Neural Comput & Applic (2013) 22 (Suppl 1):S257–S267
DOI 10.1007/s00521-012-0971-9

ORIGINAL ARTICLE

On Lagrangian twin support vector regression

S. Balasundaram · M. Tanveer

Received: 28 June 2011 / Accepted: 21 May 2012 / Published online: 21 June 2012

Abstract In this paper, a simple and linearly convergent Lagrangian support vector machine algorithm for the dual of the twin support vector regression (TSVR) is proposed. Though at the outset the algorithm requires inverse of matrices, it has been shown that they would be obtained by performing matrix subtraction of the identity matrix by a scalar multiple of inverse of a positive semi-definite matrix that arises in the original formulation of TSVR. The algorithm can be easily implemented and does not need any optimization packages. To demonstrate its effectiveness, experiments were performed on well-known synthetic and real-world datasets. Similar or better generalization performance of the proposed method in less training time in comparison with the standard and twin support vector regression methods clearly exhibits its suitability and applicability.

Keywords Machine learning · Nonparallel planes · Lagrangian support vector machines · Support vector regression · Twin support vector regression

1 Introduction

Support vector machines (SVMs) developed by Vapnik [24] are a class of kernel-based supervised learning machines for binary classification and regression. They have shown excellent performance on wide variety of problems [4, 9, 19] due to its method of constructing a hyperplane that partitions the inputs from different classes with maximum margin [6, 24]. Unlike other machine learning methods such as artificial neural networks (ANNs), training of SVMs leads to solving a linearly constrained quadratic programming problem (QPP) having unique optimal solution. Combined with the advantage of having unique optimal solution and better generalization performance, SVM becomes one of the most popular methods for solving classification and regression problems.

Although SVM provides better generalization results in comparison to other machine learning approaches, its training cost is expensive, that is $O(m^3)$ where m is the size of the total training samples [8]. To reduce its time complexity, a family of non-parallel planes learning algorithms has been proposed in the literature [8, 10, 15]. In the sprit of the generalized eigenvalue proximal support vector machine (GEPSVM) proposed in [15], Jayadeva et al. [8] introduced twin support vector machine (TWSVM) wherein two non-parallel planes are constructed by solving two related SVM-type problems of smaller size than the standard SVM. For an improved version named twin bounded support vector machines (TBSVM) based on TWSVM, see [22].

Recently, Peng [20] proposed twin support vector regression (TSVR) similar in sprit to TWSVM wherein a pair of non-parallel functions corresponding to the ε-insensitive down- and up- bounds of the unknown regressor is determined. As in TWSVM, this formulation leads to the solution of a pair of dual QPPs of smaller size rather than a single large one as in the standard support vector regression (SVR). This strategy makes TSVR works faster than SVR with the added advantage of better generalization performance over SVR [20]. However, TSVR formulation requires, like TWSVM, the inverse of a

S. Balasundaram (✉) · M. Tanveer
School of Computer and Systems Sciences,
Jawaharlal Nehru University, New Delhi 110067, India
e-mail: balajnu@gmail.com

M. Tanveer
e-mail: tanveergouri@gmail.com

positive semi-definite matrix. By reformulating TSVR to become a pair of strongly convex unconstrained minimization problems in primal and employing smooth technique, smooth twin support vector regression (STSVR) has been proposed in [5]. Finally, for the work on the formulation of TSVR as a pair of linear programming problems, we refer the reader to [25].

Motivated by the work of [1, 2, 13], we propose in this paper Lagrangian twin support vector regression (LTSVR) formulation whose solution will be obtained by applying a simple iterative algorithm that converges for any starting point given. In fact, the Karush–Kuhn–Tucker (KKT) necessary and sufficient optimal conditions for the pair of dual QPPs of TSVR are considered and solved using linearly convergent proposed iterative algorithm. Our formulation has the advantage that the ε-insensitive down- or up- bound regressor will be determined using an extremely simple iterative method rather than solving a QPP as in SVR and TSVR.

In this work, all vectors are taken as column vectors. The inner product of two vectors x, y in the n-dimensional real space R^n will be denoted by: x^ty, where x^t is the transpose of x. Whenever x is orthogonal to y, we write $x \perp y$. For $x = (x_1, \ldots, x_n)^t \in R^n$, the plus function x_+ is defined as: $(x_+)_i = \max\{0, x_i\}$, where $i = 1,\ldots, n$. The 2-norm of a vector x and a matrix Q will be denoted by $||x||$ and $||Q||$, respectively. We denote the vector of ones of dimension m by e and the identity matrix of appropriate size by I.

The paper is organized as follows. In Sect. 2, the standard SVR and TSVR are reviewed. The proposed LTSVR algorithm for solving the dual QPPs is derived in Sect. 3. Numerical experiments have been performed on a number of interesting synthetic and real-world datasets, and the results obtained are compared with that of SVR and TSVR in Sect. 4 and finally, we conclude our work in Sect. 5.

2 Related work

In this section, we briefly describe the standard SVR and TSVR formulations.

Let a set of input samples $\{(x_i, y_i)\}_{i=1,2,\ldots,m}$ be given where for each training example $x_i \in R^n$ its corresponding observed value being $y_i \in R$. Further, let the training examples be represented by a matrix $A \in R^{m\times n}$ whose ith row is defined to be the row vector x_i^t and the vector of observed values be denoted by $y = (y_1, \ldots, y_m)^t$.

2.1 Support vector regression (SVR) formulation

In SVM for regression, the estimation function is obtained by mapping the input examples into a higher dimensional feature space via a non-linear mapping $\varphi(.)$ and learning a linear regressor in the feature space. Assuming that the non-linear regression estimating function $f : R^n \to R$ is taken to be of the form:

$$f(x) = w^t\varphi(x) + b,$$

where w is a vector in the feature space and b is a scalar threshold, the ε-insensitive SVR formulation aims at determining w and b as the solution of the following constrained minimization problem [6, 24]:

$$\min_{w,b,\xi_1,\xi_2} \frac{1}{2}w^tw + C(e^t\xi_1 + e^t\xi_2)$$

subject to

$$y_i - w^t\varphi(x_i) - b \le \varepsilon + \xi_{1i},$$
$$w^t\varphi(x_i) + b - y_i \le \varepsilon + \xi_{2i}$$

and

$$\xi_{1i}, \xi_{2i} \ge 0 \quad \text{for} \quad i = 1, 2, \ldots, m,$$

where $\xi_1 = (\xi_{11}, \ldots, \xi_{1m})^t$, $\xi_2 = (\xi_{21}, \ldots, \xi_{2m})^t$ are vectors of slack variables, and $C > 0$, $\varepsilon > 0$ are input parameters.

In practice, rather than solving the primal problem, we solve its dual problem, which can be written in the following form:

$$\min_{u_1,u_2} \frac{1}{2}\sum_{i,j=1}^{m}(u_{1i} - u_{2i})(u_{1j} - u_{2j})\varphi(x_i)^t\varphi(x_j) + \varepsilon\sum_{i=1}^{m}(u_{1i} + u_{2i}) - \sum_{i=1}^{m} y_i(u_{1i} - u_{2i})$$

subject to

$$\sum_{i=1}^{m}(u_{1i} - u_{2i}) = 0 \quad \text{and} \quad 0 \le u_1, u_2 \le Ce,$$

where $u_1 = (u_{11}, \ldots, u_{1m})^t$ and $u_2 = (u_{21}, \ldots, u_{2m})^t$ in R^m are Lagrange multipliers.

