

**TWIN PARAMETRIC INSENSITIVE SUPPORT  
VECTOR REGRESSION IN 1- NORM VIA  
UNCONSTRAINED CONVEX MINIMIZATION**

*Dissertation submitted to the  
Jawaharlal Nehru University, New Delhi  
in Partial fulfillment of the requirements for the award of the degree of*

**MASTER OF TECHNOLOGY  
In  
COMPUTER SCIENCE AND TECHNOLOGY**

**By  
TARUNDEEP KAUR SAINI**

**UNDER THE SUPERVISION OF  
Dr. S. BALASUNDARAM**



**SCHOOL OF COMPUTER AND SYSTEMS SCIENCES  
JAWAHARLAL NEHRU UNIVERSITY  
NEW DELHI-110067, INDIA  
JULY 2015**



SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI – 110 067

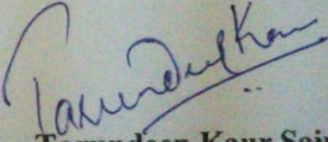
---

---

### DECLARATION

This is to certify that the dissertation entitled “**Twin Parametric Insensitive Support Vector Regression in 1-norm via Unconstrained Convex Minimization**” is being submitted to the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, in partial fulfillment of the requirements for the award of the degree of Master of Technology in Computer Science & Technology, is a record of bonafide work carried out by me under the supervision of **Prof. S. Balasundaram**.

The matter embodied in the dissertation has not been submitted in part or full to any University or Institution for the award of any degree or diploma.

  
**Tarundeep Kaur Saini**  
(Student)



SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI – 110 067

**CERTIFICATE**

This is to certify that this dissertation entitled “**Twin Parametric Insensitive Support Vector Regression in 1-norm via Unconstrained Convex Minimization**” submitted by **Miss. Tarundeep Kaur Saini** to the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, for the award of degree of **Master of Technology in Computer Science & Technology**, is a research work carried out by her under the supervision of **Prof. S. Balasundaram**.

**Prof. S. Balasundaram**

**(Supervisor)**

**Prof. R.K. Aggarwal**

Dean  
School of Computer & Systems Sciences (Dean)  
Jawaharlal Nehru University  
New Delhi-110067

**Dedicated to my Mom...**

## **Acknowledgements**

This dissertation is dedicated to my mother for her constant support and encouragement. Without her support and encouragement, this work wouldn't have taken start.

A debt of gratitude to my supervisor Professor S. Balasundaram of the School of Computer and Systems Sciences for his guidance, help, encouragement and efforts. Successful completion of this dissertation was not been possible without him.

I am thankful to my lab mates Yogendra Meena, Gagandeep and Subash. I am also thankful to my friends Deena Hijam and Aastha Mahajan for supporting me. Last but not the least to my sister Aakashdeep Kaur for always being by my side.

# CONTENTS

Declaration.....	i
Certificate.....	ii
Acknowledgements.....	iv
Contents.....	v
List of figures.....	vii
List of tables.....	viii
Notations and Abbreviations.....	ix
Abstract.....	x
Chapter 1: Support Vector Machines for Regression	
1.1 Introduction.....	1
1.2 Support Vector Regression.....	2
1.2.1 Standard Linear Support Vector Regression.....	3
1.2.2 Standard Non-Linear Support Vector Regression.....	6
1.2.3 Linear 2-norm Support Vector Regression.....	8
1.2.4 Non-Linear 2-norm Support Vector Regression.....	10
Chapter 2: Twin Support Vector Regression	
2.1 Introduction.....	11
2.2 Linear Twin Support Vector Regression.....	11

2.3 Non-Linear Twin Support Vector Regression.....	16
2.4 Twin Parametric Insensitive Support Vector Regression (TPISVR).....	19
Chapter 3: Twin Parametric Insensitive Support Vector Regression in 1-norm via	
Unconstrained Convex Minimization.....	22
Chapter 4: Numerical Experiments.....	31
Chapter 5: Conclusions.....	38
Appendix.....	I
References.....	X

## List of Figures

Fig 1.1 Error Loss Functions.....	5
Fig 1.2 Linear Support Vector Regression.....	6
Fig 1.3 Mapping into a higher dimensional feature space.....	7
Fig 2.1 Twin support vector regression.....	12
Fig 4.1 Results of comparison on servo dataset.....	36
Fig 4.2 Results of comparison on pollution dataset .....	36
Fig 4.3 Results of comparison on Hydraulic actuator dataset .....	37



## **List of Tables**

Table 1.1 Error Loss Functions.....	4
Table 1.2 Generally used Kernels.....	7
Table 4.1 Comparison of performance between the proposed method with SVR and TSVR..	34

## Notations and Abbreviations

- $R$ : Real line
- $R^n$ : Euclidean space of dimension  $n$
- $x$ : Column vector in  $R^n$
- $x^t$ : Transpose of the vector  $x$
- $x^t y$ : Inner product of the vectors  $x$  and  $y$
- $x \perp y$ :  $x$  is orthogonal to  $y$
- $\|x\|$ : 2-norm of a vector  $x$
- $\|Q\|$ : 2-norm of a matrix  $Q$
- $x_+$ : Vector  $x$  with all negative components set to zero
- $(x_*)_i$ : 
$$\begin{cases} 1 & x_i > 0 \\ 0.5 & x_i = 0 \quad \forall i, \text{ where } (\cdot)_i \text{ denotes } i^{\text{th}} \text{ component} \\ 0 & x_i < 0 \end{cases}$$
- $x \geq 0$ : Each component of the vector  $x$  is nonnegative
- $diag(x)$ : Diagonal matrix of order  $n$  whose diagonal elements are the components of the vector  $x$
- $I$ : Identity matrix of appropriate size
- $e$ : Column vector of ones of dimension  $m$
- $K = K(M, N)$ : Kernel matrix  $K$  of size  $m \times \ell$  with matrices  $M \in R^{m \times n}$  and  $N \in R^{n \times \ell}$
- $\nabla f = (\partial f / \partial x_1, \dots, \partial f / \partial x_n)^t$ : Gradient of real valued function  $f$  of the variable  $x = (x_1, \dots, x_n)^t \in R^n$ .
- $\nabla^2 f = (\partial^2 f / \partial x_i \partial x_j)_{i,j=1,\dots,n}$ : Hessian matrix of  $f$
- SVM: Support Vector Machine
- SVR: Support Vector Regression
- QPP: Quadratic Programming Problem

## **ABSTRACT**

In this work, we have introduced an efficient approach to Twin Parametric Insensitive Support Vector Regression. The problem is solved via 1-norm and its dual is further solved by functional iterative method and by the Newton method thus leading to a convex optimization problem having a unique solution. The numerical results of the proposed method are compared with the Support Vector Regression and Twin Support Vector Regression on some real world datasets.

# Chapter 1

## Support Vector Machines for Regression

### 1.1 Introduction

Under the domain of classification or regression many important problems such as face detection (Osuna et al., 1997), gene selection (Guyon et al., 2002), gene prediction (Tong et al., 2005), bio medicine (Brown, Grundy, Lin, et al., 2000), drug discovery (Demiriz et al. 2001), credit scoring (Malhotra & Malhotra, 2003), blind identification (Santamaria et al., 2004), stock exchange prediction (Bao et al., 2005), optical character recognition (Mani & Voumard, 1996), text categorization (Joachims et al., 1998), time series forecasting (Brockwell & Davis, 2002; Cao 2003), financial recognition (Ince & Trafalis, 2002) and brain computer interface (Ebrahimi et al., 2003) lie.

Machine learning methods such as Artificial Neural Networks (ANNs), determine a linear or nonlinear functional relationship between dependent and independent variables. The functional relationship is determined by training of input samples by different methods such as ANNs are trained with back propagation. These traditional learning methods are based on Empirical Risk Minimization (ERM) principle. ANNs may give better approximation capabilities but it suffers from slow convergence, presence of local minima, over-fitting and slow learning rate problems. Even the selection of the number of hidden layer neurons is a difficult task.

Classification and Regression problems have been extensively solved by a machine learning method based on Statistical Learning Theory called Support Vector Machines (SVM)

proposed by Vapnik (Cortes & Vapnik, 1995). SVM is based on principle of Structural Risk Minimization (SRM) (Cristianini & Shawe Taylor, 2000). Unlike ERM which minimizes the training error, SRM minimizes the upper bound on the generalization error thus have high prediction capabilities on unseen data.

SVM formulation leads to a Quadratic Programming Problem (QPP) whose objective function is convex with linear inequality constraints having a unique solution (Cristianini & Shawe Taylor, 2000; Vapnik, 2000). Its computational complexity is  $O(m^3)$  (Jayadeva et al., 2007, Peng, 2010a) where  $m$  is number of training samples. SVM is a widely used method because of a better generalization ability unique optimal solution. SVM was initially proposed for classification problems. It was later extended for regression estimation problems.

In many fields of research such as tsunami alerts, bioinformatics, control theory, signal processing, meteorological prediction, information science and economics the estimation by regression is used. Like SVM, Support Vector Regression (SVR) also determines a regressor model with good generalization capabilities. The resulting model obtained can be used as a tool for analysis, prediction and simulation.

## **1.2 Support vector regression**

Support Vector Regression (SVR) aims at determining a linear regressor model for a given set of data points where the input data are either taken in the original form or taken into a higher dimensional feature space by using a kernel method. The linear regressor model is obtained by determining the functional relationship between the given set of inputs and their corresponding outputs. This relationship may either be linear or nonlinear, which can further be used to predict the output for any concealed or new data.

Throughout in this work all vectors are considered as column vectors. For any two vectors  $\mathbf{x}, \mathbf{y}$  in  $R^n$ , the inner product of the two vectors will be given by  $\mathbf{x}'\mathbf{y}$  where  $\mathbf{x}'$  is the transpose of vector. For a given vector  $\mathbf{x} = (x_1, x_2, \dots, x_n)'$  in  $R^n$ , the plus function  $\mathbf{x}_+$  is given as:  $(\mathbf{x}_+)_i = \max\{0, x_i\}$  for  $i = 1, 2, \dots, n$  and the step function  $\mathbf{x}_*$  is given as:  $(\mathbf{x}_*)_i = 1$  if  $x_i > 0$ ,  $(\mathbf{x}_*)_i = 0$  if  $x_i < 0$  and  $(\mathbf{x}_*)_i = 0.5$  if  $x_i = 0$ . The 2-norm of a vector  $\mathbf{x}$  in  $R^n$  will be denoted by  $\|\mathbf{x}\|$ .  $\mathbf{e}$  is the column vector of one's of dimension  $m$ .  $I$  is the identity matrix of appropriate size. For a matrix  $A \in R^{m \times n}$ ,  $A_i$  is the  $i^{\text{th}}$  row of  $A$  which is a row vector in  $R^n$ . For the two matrices  $A \in R^{m \times n}$  and  $B \in R^{n \times k}$ , the kernel  $K(A, B)$  maps  $R^{m \times n} \times R^{n \times k}$  into  $R^{m \times k}$ .

### 1.2.1 Standard Linear Support Vector Regression

Under this, we will briefly illustrate standard 1-norm and 2-norm formulation for linear support vector regression problem.

