

## Application of support vector regression for the prediction of concrete strength

Jong Jae Lee<sup>†</sup>

*Department of Civil and Environmental Engineering, Sejong University,  
Seoul 143-747, Korea*

Doo Kie Kim<sup>‡</sup> and Seong Kyu Chang<sup>‡‡</sup>

*Department of Civil and Environmental Engineering, Kunsan National University,  
Kunsan, Jeonbuk 573-701, Korea*

Jang-Ho Lee<sup>‡‡</sup>

*Department of Mechanical Engineering, Kunsan National University,  
Kunsan, Jeonbuk 573-701, Korea*

(Received December 5, 2006, Accepted August 9, 2007)

**Abstract.** The compressive strength of concrete is a commonly used criterion in producing concrete. However, the test on the compressive strength is complicated and time-consuming. More importantly, since the test is usually performed 28 days after the placement of the concrete at the construction site, it is too late to make improvements if unsatisfactory test results are incurred. Therefore, an accurate and practical strength estimation method that can be used before the placement of concrete is highly desirable. In this study, the estimation of the concrete strength is performed using support vector regression (SVR) based on the mix proportion data from two ready-mixed concrete companies. The estimation performance of the SVR is then compared with that of neural network (NN). The SVR method has been found to be very efficient in estimation accuracy as well as computation time, and very practical in terms of training rather than the explicit regression analyses and the NN techniques.

**Keywords:** concrete strength; strength prediction; support vector regression (SVR); concrete mix proportion data; kernel function.

### 1 Introduction

Concrete is one of the most widely-used materials in the construction industry. Traditionally, concrete has been fabricated from several well-defined components: cement, water, fine, and coarse aggregates, etc. In concrete mix design and quality control, the strength of concrete is a very

---

<sup>†</sup> Assistant Professor

<sup>‡</sup> Professor, Corresponding Author, E-mail: kim2kie@kunsan.ac.kr

<sup>‡‡</sup> E-mail: s9752033@kunsan.ac.kr

<sup>‡‡</sup> E-mail: jangho@kunsan.ac.kr

important property. The strength parameters of concrete include compressive, tensile, flexural, shear, bond strength, and so on, but most concrete elements are designed on the basis of the compressive strength of the material.

The mix design of concrete targets its 28-day compressive strength which is based on a standard uniaxial compression test and is accepted conventionally as a general index of concrete strength. However, concrete testing procedures require special equipment and are time-consuming; furthermore, experimental errors are inevitable. A typical test performed 28 days after concrete placement may be too late to make improvements if the test results do not satisfy the required criterion. Therefore, accurate and realistic strength estimation, which can be performed before the placement of concrete, is highly desirable.

For many years, researchers have proposed various methods for predicting concrete strength. Conventional methods are generally based upon statistical analyses, in which many linear and nonlinear regression equations have been constructed to model such prediction problems (Snell, *et al.* 1989, Popovics 1998). However, those traditional models have been developed with a fixed equation form based on a limited number of data and parameters. As an alternative, a standard multi-layered feed-forward neural network with a back propagation algorithm (neural network, NN) has been utilized to predict the compressive strength of concrete (Lee 2003, Kim, *et al.* 2004, Oztas, *et al.* 2006, Ji and Lin 2006). Various types of data have been used as the input to the neural network, e.g. mix proportions, temperature and humidity history, measurement data such as slump, air content, concrete temperature, etc. NN has the advantage of being able to effectively consider various inputs without using the explicit form of complicated equations as in conventional regression analyses. Also, it can easily adapt new data to the prediction model through a re-training process. However, NN also has its shortcomings, i.e., it requires much effort to determine the network's architecture and much computation time in training the network (Madan 2005).

Recently, the support vector machine (SVM) has been applied to various pattern recognition applications such as text classification and image recognition (Vapnik 1995, Ye, *et al.* 2005), and has been extended to regression analysis (Mukherjee, *et al.* 1997, Muller, *et al.* 1997, Yu, *et al.* 2006, Zhang, *et al.* 2006). In this study, the support vector machine for regression (support vector regression, SVR) is applied to predict concrete compressive strength. Training and test patterns for the SVR are based on actual mix proportions of two ready-mixed concrete companies. To investigate the performance of the SVR in estimation accuracy and computation time, the predicted results by several SVRs with different kernel functions are compared with those of the NN with various transfer functions.

## **2. Support vector machine**

The foundations of Support Vector Machines (SVM) have been developed by Vapnik (1995). The formulation is based on the Structural Risk Minimization (SRM) principle. The SVM is typically used to describe classification problems with support vector methods, and this theory is extended to the domain of regression problem. In this paper, Support Vector Regression (SVR) is used to predict the compressive strengths of concrete. Linear and nonlinear SVR are briefly described in Sections 2.1 and 2.2, respectively. More details on the theory of the SVM are well described in the reference by Vapnik (1995, 1999a).