Applying the kernel trick [6, 24], that is taking:

$$k(x_i, x_j) = \varphi(x_i)^t\varphi(x_j),$$

where $k(.,.)$ is a kernel function, the above dual problem can be rewritten and solved. In this case, the decision function $f(.)$ will become [6, 24]: for any input example $x \in R^n$, its prediction is given by

$$f(x) = \sum_{i=1}^{m}(u_{1i} - u_{2i})k(x, x_i) + b.$$

2.2 Twin support vector regression (TSVR) formulation

Motivated by the work of TWSVM [8], a new regressor called twin support vector regression (TSVR) was proposed in [20] wherein two non-parallel functions are determined as estimators for the ε-insensitive down- and up-bounds of the unknown regression function. However, as opposed to solving a single QPP with $2m$ number of constraints where m is the number of input examples, TSVR formulation leads to solving a pair of QPPs each having m number of linear constraints.

Assume that the down- and up-bound regressors of TSVR in the input space are expressed as: for any $x \in R^n$,

$$f_1(x) = w_1^t x + b_1 \quad \text{and} \quad f_2(x) = w_2^t x + b_2 \tag{1}$$

respectively, where $w_1, w_2 \in R^n$ and $b_1, b_2 \in R$ are unknowns. The linear TSVR formulation leads to determining the regressors (1) as the solutions of the following pair of QPPs defined by [20]:

$$\min_{(w_1,b_1,\xi_1)\in R^{n+1+m}} \frac{1}{2}||y - \varepsilon_1 e - (Aw_1 + b_1 e)||^2 + C_1 e^t \xi_1$$

subject to

$$y - (Aw_1 + b_1 e) \geq \varepsilon_1 e - \xi_1,\ \xi_1 \geq 0 \tag{2}$$

and

$$\min_{(w_2,b_2,\xi_2)\in R^{n+1+m}} \frac{1}{2}||y + \varepsilon_2 e - (Aw_2 + b_2 e)||^2 + C_2 e^t \xi_2$$

subject to

$$(Aw_2 + b_2 e) - y \geq \varepsilon_2 e - \xi_2,\ \xi_2 \geq 0 \tag{3}$$

respectively, where $C_1,\ C_2 > 0$; $\varepsilon_1,\ \varepsilon_2 > 0$ are input parameters and ξ_1, ξ_2 are vectors of slack variables in R^m. Now using the down- and up-bound regressors, the end regression function f: $R^n \to R$ is taken as

$$f(x) = \frac{1}{2}(f_1(x) + f_2(x)) \quad \text{for all} \quad x \in R^n. \tag{4}$$

By considering the Lagrangian functions for the problems (2), (3) and using the KKT necessary and sufficient optimal conditions, one can obtain the pair of dual QPPs of TSVR as given below [20]:

$$\min_{u_1\in R^m} \frac{1}{2}u_1^t G(G^t G)^{-1} G^t u_1 - (y - \varepsilon_1 e)^t (G(G^t G)^{-1} G^t - I)u_1$$

subject to

$$0 \leq u_1 \leq C_1 e \tag{5}$$

and

$$\min_{u_2\in R^m} \frac{1}{2}u_2^t G(G^t G)^{-1} G^t u_2 - (y + \varepsilon_2 e)^t (I - G(G^t G)^{-1} G^t)u_2$$

subject to

$$0 \leq u_2 \leq C_2 e \tag{6}$$

respectively, satisfying

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (G^t G)^{-1} G^t (y - \varepsilon_1 e - u_1) \quad \text{and}$$

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (G^t G)^{-1} G^t (y + \varepsilon_2 e + u_2), \tag{7}$$

where u_1, u_2 are Lagrange multipliers and $G = [A \quad e]$ is an augmented matrix.

Following the approach of [13], the TSVR discussed above can be easily extended to kernel TSVR. In fact, for the input matrix $A \in R^{m\times n}$ define the kernel matrix $K = K(A, A^t)$ of order m whose (i, j)th element is given by

$$K(A, A^t)_{ij} = k(x_i, x_j) \in R,$$

where $k(.,.)$ is a non-linear kernel function. Also for a given vector $x \in R^n$, we define

$$K(x^t, A^t) = (k(x, x_1), \ldots, k(x, x_m)),$$

a row vector in R^m. Then, assuming that the down- and up-bound regressors are of the form: for any vector $x \in R^n$,

$$f_1(x) = K(x^t, A^t)w_1 + b_1 \quad \text{and} \quad f_2(x) = K(x^t, A^t)w_2 + b_2 \tag{8}$$

respectively, the non-linear TSVR determines the unknowns $w_1, w_2 \in R^m$ and $b_1, b_2 \in R$ as the solutions of the following pair of minimization problems:

$$\min_{(w_1,b_1,\xi_1)\in R^{m+1+m}} \frac{1}{2}||y - \varepsilon_1 e - (K(A, A^t)w_1 + b_1 e)||^2 + C_1 e^t \xi_1$$

subject to

$$y - (K(A, A^t)w_1 + b_1 e) \geq \varepsilon_1 e - \xi_1, \quad \xi_1 \geq 0$$

and

$$\min_{(w_2,b_2,\xi_2)\in R^{m+1+m}} \frac{1}{2}||y + \varepsilon_2 e - (K(A, A^t)w_2 + b_2 e)||^2 + C_2 e^t \xi_2$$

subject to

$$(K(A, A^t)w_2 + b_2 e) - y \geq \varepsilon_2 e - \xi_2, \quad \xi_2 \geq 0.$$

Now proceeding as in the linear case, the pair of dual QPPs for the kernel TSVR can be obtained. In fact, they are found to be exactly of the same form as (5), (6) satisfying (7) where $u_1, u_2 \in R^m$ are Lagrange multipliers, but the augmented matrix G is defined by: $G = [K(A, A^t) \quad e]$. Finally, the end regressor given by (4) is obtained using (7) and (8).

For a detailed discussion on the problem formulation of TSVR and its method of solution, see [20].

3 Lagrangian twin support vector regression (LTSVR)

Since the matrix $G^t G$ appearing in the dual object functions of (5), (6) is positive semi-definite, it is possible that its inverse may not exist. Assuming that it is invertible, the objective functions of (5), (6) become merely convex, and therefore, more than one optimal solution to the minimization problems may exist. To make the objective functions become strongly convex, following the approach of Mangasarian et al. [13], we consider the square of the 2-norm of the vector of slack variables instead of the usual 1-norm and propose in this section Lagrangian twin support vector regression (LTSVR) formulation whose solution will be obtained by an extremely simple iterative algorithm. The computational results, given in Sect. 4, clearly show that our proposed 2-norm formulation does not compromise on generalization performance.