For a given training set  $\{(x_i, y_i)\}_{i=1, \dots, m}$  in which  $x_i \in R^n$  and  $y_i \in R$ .  $y_i$  is the observed value corresponding to the input samples  $x_i$ . Let  $A \in R^{m \times n}$  be the matrix representation of the input samples.  $x_i^t$  be the  $i^{\text{th}}$  row of the input matrix and  $y = (y_1, \dots, y_m)^t$  be the vector of output values. In the linear SVR problem, the output  $y$  is estimated by a function  $f(\cdot)$  of the form

$$f(x) = x^t w + b, \quad (1.1)$$

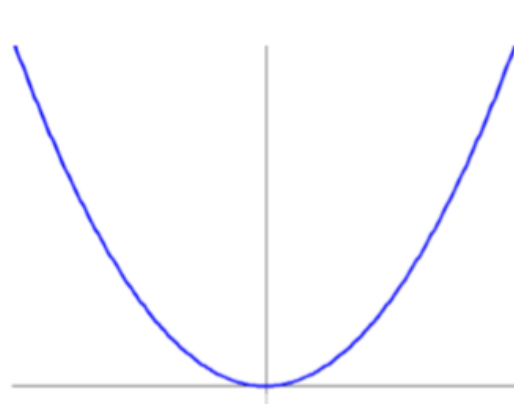
where  $w \in R^n$  and  $b \in R$  are attained by solving the following unconstrained minimization

problem 
$$\min_{(w,b) \in R^{n+1}} \frac{1}{2} \|w\|^2 + \frac{\nu}{m} \sum_{i=1}^m L(f(x_i), y_i), \quad (1.2)$$

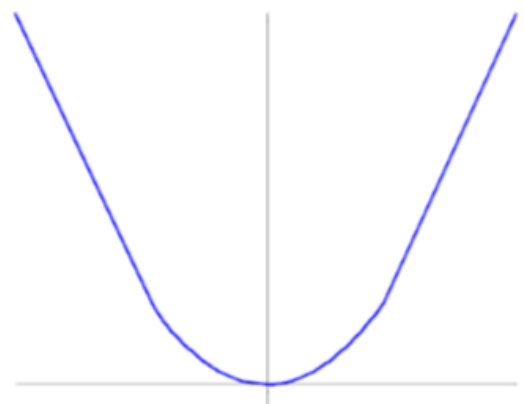
Where  $L$  is the error loss function. The term  $\frac{1}{m} \sum_{i=1}^m L(f(x_i), y_i)$  is the average loss and  $\frac{1}{2} \|w\|^2$  is the regularization term. The parameter  $\nu > 0$  is the trade-off between the regularization term and the error. There are certain generally used error loss functions. They are given as:

Error Loss Functions	Function Definition
<b>Quadratic Loss</b> (conventional loss function used in least squares method)	$(f(x), y)^2$
<b>Huber Loss</b> (Gunn, 1998)	$\begin{cases} \frac{1}{2} (f(x), y)^2, & \text{if }  f(x) - y  < \varepsilon \\ \varepsilon  f(x) - y  - \frac{\varepsilon^2}{2}, & \text{otherwise} \end{cases}$
<b><math>\varepsilon</math>-insensitive loss</b> (Vapnik, 1998)	$\begin{cases} 0, & \text{if }  f(x) - y  - \varepsilon \leq 0 \\  f(x) - y  - \varepsilon, & \text{otherwise} \end{cases}$

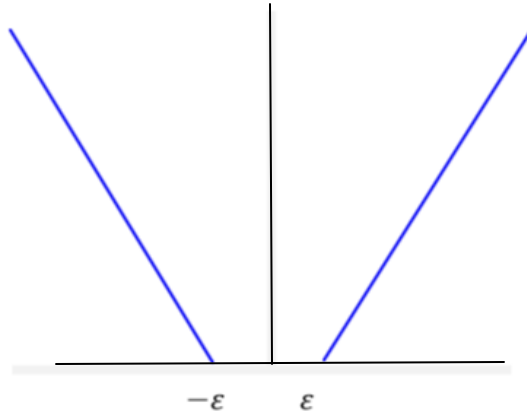
Table 1.1 Error loss functions



(a) Quadratic loss function



(b) Huber loss function



(c) Epsilon insensitive loss function

Fig 1.1 Error loss functions

Throughout the discussion we will be considering  $\varepsilon$ -insensitive loss function (Vapnik, 1998).

By considering  $\varepsilon$ -insensitive loss function in the above unconstrained minimization problem (1.2) can be illustrated as

$$\min_{(w,b) \in R^{n+1}} \frac{1}{2} \|w\|^2 + \frac{\nu}{m} \sum_{i=1}^m |x_i^t w + b - y_i|_{\varepsilon}, \quad (1.3)$$

where  $|x_i^t w + b - y_i|_{\varepsilon} = \max\{0, |x_i^t w + b - y_i| - \varepsilon\}$  and  $\nu > 0$  and  $\varepsilon > 0$  are parameters.

The unconstrained minimization problem can be given as a constrained quadratic optimization problem (Lee et al., 2005). Problem (1.3) can be given as

$$\min_{(w,b,\xi,\xi^*) \in R^{n+1+m+m}} \frac{1}{2} w^t w + \nu (\mathbf{e}^t \xi + \mathbf{e}^t \xi^*) \quad (1.4)$$

subject to  $y - Aw - b\mathbf{e} \leq \varepsilon\mathbf{e} + \xi, \quad Aw + b\mathbf{e} - y \leq \varepsilon\mathbf{e} + \xi^*$

$\xi_i, \xi_i^* \geq 0$  for  $i = 1, 2, \dots, m$

where  $\xi = (\xi_1, \xi_2, \dots, \xi_m)^t, \xi^* = (\xi_1^*, \xi_2^*, \dots, \xi_m^*)^t$  are vectors of slack variables.



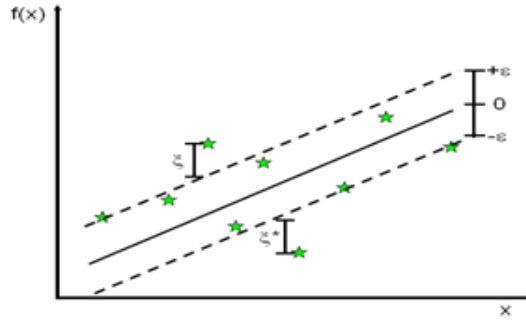


Fig 1.2 Linear support vector regression

This formulation has  $2m$  non negative variables and  $2m$  linear inequality constraints. So rather than solving this problem, it is proposed to solve its dual (Balasundram & Kapil, 2010). The dual to this problem (1.4) can be illustrated by introducing Lagrange multipliers and is

$$\text{formulated as: } \min_{u_1, u_2 \in R^m} \frac{1}{2} (u_1 - u_2)^t A A^t (u_1 - u_2) - y^t (u_1 - u_2) + \epsilon e (u_1 + u_2) \quad (1.5)$$

$$\text{subject to } e^t (u_1 - u_2) = 0 \quad \text{and} \quad 0 \leq u_1, u_2 \leq v,$$

where  $u_1, u_2 \in R^m$  and  $u_1 = (u_{11}, u_{12}, \dots, u_{1m})^t$ ,  $u_2 = (u_{21}, u_{22}, \dots, u_{2m})^t$  are Lagrange multipliers.

Thus for any input sample set the prediction  $f(\cdot)$  is given as

$$f(x) = x^t A^t (u_1 - u_2) + b.$$

### 1.2.2 Standard Non-Linear Support Vector Regression

In non-linear regression the input sample set is mapped into a higher dimensional feature space. This is done by applying a kernel function. The kernel function  $K(\cdot, \cdot)$  is defined on the input space. The support vector regression estimation is then obtained in this feature space.

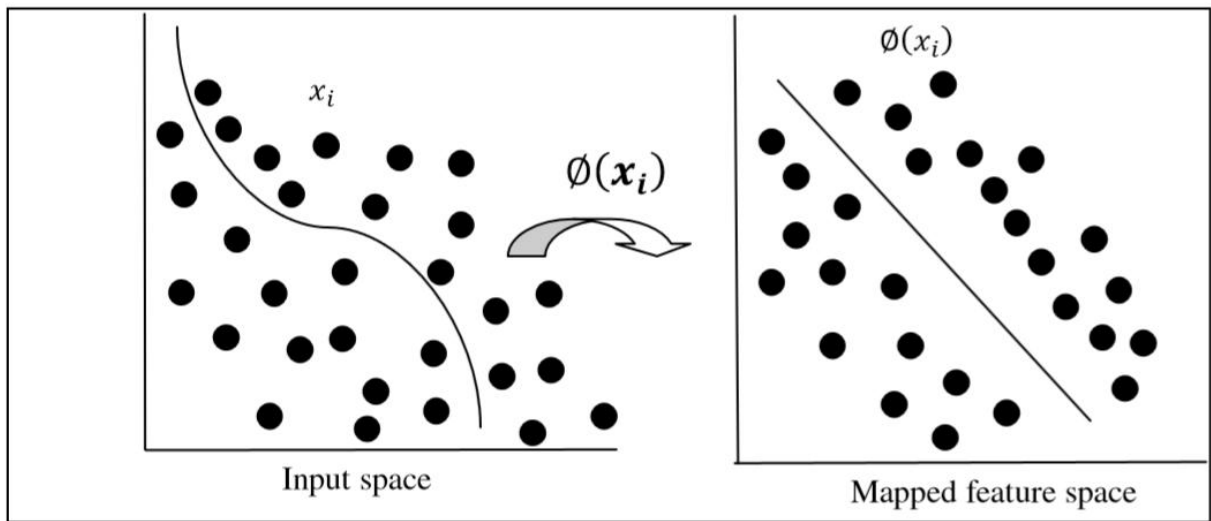


Fig 1.3 Mapping into a higher dimensional feature space.

A few generally used kernel functions are:

Polynomial kernel	$k(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x}^t \mathbf{y})^d, d > 1$
Radial basis kernel	$k(\mathbf{x}, \mathbf{y}) = \exp(-\mu \ \mathbf{x} - \mathbf{y}\ ^2), \mu > 0$
Neural network kernel	$k(\mathbf{x}, \mathbf{y}) = \tanh(\rho_1 \mathbf{x}^t \mathbf{y} + \rho_2), \rho_1 > 0, \rho_2 > 0$

Table 1.2 Generally used Kernels

where  $d, \mu, \rho_1$  and  $\rho_2$  are kernel parameters.

Throughout this work we will consider the popular radial based kernel.

Let  $K = K(A, A^t)$  be the kernel matrix obtained after applying kernel function. Then  $K(x^t, A^t) = (k(x, x_1), \dots, k(x, x_m))$  is the row vector in  $R^m$ ,  $A \in R^{m \times n}$  and the  $(i, j)^{th}$  element of this kernel matrix is given as  $K(A, A^t)_{ij} = k(x_i, x_j)$ .

By applying the kernel trick (Cristianini & Shawe-Taylor, 2000; Vapnik, 2000) to the problem (1.5) we get:

$$\min_{u_1, u_2 \in R^{m+m}} \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m (u_{1i} - u_{2i})^t k(x_i, x_j) (u_{1j} - u_{2j}) - \sum_{i=1}^m y^t (u_{1i} - u_{1i}) + \sum_{i=1}^m \varepsilon e(u_{1i} + u_{2i}), \quad (1.6)$$

subject to  $e^t(u_1 - u_2) = 0$  and  $0 \leq u_1, u_2 \leq v$ ,

where  $u_1 = (u_{11}, u_{12}, \dots, u_{1m})^t$ ,  $u_2 = (u_{21}, u_{22}, \dots, u_{2m})^t$  are Lagrange multipliers and  $k(.,.)$  is kernel function.

Using the solution of problem (1.6), for any input sample set, the regression estimation function  $f(x)$  is obtained and can be illustrated as:

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

### 1.2.3 Linear 2-Norm Support Vector Regression

The 2-norm formulation we consider the square of the 2-norm of the slack variables  $\xi$  and  $\xi^*$  instead of 1-norm (Mangasarian & Musicant (2001); Musicant & Feinberg, 2004). Also, to the objective function of problem (1.4) we add the term  $\left(\frac{b^2}{2}\right)$ . Thus we obtain the following constrained minimization problem (Musicant & Feinberg, 2004):

$$\min_{(w, b, \xi, \xi^*) \in R^{n+1+m+m}} \frac{1}{2} (w^t w + b^2) + \frac{\nu}{2} \sum_{i=1}^m (\xi_i^2 + \xi_i^{*2}) \quad (1.7)$$

subject to  $(y_i - A_i w - b) \leq (\varepsilon + \xi_i)$  and  $(A_i w + b - y_i) \leq (\varepsilon + \xi_i^*)$  for  $i = 1, 2, \dots, m$ ,

where  $\xi_i, \xi_i^*$  are slack variables and  $\nu, \varepsilon$  are input parameters.

On introduction of Lagrange multipliers:

$$u_1 = (u_{11}, u_{12}, \dots, u_{1m})^t \text{ and } u_2 = (u_{21}, u_{22}, \dots, u_{2m})^t,$$

we can obtain the Lagrangian function as:

$$L(w, b, \xi, \xi^*, u_1, u_2) = \frac{1}{2}(w^t w + b^2) + \frac{\nu}{2} \sum_{i=1}^m (\xi_i^2 + \xi_i^{*2}) + \sum_{i=1}^m u_{1i} (y_i - A_i w - b - \varepsilon - \xi_i) + \sum_{i=1}^m u_{2i} (A_i w + b - y_i - \varepsilon - \xi_i^*)$$

By using the condition that at optimality the partial derivatives of L with respect to the primal variables will be zero; the dual problem is illustrated as (Musicant & Feinberg, 2004):

$$\min_{0 \leq u_1, u_2 \in R^m} \frac{1}{2} [(u_1 - u_2)^t (A A^t + e e^t) (u_1 - u_2)] + \frac{1}{2\nu} (u_1^t u_1 + u_2^t u_2) - y^t (u_1 - u_2) + \varepsilon e^t (u_1 + u_2) \quad (1.8)$$

$$\text{where } w = A^t (u_1 - u_2) \text{ and } b = e^t (u_1 - u_2). \quad (1.9)$$

Thus by substituting (1.9) into (1.1) the linear regression estimation function is

$$\begin{aligned} f(x) &= x^t A^t (u_1 - u_2) + e^t (u_1 - u_2) \\ &= [x^t \quad 1] \begin{bmatrix} A^t \\ e^t \end{bmatrix} (u_1 - u_2). \end{aligned}$$

Let G be an augmented matrix defined as  $G = [A \ e]$ . The above estimation function can be given as:  $f(x) = [x^t \quad 1] G^t (u_1 - u_2)$ .

