2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Multi-Kernel Linear Programming Support Vector Regression with Prior Knowledge

Jinzhu Zhou, Na Li, Liwei Song
Key Laboratory of Electronic Equipment Structure Design of Ministry of Education, Xidian University
Xi'an, Shaanxi Province, P. R. China
E-mail: xidian_jzzhou@126.com

Abstract—**This paper proposes a multi-kernel linear programming support vector regression with prior knowledge in order to obtain an accurate regression model in the case of the scarcity of measured data available. In the algorithm, multi-kernel and prior knowledge which may be exact or biased from a calibrated simulator have been incorporated into the framework of linear programming support vector regression by utilizing multiple feature spaces and modifying optimization formulation. Some experiments from a synthetic example have been carried out, and the results show that the proposed algorithm is effective, and that the obtained model is sparse and accurate. The proposed algorithm shows great potential in some practical applications where the experimental data is few and the prior knowledge from a simulator is available.**

Keywords—support vector regression; multi-kernel; s prior knowledge; linear programming

I. INTRODUCTION

Support vector regression (SVR), which is based on the theory of structure risk minimization in statistical learning , has been successfully applied to many problem [**1**]. Traditionally, support vector regression can find an estimating function by solving a quadratic program problem, which is also known as QPSVR [2-4]. Subsequently, Smola has proposed the linear programming support vector regression (LPSVR) [**5, 6**]. Both LPSVR and QPSVR adopt ε - insensitive loss function and the kernel function in feature space. However, LPSVR is advantageous over QPSVR in the model sparsity, ability to use more general kernel functions and fast learning ability [2, 7-9] .

Support vector regression aims at learning an unknown function based on some training data samples. However, in some practical applications, it is complex and costly to obtain sufficient experimental data. Utilizing the fewer data, one can find that it is a little difficult to obtain an accurate model. Moreover, there are many complex functions which comprise both the steep variation and the smooth variation in the engineering. It is more difficult to obtain an accurate model from a small data set [10]. In this paper, we will focus on the problem that how to obtain an accurate model from a limited amount of experimental data.

In order to improve the accuracy of regression model, Lanckriet has proposed a multi-kernel support vector regression by using the conic combinations of kernel matrices, and formulated the algorithm as a convex quadratically constrained quadratic program (QCQP) [**11, 12**]. Although the formulation yields global optimal solutions, it is computationally inefficient and requires a commercial solver. Subsequently, the multi-kernel learning algorithm has been reformulated as a semi-infinite linear programming to obtain a general and efficient algorithm [11, 13]. Based on the principle of kernel-target alignment and predictive accuracy, Qiu has proposed three heuristics methods to speed up the computation of QCQP formulation [14]. In [15], a multi-kernel semi-parametric support vector regression is proposed by using quadratic program solver and a semi-parametric algorithm. Instead of a single kernel, multi-scale support vector regression has been presented by using the same kernel with multiple scales [**16, 17**]. All of multi-kernel support vector regression can establish an accurate model, if there is sufficient amount of data samples [**18, 19**]. However, the number of measured data is usually so little in some practical applications that the model developed by the algorithms cannot meet the desired requirement.

In a real application, a certain amount of knowledge on the problem is usually known beforehand [20]. The prior knowledge can take many forms such as a simulation model from a practical engineering, the shape of the function on a particular region and some equality and inequality constrains [**21, 22**]. By utilizing the prior knowledge, one can improve the predictive accuracy of support vector regression. In [21, 23], the author has reviewed three methods of incorporating prior knowledge in support vector machine for classification, which comprise sample methods, kernel methods and the optimization method. In [24], the author explores the incorporation of different types of prior knowledge in support vector regression by the modification of the problem formulation. In addition, prior knowledge over arbitrary region is incorporating into kernel approximation problem, and the region has to be discretized before including the prior knowledge to be in the learning as

This project was supported by National Natural Science Foundation of China (Grant No. 51305323 and 51305322), and the Fundamental Research Funds for the Central Universities (Grant No.K5051204019 and K5051304048)

a finite set of inequalities [**25, 26**]. Though the prior knowledge could make up a small amount of measured data and improve the modeling accuracy, all of the algorithms incorporating prior knowledge above have exploited a single kernel, and do not employ the advantages of multi-kernel functions.

In this paper, in order to obtain an accurate model from a limited amount of measured data, we have presented a novel multi-kernel prior knowledge linear programming support vector regression (MKPLPSVR). In the algorithm, multi-kernel and prior knowledge which may be exact or biased from a calibrated simulator have been incorporated into the framework of linear programming support vector regression by utilizing multiple feature spaces and modifying optimization formulation. Some experiments from a synthetic example have been carried out, and the results show that the proposed algorithm is effective, and that the obtained model is sparse and accurate.

II. Review of Linear Rrogramming Support Vector Regression

Let $\mathrm{M}=\{(\boldsymbol{x}_i,\boldsymbol{y}_i), i=1,...,N\}$ be an experimental dataset, where the input is $\boldsymbol{x}_i\in \mathrm{R}^d$ and the output is $\boldsymbol{y}_i\in\mathrm{R}$. The regression is considered as a linear function in the feature space which induced by a nonlinear mapping $\varphi(\boldsymbol{x})$. The regression function is written as:

$$f(\boldsymbol{x})=\boldsymbol{\omega}\cdot\varphi(\boldsymbol{x})+b \tag{1}$$

where $\boldsymbol{\omega}$ is a normal vector in the feature space, and b is a bias term.

The normal vector $\boldsymbol{\omega}$ can be considered as a linear combination of the training patterns, i.e. $\boldsymbol{\omega}=\sum_{i=1}^{N}\alpha_i\varphi(\boldsymbol{x}_i)$. Therefore, the regression function in the original space is expressed as:

$$f(\boldsymbol{x})=\sum_{i=1}^{N}\alpha_i k(\boldsymbol{x},\boldsymbol{x}_i)+b \tag{2}$$

where $k(\boldsymbol{x},\boldsymbol{x}_i)=\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x})$ is the kernel function which usually includes Gaussian radial basis function, polynomial kernel, and even non-Mercer kernel [6-8].

