Applied Soft Computing 9 (2009) 94–99

Contents lists available at ScienceDirect

Applied Soft Computing

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

Non-Mercer hybrid kernel for linear programming support vector regression in nonlinear systems identification

Zhao Lu [a], Jing Sun [b,*]

[a] Department of Electrical Engineering, Tuskegee University, Tuskegee, AL 36088, USA
[b] Department of Naval Architecture and Marine Engineering, University of Michigan, Ann Arbor, MI 48109, USA

ARTICLE INFO

Article history:
Received 5 April 2007
Received in revised form 17 February 2008
Accepted 11 March 2008
Available online 20 March 2008

Keywords:
Support vector regression
Linear programming
Hybrid kernel functions
Nonlinear systems identification

ABSTRACT

As a new sparse kernel modeling method, support vector regression (SVR) has been regarded as the state-of-the-art technique for regression and approximation. In [V.N. Vapnik, The Nature of Statistical Learning Theory, second ed., Springer-Verlag, 2000], Vapnik developed the ε-insensitive loss function for the support vector regression as a trade-off between the robust loss function of Huber and one that enables sparsity within the support vectors. The use of support vector kernel expansion provides us a potential avenue to represent nonlinear dynamical systems and underpin advanced analysis. However, in the standard quadratic programming support vector regression (QP-SVR), its implementation is often computationally expensive and sufficient model sparsity cannot be guaranteed. In an attempt to mitigate these drawbacks, this article focuses on the application of the soft-constrained linear programming support vector regression (LP-SVR) with hybrid kernel in nonlinear black-box systems identification. An innovative non-Mercer hybrid kernel is explored by leveraging the flexibility of LP-SVR in choosing the kernel functions. The simulation results demonstrate the ability to use more general kernel function and the inherent performance advantage of LP-SVR to QP-SVR in terms of model sparsity and computational efficiency.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

Models of dynamical systems are of great importance in almost all fields of science and engineering and specifically in control, signal processing and information science. Since a model is only an approximation of a real phenomenon, having an approximation theory that can facilitate the analysis on modeling performance is of substantial interest. A fundamental principle in system modeling is the Occam's razor, which argues that the model should be no more complex than is required to capture the underlying systems dynamics. This concept, known as the parsimonious principle, which emphasizes the use of the simplest possible model to explain the data, is particularly relevant in nonlinear model building because the size of a nonlinear model can easily become explosively large. For data-driven modeling approach, one parameter that characterizes the parsimonious property of a given model is the model sparsity.

During the past decade, as a new sparse kernel modeling technique, support vector machine (SVM) has been gaining popularity in the field of machine learning and has been regarded as the state-of-the-art technique for regression and classification applications [1–3]. Essentially, SVM is a universal approach for solving the problems of multidimensional function estimation. It is based on the Vapnik–Chervonenkis (VC) theory. Initially, it was designed to solve pattern recognition problem, where in order to find a decision rule with good generalization capability, a small subset of the training data, called the support vectors (SVs), are selected. Experiments showed that it is easy to recognize high-dimensional identities using a small basis constructed from the selected support vectors. Since the inception of this subject, the idea of support vector learning has also been applied to various fields successfully such as regression, density estimation and linear operator equation. When SVM is employed to tackle the problems of function approximation and regression estimation, it is often referred to as the support vector regression (SVR) [1,4]. Research on this topic has shown that the SVR type of function approximation is very effective [2–4], especially for the cases involving high-dimensional input space. Another important advantage for using SVR in function approximation is that the number of free parameters in the function approximation scheme is equal to the number of support vectors. Such a number can be obtained by defining the width of a tolerance band through the ε-insensitive loss function. Thus, the selection of the number of free parameters can be directly related to the approximation accuracy and does not have to depend on the dimensionality of the input space or other factors as that in the case of multilayer feedforward neural networks.

* Corresponding author.
E-mail address: jingsun@umich.edu (J. Sun).

1568-4946/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2008.03.007

The ε-insensitive loss function is attractive because, unlike the quadratic and Huber cost functions where all the data points will be support vectors, the SV solution derived can be sparse. In the realm of data modeling, the sparsity plays a crucial role in improving the generalization performance and computational efficiency. It has been shown that sparse data representations reduce the generalization error as long as the representation is not too sparse, which is consistent with the principle of parsimony [5,6].

