Pattern Recognition Letters 34 (2013) 1078–1084

Contents lists available at SciVerse ScienceDirect

Pattern Recognition Letters

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

Multi-output least-squares support vector regression machines

Shuo Xu [a], Xin An [b], Xiaodong Qiao [a], Lijun Zhu [a], Lin Li [c,*]

[a] Information Technology Supporting Center, Institute of Scientific and Technical Information of China No. 15 Fuxing Rd., Haidian District, Beijing 100038, China
[b] School of Economics and Management, Beijing Forestry University No. 35 Qinghua East Rd., Haidian District, Beijing 100038, China
[c] College of Information and Electrical Engineering, China Agricultural University No. 17 Qinghua East Rd., Haidian District, Beijing 100083, China

ARTICLE INFO

Article history:
Received 10 May 2012
Available online 4 February 2013

Communicated by S. Sarkar

Keywords:
Least-squares support vector regression machine (LS-SVR)
Multiple task learning (MTL)
Multi-output LS-SVR (MLS-SVR)
Positive definite matrix

ABSTRACT

Multi-output regression aims at learning a mapping from a multivariate input feature space to a multivariate output space. Despite its potential usefulness, the standard formulation of the least-squares support vector regression machine (LS-SVR) cannot cope with the multi-output case. The usual procedure is to train multiple independent LS-SVR, thus disregarding the underlying (potentially nonlinear) cross relatedness among different outputs. To address this problem, inspired by the multi-task learning methods, this study proposes a novel approach, Multi-output LS-SVR (MLS-SVR), in multi-output setting. Furthermore, a more efficient training algorithm is also given. Finally, extensive experimental results validate the effectiveness of the proposed approach.

1. Introduction

By changing the inequality constraints in the support vector regression machine (SVR) (Vapnik, 1999; Vapnik, 1998) by the equality ones, the least-squares SVR (LS-SVR) (Saunders et al., 1998; Suykens and Vandewalle, 1999; Suyken et al., 2002) replaces convex quadratic programming problem with convex linear system solving problem, thus largely speeding up training. It has been shown through a meticulous empirical study that the generalization performance of the LS-SVR is comparable to that of the SVR (Van Gestel et al., 2004). Therefore, the LS-SVR has been attracting extensive attentions during the past few years, such as (An et al., 2009; Choi, 2009; Xu et al., 2011b; Xu et al., 2011a) and references therein.

Multi-output regression aims at learning a mapping from a multivariate input space to a multivariate output space. Compared with the counterpart classification problem—multi-label classification problem (Tsoumakas and Katakis, 2007), the multi-output regression problem remains largely under-studied. To the best of our knowledge, only PLS (Partial Least Squares) regression (Abdi, 2003), kernel PLS regression (Rosipal and Trejo, 2001), MSVR (Multi-output SVR) (Tuia et al., 2011), and multi-output regression on the output manifold (Liu and Lin, 2009) have been put forward in literatures. What is more, it is difficult to generalize directly multi-label classification methods to counterpart regression ones.

Despite its potential usefulness, the standard formulation of the LS-SVR cannot cope with the multi-output case. The usual procedure considers developing a different LS-SVR to learn each parameter individually. That is to say, traditional approach treats the different outputs separately in the multi-output case, thus disregarding the underlying (potentially nonlinear) cross relatedness among different outputs. However, when there are relations between different outputs, it can be advantageous to learn all outputs simultaneously.

Then the problem is how to model the relatedness between different outputs. In fact, some clues from some multi-task learning methods such as hierarchical Bayesian methods (Bakker and Heskes, 2003; Heskes, 2000; Allenby and Rossi, 1998; Arora et al., 1998), which are based on some formal definition of the notion of relatedness of the tasks, motivate this work. Evgeniou and his coworkers (Evgeniou and Pontil, 2004; Evgeniou et al., 2005) proposed a regularized multi-task learning method by following the intuition of Hierarchical Bayes (Heskes, 2000; Allenby and Rossi, 1998; Arora et al., 1998). Our previous work (Xu et al., 2011b) is also based on the intuition with general setting. But, this paper restricts us to multi-output setting, since this setting permits us to design a more efficient training algorithm.

The organization of the rest of this paper is as follows. After LS-SVR for both single-output and multi-output cases are briefly described in Section 2, a novel multi-output regression approach, MLS-SVR, is proposed in Section 3. Similar to the LS-SVR, one only solves a convex linear system in the training phrase, too. In Section 4 and Section 5, extensive experimental evaluations are conducted, and Section 6 concludes this work.

* Corresponding author. Tel.: + 86 10 62732323.
E-mail addresses: xush@isitc.ac.cn (S. Xu), anxin927@bjfu.edu.cn (X. An), qiaox@isitc.ac.cn (X. Qiao), zhulj@isitc.ac.cn (L. Zhu), lilincau@gmail.com (L. Li).

0167-8655/$ - see front matter © 2013 Published by Elsevier B.V.
http://dx.doi.org/10.1016/j.patrec.2013.01.015

Notation

The following notations will be used in this study. Let $\mathbb{R}$ be the set of real numbers and $\mathbb{R}_+$ the subset of positive ones. For every $n \in \mathbb{N}$, the set of positive integers, we let $\mathbb{N}_n = \{1, 2, \ldots, n\}$. A vector will be written in bold case $\mathbf{x} \in \mathbb{R}^d$ with x_i as its i-th elements. The transpose of $\mathbf{x}$ is written as $\mathbf{x}^{\mathrm{T}}$. The vector $\mathbf{1}_d = [1, 1, \ldots, 1]^{\mathrm{T}} \in \mathbb{R}^d$ and $\mathbf{0}_d = [0, 0, \ldots, 0]^{\mathrm{T}} \in \mathbb{R}^d$. The inner product between two vectors is defined as $\mathbf{x}^{\mathrm{T}}\mathbf{z} = \sum_{k=1}^{d} x_k z_k$.

Matrices are denoted by capital bold letters $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $A_{i,j}$ as its (i,j)-th elements. The transpose of $\mathbf{A}$ is written as $\mathbf{A}^{\mathrm{T}}$. If $\mathbf{A}$ is an $m \times n$ matrix, we denote by $\mathbf{a}^i \in \mathbb{R}^m$ and $\mathbf{a}_j \in \mathbb{R}^n$ the i-th row and the j-th column of $\mathbf{A}$, respectively. If $\mathbf{A}$ is an $m \times m$ matrix, we define $trace(\mathbf{A}) := \sum_{i=1}^{m} A_{i,i}$. The identity matrix of dimension $m \times m$ is written as $\mathbf{I}_m$.

The function $repmat(\mathbf{A}, m, n)$ or $repmat(\mathbf{x}, m, n)$ creates a large block matrix consisting of an $m \times n$ tiling of copies of $\mathbf{A}$ or $\mathbf{x}$. The function $blockdiag(\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n)$ or $blockdiag(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ creates a block diagonal matrix, having $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n$ or $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ as main diagonal blocks, with all other blocks being zero matrices.

