 5 and 6

Table 7 Results for comparison between local recursive LS-SVM and global one

Methods	Number of rules	No. of nodes (window-size)	NMSE	Plotting
eTS-LS-SVM (global)	1	100	0.0661	
eTS-LS-SVM (local)	2	100	0.0653	
eTS-LS-SVM (local)	3	100	0.063	Figure 7
eTS-LS-SVM (local)	4	100	0.0655	
eTS-LS-SVM (local)	5	100	0.1869	

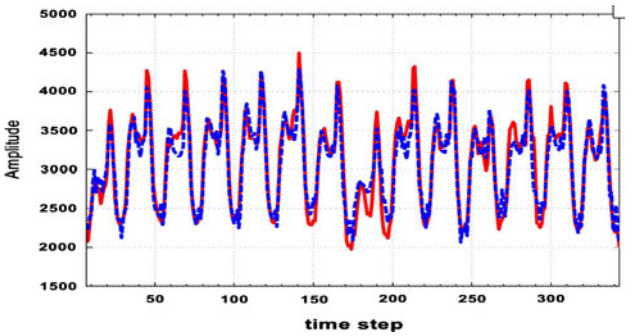

Fig. 7 Actual loads and load forecasting of test data points

parameter), a novel formulation based on a recursive gradient-based method is devised in which the parameters can be tuned adaptively as the online procedure goes on. Due to its high generalization ability, the use of LS-SVMs as the local nonlinear models can provide more local fitness and less local redundancies, compared to the methods using local linear models in the consequent part of TS model. The issue is then verified by implementation of the proposed method in two case studies (i.e. prediction of Mackey–Glass time series and sunspot indexes). By a third case study (i.e. short term load forecasting), it has been shown that local LS-SVMs may produce better results in terms of less prediction error in prediction of the nonlinear time-varying time-series, compared to the cases using LS-SVM as a global identification procedure.

Acknowledgments The authors would like to express their gratitude to Mr. Mojtaba Kharrasi for help in proof reading and text edition.

Appendix

Proof First consider $Y_K^i = [0 \quad y_k^i(1) \ldots y_k^i(N-1)]^T$ where $(N-1)$ denotes the size of moving-windows for the ith local model after the pruning stage, y_{k+1} is new added output then we can derive (36) as following:

$$
\begin{aligned}
\Theta_{k+1}^i &= P_{k+1}^i \begin{bmatrix} Y_k^i \\ y_{k+1} \end{bmatrix} \\
&= \begin{bmatrix} P_{pr}^i - \eta_{k+1}^i Z_{k+1}^i (Z_{k+1}^i)^T & \eta_{k+1}^i Z_{k+1}^i \\ \zeta_{k+1}^i \psi_{k+1}^i \left[\eta_{k+1}^i Z_{k+1}^i (Z_{k+1}^i)^T - P_N\right] & -\eta_{k+1}^i \end{bmatrix} \begin{bmatrix} Y_k^i \\ y_{k+1} \end{bmatrix} \\
&= \begin{bmatrix} P_N Y_k^i - \eta_{k+1}^i Z_{k+1}^i (Z_{k+1}^i)^T Y_k^i + \eta_{k+1}^i Z_{k+1}^i y_{k+1} \\ \zeta_{k+1}^i \psi_{k+1}^i \eta_{k+1}^i Z_{k+1}^i (Z_{k+1}^i)^T Y_k^i - \zeta_{k+1}^i (Z_{k+1}^i)^T Y_k^i - \eta_{k+1}^i y_{k+1} \end{bmatrix} \\
&= \begin{bmatrix} \Theta_{pr}^i + \eta_{k+1}^i Z_{k+1}^i (y_{k+1} - (Z_{k+1}^i)^T Y_k^i) \\ (\zeta_{k+1}^i \eta_{k+1}^i (\eta_{k+1}^i)^{-1} - \zeta_{k+1}^i)(Z_{k+1}^i)^T Y_k^i \\ -\eta_{k+1}^i (y_{k+1} - (Z_{k+1}^i)^T Y_k^i) \end{bmatrix} \\
&= \begin{bmatrix} \Theta_{pr}^i + \eta_{k+1}^i Z_{k+1}^i e_{k+1}^i \\ (\zeta_{k+1} - \zeta_{k+1})(Z_{k+1}^i)^T Y_k^i - \eta_{k+1} e_{k+1}^i \end{bmatrix} \\
&= \begin{bmatrix} \Theta_{pr}^i + \eta_{k+1}^i Z_{k+1}^i e_{k+1}^i \\ -\eta_{k+1}^i e_{k+1}^i \end{bmatrix}
\end{aligned} \tag{49}
$$