The linear TSVR in 2-norm determines the down- and up- bound regressors $f_1(.)$ and $f_2(.)$ of the form (1) as the solutions of the pair of QPPs:

$$\min_{(w_1,b_1,\xi_1)\in R^{n+1+m}} \frac{1}{2}||y-\varepsilon_1 e-(Aw_1+b_1e)||^2+\frac{C_1}{2}\xi_1^t\xi_1$$

subject to

$$y-(Aw_1+b_1e)\geq \varepsilon_1 e-\xi_1 \tag{9}$$

and

$$\min_{(w_2,b_2,\xi_2)\in R^{n+1+m}} \frac{1}{2}||y+\varepsilon_2 e-(Aw_2+b_2e)||^2+\frac{C_2}{2}\xi_2^t\xi_2$$

subject to

$$(Aw_2+b_2e)-y\geq \varepsilon_2 e-\xi_2 \tag{10}$$

respectively, where $C_1,\ C_2>0;\ \varepsilon_1,\ \varepsilon_2>0$ are input parameters and $\xi_1,\ \xi_2$ are vectors of slack variables. Note that the non-negativity constraints of the slack variables are dropped in this formulation since they will be satisfied automatically at optimality.

By considering the Lagrangian functions corresponding to (9) and (10) and using the condition that their partial derivatives with respect to the primal variables will be zero at optimality, the dual QPPs of (9) and (10) can be obtained as a pair of minimization problems of the following form:

$$\begin{aligned}\min_{0\leq u_1\in R^m} &\frac{1}{2}(y-\varepsilon_1 e)^t G(G^tG)^{-1}G^t(y-\varepsilon_1 e)-\frac{1}{2}||y-\varepsilon_1 e||^2\\ &+\frac{1}{2}u_1^t\left(\frac{I}{C_1}+G(G^tG)^{-1}G^t\right)u_1\\ &-(y-\varepsilon_1 e)^t(G(G^tG)^{-1}G^t-I)u_1\end{aligned} \tag{11}$$

and

$$\begin{aligned}\min_{0\leq u_2\in R^m} &\frac{1}{2}(y+\varepsilon_2 e)^t G(G^tG)^{-1}G^t(y+\varepsilon_2 e)-\frac{1}{2}||y+\varepsilon_2 e||^2\\ &+\frac{1}{2}u_2^t\left(\frac{I}{C_2}+G(G^tG)^{-1}G^t\right)u_2\\ &-(y+\varepsilon_2 e)^t(I-G(G^tG)^{-1}G^t)u_2,\end{aligned} \tag{12}$$

respectively, where $G=[A\quad e]$ is an augmented matrix and the Lagrange multipliers $u_1,\ u_2\in R^m$ satisfy the conditions: $u_1,u_2\geq 0$ and (7).

Define the matrix

$$H=G(G^tG)^{-1}G^t. \tag{13}$$

Then, dropping the terms that are independent of the dual variables, the above minimization problems (11) and (12) can be rewritten in the following simpler form:

$$\begin{aligned}&\min_{0\leq u_1\in R^m} L_1(u_1)=\frac{1}{2}u_1^tQ_1u_1-r_1^tu_1 \quad \text{and}\\ &\min_{0\leq u_2\in R^m} L_2(u_2)=\frac{1}{2}u_2^tQ_2u_2-r_2^tu_2\end{aligned} \tag{14}$$

respectively, where

$$\begin{aligned}&Q_1=\frac{I}{C_1}+H,\ Q_2=\frac{I}{C_2}+H,\ r_1=(H-I)(y-\varepsilon_1 e) \quad \text{and}\\ &r_2=(I-H)(y+\varepsilon_2 e).\end{aligned} \tag{15}$$

Finally, using the solutions of (14) and equations (1), (7), the end regressor (4) will be obtained.

In Sect. 2.2, we briefly described the kernel TSVR in 1-norm initially proposed in [20]. Now we discuss the TSVR for the non-linear case, but in 2-norm.

The non-linear TSVR in 2-norm determines the ε-insensitive down- and up-bound regressors in the feature space by solving the following pair of QPPs:

$$\min_{(w_1,b_1,\xi_1)\in R^{m+1+m}} \frac{1}{2}||y-\varepsilon_1 e-(K(A,A^t)w_1+b_1e)||^2+\frac{C_1}{2}\xi_1^t\xi_1$$

subject to

$$y-(K(A,A^t)w_1+b_1e)\geq \varepsilon_1 e-\xi_1 \tag{16}$$

and

$$\min_{(w_2,b_2,\xi_2)\in R^{m+1+m}} \frac{1}{2}||y+\varepsilon_2 e-(K(A,A^t)w_2+b_2e)||^2+\frac{C_2}{2}\xi_2^t\xi_2$$

subject to

$$(K(A,A^t)w_2+b_2e)-y\geq \varepsilon_2 e-\xi_2. \tag{17}$$

Proceeding as in the linear TSVR, the dual QPPs of (16) and (17) can be constructed as a pair of minimization problems of the form (14) where $u_1,u_2\in R^m$ are Lagrange multipliers and $Q_1,\ Q_2,\ r_1,\ r_2$ are determined using (13) and (15) in which the augmented matrix G is defined by: $G=[K(A,A^t)\quad e]$. In this case, the kernel regression estimation f: $R^n\rightarrow R$ will be determined using (4) and the down- and up- bound regressors (8), obtained to be: for any vector $x\in R^n$,

$$f_1(x) = [K(x^t, A^t) \quad 1](G^tG)^{-1}G^t(y - \varepsilon_1 e - u_1)$$

and

$$f_2(x) = [K(x^t, A^t) \quad 1](G^tG)^{-1}G^t(y + \varepsilon_2 e + u_2).$$

When linear kernel is used, the above down- and up-bound functions will degenerate into linear functions (1) [20].

Now we discuss our iterative LTSVR algorithm for solving the dual QPPs given by (14).

The KKT necessary and sufficient optimal conditions for the pair of dual QPPs (14) will become determining solutions for the classical complementarity problems [12]:

$$0 \le u_1 \perp (Q_1u_1 - r_1) \ge 0 \quad \text{and} \quad 0 \le u_2 \perp (Q_2u_2 - r_2) \ge 0, \tag{18}$$

respectively. However, the optimality conditions (18) are satisfied if and only if for any $\alpha_1, \alpha_2 > 0$, the relations

$$(Q_1u_1 - r_1) = (Q_1u_1 - \alpha_1u_1 - r_1)_+ \quad \text{and}$$
$$(Q_2u_2 - r_2) = (Q_2u_2 - \alpha_2u_2 - r_2)_+ \tag{19}$$

respectively, must hold [13].

For solving the above pair of problems (19), it is proposed to apply the following simple iterative scheme that constitutes our LTSVR algorithm: $i = 0, 1, 2\ldots$

$$u_1^{i+1} = Q_1^{-1}(r_1 + (Q_1u_1^i - \alpha_1u_1^i - r_1)_+) \quad \text{and}$$
$$u_2^{i+1} = Q_2^{-1}(r_2 + (Q_2u_2^i - \alpha_2u_2^i - r_2)_+) \tag{20}$$

whose global convergence will follow from the result of [13].

