TELKOMNIKA, Vol.11, No.1, January 2013, pp. 258~269
ISSN: 2302-4046

An Improved Twin Support Vector Regression with Automatic Margin Determination

LIANG Jun*, SHA Zhi-qiang, REN Ying-wen, LI Ao-xue, CHEN Long
Automotive Engineering Research Institute
Jiangsu University, China, 212013, China
*corresponding author, e-mail: liangjun@ujs.edu.cn

Abstract

In this paper, a novel regression algorithm named v-twin support vector regression (v-TSVR) is presented, improving upon the recently proposed twin support vector regression (TSVR). It also tries to seek two nonparallel down- and up-bounds for the unknown function. By treating the size of one-sided ε-insensitive tube as optimization variables with corresponding parameters ν s, we reformulate the original TSVR as a more sensible model. To this end, v-TSVR has the advantage that ε s are learned simultaneously with regressor. Meantime, we give a theoretical result concerning the meaning of ν s. Moreover, by introducing structural risk minimization principle, the over-fitting phenomenon in TSVR can be avoided. We analyze the algorithm theoretically and demonstrate its effectiveness via the experimental results on several artificial and benchmark datasets.

Keywords: *support vector regression, one-sided tube, quadratic programming, ε-insensitive loss*

1. Introduction

The support vector machine (SVM) [1, 2] is one of the leading techniques for its state-of-the-art performance in solving problems emerged in patter classification, function approximation and density estimation. Different from the conventional artificial neural networks (ANNs) which aim at reducing empirical risk, SVM is based on structural risk minimization (SRM) principle [2] which minimizes the upper bound of the generalization error which is bounded by both empirical risk and a confidence interval term. Thus, SRM principle trades off the training error and the complexity of the solution, leading to good performance for the new coming samples. Within a few years after its introduction, SVM has been applied successfully in many real-world fields including face detection [3], text categorization [4], drug discovery [5] and time series prediction [6].

Though SVM owns excellent performance, its training cost is expensive since it needs to solve a quadratic programming problem (QPP) with computational complexity $O(N^3)$, where N is the total size of training samples. So far, many fast training algorithms have been developed to accelerate the training of SVM. On the whole, they can be categorized into two kinds. One is to develop fast training algorithm for the standard SVM model. These kinds of algorithms includes Chunking [7, 8], sequential minimal optimization (SMO) [8], SVMLight, SVMTorch and LibSVM. The central idea of these algorithms is to decompose a larger scale QPP into a series of smaller scale QPPs which can be solved efficiently. On the other hand, in contrast with these decomposition strategy based methods, many variants of standard SVM model were proposed. By replacing hinge loss with squared loss, Suykens and Vandewalle [9] proposed the least squares SVM (LSSVM). LSSVM owns fast training speed since it only needs to solve a system of linear equations instead of QPP as in SVM. Recently, Jayadeva et al [10] proposed the twin support vector machine (TSVM) for binary classification which is in the spirit of GEPSVM [11]. It seeks two nonparallel planes by solving two smaller and related SVM-type problems, in which each hyper-plane is closer to its own class and is as far as possible from the other one. Empirical results show that TSVM is as competitive as SVM in terms of generalization performance whereas the former is around four times faster than the latter [10]. Some

Received October 3, 2012; Revised November 30, 2012; Accepted December 6, 2012

extensions to TSVM include least squares TSVM (LSTSVM) [12], projection TSVM (PTSVM) [13], v-TSVM [14].

As for support vector regression (SVR), there also exist some fast algorithms for learning the optimal regressor, such as SMO for SVM regression [15], Least squares SVR (LSSVR) [9], etc. Recently, a novel regressor called twin support vector regression (TSVR) was proposed [24]. TSVR extends the idea of TSVM to do regression instead of classification, which aims at generating two nonparallel functions such that each function determines the ε-insensitive down- or up-bounds of the unknown functions. The two bounds actually form two one-sided ε-insensitive tubes to respectively contain the training samples. Furthermore, the two bound functions can be obtained by solving two smaller sized QPPs rather than a larger one as in the classic SVR resulting that TSVR owns fast training speed. TSVR has become one of the popular regression methods due to its low computational cost and competitive performance. Some extensions of TSVR include the smooth TSVR [16], robust TSVR [17] and the primal TSVR [18].

Although the experimental results in [24] have shown that the TSVR compares favorably with the SVR and LSSVR, it still has some shortcomings. First, according to the formulation of TSVR, it merely minimizes the empirical risk on the training samples, which is contrast to the SRM principle. This may lead to the over-fitting to the training samples and degrade its generalization performance. Second, we will show in Section 3.1 that the predefined εs, i.e. the size of one-sided ε-insensitive tube, may virtually never affect the final regressor provided they have the same value. This problem makes TSVR depart from the inherent meaning of ε-insensitive loss model in which the deduced ε-insensitive tube is used to tolerate noises and benefits to improve generalization performance.

2. Research Method

Consider a regression problem with the training set $D=\{(x_i, y_i)\,|\, i=1,2,\cdots,m\}$, where $x_i \in R^n$ and $y_i \in R$. We also organize the samples x_i in the matrix A such that the i th row of A denotes the i th training sample x_i.

2.1 Support vector regression

As for linear v-SVR, we would like to find a linear regression function

$$f(x)=\omega^T x+b \tag{1}$$

that has at most ε deviation from the actually obtained targets y_i for all the training samples, and at the same time is as flat as possible. This can be done by minimizing the following objective function

$$\min_{\omega,b,\xi,\xi^*} \quad \frac{1}{2}\omega^T\omega + Ce^T(\xi+\xi^*) \tag{2}$$

$$\text{s.t.} \quad Y-(A\omega+b) \le e\varepsilon+\xi, \xi \ge 0$$

$$(A\omega+b)-Y \le e\varepsilon+\xi^*, \xi^* \ge 0$$

where C is a penalty parameter used to adjust the trade-off between the regression error and the regularization on f, ε is a predefined parameter to reflect the noise variance, ξ and ξ^* are slack vectors, e is the vector of ones of appropriate dimensions. The constraints of (2) make as possible as more samples locate into the ε-insensitive tube of $f(x)$, i.e.