For the purpose of modeling complex nonlinear dynamical systems using sparse representation, SVR has been exploited in the context of nonlinear black-box system identification very recently [7–10]. Although it is believed that the formulation of SVM embodies the structural risk minimization principle to combine the excellent generalization properties with a sparse model representation, some data modeling practicians have begun to realize that the capability of the standard quadratic programming SVR (QP-SVR) method to produce sparse models has perhaps been overstated. For example, it has been shown that the standard SVM technique does not always lead to parsimonious models in system identification [9]. A recent study has compared the standard SVM and uniformly regularized orthogonal least squares (UROLS) algorithms using time series prediction problems, and has found that both methods have similar excellent generalization performance but the resulting model from SVM is not sparse enough [11]. It is explained that the number of support vectors found by a quadratic programming algorithm in a SVM is only an upper bound on the number of necessary and sufficient support vectors, due to the linear dependencies between support vectors in the feature space.

On the other hand, due to the distinct mechanism used for selecting the support vectors, the linear programming support vector regression (LP-SVR) is advantageous over QP-SVR in model sparsity, ability to use more general kernel functions and fast learning based on linear programming [12,13]. The idea of linear programming support vector machines is to use the kernel expansion as an ansatz for the solution, but to use a different regularizer, namely the ℓ_1 norm of the coefficient vector. In other words, for LP-SVR, the nonlinear regression problem is treated as a linear one in the kernel space, rather than in the feature space as in the case of QP-SVR. Obviously, the choice of kernel plays a critical role in the performance of LP-SVR. In this paper, the potential of LP-SVR with hybrid kernel will be investigated for constructing sparse support vector model for the nonlinear dynamical systems identification.

Hybrid kernel is a way to combine the heterogeneous complementary characteristics of different kernels, and it was originally proposed in Refs. [14,15] for QP-SVM. Very recently, some successful applications of QP-SVR with hybrid kernels were reported [16,17]. However, the hybrid kernels used in most of literatures appear in the form of a convex combination of two different kernels. In this article, by means of the flexibility of LP-SVR in choosing the kernels, an innovative non-Mercer hybrid kernel is explored for LP-SVR, which is completely different to the hybrid kernels reported in literatures. Simulation results demonstrate substantial performance advantages using the newly proposed hybrid kernel, in terms of model sparsity and prediction accuracy.

The rest of this paper is organized as follows. In the next section, the soft-constrained LP-SVR, including details of the algorithm and its implementation, is introduced. The potentials of non-Mercer hybrid kernel for LP-SVR are investigated in Section 3. Section 4 compares and discusses the performance of LP-SVR and QP-SVR with different kernels for nonlinear dynamical systems identification through simulations. Conclusion and future works are outlined in Section 5.

The following generic notations will be used throughout this paper: lower case symbols such as x, y, $\boldsymbol{\alpha}$, ... refer to scalar valued objects, lower case boldface symbols such as $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\beta}$, ... refer to vector valued objects, and finally capital boldface symbols will be used for matrices.

2. Soft-constrained linear programming SVR

Conceptually there are some similarities between the LP-SVR and QP-SVR. Both algorithms adopt the ε-insensitive loss function, and use kernel functions in feature space.

Consider regression in the following set of functions

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

with given training data, $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_\ell, y_\ell)\}$ where ℓ denotes the total number of exemplars, $\mathbf{x}_i \in R^n$ are the inputs and $y_i \in R$ are the target output data. The nonlinear mapping φ: $R^n \to R^m$ $(m > n)$ maps the input data into a so-called high-dimensional feature space (which can be infinite dimensional) and $\boldsymbol{w} \in R^m$, $b \in R$. In $\varepsilon -$ SV regression, the goal is to find a function $f(\mathbf{x})$ that has at most ε deviation from the actually obtained targets y_i for all the training data, and at the same time, is as flat as possible. In the conventional support vector method one aims at minimizing the empirical risk subject to elements of the structure

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{\ell}(\xi_i + \xi_i^*) \\ &\text{subject to} \begin{cases} y_i - \langle \boldsymbol{w}, \varphi(\boldsymbol{x}_i)\rangle - b \le \varepsilon + \xi_i \\ \langle \boldsymbol{w}, \varphi(\boldsymbol{x}_i)\rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases}, \end{aligned} \tag{2}$$

where $\boldsymbol{\xi}_i$ and ξ_i^* are the slack variables corresponding to the size of the excess deviation for positive and negative direction, respectively. This is a classic quadratic optimization problem with inequality constraints, and the optimization criterion penalizes data points whose y-values differ from $f(\mathbf{x})$ by more than ε. The constant $C > 0$ determines the trade-off between the flatness of f and the amount up to which deviations larger than ε can be tolerated. By defining the loss function,

$$L(y_i - f(\boldsymbol{x}_i)) = \begin{cases} 0, & \text{if } |y_i - f(\boldsymbol{x}_i)| \le \varepsilon \\ |y_i - f(\boldsymbol{x}_i)| - \varepsilon, & \text{otherwise} \end{cases}, \tag{3}$$

the optimization problem (2) is equivalent to the following regularization problem,

