ϵ-SSVR: A Smooth Support Vector Machine for ϵ-Insensitive Regression

Yuh-Jye Lee, Wen-Feng Hsieh, and Chien-Ming Huang

Abstract—A new smoothing strategy for solving ϵ-support vector regression (ϵ-SVR), tolerating a small error in fitting a given data set linearly or nonlinearly, is proposed in this paper. Conventionally, ϵ-SVR is formulated as a constrained minimization problem, namely, a convex quadratic programming problem. We apply the smoothing techniques that have been used for solving the support vector machine for classification, to replace the ϵ-insensitive loss function by an accurate smooth approximation. This will allow us to solve ϵ-SVR as an unconstrained minimization problem directly. We term this reformulated problem as ϵ-smooth support vector regression (ϵ-SSVR). We also prescribe a Newton-Armijo algorithm that has been shown to be convergent globally and quadratically to solve our ϵ-SSVR. In order to handle the case of nonlinear regression with a massive data set, we also introduce the reduced kernel technique in this paper to avoid the computational difficulties in dealing with a huge and fully dense kernel matrix. Numerical results and comparisons are given to demonstrate the effectiveness and speed of the algorithm.

Index Terms—ϵ-insensitive loss function, ϵ-smooth support vector regression, kernel method, Newton-Armijo algorithm, support vector machine.

1 Introduction

In regression problems, we are given a training data set $S = \{(x^1, y_1), \ldots, (x^m, y_m)\} \subseteq R^n \times R$, where $x^i \in R^n$ is the input data and $y_i \in R$ is called the observation. The main goal of regression problems is to find a function $f(x)$ that can correctly predict the observation values, y, of new input data points, x, by *learning* from the given training data set, S. Here, learning from a given training data set means finding a linear or nonlinear surface that tolerates a small error in fitting this training data set. Disregarding the tiny errors that fall within some tolerance, say ϵ, that may lead to a better generalization ability is achieved by utilizing an ϵ-insensitive loss function. Also, applying the idea of support vector machines (SVMs) [2], [10], [30], the function $f(x)$ is made as *flat* as possible in fitting the training data. This problem is called ϵ-support vector regression (ϵ-SVR) and a data point $x^i \in R^n$ is called a support vector if $|f(x^i) - y_i| \geq \epsilon$. Conventionally, ϵ-SVR is formulated as a constrained minimization problem [10], [30], namely, a convex quadratic programming problem or a linear programming problem [14], [24], [23]. Such formulations introduce $2m$ more nonnegative variables and $2m$ inequality constraints that enlarge the problem size and could increase computational complexity for solving the problem. In our approach, we change the model slightly and apply the smooth techniques that have been extensively used for solving important mathematical programming problems [4], [5], [6], [7], [8], [9], [15], [29] and the support vector machine for classification [20] to solve the problem as an unconstrained minimization problem directly. We term this reformulated problem as ϵ-smooth support vector regression (ϵ-SSVR). Because of the infinite order of differentiability of the objective function of our unconstrained minimization problem, we use a fast Newton-Armijo method to solve this reformulation. It has been shown that the sequence generated by the Newton-Armijo algorithm converges to the unique solution globally and quadratically [20]. Taking advantage of ϵ-SSVR formulation, we only need to solve a system of linear equations iteratively instead of solving a convex quadratic program or a linear program, as is the case with a conventional ϵ-SVR. Thus, we do not need to use any sophisticated optimization (linear or nonlinear) package to solve ϵ-SSVR. We begin with the linear case of ϵ-SSVR and then use the nonlinear kernel technique to extend our results to nonlinear ϵ-SSVR. In order to handle the case of nonlinear regression with a massive data set, we also introduce the reduced kernel technique [19] in this paper to avoid the computational difficulties of dealing with a huge and fully dense kernel matrix. To demonstrate ϵ-SSVR's capability in solving linear and nonlinear regression problems, we ran numerical tests on two artificial problems with Gaussian noise and three real-world data sets including the Boston housing problem from the Irvine Machine Learning Database Repository [26]. We also compared our ϵ-SSVR with SVM^{light} [18] and LIBSVM [3], which are state of the art for solving conventional SVM classification as well as ϵ-SVR.

A word about our notation and background material is given below. All vectors will be column vectors through this paper. For a vector x in the n-dimensional real space R^n, the plus function x_+ is defined as $(x_+)_i = \max\{0, x_i\}$, $i = 1, \ldots, n$. The scalar (inner) product of two vectors x and y in the n-dimensional real space R^n will be denoted by $x^T y$ and the p-norm of x will be denoted by $\|x\|_p$. For a matrix $A \in R^{m \times n}$, A_i is the ith row of A which is a row

- *Y.-J. Lee and C.-M. Huang are with the Department of Computer Science & Information Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan 106. E-mail: {yuh-jye, M9215004}@mail.ntust.edu.tw.*
- *W.-F. Hsieh is with the Department of Computer Science & Information Engineering, National Chung Cheng University, Chia-Yi, Taiwan 621. E-mail: hwf90@cs.ccu.edu.tw.*

Manuscript received 13 July 2003; revised 30 June 2004; accepted 28 Oct. 2004; published online 17 Mar. 2005.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0121-0703.

vector in R^n. A column vector of ones of arbitrary dimension will be denoted by $\mathbf{1}$. For $A \in R^{m\times n}$ and $B \in R^{n\times l}$, the kernel $K(A,B)$ maps $R^{m\times n} \times R^{n\times l}$ into $R^{m\times l}$. In particular, if x and y are column vectors in R^n, then $K(x^T, y)$ is a real number, $K(A,x) = K(x^T, A^T)^T$ is a *column* vector in R^m and $K(A, A^T)$ is an $m \times m$ matrix. If f is a real valued function defined on the n-dimensional real space R^n, the gradient of f at x is denoted by $\nabla f(x)$ which is a row vector in R^n and the $n \times n$ Hessian matrix of second partial derivatives of f at x is denoted by $\nabla^2 f(x)$. The base of the natural logarithm will be denoted by e.