$[f(x)-\varepsilon, f(x)+\varepsilon]$. Only those samples x_i outside the $[f(x_i)-\varepsilon, f(x_i)+\varepsilon]$ have nonzero ξ_i or ξ_i^*. An intuitive geometric illustration for SVR is shown in Figure 1.

For the nonlinear regression problem, we first map a sample x into a high dimensional feature space via the feature map $\varphi: R^n \rightarrow H$ and then construct the linear regression function in as

$$f(x)=\omega^T \varphi(x)+b \tag{3}$$

In general, *H* may have a very high dimension which prevents the direct computation. However, under the Mercer theorem [20, 21], it is possible to use some kernel $k(u^T,v)$ to express the inner product in *H*, i.e. $k(u^T,v)=\varphi(u)^T\varphi(u)$. Similar to (2), the regressor in *H* is determined by

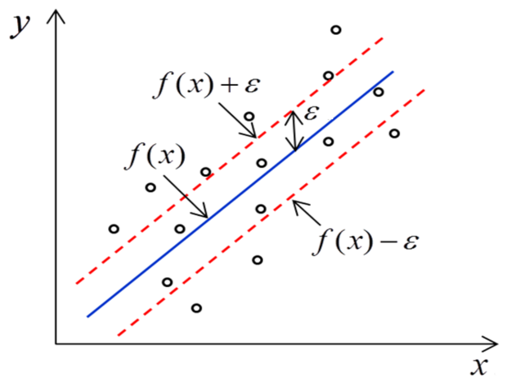


Figure 1. Geometrical interpretation of classic SVR

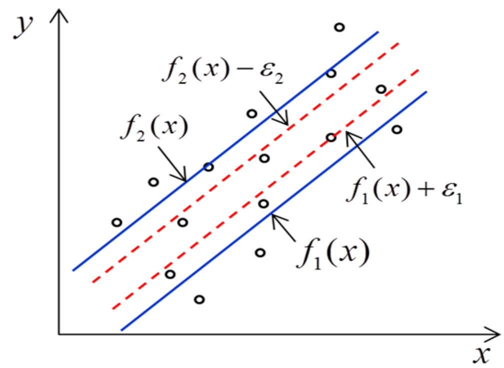


Figure 2. Geometrical interpretation of TSVR

$$\min_{\omega,b,\xi,\xi^*} \quad \frac{1}{2}\omega^T\omega + Ce^T(\xi+\xi^*) \tag{4}$$

$$\text{s.t.} \quad Y-(\varphi(A)\omega+eb)\leq e\varepsilon+\xi, \xi\geq 0$$

$$(\varphi(A)\omega+eb)-Y\leq e\varepsilon+\xi^*, \xi^*\geq 0$$

where $\varphi(A)^T=[\varphi(x_1),\varphi(x_2),\cdots,\varphi(x_m)]$. In practice, we solve the dual optimization problem of (2) and (4) to obtain the corresponding linear and nonlinear regressors. For instance, the dual problem of (4) is given by

$$\max_{\alpha,\alpha^*} \quad -\frac{1}{2}(\alpha-\alpha^*)^T K(A,A^T)(\alpha-\alpha^*)-\varepsilon e^T(\alpha+\alpha^*)+Y^T(\alpha-\alpha^*) \tag{5}$$

$$\text{s.t.} \quad e^T(\alpha+\alpha^*)=0$$

$$0\leq\alpha\leq Ce, 0\leq\alpha^*\leq Ce$$

where α and α^* are the Lagrangian multiplier vectors in R^m, $K(A, A^T)$ is a Gram matrix, called kernel matrix in that its entries are exactly $k(x_i^T, x_j)$. Notice that we must solve (5) which is more computational complex than support vector classification (SVC) because there are $2m$ variables to be optimized in SVR rather than m variables in SVC. After solving (5) to obtain α and α^*, we can predict the output of a new coming data by

$$f(x) = K(x^T, A^T)(\alpha - \alpha^*) + b \tag{6}$$

2.2 Twin support vector regression

Different from the classic SVR, TSVR seeks a pair of nonparallel functions such that each one determines the ε-insensitive down- and up-bounds of the unknown regression function. Geometrically, the concept of TSVR is depicted in Figure 2.

For the linear case, TSVR construct the following two nonparallel ε-insensitive down- and up-bound functions $f_1(x)$ and $f_2(x)$ in the input space

$$f_1(x) = \omega_1^T x + b_1 \tag{6}$$

and

$$f_2(x) = \omega_2^T x + b_2 \tag{7}$$

Then TSVR determines $f_1(x)$ and $f_2(x)$ by solving the following pair of QPPs:

(TSVR1) $$\min_{\omega_1, b_1, \xi} \frac{1}{2} \| Y - e\varepsilon_1 - (A\omega_1 + eb_1) \|^2 + C_1 e^T \xi \tag{8}$$

$$\text{s.t.} \quad Y - (A\omega_1 + eb_1) + \xi \geq e\varepsilon_1, \xi \geq 0$$

And

(TSVR2) $$\min_{\omega_2, b_2, \eta} \frac{1}{2} \| Y + e\varepsilon_2 - (A\omega_2 + eb_2) \|^2 + C_2 e^T \eta \tag{9}$$

$$\text{s.t.} \quad (A\omega_2 + eb_2) - Y + \eta \geq e\varepsilon_2, \eta \geq 0$$