Instead of choosing the flattest function, LPSVR seek the smallest combination of training patterns. According to the statistical learning theory [**1, 2**], the coefficient α_i and the bias term b can be solved by minimizing the regularized risk function:

$$\text{Min: } Q(\boldsymbol{a})+2C\sum_{i=1}^{N}L(y_i-f(\boldsymbol{x}_i)) \tag{3}$$

where $Q(\boldsymbol{a})$ is a regularization term, and it is defined as $Q(\boldsymbol{a})=\|\boldsymbol{\alpha}\|_1=\sum_{i=1}^{n}|\alpha_i|$. The vector $\boldsymbol{\alpha}=[\alpha_1,\alpha_i,\cdots\alpha_N]^{\mathrm{T}}$ in $Q(\boldsymbol{a})$ determines the function complexity. A hyper-parameter $C>0$ is introduced to tune the trade-off between the error minimization and the function sparsely. $L(y_i-f(\boldsymbol{x}_i))$ denotes the ε -insensitive loss function:

$$L(y_i-f(\boldsymbol{x}_i))=\begin{cases}0, & |y_i-f(\boldsymbol{x}_i)|\le\varepsilon\\ |y_i-f(\boldsymbol{x}_i)|-\varepsilon, & \text{otherwise}\end{cases} \tag{4}$$

By introducing a slack variable ξ_i and using ε -insensitive loss function, LPSVR is formulated as:

$$\begin{aligned}&\text{Min: } \|\boldsymbol{\alpha}\|_1+2C\sum_{i=1}^{N}\xi_i\\ &\text{s.t.}\begin{cases}y_i-\sum_{i=1}^{N}\alpha_i k(\boldsymbol{x}_i,\boldsymbol{x}_j)-b\le\varepsilon+\xi_i\\ \sum_{i=1}^{N}\alpha_i k(\boldsymbol{x}_i,\boldsymbol{x}_j)+b-y_i\le\varepsilon+\xi_i\\ \xi_i\ge 0\\ \forall i=1,2,\cdots,N\end{cases}\end{aligned} \tag{5}$$

In order to solve the optimization above, we can decompose α_i and $|\alpha_i|$ as follows:

$$\alpha_i=\alpha_i^+-\alpha_i^-,\quad |\alpha_i|=\alpha_i^++\alpha_i^- \tag{6}$$

where $\alpha_i^+,\alpha_i^-\ge 0$. Due to the nature of the constraints, typically only a subset of α_i is non-zero, and the associated training data are called support vectors [7, 8].

Substituting (6) into (5), LPSVR can be expressed as:

$$\begin{aligned}&\text{Find: } \alpha_j^+,\alpha_j^-,\xi_i,b\\ &\text{Min: } \sum_{j=1}^{N}(\alpha_j^++\alpha_j^-)+2C\sum_{i=1}^{N}\xi_i\\ &\text{s.t.}\begin{cases}y_i-\sum_{j=1}^{N}(\alpha_j^+-\alpha_j^-)k(\boldsymbol{x}_i,\boldsymbol{x}_j)-b\le\varepsilon+\xi_i\\ \sum_{j=1}^{N}(\alpha_j^+-\alpha_j^-)k(\boldsymbol{x}_i,\boldsymbol{x}_j)+b-y_i\le\varepsilon+\xi_i\\ \alpha_j^+\ge 0,\ \alpha_j^-\ge 0\\ \xi_i\ge 0.(\forall i=1,2,\cdots,N)\end{cases}\end{aligned} \tag{7}$$

The coefficients $\alpha_j^+,\alpha_j^-,\xi_i$ and b in (7) are solved by using *linprog* in *Matlab*. Substituting (6) into (2), one is express as:

$$f(\boldsymbol{x})=\sum_{j=1}^{N}(\alpha_j^+-\alpha_j^-)k(\boldsymbol{x},\boldsymbol{x}_j)+b \tag{8}$$

III. Proposed Algorithm

In order to improve the modeling accuracy from few experimental data, this subsection has proposed an algorithm which can incorporate multi-kernel and prior knowledge into LPSVR. Fig.1 shows the basic idea of developing the algorithm.

From Fig. 1, multiple feature spaces have been utilized to develop multi-kernel linear programming support vector regression (MKLPSVR). Subsequently, the optimization objectives and inequality constraints in MKLPSVR have been modified to incorporate the prior knowledge from a simulation model, which leads to multi-kernel linear programming prior knowledge support vector regression (MKPLPSVR). By incorporating the prior knowledge from a calibrated simulator into MKLPSVR, one can reduce the effect of the biased data from a simulation model on the accuracy of the data-based model. The development of MKPLPSVR is explained in the followings.

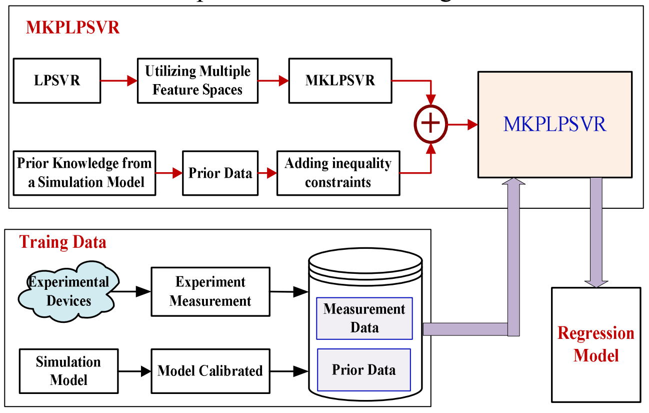


Fig.1. Block diagram of MKPLPSVR development

A. MKLPSVR

As far as LPSVR, the function in the feature space is expressed as (1). However, the non-flat function or complicated data trend cannot be described properly in a single feature space. It is promising to seek a space which can utilize the advantage of different feature spaces [**16**]. Therefore, it may be better choice to consider the regression problem in the multiple feature spaces $\boldsymbol{\omega}_1,\cdots\boldsymbol{\omega}_L$, and the function in multiple feature spaces can be written as:

$$f(\boldsymbol{x})=\sum_{r=1}^{L}\boldsymbol{\omega}_r\cdot\varphi(\boldsymbol{x})+b \tag{9}$$