2 The Smooth ϵ-Support Vector Regression (ϵ-SSVR)

We consider a given data set S which consists of m points in n-dimensional real space R^n represented by the matrix $A \in R^{m\times n}$ and m observations of real value associated with each point. That is,

$$S = \{(A_i, y_i) | A_i \in R^n, y_i \in R, \ for\ i = 1 \dots m\}.$$

We would like to find a linear or nonlinear regression function, $f(x)$, tolerating a small error in fitting this given data set. This can be achieved by utilizing the ϵ-insensitive loss function that sets an ϵ-insensitive "tube" around the data, within which errors are discarded. Also, applying the idea of support vector machines (SVMs) [2], [30], [10], the function $f(x)$ is made as *flat* as possible in fitting the training data set. We start with the linear case that is the regression function $f(x)$ and it is defined as $f(x) = x^T w + b$. This problem can be formulated as an unconstrained minimization problem given as follows:

$$\min_{(w,b)\in R^{n+1}} \frac{1}{2} w^T w + C\mathbf{1}^T |\xi|_\epsilon, \tag{1}$$

where $|\xi|_\epsilon \in R^m$, $(|\xi|_\epsilon)_i = \max\{0, |A_i w + b - y_i| - \epsilon\}$ that represent the fitting errors and the positive control parameter C here weights the tradeoff between the fitting errors and the flatness of the linear regression function $f(x)$. To deal with the ϵ-insensitive loss function in the objective function of the above minimization problem, conventionally, it is reformulated as a constrained minimization problem defined as follows:

$$\begin{array}{ll} \min\limits_{(w,b,\xi,\xi^*)\in R^{n+1+2m}} & \frac{1}{2} w^T w + C\mathbf{1}^T(\xi + \xi^*) \\ \text{s.t.} & Aw + \mathbf{1}b - y \le \mathbf{1}\epsilon + \xi \\ & -Aw - \mathbf{1}b + y \le \mathbf{1}\epsilon + \xi^* \\ & \xi, \xi^* \ge 0. \end{array} \tag{2}$$

This formulation (2), which is equivalent to the formulation (1), is a convex quadratic minimization problem with $n+1$ free variables, $2m$ nonnegative variables, and $2m$ inequality constraints. However, introducing more variables and constraints in the formulation enlarges the problem size and could increase computational complexity for solving the regression problem.

In our smooth approach, we change the model slightly and solve it as an unconstrained minimization problem directly without adding any new variable and constraint. That is, the squares of 2-norm ϵ-insensitive loss, $|||Aw + \mathbf{1}b - y|_\epsilon||_2^2$, is minimized with weight $\frac{C}{2}$ instead of the 1-norm of ϵ-insensitive loss as in (1). In addition, we add the term $\frac{1}{2}b^2$ in the objective function to induce strong convexity and to guarantee that the problem has a unique global optimal solution. These yield the following unconstrained minimization problem:

$$\min_{(w,b)\in R^{n+1}} \frac{1}{2}(w^T w + b^2) + \frac{C}{2}\sum_{i=1}^{m} |A_i w + b - y_i|_\epsilon^2. \tag{3}$$

This formulation has been proposed in active set support vector regression [27] and solved in its dual form. Inspired by smooth support vector machine for classification (SSVM) [20], the squares of ϵ-insensitive loss function in the above formulation can be accurately approximated by a smooth function which is infinitely differentiable and defined below. Thus, we are allowed to use a fast Newton-Armijo algorithm to solve the approximation problem. Before we derive the smooth approximation function, we show some interesting observations:

$$\begin{array}{rcl} |x|_\epsilon & = & \max\{0, |x| - \epsilon\} \\ & = & \max\{0, x - \epsilon\} + \max\{0, -x - \epsilon\} \\ & = & (x-\epsilon)_+ + (-x-\epsilon)_+. \end{array} \tag{4}$$

Furthermore, $(x-\epsilon)_+ \cdot (-x-\epsilon)_+ = 0$ for all $x \in R$ and $\epsilon > 0$. Thus, we have

$$|x|_\epsilon^2 = (x-\epsilon)_+^2 + (-x-\epsilon)_+^2. \tag{5}$$

In SSVM [20], the plus function x_+ is approximated by a smooth p-function, $p(x,\alpha) = x + \frac{1}{\alpha}\log(1 + e^{-\alpha x})$, $\alpha > 0$. It is straightforward to replace $|x|_\epsilon^2$ by a very accurate smooth approximation given by:

$$p_\epsilon^2(x,\alpha) = (p(x-\epsilon,\alpha))^2 + (p(-x-\epsilon,\alpha))^2. \tag{6}$$

Fig. 1 illustrates the square of ϵ-insensitive loss function and its smooth approximation in the case of $\alpha = 5$ and $\epsilon = 1$.

We call this approximation p_ϵ^2-function with smoothing parameter α. This p_ϵ^2-function is used here to replace the squares of ϵ-insensitive loss function of (3) to obtain our smooth support vector regression (ϵ-SSVR):

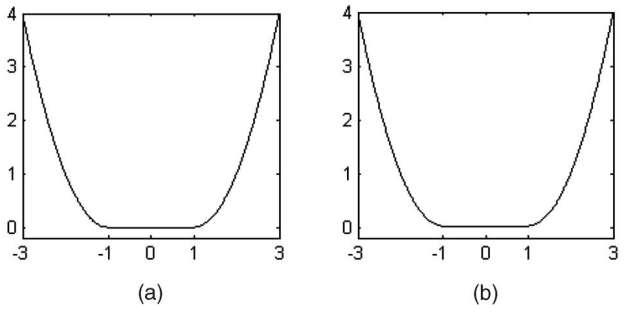

Fig. 1. (a) $|x|_\epsilon^2$ and (b) $p_\epsilon^2(x,\alpha)$ with $\alpha = 5, \epsilon = 1$.

$$\begin{aligned}
&\min_{(w,b)\in R^{n+1}} \quad \Phi_{\epsilon,\alpha}(w,b)\\
:= &\min_{(w,b)\in R^{n+1}} \quad \frac{1}{2}(w^T w + b^2)\\
&\qquad + \frac{C}{2}\sum_{i=1}^{m} p_\epsilon^2(A_i w + b - y_i, \alpha) \qquad (7)\\
= &\min_{(w,b)\in R^{n+1}} \quad \frac{1}{2}(w^T w + b^2)\\
&\qquad + \frac{C}{2}\mathbf{1}^T p_\epsilon^2(Aw + \mathbf{1}b - y, \alpha),
\end{aligned}$$

where $p_\epsilon^2(Aw + \mathbf{1}b - y, \alpha) \in R^m$ is defined by

$$p_\epsilon^2(Aw + \mathbf{1}b - y, \alpha)_i = p_\epsilon^2(A_i w + b - y_i, \alpha).$$

This problem is a strongly convex minimization problem without any constraint. It is easy to show that it has a unique solution. Moreover, the objective function in (7) is infinitely differentiable, thus we can use a fast Newton-Armijo method (only requiring twice differentiability) to solve the problem.

