Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Adaptive online prediction method based on LS-SVR and its application in an electronic system*

Yang-ming GUO†, Cong-bao RAN, Xiao-lei LI, Jie-zhong MA

(*School of Computer Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China*)

†E-mail: yangming_g@nwpu.edu.cn

Received May 24, 2012; Revision accepted Sept. 20, 2012; Crosschecked Nov. 12, 2012

Abstract: Health trend prediction has become an effective way to ensure the safe operation of highly reliable systems, and online prediction is always necessary in many real applications. To simultaneously obtain better or acceptable online prediction accuracy and shorter computing time, we propose a new adaptive online method based on least squares support vector regression (LS-SVR). This method adopts two approaches. One approach is that we delete certain support vectors by judging the linear correlation among the samples to increase the sparseness of the prediction model. This approach can control the loss of useful information in sample data, improve the generalization capability of the prediction model, and reduce the prediction time. The other approach is that we reduce the number of traditional LS-SVR parameters and establish a modified simple prediction model. This approach can reduce the calculation time in the process of adaptive online training. Simulation and a certain electric system application indicate preliminarily that the proposed method is an effective prediction approach for its good prediction accuracy and low computing time.

Key words: Adaptive online prediction, Least squares support vector regression (LS-SVR), Electronic system
doi:10.1631/jzus.C1200156 **Document code:** A **CLC number:** TP391

1 Introduction

In recent years, health condition diagnosis and prognosis technology has been an effective technique for the reliability and safety of complex systems. With the wide use of electronic systems in complex systems, electronic systems have become the key to the reliability of the whole system. According to the data collected from the electronic system, we can infer the health condition, determine the fault location, analyze the health condition, predict the future failure, and then give the necessary maintenance tips.

Based on statistical theory and the structural risk minimization principle (Vapnik, 1995), support vector machine (SVM), which characterizes a simple model structure, has excellent classification and generalization ability in solving learning problems (Vapnik, 1998). Its hidden layer can be adjusted automatically so that the complexity of machine learning can be consistent with the actual problem. SVM-based methods have received extensive attention in electronic system health condition diagnosis and prognosis.

For regression or prediction, the SVM-based model is called support vector regression (SVR). SVR is determined endogenously as part of the optimization problem, which has a unique solution of a (convex) quadratic programming (QP) problem. The solution of the QP problem provides the necessary information for choosing the most important data points, called support vectors. Based on SVR formulation, SVR uniquely defines the estimated regression function. SVR exhibits better accuracy in nonlinear and

* Project supported by the National Basic Research Program (973) of China, the National Natural Science Foundation of China (Nos. 61001023 and 61101004), the Basic Research Program of Shaanxi Province, China (No. 2010JQ8005), and the Aviation Science Fund of China (No. 2010ZD53039)

non-stationary prediction via the kernel function in practice (Pang *et al*., 2011). For large sample data, however, the QP problem is more complex and will cost much more computing time. Thus, least squares support vector regression (LS-SVR) was proposed by Suykens and Vandewale (1999). As a well-known supervised machine learning technique, LS-SVR has been applied successfully to prediction applications (Suykens *et al*., 2002; Guo *et al*., 2009; Guo and Guan, 2010; Zhao *et al*., 2011; Qu and Zuo, 2012). In LS-SVR, the loss function is replaced by the equality constraints to accelerate the training process, and fewer tuning parameters are involved. So, the training time of LS-SVR is improved greatly. However, in practice of online prediction using LS-SVR, we should consider the prediction accuracy and time cost synchronously. According to the new information principle (Deng, 2004), researchers used an adaptive algorithm (Zhang and Wang, 2006); i.e., they removed the data located farthest from the prediction point when new data was added to the sample data set, to improve prediction accuracy by enriching the prediction information and reduce computing time by keeping the amount of training sample data.

One drawback should be considered; i.e., all the training sample data is selected as support vectors in LS-SVR. This will lead to redundant information entirely fitting to LS-SVR and lack of sparseness compared with SVR (Müller *et al*., 1997; Schölkoph, *et al*., 1999). These influence the generalization capability of LS-SVR and the training complexity, giving rise to increase in prediction time. Moreover, the low generalization capability will cause low prediction accuracy in practice, because many uncertain factors exist in the sample data, such as noise.

Aiming at the existing problems of adaptive online prediction based on LS-SVR, we propose a new regression scheme, which modifies the traditional LS-SVR to a more simple form and applies a new method based on linear correlation to prune the support vectors. A sparseness method is used to improve the generalization capability of the prediction model and reduce the prediction time, and a new simple prediction model is used to reduce the adaptive calculation time in training. This way, the increase in training time due to the sparseness process can be avoided. Simulation and a certain electronic system application indicate that the proposed scheme satisfies the requirements of actual applications, including good accuracy and less computing time.

2 Least squares support vector regression

Given the labeled data set $S=\{(\boldsymbol{x}_i, y_i)|i=1, 2, \ldots, n\}$, where $\boldsymbol{x}_i\in\mathbb{R}^d$ is the input sample, and $y_i\in\mathbb{R}$ is the corresponding output label or target value, LS-SVR begins with approximating an unknown function in the primal feature space using the following equation:

$$f(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\varphi(\boldsymbol{x})+b, \tag{1}$$

where $\varphi(\cdot)$: $\mathbb{R}^d\rightarrow F$ is a nonlinear function which maps the input data to a higher dimensional feature space, $\boldsymbol{w}$ is the coefficient value, and b is the bias term. The idea of LS-SVR is to estimate $\boldsymbol{w}$ and b that optimize the generalization ability of the repressor by minimizing the regularized loss function.