The dual problem (1.8) is written as:

$$\min_{0 \leq u_1, u_2 \in R^m} \frac{1}{2} [u_1^t \quad u_2^t] Q \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} - r^t \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad (1.10)$$

where  $Q = \begin{bmatrix} \frac{1}{\nu} + G G^t & -G G^t \\ -G G^t & \frac{1}{\nu} + G G^t \end{bmatrix}$  and  $r = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} y - \varepsilon e \\ -y - \varepsilon e \end{bmatrix}$  are block matrices.

### 1.2.4 Non-Linear 2- Norm Support Vector Regression

Since,  $K = K(A, A^t)$  is defined as the kernel matrix obtained after applying kernel function.  $K(x^t, A^t) = (k(x, x_1), \dots, k(x, x_m))$  is the row vector in  $R^m$ ,  $A \in R^{m \times n}$  and  $(i, j)^{th}$  element of this kernel matrix is  $K(A, A^t)_{ij} = k(x_i, x_j)$ .

By replacing  $GG^t$  by kernel matrix  $K = K(G, G^t)$ , which is a positive semi-definite symmetric matrix, the nonlinear support vector regression problem in dual variable is formulated as

$$\min_{0 \leq u_1, u_2 \in R^m} \frac{1}{2} [u_1^t \quad u_2^t] Q \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} - r^t \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

where  $Q = \begin{bmatrix} \frac{1}{\nu} + K(G, G^t) & -K(G, G^t) \\ -K(G, G^t) & \frac{1}{\nu} + K(G, G^t) \end{bmatrix}$  and  $r = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} y - \varepsilon e \\ -y - \varepsilon e \end{bmatrix}$  are block matrices.

Thus the regression estimation function  $f(\cdot)$  is given as:

$$f(x) = K([x^t \quad 1], G^t)(u_1 - u_2).$$

## Chapter 2

### Twin Support Vectors Regression

#### 2.1 Introduction

Support vector machine (SVM) is an excellent kernel-based tool for binary data classification and regression.

Recently, Jayadeva et al., (2007) have proposed a Twin Support Vector Machine (TSVM) classifier. TSVM is used for binary data classification and is in the spirit of Generalized Eigen value Proximal Support Vector Machine (GEP-SVM) (Mangasarian & Wild, 2006). This formulation of TSVM is very similar to that of classical SVM. The only difference is that it aims at generating two nonparallel planes such that each plane is closer to one class and as far as possible from other class.

Because of its low computational complexity (Jayadeva et al., 2007), TSVM becomes one of the generally used methods in machine learning. Motivated by the work of Jayadeva et al., (2007), Twin Support Vector Regression (TSVR) formulation was proposed by Peng (2010) for epsilon insensitive regression. Like TSVM, the new formulation TSVR determines two nonparallel, up and down- bound regressors by solving two quadratic programming problems of smaller size than the classical SVR.

#### 2.2 Linear Twin Support Vector Regression (TSVR)

In this section, we formulate Twin Support Vector Regression (TSVR) proposed by Peng (2010) for epsilon insensitive regression. As it was suggested in (Peng, 2010), the linear

formulation leads to determining two functions, such that either of them approximate the  $\varepsilon$ -insensitive up- and down-bounds of unknown regression function.

$$f_1(x) = w_1^t x + b_1 \text{ and } f_2(x) = w_2^t x + b_2 . \quad (2.1)$$

where  $w_1, w_2 \in R^n$  and  $b_1, b_2 \in R$  are the unknowns and  $f_1$  and  $f_2$  may or may not be parallel.

TSVR determines the two bound functions by solving the following pair of QPPs:

$$\min \frac{1}{2} (y - e\varepsilon_1 - (Aw_1 + eb_1))^t (y - e\varepsilon_1 - (Aw_1 + eb_1)) + C_1 e^t \xi_1$$

$$\text{subject to } y - (Aw_1 + eb_1) \geq e\varepsilon_1 - \xi_1 \text{ and } \xi_1 \geq 0 \quad (2.2)$$

and

$$\min \frac{1}{2} (y - e\varepsilon_2 - (Aw_2 + eb_2))^t (y - e\varepsilon_2 - (Aw_2 + eb_2)) + C_2 e^t \xi_2$$

$$\text{subject to } y - (Aw_2 + eb_2) \geq e\varepsilon_2 - \xi_2 \text{ and } \xi_2 \geq 0 \quad (2.3)$$

where  $\varepsilon_1, \varepsilon_2 > 0$  are input parameters;  $C_1, C_2 > 0$  are regularization parameters and

$$e^t = (1, \dots, 1) \in R^m.$$

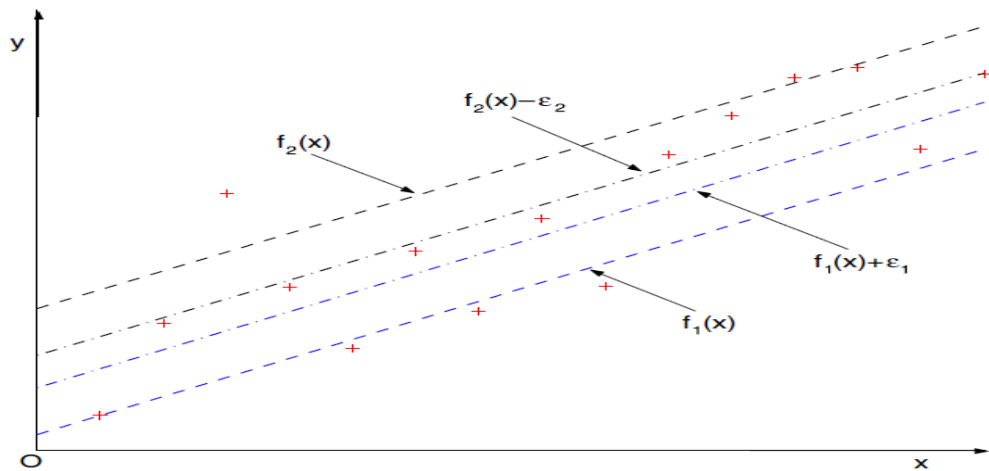


Fig 2.1 Twin support vector regression

By introducing Lagrangian multipliers  $\alpha_1, \alpha_2, \beta_1, \beta_2 \in R^m$  and for  $i = 1, 2$ ;  $\alpha_i, b_i \geq 0$  where  $\alpha_i = (\alpha_{i1}, \dots, \alpha_{im})^t, \beta_i = (\beta_{i1}, \dots, \beta_{im})^t$ , the Lagrangian functions corresponding to (2.2) and (2.3) will be given by

$$\begin{aligned}
L_1(w_1, b_1, \xi_1, \alpha_1, \beta_1) &= \frac{1}{2}(y - e\varepsilon_1 - (Aw_1 + eb_1))^t (y - e\varepsilon_1 - (Aw_1 + eb_1)) + C_1 e^t \xi_1 \\
&\quad - \alpha_1^t (y - (Aw_1 + eb_1) - e\varepsilon_1 + \xi_1) - \beta_1^t \xi_1
\end{aligned} \tag{2.4}$$

and

$$\begin{aligned}
L_2(w_2, b_2, \xi_2, \alpha_2, \beta_2) &= \frac{1}{2}(y - e\varepsilon_2 - (Aw_2 + eb_2))^t (y - e\varepsilon_2 - (Aw_2 + eb_2)) + C_2 e^t \xi_2 \\
&\quad - \alpha_2^t (y - (Aw_2 + eb_2) - e\varepsilon_2 + \xi_2) - \beta_2^t \xi_2
\end{aligned} \tag{2.5}$$

Applying the Karush-Kuhn-Tucker (KKT) necessary and sufficient optimality conditions (Mangasarian, 1994) for the problem (2.4) and (2.5) they become:

$$-A^t(y - Aw_1 - eb_1 - e\varepsilon_1) + A^t \alpha = 0, \tag{2.6}$$

$$-e^t(y - Aw_1 - eb_1 - e\varepsilon_1) + \varepsilon^t \alpha_1 = 0 \tag{2.7}$$

$$C_1 e - \alpha_1 - \beta_1 = 0 \tag{2.8}$$

$$y - (Aw_1 + eb_1) \geq e\varepsilon_1 - \xi_1, \xi_1 \geq 0, \tag{2.9}$$

$$\alpha_1^t (y - (Aw_1 + eb_1) - e\varepsilon_1 + \xi_1) = 0, \alpha_1 \geq 0, \tag{2.10}$$

$$\beta_1^t \xi_1 = 0, \beta_1 \geq 0 \tag{2.11}$$

$$\text{Since } \beta_1 \geq 0, \text{ we have } 0 \leq \alpha_1 \leq C_1 e \tag{2.12}$$

And



$$-A^t(y - Aw_2 - eb_2 - e\varepsilon_2) - A^t\alpha = 0, \quad (2.13)$$

$$-e^t(y - Aw_2 - eb_2 - e\varepsilon_2) - \varepsilon^t\alpha_2 = 0 \quad (2.14)$$

$$C_2e - \alpha_2 - \beta_2 = 0 \quad (2.15)$$

$$y - (Aw_2 + eb_2) \geq e\varepsilon_2 - \xi_2, \xi_2 \geq 0, \quad (2.16)$$

$$\alpha_2^t(y - (Aw_2 + eb_2) \geq e\varepsilon_2 - \xi_2) = 0, \alpha_2 \geq 0, \quad (2.17)$$

$$\beta_2^t\xi_2 = 0, \beta_2 \geq 0 \quad (2.18)$$

$$\text{Since } \beta_2 \geq 0, \text{ we have } 0 \leq \alpha_2 \leq C_2e \quad (2.19)$$

By combining the first two equations ((2.6) and (2.7) we get

$$-\begin{bmatrix} A^t \\ e^t \end{bmatrix} \{ (y + e\varepsilon_1) - [A \ e] \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} \} + \begin{bmatrix} A^t \\ b^t \end{bmatrix} \alpha_1 = 0 \quad (2.20)$$

Similarly combining equations (2.13) and (2.14) we get

$$-\begin{bmatrix} A^t \\ e^t \end{bmatrix} \{ (y + e\varepsilon_2) - [A \ e] \begin{bmatrix} w_2 \\ b_2 \end{bmatrix} \} - \begin{bmatrix} A^t \\ b^t \end{bmatrix} \alpha_2 = 0 \quad (2.21)$$

Introducing the notations

$$G = [A \ e]_{m \times (n+1)}, \quad f_1 = y - e\varepsilon_1, \quad u_1 = [w_1^t \ b_1^t]^t, \quad f_2 = y - e\varepsilon_2, \quad u_2 = [w_2^t \ b_2^t]^t \quad (2.22)$$

equations (2.20) and (2.21) can be rewritten as:

$$-G^t f_1 + G^t G u_1 + G^t \alpha_1 = 0, \quad i. e. u_1 = (G^t G)^{-1} G(f_1 - \alpha_1) \quad (2.23)$$

$$\text{and} \quad -G^t f_2 + G^t G u_2 - G^t \alpha_2 = 0, \quad i. e. u_2 = (G^t G)^{-1} G(f_2 + \alpha_2) \quad (2.24)$$

Notice that it may be possible that the matrix  $G^t G$  may be singular in some cases. However, the matrix  $G^t G$  should always be positive semi definite. In order to overcome such case, we introduce a regularization term  $\delta I$ , where  $\delta$  is very small positive number such as  $\delta = 1e - 7$ ,

i.e. one may consider  $(\delta l + G^t G)^{-1}$  instead of  $(G^t G)^{-1}$ . Thus, in such case, equations (2.23) and (2.24) become

$$u_1 = (\delta l + G^t G)^{-1} G(f_1 - \alpha_1) \quad (2.25)$$

$$u_2 = (\delta l + G^t G)^{-1} G(f_2 + \alpha_2) \quad (2.26)$$

respectively.

