Two-Dimensional Solution Path for Support Vector Regression

Gang Wang WANGGANG@CS.UST.HK
Dit-Yan Yeung DYYEUNG@CS.UST.HK
Frederick H. Lochovsky FRED@CS.UST.HK
Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, China

Abstract

Recently, a very appealing approach was proposed to compute the entire solution path for support vector classification (SVC) with very low extra computational cost. This approach was later extended to a support vector regression (SVR) model called ϵ-SVR. However, the method requires that the error parameter ϵ be set *a priori*, which is only possible if the desired accuracy of the approximation can be specified in advance. In this paper, we show that the solution path for ϵ-SVR is also piecewise linear with respect to ϵ. We further propose an efficient algorithm for exploring the two-dimensional solution space defined by the regularization and error parameters. As opposed to the algorithm for SVC, our proposed algorithm for ϵ-SVR initializes the number of support vectors to zero and then increases it gradually as the algorithm proceeds. As such, a good regression function possessing the sparseness property can be obtained after only a few iterations.

1. Introduction

In a typical regression problem, we are given a training set of independent and identically distributed (iid) data pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^n \subset \mathbb{R}^d \times \mathbb{R}$, where $\mathbf{x}_i$ and y_i are the input and output, respectively, of the ith pair. Support vector regression (SVR) (Vapnik, 1995; Schölkopf & Smola, 2002) is a kernel method that performs nonlinear regression based on the kernel trick. Essentially, each input $\mathbf{x}_i \in \mathbb{R}^d$ is mapped implicitly via a nonlinear feature map $\phi(\cdot)$ to some kernel-induced feature space $\mathcal{F}$ where linear regression is performed. Specifically, SVR learns the following regression function by estimating $\mathbf{w} \in \mathcal{F}$ and $w_0 \in \mathbb{R}$ from the training data:

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + w_0, \tag{1}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. It does so by minimizing some empirical risk measure that is regularized to control the capacity.

In ϵ-SVR, the so-called ϵ-insensitive loss function $|y - f(\mathbf{x})|_\epsilon = \max\{0, |y - f(\mathbf{x})| - \epsilon\}$ is used to define an empirical risk functional which exhibits the same sparseness property of the support vectors as that for support vector classifiers (SVC) using the hinge loss function. If a point $\mathbf{x}$ lies inside the insensitive zone called the ϵ-tube, i.e., $|y - f(\mathbf{x})| \leq \epsilon$, then it will not incur any loss. However, the error parameter $\epsilon \geq 0$ has to be chosen *a priori* by the user. The primal optimization problem for ϵ-SVR can be stated as follows:

$$\min_{\mathbf{w}, \xi^{(*)}} \quad \frac{\lambda}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{2}$$

$$\text{subject to} \quad y_i - (\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + w_0) \leq \epsilon + \xi_i \tag{3}$$

$$(\langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + w_0) - y_i \leq \epsilon + \xi_i^* \tag{4}$$

$$\xi_i^{(*)} \geq 0. \tag{5}$$

Here and below, $i = 1, ..., n$ and $^{(*)}$ denote both the variables with and without asterisks. The parameter $\lambda > 0$ is a regularization parameter that controls the degree of regularization. Like ϵ, λ also has to be chosen in advance by the user. The two parameters λ and ϵ play different roles in ϵ-SVR. Figure 1 shows four different combinations of the parameter values.

In practice, users often use some default values for λ and ϵ even though they are by no means optimal choices. Extensive exploration of the optimal parameter values is seldom pursued since this requires retraining the model many times under different parameter settings. To overcome the difficulty of selecting ϵ, Schölkopf et al. (2000) proposed the ν-SVR model which automatically adjusts the width of the tube so that at most a fraction ν of the data points lie outside the tube.

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

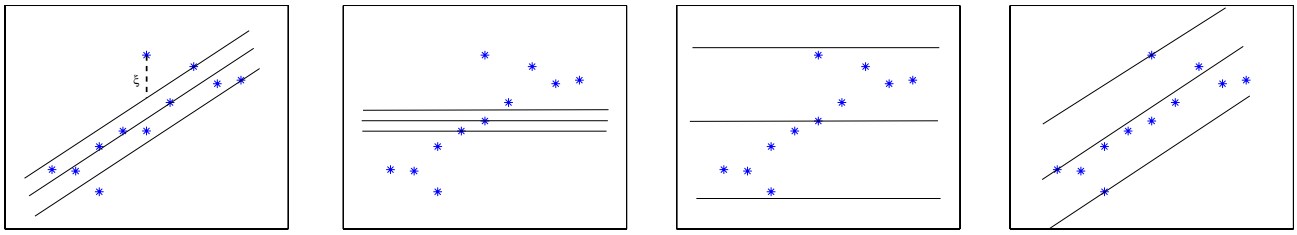


Figure 1. Linear SVR results for four different combinations of values for λ and ϵ. (a) proper values of λ and ϵ are specified; (b) $\lambda = \infty$; (c) $\epsilon > (y_{max} - y_{min})/2$; (d) $\epsilon < (y_{max} - y_{min})/2$, but all the data points are inside the ϵ-tube.