Before we solve the problem (7), we have to show that the solution of the formulation (3) can be obtained by solving problem (7) with α approaching infinity. We begin with a simple lemma that bounds the difference between the squares of ϵ-insensitive loss function, $|x|_\epsilon^2$, and its smooth approximation $p_\epsilon^2(x,\alpha)$.

Lemma 2.1. *For $x \in R$ and $|x| < \rho + \epsilon$:*

$$p_\epsilon^2(x,\alpha) - |x|_\epsilon^2 \le 2\left(\frac{\log 2}{\alpha}\right)^2 + \frac{2\rho}{\alpha}\log 2, \qquad (8)$$

where $p_\epsilon^2(x,\alpha)$ is defined in (6).

Proof. We consider three cases. For $-\epsilon \le x \le \epsilon$, $|x|_\epsilon = 0$ and $p(x,\alpha)^2$ are a monotone increasing function, so we have

$$\begin{aligned}
p_\epsilon^2(x,\alpha) - |x|_\epsilon^2 &= p(x-\epsilon,\alpha)^2 + p(-x-\epsilon,\alpha)^2\\
&\le 2p(0,\alpha)^2 = 2\left(\frac{\log 2}{\alpha}\right)^2,
\end{aligned}$$

since $x - \epsilon \le 0$ and $-x - \epsilon \le 0$.

For $\epsilon < x < \rho + \epsilon$, using the result in SSVM [20] that $p(x,\alpha)^2 - (x_+)^2 \le \left(\frac{\log 2}{\alpha}\right)^2 + \frac{2\rho}{\alpha}\log 2$ for $|x| < \rho$, we have

$$\begin{aligned}
&p_\epsilon^2(x,\alpha) - (|x|_\epsilon)^2\\
=\ &(p(x-\epsilon,\alpha))^2 + (p(-x-\epsilon,\alpha))^2 - (x-\epsilon)_+^2\\
\le\ &(p(x-\epsilon,\alpha))^2 - (x-\epsilon)_+^2 + (p(0,\alpha))^2\\
\le\ &2\left(\frac{\log 2}{\alpha}\right)^2 + \frac{2\rho}{\alpha}\log 2.
\end{aligned}$$

Similarly, for the case of $-\epsilon - \rho < x < -\epsilon$, we have

$$p_\epsilon^2(x,\alpha) - (|x|_\epsilon)^2 \le 2\left(\frac{\log 2}{\alpha}\right)^2 + \frac{2\rho}{\alpha}\log 2.$$

Hence, $p_\epsilon^2(x,\alpha) - |x|_\epsilon^2 \le 2\left(\frac{\log 2}{\alpha}\right)^2 + \frac{2\rho}{\alpha}\log 2$. □

By Lemma 2.1, we have that as the smoothing parameter α approaches infinity, the unique solution of problem (7) approaches the unique solution of problem (3). We shall do this for a function $f_\epsilon(x)$ given in (9) below that subsumes the objective function of (3) and for a function $g_\epsilon(x,\alpha)$ given in (10) below which subsumes the SSVR function of (7).

Theorem 2.2. *Let $A \in R^{m\times n}$ and $b \in R^{m\times 1}$. Define the real valued functions $f_\epsilon(x)$ and $g_\epsilon(x,\alpha)$ in the n-dimensional real space R^n:*

$$f_\epsilon(x) = \frac{1}{2}\sum_{i=1}^{m} |A_i x - b|_\epsilon^2 + \frac{1}{2}\|x\|_2^2 \qquad (9)$$

and

$$g_\epsilon(x,\alpha) = \frac{1}{2}\sum_{i=1}^{m} p_\epsilon^2(A_i x - b, \alpha) + \frac{1}{2}\|x\|_2^2, \qquad (10)$$

with $\epsilon, \alpha > 0$.

1. *There exists a unique solution $\bar{x}$ of $\min_{x\in R^n} f_\epsilon(x)$ and a unique solution $\bar{x}_\alpha$ of $\min_{x\in R^n} g_\epsilon(x,\alpha)$.*
2. *For all $\alpha > 0$, we have the following inequality:*

$$\|\bar{x}_\alpha - \bar{x}\|_2^2 \le m\left(\left(\frac{\log 2}{\alpha}\right)^2 + \xi\frac{\log 2}{\alpha}\right), \qquad (11)$$

where ξ is defined as follows:

$$\xi = \max_{1\le i\le m} |(A\bar{x} - b)_i|. \qquad (12)$$

Thus, $\bar{x}_\alpha$ converges to $\bar{x}$ as α goes to infinity with an upper bound given by (11).

The proof can be adapted from the results in SSVM [20] and, thus, omitted here. We now describe a Newton-Armijo algorithm for solving the smooth problem (7).

3 A Newton-Armijo Algorithm for ϵ-SSVR

By making use of results of the previous section and taking advantage of the twice differentiability of the objective function in (7), we prescribe a globally and quadratically convergent Newton-Armijo algorithm for solving (7).