2. Least-squares support vector regression machine (LS-SVR)

2.1. Single-output case

The single-output regression is regarded as finding the mapping between an incoming vector $\mathbf{x} \in \mathbb{R}^d$ and an observable output $y \in \mathbb{R}$ from a given set of independent and identically distributed (i.i.d.) samples, i.e., $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$. Let $\mathbf{y} = (y_1, y_2, \ldots, y_l)^{\mathrm{T}} \in \mathbb{R}^l$. The single-output LS-SVR solves this problem by finding $\mathbf{w} \in \mathbb{R}^{n_h}$ and $b \in \mathbb{R}$ that minimizes the following objective function with constraints:

$$\min_{\mathbf{w} \in \mathbb{R}^{n_h}, b \in \mathbb{R}} \mathcal{J}(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \gamma \frac{1}{2}\xi^{\mathrm{T}}\xi, \tag{1}$$

$$\text{s.t.} \quad \mathbf{y} = \mathbf{Z}^{\mathrm{T}}\mathbf{w} + b\mathbf{1}_l + \xi, \tag{2}$$

where $\mathbf{Z} = (\varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2), \ldots, \varphi(\mathbf{x}_l)) \in \mathbb{R}^{n_h \times l}$, $\varphi : \mathbb{R}^d \to \mathbb{R}^{n_h}$ is a mapping to some higher (maybe infinite) dimensional Hilbert space $\mathcal{H}$ (also known as feature space) with n_h dimensions, $\xi = (\xi_1, \xi_2, \ldots, \xi_l)^{\mathrm{T}} \in \mathbb{R}^l$ is a vector consisting of slack variables, and $\gamma \in \mathbb{R}_+$ is a positive real regularized parameter.

The Lagrangian function for the problem 1,2 is

$$\mathcal{L}(\mathbf{w}, b, \xi, \alpha) = \mathcal{J}(\mathbf{w}, \xi) - \alpha^{\mathrm{T}}(\mathbf{Z}^{\mathrm{T}}\mathbf{w} + b\mathbf{1}_l + \xi - \mathbf{y}), \tag{3}$$

where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_l)^{\mathrm{T}} \in \mathbb{R}^l$ is a vector consisting of Lagrange multipliers. The Karush–Kuhn–Tucker (KKT) conditions for optimality yield the following set of linear equations:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 0 & \Rightarrow \quad \mathbf{w} = \mathbf{Z}\alpha, \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \Rightarrow \quad \alpha^{\mathrm{T}}\mathbf{1}_l = 0, \\ \frac{\partial \mathcal{L}}{\partial \xi} = 0 & \Rightarrow \quad \alpha = \gamma\xi, \\ \frac{\partial \mathcal{L}}{\partial \alpha} = 0 & \Rightarrow \quad \mathbf{Z}^{\mathrm{T}}\mathbf{w} + b\mathbf{1}_l + \xi - \mathbf{y} = \mathbf{0}_l. \end{cases} \tag{4}$$

By eliminating $\mathbf{w}$ and ξ, one can obtain the following linear system:

$$\begin{bmatrix} 0 & \mathbf{1}_l^{\mathrm{T}} \\ \mathbf{1}_l & \mathbf{H} \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{y} \end{bmatrix}, \tag{5}$$

with the positive definite matrix $\mathbf{H} = \mathbf{K} + \gamma^{-1}\mathbf{I}_l \in \mathbb{R}^{l \times l}$. Here, $\mathbf{K} = \mathbf{Z}^{\mathrm{T}}\mathbf{Z} \in \mathbb{R}^{l \times l}$ is defined by its elements $K_{i,j} = \varphi(\mathbf{x}_i)^{\mathrm{T}}\varphi(\mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ for $\forall (i,j) \in \mathbb{N}_l \times \mathbb{N}_l$, and $\kappa(\cdot,\cdot)$ is a kernel function meeting the Mercer's theorem (Vapnik, 1999; Vapnik, 1998).

However, it is more difficult to solve directly the linear system (5), since its coefficient matrix is not positive definite. This can be overcome by reformulating it into the following one (Suyken et al., 2002; Suykens et al., 1999)

$$\begin{bmatrix} s & \mathbf{0}_l^{T} \\ \mathbf{0}_l & \mathbf{H} \end{bmatrix} \begin{bmatrix} b \\ \alpha + b\mathbf{H}^{-1}\mathbf{1}_l \end{bmatrix} = \begin{bmatrix} \mathbf{1}_l^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{y} \\ \mathbf{y} \end{bmatrix}, \tag{6}$$

where $s = \mathbf{1}_l^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{1}_l \in \mathbb{R}_+$. This new linear system (6) has a unique solution, and thus opens many opportunities for using fast and efficient numerical optimization methods. In fact, the solution of the problem (6) can be found by the following three steps (Suyken et al., 2002; Suykens et al., 1999):

1. Solve η, v from $\mathbf{H}\eta = \mathbf{1}_l$ and $\mathbf{H}v = \mathbf{y}$;
2. Compute $s = \mathbf{1}_l^{\mathrm{T}}\eta$;
3. Find solution: $b = \eta^{\mathrm{T}}\mathbf{y}/s, \alpha = v - b\eta$.

Therefore, the solution of the training procedure can be found by solving two sets of linear equations with the same positive definite coefficient matrix $\mathbf{H} \in \mathbb{R}^{l \times l}$. Since $\mathbf{H}$ is positive definite, one typically first finds the Cholesky decomposition $\mathbf{H} = \mathbf{L}\mathbf{L}^{\mathrm{T}}$. Then since $\mathbf{L}$ is lower triangular, solving the system is simply a matter of applying forward and backward substitution. Other commonly used methods include the conjugate gradient, single value decomposition (SVD) or eigendecomposition, etc.

Let the solution of (5) be $\alpha^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_l^*)^{\mathrm{T}}$ and b^*. Then, the corresponding decision function is

$$\begin{aligned} f(\mathbf{x}) &= \varphi(\mathbf{x})^{\mathrm{T}}\mathbf{w}^* + b^* = \varphi(\mathbf{x})^{\mathrm{T}}\mathbf{Z}\alpha^* + b^* = \sum_{i=1}^{l} \alpha_i^* \varphi(\mathbf{x})^{\mathrm{T}}\varphi(\mathbf{x}_i) + b^* \\ &= \sum_{i=1}^{l} \alpha_i^* \kappa(\mathbf{x}, \mathbf{x}_i) + b^*. \end{aligned} \tag{7}$$

Thus, the single-output LS-SVR can be solved using only inner products between $\varphi(\cdot)$s, not needing to know the nonlinear mapping. However, in contrast to SVR, α^* is not sparse. This means that the whole training set needs to be used at prediction time.