LS-SVR preserves the following characteristics: the error variable e_i is used to control the deviations from the regression function, and a squared loss function is used to replace the insensitive loss function. This way, the following optimization problem is given:

$$\begin{aligned}&\min J(\boldsymbol{w},b)=\frac{1}{2}\|\boldsymbol{w}\|^2+\frac{c}{2}\sum_{i=1}^{n}e_i^2\\&\text{s.t. } y_i=f(\boldsymbol{x}_i)=\boldsymbol{w}^{\mathrm{T}}\varphi(\boldsymbol{x}_i)+b+e_i,\ i=1,2,...,n,\end{aligned} \tag{2}$$

where the parameter c determines the trade-off between the model complexity and the goodness of fit to the sample data. To solve this optimization problem, its Lagrangian function is constructed as follows:

$$\begin{aligned}L(\boldsymbol{w},b,e,\alpha)=&\frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}+\frac{c}{2}\sum_{i=1}^{n}e_i^2\\&-\sum_{i=1}^{n}\alpha_i(\boldsymbol{w}^{\mathrm{T}}\varphi(\boldsymbol{x}_i)+b+e_i-y_i),\end{aligned} \tag{3}$$

where α_i is the ith Lagrangian multiplier. Eq. (3) satisfies Karush-Kuhn-Tucker (KKT) conditions and its solution can be obtained by partial differential with respect to $\boldsymbol{w}$, b, e_i, and α_i, shown as follows:

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum\limits_{i=1}^{n} \alpha_i \varphi(\boldsymbol{x}_i) = \boldsymbol{0} \Rightarrow \boldsymbol{w} = \sum\limits_{i=1}^{n} \alpha_i \varphi(\boldsymbol{x}_i), \\ \dfrac{\partial L}{\partial b} = -\sum\limits_{i=1}^{n} \alpha_i = 0 \Rightarrow \sum\limits_{i=1}^{n} \alpha_i = 0, \\ \dfrac{\partial L}{\partial \alpha_i} = \boldsymbol{w}^{\mathrm{T}} \varphi(\boldsymbol{x}_i) + b + e_i - y_i = 0 \\ \qquad\qquad \Rightarrow y_i = \boldsymbol{w}^{\mathrm{T}} \varphi(\boldsymbol{x}_i) + b + e_i, \\ \dfrac{\partial L}{\partial e_i} = c e_i - \alpha_i = 0 \Rightarrow e_i = \dfrac{1}{c} \alpha_i. \end{cases} \tag{4}$$

After eliminating $\boldsymbol{w}$ and e_i from Eq. (4), we obtain the solution with the following matrix form:

$$\begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{1}_n^{\mathrm{T}} \\ \boldsymbol{1}_n & \boldsymbol{K}+\boldsymbol{I}/\mathrm{c} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{y} \end{bmatrix}, \tag{5}$$

where $\boldsymbol{1}_n$ is an n-dimensional vector of all ones, $\boldsymbol{I}$ is a unit matrix, $\boldsymbol{\alpha}=[\alpha_1, \alpha_2, \ldots, \alpha_n]^{\mathrm{T}}$, $\boldsymbol{y}=[y_1, y_2, \ldots, y_n]^{\mathrm{T}}$, and $K_{i,j}=k(\boldsymbol{x}_i, \boldsymbol{x}_j)=\varphi(\boldsymbol{x}_i)^{\mathrm{T}}\varphi(\boldsymbol{x}_j)$. Let $\boldsymbol{H}=\boldsymbol{K}+\boldsymbol{I}/c$. We have the following equations from Eq. (5):

$$\begin{cases} \boldsymbol{1}_n^{\mathrm{T}} \boldsymbol{\alpha} = 0, \\ \boldsymbol{1}_n b + \boldsymbol{H}\boldsymbol{\alpha} = \boldsymbol{y}. \end{cases} \tag{6}$$

Thus, Lagrangian dual variables $\boldsymbol{\alpha}$ and bias term b are obtained solely by

$$\begin{cases} \boldsymbol{\alpha} = \boldsymbol{H}^{-1}(\boldsymbol{y} - \boldsymbol{1}_n b), \\ b = \boldsymbol{1}_n^{\mathrm{T}} \boldsymbol{H}^{-1} \boldsymbol{y} (\boldsymbol{1}_n^{\mathrm{T}} \boldsymbol{H}^{-1} \boldsymbol{1}_n)^{-1}. \end{cases} \tag{7}$$

Any unlabeled input $\boldsymbol{x}$ can thus be estimated by the following function:

$$\hat{y}(\boldsymbol{x}) = f(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \varphi(\boldsymbol{x}_i)^{\mathrm{T}} \varphi(\boldsymbol{x}) + b = \sum_{i=1}^{n} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) + b. \tag{8}$$

Eq. (8) shows that the kernel function is a key issue which greatly influences the performance of LS-SVR. The kernel functions of LS-SVR are Mercer functions, which meet the Mercer condition. The main kernel functions used by LS-SVR include the polynomial kernel function, radial basis function (RBF) kernel function, and sigmoid kernel function. Due to the better goodness of fit and generalization ability, the Gaussian RBF function is one of the most common kernel functions, and we choose it as the kernel function in this study.

3 The proposed adaptive online prediction method

In many adaptive online prediction applications, fast and accurate prediction is the basic requirement. To meet the demand, we modify the traditional LS-SVR with a simple form and propose a novel method to prune the support vectors in this section.

3.1 Linear correlation based method of pruning support vectors

For the prediction with LS-SVR, because almost every sample data will be a support vector, LS-SVR has one main drawback, i.e., lack of sparseness. In order to obtain a sparseness solution, several methods (Suykens *et al*., 2000; Keerthi and Shevade, 2003; de Kruif and de Vries, 2003; Hoegaerts *et al*., 2004; Zeng and Chen, 2005; Jiao *et al*., 2007) were proposed. The basic idea of these schemes is to find the smaller indication value of the support vector first, and then remove the corresponding training samples. It is obvious that the idea of the reported schemes is simple, but all the schemes require computation duplication in solving the linear equations group (Chen *et al*., 2007; Zhou *et al*., 2009). For this reason, these schemes are not suitable for adaptive online prediction.

In this study, we consider removing the support vectors by judging the linear correlation among the sample data in the mapping space. This proposed method is based on the vector-based learning theory (Cawley and Talbot, 2002; Yaakov *et al*., 2002). It can be described as follows.