Where C_1, C_2 are the balance parameters, ξ and η are the slack vectors corresponding to $f_1(x)$ and $f_2(x)$, respectively. By analyzing the constraints of (8), we may find that they make as possible as more samples locate in the one-sided ε_1-insensitive tube of $f_1(x)$, i.e. $[f_1(x) + \varepsilon_1, +\infty)$. Only those samples x_i outside the $[f_1(x_i) + \varepsilon_1, +\infty)$ have nonzero ξ_i. A similar conclusion also holds for the constraints of (9). For the sake of simplicity of expression, let us introduce the following notations

$$G = [A \;\; e],\; f_1 = Y - e\varepsilon_1,\; u_1^T = [\omega_1^T \;\; b_1],\; f_2 = Y + e\varepsilon_2,\; u_2^T = [\omega_2^T \;\; b_2] \tag{10}$$

Then, the dual problems of (8) and (9) are given by

$$\max_{\alpha} \quad -\frac{1}{2}\alpha^T G(G^T G)^{-1} G^T \alpha + f_1^T G(G^T G)^{-1} G^T \alpha - f_1^T \alpha \tag{11}$$

$$\text{s.t.} \quad 0 \le \alpha \le C_1 e$$

and

$$\max_{\gamma} \quad -\frac{1}{2}\gamma^T G(G^T G)^{-1} G^T \gamma - f_2^T G(G^T G)^{-1} G^T \gamma + f_2^T \gamma \tag{12}$$

$$\text{s.t.} \quad 0 \le \gamma \le C_2 e$$

where α and γ are the Lagrangian multiplier vectors in R^m for the optimization problem (12) and (13) respectively. Notice that $G^T G$ is always positive semi-definite. However, when it is not well-conditioned or close to singular, we should replace $G^T G$ with $G^T G + \delta I$ where δ is a small positive number and I is the unit matrix with appropriate dimension. After solving (11) and (12), we obtain the augmented vectors for $f_1(x)$ and $f_2(x)$, which are

$$\begin{bmatrix} \omega_1 \\ b_1 \end{bmatrix} = (G^T G)^{-1} G^T (f_1 - \alpha) \tag{13}$$

$$\begin{bmatrix} \omega_2 \\ b_2 \end{bmatrix} = (G^T G)^{-1} G^T (f_2 + \gamma) \tag{14}$$

Then the estimated regressor is the average of its ε-insensitive down- and up-bounds as

$$f(x) = \frac{1}{2}(f_1(x) + f_2(x)) = \frac{1}{2}(x^T(\omega_1 + \omega_2) + b_1 + b_2) \tag{15}$$

3. Experimental results and analysis

3.1 Configuration

To demonstrate the performance of the proposed v-TSVR, we compare it with TSVR, SVR and LSSVR on several synthetic and real-world benchmark datasets. All the regression methods are implemented in MATLAB (7.8.0) R2009a environment on a PC with Intel Core 2 Duo processor (2.4GHz), 3GB RAM. For v-TSVR, TSVR and SVR, we need to resolve QPP to obtain the optimal solution. Thus Mosek optimization toolbox [23] for MATLAB which implements fast interior point based algorithms for convex optimization problems is utilized. In all our experiments, Gaussian kernel $k(x_i, x_j) = \exp(-\| x_i - x_j \|^2 / \gamma)$ is used to construct nonlinear regressor. The optimal model parameters for each algorithm were tuned using the standard ten-fold cross-validation method and a grid search mechanism. To reduce the computational complexity of model selection, we tune a set comprising of 10 percent of the training set to select optimal parameters as in [24] except for the Motorcycle datasets. For TSVR, we let $\varepsilon_1 = \varepsilon_2 = 0$ according to the proposed two-step view of TSVR. Furthermore, we let , $C_1 = C_2 = C$, $\nu_1 = \nu_2 = \nu$ and $\delta_1 = \delta_2 = \delta$. Once the parameters are determined, the tuning set is returned to the whole training set to learn the final regressor. Moreover, two standard errors, root mean squared error (RMSE) and mean absolute error (MAE) of the 10-fold cross validation are used as derivation measurement between the real and the predicted values. They are defined as follows, respectively

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2} \tag{16}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}| y_i - f(x_i) | \tag{17}$$

3.2 Evaluation on synthetic datasets

We first evaluate the performance of v-TSVR on the regression of the following two functions [24], which are defined as

(TypeA) $$f(x) = \frac{\sin(x)}{x}, x \in [-3\pi, 3\pi] \tag{18}$$

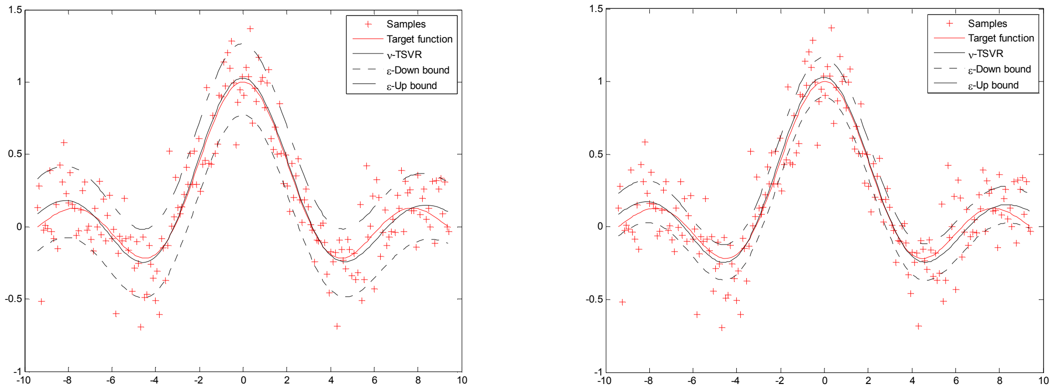
(TypeB) $$f(x) = \left|\frac{x-1}{4}\right| + \left|\sin(\pi(1+\frac{x-1}{4}))\right| + 1, x \in [-10, 10] \tag{19}$$

We generate 200 samples (x_i, y_i) by using $y_i = f(x_i) + \xi_i$ where the noise ξ_i is draw from a Gaussian distribution with zero mean and variance σ^2, $f(x_i)$ is defined by (18) and (19).