Theorem 1 [13] *Assume that the matrix $(G^tG)^{-1}$ exists and let the conditions:*

$$0 < \alpha_1 < \frac{2}{C_1} \quad \text{and} \quad 0 < \alpha_2 < \frac{2}{C_2}$$

be satisfied. Then,

(i) *The matrices Q_1, Q_2 defined by (15) are symmetric and positive-definite;*
(ii) *Starting with any initial vector $u_k^0 \in R^m$ where $k = 1, 2$, the iterate $u_k^i \in R^m$ of (20) will converge linearly to the unique solution $\bar{u}_k \in R^m$ satisfying the condition*

$$||Q_ku_k^{i+1} - Q_k\bar{u}_k|| \le ||I - \alpha_kQ_k^{-1}||\,||Q_ku_k^i - Q_k\bar{u}_k||.$$

From (20), we immediately notice that our LTSVR algorithm requires at its very beginning the inverse of the matrices Q_1 and Q_2. However, we will show in the next theorem that they need not be computed explicitly since they can be easily obtained once the matrix $(G^tG)^{-1}$ that appears in the original TSVR formulation is known.

Theorem 2 *Assume that the matrix $(G^tG)^{-1}$ exists. Then,*

(i) *$H = H^2$ will be satisfied;*
(ii) *for $k = 1, 2$,*

$$Q_k^{-1} = C_k\left(I - \frac{C_k}{1 + C_k}H\right), \tag{21}$$

where the matrices H and Q_1, Q_2 are given by (13) and (15) respectively.

Proof

(i) The result will follow from the definition of the matrix H.
(ii) Using the result (i) and (15), we have: for $k = 1, 2$,

$$\begin{aligned} Q_kC_k\left(I - \frac{C_k}{1+C_k}H\right) &= C_k\left(\frac{I}{C_k} + H\right)\left(I - \frac{C_k}{1+C_k}H\right) \\ &= I + C_k\left(H - \frac{H}{1+C_k} - \frac{C_k}{1+Ck}H^2\right) \\ &= I + C_k\left(H - \frac{H}{1+C_k} - \frac{C_kH}{1+C_k}\right) = I. \end{aligned}$$

Note that the matrix G^tG is positive semi-definite, and therefore, it may not be invertible. However, if G^tG is invertible then the dual objective functions of the minimization problems (14) are strongly convex and therefore will have unique solutions since Q_1 and Q_2 are positive-definite. In all our numerical experiments, by introducing a regularization term δI, the inverse of the matrix G^tG is computed to be $(\delta I + G^tG)^{-1}$ where δ is a very small positive number. Finally, we observe, by Theorem 2, that for the implementation of LTSVR algorithm only the inverse of G^tG needs to be known.

4 Numerical experiments and comparison of results

In order to evaluate the efficiency of LTSVR, numerical experiments were performed on eight synthetic and several well-known, publicly available, real-world benchmark datasets, and their results were compared with SVR and TSVR. All the regression methods were implemented in MATLAB R2010a environment on a PC running on Windows XP OS with 2.27 GHz Intel(R) Xeon(R) processor having 3 GB of RAM. The standard SVR was solved by MOSEK optimization toolbox [23] for MATLAB. For TSVR, however, we used the optimization toolbox of MATLAB. In all the examples considered, the Gaussian kernel function with parameter $\sigma > 0$, defined by: for $x_1, x_2 \in R^m$

$$k(x_1, x_2) = \exp\left(-\frac{||x_1 - x_2||^2}{2\sigma^2}\right),$$

is used. The 2-norm root mean square error (RMSE), given by the following formula:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2},$$

is chosen to measure the accuracy of the results obtained, where y_i and $\tilde{y}_i$ are the observed and its corresponding predicted values, respectively, and N is the number of test samples.

Since more parameters need to be selected in both TSVR and LTSVR, this will lead to, however, a slower model selection speed in comparison to SVR, and therefore, in all experiments $\varepsilon_1 = \varepsilon_2 = 0.01$, $C_1 = C_2$ and $\delta = 10^{-5}$ were assumed. For SVR, we set $\varepsilon = 0.01$. The optimal values of the parameters were determined by performing 10-fold cross-validation on the training dataset, where the regularization parameter values $C_1 = C_2 = C$ and the kernel parameter value σ were allowed to vary from the sets $\{10^{-5}, 10^{-4}, \ldots, 10^{5}\}$ and $\{2^{-10}, 2^{-9}, \ldots, 2^{10}\}$, respectively. Finally, choosing these optimal values, the RMSE on the test dataset was calculated.

4.1 Synthetic datasets

As the first example, we considered the function [11] defined as below:

$$y = f(x) = \frac{4}{|x| + 2} + \cos(2x) + \sin(3x), \quad x \in [-10, 10].$$

Using the function definition, 200 examples for training and 1,000 examples for testing were chosen randomly from the interval $[-10, 10]$. The observed values were polluted by adding two different kinds of noises: uniform noises over the interval $[-0.2, 0.2]$ and Gaussian noises with mean zero and standard deviation 0.2, that is we have taken:

$$y = f(x) + \xi,$$

so that ξ is an additive noise. Test data were taken to be free of noises. The plots of the original function and its approximations using SVR, TSVR and LTSVR for uniform and Gaussian noises were shown in Fig. 1a, b, respectively along with the noisy training samples marked by the symbol 'o'. The optimal values of the parameters were determined by the tuning procedure explained earlier. The 10-fold numerical results of non-linear SVR, TSVR and LTSVR were summarized in Table 2.

We performed numerical experiments on another seven synthetic datasets generated by functions listed in Table 1. As in the previous case, 200 examples were generated for training whose observed values were polluted by uniform and Gaussian noises, and a test set consisting of 1,000 examples, free of noises, was considered. By performing 10-fold cross-validation, the numerical results obtained were summarized in Table 2.

4.2 Real-world benchmark datasets

In this sub-section, numerical tests and comparisons were carried out on real-world datasets.

We considered the Box and Jenkins gas furnace example [3] as the first real-world dataset. It consists of 296 input–output pairs of values of the form: $(u(t), y(t))$ where $u(t)$ is input gas flow rate whose output $y(t)$ is the CO_2 concentration from the gas furnace. We predict $y(t)$ based on 10 attributes taken to be of the form: $x(t) = (y(t-1), y(t-2), y(t-3), y(t-4), u(t-1), u(t-2), \quad u(t-3), u(t-4), u(t-5), u(t-6))$ [21]. Thus, we get 290 samples in total where each sample is of the form $(x(t), y(t))$. The first 100 samples for training and the rest for testing were assumed. The performances of SVR, TSVR and LTSVR on the training and test sets were shown in Fig. 2a, b, respectively.

We performed numerical experiments non-linearly on several real-world datasets: Boston housing, Auto-Mpg, Machine CPU, Servo, Auto price, Wisconsin B.C, Concrete CS, Abalone, Kin-fh, Bodyfat, Google, IBM, Intel, Redhat, Microsoft, S&P500, Sunspots, SantaFeA and the data generated by Lorenz and Mackey–glass differential equations.