Suppose $\varphi(\boldsymbol{x}_k)$ is one of the n support vectors in the high-dimensional feature space. If and only if $\lambda_i=0$ ($i\neq k$), the equation $\varphi(\boldsymbol{x}_k) = \sum_{i=1, i\neq k}^{n} \lambda_i \varphi(\boldsymbol{x}_i)$ holds. Here $\boldsymbol{x}_k$ is called the base vector (BV) and the set consisting of $\boldsymbol{x}_k$ is called the base vector set (BVS). To reduce the number of support vectors and obtain a sparseness result in LS-SVR, any support vector that is linearly related to the BV can be deleted. Assume

that there are n support vectors in the sample data and that the BVS currently includes m support vectors ($m<n$). If the (m+1)th sample $\boldsymbol{x}_\tau$ cannot be represented by a linear combination of the m support vectors in the high-dimensional feature space, it will be added to the BVS; otherwise, it does not join in the BVS.

The requirement for strict linear dependence may lead to numerical instabilities (Yaakov *et al*., 2002). This means that the new sample data may not be absolutely linearly dependent on the existing support vectors. Thus, for the sample data $\boldsymbol{x}_\tau$, we will be content with finding coefficients $\lambda_{\tau,i}$ with at least one non-zero element satisfying the approximate linear dependence condition:

$$\begin{aligned}
&\delta_\tau=\left\|\sum_{i=1}^{m}\lambda_{\tau,i}\varphi(\boldsymbol{x}_i)-\varphi(\boldsymbol{x}_\tau)\right\|^2\le\upsilon\\
&\Rightarrow\delta_\tau=\sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_{\tau,i}\varphi(\boldsymbol{x}_i)\lambda_{\tau,j}\varphi(\boldsymbol{x}_j)^{\mathrm{T}}\\
&\qquad-2\varphi(\boldsymbol{x}_\tau)\sum_{i=1}^{m}\lambda_{\tau,i}\varphi(\boldsymbol{x}_i)^{\mathrm{T}}+\varphi(\boldsymbol{x}_\tau)\varphi(\boldsymbol{x}_\tau)^{\mathrm{T}}\le\upsilon \quad (9)\\
&\Rightarrow\delta_\tau=\sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_{\tau,i}\lambda_{\tau,j}k(\boldsymbol{x}_i,\boldsymbol{x}_j)\\
&\qquad-2\sum_{i=1}^{m}\lambda_{\tau,i}k(\boldsymbol{x}_\tau,\boldsymbol{x}_i)+k(\boldsymbol{x}_\tau,\boldsymbol{x}_\tau)\le\upsilon,
\end{aligned}$$

where υ is a small positive constant. Rewrite Eq. (9) to the following optimization problem based on the idea of Yaakov *et al*. (2002):

$$\begin{aligned}
\min_{\boldsymbol{\lambda}_\tau}\delta_\tau=&\sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_{\tau,i}\lambda_{\tau,j}k(\boldsymbol{x}_i,\boldsymbol{x}_j)-2\sum_{i=1}^{m}\lambda_{\tau,i}k(\boldsymbol{x}_\tau,\boldsymbol{x}_i)\\
&+k(\boldsymbol{x}_\tau,\boldsymbol{x}_\tau)\le\upsilon.
\end{aligned}\quad (10)$$

By minimizing Eq. (10), we can obtain the linear correlation coefficients $\boldsymbol{\lambda}_\tau=[\lambda_{\tau,1},\ \lambda_{\tau,2},\ \ldots,\ \lambda_{\tau,m}]$. To reduce the amount of BV, we may sometimes prefer to sacrificing the optimality of $\boldsymbol{\lambda}_\tau$. According to experience, let $\upsilon=\|\boldsymbol{\lambda}_\tau\|^2/(mc^2)$, which is an l_2 norm regularization. The minimization problem defined in Eq. (10) can be rewritten as

$$\begin{aligned}
\min_{\boldsymbol{\lambda}_\tau}\delta'_\tau=&\sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_{\tau,i}\lambda_{\tau,j}k(\boldsymbol{x}_i,\boldsymbol{x}_j)-2\sum_{i=1}^{m}\lambda_{\tau,i}k(\boldsymbol{x}_\tau,\boldsymbol{x}_i)\\
&+k(\boldsymbol{x}_\tau,\boldsymbol{x}_\tau)-\frac{\|\boldsymbol{\lambda}_\tau\|^2}{mc^2}\\
\text{s.t.}\quad&|\lambda_{\tau,i}|\le c,
\end{aligned}\quad (11)$$

where c is the same as the LS-SVR regularization parameter c. Solving Eq. (11) and if δ_τ' is larger than zero, add $\boldsymbol{x}_\tau$ to the BVS; otherwise, drop it.

Thus, we can draw a reasonable conclusion that the proposed scheme will hold back much of the information and will not lose too much useful information because it is based on linear correlation. So, the prediction accuracy will not be reduced largely while the model's generalization ability is improved. Compared with traditional LS-SVR, although this processing increases the calculation time, most parameter values have already been computed and stored in the process of setting up the prediction model. Therefore, the training time will not increase much, and the prediction time will be reduced greatly owing to fewer support vectors. Moreover, the proposed method can avoid a difficult problem in Yaakov's scheme, i.e., parameters selection, as no rules have been reported to solve the problem.

3.2 Modified simple prediction model based on LS-SVR

For the known adaptive online prediction methods based on LS-SVR, the parameters of LS-SVR, such as Lagrangian multipliers and the bias term, always need to be recalculated. To further reduce the computing time, based on Yaakov *et al*. (2002), we propose a simple LS-SVR based prediction model.