Substituting $\boldsymbol{\omega}=\sum_{i=1}^{N}\alpha_i\varphi(\boldsymbol{x}_i)$ into(9), one can express the regression function as:

$$f(\boldsymbol{x})=\sum_{r=1}^{L}\sum_{i=1}^{N}\alpha_{ri}k_r(\boldsymbol{x},\boldsymbol{x}_i)+b \tag{10}$$

where L denotes the number of the kernels which are induced by a set of different feature spaces $\boldsymbol{\omega}_1,\cdots\boldsymbol{\omega}_L$. The function $k_r(\boldsymbol{x},\boldsymbol{x}_i)=\varphi(\boldsymbol{x}_{ri})\cdot\varphi(\boldsymbol{x})$ denotes the r-th kernel, and α_{ri} is the coefficient of the corresponding kernel.

Utilizing the method in (6), one can reformulate the function in (10) as:

$$f(\boldsymbol{x})=\sum_{r=1}^{L}\sum_{i=1}^{N}(\alpha_{ri}^{+}-\alpha_{ri}^{-})k_r(\boldsymbol{x},\boldsymbol{x}_i)+b \tag{11}$$

Eq.(11) can be estimated by minimizing the risk (3) like the previous method. Since the target to be estimated is a complicated data-trend function, the minimization of the regularization term means the maximum of the function fatness, which may result in under-fitting result [**16**]. To avoid the problem, we have utilized the generalization theory to control the regularization term by introducing a non-negative constant C_r. Therefore, analogous to (3), the risk function in a multi-kernel framework can be expressed as:

$$\text{Min: }\sum_{r=1}^{L}C_r\|\boldsymbol{\alpha}_r\|_1+2C\sum_{i=1}^{N}L(y_i-f(\boldsymbol{x}_i)) \tag{12}$$

where $Q(\boldsymbol{a})=\sum_{r=1}^{L}C_r\|\boldsymbol{\alpha}_r\|_1$ is a regularization term, and the constant C_r penalizes nonzero coefficients $\boldsymbol{\alpha}_r$. The vector $\boldsymbol{\alpha}_r=[\alpha_{r1},\alpha_{ri},\cdots\alpha_{rN}]^{\mathrm{T}}$ denotes the coefficient of the r-th kernel, and non-zero elements in the vector $\boldsymbol{\alpha}_r$ are also called support vectors.

Utilizing the method in (6) and (7), MKLPSVR is expressed as:

$$\text{Find: }\alpha_{ri}^{+},\alpha_{ri}^{-},\xi_i,b$$

$$\text{Min: }\sum_{r=1}^{L}C_r\sum_{i=1}^{N}(\alpha_{ri}^{+}+\alpha_{ri}^{-})+2C\sum_{i=1}^{N}\xi_i$$

$$\text{s.t.}\begin{cases}y_i-\sum_{r=1}^{L}\sum_{j=1}^{N}(\alpha_{rj}^{+}-\alpha_{rj}^{-})k_r(\boldsymbol{x}_i,\boldsymbol{x}_j)-b\le\varepsilon+\xi_i\\ \sum_{r=1}^{L}\sum_{j=1}^{N}(\alpha_{rj}^{+}-\alpha_{rj}^{-})k_r(\boldsymbol{x}_i,\boldsymbol{x}_j)+b-y_i\le\varepsilon+\xi_i\\ \alpha_{rj}^{+}\ge0,\alpha_{rj}^{-}\ge0\\ \xi_i\ge0\qquad(\forall i=1,2,\cdots,N)\end{cases} \tag{13}$$

where C_r depends on the kernel parameter of the used kernel function. The coefficient $\alpha_{ri}^{+},\alpha_{ri}^{-}$ satisfy $\alpha_{ri}=\alpha_{ri}^{+}-\alpha_{ri}^{-}$ and $|\alpha_{ri}|=\alpha_{ri}^{+}+\alpha_{ri}^{-}$.

Utilizing linear programming to solve (13), one can obtain the function as shown in (11).

B. MKPLPSVR

In practice, it is complex and costly to obtain sufficient measured data. The amount of experimental data is so little that a satisfactory model cannot be obtained. On the other hand, a simulation model built from some physical knowledge is available [27, 28]. Using a calibrated simulator, one can obtain enough prior data, but which may be biased from the measured results. In order to reduce the effect of the biased prior data, this subsection will present an approach to incorporate the prior data from a calibrated simulator into MKLPSVR.

Let the prior dataset $\mathrm{P}=\{(\boldsymbol{z}_k^p,y_k^p),\boldsymbol{z}_k^p\in\mathrm{R}^d,y_k^p\in\mathrm{R},k=1,2,\cdots N_k\}$ from a simulator. The superscript p and the subscript k refer to the prior knowledge and the number of prior data, respectively. Obviously, the prior data will satisfy the equation in the simulator:

$$f(\boldsymbol{z}_k^p)=y_k^p\quad(k=1,2,\cdots,N_k) \tag{14}$$

The equality constraints can be added to the formulation (13) without changing the linear programming nature. However, this will lead to an exact fit to the data points, which may not be advised if the prior data is biased from the measured results. Moreover, all the equality constraints may lead to an unfeasible problem if they cannot be satisfied simultaneously [24, 27-29].Therefore, some soft constraints have been utilized in Eq.(14) by introducing a positive slack variable $\boldsymbol{u}=[u_1,u_1,\cdots,u_k]^{\mathrm{T}}$. The slack variable can bound the errors between the prior data $(\boldsymbol{z}_k^p, y_k^p)$ and the regression function $f(\boldsymbol{z}_k^p)$ in the following inequality:

$$\left|y_k^p - f(\boldsymbol{z}_k^p)\right| \le u_k \quad (\forall k=1,2,\cdots,N_k) \tag{15}$$