The Boston housing, Auto-Mpg, Machine CPU, Servo, Auto price, Wisconsin B.C and Concrete CS are popular regression benchmark dataset available at: http://archive.ics.uci.edu/ml/datasets. The details on the number of training and test samples considered for our experiment along with the number of attributes are listed in Table 3.

We conducted our experiment on Abalone dataset [18] using 1,000 samples for training and the remaining 3,177 samples for testing. The predicted value is the age of an abalone using 8 physical measurements as their features. The Kin-fh dataset [7] represents a realistic simulation of the forward dynamics of eight links all revolute robot arm. The observed value is the prediction of the end-effector from a target with 32 features. The first 1,000 samples were taken for training and the remaining 7,192 samples for testing.

In all the previous real-world examples considered, each attribute of the original data is normalized as follows:

$$\bar{x}_{ij} = \frac{x_{ij} - x_j^{\min}}{x_j^{\max} - x_j^{\min}}$$

where x_{ij} is the (i, j)th element of the input matrix A, $\bar{x}_{ij}$ is its corresponding normalized value and $x_j^{\min} = \min_{i=1}^{m}(x_{ij})$ and $x_j^{\max} = \max_{i=1}^{m}(x_{ij})$ denote the minimum and maximum values, respectively, of the jth column of A. However, in the following examples, the original data are normalized with mean zero and standard deviation equals to 1.

Bodyfat is another benchmark dataset and is taken from the Statlib collection: http://lib.stat.cmu.edu/datasets, where

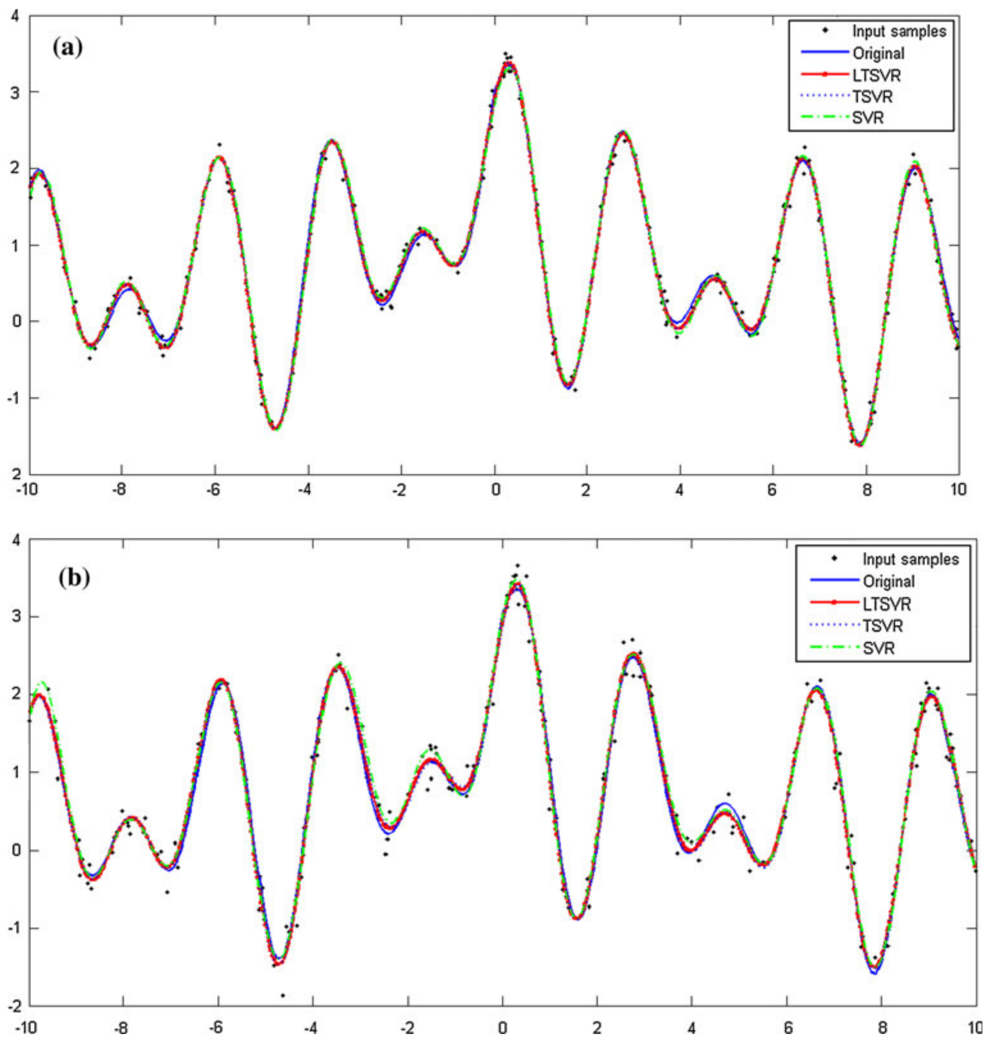

Fig. 1 Results of approximation of $\frac{4}{|x|+2} + \cos(2x) + \sin(3x)$ with LTSVR, SVR and TSVR when different kinds of additive noises were used. Gaussian kernel was employed. **a** Uniform noises over the interval $[-0.2, 0.2]$. **b** Gaussian noises with mean zero and standard deviation 0.2

Table 1 Functions used for generating synthetic datasets

Name	Function definition	Domain of definition
Function 1	$\frac{4}{\lvert x\rvert+2} + \cos(2x) + \sin(3x)$	$x \in [-10,\ 10]$
Function 2	$\frac{4}{\lvert x\rvert+2} + \cos(2x)$	$x \in [-10,\ 10]$
Function 3	$\frac{1+\sin(2x_1+3x_2)}{3.5+\sin(x_1-x_2)}$	$x_1,\ x_2 \in [-2,\ 2]$
Function 4	$\frac{\sin\sqrt{x_1^2+x_2^2}}{\sqrt{x_1^2+x_2^2}}$	$x_1, \mathrm{x}_2 \in [-4\pi, 4\pi]$
Function 5	$\tan^{-1}\frac{x_2x_3-\frac{1}{x_2x_4}}{x_1}$	$x_1 \in [0, 100],\ \ x_2 \in [40\pi, 560\pi],$ $x_3 \in [0, 1],\ \ x_4 \in [1, 11]$
Function 6	$0.79 + 1.27x_1x_2+1.56x_1x_4 + 3.42x_2x_5 + 2.06x_3x_4x_5$	$x_1, x_2, x_3, x_4, x_5 \in [0, 1]$
Function 7	$1.3356(e^{3(x_2-0.5)}\sin(4\pi(x_2-0.9)^2)$ $+ 1.5(1-x_1) + e^{(2x_1-1)}\sin(3\pi(x_1-0.6)^2))$	$x_1, x_2 \in [-0.5, 0.5]$
Function 8	$\frac{40e^{8\{(x_1-0.5)^2+(x_2-0.5)^2\}}}{e^{8\{(x_1-0.2)^2+(x_2-0.7)^2\}}+e^{8\{(x_1-0.7)^2+(x_2-0.2)^2\}}}$	$x_1, x_2 \in [-1, 1]$