$$\text{minimize} \quad R_{\mathrm{reg}}[f] = \sum_{i=1}^{\ell} L(y_i - f(\boldsymbol{x}_i)) + \lambda||\boldsymbol{w}||^2, \tag{4}$$

where $f(\mathbf{x})$ is in the form of (1) and $\lambda||\boldsymbol{w}||^2$ is the regularization term. According to the well-known Representer Theorem [3], the solution to the regularization problem (4) can be written as the SV kernel expansion provided $k(\mathbf{x}_i, \mathbf{x}_i) = 1$

$$f(\boldsymbol{x}) = \sum_{i=1}^{\ell} \beta_i k(\boldsymbol{x}_i, \boldsymbol{x}), \tag{5}$$

where $k(\mathbf{x}_i, \mathbf{x})$ is the kernel function. Defining

$$\boldsymbol{\beta} = [\beta_1 \quad \beta_2 \quad \cdots \quad \beta_\ell]^{\mathrm{T}},$$

LP-SVR replaces (4) by

$$\text{minimize} \quad R_{\mathrm{reg}}[f] = \sum_{i=1}^{\ell} L(y_i - f(\boldsymbol{x}_i)) + \lambda||\boldsymbol{\beta}||_1, \tag{6}$$

where $f(\mathbf{x})$ is in the form of (5) and $||\beta||_1$ denotes the ℓ_1 norm in coefficient space. This regularization problem is equivalent to the

following constrained optimization problem

$$\text{minimize} \quad ||\boldsymbol{\beta}||_1 + C\sum_{i=1}^{\ell}(\xi_i + \xi_i^*)$$
$$\text{subject to} \begin{cases} y_i - \sum_{j=1}^{\ell}\beta_j k(\boldsymbol{x}_j, \boldsymbol{x}_i) \le \varepsilon + \xi_i \\ \sum_{j=1}^{\ell}\beta_j k(\boldsymbol{x}_j, \boldsymbol{x}_i) - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases} . \tag{7}$$

From the geometric perspective, it can be followed that $\xi_i\xi_i^* = 0$ in SV regression. Therefore, it is sufficient to just introduce slack variable $\boldsymbol{\xi}_i \ge 0$ in the constrained optimization problem (7). Thus, we arrive at the following formulation of SV regression with fewer slack variables

$$\text{minimize} \quad ||\boldsymbol{\beta}||_1 + 2C\sum_{i=1}^{\ell}\xi_i$$
$$\text{subject to} \begin{cases} y_i - \sum_{j=1}^{\ell}\beta_j k(\boldsymbol{x}_j, \boldsymbol{x}_i) \le \varepsilon + \xi_i \\ \sum_{j=1}^{\ell}\beta_j k(\boldsymbol{x}_j, \boldsymbol{x}_i) - y_i \le \varepsilon + \xi_i \\ \xi_i \ge 0 \end{cases} . \tag{8}$$

In an attempt to convert the optimization problem above into a linear programming problem, we decompose β_i and $|\beta_i|$ as follows

$$\beta_i = \alpha_i^+ - \alpha_i^- \qquad |\beta_i| = \alpha_i^+ + \alpha_i^-, \tag{9}$$

where $\alpha_i^+, \alpha_i^- \ge 0$. It is worth noting that the decompositions in (9) are unique, i.e., for a given β_i there is only one pair (α_i^+, α_i^-) which fulfils both equations. Furthermore, both variables cannot be larger than zero at the same time, i.e., $\alpha_i^+ \cdot \alpha_i^- = 0$. In this way, the ℓ_1 norm of β can be written as

$$||\boldsymbol{\beta}||_1 = \left(\underbrace{1, 1, \ldots, 1}_{\ell}, \underbrace{1, 1, \ldots, 1}_{\ell}\right)\begin{pmatrix}\boldsymbol{\alpha}^+ \\ \boldsymbol{\alpha}^-\end{pmatrix}, \tag{10}$$

where $\boldsymbol{\alpha}^+ = (\alpha_1^+, \alpha_2^+, \ldots, \alpha_\ell^+)^{\mathrm{T}}$ and $\boldsymbol{\alpha}^- = (\alpha_1^-, \alpha_2^-, \ldots, \alpha_\ell^-)^{\mathrm{T}}$. Furthermore, the constraints in the formulation (8) can be written in the following vector form

$$\begin{pmatrix}\boldsymbol{K} & -\boldsymbol{K} & -\boldsymbol{I} \\ -\boldsymbol{K} & \boldsymbol{K} & -\boldsymbol{I}\end{pmatrix} \cdot \begin{pmatrix}\boldsymbol{\alpha}^+ \\ \boldsymbol{\alpha}^- \\ \boldsymbol{\xi}\end{pmatrix} \le \begin{pmatrix}\boldsymbol{y} + \varepsilon \\ \varepsilon - \boldsymbol{y}\end{pmatrix}, \tag{11}$$