Figure 3 (a) and (b) respectively illustrates the results of the proposed v-TSVR regression algorithm on TypeA dataset with noise variance $\sigma = 0.2$ and ninety percent of the dataset are used as training set while the remainder as testing set. We test v-TSVR algorithm with $\nu_1 = 0.1$ and $\nu_1 = 0.3$. The larger ν_1 (ν_2) allows more samples to lie below (above) ε_1 (ε_2)-insensitive tube. As indicated by Figure 4, the width of insensitive tube decreases as ν_1 is increased. Therefore, v-TSVR algorithm automatically adjusts the two one-sided insensitive tubes to include the data. The RMSE on testing set are 0.2241 and 0.2221 for $\nu_1 = 0.1$ and $\nu_1 = 0.3$, respectively.

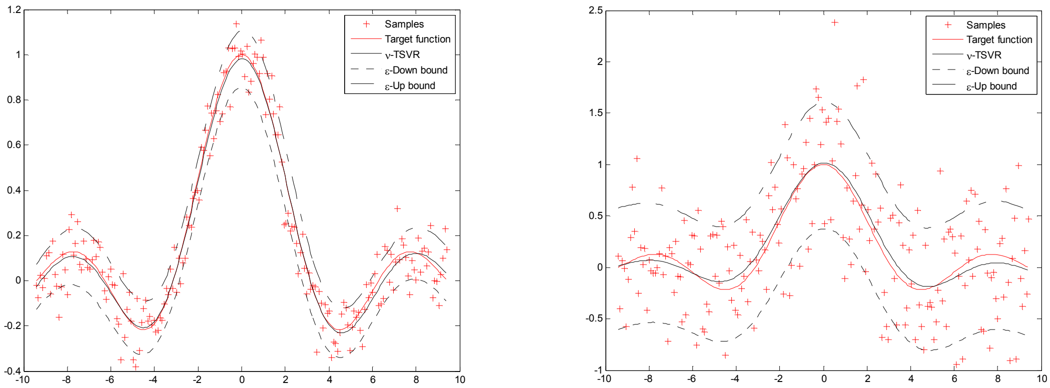
the noisy TypeA data with different noise variance $\sigma = 0.1$ and $\sigma = 0.5$. For v-TSVR, we let $\nu_1 = \nu_2 = 0.1$ for both cases. As seen from Fig.5, the width of insensitive tube increases automatically as the noise variance σ is increased. Therefore, v-TSVR can adjust the insensitive tubes to tolerate different noise distribution. The RMSE on testing set are 0.1105 and 0.5322 for $\sigma = 0.1$ and $\sigma = 0.5$ respectively.

Figure 4(a) and (b) respectively illustrate the results of v-TSVR regression algorithm on We further conduct ten independent run on TypeA and TypeB datasets with σ=0.2 and report the average results to depress the bias induced by randomness. Table 1 reports the average results of TSVR, LSSVR, SVR and the proposed v-TSVR. Figure 5 shows the one-run simulation results for these four algorithms From Table 1, we can see that the proposed v-TSVR owns better generalization than the original TSVR and the other two algorithms. We also show the training time for these methods. We can observe that LSSVR is the fastest one among these methods since it only needs to solve a system of linear equations whereas the other three methods need to solve QPP. Moreover, we find the proposed v-TSVR is as comparable as TSVR in terms of training speed and both owns faster speed than the classic SVR.



(a) (b)

Figure 3. The proposed v-TSVR regression with (a) v1=0.1 (b) v1=0.3



(a) (b)

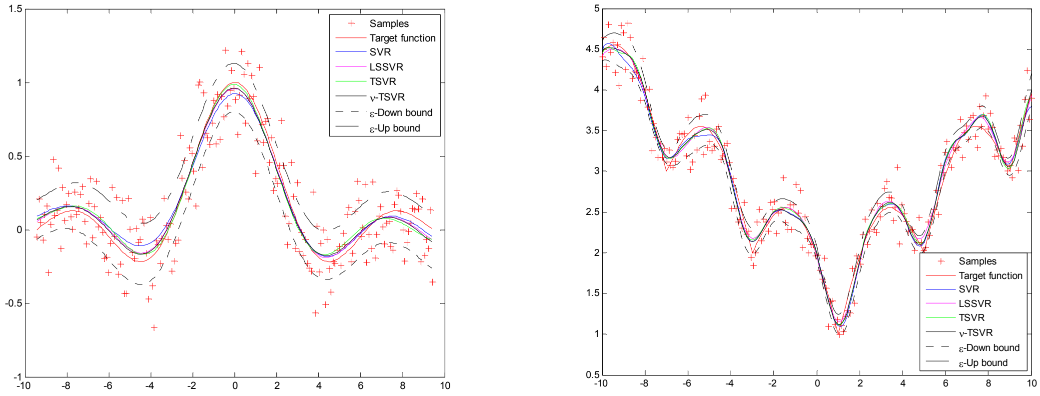
Figure 4. The proposed v-TSVR regression with (a) σ=0.1 (b) σ=0.5

Table 1. Comparisons among v-TSVR, TSVR, LSSVR and SVR on TypeA and TypeB datasets

Data set	v-TSVR RMSE±STD MAE±STD CPU time (s)	TSVR RMSE±STD MAE±STD CPU time (s)	LSSVR RMSE±STD MAE±STD CPU time (s)	SVR RMSE±STD MAE±STD CPU time (s)
TypeA	0.2011±0.0254 0.1641±0.0208 0.0755	0.2052±0.0236 0.1665±0.0209 0.0672	0.2135±0.0184 0.1743±0.0165 0.0058	0.2092±0.0264 0.1654±0.0201 0.1394
TypeB	0.2230±0.0341 0.1884±0.0311 0.0765	0.2268±0.0349 0.1910±0.0282 0.0607	0.2507±0.0570 0.2139±0.0508 0.0050	0.2489±0.0528 0.2087±0.0489 0.1170