where e_{k+1}^i is the prediction error computed by the difference between the desired signal and the output of the ith local model after the pruning stage (Dovžan and Škrjanc 2011).

References

Abonyi J, Babuska R (2000) Local and global identification and interpretation of parameters in Takagi–Sugeno fuzzy models. In: Proceedings IEEE international conference on fuzzy systems, pp 835–840

An S, Liu W, Venkatesh S (2007) Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression. Pattern Recognit 40(8):2154–2162

Angelov P, Filev D (2004) An approach to online identification of Takagi–Sugeno fuzzy models. IEEE Trans Syst Man Cybern B 34(1):484–498

Angelov PP, Filev D (2006) Simple_ eTS: a simplified method for learning evolving Takagi–Sugeno fuzzy models. In: Proceedings of the 11th IEEE international conference on fuzzy systems, pp 1068–1073

Angelov PP, Zhou X (2006) Evolving fuzzy systems from data streams in real-time. In: IEEE symposium on evolving fuzzy systems, Ambleside, Lake District, UK, pp 29–35

Angelov PP, Filev D, Kasabov NK (2008) Evolving fuzzy systems—preface to the special section. IEEE Trans Fuzzy Syst 16(6): 1390–1392

Cheng WY, Juang CF (2011) An incremental support vector machine-trained TS-type fuzzy system for online classification problems. Fuzzy Sets Syst 163(1):24–44

Chi HM, Ersoy KO (2003) Recursive update algorithm for least squares support vector machines. Neural Process Lett 17:165–173

Diehl CP, Cauwenberghs G (2003) SVM incremental learning, adaptation and optimization. Proc Int Jt Conf Neural Netw Boston 4(1–4):2685–2690

Dovžan D, Škrjanc I (2011) Recursive clustering based on a Gustafson-Kessel algorithm. Evol Syst 2:15–24

Duda RO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. Wiley, New York

Engel Y, Mannor S, Meir R (2004) The kernel recursive least-squares algorithm. IEEE Trans Signal Process 52(8):2275–2285

Kalhor A, Araabi BN, Lucas C (2009) Online identification of a neuro-fuzzy model through indirect fuzzy clustering of data space. In: IEEE international conference on fuzzy systems, Korea

Kalhor A, Araabi BN, Lucas C (2010) An online predictor model as adaptive habitually linear and transiently nonlinear model. Evol Syst 1:29–41

Kasabov NK, Song Q (2002) DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction. IEEE Trans Fuzzy Syst 10(2):144–154

Kim CH, Kim MS, Lee JJ (2007) Incremental hyperplane-based fuzzy clustering for system modeling. In: Proceedings of 33rd conference of IEEE industrial electronics society, Taipei, Taiwan

de Kruif B, de Vries T (2003) Pruning error minimization in least squares support vector machines. IEEE Trans Neural Netw 14(3):696–702

Leuven KU, Suykens JAK, Lukas L (2000) Sparse least squares support vector machine classifiers, Suykens JAK, Lukas L and Vandewalle J. In: Neural processing letters, pp 293–300

Li L, Yu H, Liu J, Zhang S (2010) Local weighted LS-SVM online modeling and the application in continuous processes. In: Wang F, Deng H, Gao Y, sheng Lei J (eds) Artificial intelligence and computational intelligence. Lecture notes in computer science, vol 6320. Springer, Berlin, pp 209–217