More recently, a novel approach has emerged that seeks to explore the entire solution path for all parameter values without having to re-train the model multiple times. By estimating the generalization errors under different parameter values, the optimal combination can be found with a low extra computational cost. Efron et al. (2002) developed the Least Angle Regression (LARS) algorithm which fits the coefficient path for the linear least square regression problem regularized with the L_1 norm. This is probably the first work that explores the correspondence between every regularization parameter value and the solution. An important finding is that the coefficient path is piecewise linear and hence it is efficient to explore the entire solution path by monitoring the breakpoints only. Inspired by this pioneering work, Rosset et al. (2004) showed that boosting approximately follows the regularization path with an appropriate loss and an L_1 penalty. Zhu et al. (2003) proposed an algorithm to compute the entire regularization path for the L_1-norm SVC and Hastie et al. (2004) proposed one for the standard L_2-norm SVC. They are again based on the property that the regularization paths are piecewise linear. More generally, Rosset and Zhu (2003) showed that any model with an L_1 regularization and a quadratic, piecewise quadratic, piecewise linear, or linear loss function has a piecewise linear coefficient path and hence the entire solution path can be computed efficiently.

For SVR, Gunter and Zhu (2005) derived an algorithm that computes the entire solution path for ϵ-SVR with respect to λ when ϵ is fixed. The algorithm starts from $\lambda = \infty$, with the initial solution obtained by solving a linear programming problem. As λ decreases, the algorithm computes the solution for every value of λ. However, sometimes the solution path cannot be traced successfully when some points lie at the boundaries of the ϵ-tube. When there exists only one or even no point at the boundaries, the tube has to move and rotate until two valid points enter the tube. Since the search strategy for valid points is random, it is impossible to estimate the number of steps before two such points are encountered. Therefore, their algorithm for exploring the λ-path cannot be realized easily in practice. In this paper, we show that the solution path for ϵ-SVR is also piecewise linear with respect to ϵ, although ϵ is not a regularization parameter like λ. We refer to this path as the ϵ-path. It is important to note that the ϵ-path algorithm, unlike the λ-path algorithm, can always proceed without any difficulty. By integrating the λ-path and ϵ-path algorithms, we propose an efficient algorithm to explore the two-dimensional solution space for ϵ-SVR. The width of the tube decreases from infinity as the algorithm proceeds. Thus, the points originally lying inside the tube will move outside and the Lagrange multipliers will change from 0 to 1. As such, similar to ν-SVR, the number of support vectors can be controlled exactly. A good regression function with the desirable sparseness property can be obtained after only a small number of iterations to decrease ϵ.

The rest of this paper is organized as follows. In Section 2, we review the λ-path algorithm and devise the ϵ-path algorithm. In Section 3, the two algorithms are integrated to give a new algorithm for exploring the two-dimensional solution space for ϵ-SVR efficiently. Some experimental results are then presented in Section 4, followed by a conclusion of the paper in the last section.

2. One-Dimensional Solution Paths

2.1. Problem Setup

From the primal optimization problem for ϵ-SVR, we can obtain

$$\mathbf{w} = \frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\phi(\mathbf{x}_i). \tag{6}$$

Applying the method of Lagrange multipliers to the primal optimization problem, we arrive at the following dual optimization problem:

$$\max_{\boldsymbol{\alpha}^{(*)}} \quad -\frac{1}{2\lambda}\sum_{i,j=1}^{n}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)K(\mathbf{x}_i,\mathbf{x}_j)$$
$$-\epsilon\sum_{i=1}^{n}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)y_i \qquad (7)$$
$$\text{subject to} \quad 0\le\alpha_i^{(*)}\le 1 \qquad (8)$$
$$\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)=0. \qquad (9)$$

Using (6), the regression function in (1) can be rewritten as:

$$\begin{aligned} f(\mathbf{x}) &= \langle\mathbf{w},\phi(\mathbf{x})\rangle + w_0 \\ &= \frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)\langle\phi(\mathbf{x}_i),\phi(\mathbf{x})\rangle + w_0 \\ &= \frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)K(\mathbf{x}_i,\mathbf{x}) + w_0. \qquad (10)\end{aligned}$$

From the Karush-Kuhn-Tucker (KKT) conditions, we can derive the following relationships:

$$\begin{array}{lcll} y_i-f(\mathbf{x}_i)>\epsilon & \Rightarrow & \alpha_i=1, & \alpha_i^*=0 \\ y_i-f(\mathbf{x}_i)=\epsilon & \Rightarrow & \alpha_i\in[0,1], & \alpha_i^*=0 \\ y_i-f(\mathbf{x}_i)\in(-\epsilon,\epsilon) & \Rightarrow & \alpha_i=0, & \alpha_i^*=0 \\ y_i-f(\mathbf{x}_i)=-\epsilon & \Rightarrow & \alpha_i=0, & \alpha_i^*\in[0,1] \\ y_i-f(\mathbf{x}_i)<-\epsilon & \Rightarrow & \alpha_i=0, & \alpha_i^*=1 \end{array}$$

As a consequence, $f(\mathbf{x})$ can be expressed as an expansion in terms of only a subset of data points for which either α_i or α_i^* is nonzero. These points are referred to as support vectors which, like those for SVC, contribute to the sparseness property of $f(\mathbf{x})$ with representational and computational advantages. Following the convention of (Gunter & Zhu, 2005), we partition the set of data points into the following five subsets as illustrated in Figure 2:

- $\mathcal{R}=\{i: y_i-f(\mathbf{x}_i)>\epsilon, \alpha_i=1, \alpha_i^*=0\}$
- $\mathcal{E}_\mathcal{R}=\{i: y_i-f(\mathbf{x}_i)=\epsilon, \alpha_i\in[0,1], \alpha_i^*=0\}$
- $\mathcal{C}=\{i: |y_i-f(\mathbf{x}_i)|<\epsilon, \alpha_i=0, \alpha_i^*=0\}$
- $\mathcal{E}_\mathcal{L}=\{i: y_i-f(\mathbf{x}_i)=-\epsilon, \alpha_i=0, \alpha_i^*\in[0,1]\}$
- $\mathcal{L}=\{i: y_i-f(\mathbf{x}_i)<-\epsilon, \alpha_i=0, \alpha_i^*=1\}$

As we change the value of λ or ϵ, the tube will move, rotate, shrink, expand or remain unchanged. Some events may occur during this process. An event is said to occur when a point enters or leaves an elbow, causing some point sets to change. We categorize these events as follows:

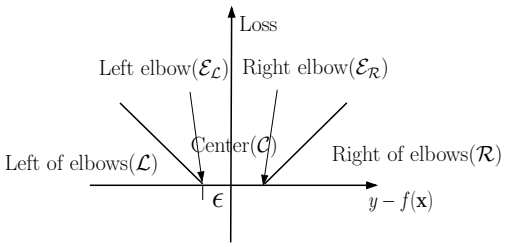


Figure 2. The set of data points is partitioned into five subsets according to the ϵ-insensitive loss function.