Substituting (2.22) and the KKT conditions in (2.2) and eliminating all the terms which are independent of  $\alpha_1$  and  $\alpha_2$  the dual of the pair of problems (2.1) and (2.2) as QPPs can be written as

$$\min \frac{1}{2} \alpha_1^t G(G^t G)^{-1} G \alpha_1 - f_1^t G(G^t G)^{-1} G \alpha_1 + f_1^t \alpha_1$$

$$\text{subject to} \quad 0 \leq \alpha_1 \leq eC_1 \quad (2.27)$$

and

$$\min \frac{1}{2} \alpha_2^t G(G^t G)^{-1} G \alpha_2 + f_2^t G(G^t G)^{-1} G \alpha_2 - f_2^t \alpha_2$$

$$\text{subject to} \quad 0 \leq \alpha_2 \leq eC_2 \quad (2.28)$$

Solving the problems (2.27) and (2.28) for  $\alpha_1$  and  $\alpha_2$  the unknowns of the up and down-bounds, i.e.  $\begin{bmatrix} w_2 \\ b_2 \end{bmatrix}$  and  $\begin{bmatrix} w_1 \\ b_1 \end{bmatrix}$  can be derived, satisfying the conditions

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (G^t G)^{-1} G^t (f_1 - \alpha_1) \quad (2.29)$$

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (G^t G)^{-1} G^t (f_2 + \alpha_2) \quad (2.30)$$

TSVR uses the mean of its up- and down- bounds to determine its value, which is given by

$$f_1(x) = x^t w_1 + b_1 = [x^t \quad 1] \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = [x^t \quad 1] (G^t G)^{-1} G^t (f_1 - \alpha_1) \quad (2.31)$$

$$f_2(x) = x^t w_2 + b_2 = [x^t \quad 1] \begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = [x^t \quad 1] (G^t G)^{-1} G^t (f_2 + \alpha_2) \quad (2.32)$$

Finally, the regression estimation function is determined as:

$$f(x) = \frac{1}{2} (f_1(x) + f_2(x)) \quad \text{for any } x \in R^n \quad (2.33)$$

### 2.3 Nonlinear Twin Support Vector Regression

The non-linear up- and down-bounds of unknown regression function are

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

The optimization problem is constructed as

$$\min \frac{1}{2} (y - e\varepsilon_1 - (K(x^t, A^t) w_1 + eb_1))^t (y - e\varepsilon_1 - (K(x^t, A^t) w_1 + eb_1)) + C_1 e^t \xi_1$$

$$\text{subject to} \quad y - (K(x^t, A^t) w_1 + eb_1) \geq e\varepsilon_1 - \xi_1 \quad \text{and} \quad \xi_1 \geq 0 \quad (2.34)$$

and

$$\min \frac{1}{2} (y - e\varepsilon_2 - (K(x^t, A^t) w_2 + eb_2))^t (y - e\varepsilon_2 - (K(x^t, A^t) w_2 + eb_2)) + C_2 e^t \xi_2$$

$$\text{subject to} \quad y - (K(x^t, A^t) w_2 + eb_2) \geq e\varepsilon_2 - \xi_2 \quad \text{and} \quad \xi_2 \geq 0 \quad (2.35)$$

where  $\varepsilon_1, \varepsilon_2 > 0$  are input parameters;  $C_1, C_2 > 0$  are regularization parameters and

$$e^t = (1, \dots, 1) \in R^m.$$

By introducing Lagrangian multipliers  $\alpha_1, \alpha_2, \beta_1, \beta_2 \in R^m$  and for  $i = 1, 2$ ;  $\alpha_i, b_i \geq 0$  where  $\alpha_i = (\alpha_{i1}, \dots, \alpha_{im})^t, \beta_i = (\beta_{i1}, \dots, \beta_{im})^t$ , the Lagrangian functions corresponding to (2.2) and (2.3) will be given by

$$\begin{aligned}
L_1(w_1, b_1, \xi_1, \alpha_1, \beta_1) & \\
&= \frac{1}{2} (y - e\varepsilon_1 - (K(x^t, A^t)w_1 + eb_1))^t (y - e\varepsilon_1 - (K(x^t, A^t)w_1 + eb_1)) \\
&\quad + C_1 e^t \xi_1 - a_1^t (y - (K(x^t, A^t)w_1 + eb_1) - e\varepsilon_1 + \xi_1) - \beta_1^t \xi_1
\end{aligned} \tag{2.36}$$

and

$$\begin{aligned}
L_2(w_2, b_2, \xi_2, \alpha_2, \beta_2) & \\
&= \frac{1}{2} (y - e\varepsilon_2 - (K(x^t, A^t)w_2 + eb_2))^t (y - e\varepsilon_2 - (K(x^t, A^t)w_2 + eb_2)) \\
&\quad + C_2 e^t \xi_2 - a_2^t (y - (K(x^t, A^t)w_2 + eb_2) - e\varepsilon_2 + \xi_2) - \beta_2^t \xi_2
\end{aligned} \tag{2.37}$$

Applying the Karush-Kuhn-Tucker (KKT) necessary and sufficient optimality conditions (Mangasarian, 1994) for the problem (2.36) they become:

$$-K(x^t, A^t)^t (y - K(x^t, A^t)w_1 - eb_1 - e\varepsilon_1) + K(x^t, A^t)^t \alpha = 0, \tag{2.38}$$

$$-e^t (y - K(x^t, A^t)w_1 - eb_1 - e\varepsilon_1) + \varepsilon^t \alpha_1 = 0 \tag{2.39}$$

$$C_1 e - \alpha_1 - \beta_1 = 0 \tag{2.40}$$

$$y - (K(x^t, A^t)w_1 + eb_1) \geq e\varepsilon_1 - \xi_1, \quad \xi_1 \geq 0, \tag{2.41}$$

$$\alpha_1^t (y - (K(x^t, A^t)w_1 + eb_1) - e\varepsilon_1 + \xi_1) = 0, \quad \alpha_1 \geq 0, \tag{2.42}$$

$$\beta_1^t \xi_1 = 0, \quad \beta_1 \geq 0 \tag{2.43}$$

$$\text{Since } \beta_1 \geq 0, \text{ we have } 0 \leq \alpha_1 \leq C_1 e \tag{2.44}$$

Combining (2.39) and (2.40), we have

$$-\begin{bmatrix} K(x^t, A^t)^t \\ e^t \end{bmatrix} \{ (y + e\varepsilon_1) - [K(x^t, A^t) e] \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} \} + \begin{bmatrix} K(x^t, A^t)^t \\ b^t \end{bmatrix} \alpha_1 = 0 \tag{2.45}$$

Similarly for (2.37), we have

$$-\begin{bmatrix} K(x^t, A^t)^t \\ e^t \end{bmatrix} \{ (y + e\varepsilon_2) - [K(x^t, A^t) \ e] \begin{bmatrix} w_2 \\ b_2 \end{bmatrix} \} - \begin{bmatrix} K(x^t, A^t)^t \\ b^t \end{bmatrix} \alpha_2 = 0 \quad (2.46)$$

Introducing the notations

$$H = [K(x^t, A^t) \ e]_{m \times (n+1)}, \quad f_1 = y - e\varepsilon_1, \quad u_1 = [w_1^t \ b_1]^t, \quad f_2 = y - e\varepsilon_2, \quad u_2 = [w_2^t \ b_2]^t$$

equations (2.45) and (2.46) can be rewritten as:

$$-H^t f_1 + H^t H u_1 + H^t \alpha_1 = 0, \quad i.e. \ u_1 = (H^t H)^{-1} H (f_1 - \alpha_1) \quad (2.47)$$

$$\text{and} \quad -H^t f_2 + H^t H u_2 - H^t \alpha_2 = 0, \quad i.e. \ u_2 = (H^t H)^{-1} H (f_2 + \alpha_2) \quad (2.48)$$

Substituting the above notations and the KKT conditions in (2.34) and eliminating all the terms which are independent of  $\alpha_1$  and  $\alpha_2$  the dual of the pair of problems (2.34) and (2.35) as QPPs can be written as

$$\min \frac{1}{2} \alpha_1^t H (H^t H)^{-1} H \alpha_1 - f_1^t H (H^t H)^{-1} H \alpha_1 + f_1^t \alpha_1$$

$$\text{subject to} \quad 0 \leq \alpha_1 \leq eC_1 \quad (2.49)$$

and

$$\min \frac{1}{2} \alpha_2^t H (H^t H)^{-1} H \alpha_2 + f_2^t H (H^t H)^{-1} H \alpha_2 - f_2^t \alpha_2$$

$$\text{subject to} \quad 0 \leq \alpha_2 \leq eC_2 \quad (2.50)$$

Solving the problems (2.49) and (2.50) for  $\alpha_1$  and  $\alpha_2$  the unknowns of the down and up-bounds, i.e.  $\begin{bmatrix} w_1 \\ b_1 \end{bmatrix}$  and  $\begin{bmatrix} w_2 \\ b_2 \end{bmatrix}$  can be derived, satisfying the conditions

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (H^t H)^{-1} H^t (f_1 - \alpha_1) \quad (2.51)$$

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (H^t H)^{-1} H^t (f_2 + \alpha_2) \quad (2.52)$$

TSVR uses the mean of its up- and down- bounds to determine its value, which is given by

$$f_1(x) = K(x^t, A^t)^t w_1 + b_1 = [K(x^t, A^t)^t \quad 1] \begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = [K(x^t, A^t)^t \quad 1] (H^t H)^{-1} H^t (f_1 - \alpha_1) \quad (2.53)$$

$$f_2(x) = K(x^t, A^t)^t w_2 + b_2 = [K(x^t, A^t)^t \quad 1] \begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = [K(x^t, A^t)^t \quad 1] (H^t H)^{-1} H^t (f_2 + \alpha_2) \quad (2.54)$$

Finally, the regression estimation function is determined as:

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

$$f(x) = \frac{1}{2} (f_1(x) + f_2(x)) = \frac{1}{2} (w_1 + w_2)^t K(A, x) + \frac{1}{2} (b_1 + b_2).$$

#### 2.4 Twin Parametric Insensitive Support Vector Regression (TPISVR)

Just like TSVR, TPISVR (Peng, 2012) also derives two nonparallel functions. These nonparallel lines around the given input sample is derived by solving two quadratic programming problems (QPP). It determines two linear functions  $f_1$  and  $f_2$  in feature space:

$$f_1(x) = w_1^t x + b_1, \quad \text{and} \quad f_2(x) = w_2^t x + b_2 \quad (2.55)$$

where  $w \in R^n$  and  $b \in R$ .  $f_1(x)$  and  $f_2(x)$  represents the insensitive function.  $f_1(x)$  is the parametric insensitive down bound regression function and  $f_2(x)$  is the parametric insensitive up-bound regression functions.

Optimization of the parametric insensitive up- and down-bound regression functions is done by determining the given pair of QPPs:

$$\min \frac{1}{2} \|w_1\|^2 - \frac{v_1}{l} e^t (\varphi(A)w_1 + b_1 e) + \frac{C_1}{l} e^t \xi$$

such that  $Y \geq \varphi(A)w_1 + b_1e - \xi$  ,  $\xi \geq 0$  (2.56)

where  $Y = (y_1, y_2, \dots, y_m)^t$  is a vector of output values. And

$$\min \frac{1}{2} \|w_2\|^2 - \frac{v_2}{l} e^t (\varphi(A)w_2 + b_2e) + \frac{C_2}{l} e^t \eta$$

such that  $Y \geq \varphi(A)w_2 + b_2e + \eta$  ,  $\eta \geq 0$  (2.57)

where  $Y = (y_1, y_2, \dots, y_m)^t$  is a vector of output values.

The model complexity of the objective function of (2.56) is controlled by the first term.  $f_1(x)$  is made as smooth as possible by minimizing  $\|w\|^2/2$ . The second term  $\frac{v_1}{l} e^t (\varphi(A)w_1 + b_1e)$  maximizes with penalty factor  $v_1$  thus leading the function  $f_1(x)$  to be as large as possible. The third term minimizes the sum of error variables.

For optimizing the equations, the Lagrangian function for the problem is given as

$$L = \frac{1}{2} \|w_1\|^2 - \frac{v_1}{l} e^t (\varphi(A)w_1 + b_1e) + \frac{C_1}{l} e^t \xi - \alpha^t (Y - (\varphi(A)w_1 + b_1e) + \xi) - \gamma^t \xi$$

(2.58)

the KKT necessary and sufficient optimality conditions for the problem are obtained as follows:

$$\frac{\partial L}{\partial w_1} = w_1 + \varphi(A)^t \alpha - \frac{v_1}{l} \varphi(A)^t e = 0 \Rightarrow w_1 = \varphi(A)^t \left( \frac{v_1}{l} e - \alpha \right)$$

(2.59)

$$\frac{\partial L}{\partial w_1} = -\frac{v_1}{l} e^t e + e^t \alpha = 0 \Rightarrow e^t \alpha = v_1$$

(2.60)

$$\frac{\partial L}{\partial \xi} = \frac{C_1}{l} e - \alpha - \gamma = 0$$

(2.61)

$$Y \geq \varphi(A)w_1 + b_1e - \xi, \quad \xi \geq 0$$

(2.62)

$$\alpha^t (Y - (\varphi(A)w_1 + b_1e) + \xi) = 0, \quad \alpha \geq 0$$

(2.63)

$$\gamma^t \xi = 0, \quad \gamma \geq 0$$

(2.64)

Since  $\gamma \geq 0$ , we have

$$0 \leq \alpha \leq \frac{c_1}{l} e \quad (2.65)$$

Substituting the equation (2.59) into the Lagrangian function (2.58) and combining it with (2.60) and (2.65), we attain the dual QPP of (2.56), and is given as

$$\max -\frac{1}{2} \alpha^t K \alpha - Y^t \alpha + \frac{v_1}{l} e^t K \alpha$$

Such that  $0 \leq \alpha \leq \frac{c_1}{l} e, e^t \alpha = v_1$  (2.66)

After the optimization of (2.66), according to (2.59) we obtain the weight vector  $w_1$ . The bias term  $b_1$  is given as

$$b_1 = \frac{1}{|sv_1|} \sum_{i \in sv_1} (y_i - \varphi(A_i) w_i) \quad (2.67)$$

where  $sv_1$  is the index set of samples satisfying  $\alpha_i \in (0, c_1/l), i=1, \dots, l$  where  $l$  is the size of training samples and  $|\cdot|$  is the cardinality of a set.