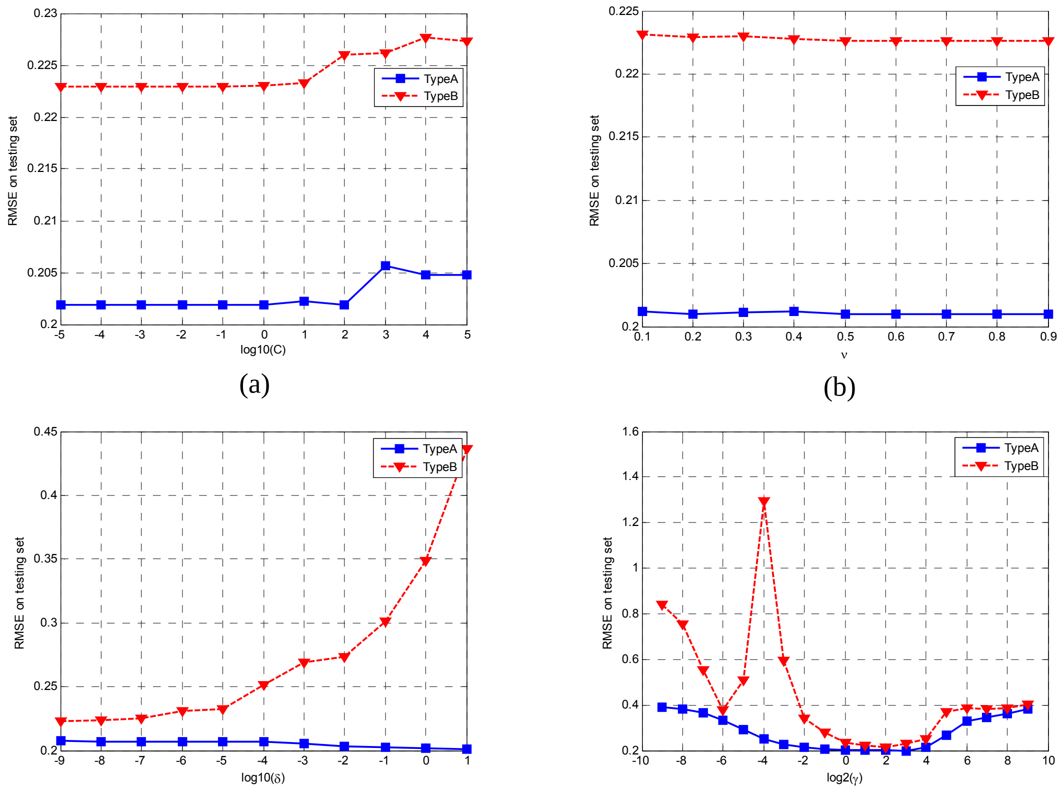
To train the proposed v-TSVR regressor, the parameters (C, ν, δ ,kernel width γ) need to be set to their optimal values beforehand. Thus, the choice of these parameters is very important to build a good regressor. Next, we perform various experiments to show the influence of these parameters on the generalization ability of the proposed v-TSVR. Figure 6(a)~(d) illustrate the results (RMSEs) on the TypeA and TypeB datasets with different settings of the following parameters: C, v, δ and σ. From Figure 7. (a) and (b) , we observe that over a large range of C and v, the testing error in v-TSVR is insensitive toward changes in C and v.

On the other hand, we find v-TSVR is sensitive to both complexity constant δ and kernel width σ. It implies that the introduction of δ is significant to improve generalization.



(a) (b)

Figure. 5 Predictions of v-TSVR, TSVR, LSSVR and SVR on (a) TypeA (b) TypeB datasets



(a) (b)

(c) (d)

Figure 6. The testing error of v-TSVR for different values of (a) balance constant C (b) error constant ν (c) penalty constant δ (d) kernel width γ

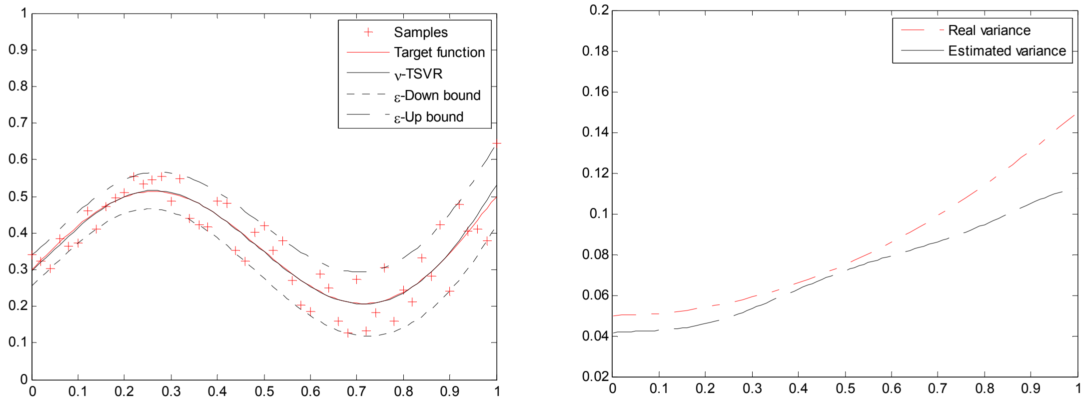
To investigate the advantages of learning two nonparallel one-sided insensitive tubes, we apply v-TSVR on dataset with a heteroscedastic error structure, i.e., the noise strongly depends on the input value x. Specifically, we generate training samples (x_i, y_i) by using $y_i = f(x_i) + \xi_i$ where

$$f(x_i) = 0.2\sin(2\pi x_i) + 0.2x_i^2 + 0.3 \quad (20)$$

$$\xi_i = (0.1x_i^2 + 0.05)\tilde{\xi}_i, \quad \tilde{\xi}_i \sim \mathrm{U}[-1,1]$$

$$x_i = 0.02(i-1), \quad i = 1, 2, \cdots, 51$$

where $\mathrm{U}[a,b]$ means a uniform distribution over the interval spanned by a and b. This example was also used in previous research [25, 26]. Figure 7(a) show the true function and the estimated function by v-TSVR. Figure 8(b) show the true variance function and the estimated variance function by v-TSVR which is given by $A = 1/2(f_2(x) - f_1(x) - \varepsilon 2 + \varepsilon 1)$ Without requiring prior knowledge about the heteroscedastic structure of noise, the proposed v-TSVR automatically adjusts the two one-sided insensitive tubes with minimal size to include the data.