In order to include almost exact or biased knowledge from a prior simulator, it is possible to authorize violations of the constraints (15) that are less than a threshold ε^p . Therefore, by applying ε -insensitive loss function to the errors u_k , one can obtain the following inequality:

$$\left|y_k^p - f(\boldsymbol{z}_k^p)\right| \le u_k + \varepsilon^p \quad (\forall k=1,2,\cdots,N_k) \tag{16}$$

In order to minimize the errors $\boldsymbol{u}=[u_1,u_1,\cdots,u_k]^{\mathrm{T}}$, the l_1 norm of $\boldsymbol{u}$ is added to (12) by introducing a trade-off parameter λ which can tune the influence of the prior data on the regression function. Therefore, by adding inequality constraints (16) and the l_1 norm of the slack vector, MKLPSVR in (13) has been modified to reduce the influence of biased prior data from a simulator on the modeling accuracy. The modified algorithm which is called as MKPLPSVR is expressed as:

Find: $\alpha_{ri}^+, \alpha_{ri}^-, \xi_i, u_k, b$

$$\text{Min: } \sum_{r=1}^{L} C_r \sum_{i=1}^{N} (\alpha_{ri}^+ + \alpha_{ri}^-) + 2C\sum_{i=1}^{N}\xi_i + \lambda\sum_{k=1}^{N_k} u_k$$

$$\text{s.t.}\begin{cases} y_i - \sum_{r=1}^{L}\sum_{j=1}^{N}(\alpha_{rj}^+ - \alpha_{rj}^-)k_r(\boldsymbol{x}_i,\boldsymbol{x}_j) - b \le \varepsilon + \xi_i \\ \sum_{r=1}^{L}\sum_{j=1}^{N}(\alpha_{rj}^+ - \alpha_{rj}^-)k_r(\boldsymbol{x}_i,\boldsymbol{x}_j) + b - y_i \le \varepsilon + \xi_i \\ y_k^p - \sum_{r=1}^{L}\sum_{j=1}^{N}(\alpha_{rj}^+ - \alpha_{rj}^-)k_r(\boldsymbol{z}_k^p,\boldsymbol{x}_j) - b \le \varepsilon^p + u_k \\ \sum_{r=1}^{L}\sum_{j=1}^{N}(\alpha_{rj}^+ - \alpha_{rj}^-)k_r(\boldsymbol{z}_k^p,\boldsymbol{x}_j) + b - y_k^p \le \varepsilon^p + u_k \\ \alpha_{rj}^+ \ge 0,\ \alpha_{rj}^- \ge 0 \quad (\forall i=1,2,\cdots,N) \\ \xi_i \ge 0,\ \ u_k \ge 0 \quad (\forall k=1,2,\cdots N_k) \end{cases} \tag{17}$$

Using linear programming to solve the formulation, one will obtain a regression function in (11). The solution procedure is summarized below.

Algorithm: MKPLPSVR

```
1. Prepare a training dataset which includes a measured dataset $\{(\boldsymbol{x}_i, \boldsymbol{y}_i), i=1,...,N_m\}$ from a fine model and a prior dataset $\{(\boldsymbol{z}_k^p, y_k^p), k=1,2,\cdots N_k\}$ from a simulator.
2. Determine the total number $N=N_m+N_k$ of the training dataset.
3. Define the number $L$ and kernel parameters of the used kernel functions.
4. Select some hyper-parameters such as the parameters $C$, $\lambda$, $\varepsilon$ and $\varepsilon^p$.
5. Calculate the $N\times N$ kernel matrix $\boldsymbol{K}_r\ (r=1,2,\cdots,L)$ from the training dataset.
6. Calculate the $N_k\times N$ kernel matrix $\boldsymbol{K}_r^p\ (r=1,2,\cdots,L)$ from the prior dataset
7. Construct the vector form of the formulation (17).
8. Solve the optimization by using linprog in Matlab.
9. Computer $\alpha_{ri}=\alpha_{ri}^+-\alpha_{ri}^-$ and obtain the function (11).
```

IV. Experimental Results

In this section, we will exploit a complex function approximating experiment to validate the algorithm proposed in the paper. Moreover, the following two criteria are used for evaluating the generalization performance based on the same test dataset

$$RMSE = \sqrt{N^{-1}\sum_{i=1}^{N}(y_i - f_i(\boldsymbol{x}))^2} \tag{18}$$

$$MAE = \max(|y_i - f_i(\boldsymbol{x})|) \quad (i=1,2,\ldots,N) \tag{19}$$

where $f_i(\boldsymbol{x})$ is the predicted value, y_i is the corresponding actual value, N is the number of testing samples.

A. Example1

In this example, we will utilize the algorithms above to identify a model from a control system. For realizing the identification process, we will use the following equation with the input $x(k)=\sin\left(\frac{2\pi k}{250}\right)$ to obtain some data.

$$\begin{aligned} y(k+1) &= 0.3y(k)+0.6y(k-1)+0.6\sin(\pi x(k)) \\ &\quad +0.3\sin(3\pi x(k))+0.1\sin(5\pi x(k)) \end{aligned} \tag{20}$$

From the function, we have generated 10 training data and 10 priori data separately by adding an independent Gaussian noise $N(0,0.1^2)$. In the range $[0,300]$ of input space of the function, 300 points are taken uniformly for a testing data. Fig. 2 shows the testing data, the training data as well as the priori data.