Table 2 Performance comparison of the proposed LTSVR with SVR and TSVR on synthetic datasets for Uniform and Gaussian additive noises

Datasets	Uniform noises			Gaussian noises		
(Train size, test size)	SVR RMSE Time (C, σ)	TSVR RMSE Time $(C_1 = C_2, \sigma)$	LTSVR RMSE Time $(C_1 = C_2, \sigma)$	SVR RMSE Time (C, σ)	TSVR RMSE Time $(C_1 = C_2, \sigma)$	LTSVR RMSE Time $(C_1 = C_2, \sigma)$
Function 1	0.0710	0.0516	**0.0483**	0.1020	0.0676	**0.0498**
	0.2133	0.1420	0.0147	0.3054	0.1521	0.0156
(200 × 1, 1,000 × 1)	$(10^3, 2^0)$	$(10^5, 2^0)$	$(10^0, 2^0)$	$(10^3, 2^0)$	$(10^0, 2^0)$	$(10^0, 2^0)$
Function 2	0.0628	**0.0474**	0.0510	**0.0797**	0.0860	0.0855
	0.3532	0.2812	0.0181	0.2824	0.0826	0.0146
(200 × 1, 1,000 × 1)	$(10^3, 2^1)$	$(10^1, 2^0)$	$(10^0, 2^0)$	$(10^3, 2^1)$	$(10^{-1}, 2^0)$	$(10^{-5}, 2^0)$
Function 3	0.0215	0.0196	**0.0133**	0.0520	0.0309	**0.0307**
	0.2563	0.8093	0.0197	0.3123	0.2621	0.0139
(200 × 2, 1,000 × 2)	$(10^3, 2^0)$	$(10^4, 2^0)$	$(10^0, 2^0)$	$(10^3, 2^1)$	$(10^{-5}, 2^0)$	$(10^0, 2^0)$
Function 4	0.0578	**0.0307**	0.0400	0.1240	**0.0065**	0.0085
	0.2438	0.0854	0.0219	0.2188	0.0802	0.0199
(200 × 2, 1,000 × 2)	$(10^3, 2^2)$	$(10^3, 2^2)$	$(10^0, 2^2)$	$(10^1, 2^2)$	$(10^{-1}, 2^2)$	$(10^{-2}, 2^2)$
Function 5	0.0318	0.0240	**0.0221**	**0.0675**	0.0961	0.0962
	0.2582	0.1308	0.0112	0.2160	0.1905	0.0108
(200 × 4, 1,000 × 4)	$(10^3, 2^{10})$	$(10^5, 2^9)$	$(10^1, 2^9)$	$(10^3, 2^6)$	$(10^{-2}, 2^9)$	$(10^{-5}, 2^9)$
Function 6	0.0286	**0.0152**	0.0160	0.0311	0.0145	**0.0136**
	0.3128	0.1205	0.0147	0.3442	0.8027	0.0156
(200 × 5, 1,000 × 5)	$(10^3, 2^1)$	$(10^5, 2^0)$	$(10^0, 2^1)$	$(10^3, 2^1)$	$(10^{-5}, 2^1)$	$(10^0, 2^1)$
Function 7	0.0658	**0.0246**	0.0349	0.0946	0.0411	**0.0396**
	0.3578	0.2508	0.0145	0.3054	0.1030	0.0087
(200 × 2, 1,000 × 2)	$(10^3, 2^{-3})$	$(10^1, 2^{-3})$	$(10^0, 2^{-3})$	$(10^2, 2^{-4})$	$(10^{-1}, 2^{-2})$	$(10^{-5}, 2^{-2})$
Function 8	0.0721	**0.0311**	0.0455	**0.0397**	0.0555	0.0527
	0.3065	0.3023	0.0143	0.3823	0.2401	0.0144
(200 × 2, 1,000 × 2)	$(10^3, 2^{-1})$	$(10^4, 2^{-1})$	$(10^0, 2^{-1})$	$(10^3, 2^{-1})$	$(10^0, 2^0)$	$(10^0, 2^0)$

RMSE and training time were used for comparison. Gaussian kernel was employed. Bold type shows the best result

the observed value being the estimation of the body fat from the body density values. It consists of 252 samples with 14 attributes. The first 150 samples were taken for training and the rest for testing.

In the following time series examples considered, five previous values are used to predict the current value.

Google, IBM, Intel, Red Hat, Microsoft and Standard & Poor 500 (S&P500) are financial datasets of stock index taken from the Yahoo website: http://finance.yahoo.com. For our experimental study, we considered 755 closing prices starting from 01–01–2006 to 31–12–2008, and using them, we get 750 samples. For each of the above financial time series examples considered, the first 200 samples were taken for training and the rest 550 for testing.

The sunspots dataset is well-known and is commonly used to compare regression algorithms. It is taken from http://www.bme.ogi.edu/~ericwan/data.html. We considered the annual readings of sunspots from the year 1700 to 1994. With five previous values being used to predict its current value, we get 290 samples in total and from which the first 100 samples were taken for training and the rest for testing. SantaFe-A is a laser time series dataset recorded from a Far-Intrared-Laser in a chaotic state. It consists of 1,000 numbers of time series values, and therefore, we get 995 samples in total. The first 200 samples were considered for training and the rest for testing. This dataset is available at: http://www-psych.stanford.edu/~andreas/Time-Series/SantaFe.html.

Taking the values of the parameters as: $\rho = 10$, $r = 28$ and $b = 8/3$, two datasets ,Lorenz$_{0.05}$ and Lorenz$_{0.20}$, corresponding to the sampling rates $\tau = 0.05$ and $\tau = 0.20$, respectively, were generated using the time series values associated to the variable x of the Lorenz differential equation [16, 17]:

$$\dot{x} = \rho(y - x), \dot{y} = rx - y - xz \quad \text{and} \quad \dot{z} = xy - bz,$$

obtained by fourth-order Runge–Kutta method. They consist of 30,000 number of time series values. The first 1,000