Algorithm 3.1 Newton-Armijo Algorithm for ϵ-SSVR (7). Start with any choice of initial point $(w^0, b_0) \in R^{n+1}$. Having (w^i, b_i), stop if the gradient of the objective function of (7) is zero, that is, $\nabla\Phi_{\epsilon,\alpha}(w^i, b_i) = 0$. Else compute (w^{i+1}, b_{i+1}) as follows:

1. **Newton Direction**: Determine the direction $d^i \in R^{n+1}$ by setting equal to zero the linearization of $\nabla\Phi_{\epsilon,\alpha}(w,b)$ around (w^i, b_i), which results in $n+1$ linear equations with $n+1$ variables:

$$\nabla^2\Phi_{\epsilon,\alpha}(w^i, b_i)d^i = -\nabla\Phi_{\epsilon,\alpha}(w^i, b_i)^T. \qquad (13)$$

2. **Armijo Stepsize** [1]: Choose a stepsize $\lambda_i \in R$ such that:

$$(w^{i+1}, b_{i+1}) = (w^i, b_i) + \lambda_i d^i, \qquad (14)$$

where $\lambda_i = \max\{1, \frac{1}{2}, \frac{1}{4}, \ldots\}$ such that:

$$\begin{aligned}
&\Phi_{\epsilon,\alpha}(w^i, b_i) - \Phi_{\epsilon,\alpha}((w^i, b_i) + \lambda_i d^i)\\
&\ge -\delta\lambda_i\nabla\Phi_{\epsilon,\alpha}(w^i, b_i)d^i,
\end{aligned} \qquad (15)$$

where $\delta \in (0, \frac{1}{2})$.

Note that a key difference between our smoothing approach and that of the conventional SVR [14], [24], [23]

is that we are solving a linear system of equations (13) here, instead of solving a quadratic program, as is the case with the conventional SVR. Furthermore, Algorithm 3.1 has been shown to converge globally to the unique solution and takes a pure Newton step after a finite number of iterations. This leads to the following quadratic convergence result:

Theorem 3.2. *Let $\{(w^i, b_i)\}$ be a sequence generated by Algorithm 3.1 and $(\bar{w}, \bar{b})$ be the unique solution of problem (7).*

1. *The sequence $\{(w^i, b_i)\}$ converges to the unique solution $(\bar{w}, \bar{b})$ from any initial point (w^0, b_0) in R^{n+1}.*
2. *For any initial point (w^0, b_0), there exists an integer $\bar{i}$ such that the stepsize λ_i of Algorithm 3.1 equals 1 for $i \geq \bar{i}$ and the sequence $\{(w^i, b_i)\}$ converges to $(\bar{w}, \bar{b})$ quadratically.*

Theorem 3.2 can be inferred from [13, pp. 123-125]. Theoretically, we use an Armijo stepsize to guarantee that Algorithm 3.1 is globally convergent. In our numerical tests, we only use Armijo's rule to decide the stepsize when the objective function value did not decrease in two consecutive iterations.

4 ϵ-SSVR with a Nonlinear Kernel

In Section 2, the smooth support vector regression constructed a linear regression function in fitting the given training data points under the criterion that minimizes the squares of the ϵ-insensitive loss function. That is approximating $y \in R^m$ by a linear function of the form:

$$y \approx Aw + \mathbf{1}b, \tag{16}$$

where $w \in R^n$ and $b \in R$ are parameters to be determined by minimizing the objective function in (3). Applying the duality theorem in convex minimization problem [22], [27], w can be represented by $A^T u$ for some $u \in R^m$. Hence, we have

$$y \approx AA^T u + \mathbf{1}b. \tag{17}$$

This motivated the nonlinear support vector regression model. In order to generalize our results from the linear case to nonlinear case, we employ the kernel technique that has been used extensively in kernel-based learning algorithms [2], [10], [30]. We simply replace the AA^T in (17) by a nonlinear kernel matrix $K(A, A^T)$ where $K(A, A^T)_{ij} = K(A_i, A_j^T)$ and $K(x^T, z)$ is a nonlinear kernel function. Using the same loss criterion with the linear case, this will give us the nonlinear support vector regression formulation as follows:

$$\min_{(u,b)\in R^{m+1}} \frac{1}{2}(u^T u + b^2) + \frac{C}{2}\sum_{i=1}^{m} |K(A_i, A^T)u + b - y_i|_\epsilon^2, \tag{18}$$

where $K(A_i, A^T)$ is a kernel map from $R^{1\times n} \times R^{n\times m}$ to $R^{1\times m}$.

We repeat the same arguments as in Section 2 to obtain the ϵ-SSVR with a nonlinear kernel $K(A, A^T)$:

$$\begin{aligned} \min_{(u,b)\in R^{m+1}} \quad & \frac{1}{2}(u^T u + b^2) \\ & + \frac{C}{2}\sum_{i=1}^{m} p_\epsilon^2(K(A_i, A^T)u + b - y_i, \alpha) \\ = \min_{(u,b)\in R^{m+1}} \quad & \frac{1}{2}(u^T u + b^2) \\ & + \frac{C}{2}\mathbf{1}^T p_\epsilon^2(K(A, A^T)u + \mathbf{1}b - y, \alpha), \end{aligned} \tag{19}$$

where $p_\epsilon^2(K(A, A^T)u + \mathbf{1}b - y, \alpha) \in R^m$ defined by

$$p_\epsilon^2(K(A, A^T)u + \mathbf{1}b - y, \alpha)_i = p_\epsilon^2(K(A_i, A^T)u + b - y_i, \alpha).$$

This problem still retains the strong convexity and differentiability properties for any arbitrary kernel. All of the results of the previous sections still hold. Hence, we can apply the Newton-Armijo Algorithm 3.1 directly to solve (19). The solution of this unconstrained minimization problem for u and b leads to the nonlinear regression function:

$$\begin{aligned} f(x) &= K(x^T, A^T)u + b \\ &= u^T K(A, x) + b \\ &= \sum_{i=1}^{m} u_i K(A_i, x) + b. \end{aligned} \tag{20}$$