1. A point enters an elbow:
   - From $\mathcal{C}$ to $\mathcal{E}_\mathcal{R}$ with $\alpha_i=0$
   - From $\mathcal{C}$ to $\mathcal{E}_\mathcal{L}$ with $\alpha_i^*=0$
   - From $\mathcal{R}$ to $\mathcal{E}_\mathcal{R}$ with $\alpha_i=1$
   - From $\mathcal{L}$ to $\mathcal{E}_\mathcal{L}$ with $\alpha_i^*=1$
2. A point leaves an elbow:
   - From $\mathcal{E}_\mathcal{R}$ to $\mathcal{R}$ with $\alpha_i=1$
   - From $\mathcal{E}_\mathcal{R}$ to $\mathcal{C}$ with $\alpha_i=0$
   - From $\mathcal{E}_\mathcal{L}$ to $\mathcal{C}$ with $\alpha_i^*=0$
   - From $\mathcal{E}_\mathcal{L}$ to $\mathcal{L}$ with $\alpha_i^*=1$

For the points inside or outside the tube, i.e., in $\mathcal{R}\cup\mathcal{C}\cup\mathcal{L}$, their $\alpha_i^{(*)}$ values remain fixed until an event occurs. Hence, it suffices to focus on the points at the elbows, i.e., in $\mathcal{E}_\mathcal{R}\cup\mathcal{E}_\mathcal{L}$. As a point passes through $\mathcal{E}_\mathcal{R}$ or $\mathcal{E}_\mathcal{L}$, its α_i or α_i^* value will change from 0 to 1 or from 1 to 0.

2.2. λ-Path

The λ-path algorithm explores the correspondence between every λ value and the corresponding solution $\alpha_i^{(*)}(\lambda)$ for a fixed ϵ. The path can start from the solution of an ϵ-SVR model for any initial value of λ, since the values of $\alpha_i^{(*)}$ fully determine the sets $\mathcal{R}$, $\mathcal{E}_\mathcal{R}$, $\mathcal{C}$, $\mathcal{E}_\mathcal{L}$ and $\mathcal{L}$. However, finding the solution requires solving a quadratic programming problem (Schölkopf & Smola, 2002) which is computationally demanding. The problem to solve becomes simpler if we set $\lambda=\infty$ initially. Doing so will make the first term of the objective function (7) vanish, leaving only the last two terms. Thus the quadratic programming problem degenerates to a linear programming problem which is easier to solve. Similarly, the first term of (10) vanishes so that the regression function becomes $f(\mathbf{x})=w_0$, which corresponds to the case shown in Figure 1(b). The initial values of $\alpha_i^{(*)}$, denoted as $\alpha_i^{(*)0}$, are either 0 or 1 if all the values of y_i are distinct (Gunter & Zhu, 2005). The ϵ-tube is bounded by the sets $\mathcal{R}$, $\mathcal{C}$ and $\mathcal{L}$. The tube can move around by changing λ and w_0, while no point is allowed to pass through any elbow. Hence the

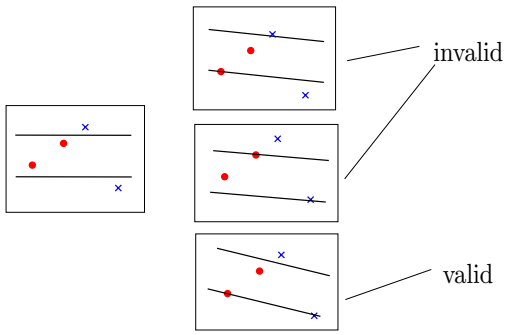


Figure 3. The tube rotates in a clockwise direction as λ decreases. The one on the left is the initial case. Shown on the right are three possible rotation results where two points enter the elbows together. Only the last one is a valid case.

following constraints are satisfied:

$$y_j - \frac{1}{\lambda}\sum_i (\alpha_i^0 - \alpha_i^{*0})K(\mathbf{x}_i, \mathbf{x}_j) - w_0 > \epsilon \text{ for } j \in \mathcal{R} \quad (11)$$

$$\left| y_j - \frac{1}{\lambda}\sum_i (\alpha_i^0 - \alpha_i^{*0})K(\mathbf{x}_i, \mathbf{x}_j) - w_0 \right| < \epsilon \text{ for } j \in \mathcal{C} \quad (12)$$

$$y_j - \frac{1}{\lambda}\sum_i (\alpha_i^0 - \alpha_i^{*0})K(\mathbf{x}_i, \mathbf{x}_j) - w_0 < -\epsilon \text{ for } j \in \mathcal{L} \quad (13)$$