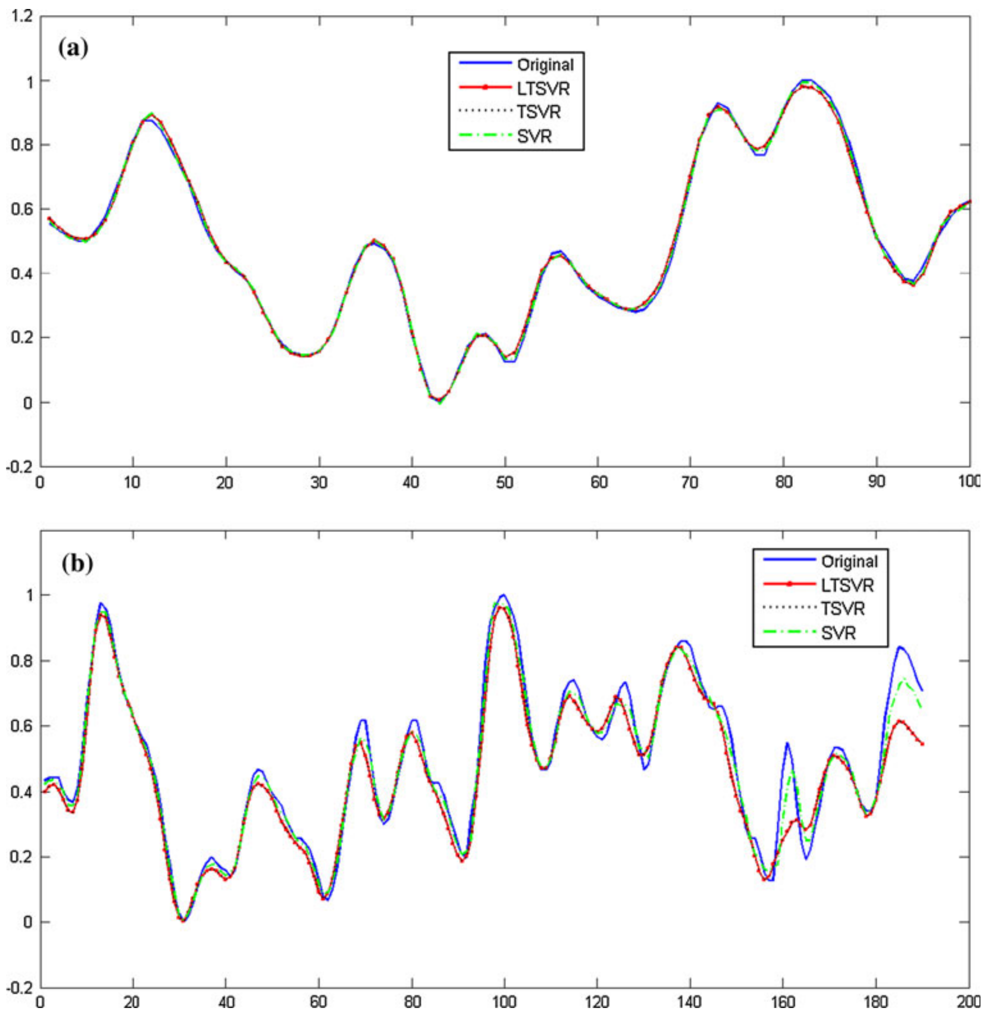

Fig. 2 Results of comparison for Gas furnace. Gaussian kernel was employed. **a** Prediction over the training set. **b** Prediction over the test set

values were discarded to avoid the initial transients. The next 3,000 values were taken for our experiment. Among them, the first 500 samples were taken for training and the remaining 2,495 samples for testing.

Finally, for comparing the performance of LTSVR with SVR and TSVR, two time series generated by the Mackey–Glass time delay differential equation [16, 17], given by:

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t-\tau)}{1+x(t-\tau)^{10}},$$

with respect to the parameters $\tau = 17, 30$, were considered. Let us denote these two time series by MG_{17} and MG_{30}. They are available at: http://www.cse.ogi.edu/~ericwan. Among the total of 1,495 samples obtained, the first 500 were considered for training and the rest for testing.

The 10-fold numerical results obtained by SVR, TSVR and LTSVR on each dataset along with the number of training and test samples taken, the number of attributes, the optimal values of the parameters determined and the training time were summarized in Table 3. Similar or better generalization performance of the proposed method in less execution time for training clearly demonstrates that the proposed algorithm is a powerful method of solution for regression problems.

5 Conclusions

A new iterative Lagrangian support vector machine algorithm for the twin SVR in its dual is proposed. Our formulation leads to minimization problems having their objective functions strongly convex with non-negativity constraints only. Though at the outset, the algorithm requires inverse of matrices, it was shown that they could be obtained by performing a simple matrix subtraction of the identity matrix with a scalar multiple of the inverse of a positive semi-definite matrix that arises in the original twin SVR formulation. Our formulation has the advantage that the unknown regressor is obtained using an extremely simple linearly convergent iterative algorithm rather than solving QPPs as in SVR and twin SVR. Also the proposed algorithm does not need any specialized optimization

Table 3 Performance comparison of the proposed LTSVR with SVR and TSVR on real-world datasets

Datasets (Train size, test size)	SVR RMSE Time (C, σ)	TSVR RMSE Time $(C_1 = C_2, \sigma)$	LTSVR RMSE Time $(C_1 = C_2, \sigma)$
Gas furnace	**0.0344**	0.0577	0.0634
	0.0958	0.0902	0.0039
(100 × 10, 190 × 10)	$(10^4, 2^{-5})$	$(10^4, 2^2)$	$(10^0, 2^2)$
Boston housing	0.1505	0.1365	**0.1241**
	0.3026	0.1232	0.0114
(200 × 13, 306 × 13)	$(10^4, 2^3)$	$(10^0, 2^2)$	$(10^{-1}, 2^4)$
Auto-Mpg	0.1635	**0.1539**	0.1546
	0.0926	0.0722	0.0047
(100 × 7, 292 × 7)	$(10^2, 2^1)$	$(10^{-2}, 2^1)$	$(10^0, 2^1)$
Machine CPU	0.0509	0.0274	**0.0231**
	0.2054	0.1502	0.0115
(150 × 7, 59 × 7)	$(10^5, 2^4)$	$(10^2, 2^4)$	$(10^0, 2^1)$
Servo	0.1906	0.1554	**0.1393**
	0.1028	0.0827	0.0048
(100 × 4, 67 × 4)	$(10^3, 2^1)$	$(10^5, 2^0)$	$(10^2, 2^{-2})$
Auto price	0.1540	0.1400	**0.1369**
	0.1094	0.0520	0.0099
(120 × 15, 39 × 15)	$(10^3, 2^7)$	$(10^5, 2^5)$	$(10^0, 2^5)$
Wisconsin B.C.	0.2187	**0.2131**	0.2152
	0.0820	0.0598	0.0036
(100 × 34, 94 × 34)	$(10^4, 2^6)$	$(10^{-1}, 2^3)$	$(10^{-4}, 2^3)$
Concrete CS	0.177	**0.1539**	0.1544
	5.1563	0.8622	0.2751
(700 × 8, 330 × 8)	$(10^1, 2^{-1})$	$(10^{-2}, 2^1)$	$(10^{-3}, 2^1)$
Abalone	0.1972	0.1499	**0.1432**
	14.963	2.1902	0.5769
(1,000 × 8, 3,177 × 8)	$(10^1, 2^0)$	$(10^{-1}, 2^1)$	$(10^{-5}, 2^1)$
Kin-fh	0.1340	0.0990	**0.0933**
	13.468	4.0206	0.7897
(1,000 × 32, 7,192 × 32)	$(10^1, 2^8)$	$(10^1, 2^4)$	$(10^1, 2^4)$
Bodyfat	**0.0594**	0.0773	0.1176
	0.1902	0.0821	0.0108
(150 × 14, 102 × 14)	$(10^3, 2^7)$	$(10^0, 2^5)$	$(10^{-1}, 2^5)$
Google	**0.1324**	0.1383	0.1408
	0.2890	0.0701	0.0186
(200 × 5, 550 × 5)	$(10^3, 2^5)$	$(10^0, 2^4)$	$(10^{-1}, 2^4)$
IBM	0.1505	0.1614	**0.1431**
	0.2016	0.1006	0.0148
(200 × 5, 550 × 5)	$(10^1, 2^2)$	$(10^3, 2^5)$	$(10^0, 2^4)$
Intel	0.1566	0.1560	**0.1541**
	0.2572	0.0903	0.0188
(200 × 5, 550 × 5)	$(10^3, 2^5)$	$(10^0, 2^4)$	$(10^{-1}, 2^4)$
Red Hat	0.1636	**0.1624**	0.1637
	0.2630	0.0712	0.0161
(200 × 5, 550 × 5)	$(10^3, 2^7)$	$(10^0, 2^4)$	$(10^0, 2^4)$
Microsoft	**0.1726**	**0.1726**	0.1743
	0.2762	0.1372	0.0160
(200 × 5, 550 × 5)	$(10^3, 2^5)$	$(10^0, 2^4)$	$(10^0, 2^4)$
S&P500	0.1346	0.1812	**0.1811**
	0.2340	0.9420	0.0139
(200 × 5, 550 × 5)	$(10^4, 2^{-5})$	$(10^{-5}, 2^4)$	$(10^{-3}, 2^4)$
Sunspots	0.3778	**0.3579**	0.3592
	0.0834	0.0521	0.0053
(100 × 5, 190 × 5)	$(10^1, 2^2)$	$(10^0, 2^3)$	$(10^0, 2^3)$
SantafeA	0.2752	0.3229	**0.2614**
	0.2644	0.1025	0.0091
(200 × 5, 795 × 5)	$(10^1, 2^0)$	$(10^1, 2^1)$	$(10^{-5}, 2^1)$
$\text{Lorenz}_{0.2}$	**0.0224**	0.0524	0.0524
	2.8613	0.5302	0.1255
(500 × 5, 2,495 × 5)	$(10^3, 2^4)$	$(10^0, 2^1)$	$(10^{-1}, 2^1)$
$\text{Lorenz}_{0.05}$	0.0157	**0.0153**	0.0155
	2.9212	0.6432	0.1163
(500 × 5, 2,495 × 5)	$(10^3, 2^5)$	$(10^0, 2^1)$	$(10^{-1}, 2^1)$
MG_{17}	0.0119	**0.0117**	0.0122
	2.2031	0.3805	0.1197
(500 × 5, 995 × 5)	$(10^3, 2^1)$	$(10^{-1}, 2^0)$	$(10^{-1}, 2^0)$
MG_{30}	**0.0886**	0.0921	0.0925
	2.1563	0.3328	0.1096
(500 × 5, 995 × 5)	$(10^1, 2^0)$	$(10^{-2}, 2^0)$	$(10^{-3}, 2^0)$