Similarly, we obtain the dual QPP for (2.57), which is

$$\max -\frac{1}{2} \beta^t K \beta + Y^t \beta + \frac{v_2}{l} e^t K \beta$$

Such that  $0 \leq \beta \leq \frac{c_2}{l} e, e^t \beta = v_2$  (2.68)

Optimizing it we obtain the weight vector  $w_2$  and bias term  $b_2$  to be

$$w_2 = \varphi(A)^t \left( \beta - \frac{v_2}{l} e \right), b_2 = \frac{1}{|sv_2|} \sum_{i \in sv_2} (y_i - \varphi(A_i) w_2)$$

where  $sv_2$  is the index set of samples satisfying  $\beta_i \in (0, c_2/l), i=1, \dots, l$ .

Once (2.66) and (2.68) are optimized, the estimated regressor is thus constructed as follows:

$$f(x) = \frac{1}{2} (f_1(x) + f_2(x)) = \frac{1}{2} \left( \beta - \alpha + \frac{v_1 - v_2}{l} e \right)^t K(A, x) + \frac{1}{2} (b_1 + b_2).$$



## Chapter 3

### Twin Parametric Insensitive Support Vector Regression in 1-norm via Unconstrained Convex Minimization

Motivated by the work on the Twin Parametric Insensitive Support Vector Regression (TPISVR) by Peng (2012), we compare the computational results of the proposed method with SVR and TSVR in terms of performance on some real world data sets.

The TPISVR problem is illustrated in the last chapter in section 2.4. The main goal of twin parametric insensitive support vector regression is in seeking parametric nonlinear regression functions by mapping the training samples from the input space to higher dimensional space.

Consider the input set of examples  $\{(x_i, y_i) \mid i=1, 2, \dots, m\}$  where  $x_i \in R^m$  and its corresponding output being  $y_i \in R$ . Let the input be arranged in the form of a matrix  $A \in R^{m \times n}$  whose  $i^{th}$  row is denoted by  $A_i$  and let  $y = (y_1, y_2, \dots, y_m)^t$  be the output vector. It finds a pair of linear functions in feature space:

$$f_1(x) = w_1^t x + b_1, \text{ and } f_2(x) = w_2^t x + b_2 \quad (3.1)$$

where  $w \in R^n$  is the weight vector and  $b \in R$  is the bias.  $f_1(x)$  and  $f_2(x)$  are the parametric insensitive down- and up-bound regression functions, respectively. For the given training data points  $(A, Y)$ , the function  $f_1(x)$  evaluates the parametric insensitive down-bound regression function, and the function  $f_2(x)$  evaluates the insensitive up-bound function.

By taking  $G = [K(A, A^t) \ e]$  ,  $\bar{u}_1 = \begin{bmatrix} W_1 \\ b_1 \end{bmatrix}$  ,  $\bar{u}_2 = \begin{bmatrix} W_2 \\ b_2 \end{bmatrix}$ ;

the 1-norm TPISVR can be formulated as:

$$\min \|u_1\|_1 - v_1 e^t G \bar{u}_1 + C_1 e^t \bar{\xi}$$

such that  $G \bar{u}_1 - \bar{y} \leq \bar{\xi}$  ,  $\bar{\xi} \geq 0$  (3.2)

and  $\min \|u_2\|_1 - v_2 e^t G \bar{u}_2 + C_2 e^t \bar{\xi}$

such that  $-G \bar{u}_2 + \bar{y} \leq \bar{\eta}$  ,  $\bar{\eta} \geq 0$  (3.3)

and  $f(\bar{x}) = [k(\bar{x}^t, A^t) \ 1] \left( \frac{(\bar{u}_1 + \bar{u}_2)}{2} \right)$

Assume:  $\bar{u}_1 = \bar{r}_1 - \bar{s}_1$ ,  $\bar{r}_1, \bar{s}_1 \geq 0$  ,  $\bar{r}_1, \bar{s}_1 \in R^{m+1}$

Then the problem (3.2) becomes:

$$\min \bar{e}_{m+1}^t (\bar{r}_1 + \bar{s}_1) - v_1 e^t G (\bar{r}_1 - \bar{s}_1) + C_1 e^t \bar{\xi}$$

such that  $G(\bar{r}_1 - \bar{s}_1) - \bar{y} \leq \bar{\xi}$  ,  $\bar{\xi} \geq 0$  (3.4)

which can be written as:

$$\min (-v_1 e^t G + \bar{e}_{m+1}^t) \bar{r}_1 + (v_1 e^t G + \bar{e}_{m+1}^t) \bar{s}_1 + C_1 e^t \bar{\xi}$$

such that  $[G \ -G] \begin{bmatrix} \bar{r}_1 \\ \bar{s}_1 \end{bmatrix} - \bar{\xi} \leq \bar{y}$  ,  $\bar{r}_1, \bar{s}_1, \bar{\xi} \geq 0$  (3.5)

or

$$\min [-v_1 e^t G + \bar{e}_{m+1}^t \quad v_1 e^t G + \bar{e}_{m+1}^t \quad C_1 e^t] \begin{bmatrix} \bar{r}_1 \\ \bar{s}_1 \\ \bar{\xi} \end{bmatrix}$$

such that  $[-G_{(m) \times (m+1)} \quad G_{(m) \times (m+1)} \quad I_{m \times m}] \begin{bmatrix} \bar{r}_1 \\ \bar{s}_1 \\ \bar{\xi} \end{bmatrix} \geq -\bar{y}$  ,

$$\bar{r}_1, \bar{s}_1, \bar{\xi} \geq 0 \tag{3.6}$$

So the exterior penalty problem for the dual linear problem(3.2) is:

$$\begin{aligned} \min_{\bar{z}_1 \in R^m} L_1(\bar{z}_1) \\ = \min_{\bar{z}_1 \in R^m} \theta_1(-(-\bar{y})^t \bar{z}_1) \\ + \frac{1}{2} \left\{ \left\| \begin{pmatrix} -G^t \bar{z}_1 + v_1 G^t e - e_{m+1} \\ G^t \bar{z}_1 - v_1 G^t e - e_{m+1} \\ \bar{z}_1 - C_1 e \end{pmatrix} \right\|^2 + \|(-\bar{z}_1)_+\|^2 \right\} \end{aligned}$$

so that

$$\begin{aligned} \bar{r}_1 &= \frac{1}{\theta_1} \{(-G^t \bar{z}_1 + v_1 G^t e - e_{m+1})_+\} \\ \bar{s}_1 &= \frac{1}{\theta_1} \{(G^t \bar{z}_1 - v_1 G^t e - e_{m+1})_+\} \end{aligned} \quad (3.7)$$

where  $(x)_+$  function is defined as  $\max\{x, 0\}$ ,  $L_1(\bar{z}_1)$  is the implicit Lagrangian (Mangasarian and Musicant 2001; Smola and Scholkopf, 2002),  $\theta_1 > 0$  and  $e_{m+1}$  is vector of ones of dimension  $m + 1$ . The implicit Lagrangian formulation (Mangasarian and Musicant 1999, 2001) replaces the negativity constrained quadratic minimization problem by the equivalent unconstrained piecewise quadratic minimization of the implicit Lagrangian thus giving  $L_1(\bar{z}_1)$ .

We apply Newton's method to this unconstrained minimization problem and shows that it terminates in a finite number of steps at the global minima. The gradient of  $L_1(\bar{z}_1)$  is:

$$\begin{aligned} \partial L_1(\bar{z}_1) &= \theta_1 \bar{y} + G(G^t \bar{z}_1 - v_1 G^t e - e_{m+1})_+ - G(-G^t \bar{z}_1 + v_1 G^t e - e_{m+1})_+ + \\ &(\bar{z}_1 - C_1 e)_+ - (-\bar{z}_1)_+ \end{aligned} \quad (3.8)$$

In order to apply the Newton method,  $m \times m$  Hessian matrix of second partial derivatives of  $\partial L_1(\bar{z}_1)$  is required. Because its gradient  $\partial L_1(\bar{z}_1)$  is not differentiable, therefore, it does not exist in ordinary sense. However a generalized hessian of  $L_1(\bar{z}_1)$  is defined as (Hiriart-Urruty et al., 1984; Facchinei, 1995; Mangasarian, 2001):

$$\begin{aligned}
\partial^2 L_1(z_1) &= G \operatorname{diag}((G^t \bar{z}_1 - v_1 G^t e - e_{m+1})_*) G^t \\
&+ G \operatorname{diag}((-G^t \bar{z}_1 + v_1 G^t e - e_{m+1})_*) G^t + \operatorname{diag}((z_1 - C_1 e)_*) \\
&+ \operatorname{diag}((-z_1)_*)
\end{aligned} \tag{3.9}$$

Since

$$\begin{aligned}
\partial L_1(z_1) &= \theta_1 \bar{y} + G(G^t \bar{z}_1 - v_1 G^t e - e_{m+1})_+ - G(-G^t \bar{z}_1 + v_1 G^t e - e_{m+1})_+ + (z_1 - C_1 e)_+ \\
&\quad - (-z_1)_+ \\
&= \theta_1 \bar{y} + G \left\{ \frac{(G^t \bar{z}_1 - v_1 G^t e - e_{m+1}) - |(G^t \bar{z}_1 - v_1 G^t e - e_{m+1})|}{2} \right. \\
&\quad \left. - \frac{(-G^t \bar{z}_1 + v_1 G^t e - e_{m+1}) - |(-G^t \bar{z}_1 + v_1 G^t e - e_{m+1})|}{2} \right\} \\
&\quad + \frac{(z_1 - C_1 e) - |(z_1 - C_1 e)|}{2} - \frac{(-z_1) - |-z_1|}{2} \\
&= \theta_1 \bar{y} + G \left\{ G^t \bar{z}_1 - v_1 G^t e + \frac{|(G^t \bar{z}_1 - v_1 G^t e - e_{m+1})| - |(-G^t \bar{z}_1 + v_1 G^t e - e_{m+1})|}{2} \right\} + z_1 \\
&\quad - \frac{C_1 e}{2} + \frac{|(z_1 - C_1 e)| - |-z_1|}{2} \\
&= (I + GG^t) \bar{z}_1 + G \left\{ \frac{|(G^t \bar{z}_1 - v_1 G^t e - e_{m+1})| - |(-G^t \bar{z}_1 + v_1 G^t e - e_{m+1})|}{2} \right\} \\
&\quad + \frac{|(\bar{z}_1^{(i)} - C_1 e)| - |\bar{z}_1^{(i)}|}{2} + \theta_1 \bar{y} - \frac{C_1 e}{2} - v_1 GG^t e
\end{aligned} \tag{3.10}$$

taking  $\partial L_1(z_1) = 0$  the problem (3.10) is reduced to:

$$\begin{aligned} & (I + GG^t)\bar{z}_1^{(i+1)} \\ &= G \left\{ \frac{\left| (-G^t\bar{z}_1^{(i)} + v_1G^te - e_{m+1}) \right| - \left| (G^t\bar{z}_1^{(i)} - v_1G^te - e_{m+1}) \right|}{2} \right\} \\ & \quad - \frac{|z_1 - C_1e| - |z_1|}{2} - \theta_1\bar{y} + \frac{C_1e}{2} + v_1GG^te \end{aligned}$$

for all  $i = 0, 1, \dots$  (3.11)

Similarly, considering problem (3.3)

$$\min \|u_2\|_1 - v_2e^tG\bar{u}_2 + C_2e^t\bar{\xi}$$

such that  $-G\bar{u}_2 + \bar{y} \leq \bar{\eta}$ ,  $\bar{\eta} \geq 0$

taking  $\bar{u}_2 = \bar{r}_2 - \bar{s}_2$ ,  $\bar{r}_2, \bar{s}_2 \geq 0$ ,  $\bar{r}_2, \bar{s}_2 \in R^{m+1}$

we get  $\min \bar{e}_{m+1}^t(\bar{r}_2 + \bar{s}_2) + v_2e^tG(\bar{r}_2 - \bar{s}_2) + C_2e^t\bar{\eta}$

such that  $-G(\bar{r}_2 - \bar{s}_2) + \bar{y} \leq \bar{\eta}$ ,  $\bar{\eta} \geq 0$  (3.12)

which can be written as:

$$\min(v_2e^tG + \bar{e}_{m+1}^t)\bar{r}_2 + (-v_2e^tG + \bar{e}_{m+1}^t)\bar{s}_2 + C_2e^t\bar{\eta}$$

such that  $[G \quad -G] \begin{bmatrix} \bar{r}_2 \\ \bar{s}_2 \end{bmatrix} - \bar{\eta} \geq \bar{y}$ ,  $\bar{r}_2, \bar{s}_2, \bar{\eta} \geq 0$  (3.13)

or

$$\min[v_2e^tG + \bar{e}_{m+1}^t \quad -v_2e^tG + \bar{e}_{m+1}^t \quad C_2e^t] \begin{bmatrix} \bar{r}_2 \\ \bar{s}_2 \\ \bar{\eta} \end{bmatrix}$$

such that  $[-G_{(m) \times (m+1)} \quad G_{(m) \times (m+1)} \quad I_{m \times m}] \begin{bmatrix} \bar{r}_2 \\ \bar{s}_2 \\ \bar{\eta} \end{bmatrix} \leq -\bar{y}$ ,