We note that one may regard the nonlinear regression function (20) as a linear combination of a dictionary of atom functions, $\{1\} \cup \{K(A_i, \cdot)\}_{i=1}^m$ and the coefficients u_is and b are determined by solving (19). Furthermore, like classification problems, using nonlinear kernels in ϵ-SSVR on massive data sets will lead to the computational difficulties in solving the potentially huge unconstrained minimization problem (19). To avoid these difficulties, we introduce the reduced kernel technique that has been used in support vector machine classification for large data set successfully [16], [19]. There is an experimental study on reduced support machines that showed the efficiency of reduced kernel technique [21]. Also, a theoretical study on reduced kernels was proposed recently. It shows that uniform random selection of reduced sets in reduced support vector machines is optimal in terms of many good statistical criteria [17]. Applying this reduced kernel idea to our regression problem, we randomly select a small portion of atom functions to generate the nonlinear regression function which is expressed as the linear combination of these selected atom functions and determined by fitting the entire data set. That is, using only a very small random portion of entire data set to form a reduced set, say $\bar{A}^T$. The size of the reduced set $\bar{A}^T$ is $\bar{m}$ that is much smaller than the original training set size m. We replace A^T in both of the unconstrained minimization problems (20), to cut problem size and computational time, and for the same purposes in generating the nonlinear regression function (20) by the reduced set $\bar{A}^T$. Thus, we have

$$\min_{(\bar{u},b)\in R^{\bar{m}+1}} \frac{1}{2}(\bar{u}^T \bar{u} + b^2) + \frac{C}{2}\mathbf{1}^T p_\epsilon^2(K(A, \bar{A}^T)\bar{u} + \mathbf{1}b - y, \alpha), \tag{21}$$

where $K(A, \bar{A}^T)$ is a reduced kernel map from $R^{m\times n} \times R^{n\times \bar{m}}$ to $R^{m\times \bar{m}}$. This problem also can be solved by the Newton-Armijo Algorithm 3.1 and gives us the nonlinear regression function:

$$\begin{aligned} f(x) &= K(x^T, \bar{A}^T)\bar{u} + b \\ &= \bar{u}^T K(\bar{A}, x) + b \\ &= \sum_{j=1}^{\bar{m}} \bar{u}_j K(\bar{A}_j, x) + b. \end{aligned} \tag{22}$$

Similar to the above arguments, one may regard the nonlinear regression function (22) as a linear combination of a dictionary of atom functions, $\{1\} \cup \{K(\bar{A}_j, \cdot)\}_{j=1}^{\bar{m}}$ and the coefficients $\bar{u_j}$s and b are determined by solving (21). We turn our attention now to numerical testing.

In order to demonstrate the efficiency and speed of our ϵ-smooth support vector regressions we tested our ϵ-SSVR on two artificial data sets and three real data sets and compared the results with the conventional ϵ-SVR, which was solved by LIBSVM [3] and SVM^{light} [18].

We implemented ϵ-SSVR in C and utilized ATLAS library [31] for the matrix operations, including solving a system of linear equations. For the convenient purpose, we combined our C code with MATLAB API [25]. All experiments were run on a personal computer, which consisted of a 2.0 GHz Pentium-4 processor and 512 megabytes of memory. Based on the first order optimality conditions of unconstrained convex minimization problem, our stopping criterion for ϵ-SSVR was satisfied when the 2-norm of gradient of the objective function is less than 10^{-5}. When we computed the gradient and Hessian matrix in (13), we used the limit values of them as the smoothing parameter α went to infinity, respectively. This gave slightly faster computational times but made no difference in the computational results from those obtained with $\alpha = 5$. In all experiments, 2-norm relative error was chosen to evaluate the tolerance between the predicted values and the observations. For an observation vector y and the predicted vector $\hat{y}$, the 2-norm relative error of two vectors y and $\hat{y}$ was defined as follows:

$$\frac{\|y - \hat{y}\|_2}{\|y\|_2}. \tag{23}$$

In order to evaluate how well each method generalized to unseen data, we split the entire data set into two parts, the training set and testing set. The training data was used to generate the regression function that is learning from training data; the testing set, which is not involved in the training procedure, was used to evaluate the prediction ability of the resulting regression function. We also used a *stratification* scheme in splitting the entire data set to keep the "similarity" between training and testing data sets [32]. That is, we tried to make the training set and the testing set have the similar observation distributions. A smaller testing error indicates a better prediction ability. We performed tenfold cross-validation on each data set [28] and reported the average testing error in our numerical results. To generate a highly nonlinear function, a Gaussian kernel was used for all nonlinear numerical tests defined as below:

$$K(A_i, A_j^T) = e^{-\mu\|A_i - A_j\|_2^2}, i, j = 1, 2, 3 \ldots, m. \tag{24}$$

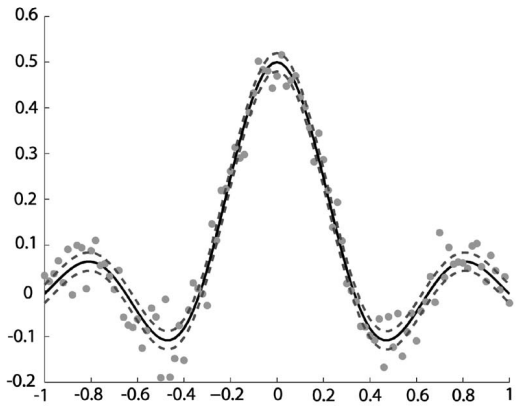

Fig. 2. The ϵ-SSVR took 0.016 seconds to generate the regression function with 0.059 2-norm relative error. LIBSVM took 0.015 seconds and SVM^{light} took 0.090 second with 0.068 2-norm relative error.

The parameters μ and C were determined by a tuning procedure [32]. Because our ϵ-SSVR model is different from the conventional ϵ-SVR, we tuned and selected the parameters for ϵ-SSVR and LIBSVM (SVM^{light}) separately.

We selected 101 points evenly from $[-1, 1]$ as the input data of the first artificial data set and the observation was generated from a simple function in R^1 as follows:

$$f(x) = 0.5 \cdot \frac{\sin(\frac{30}{\pi}x)}{\frac{30}{\pi}x} + \rho, \tag{25}$$

where ρ is an additive Gaussian noise with $\text{mean} = 0$ and standard deviation $\sigma = 0.04$. We set $\epsilon = 0.02$, which is one half standard deviation of the Gaussian noise. The rest of the parameters, $\mu = 33$ and $C = 6$, were determined by a tuning procedure. The ϵ-SSVR has the smallest 2-norm relative error. LIBSVM took 0.015 CPU seconds and SVM^{light} took 0.09 CPU seconds, while ϵ-SSVR took 0.016 seconds. We summarized the results in Fig. 2.