The scale of the training sample data remains unchanged for the adaptive algorithm; i.e., the sample data is always moved as a window. We use a window size of length l, and the training sample data is described by the input $\{(\boldsymbol{x}_i(t),y_i(t))\}_{i=t-l+1}^{i=t}$ ($\boldsymbol{x}_i\in\mathbb{R}^d$, $y_i\in\mathbb{R}$). At time t, give the sample data set $\{\boldsymbol{x}(t),\boldsymbol{y}(t)\}$, $\boldsymbol{x}(t)=[\boldsymbol{x}_{t-l+1}(t),\boldsymbol{x}_{t-l+2}(t),\ \ldots,\ \boldsymbol{x}_t(t)]$, $\boldsymbol{y}(t)=[y_{t-l+1}(t),y_{t-l+2}(t),\ \ldots,\ y_t(t)]^{\mathrm{T}}$. As time goes on, the new sample data is added and the oldest one is removed. In this subsection, we give an approach to eliminating the bias term b.

Let $\boldsymbol{w}(t)=[w(t)\ \ b/\gamma]^{\mathrm{T}}$ and $\boldsymbol{\phi}_i(t)=[\varphi_i(t)\ \ \gamma]^{\mathrm{T}}$, where γ is a constant and $\varphi_i(t)$ is the mapping function of the ith input data at time t. Then rewrite the optimization problem of Eq. (2) as follows:

$$\begin{aligned}
\min J(\boldsymbol{w},\boldsymbol{e})=&\frac{1}{2}\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{w}(t)+\frac{c}{2}\sum_{i=t-l+1}^{t}e_i^2(t)\\
\text{s.t.}\quad&y_i(t)=\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{\phi}_i(t)+e_i(t),\\
&i=t-l+1,\ t-l+2,\ldots,t.
\end{aligned}\quad (12)$$

The solution of Eq. (12) is obtained after constructing the Lagrangian function

$$L(\boldsymbol{w},\boldsymbol{e},a)=\frac{1}{2}\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{w}(t)+\frac{c}{2}\sum_{i=t-l+1}^{t}e_i^2(t)-\sum_{i=t-l+1}^{t}a_i(t)[\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{\phi}_i(t)+e_i(t)-y_i(t)], \tag{13}$$

where $a_i(t)$ is the ith Lagrangian multiplier at time t. The optimality conditions are shown as follows:

$$\begin{cases}\dfrac{\partial L}{\partial \boldsymbol{w}(t)}=\boldsymbol{w}(t)-\sum\limits_{i=t-l+1}^{t}a_i(t)\boldsymbol{\phi}_i(t)=\boldsymbol{0}\\ \qquad\Rightarrow \boldsymbol{w}(t)=\sum\limits_{i=t-l+1}^{t}a_i(t)\boldsymbol{\phi}_i(t),\\ \dfrac{\partial L}{\partial a_i(t)}=\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{\phi}_i(t)+e_i(t)-y_i(t)=0\\ \qquad\Rightarrow y_i(t)=\boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{\phi}_i(t)+e_i(t),\\ \dfrac{\partial L}{\partial e_i(t)}=ce_i(t)-a_i(t)=0\Rightarrow e_i(t)=\dfrac{1}{c}a_i(t).\end{cases} \tag{14}$$

The solution to Eq. (14) is given by Eq. (15) (at the bottom of this page) after eliminating $e_i(t)$ and $\boldsymbol{w}(t)$. Let

$$k_{ij}(t)=\boldsymbol{\varphi}_i(t)^{\mathrm{T}}\boldsymbol{\varphi}_j(t),\ \boldsymbol{H}(t)=\{H_{ij}(t)\}_{i,j=t-l+1}^{t}+\boldsymbol{I}/c,$$
$$\boldsymbol{y}(t)=[y_{t-l+1}(t),\ y_{t-l+2}(t),\ \cdots,\ y_{t-l+l}(t)]^{\mathrm{T}},$$
$$H_{ij}(t)=k_{ij}(t)+\gamma^2,$$
$$\boldsymbol{a}(t)=[a_{t-l+1}(t),\ a_{t-l+2}(t),\ \cdots,\ a_t(t)]^{\mathrm{T}}.$$

Eq. (15) can be rewritten as

$$\boldsymbol{a}(t)=\boldsymbol{y}(t)\boldsymbol{H}(t)^{-1}, \tag{16}$$

where $\boldsymbol{I}$ is an $l\times l$ identity matrix. Then the new regression function is presented as follows:

$$y_t(t+1)=y_{t+1}(t)=\sum_{i=t-l+1}^{t}a_i(t)[k(\boldsymbol{x}_i(t),\boldsymbol{x}_{t+1}(t))+\gamma^2]$$
$$=\sum_{i=t-l+1}^{t}a_i(t)[k(\boldsymbol{x}_i(t),\boldsymbol{x}_t(t+1))+\gamma^2]. \tag{17}$$

According to Eqs. (16) and (17), when the new sample data pair $(\boldsymbol{x}_{t+1}(t),\ y_{t+1}(t))$ or $(\boldsymbol{x}_t(t+1),\ y_t(t+1))$ is added and $(\boldsymbol{x}_{t-l+1}(t),\ y_{t-l+1}(t))$ is dropped, we do not need to compute b. This way, at time $t+1$, $\boldsymbol{a}$ can be calculated via Eq. (16), i.e., $\boldsymbol{a}(t+1)=\boldsymbol{y}(t+1)\boldsymbol{H}(t+1)^{-1}$. Here $\boldsymbol{H}(t)$ can be rewritten as

$$\boldsymbol{H}(t)=\begin{bmatrix}f(t) & \boldsymbol{F}(t)\\ \boldsymbol{F}(t)^{\mathrm{T}} & \boldsymbol{Q}(t)\end{bmatrix}, \tag{18}$$

where

$$f(t)=\boldsymbol{\varphi}_{t-l+1}(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+1}(t)+\gamma^2+\frac{1}{c},$$
$$\boldsymbol{F}(t)=[\boldsymbol{\varphi}_{t-l+2}(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+1}(t)+\gamma^2,\ ...,\ \boldsymbol{\varphi}_t(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+1}(t)+\gamma^2],$$
$$\boldsymbol{Q}(t)=\begin{bmatrix}\boldsymbol{\varphi}_{t-l+2}(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+2}(t)+\gamma^2+\dfrac{1}{c} & \cdots & \boldsymbol{\varphi}_t(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+2}(t)+\gamma^2\\ \vdots & & \vdots\\ \boldsymbol{\varphi}_{t-l+2}(t)^{\mathrm{T}}\boldsymbol{\varphi}_t(t)+\gamma^2 & \cdots & \boldsymbol{\varphi}_t(t)^{\mathrm{T}}\boldsymbol{\varphi}_t(t)+\gamma^2+\dfrac{1}{c}\end{bmatrix}.$$