$\bar{r}_2, \bar{s}_2, \bar{\eta} \geq 0$  (3.14)

So the exterior penalty problem for the dual linear problem(3.3) is :

$$\min_{\bar{z}_2 \in R^m} L_2(\bar{z}_2) = \min_{\bar{z}_2 \in R^m} \theta_2(-\bar{y}^t \bar{z}_2) + \frac{1}{2} \left\{ \left\| \begin{pmatrix} G^t \bar{z}_2 - v_2 G^t e - e_{m+1} \\ -G^t \bar{z}_2 + v_2 G^t e - e_{m+1} \\ \bar{z}_2 - C_2 e \end{pmatrix} \right\|^2 + \|(-\bar{z}_2)_+\|^2 \right\}$$

so that 
$$\bar{r}_2 = \frac{1}{\theta_2} \{(G^t \bar{z}_2 - v_2 G^t e - e_{m+1})_+\}$$

$$\bar{s}_2 = \frac{1}{\theta_2} \{(-G^t \bar{z}_2 + v_2 G^t e - e_{m+1})_+\} \quad (3.15)$$

where  $\theta_2 > 0$ ,  $(x)_+$  function is defined as  $\max\{x, 0\}$ ,  $L_2(\bar{z}_2)$  is the implicit Lagrangian (Mangasarian and Musicant 2001; Smola and Scholkopf, 2002),  $\theta_2 > 0$  and  $e_{m+1}$  is vector of ones of dimension  $m + 1$ .

The gradient of  $L_2(\bar{z}_2)$  is:

$$\begin{aligned} \partial L_2(\bar{z}_2) = & -\theta_2 \bar{y} + G(G^t \bar{z}_2 - v_2 G^t e - e_{m+1})_+ - G(-G^t \bar{z}_2 + v_2 G^t e - \\ & e_{m+1})_+ + (\bar{z}_2 - C_2 e)_+ - (-\bar{z}_2)_+ \end{aligned} \quad (3.16)$$

generalized hessian of  $L_2(\bar{z}_2)$  is defined as:

$$\begin{aligned} \partial^2 L_2(\bar{z}_2) = & G \text{diag}((G^t \bar{z}_2 - v_2 G^t e - e_{m+1})_*) G^t - G \text{diag}((-G^t \bar{z}_2 + v_2 G^t e - e_{m+1})_*) G^t \\ & + \text{diag}((\bar{z}_2 - C_2 e)_*) + \text{diag}((-\bar{z}_2)_*) \end{aligned} \quad (3.17)$$

Since

$$\begin{aligned} \partial L_2(\bar{z}_2) & = -\theta_2 \bar{y} + G(G^t \bar{z}_2 - v_2 G^t e - e_{m+1})_+ - G(-G^t \bar{z}_2 + v_2 G^t e - e_{m+1})_+ \\ & + (\bar{z}_2 - C_2 e)_+ - (-\bar{z}_2)_+ \end{aligned}$$

$$= \theta_2 \bar{y}$$

$$+ G \left\{ \frac{(G^t \bar{z}_2 - v_2 G^t e - e_{m+1}) - |(G^t \bar{z}_2 - v_2 G^t e - e_{m+1})|}{2} \right. \\ \left. - \frac{(-G^t \bar{z}_2 + v_2 G^t e - e_{m+1}) - |(-G^t \bar{z}_2 + v_2 G^t e - e_{m+1})|}{2} \right\} \\ + \frac{(z_2 - C_2 e) - |(z_2 - C_2 e)|}{2} - \frac{(-z_2) - |(-z_2)|}{2}$$

$$= \theta_2 \bar{y}$$

$$+ G \left\{ G^t \bar{z}_2 - v_2 G^t e \right. \\ \left. + \frac{|(G^t \bar{z}_2 - v_2 G^t e - e_{m+1})| - |(-G^t \bar{z}_2 + v_2 G^t e - e_{m+1})|}{2} \right\} \\ + \bar{z}_2 - \frac{C_2 e}{2} + \frac{|(\bar{z}_2 - C_2 e)| - |(z_2)|}{2}$$

$$= (I + GG^t) \bar{z}_2$$

$$+ G \left\{ \frac{|(G^t \bar{z}_2 - v_2 G^t e - e_{m+1})| - |(-G^t \bar{z}_2 + v_2 G^t e - e_{m+1})|}{2} \right\} \\ + \frac{|(z_2 - C_2 e)| - |z_2|}{2} - \theta_2 \bar{y} - \frac{C_2 e}{2} - v_2 GG^t e$$

(3.18)

taking  $\partial L_2(z_2) = 0$  the problem (3.18) is reduced to:

$$\begin{aligned}
& (I + GG^t)\bar{z}_2^{(i+1)} \\
&= G \left\{ \frac{\left| (-G^t\bar{z}_2^{(i)} + v_2G^te - e_{m+1}) \right| - \left| (G^t\bar{z}_2^{(i)} - v_2G^te - e_{m+1}) \right|}{2} \right\} \\
&\quad - \frac{\left| (\bar{z}_2^{(i)} - C_2e) \right| - \left| \bar{z}_2^{(i)} \right|}{2} + \theta_2\bar{y} + \frac{C_2e}{2} + v_2GG^te
\end{aligned}$$

for all  $i = 0, 1, \dots$  (3.19)

The problems (3.11) and (3.19) are solved by functional iterative method as well as Newton's method.

Newton Iteration is given as:

$$\partial L(z^i) + \partial L(z^i)(z^{i+1} - z^i) = 0 \quad (3.20)$$

For determining the piecewise quadratic minimization problem for an arbitrary positive definite  $Q$  applying simplified iteration (3.20) together with Armijo step size the Newton algorithm is applied. The Newton algorithm is given as:

*Start with any  $u^0 \in R^m$ . For  $i = 0, 1, \dots$ :*

i. *Stop if  $h\left(u^i - \partial h(u^i)^{-1}h(u^i)\right) = 0$*

ii.  $u^{i+1} = u^i - \lambda_i \partial h(u^i)^{-1}h(u^i) = u^i + \lambda_i d^i,$

where  $\lambda_i = \max\left\{1, \frac{1}{2}, \frac{1}{4}, \dots\right\}$  is the Armijo step size such that:

$$L(u^i) - L(u^i + \lambda_i d^i) \geq -\delta \lambda_i \nabla L(u^i)' d^i$$

for some  $\delta \in \left(0, \frac{1}{2}\right)$  and  $d^i$  is the newton direction:



$$d^i = -\partial h(u^i)^{-1} h(u^i)$$

iii.  $i = i + 1$ . Goto (i).

The Newton method is given as:

$$\partial^2 L_1(\bar{z}_1^i) (\bar{z}_1^{(i+1)} - \bar{z}_1^i) = -\partial L_1(\bar{z}_1^i); \partial^2 L_2(\bar{z}_2^i) (\bar{z}_2^{(i+1)} - \bar{z}_2^i) = -\partial L_2(\bar{z}_2^i)$$

for all  $i = 0, 1, \dots$  (3.21)

Since  $f(x) = \frac{1}{2}(f_1(x) + f_2(x)) = [k(x^t, A^t) \ 1] \left( \frac{(u_1 + u_2)}{2} \right)$ .

## Chapter 4

### Numerical experiments

In this chapter, we determine the performance of the proposed method by comparing it to SVR and TSVR on some real world data sets. For this purpose we consider some real world datasets- IBM, Google, Intel, Redhat, Santafe, Flexible robotic arm, pollution, Citigroup Sunspots, Boston housing, Gas furnace, Servo, Concrete Comparison Strength (CS), NO<sub>2</sub>, Hydraulic actuator.

All the algorithms were implemented in MATLAB R2008b on Microsoft Windows7 professional operating system with system configuration Intel(R) Core (TM) i5-3470 processor (2.27 GHz) having 4GB of RAM. We used MOSEK optimization toolbox for MATLAB to realize the standard SVR. It is available at <http://www.mosaik.com>. For TSVR, we used the “quadprog” function in MATLAB for solving the quadratic programming problem.

At the first example, we examined the Box and Jenkins gas furnace data set (Box and Jenkins, 1976). It consists of 296 input-output pairs of samples of the form:  $(u(t), y(t))$  where  $u(t)$  is the input gas flow rate. Its corresponding output  $y(t)$  is the CO<sub>2</sub> concentration from the gas furnace. The output  $y(t)$  is predicted based on 10 attributes taken to be of the form:  $x(t) = (y(t - 1), y(t - 2), y(t - 3), y(t - 4), u(t - 1), u(t - 2), u(t - 3), u(t - 4), u(t - 5), u(t - 6))$ . This makes the total number of samples 290 where each sample is of the form  $(u(t), y(t))$ . For our experiment, we considered the first 100 samples for training and remaining for testing.

The Boston housing, Servo and Concrete CS datasets from UCI repository (Murphy & Aha, 1992); the financial time series datasets: Citigroup, RedHat, Google, Intel and IBM are taken from the website: <http://finance.yahoo.com>. In all the financial time series datasets, we considered 755 closing prices starting from 01-01-2006 to 31-12-2008. Since we used five previous values to predict the current value, we get in total 750 data samples. In our experiments the first 200 of them are taken for training and remaining 550 for testing.

NO2 dataset published by Department for environment, food and rural affairs. It gives the concentration of nitrogen dioxide at background and roadside locations on the geographic coverage of England and Wales. It is taken from [http://data.gov.uk/dataset/air\\_quality\\_-\\_nitrogen\\_dioxide\\_no2\\_concentrations/resource/30233b2d-a8b1-446e-9754-3c74599f2da7](http://data.gov.uk/dataset/air_quality_-_nitrogen_dioxide_no2_concentrations/resource/30233b2d-a8b1-446e-9754-3c74599f2da7).

Pollution dataset is taken from <http://www.eea.europa.eu/themes/air/air-emissions-data>. pollution dataset is compiled and maintained by European Environment Agency. The data is the amount of air pollutants emitted from different sources. These sources are anthropogenic.

The commonly used and well known dataset sunspots is taken from <http://www.bme.ogi.edu/~ericwan/data.html>. It consist of 295 yearly readings from the year 1700 to 1994. Since five previous values are used to predict its current value, we obtained in total 290 samples. In our proposed method we considered the first 100 samples for training and remaining 190 samples for testing.

The Boston dataset have the information of united state census service concerning housing in area of Boston mass. It has total 206 samples and 13 features from which first 200 were taken for training and remaining for testing.

In our last example, we consider the Hydraulic actuator dataset (Gretton et al., 2001; Sjoberg et al., 1995). It consists of 1024 pair of samples of the form  $(u(y), y(t))$  where  $u(t)$  is the

input gas flow rate. Its corresponding output  $y(t)$  is the oil pressure. The output  $y(t) = f(x(t))$  is predicted based on 5 attributes taken to be of the form:  $x(t) = (y(t - 1), y(t - 2), y(t - 3), u(t - 1), u(t - 2))$ . This makes the total number of samples becomes 1021 such that each sample is of the form  $(x(t), y(t))$ . For our experiment, we considered the first 511 samples for training and the remaining samples for testing.

For SVR  $\epsilon = 0.01$

For TSVR  $\epsilon_1 = \epsilon_2 = \epsilon = 0.01$ , regularization parameter  $C_1 = C_2 = C$  and  $\mu$  varies from  $\{ 2^{-5}, \dots, 2^5 \}$

For TPISVR via 1-norm by functional iterative method  $C_1 = C_2 = C$  and varies from  $\{ 10^{-4}, \dots, 10^4 \}$ ,  $n$  and  $\mu$  varies from  $\{ 2^{-5}, \dots, 2^5 \}$  and  $\nu$  is given as  $(n * C)$ , and  $\epsilon_1 = \epsilon_2 = \epsilon = \{ 0.001, 0.01, 0.1 \}$ .

For TPISVR via 1-norm by Newton method  $\mu$  and  $C_1 = C_2 = C$  and varies from  $\{ 10^{-5}, \dots, 10^5 \}$ ,  $n$  varies from 0.01 to 0.1 and  $\nu$  is given as  $(n * C)$ , } and  $\epsilon_1 = \epsilon_2 = \epsilon = \{ 0.001, 0.01, 0.1 \}$ .  
 $\sigma_1 = \sigma_2 = \sigma = \{ 10^{-5}, 10^{-6} \}$ .