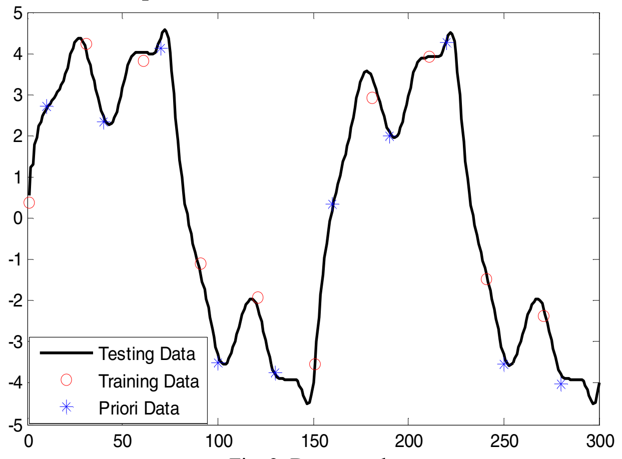


Fig. 2. Data samples

The identified model $g(k)$ can be expressed as:

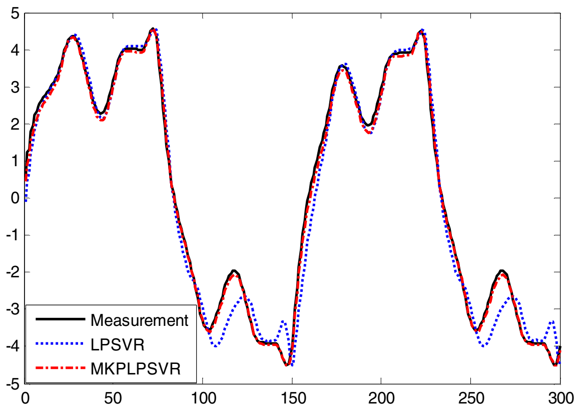
$$y(k) = g\left(y(k-1), y(k-2), f\left(x(k-1)\right)\right) \tag{21}$$

Utilizing the data above, we will identify the function $g(k)$ separately by LPSVR, PLPSVR in [29], MKLPSVR and MKPLPSVR. Two groups of experiments have been also designed to validate the proposed algorithm. In the first group, 10 training samples are used to train the regression function; meanwhile, 10 samples from the prior knowledge are only used during the course of calculating the constraints of the optimization formulation. In the second group, we will firstly extend the 10 training samples with the 10 prior samples, and then use the extended data to train the model. After we have obtained a regression model, we will use the 300 testing data to separately validate the model.

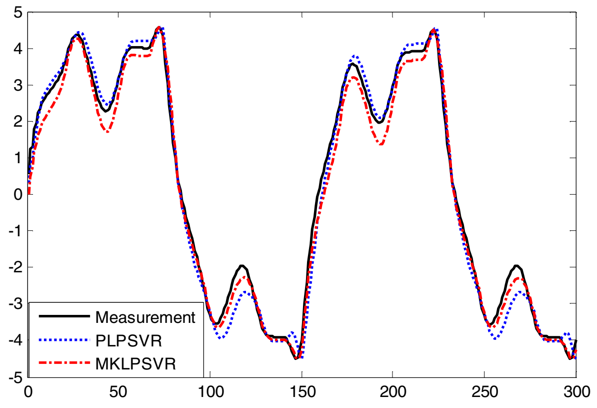
The method of 5-fold cross-validation is used to choose proper model parameters of the four algorithms. In the first group, we choose $C = 100$, $\varepsilon = 0.01$, $\varepsilon_p = 0.001$ for all the algorithms, and both LPSVR and PLPSVR exploit only a Gaussian kernel with the kernel parameter $\sigma = 1$, however, MKLPSVR and MKPLPSVR use a Gaussian kernel and a polynomial kernel, the kernel parameters of which are 1 and 0. 8, respectively. In the second group, we choose $C = 100$, $\varepsilon = 0.01$, $\varepsilon_p = 0.001$. Similarly, both LPSVR and PLPSVR exploit a Gaussian kernel with $\sigma = 1$, however, MKLPSVR and MKPLPSVR use a Gaussian kernel and a polynomial kernel, the kernel parameters of which are 1.5 and 0.8, respectively.

According to the data samples and parameters above, we will establish the models separately by using four algorithms. Fig. 3 shows the approximating results in the first group experiment. Table 1 gives the comparing results of the four regression models calculated by 300 testing data in the first group of experiment. From the table, we can find that the result obtained by MKPLPSVR has the smallest *RMSE* and *MAE*, in spite of the same number of the support vector.

Fig. 4 gives the approximating results of four algorithms in the second group experiment. Table 2 shows the results of the four functions separately calculated by 300 testing data in the second group of experiment.



(a)

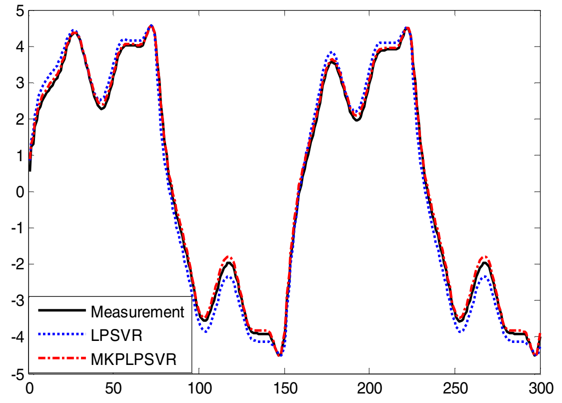


(b)

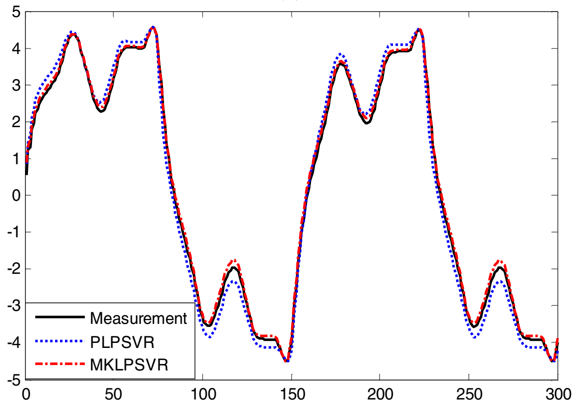
Fig. 3. Comparison in the first group of experiment (a) (b)