(a) (b)

Figure 7. v-TSVR on the heteroscedastic data (a) the true and estimated function (b) the true and estimated variance

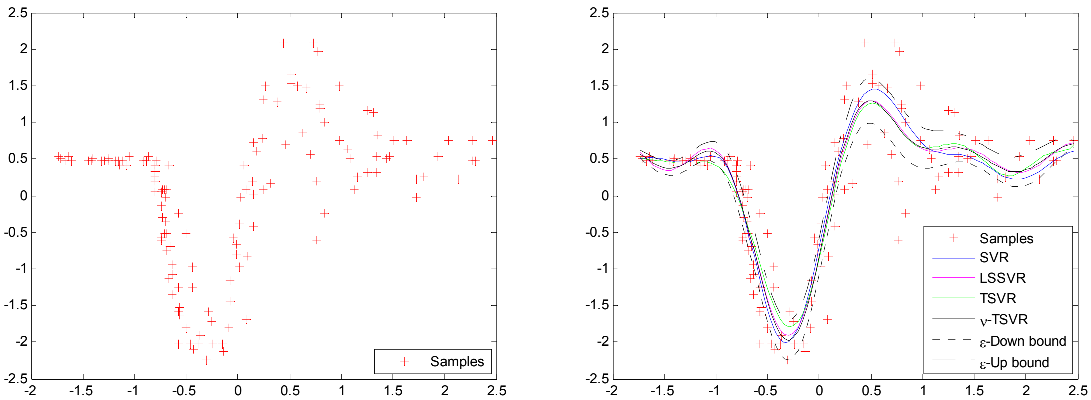


Figure 8. Motorcycle dataset and regression results of v-TSVR, TSVR, LSSVR and SVR

As seen from Figure 8, by learning the ε-insensitive down- and up-bounds respectively, v-TSVR captures the characteristics of the heteroscedastic error structure. The RMSE for this heteroscedastic data is 0.0484. Therefore, v-TSVR is also suitable for regression under heteroscedastic noise.

Finally, we further synthesize eight artificial datasets, tabulated in Table 2, to evaluate the performance of v-TSVR. All these datasets have been used in the literatures, i.e. [16, 27], to explore the performance of regressors. The average results of ten groups test are reported in Table 3. As seen from Table 3, the proposed v-TSVR owns superior performance to the other methods in most cases. The CPU time on these datasets indicates v-TSVR is also an efficient methods compared with TSVR.

Table 2. The synthetic datasets

Data set	Function	Attribute
Multi	$f(x) = 0.79 + 1.27x_1x_2 + 1.56x_1x_4 + 3.42x_2x_5 + 2.06x_3x_4x_5$	$x_1, x_2, x_3, x_4, x_5 \sim \mathrm{U}[0,1]$
Plane	$f(x) = 0.6x_1 + 0.3x_2$	$x_1, x_2 \sim \mathrm{U}[0,1]$
Polynomial	$f(x) = 1 + 2x + 3x^2 + 4x^3 + 5x^4$	$x \sim \mathrm{U}[0,1]$
Friedman #1	$f(x) = 10\sin(\pi x_1x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5$	$x_1, x_2, x_3, x_4, x_5 \sim \mathrm{U}[0,1]$
Friedman #2	$f(x) = \sqrt{x_1^2 + (x_2x_3 - \frac{1}{x_2x_4})^2}$	$x_1 \sim \mathrm{U}[0,100], x_2 \sim \mathrm{U}[40\pi, 560\pi]$ $x_3 \sim \mathrm{U}[0,1], x_4 \sim \mathrm{U}[1,11]$
Friedman #3	$f(x) = \tan^{-1}\frac{x_2x_3 - \frac{1}{x_2x_4}}{x_1}$	$x_1 \sim \mathrm{U}[0,100], x_2 \sim \mathrm{U}[40\pi, 560\pi]$ $x_3 \sim \mathrm{U}[0,1], x_4 \sim \mathrm{U}[1,11]$
2-d Mexican Hat	$f(x) = \frac{\sin\|x\|}{\|x\|}$	$x \sim \mathrm{U}[-2\pi \quad 2\pi]$
3-d Mexican Hat	$f(x) = \frac{\sin\sqrt{x_1^2 + x_2^2}}{\sqrt{x_1^2 + x_2^2}}$	$x_1, x_2 \sim \mathrm{U}[-4\pi, 4\pi]$