In order for some parameters in $\boldsymbol{\alpha}^{(*)}$ to change with respect to λ, at least two points should be at the elbows. Gunter and Zhu (2005) proposed a strategy for finding a feasible (λ, w_0) combination so that two points enter the elbows simultaneously. It first moves the tube until a point enters one elbow through changing w_0. Then it decreases λ until another point also enters an elbow. Given the sets $\mathcal{R}$, $\mathcal{C}$ and $\mathcal{L}$, in fact there exist many possible combinations of two points that can enter the elbows simultaneously. However, some combinations are invalid because the λ-path algorithm focusing on these points cannot proceed further. This can be explained by considering the equality constraint in (9). If two points are at the elbows, then their corresponding parameters in $\boldsymbol{\alpha}^{(*)}$ have to increase or decrease together. For example, if a point in $\mathcal{R}$ or $\mathcal{L}$ wants to enter one elbow and another point in $\mathcal{C}$ wants to enter another elbow simultaneously, $\boldsymbol{\alpha}^{(*)}$ cannot change to satisfy (9) due to the constraints (8). Figure 3 depicts some possible valid and invalid cases. Since the algorithm moves the tube randomly to search for two valid points, it is hard to implement the program and to estimate the computational requirement of the algorithm. In fact, whenever the elbows contain less than two points, the algorithm needs to perform such a random search, making it unappealing in practice.

For convenience, we define $\alpha_0 = \lambda w_0$ and rewrite the regression function in (10) as

$$f(\mathbf{x}) = \frac{1}{\lambda}\Big[\sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}) + \alpha_0\Big]. \quad (14)$$

Let $\alpha_i^{(*)l}$, α_0^l and λ^l denote the parameter values right after the lth event has occurred and $f^l(\mathbf{x})$ is the regression function at this point. As argued above, there should be at least two points at the elbows, i.e., $|\mathcal{E}_{\mathcal{R}}^l| + |\mathcal{E}_{\mathcal{L}}^l| = p \geq 2$.[1]

For λ values such that $\lambda^{l+1} < \lambda < \lambda^l$, the regression function can be expressed as

$$f(\mathbf{x}) = \frac{1}{\lambda}\Big[\sum_{i\in\mathcal{E}_{\mathcal{R}}^l}(\alpha_i - \alpha_i^l)K(\mathbf{x}_i, \mathbf{x}) - \sum_{j\in\mathcal{E}_{\mathcal{L}}^l}(\alpha_j^* - \alpha_j^{*l})K(\mathbf{x}_j, \mathbf{x}) + (\alpha_0 - \alpha_0^l)\Big] + \frac{\lambda^l}{\lambda}f^l(\mathbf{x}). \quad (15)$$

This is because only those points $i \in \mathcal{E}_{\mathcal{R}}^l \cup \mathcal{E}_{\mathcal{L}}^l$ can change their $\alpha_i^{(*)}$ values with λ, while all other points $i \in \mathcal{R}^l \cup \mathcal{C}^l \cup \mathcal{L}^l$ have their $\alpha_i^{(*)}$ values fixed at 0 or 1. For notational simplicity, let us denote $v_i = \alpha_i - \alpha_i^l$, $v_j^* = \alpha_j^* - \alpha_j^{*l}$, and $v_0 = \alpha_0 - \alpha_0^l$. For the points lingering at the elbows, from (15), we have

$$\begin{aligned} &\sum_{i\in\mathcal{E}_{\mathcal{R}}^l} v_i K(\mathbf{x}_i, \mathbf{x}_k) - \sum_{j\in\mathcal{E}_{\mathcal{L}}^l} v_j^* K(\mathbf{x}_j, \mathbf{x}_k) + v_0 \\ = \;& (\lambda - \lambda^l)(y_k - \epsilon), \quad \forall k \in \mathcal{E}_{\mathcal{R}}^l, \quad (16) \\ &\sum_{i\in\mathcal{E}_{\mathcal{R}}^l} v_i K(\mathbf{x}_i, \mathbf{x}_m) - \sum_{j\in\mathcal{E}_{\mathcal{L}}^l} v_j^* K(\mathbf{x}_j, \mathbf{x}_m) + v_0 \\ = \;& (\lambda - \lambda^l)(y_m + \epsilon), \quad \forall m \in \mathcal{E}_{\mathcal{L}}^l. \quad (17) \end{aligned}$$

From (9), we also have

$$\sum_{i\in\mathcal{E}_{\mathcal{R}}^l} v_i - \sum_{j\in\mathcal{E}_{\mathcal{L}}^l} v_j^* = 0. \quad (18)$$

Thus, (16)–(18) constitute $p+1$ linear equations in $p+1$ unknowns v_i, v_j^* and v_0. Note that p is usually very small.

We denote by $\mathbf{K}_l$ the $p \times p$ kernel sub-matrix for those points at the elbows, $\mathbf{v} = (v_i)\ \forall i \in \mathcal{E}_{\mathcal{R}}^l$, $\mathbf{v}^* = (v_j^*)\ \forall j \in \mathcal{E}_{\mathcal{L}}^l$, $\mathbf{y}_R^l = (y_i - \epsilon)\ \forall i \in \mathcal{E}_{\mathcal{R}}^l$, $\mathbf{y}_L^l = (y_j + \epsilon)\ \forall j \in \mathcal{E}_{\mathcal{L}}^l$, and

$$\mathbf{A}_l = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K}_l \end{bmatrix}, \mathbf{v}^a = \begin{bmatrix} v_0 \\ \mathbf{v} \\ -\mathbf{v}^* \end{bmatrix}, \mathbf{y}^a = \begin{bmatrix} 0 \\ \mathbf{y}_R^l \\ \mathbf{y}_L^l \end{bmatrix}.$$

[1] Whenever $p < 2$, we face a problem in the initialization setup and hence need to change λ and w_0 to look for valid points for updating.