where $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_\ell)^{\mathrm{T}}$ and $\mathbf{I}$ is $\ell \times \ell$ identity matrix. Thus, the constrained optimization problem (8) can be implemented by the following linear programming problem with the variables $\boldsymbol{\alpha}^+$, $\boldsymbol{\alpha}^-$, and $\boldsymbol{\xi}$

$$\text{minimize} \quad \boldsymbol{c}^{\mathrm{T}}\begin{pmatrix}\boldsymbol{\alpha}^+ \\ \boldsymbol{\alpha}^- \\ \boldsymbol{\xi}\end{pmatrix}$$
$$\text{subject to} \begin{cases} \begin{pmatrix}\boldsymbol{K} & -\boldsymbol{K} & -\boldsymbol{I} \\ -\boldsymbol{K} & \boldsymbol{K} & -\boldsymbol{I}\end{pmatrix} \cdot \begin{pmatrix}\boldsymbol{\alpha}^+ \\ \boldsymbol{\alpha}^- \\ \boldsymbol{\xi}\end{pmatrix} \le \begin{pmatrix}\boldsymbol{y} + \varepsilon \\ \varepsilon - \boldsymbol{y}\end{pmatrix} \\ \boldsymbol{\alpha}^+, \boldsymbol{\alpha}^- \ge 0, \quad \boldsymbol{\xi} \ge 0 \end{cases} , \tag{12}$$

where

$$\boldsymbol{c} = \left(\underbrace{1, 1, \ldots, 1}_{\ell}, \underbrace{1, 1, \ldots, 1}_{\ell}, \underbrace{2C, 2C, \ldots, 2C}_{\ell}\right)^{\mathrm{T}} .$$

In the QP-SVR case, the set of points not inside the tube coincides with the set of SVs. While, in the LP context, this is no longer true—although the solution is still sparse, any point could be an SV, even if it is inside the tube [18]. Actually, the sparse solution still can be obtained in LP-SVR even though the size of the insensitive tube was set to zero [19], due to the soft constraints used. However, sparser solution can usually be obtained by setting non-zero ε.

3. Hybrid kernel functions for LP-SVR

Obviously, the kernel functions $k(\mathbf{x}_i, \mathbf{x})$ play a crucial role in determining the characteristics of the model (5), and choosing different kernel functions may result in different performance. In support vector learning algorithms, the kernel function provides an elegant way of working in the feature space, thereby avoiding all the troubles and difficulties inherent in high dimensions. This method is applicable whenever an algorithm can be cast in terms of dot-product. In the conventional QP-SVR, a kernel corresponding to a dot-product in an associated space has to be a positive definite function, i.e., satisfying Mercer's condition. Several commonly used kernel functions in literatures are:

- Gaussian radial basis function (GRBF) kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(\frac{-||\boldsymbol{x} - \boldsymbol{x}'||^2}{2\sigma^2}\right). \tag{13}$$

- Polynomial kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}') = (1 + \langle \boldsymbol{x}, \boldsymbol{x}' \rangle)^q. \tag{14}$$

- Sigmoid kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \tanh(\alpha\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + \gamma). \tag{15}$$

- Thin plate spline kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}') = ||\boldsymbol{x} - \boldsymbol{x}'||^2 \ln||\boldsymbol{x} - \boldsymbol{x}'||. \tag{16}$$

where $\sigma, q, \boldsymbol{\alpha}, \gamma$ are adjustable parameters of the kernel functions. The GRBF kernel (13) and thin plate spline kernel (16) are in the class of translation-invariant kernels, and the polynomial kernel (14) and sigmoid kernel (15) are examples of rotation invariant kernels. If one uses non-Mercer kernel (a kernel which does not satisfy Mercer's condition), in general, there may exist data such that the corresponding Hessian matrix is indefinite, for which the quadratic programming problem has no solution since the dual objective function might become arbitrarily large in this ill-posed case. However, in the linear programming formulation of support vector learning, even for non-Mercer kernels, a solvable linear programming problem can still be obtained, thereby providing more flexibility in designing the kernel functions for LP-SVR.