2.2. Multi-output case

One can easily extend the single-output regression to the multiple output case (An et al., 2009). Let $\mathbf{Y} = [y_{ij}] \in \mathbf{R}^{l \times m}$. Given a set of i.i.d. samples $\{(\mathbf{x}_i, \mathbf{y}^i)\}_{i=1}^{l}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $\mathbf{y}^i \in \mathbb{R}^m$, the multi-output regression aims at predicting an output vector $\mathbf{y} \in \mathbb{R}^m$ from a given input vector $\mathbf{x} \in \mathbb{R}^d$. That is to say, the multi-output regression problem can be formulated as learning a mapping from $\mathbb{R}^d$ to $\mathbb{R}^m$. The multi-output LS-SVR (MLS-SVR) solves this problem by finding $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m) \in \mathbb{R}^{n_h \times m}$ and $\mathbf{b} = (b_1, b_2, \ldots, b_m)^{\mathrm{T}} \in \mathbb{R}^m$ that minimizes the following objective function with constraints:

$$\min_{\mathbf{W} \in \mathbb{R}^{m \times n_h}, \mathbf{b} \in \mathbb{R}^m} \mathcal{J}(\mathbf{W}, \Xi) = \frac{1}{2} trace(\mathbf{W}^{\mathrm{T}}\mathbf{W}) + \gamma \frac{1}{2} trace(\Xi^{\mathrm{T}}\Xi), \tag{8}$$

$$\text{s.t.} \quad \mathbf{Y} = \mathbf{Z}^{\mathrm{T}}\mathbf{W} + repmat(\mathbf{b}^{\mathrm{T}}, l, 1) + \Xi, \tag{9}$$

where $\Xi = (\xi_1, \xi_2, \ldots, \xi_m) \in \mathbb{R}_+^{l \times m}$.

On closer examination, it is not difficult to see that this is equivalent to m optimization problems similar to the problem 1,2. That is to say, the solution to the regression problem 8,9 decouples between the different output variables, and we need only compute a single inverse matrix, which is shared by all of the vectors $\mathbf{w}_i (\forall i \in \mathbb{N}_m)$. But it is much more efficient to solve 8,9 directly than to solve 1,2 m times, since they all share the same matrix $\mathbf{H} \in \mathbb{R}^{l \times l}$, the inverse matrix of which need be computed only once with the Cholesky decomposition, conjugate gradient, or SVD, etc.

3. Multi-output LS-SVR (MLS-SVR)

In order to formulate the intuition of Hierarchical Bayes (Heskes, 2000; Allenby and Rossi, 1998; Arora et al., 1998), we assume all $\mathbf{w}_i \in \mathbb{R}^{n_h}(i \in \mathbb{N}_m)$ can be written as $\mathbf{w}_i = \mathbf{w}_0 + \mathbf{v}_i$, where the vectors $\mathbf{v}_i \in \mathbb{R}^{n_h}(i \in \mathbb{N}_m)$ are "small" when the different outputs are similar to each other, otherwise the mean vector $\mathbf{w}_0 \in \mathbb{R}^{n_h}$ are "small". Another way to say this is that $\mathbf{w}_0$ carries the information of the commonality and $\mathbf{v}_i(i \in \mathbb{N}_m)$ carries the information of the specialty. Fig. 1 illustrates the intuition underling the MLS-SVR.

To solve $\mathbf{w}_0 \in \mathbb{R}^{n_h}, \mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m) \in \mathbb{R}^{n_h \times m}$ and $\mathbf{b} = (b_1, b_2, \ldots, b_m)^{\mathrm{T}} \in \mathbb{R}^m$ simultaneously, one can minimize the following objective function with constraints:

$$\min_{\mathbf{w}_0 \in \mathbb{R}^{n_h}, \mathbf{V} \in \mathbb{R}^{n_h}, \mathbf{b} \in \mathbb{R}^m} \mathcal{J}(\mathbf{w}_0, \mathbf{V}, \Xi) = \frac{1}{2}\mathbf{w}_0^{\mathrm{T}}\mathbf{w}_0 + \frac{1}{2}\frac{\lambda}{m} trace(\mathbf{V}^{\mathrm{T}}\mathbf{V}) + \gamma\frac{1}{2} trace(\Xi^{\mathrm{T}}\Xi), \tag{10}$$

$$\text{s.t.} \quad \mathbf{Y} = \mathbf{Z}^{\mathrm{T}}\mathbf{W} + repmat(\mathbf{b}^{\mathrm{T}}, l, 1) + \Xi, \tag{11}$$

where $\Xi = (\xi_1, \xi_2, \ldots, \xi_m) \in \mathbb{R}^{l \times m}, \mathbf{W} = (\mathbf{w}_0 + \mathbf{v}_1, \mathbf{w}_0 + \mathbf{v}_2, \ldots, \mathbf{w}_0 + \mathbf{v}_m) \in \mathbb{R}^{n_h \times m}, \mathbf{Z} = (\varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2), \ldots, \varphi(\mathbf{x}_l)) \in \mathbb{R}^{n_h \times l}$, and $\lambda, \gamma \in \mathbb{R}_+$ are two positive real regularized parameters.

The Lagrangian function for the problem 10,11 is

$$\mathcal{L}(\mathbf{w}_0, \mathbf{V}, \mathbf{b}, \Xi, \mathbf{A}) = \mathcal{J}(\mathbf{w}_0, \mathbf{V}, \Xi) - trace(\mathbf{A}^{\mathrm{T}}(\mathbf{Z}^{\mathrm{T}}\mathbf{W} + repmat(\mathbf{b}^{\mathrm{T}}, l, 1) + \Xi - \mathbf{Y})), \tag{12}$$

where $\mathbf{A} = (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \mathbb{R}^{l \times m}$ is a matrix consisting of Lagrange multipliers. The KKT conditions for optimality yield the following set of linear equations:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial \mathbf{w}_0} = 0 & \Rightarrow \quad \mathbf{w}_0 = \sum_{i=1}^{m} \mathbf{Z}\alpha_i, \\ \frac{\partial \mathcal{L}}{\partial \mathbf{V}} = 0 & \Rightarrow \quad \mathbf{V} = \frac{m}{\lambda}\mathbf{Z}\mathbf{A}, \\ \frac{\partial \mathcal{L}}{\partial \mathbf{b}} = 0 & \Rightarrow \quad \mathbf{A}^{\mathrm{T}}\mathbf{1}_l = \mathbf{0}_l, \\ \frac{\partial \mathcal{L}}{\partial \Xi} = 0 & \Rightarrow \quad \mathbf{A} = \gamma\Xi, \\ \frac{\partial \mathcal{L}}{\partial \mathbf{A}} = 0 & \Rightarrow \quad \mathbf{Z}^{\mathrm{T}}\mathbf{W} + repmat(\mathbf{b}^{\mathrm{T}}, l, 1) + \Xi - \mathbf{Y} = \mathbf{0}_{l \times m}. \end{cases} \tag{13}$$

It is easy to see from (13) that the mean vector $\mathbf{w}_0 \in \mathbb{R}^{n_h}$ and the vectors $\mathbf{v}_i \in \mathbb{R}^{n_h}(i \in \mathbb{N}_m)$ meet the following relation: $\mathbf{w}_0 = \frac{\lambda}{m}\sum_{i=1}^{m}\mathbf{v}_i$. In other words, $\mathbf{w}_0$ is a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$. Since for $\forall i \in \mathbb{N}_m, \mathbf{w}_i$ is assumed to be $\mathbf{w}_i = \mathbf{w}_0 + \mathbf{v}_i, \mathbf{w}_i$ can also be expressed as a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$. This suggests that one can obtain an equivalent optimization problem with constraints involving only the $\mathbf{V}$ and $\mathbf{b}$ as follows.