Table 3. Comparisons among v-TSVR, TSVR, LSSVR and SVR on synthetic datasets

Data set	v-TSVR RMSE±STD MAE±STD CPU time (s)	TSVR RMSE±STD MAE±STD CPU time (s)	LSSVR RMSE±STD MAE±STD CPU time (s)	SVR RMSE±STD MAE±STD CPU time (s)
Multi	0.1769±0.0231 0.1414±0.0219 0.0680	0.1822±0.0254 0.1460±0.0239 0.0719	0.1774±0.0168 0.1432±0.0188 0.0049	0.1952±0.0367 0.1585±0.0364 0.1343
Plane	0.6555±0.1629 0.5252±0.1322 0.0555	0.6672±0.1747 0.5292±0.1406 0.0724	0.6720±0.1478 0.5329±0.1110 0.0045	0.6614±0.1338 0.5250±0.1008 0.1141
Polynomial	0.0515±0.0067 0.0423±0.0056 0.0734	0.0519±0.0060 0.0426±0.0048 0.0695	0.0520±0.0071 0.0428±0.0060 0.0043	0.0526±0.0059 0.0435±0.0055 0.1294
Friedman #1	0.1853±0.0902 0.1372±0.0663 0.0652	0.1296±0.0615 0.0910±0.0354 0.0776	0.1368±0.0539 0.1006±0.0361 0.0052	0.1600±0.0979 0.1154±0.0651 0.1273
Friedman #2	0.0101±0.0029 0.0073±0.0019 0.0464	0.0100±0.0029 0.0073±0.0019 0.0660	0.0099±0.0032 0.0069±0.0020 0.0047	0.0098±0.0057 0.0059±0.0026 0.1444
Friedman #3	0.6428±0.1110 0.4966±0.0579 0.0707	0.6746±0.0994 0.5096±0.0345 0.0784	0.6523±0.1133 0.5012±0.0584 0.0090	0.6581±0.1297 0.5120±0.0615 0.1454
2-d Mexican Hat	0.4351±0.0834 0.3513±0.0693 0.0740	0.4412±0.0895 0.3534±0.0709 0.0757	0.4386±0.0892 0.3509±0.0675 0.0055	0.4444±0.0858 0.3629±0.0687 0.1293
3-d Mexican Hat	1.0166±0.1730 0.8167±0.1253 0.0703	1.0510±0.1357 0.8445±0.1098 0.0705	1.3547±0.6480 1.0660±0.4369 0.0048	1.0384±0.1597 0.8347±0.1188 0.1265

3.3 Evaluation on benchmark datasets

To further test the performance of these algorithms, we run them on several publicly available benchmark datasets, including Boston Housing, AutoMPG, Servo, MachineCPU, Triazine, AutoPrice, Breast Cancer, ConcreteCS [29] and Motorcycle [30]. These datasets are commonly used in testing regression algorithms. Notice that all samples are normalized such that they have zero mean and unity variance before learning.

Table 4 presents the average results of four algorithms with ten-fold cross-validation on these benchmark datasets. Figure 9 shows the motorcycle dataset and regression results of the four algorithms. As seen from Table 4, v-TSVR derives better performance in most cases comparing with the other three methods. As for the training time, LSSVR still spends on the least CPU time among these algorithms, while our v-TSVR is as comparable as the original TSVR and both are faster than SVR. In a word, the experimental results indicate v-TSVR is an effective algorithm for real-world regression problems.

Table 4. Comparisons among v-TSVR, TSVR, LSSVR and SVR on benchmark datasets

Data set	v-TSVR RMSE±STD MAE±STD CPU time (s)	TSVR RMSE±STD MAE±STD CPU time (s)	LSSVR RMSE±STD MAE±STD CPU time (s)	SVR RMSE±STD MAE±STD CPU time (s)
Boston Housing 506×14	0.3186±0.0764 0.2332±0.0433 0.5537	0.3318±0.0774 0.2431±0.0443 0.6048	0.3226±0.0914 0.2260±0.0445 0.0401	0.3496±0.0923 0.2250±0.0385 1.1566
AutoMPG 398×8	0.3583±0.0626 0.2603±0.0419 0.3061	0.3698±0.0743 0.2722±0.0520 0.2911	0.3709±0.0795 0.2693±0.0528 0.0229	0.3947±0.1483 0.2857±0.1002 0.6282
Servo 164×13	0.6826±0.1339 0.5416±0.0930 0.0501	0.6184±0.1522 0.4891±0.1059 0.0554	0.6847±0.1702 0.5278±0.1234 0.0047	0.6320±0.2055 0.4425±0.1758 0.0848
Machine CPU 209×8	0.0596±0.0477 0.0255±0.0146 0.0616	0.0584±0.0325 0.0271±0.0116 0.0675	0.0502±0.0332 0.0227±0.0111 0.0043	0.0614±0.0564 0.0231±0.0148 0.1537
Triazine 186×61	0.8868±0.1146 0.6669±0.0811 0.0565	0.8885±0.1182 0.6986±0.0796 0.0687	0.9236±0.1293 0.6809±0.0933 0.0050	0.9197±0.1626 0.6606±0.1134 0.0996
AutoPrice 159×16	0.4246±0.1010 0.2929±0.0717 0.0420	0.4314±0.0867 0.2868±0.0546 0.0402	0.4702±0.2896 0.3029±0.1348 0.0028	0.4402±0.1572 0.3017±0.1219 0.0709
Breast Cancer 683×11	0.3303±0.0659 0.1718±0.0265 1.1288	0.3451±0.0462 0.1937±0.0264 1.0457	0.3431±0.0854 0.1706±0.0321 0.0729	0.3379±0.0629 0.1991±0.0431 2.7749
ConcreteCS 1030×9	0.3835±0.0868 0.2630±0.0619 3.0028	0.3948±0.0819 0.2634±0.0632 3.0171	0.3746±0.0592 0.2602±0.0406 0.1852	0.3932±0.0618 0.2820±0.0378 7.4709
Motorcycle 133×2	0.4649±0.0890 0.3537±0.0596 0.0457	0.4659±0.0847 0.3570±0.0641 0.0469	0.4671±0.0884 0.3594±0.0607 0.0076	0.4662±0.0936 0.3396±0.0609 0.0620

4. Conclusion

In this paper, we improve the recently proposed TSVR to a novel regression algorithm, coined v-TSVR. We first give a two-step view for the original TSVR to show that the sizes of one-sided tubes in TSVR are independent on its optimization formulations which may degrade the performance of TSVR. To this end, we propose a new pair of QPPs which can directly learn two nonparallel one-sided tubes by introducing the new parameters vs. Similar to the v-SVR, vs have greater theoretical interpretation, such as the determination of the bounds for the fractions of pSVs and margin errors. Moreover, v-TSVR automatically adjusts the one-sided tubes with minimal radius to include the given data. The experimental results on both synthetic and benchmark datasets have indicated that v-TSVR is an efficient and effective method for regression.