Rather than design a kernel from scratch, one might be tempted to generate a kernel from a family of available kernels. In fact, every kernel function has its specific feature in measuring the similarity of the different patterns, and many functions of above-mentioned kernels are also kernels, such as the nonnegative linear combination of Mercer's kernels. It is also known that the set of Mercer's kernels is closed under the topology of pointwise convergence and pointwise multiplication [15]. By using the useful closure properties of kernel functions, it can be followed that the exponential function of Gaussian radial basis function kernel, $\exp(k_{\mathrm{GRBF}})$ is also a Mercer's kernel because

$$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \cdots. \tag{17}$$

Two typical kernel functions often used in SVMs are the local and global kernels [16]. For local kernel, only the data that are close to each other have an influence on the kernel values. An example of

a typical local kernel is the GRBF kernel (13), where the kernel parameter is the width σ of the radial basis function. The local kernel has good interpolation capabilities, but fails to provide longer range extrapolation. In contrast, a global kernel allows data points that are far away from each other to have an influence on the kernel values as well. The polynomial kernel (14), where the kernel parameter q is the degree of the polynomial to be used, is a typical example of a global kernel. The global kernel shows better extrapolation abilities at lower orders, but requires higher orders for good interpolation. Sometimes the 'good' characteristics of two or more kernel functions should be combined and the mixtures of them may generate better performance than any single kernel. Therefore, the advantages of local kernel and global kernel can be incorporated to form a class of new hybrid kernel functions by using their mixtures.

It is apparent that the interpolation ability plays an important role in almost all regression analysis applications, which denotes the performance to fit the given data points and predict within the range of the given data points. On the other hand, in the applications of nonlinear systems identification, the systems to be identified may work at different operation points or regions, which makes the extrapolation capability of the regression model crucial. Hence, it is plausible to employ the hybrid kernels in an attempt to obtain the excellent performance of SVR for nonlinear systems identification. Particularly, LP-SVR provides more flexibility in designing hybrid kernels.

In literatures, one popular way of combining the GRBF kernel k_{GRBF} (local kernel) and polynomial kernel k_{ploy} (global kernel) is to use their convex combination to get a mixed kernel k_{mix}, as follows:

$$k_{\mathrm{mix}} = \rho k_{\mathrm{ploy}} + (1-\rho)k_{\mathrm{GRBF}}, \tag{18}$$

where ρ is the optimal mixing coefficient, which varies between 0.5 and 0.95. In our research, from a radically different perspective, an innovative non-Mercer hybrid kernel function was coined in the form of

$$k_{\mathrm{hybrid}} = \sin(k_{\mathrm{GRBF}}), \tag{19}$$

which is similar to the Mercer's kernel $\exp(k_{\mathrm{GRBF}})$ mentioned before in the sense that $\sin(\cdot)$ can also be written as a power series by Taylor expansion

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots. \tag{20}$$

It is worth noticing that an important difference between the Mercer's kernel $\exp(k_{\mathrm{GRBF}})$ and the kernel function (19) mainly lies in that $\sin(\cdot)$ cannot be expanded as a polynomial with positive coefficients, which means that the kernel (19) does not satisfy the Mercer's positive definiteness condition. However, it does not hinder the use of (19) as the kernel in the linear programming formulation of support vector learning. Note that it is a very different way than (18) to generate a hybrid kernel from standard kernels, the hybrid kernel (19) also provides a new insight of combining the polynomial characteristic into the GRBF kernel by compounding them.

4. Simulation

In this section, the soft-constrained LP-SVR with hybrid kernel is applied on a nonlinear black-box systems identification problem for a hydraulic robot arm. This is a benchmark problem in nonlinear systems identification, and it has been used widely for testing the various identification methods [7,20]. In this nonlinear black-box dynamical system, the input $u(t)$ represents the size of the valve through which oil flows into the actuator, and the output

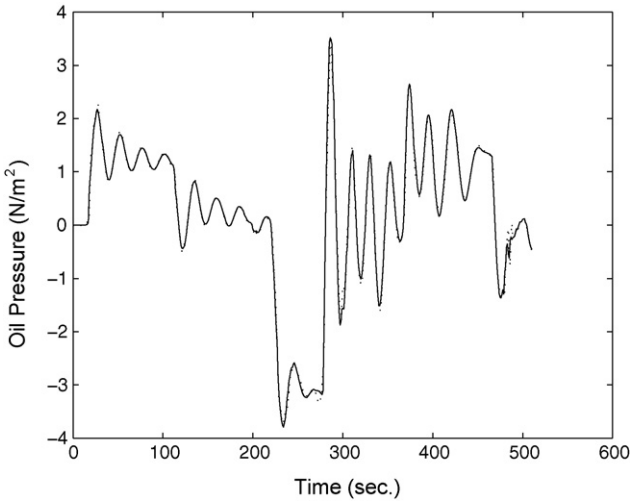

Fig. 1. Identification of hydraulic robot arm dynamics by GRBF kernel LP-SVR on the training set (solid line: observation, dotted line: model from GRBF kernel LP-SVR).