Then

$$\boldsymbol{H}(t+1)=\begin{bmatrix}\boldsymbol{Q}(t) & \boldsymbol{V}(t+1)\\ \boldsymbol{V}(t+1)^{\mathrm{T}} & v(t+1)\end{bmatrix}, \tag{19}$$

where

$$\boldsymbol{V}(t+1)=\begin{bmatrix}\boldsymbol{\varphi}_t(t+1)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+2}(t)+\gamma^2\\ \boldsymbol{\varphi}_t(t+1)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+3}(t)+\gamma^2\\ \vdots\\ \boldsymbol{\varphi}_t(t+1)^{\mathrm{T}}\boldsymbol{\varphi}_t(t)+\gamma^2\end{bmatrix},$$

$$\begin{bmatrix}\boldsymbol{\varphi}_{t-l+1}(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+1}(t)+\gamma^2+\dfrac{1}{c} & \boldsymbol{\varphi}_{t-l+2}(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+1}(t)+\gamma^2 & \cdots & \boldsymbol{\varphi}_t(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+1}(t)+\gamma^2\\ \boldsymbol{\varphi}_{t-l+1}(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+2}(t)+\gamma^2 & \boldsymbol{\varphi}_{t-l+2}(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+2}(t)+\gamma^2+\dfrac{1}{c} & \cdots & \boldsymbol{\varphi}_t(t)^{\mathrm{T}}\boldsymbol{\varphi}_{t-l+2}(t)+\gamma^2\\ \vdots & \vdots & & \vdots\\ \boldsymbol{\varphi}_{t-l+1}(t)^{\mathrm{T}}\boldsymbol{\varphi}_t(t)+\gamma^2 & \boldsymbol{\varphi}_{t-l+2}(t)^{\mathrm{T}}\boldsymbol{\varphi}_t(t)+\gamma^2 & \cdots & \boldsymbol{\varphi}_t(t)^{\mathrm{T}}\boldsymbol{\varphi}_t(t)+\gamma^2+\dfrac{1}{c}\end{bmatrix}\begin{bmatrix}a_{t-l+1}(t)\\ a_{t-l+2}(t)\\ \vdots\\ a_t(t)\end{bmatrix}=\begin{bmatrix}y_{t-l+1}(t)\\ y_{t-l+2}(t)\\ \vdots\\ y_t(t)\end{bmatrix}. \tag{15}$$

$v(t+1)=\boldsymbol{\varphi}_t(t+1)^{\mathrm{T}}\boldsymbol{\varphi}_t(t+1)+\gamma^2+1/c$, $\boldsymbol{\varphi}_t(t+1)=\boldsymbol{\varphi}_{t+1}(t)$, and $\boldsymbol{H}(t+1)^{-1}$ can be computed quickly using the block matrix approach (Cowen, 1996).

It is obvious that the modified prediction model is simple, and it may be applied to reduce the computing time due to fewer parameters.

3.3 The proposed adaptive online prediction method

Following the analysis in Sections 3.1 and 3.2, we can obtain the regression function and set up the simple prediction method.

First, we choose two sample data corresponding to the two largest Lagrangian multipliers as the initial BV. When the rth support vector sample data $\boldsymbol{x}_r$ is mapped to the high-dimensional feature space, $\boldsymbol{\varphi}(\boldsymbol{x}_r)$ depends linearly on the other existing support vectors (Eq. (20)). This new support vector can be moved away.

$$\boldsymbol{\varphi}(\boldsymbol{x}_r)=\sum_{j=1}^{m}\lambda_j\boldsymbol{\varphi}(\boldsymbol{x}_j), \tag{20}$$

where λ_j is a constant coefficient, m is the number of independent support vectors contained in the BVS. Substituting Eq. (20) into Eq. (17), Eq. (17) can be rewritten as follows:

$$\begin{aligned} y_t(t+1) &= y_{t+1}(t) \\ &= \sum_{i=t-l+1,i\neq r}^{t} a_i(t)[k(\boldsymbol{x}_i(t),\boldsymbol{x}_{t+1}(t))+\gamma^2] \\ &\quad + a_r(t)[k(\boldsymbol{x}_r(t),\boldsymbol{x}_{t+1}(t))+\gamma^2] \\ &= \sum_{i=t-l+1,i\neq r}^{t} a_i(t)[k(\boldsymbol{x}_i(t),\boldsymbol{x}_t(t+1))+\gamma^2] \\ &\quad + a_r(t)[k(\boldsymbol{x}_r(t),\boldsymbol{x}_t(t+1))+\gamma^2] \\ &= \sum_{i=t-l+1,i\neq r}^{t} a_i(t)[k(\boldsymbol{x}_i(t),\boldsymbol{x}_t(t+1))+\gamma^2] \\ &\quad + a_r(t)\sum_{j=1}^{m}[\lambda_j k(\boldsymbol{x}_j(t),\boldsymbol{x}_{t+1}(t))+\gamma^2], \end{aligned} \tag{21}$$

where $\boldsymbol{x}_j$ is the sample data corresponding to BV. By inference from Eq. (21), we obtain a new regression function with the BV. It can be described as

$$\hat{y}(\boldsymbol{x})=f(\boldsymbol{x})=\sum_{i=1}^{m}\beta_i[k(\boldsymbol{x}_i,\boldsymbol{x})+\gamma^2]. \tag{22}$$

Here, β_i is combined with the Lagrangian multiplier and the linear combination coefficient.