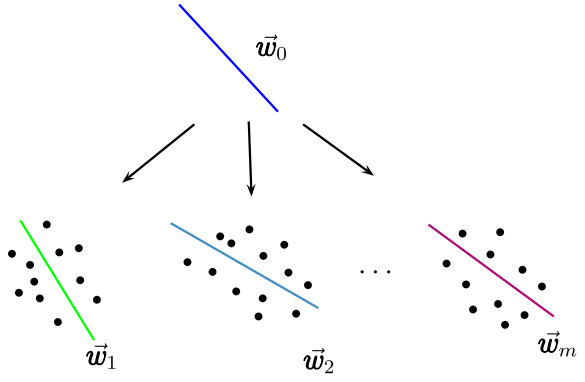


Fig. 1. An illustration of the intuition underlying the MLS-SVR.

$$\min_{\mathbf{V} \in \mathbb{R}^{n_h}, \mathbf{b} \in \mathbb{R}^m} \mathcal{J}(\mathbf{V}, \Xi) = \frac{1}{2}\frac{\lambda^2}{m^2}\mathbf{V}\mathbf{1}_m\mathbf{1}_m^{\mathrm{T}}\mathbf{V}^{\mathrm{T}} + \frac{1}{2}\frac{\lambda}{m} trace(\mathbf{V}^{\mathrm{T}}\mathbf{V}) + \gamma\frac{1}{2} trace(\Xi^{\mathrm{T}}\Xi), \tag{14}$$

$$\text{s.t.} \quad \mathbf{Y} = \mathbf{Z}^{\mathrm{T}}\mathbf{V} + repmat(\frac{\lambda}{m}\mathbf{Z}^{\mathrm{T}}\mathbf{V}\mathbf{1}_m, 1, m) + repmat(\mathbf{b}^{\mathrm{T}}, l, 1) + \Xi. \tag{15}$$

From (14), one can see that our MLS-SVR tries to find a trade off between small size vectors for each output, $trace(\mathbf{V}^{\mathrm{T}}\mathbf{V})$, and closeness of all vectors to the average vector, $\mathbf{V}\mathbf{1}_m\mathbf{1}_m^{\mathrm{T}}\mathbf{V}^{\mathrm{T}}$. But (8) only tries to find small size vectors for each output, which results in decoupling between the different output variables.

Similar to LS-SVR, by eliminating $\mathbf{W}$ and Ξ from (13), one can obtain the following linear system:

$$\begin{bmatrix} \mathbf{0}_{ml \times m} & \mathbf{P}^{\mathrm{T}} \\ \mathbf{P} & \mathbf{H} \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ \alpha \end{bmatrix} = \begin{bmatrix} \mathbf{0}_m \\ \mathbf{y} \end{bmatrix}, \tag{16}$$

where $\mathbf{P} = blockdiag(\overbrace{\mathbf{1}_l, \mathbf{1}_l, \ldots, \mathbf{1}_l}^{m}) \in \mathbb{R}^{ml \times m}$, the positive definite matrix $\mathbf{H} = \Omega + \gamma^{-1}\mathbf{I}_{ml} + (m/\lambda)\mathbf{Q} \in \mathbb{R}^{ml \times ml}, \Omega = repmat(\mathbf{K}, m, m) \in \mathbb{R}^{ml \times ml}$, $\mathbf{Q} = blockdiag(\overbrace{\mathbf{K}, \mathbf{K}, \ldots, \mathbf{K}}^{m}) \in \mathbb{R}^{ml \times ml}, \mathbf{K} = \mathbf{Z}^{\mathrm{T}}\mathbf{Z} \in \mathbb{R}^{l \times l}, \alpha = (\alpha_1^{\mathrm{T}}, \alpha_2^{\mathrm{T}}, \ldots, \alpha_m^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{ml}$, and $\mathbf{y} = (\mathbf{y}_1^{\mathrm{T}}, \mathbf{y}_2^{\mathrm{T}}, \ldots, \mathbf{y}_m^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{ml}$. Thus, the linear system (16) consists of $(l+1) \times m$ equations.

Let the solution of (16) be $\alpha^* = (\alpha_1^{*\mathrm{T}}, \alpha_2^{*\mathrm{T}}, \ldots, \alpha_m^{*\mathrm{T}})^{\mathrm{T}}$ and $\mathbf{b}^*$. Then, the corresponding decision function for the multiple outputs is

$$\begin{aligned} f(\mathbf{x}) &= \varphi(\mathbf{x})^{\mathrm{T}}\mathbf{W}^* + \mathbf{b}^{*\mathrm{T}} = \varphi(\mathbf{x})^{\mathrm{T}} repmat(\mathbf{w}_0^*, 1, m) + \varphi(\mathbf{x})^{\mathrm{T}}\mathbf{V}^* + \mathbf{b}^{*\mathrm{T}} \\ &= \varphi(\mathbf{x})^{\mathrm{T}} repmat(\sum_{i=1}^{m}\mathbf{Z}\alpha_i^*, 1, m) + \frac{m}{\lambda}\varphi(\mathbf{x})^{\mathrm{T}}\mathbf{Z}\mathbf{A}^* + \mathbf{b}^{*\mathrm{T}} \\ &= repmat\left(\sum_{i=1}^{m}\sum_{j=1}^{l}\alpha_{ij}^*\kappa(\mathbf{x}, \mathbf{x}_j), 1, m\right) + \frac{m}{\lambda}\sum_{j=1}^{l}\alpha^{j*}\kappa(\mathbf{x}, \mathbf{x}_j) + \mathbf{b}^{*\mathrm{T}}. \end{aligned} \tag{17}$$

4. More efficient training algorithm

Again, similar to LS-SVR, the linear system (16) is not positive definite, so solving (16) directly is more difficult. But it is reformulated into the following one:

$$\begin{bmatrix} \mathbf{S} & \mathbf{0}_{ml \times ml} \\ \mathbf{0}_{m \times m} & \mathbf{H} \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ \mathbf{H}^{-1}\mathbf{P}\mathbf{b} + \alpha \end{bmatrix} = \begin{bmatrix} \mathbf{P}^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{y} \\ \mathbf{y} \end{bmatrix}, \tag{18}$$

with $\mathbf{S} = \mathbf{P}^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{P} \in \mathbb{R}^{m \times m}$. It is very easy to show that $\mathbf{S}$ is a positive definite matrix. In this way, this new linear system (18) is positive definite, whose solution can be found in the following three steps:

1. Solve η, v from $\mathbf{H}\eta = \mathbf{P}$ and $\mathbf{H}v = \mathbf{y}$;
2. Compute $\mathbf{S} = \mathbf{P}^{\mathrm{T}}\eta$;
3. Find solution: $\mathbf{b} = \mathbf{S}^{-1}\eta^{\mathrm{T}}\mathbf{y}, \alpha = v - \eta\mathbf{b}$.