Thus, the $p+1$ linear equations can be represented in matrix form as

$$\mathbf{A}_l \mathbf{v}^a = (\lambda - \lambda^l)\mathbf{y}^a. \tag{19}$$

If $\mathbf{A}_l$ is of full rank, then $\mathbf{A}_l^{-1}$ exists and we have

$$\begin{aligned} \alpha_0 &= \alpha_0^l + (\lambda - \lambda^l)b_0 && (20)\\ \alpha_i &= \alpha_i^l + (\lambda - \lambda^l)b_i \quad \forall i \in \mathcal{E}_{\mathcal{R}}^l && (21)\\ \alpha_j^* &= \alpha_j^{*l} - (\lambda - \lambda^l)b_j \quad \forall j \in \mathcal{E}_{\mathcal{L}}^l, && (22) \end{aligned}$$

where $\mathbf{b} = (b_0, b_i\,(\forall i \in \mathcal{E}_{\mathcal{R}}^l), b_j\,(\forall j \in \mathcal{E}_{\mathcal{L}}^l))^T = \mathbf{A}_l^{-1}\mathbf{y}^a$. Hence, $\alpha_i^{(*)}$ $(\forall i \in \mathcal{E}_{\mathcal{R}}^l \cup \mathcal{E}_{\mathcal{L}}^l)$ proceed linearly in λ. If $\mathbf{A}_l^{-1}$ does not exist, the solution paths for some $\alpha_i^{(*)}$ are not unique. For example, if three points in $\mathbb{R}$ are at the elbows and the linear kernel is applied, their kernel matrix is not of full rank. Then the λ-path algorithm has multiple updating possibilities through choosing any two of the points. In this paper, we do not consider this case. Rather, we will use a nonlinear kernel and a ridge term to ensure that $\mathbf{A}_l^{-1}$ always exists.

Plugging (20)–(22) into (15), we can obtain

$$f(\mathbf{x}) = \frac{\lambda^l}{\lambda}\big[f^l(\mathbf{x}) - h^l(\mathbf{x})\big] + h^l(\mathbf{x}), \tag{23}$$

where

$$h^l(\mathbf{x}) = \sum_{i \in \mathcal{E}_{\mathcal{R}}^l} b_i K(\mathbf{x}_i, \mathbf{x}) - \sum_{j \in \mathcal{E}_{\mathcal{L}}^l} b_j K(\mathbf{x}_j, \mathbf{x}) + b_0. \tag{24}$$

As λ decreases, the algorithm monitors the occurrence of any of the following events:

- The α_i or α_i^* value of a point $i \in \mathcal{E}_{\mathcal{R}}^l \cup \mathcal{E}_{\mathcal{L}}^l$ reaches 0 or 1. The value of λ for which this event occurs can be calculated from (21) or (22).
- A point $i \in \mathcal{R}^l \cup \mathcal{C}^l \cup \mathcal{L}^l$ hits an elbow, i.e., $|y_i - f(\mathbf{x}_i)| = \epsilon$. The value of λ for which this event occurs can be calculated from (23).

By monitoring the occurrence of these events, we can find the largest $\lambda < \lambda^l$ for which an event occurs. This is the $(l+1)$st event and the λ value is denoted as λ^{l+1}. The algorithm then updates the point sets and continues until either $p < 2$ or λ is close to 0. In each λ-path update, a set of linear equations is solved with $O(p^3)$ time complexity where p is typically quite small. Moreover, scanning through the training set to evaluate the next move has $O(n)$ time complexity.

2.3. ϵ-Path

Similar to the λ-path algorithm, the ϵ-path algorithm focuses on the points at the elbows only. If we set $\epsilon = \infty$, it is trivial to solve the optimization problem in (7)–(9). The solution is simply $\alpha_i^{(*)} = 0$ for all i, meaning that all the points are inside the tube (i.e., in $\mathcal{C}$) and, from (10), $f(\mathbf{x}) = w_0$. This is similar to the case shown in Figure 1(c). Since $|y_i - f(\mathbf{x}_i)| < \epsilon$ for all i, the initial parameter values are set as $\epsilon > (y_{max} - y_{min})/2$ and $w_0 = (y_{max} + y_{min})/2$. Compared with the λ-path algorithm which has to solve a linear programming problem to find the initial parameter values, the initialization problem for the ϵ-path algorithm is much easier to solve.

As before, let ϵ^l denote the value of ϵ right after the lth event has occurred. We assume that λ is prespecified by the user and remains fixed during the execution of the ϵ-path algorithm. At this time, we have $|\mathcal{E}_{\mathcal{R}}| > 0$ and $|\mathcal{E}_{\mathcal{L}}| > 0$. If this does not hold, we reduce ϵ until each elbow contains at least one point. Let $i_+ = \arg\max_{i \in \mathcal{C}} \big(y_i - f(\mathbf{x}_i)\big)$ and $i_- = \arg\min_{i \in \mathcal{C}} \big(y_i - f(\mathbf{x}_i)\big)$. Then $\mathcal{E}_{\mathcal{R}}^l = \{i_+\}$, $\mathcal{E}_{\mathcal{L}}^l = \{i_-\}$ and $\epsilon^l = (y_{i_+} - y_{i_-})/2$. This procedure only involves shrinking the tube to reduce its width without rotating it. The algorithm still holds even when more than one point enters an elbow simultaneously.