In this case, we can use the modified simple prediction model and the proposed sparseness method to execute the adaptive online prediction. The prediction steps are described as follows:

1. Initialize t and choose appropriate basis kernels γ and window size l.

2. Compute $\boldsymbol{H}(t)$ and $\boldsymbol{a}(t)$.

3. Judge linear correlation and prune support vectors and compute the new BV-based Lagrangian multiplier $\boldsymbol{\beta}(t)$.

4. Set up the simple prediction model and predict $y_{t+1}(t)$ or $y_t(t+1)$.

5. Add $(\boldsymbol{x}_{t+1}(t), y_{t+1}(t))$ to the sample data and delete the oldest one, then let $t\leftarrow t+1$, and return to step 2.

To evaluate the prediction accuracy, we use the root mean square error (RMSE) metric:

$$\mathrm{RMSE}=\sqrt{\frac{1}{n}\sum_{k=1}^{n}(y_i(k)-\hat{y}_i(k))^2}, \tag{23}$$

where n is the size of the original test sample data and $y_i(k)$ and $\hat{y}_i(k)$ are the prediction and actual values, respectively.

4 Simulation and real application

To evaluate the efficiency of the proposed methods, we perform two simulations and an application experiment, using MatlabR2011b with LS-SVMlab1.8 Toolbox under the Windows XP operating system (The software and guide book can be downloaded from http://www.esat.kuleuven.be/sista/lssvmlab).

4.1 Simulation and results analysis

The two simulations are carried out to estimate our approaches presented in Sections 3.1 and 3.2. One is to test the proposed sparseness method based on linear correlation (Simulation 1), and the other is to test the modified simple prediction model (Simulation 2). All tests are run 50 times.

We perform the simulations with $x=\sin t/t+\xi$ defined on the interval $t\in[0.1, 20]$. Here ξ is Gaussian

noise with standard 0.01; i.e., the simulations are implemented on a random set of samples corrupted by an additive zero-mean Gaussian noise. We select 200 values of $\boldsymbol{x}$ as the sample data and set the first 100 values as the training sample data points. Any continuous 11 data points are taken as a sample, where the first 10 data points compose an input sample vector and the last one as the output vector. Thus, in the simulation, we have 90 training data points. Then we predict Nos. 101 to 200 data points using the trained model.

Simulation 1 According to Chen *et al.* (2011), the classification or prediction accuracy of LS-SVR is almost the same when $\upsilon \leq 0.001$. We thus compare the proposed method with traditional LS-SVR and Yaakov's scheme (Yaakov *et al.*, 2002) with υ=0.01 and υ=0.001, respectively. Here the Gaussian RBF kernel $k(\boldsymbol{x}, \boldsymbol{y})=\exp[-\|\boldsymbol{x}-\boldsymbol{y}\|^2/(2\sigma^2)]$ with standard deviation σ^2=3 is chosen as the kernel function. The mean values of training time (TrTime), prediction time (PrTime), prediction RMSE and the number of support vectors (NSV) are measured. The prediction results are shown in Figs. 1–3 and Table 1.

Figs. 1–3 and Table 1 show that using the proposed sparseness method although the sparseness process causes the training time to increase, we can still obtain the following conclusions: (1) The proposed sparseness method has good prediction accuracy, specifically, better than Yaakov's scheme with υ=0.01 and almost the same as the traditional LS-SVR and Yaakov's scheme with υ=0.001; (2) The proposed sparseness method costs the least prediction time under similar prediction accuracy. This may be due to the fact that the linear correlation based sparseness strategy ensures the information in the data to be maximally retained and the sparseness prediction model improves the generalization capability to improve the model's adaptability to noisy sample data. The other important merit is that the proposed sparseness method can avoid parameter selection. Thus, it may be more valuable in practice.

Simulation 2 We compare the following two methods: one is the modified simple prediction model and the other is the traditional LS-SVR. The selection of the kernel function and parameters is the same as in Simulation 1. The prediction results are reported in Table 2.

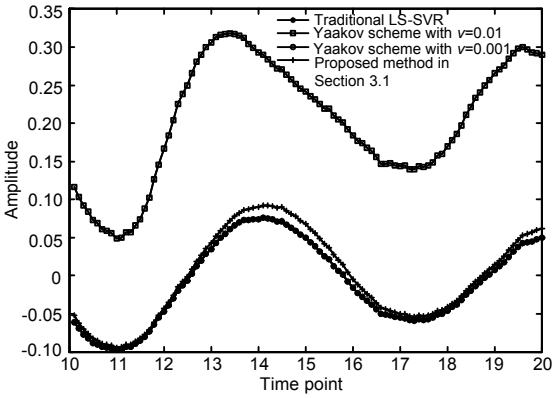

Fig. 1 Prediction results in Simulation 1

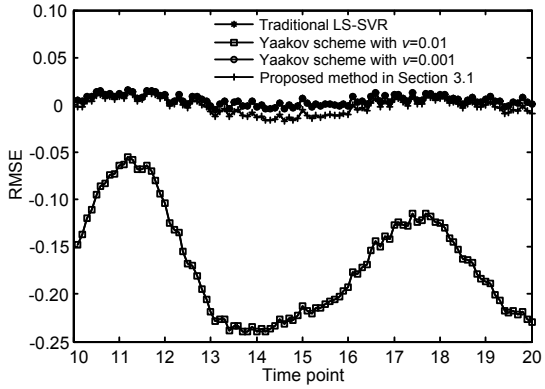

Fig. 2 Prediction RMSE in Simulation 1

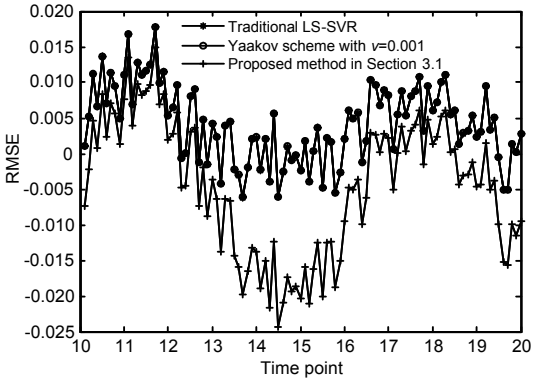

Fig. 3 Prediction RMSE of traditional LS-SVR, Yaakov's scheme with υ=0.001, and the proposed sparseness method in Simulation 1