$y(t)$ is a measure of oil pressure which determines the robot arm position. For the purpose of comparison, we use the same regressor

$$\boldsymbol{x}(t) = [y(t-1) \quad y(t-2) \quad y(t-3) \quad u(t-1) \quad u(t-2)]^{\mathrm{T}},$$

as that in [7,20]. Following the procedure in [7,20], we also used half of the data set containing 511 training data pairs for training, and the remaining half as validation data. The prediction results of the LP-SVR model with the hybrid kernel are compared with those acquired from QP-SVR or LP-SVR without using the hybrid kernel in terms of sparsity and prediction accuracy. The generalization capability and accuracy of regression algorithms could be evaluated using the root mean square (RMS) error

$$E_{\mathrm{RMS}} = \sqrt{\frac{1}{N}\sum_{i=0}^{N}[\hat{f}(\boldsymbol{x}_i) - f(\boldsymbol{x}_i)]^2}, \tag{21}$$

where $\hat{f}(\boldsymbol{x}_i)$ is the estimated value at point $\mathbf{x}_i$ from the SVR model.

In our simulation, the soft-constrained LP-SVR and QP-SVR with GRBF kernel are first applied to model the dynamic behavior of the hydraulic robot arm, respectively. For the sake of comparison, the same width of tolerance band and kernel parameters for LP-SVR

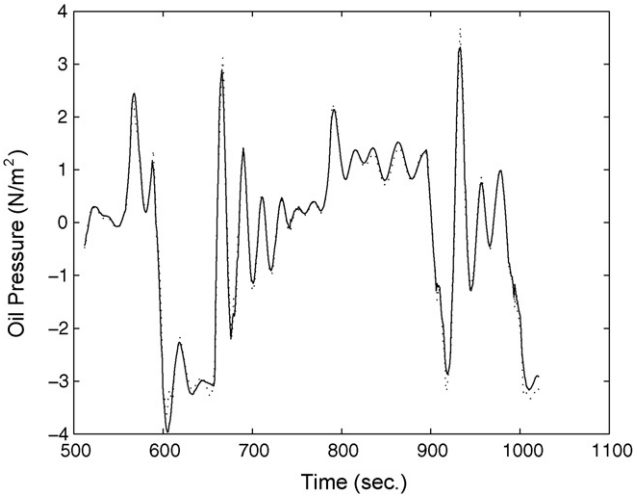

Fig. 2. Identification of hydraulic robot arm dynamics by GRBF kernel LP-SVR on the validation set (solid line: observation, dotted line: model from GRBF kernel LP-SVR).

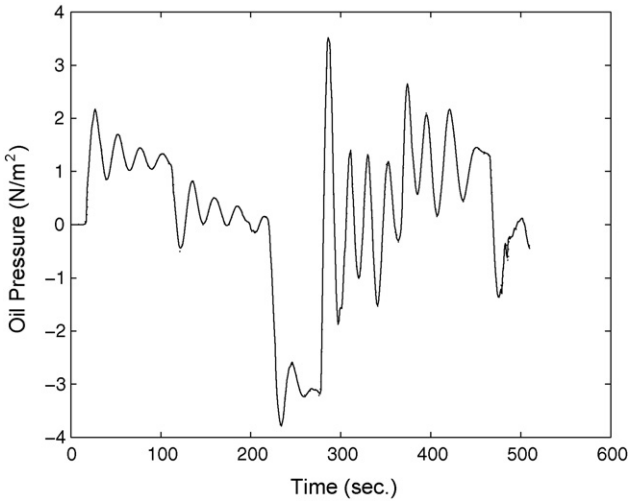

Fig. 3. Identification of hydraulic robot arm dynamics by GRBF kernel QP-SVR on the training set (solid line: observation, dotted line: model from GRBF kernel QP-SVR).

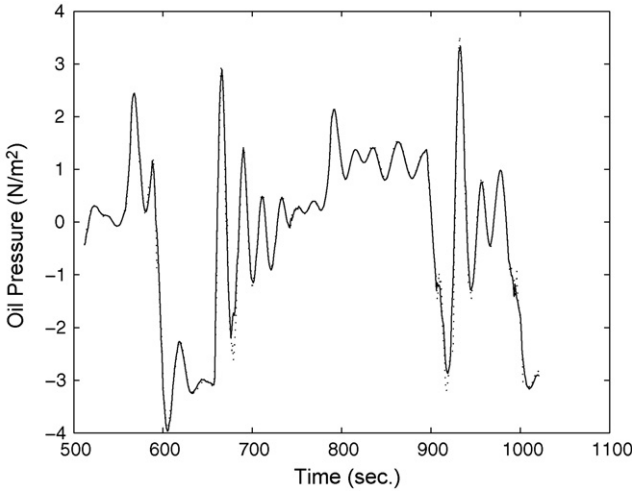

Fig. 4. Identification of hydraulic robot arm dynamics by GRBF kernel QP-SVR on the validation set (solid line: observation, dotted line: model from GRBF kernel QP-SVR).