For ϵ values such that $\epsilon^{l+1} < \epsilon < \epsilon^l$, we can write the regression function as

$$f(\mathbf{x}) = \frac{1}{\lambda}\Big[\sum_{i \in \mathcal{E}_{\mathcal{R}}^l} (\alpha_i - \alpha_i^l)K(\mathbf{x}_i, \mathbf{x}) - \sum_{j \in \mathcal{E}_{\mathcal{L}}^l} (\alpha_j^* - \alpha_j^{*l})K(\mathbf{x}_j, \mathbf{x}) + (\alpha_0 - \alpha_0^l)\Big] + f^l(\mathbf{x}). \tag{25}$$

Let $\mathbf{1}_R^l = (1)_{|\mathcal{E}_{\mathcal{R}}^l| \times 1}$, $\mathbf{1}_L^l = (1)_{|\mathcal{E}_{\mathcal{L}}^l| \times 1}$, and $\mathbf{1}^a = \begin{bmatrix} 0 \\ \mathbf{1}_R \\ -\mathbf{1}_L \end{bmatrix}$. Using the same procedure as that for the λ-path algorithm, we can obtain the following system of $p+1$ linear equations:

$$\mathbf{A}_l \mathbf{v}^a = \lambda(\epsilon^l - \epsilon)\mathbf{1}^a. \tag{26}$$

If $\mathbf{A}_l^{-1}$ exists, then we let $\mathbf{c} = \mathbf{A}_l^{-1}\mathbf{1}^a$ and we have

$$\begin{aligned} \alpha_0 &= \alpha_0^l + \lambda(\epsilon^l - \epsilon)c_0 && (27)\\ \alpha_i &= \alpha_i^l + \lambda(\epsilon^l - \epsilon)c_i \quad \forall i \in \mathcal{E}_{\mathcal{R}}^l && (28)\\ \alpha_j^* &= \alpha_j^{*l} - \lambda(\epsilon^l - \epsilon)c_j \quad \forall j \in \mathcal{E}_{\mathcal{L}}^l. && (29) \end{aligned}$$

Hence, $\alpha_i^{(*)}$ $(\forall i \in \mathcal{E}_{\mathcal{R}}^l \cup \mathcal{E}_{\mathcal{L}}^l)$ also proceeds linearly in ϵ. We can obtain the regression function as

$$f(\mathbf{x}) = (\epsilon^l - \epsilon)h^l(\mathbf{x}) + f^l(\mathbf{x}), \tag{30}$$

where

$$h^l(\mathbf{x}) = \sum_{i \in \mathcal{E}_{\mathcal{R}}^l} c_i K(\mathbf{x}_i, \mathbf{x}) - \sum_{j \in \mathcal{E}_{\mathcal{L}}^l} c_j K(\mathbf{x}_j, \mathbf{x}) + c_0. \tag{31}$$

Similar to the λ-path algorithm, as ϵ decreases, the Lagrange multipliers change until either one more point enters an elbow or an existing point at an elbow leaves. We find the largest $\epsilon < \epsilon^l$ for which an event occurs, and then assign it as ϵ^{l+1} and update the point sets. This process can be understood geometrically in the linear space. If $d = 1$ and each elbow contains one point, then decreasing ϵ will rotate the tube with the two points at the two elbows as centers of rotation. Figure 1(d) shows one such example. The resulting rotation causes the width of the tube to decrease while the two elbow points remain at the elbows. The computational cost of the ϵ-path update is similar to that of the λ-path update.

Considering further the above example, the ϵ-path algorithm cannot proceed if a new point enters an elbow, i.e., $|\mathcal{E}_{\mathcal{R}}^l| + |\mathcal{E}_{\mathcal{L}}^l| > 2$, $|\mathcal{E}_{\mathcal{R}}^l| \geq 1$ and $|\mathcal{E}_{\mathcal{L}}^l| \geq 1$. In the $d = 1$ linear space, the width of the tube will be fixed by these elbow points and hence the tube can neither rotate nor shrink. As a result, ϵ cannot decrease. This problem always occurs in the linear space. If the dimensionality of the linear space is d, the rank of $\mathbf{K}_l$ is at most d no matter how many points are involved. Thus, $\mathbf{A}_l^{-1}$ in (26) does not exist when the elbows contain more than $d + 1$ points. This problem can also be overcome by using a nonlinear kernel and a ridge term. For example, if the Gaussian kernel is used, we can always execute the ϵ-path algorithm no matter how many points are at the elbows.

3. Two-Dimensional Solution Path

The ϵ-path algorithm can overcome the limitations of the λ-path algorithm. Specifically, when the λ-path algorithm needs to move the tube to look for valid points, we can shrink the tube by reducing ϵ until some points inside the tube enter the elbows. Since the ϵ-path algorithm can always proceed when a nonlinear kernel is used, we can explore the two-dimensional solution space of ϵ-SVR by executing the ϵ-path algorithm several times with different λ values. Starting from $\epsilon = \infty$, most points move from inside the tube to outside as the width of the tube decreases. However, some points may re-enter the tube after leaving it and pass through the elbows multiple times as ϵ approaches 0.

Enumerating the λ values and executing the ϵ-path algorithm many times is computationally unappealing. The λ-path algorithm can cooperate with the ϵ-path algorithm to explore the two-dimensional solution space. At any intermediate step of the ϵ-path algorithm, λ-path update can be applied as long as each elbow contains at least one point. In fact when ϵ decreases so that the tube fits the data well, the tube has been spline-fit by some points and the elbow sets $\mathcal{E}_{\mathcal{R}}$ and $\mathcal{E}_{\mathcal{L}}$ are not empty. Then λ-path update is always applicable with no difficulty. However, as we shift from ϵ-path update to λ-path update, if an elbow point has its $\alpha_i^{(*)}$ value equal to 0 or 1, it is somewhat complicated to implement λ-path update directly. Since λ can either increase or decrease, we are not sure whether this point should be included into the updating set for the λ-path algorithm. As a result, an additional check is required to ensure that the λ-path algorithm can proceed correctly. This overhead can be saved if the $\alpha_i^{(*)}$ values of all elbow points are neither 0 nor 1. Such solution can be obtained by terminating the ϵ-path algorithm between two consecutive events. The λ-path algorithm can then proceed successfully in both increasing and decreasing directions.