Therefore, the solution of the training procedure can be found by solving two sets of linear equations with the same positive definite coefficient matrix $\mathbf{H} \in \mathbb{R}^{ml \times ml}$. Since $\mathbf{H}$ is symmetric positive-definite, many fast and efficient numerical optimization methods can be adopted, such as Cholesky decomposition, conjugate gradient, SVD and eigendecomposition, etc. Additionally, since the number of outputs m is usually very small relative to the number of samples l, one can easily obtain the inverse of $\mathbf{S} \in \mathbb{R}_+^{m \times m}$ just using matrix multiplications.

5. Experiments and discussions

In order to assess prediction performance, average relative error $\delta = \frac{1}{l}\sum_{i=1}^{l}\frac{|y_i-\hat{y}_i|}{y_i}$ and correlation coefficient $R = \frac{\sum_{i=1}^{l}(y_i-\bar{y})(\hat{y}_i-\bar{\hat{y}})}{\sqrt{\sum_{i=1}^{l}(y_i-\bar{y})^2\sum_{i=1}^{l}(\hat{y}_i-\bar{\hat{y}})^2}}$ indices are used, where y_i and $\hat{y}_i$ are actual and predicted outputs, respectively, and $\bar{y}$ and $\bar{\hat{y}}$ are averages of actual and predicted outputs, respectively.

Here, the radial basis function (RBF) kernel function is adopted: $\kappa(\mathbf{x},\mathbf{z}) = \exp(-p\|\mathbf{x}-\mathbf{z}\|^2), p > 0$. The reasons are threefold: (a) the linear kernel function is a special case of RBF (Keerthi and Lin, 2003); (b) The Sigmoid kernel function is not positive definite, and for certain parameters, and the Sigmoid kernel function behaves like RBF (Lin and Lin, 2003); (c) Relatively, there are more parameters in the polynomial kernel function so that it is more difficult for model selection. In addition, the polynomial kernel function has also numerical difficulties, such as overflow or underflow.

Finally, in order to identify proper parameters, the grid search (Xu et al., 2007; Hsu et al., 2010) is used. Let $\gamma \in \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}, \lambda \in \{2^{-10}, 2^{-8}, \ldots, 2^{10}\}$ and $p \in \{2^{-15}, 2^{-13}, \ldots, 2^{3}\}$. For all possible combinations (γ, λ, p), the average relative error δ is calculated using leave-one-out (LOO) procedure. Thus, an optimal pair $(\gamma^*, \lambda^*, p^*)$ can be determined. We have implemented all related approaches in MATLAB R2010a on an IBM 3850 M2. The corresponding toolbox can be available from the first author upon request for academic use.

5.1. Synthetic data set

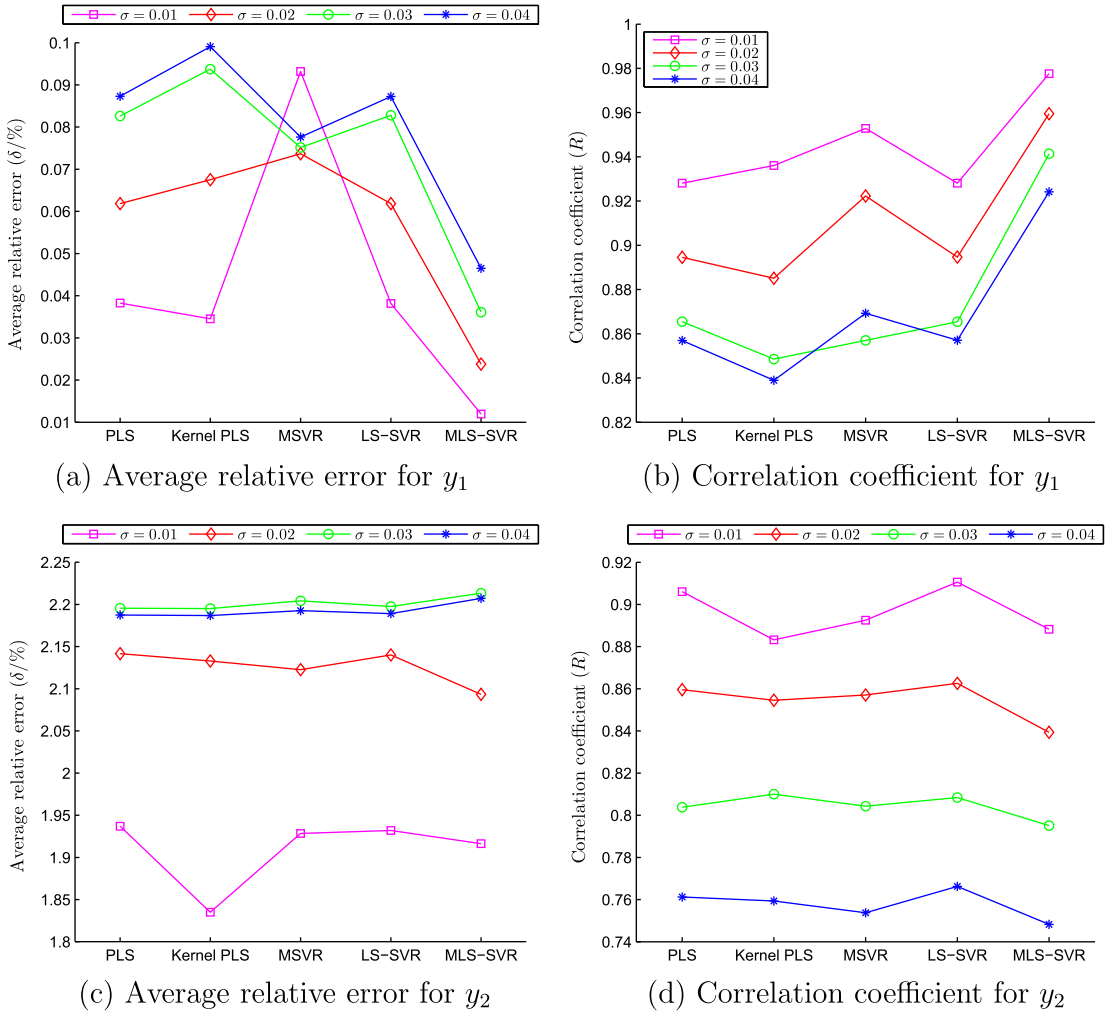
The data set contains 1000 noisy observations, generated using a simulated two-output time series process (Chen, 2002) as follows.