TABLE 1 ERRORS AND NUMBER OF SUPPORT VECTOR

Algorithm	NSV	*RMSE*	*MAE*
LPSVR	3	0.482555	1.241319
PLPSVR	3	0.368077	0.982214
MKLPSVR	3	0.324137	0.76772
MKPLPSVR	3	0.114968	0.334509



(a)



(b)

Fig. 4. Comparison in the second group of experiment (a) (b)

TABLE 2 ERRORS AND NUMBER OF SUPPORT VECTOR

Algorithm	NSV	*RMSE*	*MAE*
LPSVR	4	0.296777	0.745348
PLPSVR	4	0.285432	0.716359
MKLPSVR	4	0.112244	0.329322
MKPLPSVR	4	0.109848	0.324031

Compared Table 1 with Table 2, we can find that the model developed in the second group of experiment is more accurate than the one developed in the first group of experiment. The reason is that the prior knowledge has been incorporated into LPSVR, which has improved the accuracy of data-based model. The results can also indicate that that it is feasible to improve the modeling accuracy from a limited amount of experimental data by simultaneously incorporating prior knowledge and multiple kernels into LPSVR, and that the proposed algorithm MKPLPSVR is more effective to approximate a real function than the others which have only exploited multiple kernel or prior knowledge in the framework of LPSVR or QPSVR.

B. *Example 2*

In this experiment, we approximate the following single-variable function which is modified from.

$$y=\begin{cases}-4x-8, & -3\le x<-1\\ -3x^3-5x^2+5x+3, & -1\le x<1\\ 2\sin\left(\exp\left(1.2x\right)\right)+0.3552, & 1\le x\le 3\end{cases} \qquad (22)$$

From the function, we have generated 13 training data and 35 priori data by adding an independent Gaussian noise $N\left(0,0.1^2\right)$. In the range $\left[-3,3\right]$, 201 points are also taken uniformly for a test data. Figure 5 shows the test data, the training data as well as the priori data.

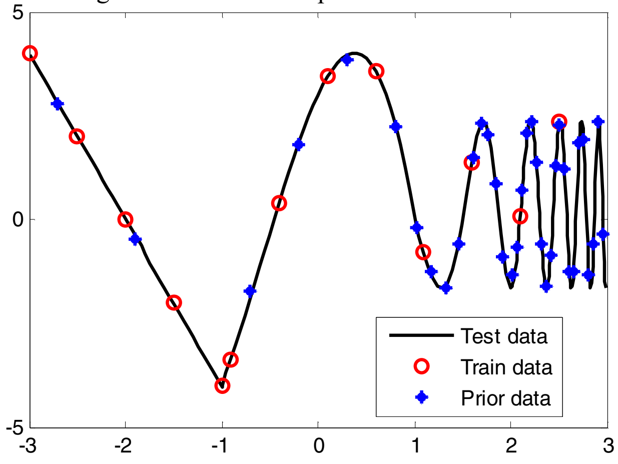


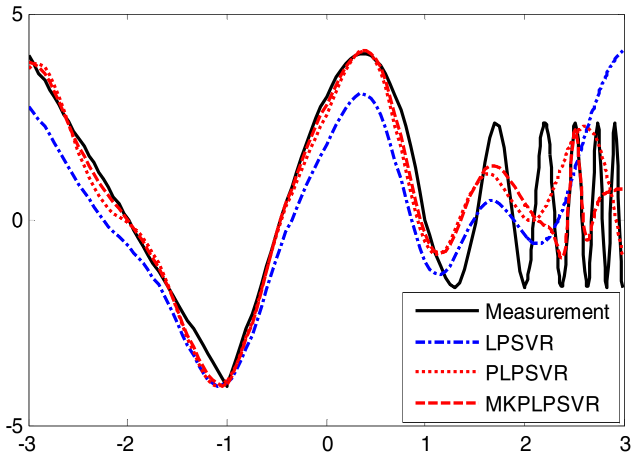
Fig.5 Data samples

Utilizing the data, we will approximate the function separately by LPSVR, MKLPSVR and MKPLPSVR. In addition, SimpleMKL in [31] and PLPSVR in [29] have been used to compare their performance with MKPLPSVR.

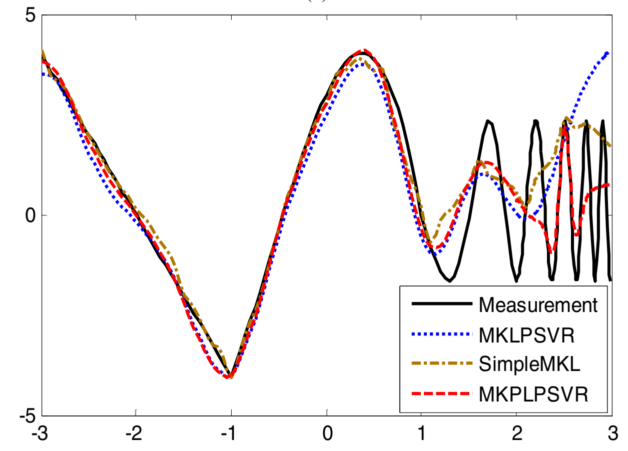
During the course of data-driven modeling, we have designed two groups of experiments. In the first group of experiment, 13 training data will used to develop a regression function, and 35 data samples from prior knowledge are only used during the course of calculating the constraints of the optimization formulation. In the second group of experiment, we will utilize the 35 data samples from prior knowledge to extend the 13 training samples, and then apply the extended data samples to develop a regression function. After obtaining a regression model, we will verify the model by using the 201 testing data.