Note that there is a dramatic difference between the ϵ-path algorithm in ϵ-SVR and the λ-path algorithm in SVC. In the SVC formulation, the support vectors are those points inside the margin. To simplify the initialization step for the λ-path algorithm, λ is initialized to be very large so that most points are inside the margin. As λ decreases, both the width of the margin and the number of support vectors decrease. Since a classifier that generalizes well typically has a sparse representation involving a small number of support vectors, almost the entire solution path has to be explored until λ becomes very small such that most points are excluded from the margin. Thus the total number of moves is $O(n)$. Based on empirical findings, Hastie et al. (2004) suggested that this number is some small multiple of n. In the ϵ-SVR formulation, on the other hand, the support vectors are those points outside the ϵ-tube. With ϵ initialized to infinity, all points are inside the tube and hence there is no support vector in the beginning. As ϵ decreases, the points pass through the elbows from inside the tube to outside. This has a similar effect as increasing ν from 0 to 1 in ν-SVR, but the number of support vectors can be controlled exactly. To obtain a desirable regression function with the sparseness property, we only need a small number of steps. Therefore, there is no need to explore the entire ϵ-path and hence a good result can be obtained very efficiently.

4. Experimental Results

The behavior of the above algorithms can best be illustrated using videos. We have prepared some illustrative examples as videos in `http://www.cs.ust.hk/~wanggang/svrpath.htm`.

We randomly generate a set of 80 data points $\{(x_i, y_i)\}$

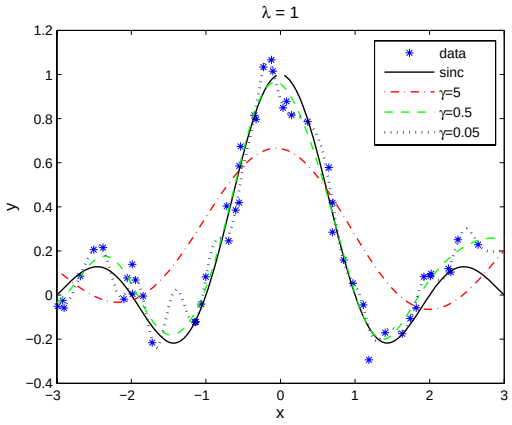


Figure 4. Based on three ϵ-paths with $\lambda = 1$ and $\gamma = 0.05, 0.5, 5$, the optimal solution for each path in terms of the mean squared error on the validation set is plotted.

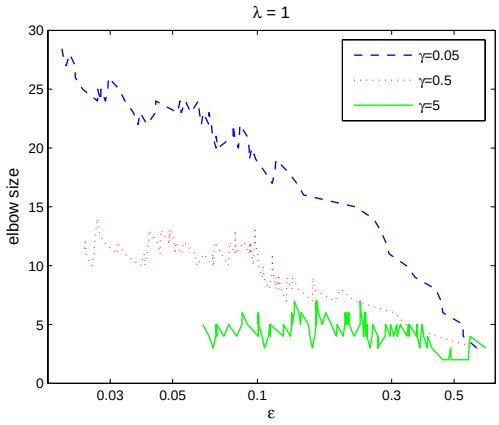


Figure 5. Change in elbow size as a function of ϵ for three ϵ-paths with $\lambda = 1$ and $\gamma = 0.05, 0.5, 5$. Since ϵ decreases rapidly in the beginning, the horizontal axis is shown in log scale.

with x_i drawn uniformly from $[-3, 3]$ and $y_i = \sin(\pi x_i)/(\pi x_i) + e_i$, where e_i is a Gaussian noise term with zero mean and a variance of 0.2. We randomly partition the data set into a training set of 50 points and a validation set of 30 points. The Gaussian RBF kernel $K(x_i, x_j) = \exp(\|x_i - x_j\|^2/\gamma)$ is used with three different γ values, 0.05, 0.5 and 5. We first run the ϵ-path algorithm with $\lambda = 1$. The algorithm terminates when 80% of the points become support vectors. For each solution path, we compute the mean squared error (MSE) on the validation set for every regression function solution along the path. The solution that minimizes the MSE is then chosen and the corresponding regression function is plotted in Figure 4. The optimal regression function overfits the data when $\gamma = 0.05$ but underfits the data when $\gamma = 5$. On the other hand, it fits the data well when $\gamma = 0.5$. In Figure 5, we plot the elbow size $|\mathcal{E}_L| + |\mathcal{E}_R|$ as a function of ϵ for different values of γ. When $\gamma = 0.05$, the tube is very elastic. The elbow size generally increases as ϵ decreases. During this process, more and more points move into the elbows and then settle down there. The regression function is thus sensitive to many points, leading to overfitting of the data. When $\gamma = 5$, on the other hand, the elbow size always remains small. Since the function is not flexible enough, many points are not likely to stay at the elbows simultaneously. This leads to underfitting of the data.

Figure 6 shows the effect of different values of the regularization parameter λ on the ϵ-path algorithm. Figure 6(a) shows that the regression function is not very sensitive to the value of λ. For $\lambda = 0.01, 0.1$ and 1, the solution paths show similar MSE curves as ϵ decreases and the optimal regression functions are obtained in the same ϵ range. However, when λ is quite large ($\lambda = 10$), it always tends to give a flat function leading to large MSE. Figure 6(b) shows that ϵ decreases rapidly during the first few steps of the ϵ-path algorithm. Afterwards, the rate of decrease in ϵ slows down significantly. As ϵ decreases, more and more points move towards the elbows. We next examine the relationships between the MSE and the number of steps in Figure 6(c). Similar to Figure 6(b), the MSE decreases rapidly during the first few steps.