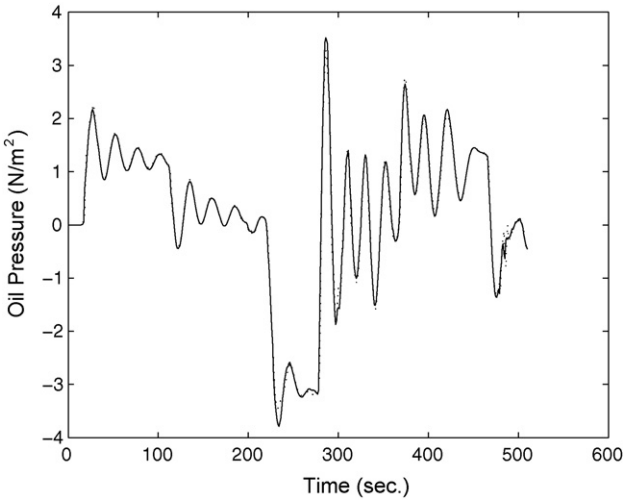

Fig. 5. Identification of hydraulic robot arm dynamics by non-Mercer hybrid kernel LP-SVR on the training set (solid line: observation, dotted line: model from hybrid kernel LP-SVR).

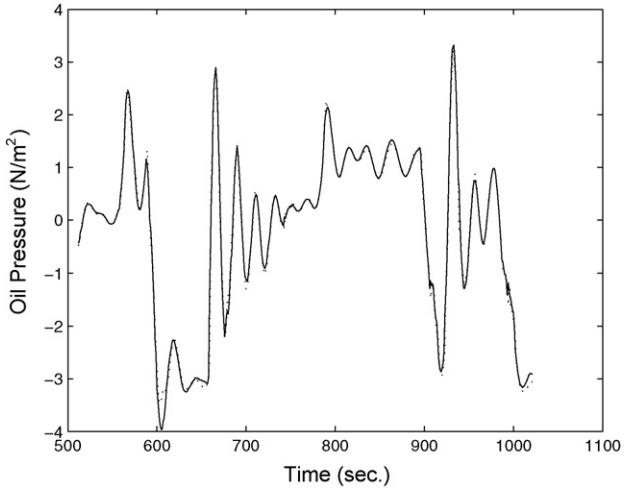

Fig. 6. Identification of hydraulic robot arm dynamics by non-Mercer hybrid kernel LP-SVR on the validation set (solid line: observation, dotted line: model from hybrid kernel LP-SVR).

and QP-SVR are used: $\varepsilon = 0.03$; GRBF kernel width parameter $\sigma = 1$. The identification results by LP-SVR and QP-SVR with GRBF kernel are illustrated in Figs. 1–4, and the ratio of SVs to the training vectors and RMS error on validation data are given in Table 1. Obviously, LP-SVR enables us to obtain a much sparser approximation model and better prediction accuracy with less computation time.

In Refs. [7,20], the RMS error on validation set acquired by using other identification algorithms are reported, such as 0.579 from wavelet networks, 0.467 from one hidden-layer sigmoid neural networks with 10 hidden nodes, 0.280 from ν-support vector regression. Evidently, higher prediction accuracy and better generalization performance could be obtained by using the soft-constrained LP-SVR or QP-SVR.

Table 1
Comparison on model sparsity and RMS error of different SVRs

SVR algorithm	SV ratio (%)	RMS error on validation set
QP-SVR with GRBF kernel	39.7	0.1597
LP-SVR with GRBF kernel	6.7	0.1531
LP-SVR with hybrid kernel (19)	3.3	0.1387

Further, for the purpose of demonstrating the power of hybrid kernel in LP-SVR, the LP-SVR with the hybrid kernel (19) is applied for identifying the hydraulic robot arm system. The kernel width parameter σ is set to 1.6, and the width of tolerance band ε is the same as before. The ratio of SVs to the training vectors and the RMS error on validation data obtained using LP-SVR with hybrid kernel are also given in Table 1. It can be found that even compared with the model from LP-SVR with RBF kernel, the model sparsity is improved dramatically, while the prediction accuracy is enhanced simultaneously. Compared to the model from QP-SVR, the model sparsity is improved more than 10 times, and with the higher prediction accuracy. Figs. 5 and 6 visualize the identification results by LP-SVR with the hybrid kernel.