In the first group of experiment, we have chosen $C=100$, $\varepsilon=0.01$, $\varepsilon^p=0.04$ for all the algorithms. Both LPSVR and PLPSVR exploit only a Gaussian kernel with the kernel parameter $\sigma=0.0803$. However, MKLPSVR, SimpleMKL and MKPLPSVR have employed a Gaussian kernel, a polynomial kernel and a wavelet kernel [30] with the kernel parameters 2, 0.058 and 0.0125, respectively. In the second group, we have chosen $C=150$, $\varepsilon=0.01$, $\varepsilon^p=0.04$ for all the algorithms. Similarly, both LPSVR and PLPSVR only exploit a Gaussian kernel with the kernel parameter $\sigma=0.013$. MKLPSVR, SimpleMKL and MKPLPSVR have utilized a Gaussian kernel, a polynomial kernel and a wavelet kernel with the kernel parameters 2, 0.055 and 0.0122, respectively.

Utilizing the data samples and parameters above, we will establish the models separately by using five algorithms. Fig. 6 shows the approximating results in the first group of experiment.



(a)



(b)

Fig. 6 Comparison of predicted results in the first group of experiment (a) (b)

Fig. 6 shows that all of the algorithms cannot accurately approximate the steep variation of the actual function, due to only the 13 training samples. However, compared with Fig.6 (a) and Fig.6 (b), we can find that the multi-kernel algorithms such as MKLPSVR, MKPLPSVR and

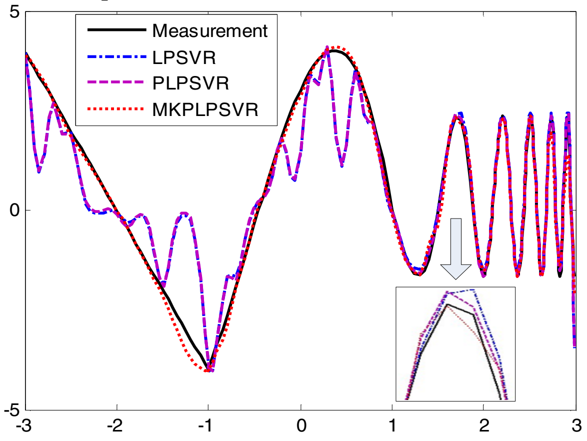
SimpleMKL are able to more accurately approximate the flat variation than other algorithms.

In order to clearly show the performance, we have presented some results verified by 201 testing data in Table 3. From the Table 3, we can also find that the number of support vector (NSV) is almost the same among the functions. However, the model calculated by MKPLPSVR has the smallest *RMSE* and *MAE* among the five models.

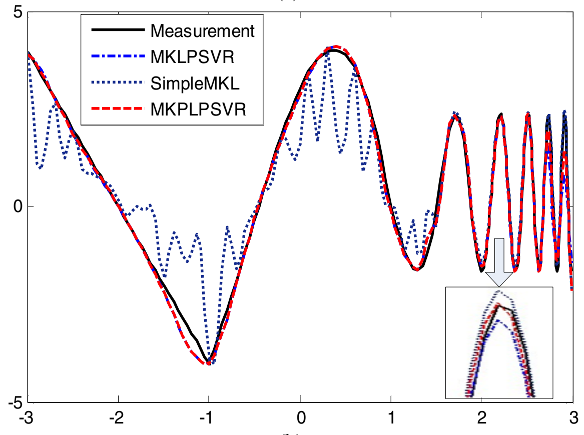
TABLE 3 ERRORS AND NUMBER OF SUPPORT VECTOR

Algorithm	NSV	*RMSE*	*MAE*
LPSVR	11	1.848857	5.735477
PLPSVR	12	1.328779	3.880107
MKLPSVR	12	1.845337	5.735477
SimpleMKL	12	1.4378	3.8105
MKPLPSVR	12	0.972362	2.486516

Fig. 7 shows the approximating results in the second group of experiment. From Fig.7(a), we can find that all of the algorithms can approximate the steep variation of the function. However, the results from Fig.7 (b) show that other algorithms besides MKPLPSVR and MKLPSVR cannot accurately approximate the flat variation. A possible explanation is that incorporating multi-kernel into LPSVR has improved the accuracy of approximating a function with both the steep variation and smooth variation.



(a)



(b)

Fig. 7 Comparison of predicted results in the second group of experiment (a) (b)

Table 4 shows the results of the five regression functions separately calculated by 201 testing data. From the Table 4, we can find that the function developed by MKPLPSVR is the most accurate among all of the functions. In addition, compared with SimpleMKL, MKPLPSVR is also advantageous over SimpleMKL in the aspects of model sparsity and generation performance.

TABLE 4 ERRORS AND NUMBER OF SUPPORT VECTOR

Algorithm	NSV	*RMSE*	*MAE*
LPSVR	37	0.737101	2.988316
PLPSVR	37	0.736426	2.053115
MKLPSVR	37	0.325079	1.573869
SimpleMKL	42	0.7522	2.7198
MKPLPSVR	37	0.212162	0.955358

Compared with Table 3 and Table 4, we can find that the function developed in the second group of experiment is more accurate than the one developed in the first group of experiment. The reason is that the prior knowledge has been incorporated into LPSVR. The results also indicate that that it is feasible to improve the modeling accuracy from a limited amount of experimental data by simultaneously incorporating prior knowledge and multi-kernel function into the learning framework of LPSVR, and that MKPLPSVR is more effective to approximate a real function than the other algorithms.

V. CONCLUSION

This paper has presented a multi-kernel linear program support vector regression with prior knowledge to solve the modeling problem of small data set. By modifying optimization objectives and inequality constraints, we can incorporate the data which is possible biased from a prior simulator into the multi-kernel linear programming support vector regression. Synthetic examples show the effectiveness of the proposed algorithm, and comparing results show that the proposed MKPLPSVR algorithm is more effective to approximate a real function than the others which have only exploited multiple kernel or prior knowledge in the framework of LPSVR or QPSVR. The proposed algorithm can be exploited in some engineering such as computer-aided modeling, system identification as well as auto-tuning system of microwave filters. In particular, if there is an insufficient amount of measured data, one can improve the data-based modeling accuracy by using MKPLPSVR. In the future, we will solve the problem how to choose a proper parameter for the MKPLPSVR.