The second artificial data set was obtained by using MATLAB command "peaks(170)" to generate 28,900 data points in R^2. Like our first experiment, the Gaussian noises ($\text{mean} = 0$ and standard deviation $\sigma = 0.4$) were added. Similar to the first experiment, we set $\epsilon = 0.02$, $\mu = 1$, and $C = 1,000$. Because of storing the fully dense kernel matrix required in nonlinear ϵ-SSVR, (19) will exceed the memory capacity and the reduced kernel technique was applied here. We randomly selected 300 points which are slightly over 10 percent of the entire training data set to form a reduced set and used the reduced kernel formulation (21) to generate the nonlinear regression function. The resulting function of ϵ-SSVR (left) and the original function (right, without noises) were shown in Fig. 3. The dots that were shown in Fig. 3 (both sides) form the reduced set $\bar{A}$. This result was generated in 22.578 seconds with 0.016 2-norm relative error. We also tested LIBSVM and SVM^{light} on this artificial data set. Since LIBSVM and SVM^{light} utilize the decomposition strategy to solve the conventional SVM classification as well as regression problems, they do not need to generate the whole kernel matrix in the training process. However, they took a much longer time to get

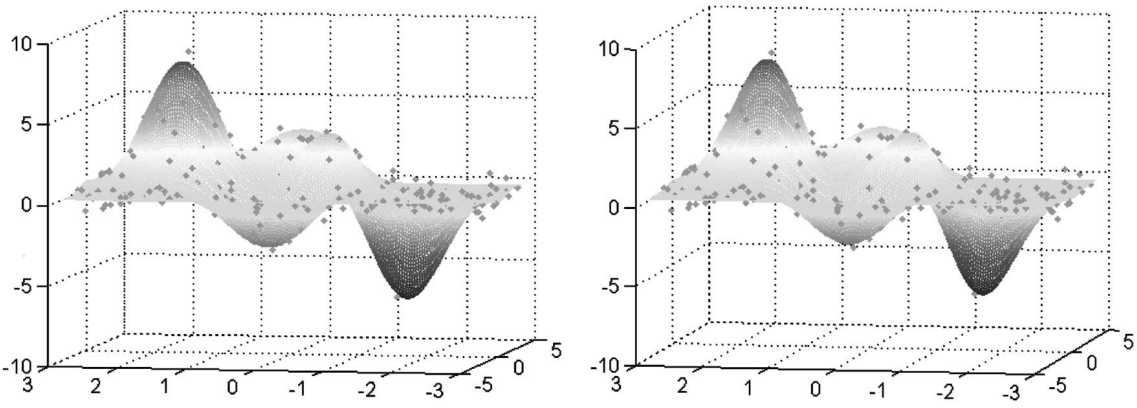

Fig. 3. The ϵ-SSVR (left) took 22.6 seconds to generate the regression function with 0.016 2-norm relative error by using the reduced kernel, $K(A, \bar{A}^T) \in R^{28,900\times 300}$. The original function is shown on the right and the dots in the figure (both sides) formed the reduced set.

the solution with the same level of accuracy. We also summarized the numerical results of these two artificial data sets in Table 2.

We also ran the numerical tests linearly and nonlinearly on three publicly real data sets: Boston Housing, Computer-Activity, and Kin-fh data sets. The first real data set, Boston Housing, is a popular data set for testing regression methods. It contains 506 data points with 12 numeric attributes and one nominal attribute. The predicted value is to determine median home values. This data set is available from UCI [26]. The second one, Comp-Activ, contains 8,192 data points and each data point consists of 25 attributes. We used "cpu prototask" subset containing 21 attributes to decide the percentage of CPU usage in user mode [11]. The third data set, Kin-fh, contains 32 attributes that represent a realistic simulation of the forward dynamics of eight link all-revolute robot arm. The goal is to predict the distance of the end-effector from a target [12]. We tested our linear ϵ-SSVR first. Since we solved the support vector regression problems in the primal space, the computational complexity depends on the number of attributes mainly. The linear ϵ-SSVR will take this advantage for the linear case. Thus, ϵ-SSVR is much faster than LIBSVM and SVM^{light}, while they have almost the same testing errors as well as the number of support vectors. We summarized the tenfold numerical results in Table 1.

For the nonlinear case of experiments, we randomly selected 1,000 data points out of Comp-Activ and Kin-fh, respectively, to form another two data sets. The full Gaussian kernel matrix will be used if the training set size is less than 1,000, otherwise, reduced kernel will be used. Our ϵ-SSVR is slower than LIBSVM and SVM^{light} for the moderate size data sets using a full squared kernel matrix. While for the massive data sets, the reduced kernel technique was applied here to avoid the difficulties in dealing with a huge and dense kernel matrix. We randomly select a small portion 5 percent to 10 percent of data points from the entire data set to form the reduced set in our experiments. In this case, our ϵ-SSVR became the fastest one among LIBSVM and SVM^{light} without scarifying any prediction accuracy. Table 2 summarized the tenfold numerical results and comparisons of nonlinear ϵ-SSVR. The numerical results of two artificial data sets were also included in this table.

TABLE 1
Tenfold Numerical Results of Linear ϵ-SSVR for Three Real World Data Sets

Dataset	Methods	(C, ϵ)	#SVs	Train Error	Test Error	CPU Sec.
Housing	ϵ-SSVR	(16, 0.1)	444	0.1934	0.1851	0.015
Train Size: 456×13	LIBSVM	(16, 0.1)	447	0.2049	0.1855	0.047
Test Size: 50×13	SVM^{light}	(16, 0.1)	447	0.2049	0.1855	0.530
Comp-Activ	ϵ-SSVR	(1, 0.1)	7114	0.0338	0.0347	0.110
Train Size: 7373×21	LIBSVM	(1, 0.1)	7048	0.0350	0.0360	16.656
Test Size: 819×21	SVM^{light}	(1, 0.1)	7047	0.0350	0.0360	17.020
Kin-fh	ϵ-SSVR	(1, 0.1)	5316	0.1343	0.1365	0.188
Train Size: 7373×32	LIBSVM	(1, 0.1)	5323	0.1345	0.1371	39.547
Test Size: 819×32	SVM^{light}	(1, 0.1)	5321	0.1345	0.1370	39.740

ϵ-SSVR, LIBSVM, and SVM^{light} have almost the same 2-norm relative errors as well as the number of support vectors, while ϵ-SSVR is much faster among them. #SVs in the table denotes the number of support vectors.