Table 1 Prediction results of Simulation 1

Method	TrTime (s)	PrTime (s)	RMSE	NSV
Traditional LS-SVR	1.1856	1.1790	0.0584	90
Yaakov's scheme				
υ=0.01	1.3072	1.0961	1.6642	74
υ=0.001	1.7384	1.1733	0.0583	86
Proposed method in Section 3.1	1.4725	1.1040	0.0595	77

Table 2 Prediction results of Simulation 2

Method	TrTime (s)	PrTime (s)	RMSE
Traditional LS-SVR	1.1856	1.1790	0.0584
Modified simple prediction model in Section 3.2	0.8824	1.1801	0.0584

Table 2 shows that the modified simple prediction model can reduce the computing time with the same prediction accuracy as traditional LS-SVR. The reason may be that the modified LS-SVR model has fewer parameters, which reduces the computing complexity.

4.2 A real application and results analysis

The proposed adaptive online prediction method is applied on a certain electronic system to demonstrate its efficiency. We compare two methods on prediction accuracy and computing time. One method is the traditional adaptive online prediction based on traditional LS-SVR (Method 1) (Zhang and Wang, 2006). The other method is the modified simple prediction model with the linear correlation-based sparseness method (Method 2). We collect 5000 data points from the raw data as training and testing samples (Fig. 4). The first 2500 data points are taken as the 2490 initial training samples; i.e., we take any 11 continuous data points as a sample, where the first 10 data points compose an input sample vector and the last one as the output vector. Then we predict Nos. 2501 to 5000 data points. To avoid overfitting, we run the prediction 100 times and take the average of the results.

Here we choose the Gaussian RBF kernel with σ^2=5 as the kernel function. Other parameters are c=100 and γ=200.The results are shown in Figs. 5–7.

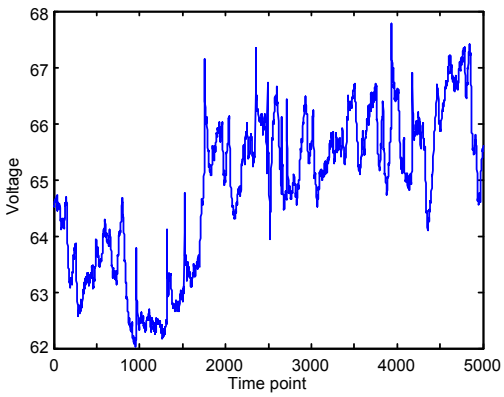

Fig. 4 Raw data of the application

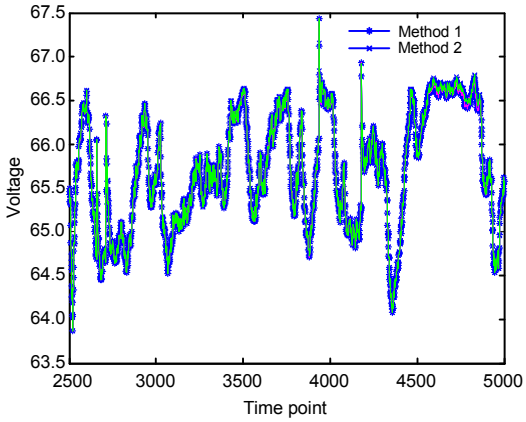

Fig. 5 Prediction results of the application

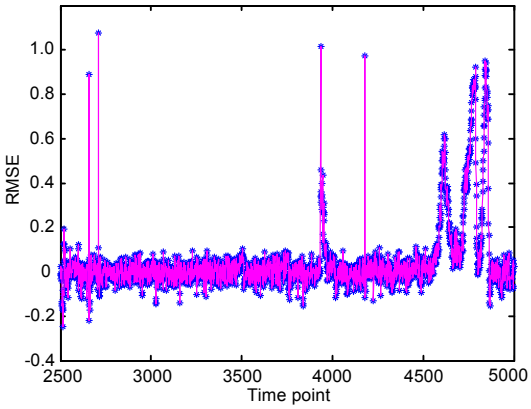

Fig. 6 Prediction RMSE of the application using Method 1

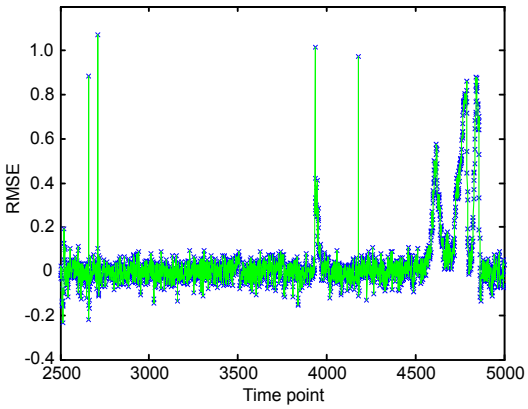

Fig. 7 Prediction RMSE of the application using Method 2

To show the prediction results more clearly, we report the results in Table 3.

Table 3 Prediction results of the application experiment

Method	TrTime (s)	PrTime (s)	RMSE
1	158.2672	64.4453	3.5738
2	140.0364	51.8336	3.3639

Table 3 and Figs. 3–5 show that the proposed adaptive online prediction method has better prediction accuracy and less computing time. This may be due to the fact that we prune the support vectors by judging the linear correlation among the support vectors, and we modify the LS-SVR model to reduce the number of parameters in the model. The pruning method and modified LS-SVR model can reduce the prediction time, improve the prediction accuracy, and cut down the training time. The proposed method meets the requirements of fast and accurate prediction.

5 Conclusions

We deal with the issue of adaptive online prediction based on LS-SVR. We use two approaches to achieve good prediction accuracy and less computation time. First, we remove the support vectors by judging the linear correlation. This approach can control the loss of useful information in the sample data, improve the generalization capability of the prediction model, and reduce the prediction time. Second, we reduce the number of LS-SVR model parameters and establish a modified simple prediction model. This means that less calculation time is consumed in the process of adaptive online training.