REFERENCES

[1] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Berlin: Springer-Verlag, 1995.
[2] B. S. A. J. Smola, "A tutorial on support vector regression," *Statistics and Computing,* vol. 14, p. 199~220, 2004.
[3] J. S.-T. N. Cristianini, *An Introduction to Support Vector Machines*. Cambridge: Cambridge University Press, 2000.
[4] K. R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE Transactions on Neural Networks,* vol. 12, pp. 181-201, 2001.
[5] B. S. o. a. A. Smola, *Learning with Kernels*. Cambridge: MIT Press, 2002.
[6] A. Smola, B. Schoelkopf, and G. Raetsch, "Linear programs for automatic accuracy control in regression," in *Proceedings of the*

Ninth International Conference on Artificial Neural Networks, Edinburgh, UK, 1999, pp. 575-580.

[7] Z. Lu, J. Sun, and K. R. Butts, "Linear programming support vector regression with wavelet kernel: A new approach to nonlinear dynamical systems identification," *Mathematics and Computers in Simulation,* vol. 79, pp. 2051-2063, 2009.

[8] Z. Lu and J. Sun, "Non-Mercer hybrid kernel for linear programming support vector regression in nonlinear systems identification," *Applied Soft Computing Journal,* vol. 9, pp. 94-99, 2009.

[9] Z. Yong-Ping and S. Jian-Guo, "Multikernel semiparametric linear programming support vector regression," *Expert Systems with Applications,* vol. 38, pp. 1611-18, 2011.

[10] S. M. Clarke, J. H. Griebsch, and T. W. Simpson, "Analysis of support vector regression for approximation of complex engineering analyses," *Transactions of the ASME. Journal of Mechanical Design,* vol. 127, pp. 1077-87, 2005.

[11] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine Learning Research,* vol. 5, 2004.

[12] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," Banff, Alta, Canada, 2004, pp. 41-48.

[13] S. Sonnenburg, G. Ratsch, C. Schafer, and B. Scholkopf, "Large scale multiple kernel learning," *Journal of Machine Learning Research,* vol. 7, pp. 1531-1565, 2006.

[14] S. Qiu and T. Lane, "A framework for multiple kernel support vector regression and its applications to siRNA efficacy prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics,* vol. 6, pp. 190-199, 2009.

[15] C.-V. Nguyen and D. B. H. Tay, "Regression using multikernel and semiparametric support vector algorithms," *IEEE Signal Processing Letters,* vol. 15, pp. 481-484, 2008.

[16] D. Zheng, J. Wang, and Y. Zhao, "Non-flat function estimation with a multi-scale support vector regression," *Neurocomputing,* vol. 70, pp. 420-429, 2006.

[17] Y. Yu and F. Qian, "Multi-scale linear programming support vector regression for ethylene distillation modeling," Chongqing, China, 2008, pp. 1548-1552.

[18] N. Subrahmanya and Y. C. Shin, "Sparse multiple kernel learning for signal processing applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 32, pp. 788-798.

[19] Mingqing Hu, Yiqiang Chen, and J. T. Y. Kwok, "Building sparse multiple-kernel SVM classifiers," *IEEE Transactions on Neural Networks,* vol. 20, pp. 827-39, 2009.

[20] V. D. Sanchez A, "Advanced support vector machines and kernel methods," *Neurocomputing,* vol. 55, pp. 5-20, 2003.

[21] F. Lauer and G. Bloch, "Incorporating prior knowledge in support vector machines for classification: A review," *Neurocomputing,* vol. 71, pp. 1578-1594, 2008.

[22] P. Trnka and V. Havlena, "Subspace like identification incorporating prior information," *Automatica,* vol. 45, pp. 1086-1091, 2009.

[23] G. Bloch, F. Lauer, G. Colin, and Y. Chamaillard, "Support vector regression from simulation data and few experimental samples," *Information Sciences,* vol. 178, pp. 3813-3827, 2008.

[24] F. Lauer and G. Bloch, "Incorporating prior knowledge in support vector regression," *Machine Learning,* vol. 70, pp. 89-118, 2008.

[25] J. W. S. O. L. Mangasarian, and E. W. Wild, "Knowledge based kernel approximation," *Journal of Machine Learning Research,* vol. 5, pp. 1127–1141, 2004.

[26] O. L. Mangasarian and E. W. Wild, "Nonlinear knowledge in kernel approximation," *IEEE Transactions on Neural Networks,* vol. 18, pp. 300-306, 2007.

[27] J. Zhou J. Huang, "Support-vector modeling and optimization for microwave filters manufacturing using small data sets," in *2012 10th IEEE International Conference on Industrial Informatics (INDIN)* China, 2012, pp. 202-207.

[28] J. Zhou, B. Duan, J. Huang, "Support-vector modeling of electromechanical coupling for microwave filter tuning," *International Journal of RF and Microwave Computer-Aided Engineering,* vol. 23, 2013.

[29] J. Zhou, J. Huang, "Incorporating priori knowledge into linear programming support vector regression," in *2010 IEEE International Conference on Intelligent Computing and Integrated Systems, ICISS2010, October 22, 2010 - October 24, 2010*, Guilin, China, 2010, pp. 591-595.

[30] L. Zhang, W. Zhou, and L. Jiao, "Wavelet Support Vector Machine," *IEEE Transactions on Systems, Man, and Cybernetics, Part B:* Cybernetics, vol. 34, pp. 34-39, 2004.

[31] Rakotomamonjy A, Bach FR, Canu S, Grandvalet Y, "SimpleMKL," *Journal of Machine Learning Research* , vol. 9, pp. 2491-2521,2008.