$$y_1(k) = 0.1\sin(\pi y_2(k-1)) + (0.8 - 0.5\exp(-y_1^2(k-1)))y_1(k-1) - (0.3 + 0.9\exp(-y_1^2(k-1)))y_1(k-2) + \epsilon_1(k), \quad (19)$$

$$y_2(k) = 0.6y_2(k-1) + 0.2y_2(k-1)y_2(k-2) + 1.2\tanh(y_1(k-2)) + \epsilon_2(k), \quad (20)$$

given the initial conditions $y_1(0) = y_1(-1) = y_2(0) = y_2(-1) = 0$, where the zero-mean Gaussian noise $\epsilon(k) = (\epsilon_1(k), \epsilon_2(k))^{\mathrm{T}}$ has a covariance $\sigma\mathbf{I}_2$ with $\mathbf{I}_2$ being the 2×2 identity matrix. The first 500 data samples are used for training and the other 500 samples for validating the obtained model. The input vector is given by $\mathbf{x}(k) = (y_1(k-1), y_1(k-2), y_2(k-1), y_2(k-2))^{\mathrm{T}}$. In the study, we let $\sigma \in \{0.01, 0.02, 0.03, 0.04\}$.

The average relative error and correlation coefficient for y_1 and



(a) Average relative error for y_1 (b) Correlation coefficient for y_1

(c) Average relative error for y_2 (d) Correlation coefficient for y_2

Fig. 2. Comparisons of the predicted results on synthetic data set.

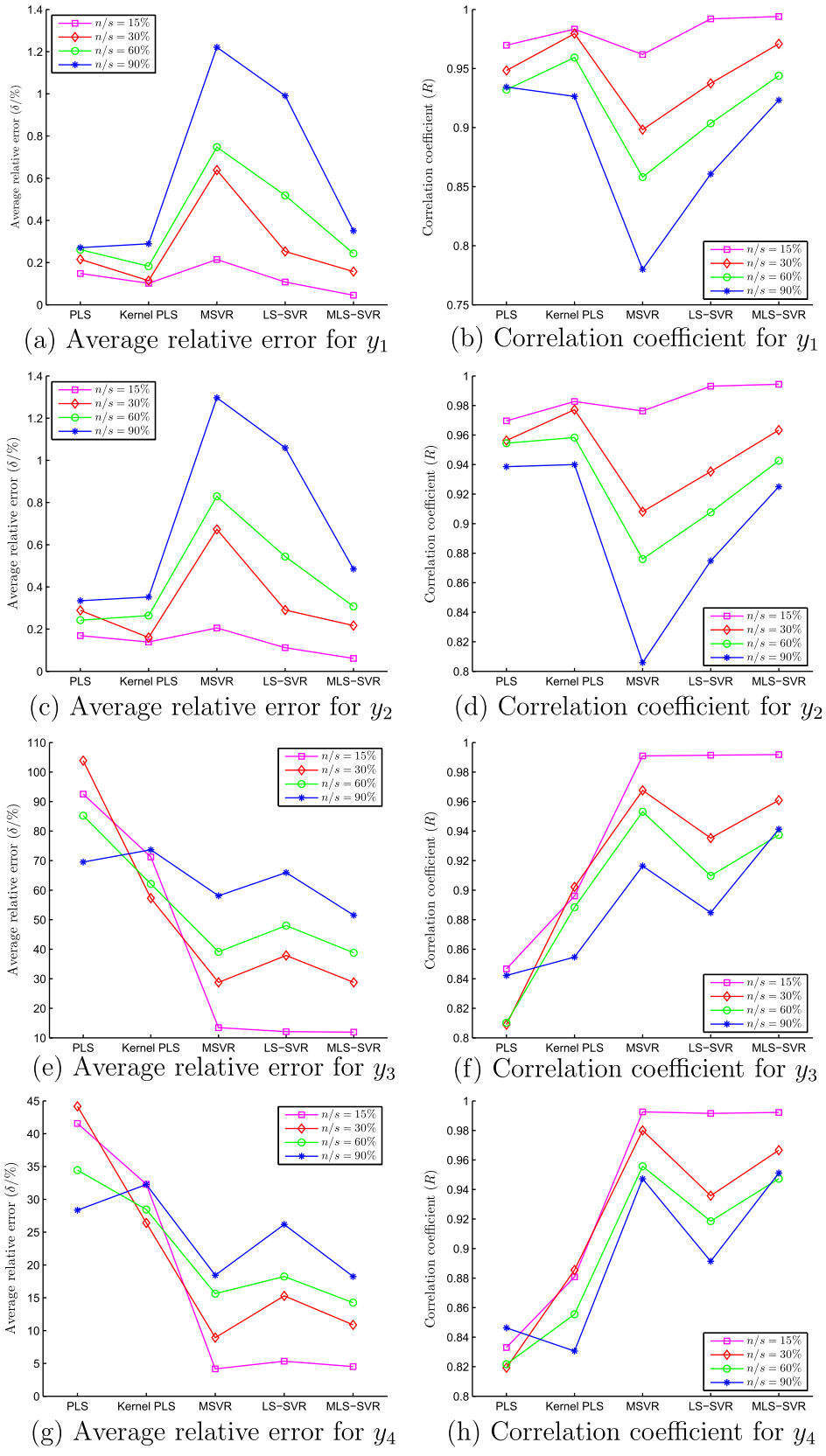


Fig. 3. Comparisons of the predicted results on *corn* data set.

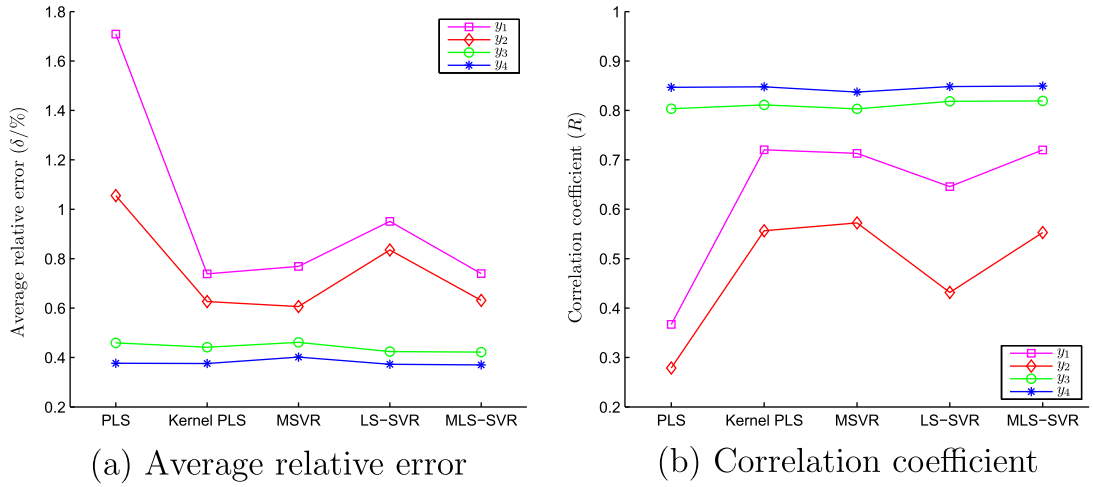


Fig. 4. Comparisons of the predicted results on *polymer* data set.