RMSE and training time were used for comparison. Gaussian kernel was employed. The best result is indicated by bold face

software. Similar or better generalization performance of the proposed method on synthetic and real-world datasets in less computational time than the standard and twin SVR methods clearly illustrates its effectiveness and suitability. Future work will include the study of implicit Lagrangian formulation [14] for the dual twin SVR problem and its applications.

Acknowledgments The authors are extremely thankful to the learned referees for their critical and constructive comments that greatly improved the earlier version of the paper. Mr. Tanveer acknowledges the financial support given as scholarship by Council of Scientific and Industrial Research, India.

References

1. Balasundaram S, Kapil (2010) On Lagrangian support vector regression. Exp Syst Appl 37:8784–8792
2. Balasundaram S, Kapil (2010) Application of Lagrangian twin support vector machines for classification. In: 2nd international

conference on machine learning and computing, ICMLC2010, IEEE Press, pp 193–197

3. Box GEP, Jenkins GM (1976) Time series analysis: forecasting and control. Holden-Day, San Francisco
4. Brown MPS, Grundy WN, Lin D et al (2000) Knowledge-based analysis of microarray gene expression data using support vector machine. Proc Nat Acad Sci USA 97(1):262–267
5. Chen X, Yang J, Liang J, Ye Q (2011) Smooth twin support vector regression. Neural Comput Appl. doi:10.1007/s00521-010-0454-9
6. Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines and other kernel based learning method. Cambridge University Press, Cambridge
7. DELVE (2005) Data for evaluating learning in valid experiments. http://www.cs.toronto.edu/~delve/data
8. Jayadeva, Khemchandani R, Chandra S (2007) Twin support vector machines for pattern classification. IEEE Trans Patt Anal Mach Intell 29(5):905–910
9. Joachims T, Ndellec C, Rouveriol (1998) Text categorization with support vector machines: learning with many relevant features. In: European conference on machine learning no. 10, Chemnitz, Germany, pp 137–142
10. Kumar MA, Gopal M (2009) Least squares twin support vector machines for pattern classification. Expert Syst Appl 36:7535–7543
11. Li G, Wen C, Guang G-B, Chen Y (2011) Error tolerance based support vector machine for regression. Neurocomputing 74(5):771–782
12. Mangasarian OL (1994) Nonlinear programming. SIAM Philadelphia, PA
13. Mangasarian OL, Musicant DR (2001) Lagrangian support vector machines. J Mach Learn Res 1:161–177
14. Mangasarian OL, Solodov MV (1993) Nonlinear complementarity as unconstrained and constrained minimization. Math Program B 62:277–297
15. Mangasarian OL, Wild EW (2006) Multisurface proximal support vector classification via generalized eigenvalues. IEEE Trans Patt Anal Mach Intell 28(1):69–74
16. Mukherjee S, Osuna E, Girosi F (1997) Nonlinear prediction of chaotic time series using support vector machines. In: NNSP'97: Neural networks for signal processing VII: Proceedings of IEEE signal processing society workshop, Amelia Island, FL, USA, pp 511–520
17. Muller KR, Smola AJ, Ratsch G, Schölkopf B, Kohlmorgen J (1999) Using support vector machines for time series prediction. In: Schölkopf B, Burges CJC, Smola AJ (eds) Advances in Kernel methods- support vector learning. MIT Press, Cambridge, pp 243–254
18. Murphy PM, Aha DW (1992) UCI repository of machine learning databases. University of California, Irvine. http://www.ics.uci.edu/~mlearn
19. Osuna E, Freund R, Girosi F (1997) Training support vector machines: an application to face detection. In: Proceedings of computer vision and pattern recognition, pp 130–136
20. Peng X (2010) TSVR: an efficient twin support vector machine for regression. Neural Netw 23(3):365–372
21. Ribeiro B (2002) Kernelized based functions with Minkovsky's norm for SVM regression. In: Proceedings of the international joint conference on neural networks, 2002, IEEE press, pp 2198–2203
22. Shao Y-H, Zhang C-H, Wang X-B, Deng N-Y (2011) Improvements on twin support vector machines. IEEE Trans Neural Netw 22(6):962–968
23. The MOSEK optimization tools (2008) Version 5.0, Denmark. http://www.mosek.com
24. Vapnik VN (2000) The nature of statistical learning theory, 2nd edn. Springer, New York
25. Zhong P, Xu Y, Zhao Y (2011) Training twin support vector regression via linear programming. Neural Comput Appl. doi:10.1007/s00521-011-0526-6