TABLE 2
Numerical Results of Nonlinear ϵ-SSVR for Three Real World Data Sets and Two Artificial Data Sets

Dataset	Methods	(μ, C, ϵ)	#SVs	Train Error	Test Error	CPU Sec.
Housing	ϵ-SSVR	(1.1, 2750, 0.1)	442	0.0775	0.1171	0.895
Train Size: 456×13	LIBSVM	(0.98, 90, 0.1)	438	0.0942	0.1168	0.400
Test Size: 50×13	SVM^{light}	(0.98, 90, 0.1)	438	0.0942	0.1168	4.908
Comp-Activ	ϵ-SSVR	(0.08, 7250, 0.1)	868	0.0287	0.0300	3.461
Train Size: 900×21	LIBSVM	(0.14, 100, 0.1)	859	0.0293	0.0307	0.502
Test Size: 100×21	SVM^{light}	(0.14, 100, 0.1)	858	0.0293	0.0307	2.881
Comp-Activ	ϵ-SSVR(368)	(0.3, 3800, 0.1)	7121	0.0296	0.0299	8.772
Train Size: 7373×21	LIBSVM	(0.3, 1000, 0.1)	6943	0.0271	0.0280	237.631
Test Size: 819×21	SVM^{light}	(0.3, 1000, 0.1)	6930	0.0271	0.0280	8624.250
Kin-fh	ϵ-SSVR	(0.05, 1, 0.1)	646	0.1308	0.1385	2.523
Train Size: 900×32	LIBSVM	(0.05, 1, 0.1)	645	0.1090	0.1403	0.371
Test Size: 100×32	SVM^{light}	(0.05, 1, 0.1)	646	0.1090	0.1403	0.763
Kin-fh	ϵ-SSVR(368)	(0.04, 101, 0.1)	5322	0.1284	0.1319	8.999
Train Size: 7373×32	LIBSVM	(0.02, 1, 0.1)	5280	0.1276	0.1322	24.692
Test Size: 819×32	SVM^{light}	(0.02, 1, 0.1)	5281	0.1276	0.1322	62.002
2D_Reg	ϵ-SSVR	(33, 6, 0.02)	67	0.0589		0.016
Size: 101×1	LIBSVM	(10, 1, 0.02)	67	0.0681		0.015
	SVM^{light}	(10, 1, 0.02)	66	0.0680		0.090
3D_Reg	ϵ-SSVR(300)	(1, 10000, 0.2)	17824	0.0161		22.578
Size: 28900×2	LIBSVM	(1, 1, 0.2)	17845	0.0196		166.640
	SVM^{light}	(1, 1, 0.2)	17838	0.0196		2537.750

ϵ-SSVR($\bar{m}$) means that we used the reduced kernel technique to solve the problem and $\bar{m}$ denotes the reduced set size.

Our numerical results have demonstrated that ϵ-SSVR is a powerful tool for solving linear and nonlinear regression problems. The reduced kernel technique can be used here to handle the massive data sets without scarifying any prediction accuracy. In the tuning process of these experiments, we found out that LIBSVM and SVM^{light} become very slow when the control parameter C becomes bigger, while ϵ-SSVR is quite robust to the control parameter C. Also, the testing time of LIBSVM and SVM^{light} depend on the number of support vectors. If we applied the reduced kernel technique to ϵ-SSVR, the testing time will depend on the size of the reduced set only which is always much smaller than the number of support vectors. There is an interesting observation of the number of support vectors. Although we solve the linear and nonlinear ϵ-insensitive regression problems as a unconstrained minimization problem, the number of support vectors is very close to the number of support vectors in LIBSVM and SVM^{light}. We still have the same observation when we use the reduced kernel technique to deal with the massive data sets. We conclude our paper now.

5 Conclusion

We have proposed a new formulation, ϵ-SSVR, which is a smooth unconstrained optimization reformulation of the traditional quadratic program associated with an ϵ insensitive support vector regression. We have prescribed a very fast Newton-Armijo algorithm to solve the ϵ-SSVR that has been shown convergent globally and quadratically to the solution. The numerical results show the effectiveness and correctness of ϵ-SSVR. For the linear case, ϵ-SSVR is much faster than the conventional SVR implemented by LIBSVM and SVM^{light} while with comparable correctness. We also extended ϵ-SSVR to the nonlinear data fitting problem by using kernel techniques. In dealing with a massive data set for nonlinear regression problems, the reduced kernel technique is also introduced in this paper. A nice feature of our ϵ-SSVR formulation is that we only need to solve a system of linear equations iteratively instead of solving a convex quadratic program, as is the case with a conventional ϵ-SVR.

Acknowledgments

This research is supported by the National Science Council of Taiwan via the grant NSC 90-2218-E-194-004. The authors would like to thank Dr. O.L. Mangasarian, Dr. D.R. Musicant, and Dr. S.-Y. Huang for their very helpful comments.

References

[1] D.P. Bertsekas, *Nonlinear Programming*. Belmont, Mass.: Athena Scientific, 1995.
[2] C.J.C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121-167, 1998.
[3] C.-C. Chang and C.-J. Lin, *LIBSVM*: A Library for Support Vector Machines, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[4] B. Chen and P.T. Harker, "Smooth Approximations to Nonlinear Complementarity Problems," *SIAM J. Optimization,* vol. 7, pp. 403-420, 1997.

[5] C. Chen and O.L. Mangasarian, "Smoothing Methods for Convex Inequalities and Linear Complementarity Problems," *Math. Programming,* vol. 71, no. 1, pp. 51-69, 1995.