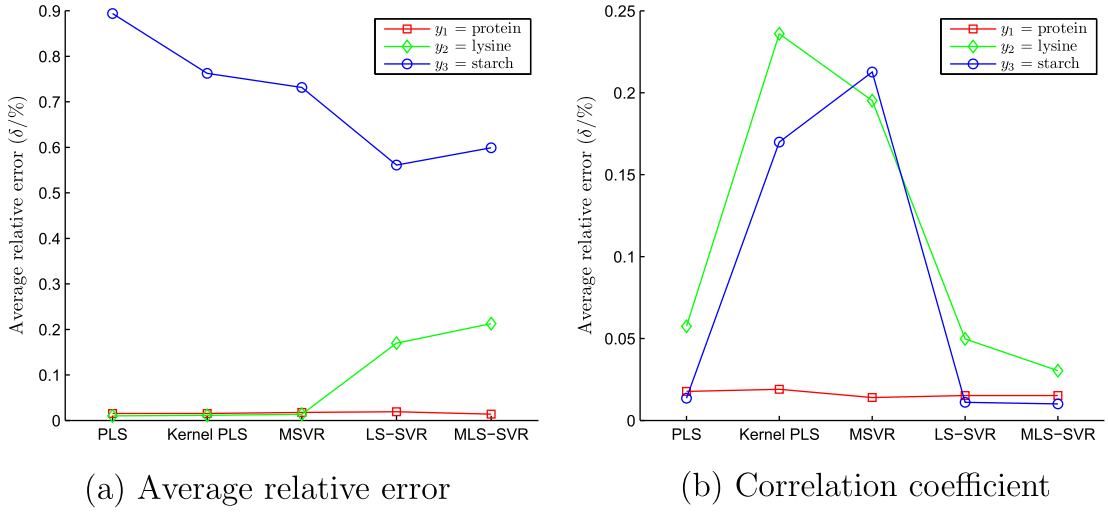


Fig. 5. Comparisons of the predicted results on *broomcorn* data set.

y_2 are plotted in Fig. 2. It can be easily seen that our proposed MLS-SVR can capture the underlying dynamics of the system better that other methods at all different noise conditions. This means that the cross-output information helps improve the performance single output regression methods.

5.2. Corn data set

Corn data set [1] consists of 80 examples of corn measured on 3 different near-infra-red spectrometers, m5, mp5 and mp6. In this study, spectra from instrument m5 are used, where the wavelength range is 1100–2498 nm at 2 nm intervals. The moisture, oil, protein and starch values represent four output/dependent variables. As the first principal component describes 99% of the overall variance, this indicates high multi-collinearity among the input/independent variables. Similar to (Rosipal and Trejo, 2001), instead of modeling the real response we generated four different outputs as follows: $y_1 = \exp(\mathbf{x}^T\mathbf{x}/2c), y_2 = \exp(\mathbf{x}^T\mathbf{B}^{-1}\mathbf{x}/2c_1), y_3 = (\mathbf{x}^T\mathbf{x}/c)^3 \exp(\mathbf{x}^T\mathbf{x}/2c)$, and $y_4 = 0.3y_1 + 0.25y_2 - 0.7y_3$, where $\mathbf{B}$ is a symmetric matrix with off-diagonal elements set to 0.8 and diagonal elements set to 1.0, and c and c_1 are averages of $\{\mathbf{x}_i^T\mathbf{x}_i\}_{i=1}^{80}$ and $\{\mathbf{x}_i^T\mathbf{B}^{-1}\mathbf{x}_i\}_{i=1}^{80}$.

The first 60 examples are used to create a training data set, and the remaining 20 examples form a testing data set. In order to make the synthetic outputs more realistic, Gaussian white noise with different levels is added, where noise level, denoted as n/s, corresponds to ratios of the standard deviation of the noise and the clean output variables. In this study, we set $n/s \in \{15\%, 30\%, 60\%, 90\%\}$. For each noise level, 25 different training sets are generated. It is worth mentioning that the values in Fig. 3 correspond to average of 25 different simulations.

From Fig. 3, it is not difficult to see that our MLS-SVR outperforms obviously LS-SVR. In our opinion, the main reason is that MLS-SVR considers the underlying (potentially nonlinear) cross relatedness among different outputs. For lower noise level, such as $n/s = 15\%, 30\%$ and 60%, MLS-SVR outperforms or matches at least the other multi-output methods. But when $n/s = 90\%$, PLS regression and kernel PLS regression has a certain advantage over MLS-SVR. We think that the main reason is as follows. The PLS regression and kernel PLS regression represents input and output matrix with principal components in input or feature space, so that the information underlying in input and output matrix can be made use of. Furthermore, principal components themselves can greatly reduce the effect of noise.

[1] Corn data set can be available from http://www.eigenvector.com/data/Corn/index.html

5.3. Polymer data set

This data set is taken from a polymer test plant.[2] There are 10 input variables, measurements of controlled variables in a polymer processing plant (temperatures, feed rates, etc.), and 4 output variables are measures of the output of that plant. It is claimed that this data set is particularly good for testing the robustness of nonlinear modeling methods to irregularly spaced data. The first 41 samples are taken as training set, and the remaining 20 samples as testing set.

From Fig. 4, one can conclude that our MLS-SVR is comparable to kernel PLS and MSVR. However, similar performance is also obtained by training 4 independent LS-SVR. Hence one may conjecture that these four outputs are weakly related to each other or unrelated in the way we define in this study.

5.4. Broomcorn data set

The data set (Xu et al., 2011b) consists of 128 samples of broomcorn from Institute of Grop Germplasm Resources, Chinese Academy of Agricultural Sciences. The used instrument is a Bran + Luebbe (Technicon) InfraAlyzer IA 450 NIR spectrophotometer, which gives rapid measurements of protein, lysine and starch components of broomcorn sample. This workhorse instrument covers the 1445 to 2348 nm range with 19 filters, and can be programmed with calibrations to make measurements of up to 10 constituents per product, and for multiple products.