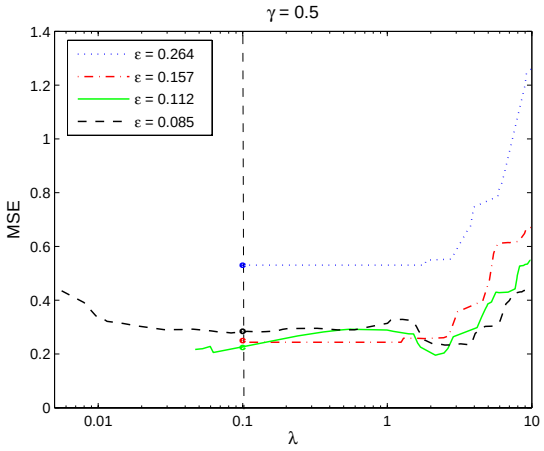
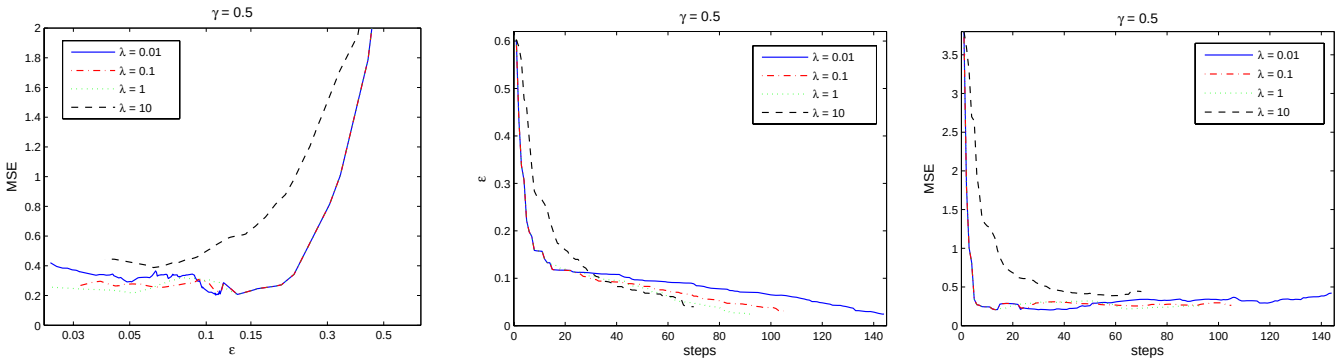


Figure 7. Shifting from the ϵ-path algorithm to the λ-path algorithm at four shifting points with different values of ϵ. The horizontal axis is in log scale.

Further executing the ϵ-path algorithm cannot lead to continued improvement in the generalization ability. Instead, the resulting regression function becomes more redundant and is likely to lead to overfitting. Moreover, it incurs unnecessarily high computational cost. Consequently, it is not necessary to explore all



(a) (b) (c)

Figure 6. Relationships between MSE, ϵ, and the number of steps in the algorithm for different values of λ. (a) MSE vs. ϵ, with the horizontal axis in log scale; (b) ϵ vs. number of steps; (c) MSE vs. number of steps.

solutions along the ϵ-path. The optimal solution preserving the sparseness property can be obtained very efficiently.

We now consider the scenario of shifting from the ϵ-path algorithm to the λ-path algorithm. The result is shown in Figure 7. We first run the ϵ-path algorithm with $\lambda = 0.1$ and $\gamma = 0.5$. As ϵ decreases, we select four different shifting points to launch the λ-path algorithm. The four points on the vertical line indicate the shifting points from the ϵ-path algorithm to the λ-path algorithm. λ can either increase or decrease. We stop λ from increasing further when it exceeds 10 and terminate the decreasing λ-path when no event occurs. There are two points in which the λ-path cannot extend towards the decreasing direction. For these cases, no event occurs until λ decreases to zero.

5. Conclusion

In this paper, we have proposed an efficient algorithm for computing the ϵ-path for ϵ-SVR. The coefficients of the regression function are piecewise linear with respect to both the regularization parameter λ and the error parameter ϵ. Since the ϵ-path algorithm can overcome some limitations of the λ-path algorithm, their integrated use allows efficient exploration of the two-dimensional solution space. Moreover, based on the special properties of the ϵ-path algorithm, the optimal solution can be obtained very efficiently.

Acknowledgment

This research was supported by the Competitive Earmarked Research Grant HKUST6174/04E from the Research Grants Council of the Hong Kong Special Administrative Region, China.

References

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2002). *Least angle regression* (Technical Report). Stanford University.

Gunter, L., & Zhu, J. (2005). Computing the solution path for the regularized support vector regression. *Advances in Neural Information Processing Systems 18 (NIPS-05).*

Hastie, T., Rosset, S., Tibshirani, R., & Zhu, J. (2004). The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, *5*, 1391–1415.

Rosset, S., & Zhu, J. (2003). *Piecewise linear regularized solution paths* (Technical Report). Stanford University.

Rosset, S., Zhu, J., & Hastie, T. (2004). Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, *5*, 941–973.

Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels.* MIT Press.

Schölkopf, B., Smola, A. J., Williamson, R. C., & Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation*, *12*, 1207–1245.

Vapnik, V. N. (1995). *The nature of statistical learning theory.* Springer-Verlag.

Zhu, J., Rosset, S., Hastie, T., & Tibshirani, R. (2003). 1-norm support vector machines. *Advances in Neural Information Processing Systems 16 (NIPS-03).*