Table 4.1 Comparison of performance between the proposed method with SVR and TSVR.

Data sets	SVR RMSE (C, $\mu, \epsilon$ )	TSVR RMSE ( $C_1=C_2=C, \mu, \epsilon$ )	TPISVR via 1-norm by functional iterative method (FTSVR) RMSE (n, $C_1=C_2=C, \mu, \epsilon$ )	TPISVR via 1-norm by Newton method (NTSVR) RMSE (n, $C_1=C_2=C, \mu, \epsilon, \sigma$ )
Gas Furnace (293 X 6)	<b>0.0169</b> ( $10^2, 2^{-4}, 10^{-2}$ )	0.0200 ( $10^{-1}, 2^{-1}, 10^{-1}$ )	0.0319 ( $10^3, 10^2, 2^1, 0.1$ )	0.0178 (0.1, $10^4, 2^{-5}, 0.1, 10^{-6}$ )
Boston Housing (506 X 13)	0.0874 ( $10^2, 2^{-5}, 10^{-2}$ )	0.0859 ( $10^{-1}, 2^{-1}, 3, 10^{-2}$ )	0.0773 ( $10^5, 10^0, 2^{-1}, 0.1$ )	<b>0.0754</b> (0.09, $10^4, 2^{-5}, 0.1, 10^{-6}$ )
Servo (167 X 4)	0.0779 ( $10^2, 2^{-1}, 10^{-3}$ )	0.0812 ( $10^2, 2^0, 10^{-1}$ )	0.0775 ( $10^4, 10^2, 2^2, 0.001$ )	<b>0.0606</b> (0.01, $10^5, 2^0, 0.1, 10^{-6}$ )
ConcreteCS (1030 X 8)	0.1016 ( $10^1, 2^{-1}, 10^{-2}$ )	0.1051 ( $10^{-1}, 2^{-1}, 10^{-1}$ )	0.1028 ( $10^4, 10^3, 2^2, 0.1$ )	<b>0.0978</b> (0.09, $10^3, 2^{-4}, 0.01, 10^{-5}$ )
NO2 (500X 7)	0.0976 ( $10^0, 2^0, 10^{-2}$ )	0.0974 ( $10^{-2}, 2^{-1}, 10^{-1}$ )	<b>0.0888</b> ( $10^3, 10^2, 2^1, 0.01$ )	0.0955 (0.1, $10^4, 2^{-4}, 0.1, 10^{-6}$ )
Hydraulic Actuator (1021 X 5)	<b>0.0117</b> ( $10^1, 2^1, 10^{-3}$ )	0.0124 ( $10^{-2}, 2^{-2}, 10^{-1}$ )	0.0134 ( $10^2, 10^1, 2^1, 0.1$ )	0.0128 (0.1, $10^4, 2^{-2}, 0.1, 10^{-6}$ )
Google (750 X 5)	<b>0.0218</b> ( $10^2, 2^{-5}, 10^{-3}$ )	0.0220 ( $10^{-2}, 2^{-2}, 10^{-1}$ )	0.0338 ( $10^3, 10^{-2}, 2^{-5}, 0.01$ )	0.0219 (0.08, $10^3, 2^{-5}, 0.001, 10^{-5}$ )
IBM (750 X 5)	0.0241 ( $10^2, 2^{-5}, 10^{-2}$ )	<b>0.0241</b> ( $10^{-1}, 2^{-2}, 10^{-1}$ )	0.0317 ( $10^1, 10^2, 2^1, 0.1$ )	0.0267 (0.1, $10^4, 2^{-3}, 0.1, 10^{-5}$ )
Intel (750 X 5)	<b>0.0285</b> ( $10^2, 2^{-5}, 10^{-3}$ )	0.0286 ( $10^{-2}, 2^{-1}, 10^{-1}$ )	0.0317 ( $10^3, 10^0, 2^1, 0.1$ )	0.0291 (0.08, $10^4, 2^{-1}, 0.1, 10^{-5}$ )
Redhat (750 X 5)	<b>0.0248</b> ( $10^1, 2^{-1}, 10^{-3}$ )	0.0263 ( $10^{-1}, 2^{-5}, 10^{-1}$ )	0.0309 ( $10^4, 10^{-1}, 2^1, 0.1$ )	0.0311 (0.08, $10^4, 2^{-5}, 0.1, 10^{-6}$ )
Sunsports (290 X 5)	0.0722 ( $10^1, 2^{-1}, 10^{-2}$ )	0.0717 ( $10^{-1}, 2^0, 10^{-1}$ )	0.0716 ( $10^5, 10^{-3}, 2^1, 0.01$ )	<b>0.0675</b> (0.1, $10^4, 2^{-1}, 0.1, 10^{-6}$ )

Flexible robotic arm (1019 X 9)	<b>0.0157</b> ( $10^2, 2^{-1}, 10^{-2}$ )	0.0262 ( $10^2, 2^0, 10^{-1}$ )	0.05194 ( $10^3, 10^0, 2^1, 0.1$ )	0.0177 ( $0.1, 10^4, 2^{-4}, 0.1, 10^{-6}$ )
Pollution (60 X 15)	0.1108 ( $10^0, 2^{-3}, 10^{-2}$ )	0.1131 ( $10^{-2}, 2^{-5}, 10^{-1}$ )	0.1022 ( $10^3, 10^4, 2^{-1}, 0.01$ )	<b>0.0920</b> ( $0.02, 10^4, 2^{-4}, 0.1, 10^{-5}$ )
Citigroup (750 X 5)	<b>0.0139</b> ( $10^1, 2^{-2}, 10^{-3}$ )	0.0142 ( $10^0, 2^{-2}, 10^{-3}$ )	0.0229 ( $10^5, 10^{-4}, 2^{-5}, 0.1$ )	0.0148 ( $0.05, 10^3, 2^{-2}, 0.1, 10^{-5}$ )

The performance of the proposed methods is graphically demonstrated on servo, pollution and hydraulic actuator data sets for training and testing sets in Fig 4.1- Fig 4.3.

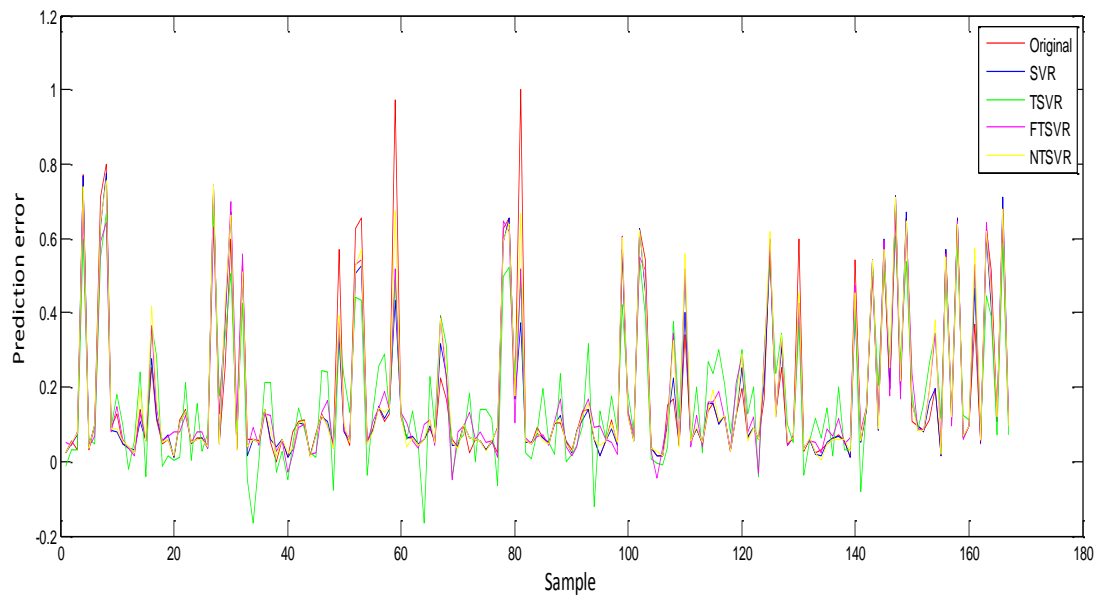


Fig 4.1 Results of comparison on servo dataset

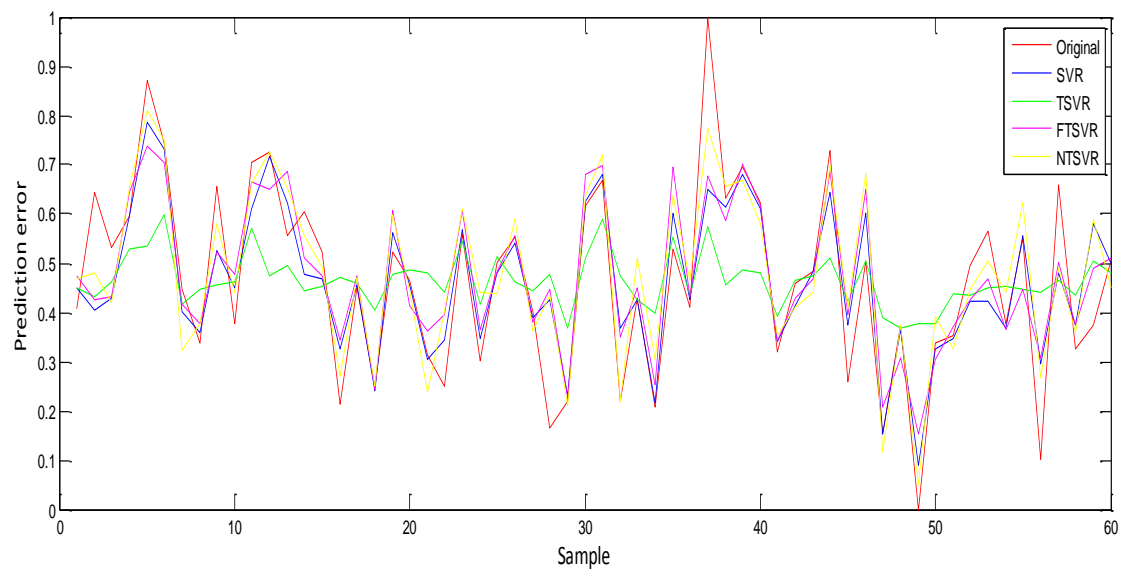


Fig 4.2 Results of comparison on pollution dataset

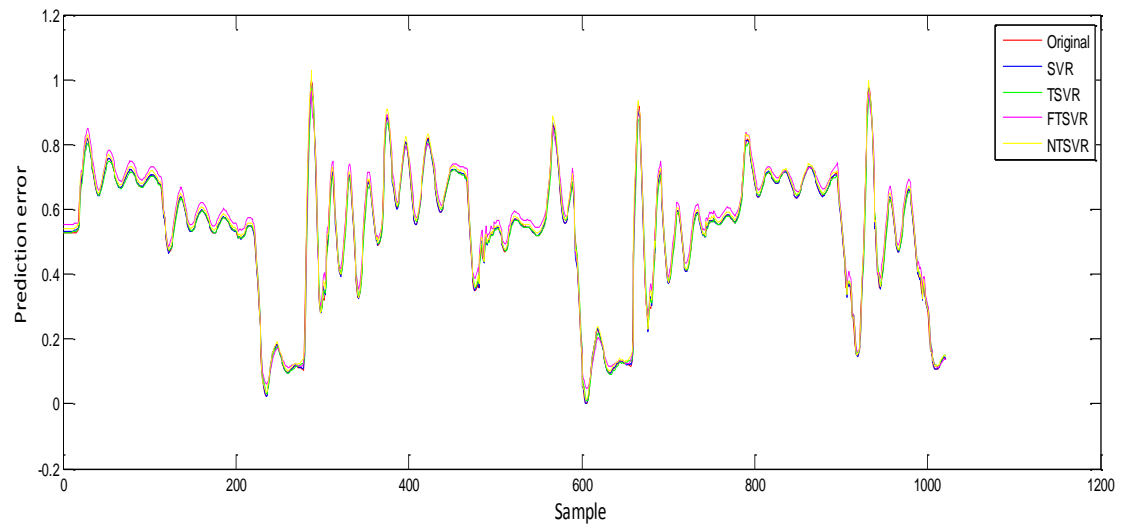


Fig 4.3 Results of comparison on hydraulic actuator dataset



## **Chapter 5**

### **Conclusion**

In this work, a novel regression is considered and further proposed to solve its dual optimization problem by functional iterative method and Newton method. Our proposed methods are implemented in MATLAB and it does not need any special optimization toolbox. Numerical experiments were carried-out and the results of the proposed method were compared with the results obtained using SVR and TSVR. It was observed that the proposed method shows competitive generalization performance from which we can conclude the effectiveness of the proposed methods.