5. Conclusion and future works

In the conventional QP-SVR, when a non-Mercer kernel is used, the associated non-convex quadratic program may not be solvable

[21]. For LP-SVR, however, the assumptions of symmetry or positive definiteness on kernel functions are not needed, thereby providing a great deal of flexibility in the choice of the kernel function. In this paper, for the purpose of exploiting the heterogeneous characteristics of different kernels, the possibility and superiority of employing the non-Mercer hybrid kernels in LP-SVR is explored and discussed. In the context of nonlinear systems identification, the emphasis is placed on the sparsity of data representation, and it has been shown that the LP-SVR is very potential in modeling nonlinear dynamical systems and outperforms QP-SVR in terms of model sparsity and computational efficiency. It is also believed that the sparsity can be further improved by the reduced-set method [22].

Future research will be conducted in two different directions. The first one concentrates on the analysis of the mechanism of support vectors selection in LP-SVR, such that the on-line version of LP-SVR algorithm can be developed. Inspired by the flexibility of choosing kernel functions in LP-SVR, the other one will investigate the roles of indefinite operator used as kernel functions in LP-SVR.

Acknowledgement

Thanks to Dr. Arthur Gretton for his providing the hydraulic robot arm dataset.

References

[1] V.N. Vapnik, The Nature of Statistical Learning Theory, second ed., Springer-Verlag, 2000.
[2] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, 2000.
[3] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2002.
[4] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and Computing 14 (2004) 199–222.
[5] N. Ancona, R. Maglietta, E. Stella, Data representation in kernel based learning machines, in: International Conference on Artificial Intelligence and Soft Computing, 2004.
[6] S. Chen, Local regularization assisted orthogonal least squares regression, Neurocomputing 69 (2006) 559–585.
[7] A. Gretton, A. Doucet, R. Herbrich, P.J.W. Rayner, B. Schölkopf, Support vector regression for black-box system identification, in: Proceedings of the 11th IEEE Workshop on Statistical Signal Processing, 2001.
[8] J.L. Rojo-Alvarez, M. Martinez-Ramon, M. Prado-Cumplido, A. Artes-Rodriguez, A.R. Figueiras-Vidal, Support vector machines for nonlinear kernel ARMA system identification, IEEE Transactions on Neural Networks 17 (2006) 1617–1622.
[9] P.M.L. Drezet, R.F. Harrison, Support vector machines for system identification, in: UKACC International Conference on Control, 1998.
[10] W.C. Chan, C.W. Chan, K.C. Cheung, C.J. Harris, On the modeling of nonlinear dynamic systems using support vector neural networks, Engineering Applications of Artificial Intelligence 14 (2001) 105–113.
[11] K.L. Lee, S.A. Billings, Time series prediction using support vector machines, the orthogonal and the regularized orthogonal least-squares algorithms, International Journal of Systems Science 33 (2002) 811–821.
[12] V. Kecman, Learning and Soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models, MIT Press, 2001.
[13] I. Hadzic, V. Kecman, Support vector machines trained by linear programming: theory and application in image compression and data classification, in: IEEE 5th Seminar on Neural Network Applications in Electrical Engineering, 2000.
[14] Y. Tan, J. Wang, A support vector machine with a hybrid kernel and minimal Vapnik-Chervonenkis dimension, IEEE Transactions of Knowledge and Data Engineering 16 (2004) 385–395.
[15] J.P. Vert, K. Tsuda, B. Schölkopf, in: J.P. Vert, K. Tsuda, B. Schölkopf (Eds.), Kernel Methods in Computational Biology, MIT Press, Cambridge, MA, 2004, pp. 35–70.
[16] S. Zheng, J. Liu, J.W. Tian, An efficient star acquisition method based on SVM with mixtures, Pattern Recognition Letters 26 (2005) 147–165.
[17] X. Liu, W.C. Lu, S.L. Jin, Y.W. Li, N.Y. Chen, Support vector regression applied to materials optimization of sialon ceramics, Chemometrics and Intelligent Laboratory Systems 82 (2006) 8–14.
[18] A. Smola, B. Schölkopf, G. Ratsch, Linear programs for automatic accuracy control in regression, in: International Conference on Artificial Neural Networks, Berlin, 1999.
[19] P.M.L. Drezet, R.F. Harrison, A new method for sparsity control in support vector classification and regression, Pattern Recognition 34 (2001) 111–125.
[20] J. Sjöberg, Q. Zhang, L. Ljung, A. Berveniste, B. Delyon, P. Glorennec, H. Hjalmarsson, A. Juditsky, Nonlinear black-box modeling in system identification: a unified overview, Automatica 31 (1995) 1691–1724.
[21] O.L. Mangasarian, in: A. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans (Eds.), Advances in Large-margin Classifiers, MIT Press, Cambridge, MA, 2000, pp. 135–146.
[22] T.T. Frieβ, R.F. Harrison, Linear programming support vector machines for pattern classification and regression estimation; and the SR algorithm: improving speed and tightness of VC Bounds in SV algorithms, Res. Rep. 706, Dept. Automatic Control and Syst. Eng., Univ. Sheffield, Sheffield, 1998.