Li LJ, Su HY, Chu J (2007) Generalized predictive control with online least squares support vector machines. Acta Autom Sin 33(11):1182–1188

Lin CJ, Chen CH, Lin CT (2011) An efficient evolutionary algorithm for fuzzy inference systems. Evol Syst 2:83–99

Liu W, Park I, Wang Y, Príncipe JC (2009) Extended kernel recursive least squares algorithm. IEEE Trans Signal Process 57(10): 3801–3814

Liu Y, Wang H, Li P (2007) Local least squares support vector regression with application to online modeling for batch processes. J Chem Ind Eng 58:2846–2851

Liu Y, Wang H, Yu J, Li P (2010) Selective recursive kernel learning for online identification of nonlinear systems with NARX form. J Process Control 20(2):181–194

Lughofer E (2008) FLEXFIS: a robust incremental learning approach for evolving Takagi–Sugeno fuzzy models. IEEE Trans Fuzzy Syst 16(6):139–1410

Lughofer E, Klement E (2005) FLEXFIS: a variant for incremental learning of Takagi–Sugeno fuzzy systems. In: Proceedings of FUZZ-IEEE, Reno, Nevada, USA, pp 915–920

Lughofer E, Bouchot JL, Shaker A (2011) On-line elimination of local redundancies in evolving fuzzy systems. Evol Syst. doi: 10.1007/s12530-011-9032-3

Maia C, Goncalves M (2009) A methodology for short-term electric load forecasting based on specialized recursive digital filters. Comput Ind Eng 57(3):724–731

Martinez B, Herrera F, Fernandez J, Marichal E (2008) An incremental clustering method and its application in online fuzzy modeling. Stud Fuzziness Soft Comput 224:163–178

Mirmomeni M, Lucas C, Araabi B, Moshiri B, Bidar M (2011) Online multi-step ahead prediction of time-varying solar and geomagnetic activity indices via adaptive neurofuzzy modeling and recursive spectral analysis. Sol Phys 272:189–213

Ngia LS, Sjóberg J, Viberg M (1998) Adaptive neural nets filter using a recursive Levenberg-Marquardt search direction. In: Proceedings of the 32nd asilomar conference on signals, systems and computers, pp 697–701

Pandian SC, Duraiswamy K, Rajan CCA, Kanagaraj N (2006) Fuzzy approach for short term load forecasting. Electr Power Syst Res 76(6–7):541–548

Pouzols F, Lendasse A (2010) Evolving fuzzy optimally pruned extreme learning machine for regression problems. Evol Syst 1:43–58

Ramos JV, Pereira C, Dourado A (2010) The building of interpretable systems in real-time. In: Angelov PP, Filev D, Kasabov N (eds) Evolving intelligent systems: methodology and applications. Wiley, New York, pp 127–150

Smola AJ, Schölkopf B (2004) A tutorial on support vector regression. Stat Comput 14:199–222

Soleimani-B H, Lucas C, Araabi BN (2010) Recursive Gath-Geva clustering as a basis for evolving neuro-fuzzy modeling. Evol Syst 1:59–71

Suykens J, Brabanter JD, Lukas L, Vandewalleb J (2002) Weighted least squares support vector machines: robustness and sparse approximation. Neurocomputing 48(1–4):85–105

Suykens JAK, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Process Lett 9:293–300

Tang HS, Xue ST, Chen R, Sato T (2006) Online weighted LS-SVM for hysteretic structural system identification. Eng Struct 28(12):1728–1735

Vapnik V (1995) The nature of statistical learning theory. Springer, New York

Yamauchi K (2010) Incremental model selection and ensemble prediction under virtual concept drifting environments. Evol Syst 6230:570–582

Yongping Zhao JS (2009) Recursive reduced least squares support vector regression. Pattern Recognit 42:837–842