[6] C. Chen and O.L. Mangasarian, "A Class of Smoothing Functions for Nonlinear and Mixed Complementarity Problems," *Computational Optimization and Applications,* vol. 5, no. 2, pp. 97-138, 1996.

[7] X. Chen, L. Qi, and D. Sun, "Global and Superlinear Convergence of the Smoothing Newton Method and Its Application to General Box Constrained Variational Inequalities," *Math. of Computation,* vol. 67, pp. 519-540, 1998.

[8] X. Chen and Y. Ye, "On Homotopy-Smoothing Methods for Variational Inequalities," *SIAM J. Control and Optimization,* vol. 37, pp. 589-616, 1999.

[9] P.W. Christensen and J.-S. Pang, "Frictional Contact Algorithms Based on Semismooth Newton Methods," *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods,* M. Fukushima and L. Qi, eds., pp. 81-116, Kluwer Academic Publishers, 1999.

[10] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines.* Cambridge: Cambridge Univ. Press, 2000.

[11] DELVE, Data for Evaluating Learning in Valid Experiments, Comp-Activ Dataset, http://www.cs.toronto.edu/~delve/data/comp-activ/desc.html, 2005.

[12] DELVE, Data for Evaluating Learning in Valid Experiments, Kin-family Dataset, http://www.cs.toronto.edu/~delve/data/kin/desc.html, 2005.

[13] J.E. Dennis and R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Englewood Cliffs, N.J.: Prentice-Hall, 1983.

[14] H. Drucker, C.J.C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support Vector Regression Machines," *Advances in Neural Information Processing Systems -9-,* M.C. Mozer, M.I. Jordan, and T. Petsche, eds., pp. 155-161, Cambridge, Mass.: MIT Press, 1997.

[15] M. Fukushima and L. Qi, *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth, and Smoothing Methods.* Dordrecht, The Netherlands: Kluwer Academic Publishers, 1999.

[16] G. Fung and O.L. Mangasarian, "Proximal Support Vector Machine Classifiers," *Proc. KDD-2001: Knowledge Discovery and Data Mining,* F. Provost and R. Srikant, eds., pp. 77-86, 2001, ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/01-02.ps.

[17] S.-Y. Huang and Y.-J. Lee, "Reduced Support Vector Machines: A Statistical Theory," Preprint, Inst. of Statistical Science, Academia Sinica, 2004, http://www.stat.sinica.edu.tw/syhuang/.

[18] T. Joachims, SVM^{light}, 2002, http://svmlight.joachims.org.

[19] Y.-J. Lee and O.L. Mangasarian, "RSVM: Reduced Support Vector Machines," Technical Report 00-07, Data Mining Inst., Computer Sciences Dept., Univ. of Wisconsin, Madison, Wisconsin, July 2000, also *Proc. First SIAM Int'l Conf. Data Mining,* 2001, ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/00-07.ps.

[20] Y.-J. Lee and O.L. Mangasarian, "SSVM: A Smooth Support Vector Machine," *Computational Optimization and Applications,* vol. 20, pp. 5-22, 2001, also Data Mining Inst., Univ. of Wisconsin, Technical Report 99-03, ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/99-03.ps.

[21] K.-M. Lin and C.-J. Lin, "A Study on Reduced Support Vector Machines," *IEEE Trans. Neural Networks,* vol. 14, no. 6, pp. 1449-1459, 2003.

[22] O.L. Mangasarian, "Generalized Support Vector Machines," *Advances in Large Margin Classifiers,* A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, eds., pp. 135-146, Cambridge, Mass: MIT Press, 2000, ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-14.ps.

[23] O.L. Mangasarian and D.R. Musicant, "Robust Linear and Support Vector Regression," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 9, pp. 950-955, 2000, ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/99-09.ps.

[24] O.L. Mangasarian and D.R. Musicant, "Large Scale Kernel Regression via Linear Programming," *Machine Learning,* vol. 46, pp. 255-269, 2002, ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/99-02.ps.

[25] MATLAB, *User's Guide.* Natick, Mass.: The MathWorks, Inc., 1994-2001, http://www.mathworks.com.

[26] C.L. Blake and C.J. Merz UCI Repository of Machine Learning Databases, 1998, http://www.ics.uci.edu/~mlearn/MLRepository.htm.

[27] D.R. Musicant and A. Feinberg, "Active Set Support Vector Regression," *IEEE Trans. Neural Networks,* vol. 15, no. 2, pp. 268-275, 2004.

[28] M. Stone, "Cross-Validatory Choice and Assessment of Statistical Predictions," *J. Royal Statistical Soc.,* vol. 36, pp. 111-147, 1974.

[29] P. Tseng, "Analysis of a Non-Interior Continuation Method Based on Chen-Mangasarian Smoothing Functions for Complementarity Problems," *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth, and Smoothing Methods,* M. Fukushima and L. Qi, eds., pp. 381-404, Dordrecht, Netherlands: Kluwer Academic Publishers, 1999.

[30] V.N. Vapnik, *The Nature of Statistical Learning Theory.* New York: Springer, 1995.

[31] R.C. Whaley, A. Petitet, and J.J. Dongarra, "Automated Empirical Optimization of Software and the ATLAS Project," *Parallel Computing,* vol. 27, nos. 1-2, pp. 3-35, 2001, also available as Univ. of Tennessee LAPACK Working Note #147, UT-CS-00-448, www.netlib.org/lapack/lawns/lawn147.ps, 2000.

[32] I.H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* San Francisco: Morgan Kaufmann, 1999.

Yuh-Jye Lee received the master's degree in applied mathematics from the National Tsing Hua University, Taiwan, and the PhD degree in computer sciences from the University of Wisconsin-Madison in 2001. Since 2002, he has been an assistant professor at the National Taiwan University of Science and Technology. His research interests are in machine learning, data mining, and optimization.

Wen-Feng Hsieh received the BS degree and MS degree in 2001 and 2003, respectively, in computer science and information engineering, from the National Chung-Cheng University. His research interests are in data mining and support vector machines.

Chien-Ming Huang received the BS degree in computer science and information engineering from Yuan-Ze University. Now, he is a graduate student studying computer science and information engineering at the National Taiwan University of Science and Technology. His research interests are in data mining, machine learning, and programming language.