## **APPENDIX 1**

### **MATLAB code for the proposed methods**

#### **For twin parametric insensitive support vector regression via 1-norm by functional iterative method**

```
function [RMSE,ytest0, time]=norm_peng4(train,test,nu,cv,mu,ep)
```

```
[no_input,no_col] = size(train);
```

```
x1 = train(:,1:no_col-1);
```

```
y1 = train(:,no_col);
```

```
[no_test,no_col] = size(test);
```

```
xtest = test(:,1:no_col-1);
```

```
ytest = test(:,no_col);
```

```
xtest0 = xtest;
```

```
mew = mu;
```

```
C1 = cv;
```

```
C2 = C1;
```

```
nu1 = nu * C1;
```

```
nu2 = nu * C2 ;
```

```
ep1 = ep;
```

```
ep2 = ep1;
```

```

no_train=no_input;

tol = 0.0001;

itmax = 30;

tic

K=zeros(no_train,no_train);

for i=1:no_train

    for j=1:no_train

        nom = norm( x1(i,:) - x1(j,:) );

        K(i,j) = exp( -mew * nom * nom ) ;

    end

end

end

[m,n] = size(K)

e = ones(m,1);

em1 = ones(m+1,1);

y = y1;

G = [K e];

GT = G';

GGT = G * GT;

I = speye(m);

invGGT = inv(I+GGT);

ter11 = 0.5 * C1 * e + nu1 *GGT * e - ep1 * y;

ter22 = 0.5 * C2 * e + nu2 *GGT * e + ep2 * y;

```

```

iter = 0;

z1 = zeros(m,1);

z1new = ones(m,1);

while( iter < itmax & norm (z1new - z1) > tol )

    iter = iter + 1;

    z1 = z1new;

    zterm11 = GT * z1 - nu1 * GT * e;

    rhs11 = 0.5 * G * ( abs( -zterm11 - em1 ) - abs( zterm11 - em1 ) );

    rs11 = 0.5 * ( abs(z1) - abs( z1 - C1 * e ) );

    z1new = invGGT * ( rhs11 + rs11 + ter11 );

    norm(z1new-z1)

end

iter

```

```

iter = 0;

z2 = zeros(m,1);

z2new = ones(m,1);

while( iter < itmax & norm (z2new - z2) > tol )

    iter = iter + 1;

    z2 = z2new;

    zterm22 = GT * z2 - nu2 * GT * e;

    rhs22 = 0.5 * G * ( abs( -zterm22 - em1 ) - abs( zterm22 - em1 ) );

    rs22 = 0.5 * ( abs(z2) - abs( z2 - C2 * e ) );

```

```

z2new = invGGT * ( rhs22 + rs22 + ter22 );
norm(z2new-z2)
end
iter

u11 = -GT * z1new + nu1 * GT * e - em1 ;
u12 = GT * z1new - nu1 * GT * e - em1 ;

u1new = ( max( u11, 0 ) - max( u12, 0 ) ) / ep1 ;

u21 = GT * z2new - nu2 * GT * e - em1 ;
u22 = - GT * z2new + nu2 * GT * e - em1 ;

u2new = ( max( u21, 0 ) - max( u22, 0 ) ) / ep2 ;

time=toc;
ktest = zeros( no_test, no_train );
for k=1:no_test
    for i=1:no_train
        nom = norm( x1(i,:) - xtest0(k,:) );
        ktest(k,i) = ktest(k,i) + exp( -mew * nom * nom ) ;
    end
end
e1 = ones(no_test,1);

```

```
ytest0 = [ktest e1] * ( u1new +u2new ) * 0.5;
```

```
RMSE = sqrt( norm(ytest-ytest0)* norm(ytest-ytest0) /no_test )
```

```
end
```

### **For twin parametric insensitive support vector regression via 1-norm using Newton method**

```
function [RMSE,ytest0, time]=norm_pengnew4(train,test,nu,cv,mu,ep,sig)
```

```
[no_input,no_col] = size(train);
```

```
x1 = train(:,1:no_col-1);
```

```
y1 = train(:,no_col);
```

```
[no_test,no_col] = size(test);
```

```
xtest = test(:,1:no_col-1);
```

```
ytest = test(:,no_col);
```

```
xtest0 = xtest;
```

```
mew = mu;
```

```
C1 = cv;
```

```
C2 = C1;
```

```
sigma1=sig;
```

```
nu1 = nu * C1;
```

```
nu2 = nu * C2 ;
```

```
ep1 = ep;
```

```
ep2 = ep1;
```

```

no_train=no_input;
tol = 0.0001;
itmax = 30;

tic

K=zeros(no_train,no_train);
for i=1:no_train
    for j=1:no_train
        nom = norm( x1(i,:) - x1(j,:) );
        K(i,j) = exp( -mew * nom * nom ) ;
    end
end

[m,n] = size(K);
e = ones(m,1);
em1 = ones(m+1,1);
y = y1;

G = [K e];
GT = G';
I = speye(m);

iter = 0;
z1 = ones(m,1);
delphi = ones(m,1) ;
delta = ones(m,1);
del11 = zeros(m+1,1);

```

```

del12 = zeros(m+1,1);
del13 = zeros(m,1);
del14 = zeros(m,1);
hessian = zeros(m,m);

while( iter < itmax & norm (delphi) > tol )
    iter = iter + 1;

    zterm11 = GT * z1 - nu1 * GT * e;
    del11 = max( zterm11 - em1, 0 );
    del12 = max( -zterm11 - em1, 0 );
    del13 = max( z1 - C1 * e, 0 );
    del14 = max( -z1, 0 );

    delphi = ep1 * y + G * ( del11 - del12 ) + del13 - del14;
    hessian = G *( diag( sign(del11) ) + diag( sign(del12) ) ) * GT + diag( sign(del13) ) +
diag( sign(del14) );

    delta = ( hessian + sigma1 * I ) \ delphi ;
    z1 = z1 - delta;
    norm(delphi);
end

iter

iter = 0;
z2 = ones(m,1);
delphi = 1;

```



```

delta = zeros(m,1);
del11 = zeros(m+1,1);
del12 = zeros(m+1,1);
del13 = zeros(m,1);
del14 = zeros(m,1);
hessian = zeros(m,m);

```

```

while( iter < itmax & norm (delphi) > tol )

```

```

    iter = iter + 1;

```

```

    zterm22 = GT * z2 - nu2 * GT * e;

```

```

    del11 = max( zterm22 - em1, 0 );

```

```

    del12 = max( -zterm22 - em1, 0 );

```

```

    del13 = max( z2 - C2 * e, 0 );

```

```

    del14 = max( -z2, 0 );

```

```

    delphi = - ep2 * y + G * ( del11 - del12 ) + del13 - del14;

```

```

    hessian = G *( diag( sign(del11) ) + diag( sign(del12) ) ) * GT + diag( sign(del13) ) +
diag( sign(del14) );

```

```

    delta = ( hessian + sigma1 * I ) \ delphi ;

```

```

    z2 = z2 - delta;

```

```

    norm(delphi);

```

```

end

```

```

iter

```

```

u11 = -GT * z1 + nu1 * GT * e - em1 ;

```

```

u12 = GT * z1 - nu1 * GT * e - em1 ;

u1new = ( max( u11, 0 ) - max( u12, 0 ) ) / ep1 ;

u21 = GT * z2 - nu2 * GT * e - em1 ;
u22 = - GT * z2 + nu2 * GT * e - em1 ;

u2new = ( max( u21, 0 ) - max( u22, 0 ) ) / ep2 ;
time=toc;
ktest = zeros( no_test, no_train );
for k=1:no_test
    for i=1:no_train
        nom = norm( x1(i,:) - xtest0(k,:) );
        ktest(k,i) = ktest(k,i) + exp( -mew * nom * nom ) ;
    end
end
e1 = ones(no_test,1);

ytest0 = [ktest e1] * ( u1new +u2new ) * 0.5;

RMSE = sqrt( norm(ytest-ytest0)* norm(ytest-ytest0) /no_test )
end

```

## References

A. Samola, B. Scholkopf, Learning with Kernal, MIT Press, Cambridge, MA, 2002.

B. Balasundram & Kapil, (2010). On Lagrangian support vector regression, *Expert systems with applications*, 137, 8787-8792.

Bao, Y-K., Liu, Z-T., Guo, L., Wang, W., “Forecasting stock composite index by fuzzy support vector machines regression”, In Proc: *International Conference on Machine Learning and Cybernetics*, 6 (2005), 3535 - 3540.

Brockwell, P.J., Davis, R.A., *Introduction to time series forecasting*, 2nd ed., Springer, Berlin (2002).

Brown, M.P.S., Grundy, W.N., Lin, D., et al., (2000). Knowledge-based analysis of microarray gene expression data by using support vector machine. *Proceedings of National Academy of Science USA*, 97(1), 262-267.

Cortes, C., & Vapnik, V. N., (1995). Support Vector Networks. *Machine Learning*, 20, 273-297.

Cristianini, N., Shawe Taylor, J., *An Introduction to Support Vector Machines and Kernel based Learning Methods*, Cambridge University Press, (2000).

Demiriz, A., Bennett, K., Breneman, C., Embrechts, M., “Support vector machine regression in chemometrics”, *Computing Science & Statistics* (2001).

Ebrahimi, T., Garcia, G. N., & Vesin J. M. (2003). Joint time-frequency-space classification of EEG in a brain-computer interface application. *Journal of Apply Signal Proces*,1(7), 713-729.

F. Facchinei, Minimization of SC functions and the Maratos effect, *Oper. Res. Lett.* 17(1995) 131-137.

Gunn, S., “Support Vector Machines for Classification and Regression”, (1998).

Guyon, I., Weston, J., Barnhill, S., Vapnik, V., “Gene selection for cancer classification using support vector machines”, *Machine Learning*, 46 (2002), 389-422.

Ince, H., & Trafalis, t. b. (2002). Support Vector Machine for Regression and applications to financial forecasting. In *International joint conference on neural networks*. IEEE-INNS-ENNS.

Jayadeva, Khemchandani, R., & Chandra, S. (2007) Twin support vector machine for pattern classification. *IEEE Trans Pattern Anal Mach Intell* 29(5):905-910.

Jochims, T., Nedellec, C., & Rouveriol, C. (1998). Text categorization with support vector machines: Learning with many relative features. *In European conference on machine learning no.10*, Chemnitz, Germany, 1398:137-142.

J.-B. Hiriart-Urruty, J.J. Strodiot, V.H. Nguyen, Generalized Hessian matrix and second order optimality conditions for problems with  $C^{L1}$  data, *Appl. Math. Optim.* 11(1984) 43-56.

Lee, Y.J., Hsieh, W.-F., & Huang, C.-M. (2005).  $\epsilon$ -SSVR: A smooth support vector machine for  $\epsilon$ -insensitive regression, *IEEE Transactions on Knowledge and Data Engineering*, 17(5),678-685.

Malhotra, R., Malhotra, D.K., “Evaluating consumer loans using neural networks”, *Omega* 31 (2003), 83-96.

Mangasarian, O.L., & Musicant, D.R. (1999). Successive Over relaxation for Support Vector Machines, *IEEE transactions on Neural Networks* 10(1999) 1032-1037

Mangasarian, O.L., & Musicant, D.R. (2001). Lagrangian support vector machines, *Journal of Machine Learning Research*, 1, 161-177.

Mangasarian, O.L., & Wild, E.W. (2006). Multisurface proximal support vector classification via generalized eigenvalues, *IEEE Transactions on Pattern Analysis and machine Intelligence* 28(1), pp.69-74.

Mangasarian, O.L., (1994) *Non-linear programming*, SIAM Philadelphia, PA.

Mangasarian, O.L., (2001) A finite Newton method for classification problems, Technical report 01-11, Data Mining Institute, Computer Science Department, University of Wisconsin, Madison, Wisconsin.

Mani, N., Voumard, P., “An optical character recognition using artificial neural network”, *IEEE Conference on Systems, Man and Cybernetics*, 3 (1996), 2244-2247.

Murphy, P.M., & Aha, D.W. (1992). UCI repository of machine learning databases. University of California, Irvine. <http://www.ics.uci.edu/~mllearn>

Musiant & Feinberg, 2004, Active set support vector regression, *IEEE transactions on neural networks*, 15(2).

Osuna, E., Freund, R., & Girosi, F. (1997) Training Support Vector Machines: An application of face detection. *In proceedings of IEEE computer vision and pattern recognition* (pp.130-136).

Peng, X. (2012) Efficient twin parametric insensitive support vector regression model. *Neurocomputing* 79:26-38.

Peng, X. (2010). TSVR: An efficient Twin Support Vector Machine for regression, *Neural Networks*, 23(3), 365-372.

Vapnik, V. N., *Statistical Learning Theory*, John Wiley & Sons, New York, (1998).

Vapnik, V. N., *The Nature of Statistical Learning Theory*. 2<sup>nd</sup> ed., Springer, New York, (2000).