This is a 3-outputs regression problem. The first 96 samples are used to create a training data set, and the remaining 32 samples form a testing data set. Fig. 5 gives the comparisons of the predicted on *broomcorn* data set. From Fig. 5, one can see that all methods, modeling the cross-output information, are superior than LS-SVR. This means that is is advantageous to learn all outputs simultaneously. What's more, the performance of our proposed MLS-SVR is better than other methods in terms of average relative error and correlation coefficient.

6. Conclusions

In this study, we study the multi-output regression problem, which aims at learning a mapping from a multivariate input space to a multivariate output space. Despite its potential usefulness, compared with the counterpart classification problem—multi-label classification problem, the multi-output regression problem remains largely under-studied.

It has been shown through a meticulous empirical study that the generalization performance of the LS-SVR is comparable to that of the SVR. However, the standard formulation of the LS-SVR cannot cope with the multi-output case. The usual procedure is to train multiple independent LS-SVR, thus disregarding the underlying (potentially nonlinear) cross relatedness among different outputs.

To address this problem, inspired by the multi-task learning methods, this study proposes a novel approach in multi-output setting. Furthermore, a more efficient training algorithm is also given. Finally, extensive experimental evaluation is conducted on synthetic, *corn*, *polymer*, and *broomcorn* data sets. The experimental results validate the effectiveness of the proposed approach.

Acknowledgements

This work was funded partially by the "Key Technologies Research on Large Scale Semantic Computation for Foreign Scientific & Technical Knowledge Organization System", "Application Demonstration of Knowledge Service based on STKOS", and "The Key Index of Arable Land Quality Remote Sensing Monitoring Technology" which are sponsored by Key Technologies R&D Program of Chinese 12th Five-Year Plan (2011–2015) under Grant Nos. 2011BAH10B04, 2011BAH10B06 and 2012BAH29B01, respectively; The Predictive Project of Institute of Scientific and Technical Information of China (ISTIC): "Discovery of Domain Topics based on Vocabulary System" under grant YY-201216; National Natural Science Foundation: Multilingual Documents Clustering based on Comparable Corpus under grant 70903032; Social Science Foundation of Jiangsu Province: Study on Automatic Indexing of Digital Newspapers under grant 09TQC011 and MOE Project of Humanitites and Social Sciences: Research on Further Processing of e-Newspaper under grant 09YJC870014. Our gratitude also goes to the anonymous reviewers for their valuable comments.

References

Abdi, H., 2003. Encyclopedia for Research Methods for the Social Sciences. Sage. chapter Partial Least Squares (PLS) Regression. pp. 792–795.
Allenby, G.M., Rossi, P.E., 1998. Marketing models of consumer heterogeneity. J. Econom. 89, 57–78.
An, X., Xu, S., Zhang, L., Su, S., 2009. Multiple dependent variables LS-SVM regression algorithm and its application in NIR spectral quantitative analysis. Spectrosc. Spect. Anal. 29, 127–130.
Arora, N., Allenby, G.M., Ginter, J.L., 1998. A hierarchical bayes model of primary and secondary demand. Market. Sci. 17, 29–44.
Bakker, B., Heskes, T., 2003. Task clustering and gating for bayesian multitask learning. J. Machine Learn. Res. 4, 83–99.
Chen, S., 2002. Multi-output regression using a locally regularised orthogonal least square algorithm. IEE Proc. -Vision Image Signal 149, 185–195.
Choi, Y.S., 2009. Least squares one-class support vector machine. Pattern Recognition Lett. 30, 1236–1240.
Evgeniou, T., Pontil, M., 2004. Regularized multi-task learning. In: Proc. KDD'04, ACM, New York, NY, USA, pp. 109–117.
Evgeniou, T., Micchelli, C.A., Pontil, M., 2005. Learning multiple tasks with kernel methods. J. Machine Learn. Res. 6, 615–637.
Heskes, T., 2000. Empirical bayes for learning to learn. In: Proc. ICML'00. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 367–374.
Hsu, C.W., Chang, C.C., Lin, C.J., 2010. A practical guide to support vector classification, National Taiwan University, Department of Computer Science.
Keerthi, S.S., Lin, C.J., 2003. Asymptotic behaviors of support vector machines with gaussian kernel. Neural Comput. 15, 1667–1689.
Lin, H.T., Lin, C.J., 2003. A study on sigmoid kernels for svm and the training of non-PSD kernels by SMO-type methods. Department of Computer Science, National Taiwan University, Technical Report.
Liu, G., Lin, Z., 2009. Multi-output regression on the output manifold. Pattern Recognition 42, 2737–2743.
Rosipal, R., Trejo, L.J., 2001. Kernel partial least squares regression in reproducing kernel Hilbert space. J. Machine Learn. Res. 2, 97–123.
Saunders, C., Gammerman, A., Vovk, V., 1998. Ridge regression learning algorihtm in dual variables. In: Shavlik, J.W. (Ed.), Proc. ICML'98. Morgan Kaufmann Publishers Inc.,, San Francisco, CA, USA, pp. 515–521.
Suyken, J.A.K., Van Gestel, T., De Brabanter, J., De Moor, B., Vandewalle, J., 2002. Least Squares Support Vector Machines. World Scientific Pub. Co., Singapore.
Suykens, J.A., Vandewalle, J., 1999. Least squares support vector machine classifiers. Neural Process. Lett. 9, 293–300.
Suykens, J.A.K., Lukas, L., Van Dooren, P., De Moor, B., Vandewalle, J., 1999. Least squares support vector machine classifiers: a large scale algorithm. In: Proc. ECCTD'99, pp. 839–842.
Tsoumakas, G., Katakis, I., 2007. Multi-label classification: an overview. Internat. J. Data Warehous Min. 3, 1–13.
Tuia, D., Verrelst, J., Alonso, L., Pérez-Cruz, F., Camps-Valls, G., 2011. Multioutput support vector regression for remote sensing biophysical parameter estimation. IEEE Geosci. Remote Sens. Lett. 8, 804–808.
Van Gestel, T., Suykens, J.A.K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., De Moor, B., Vandewalle, J., 2004. Benchmarking least squares support vector machine classifiers. Machine Learn. 54, 5–32.
Vapnik, V.N., 1998. Statistical Learning Theory. John Wiley & Sons, Inc., New York.
Vapnik, V.N., 1999. The Nature of Statistical Learning Theory, second ed. second ed. Springer Verlag, New York.
Xu, S., Ma, F., Tao, L., 2007. Learn from the information contained in the false splice sites as well as in the true splice sites using SVM. In: Proc. ISKE'07, pp. 1360–1366.
Xu, S., An, X., Qiao, X., Zhu, L., Li, L., 2011a. Semi-supervised least-squares support vector regression machines. J. Inform. Comput. Sci. 8, 885–892.
Xu, S., Qiao, X., Zhu, L., An, X., Zhang, L., 2011b. Multi-task least-square support vector regression machines and their applications in NIR spectral analysis. Spectrosc. Spect. Anal. 31, 1208–1211.

[2] Polymer data set can be available from ftp://ftp.cis.upenn.edu/pub/ungar/chemdata