Acknowledgment

The authors would like to thank the anonymous reviewers for their critical and constructive comments and suggestions. The authors are also thankful to the National Natural Science Foundation of China (Grant No. 51108209, No. 50875112 and No. 70972048), the Natural Science Foundation of Jiangsu Province(Grant No.BK2010339) , the Natural Science Foundation of the Jiangsu Higher Education Institutions of China(Grant No.10KJD580001) and the College Graduate Research and Innovation Program of Jiangsu Province (Dr.

Innovation)(Grant No.CXLX11_0593),A Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions(PAPD).

References

[1] C Cortes, V Vapnik. Support-vector networks. *Machine learning*. 1995; 20: 273-297.
[2] VN Vapnik. *Statistical learning theory*. Wiley, New York, 1998.
[3] E Osuna, R Freund, F Girosi. Training support vector machines: an application to face detection, in Computer Vision and Pattern Recognition. Published by the *IEEE Computer Society*. 1997; 130-136.
[4] T Joachims. Text categorization with support vector machines: Learning with many relevant features, Machine Learning: *ECML-98*. (1998) 137-142.
[5] A Demiriz, K Bennett, C Breneman, M Embrechts. Support vector machine regression in chemometrics. in Computing Science and Statistics*: Proceedings of the 33rd Symposium on the Interface*. 2001.
[6] L Cao. Support vector machines experts for time series forecasting. *Neurocomputing*. 2003; 51: 321-339.
[7] E Osuna, R Freund, F Girosi. An improved training algorithm for support vector machines, in Neural Networks for Signal Processing. Proceedings of *the 1997 IEEE Workshop*. pp. 276-285.
[8] J Platt. "Fast training of support vector machines using sequential minimal optimization" in B. Schölkopf, C. Burges and A. Smola, editors, Advances in Kernel Methods – *Support Vector Learning*, pp. 185–208. MIT Press, 1999.
[9] J Suykens, J Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*. 9 (1999) 293-300.
[10] Jayadeva, R Khemchandani, S Chandra. Twin Support Vector Machines for pattern classification. *IEEE Trans Pattern Anal Mach Intell*. 29 (2007) 905-910.
[11] OL Mangasarian, EW Wild. Multisurface proximal support vector machine classification via generalized eigenvalues. *IEEE Trans Pattern Anal Mach Intell*. 28; (2006): 69-74.
[12] M Arun Kumar, M Gopal. Least squares twin support vector machines for pattern classification. *Expert Systems with Applications*. 36; (2009): 7535-7543.
[13] X Chen, J Yang, Q Ye, J Liang. Recursive projection twin support vector machine via within-class variance minimization. *Pattern Recognition*, (2011), 44(10-11): 2643-2655.
[14] X Peng. A v-twin support vector machine (v-TSVM) classifier and its geometric algorithms, Information Sciences: an International Journal, 180 (2010) 3863-3875.
[15] S Shevade, S Keerthi, C Bhattacharyya, K Murthy. Improvements to the SMO algorithm for SVM regression. *Neural Networks, IEEE Transactions on*, 11 (2002) 1188-1193.
[16] X Chen, J Yang, J Liang, Q Ye. Smooth twin support vector regression. Neural Computing & Applications. (2010) doi:10.1007/s00521-010-0454-9.
[17] X Chen, J Yang, J Liang, Q Ye. Robust and Sparse Twin Support Vector Regression via Linear Programming. in Chinese Conference on Pattern Recognition (2010). *IEEE*. pp. 1-6.
[18] X Peng. Primal twin support vector regression and its sparse approximation. *Neurocomputing*. 2010.
[19] B Schölkopf, A Smola, R Williamson, P Bartlett. New support vector algorithms. *Neural Computation*. 12: (2000); 1207-1245.
[20] J Mercer. Functions of positive and negative type, and their connection with the theory of integral equations, Philosophical Transactions of the Royal Society of London. *Series A, Containing Papers of a Mathematical or Physical Character*. 209; (1909): 415-446.
[21] B Schölkopf, AJ Smola. Learning with kernels : support vector machines, regularization, optimization, and beyond. *MIT Press, Cambridge, Mass*., 2002.
[22] Y Lee, O Mangasarian. RSVM: Reduced support vector machines, in International Conference on Data Mining, 2001.
[23] A MOSEK. The MOSEK optimization software, Online at http://www. mosek. com, (2010).
[24] X Peng. TSVR: an efficient Twin Support Vector Machine for regression. *Neural Netw*. 23; (2010): 365-372.
[25] C Hwang, D Hong, K Ha Seok. Support vector interval regression machine for crisp input and output data. *Fuzzy Sets and Systems*. 157; (2006): 1114-1125.
[26] P Hao. New support vector algorithms with parametric insensitive/margin model. *Neural Networks*. 23; (2010): 60-73.
[27] Z Zhou, M Li. Semisupervised regression with cotraining-style algorithms. *IEEE Transactions on Knowledge and Data Engineering*. (2007): 1479-1493.
[28] S Boyd, L Vandenberghe. Convex optimization, Cambridge Univ Pr, 2004.
[29] Blake, CI, & Merz CJ (1998). UCI repository for machine learning databases: http://www.ics.uci.edu/~mlearn/MLRepository.html Irvine, CA: University of California, Department of Information and Computer Sciences.
[30] Eubank RL (1999). Statistics: Textbooks and monographs: Vol. 157. Nonparametric regression and spline smoothing (second ed.). New York: Marcel Dekker.