We conduct two simulations to assess the proposed methods and apply them to a real electronic system, to demonstrate the effectiveness of the proposed adaptive online prediction method via comparison with the traditional scheme using LS-SVR. The results show that the sparseness method and the modified simple prediction model have better performance, and the adaptive online prediction method combining the two proposed approaches is suitable for practical applications.

Acknowledgements

The authors would like to thank Prof. Ming J. ZUO, director of the Reliability Research Laboratory of University of Alberta, for his comments and suggestions.

References

Cawley, G.C., Talbot, N.L.C., 2002. Reduced rank kernel ridge regression. *Neur. Process. Lett.*, **16**(3):293-302. [doi:10.1023/A:1021798002258]

Chen, A.J., Song, Z.H., Li, P., 2007. Modeling method of least squares support vector regression based on vector base learning. *Control Theory Appl.*, **24**(1):1-5 (in Chinese).

Chen, S.L., Chen, G., Xue, H., 2011. Research on sparsification approach for least squares support vector machine classification. *Comput. Eng.*, **37**(22):145-147, 150 (in Chinese).

Cowen, C.C., 1996. Linear Algebra for Engineering and Science. Indiana West Pickle Press, West Lafayette, USA, p.124-265.

de Kruif, B.J., de Vries, T.J.A., 2003. Pruning error minimization in least squares support vector machines. *IEEE Trans. Neur. Networks*, **14**(3):696-702. [doi:10.1109/TNN.2003.810597]

Deng, J.L., 2004. The Primary Methods of Grey System Theory. Huazhong University of Science and Technology Press, Wuhan, China, p.34-45 (in Chinese).

Guo, H.B., Guan, X.Q., 2010. Application of Least Squares Support Vector Regression in Network Flow Forecasting. 2nd Int. Conf. on Computer Engineering and Technology, p.V7-342-V7-345. [doi:10.1109/ICCET.2010.5485452]

Guo, Y.M., Zhai, Z.J., Jiang, H.M., 2009. Weighted prediction of multi-parameter chaotic time series using least squares support vector regression (LS-SVR). *J. Northwestern Polytechn. Univ.*, **27**(1):83-86 (in Chinese).

Hoegaerts, L., Suykens, J.A.K., Vandewalle, J., de Moor, B., 2004. A comparison of pruning algorithms for sparse least squares support vector machines. *LNCS*, **3316**:1247-1253. [doi:10.1007/978-3-540-30499-9_194]

Jiao, L.C., Bo, L.F., Wang, L., 2007. Fast sparse approximation for sparse least squares support vector machine. *IEEE Trans. Neur. Networks*, **18**(3):685-697. [doi:10.1109/TNN.2006.889500]

Keerthi, S.S., Shevade, S.K., 2003. SMO algorithm for least squares SVM formulations. *Neur. Comput.*, **15**(2):487-507. [doi:10.1162/089976603762553013]

Müller, K.R., Smola, A.J., Rätsch, G., Schölkopf, B., Kohlmorgen, J., Vapnik, V.N., 1997. Predicting time series with support vector machines. *LNCS*, **1327**:999-1004. [doi:10.1007/BFb0020283]

Pang, H.X., Dong, W.D., Xu, Z.H., Feng, H.J., Li, Q., Chen, Y.T., 2011. Novel linear search for support vector machine parameter selection. *J. Zhejiang Univ.-Sci. C (Comput. & Electron.)*, **12**(11):885-896. [doi:10.1631/jzus.C1100006]

Qu, J., Zuo, M.J., 2012. An LSSVR-based algorithm for online system condition prognostics. *Expert Syst. Appl.*, **39**(5):6089-6102. [doi:10.1016/j.eswa.2011.12.002]

Schölkoph, B., Burges, J., Smola, A., 1999. Advance in Kernel Methods: Support Vector Machines. MIT Press, Cambridge, Massachusetts, USA, p.55-90.

Suykens, J.A.K., Vandewale, J., 1999. Least squares support vector machine classifiers. *Neur. Process. Lett.*, **9**(3):293-300. [doi:10.1023/A:1018628609742]

Suykens, J.A.K., Lukas, L., Vandewalle, J., 2000. Sparse Approximation Using Least Squares Support Vector Machines. Proc. IEEE Int. Symp. on Circuits and Systems, p.757-760. [doi:10.1109/ISCAS.2000.856439]

Suykens, J.A.K., Gestel, T.V., Brabanter, J.D., Moor, B.D., Vandewaller, J., 2002. Least Squares Support Vector Machines. World Scientific Publishing, New Jersey, USA, p.71-111. [doi:10.1142/9789812776655_0003]

Vapnik, V.N., 1995. The Nature of Statistical Learning Theory. Springer-Verlag, New York, USA, p.267-287.

Vapnik, V.N., 1998. Statistical Learning Theory. John Wiley and Sons, Hoboken, USA, p.93-110.

Yaakov, E., Shie, M., Ron, M., 2002. Sparse online greedy support vector regression. *LNCS*, **2430**:1-3. [doi:10.1007/3-540-36755-1_8]

Zeng, X.Y., Chen, X.W., 2005. SMO-based pruning methods for sparse least squares support vector machines. *IEEE Trans. Neur. Networks*, **16**(6):1541-1546. [doi:10.1109/TNN.2005.852239]

Zhang, H.R., Wang, X.D., 2006. Incremental and online learning algorithm for regression least squares support vector machine. *Chin. J. Comput.*, **29**(3):400-406 (in Chinese).

Zhao, Y.H., Zhong, P., Wang, K.N., 2011. Application of least squares support vector regression based on time series in prediction of gas. *J. Conv. Inf. Technol.*, **6**(1):243-250. [doi:10.4156/jcit.vol6.issue1.28]

Zhou, X.R., Teng, Z.S., Yi, Z., 2009. Fast pruning algorithm for designing sparse least squares support vector machine. *Electr. Mach. Control*, **13**(4):626-630 (in Chinese).