### 2.1. Linear support vector regression

Assume that a set of training data  $\{(\mathbf{x}_i, y_i), i=1, \dots, n\}$  is given. Here,  $n$  is the number of training data, and  $\mathbf{x}_i$  and  $y_i$  are the  $i^{\text{th}}$  input training pattern and corresponding target output respectively.

The linear SVR finds a linear regression function as

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w} \cdot \mathbf{x} + b \quad (1)$$

where  $\cdot$  denotes the inner product,  $\mathbf{w}$  and  $b$  are the parameters of the function, and  $\mathbf{x}$  is the test pattern in a normalized form. The structural risk minimization principle can be realized by minimizing the empirical risk  $R_{\text{emp}}(\mathbf{w}, b)$  defined as Eq. (2), and the empirical risk can be described by the  $\varepsilon$ -insensitive loss function  $L_\varepsilon(y_i, f(\mathbf{x}_i, \mathbf{w}))$  defined as Eq. (3) (Stitson, et al. 1996).

$$R_{\text{emp}}(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^n L_\varepsilon(y_i, f(\mathbf{x}_i, \mathbf{w})) \quad (2)$$

$$L_\varepsilon(y_i, f(\mathbf{x}_i, \mathbf{w})) = \begin{cases} \varepsilon, & \text{if } |y_i - f(\mathbf{x}_i, \mathbf{w})| \leq \varepsilon \\ |y_i - f(\mathbf{x}_i, \mathbf{w})| - \varepsilon, & \text{otherwise} \end{cases} \quad (3)$$

$L_\varepsilon$  is the  $\varepsilon$ -insensitive loss function, or the tolerance error between the target output ( $y_i$ ) and the estimated output values ( $f(\mathbf{x}_i, \mathbf{w})$ ) in optimization process, and  $\mathbf{x}_i$  is a training pattern. The problem of finding  $\mathbf{w}$  and  $b$  to reduce the empirical risk with respect to an  $\varepsilon$ -insensitive loss function is equivalent to the convex optimization problem that minimizes the margin ( $\mathbf{w}$ ) and slack variables ( $\xi_i^*, \xi_i$ ) as

$$\lim_{\mathbf{w}, b, \xi, \xi^*} \left[ \frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \left( \sum_{i=1}^n \xi_i^* + \sum_{i=1}^n \xi_i \right) \right] \quad (4)$$

$$\text{subject to } \begin{cases} y_i - \mathbf{w} \cdot \mathbf{x}_i - b \leq \varepsilon + \xi_i^* \\ \mathbf{w} \cdot \mathbf{x}_i + b - y_i \leq \varepsilon + \xi_i, \quad i = 1, \dots, n \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

where the first term ( $\frac{1}{2} \mathbf{w} \cdot \mathbf{w}$ ) is the margin and is derived in Appendix A; the parameter  $C$  is a positive constant.

To solve the above optimization problem, one has to find a saddle point of the Lagrange function described as

$$\begin{aligned} L(\mathbf{w}, \xi^*, \xi, \alpha^*, \alpha, C, \gamma^*, \gamma) = & \frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \left( \sum_{i=1}^n \xi_i^* + \sum_{i=1}^n \xi_i \right) - \sum_{i=1}^n \alpha_i [y_i - \mathbf{w} \cdot \mathbf{x}_i - b + \varepsilon + \xi_i] \\ & - \sum_{i=1}^n \alpha_i^* [\mathbf{w} \cdot \mathbf{x}_i + b - y_i + \varepsilon + \xi_i^*] - \sum_i (\gamma_i^* \xi_i^* + \gamma_i \xi_i) \end{aligned} \quad (5)$$

The Lagrange function has to be minimized with respect to  $\mathbf{w}$ ,  $b$ ,  $\xi_i^*$ , and  $\xi_i$ , such that partial differential of  $L$  in Eq. (5) can be performed with respect to the following Karush-Kuhn-Tucker

(KKT) conditions

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} + \sum_{i=1}^n \alpha_i \mathbf{x}_i - \sum_{i=1}^n \alpha_i^* \mathbf{x}_i = 0 \quad \mathbf{w} = \sum_{i=1}^n (\alpha_i^* - \alpha_i) \mathbf{x}_i \quad (6a)$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^n \alpha_i - \sum_{i=1}^n \alpha_i^* = 0 \quad \sum_{i=1}^n \alpha_i = \sum_{i=1}^n \alpha_i^* \quad (6b)$$

$$\frac{\partial L}{\partial \xi^*} = C - \sum_{i=1}^n \gamma_i^* - \sum_{i=1}^n \alpha_i^* = 0 \quad \sum_{i=1}^n \gamma_i^* = C - \sum_{i=1}^n \alpha_i^* \quad (6c)$$

$$\frac{\partial L}{\partial \xi} = C - \sum_{i=1}^n \gamma_i - \sum_{i=1}^n \alpha_i = 0 \quad \sum_{i=1}^n \gamma_i = C - \sum_{i=1}^n \alpha_i \quad (6d)$$

where parameter  $\mathbf{w}$  of Eq. (6a) is relevant to parameter  $\mathbf{w}$  of Eq. (1). Then, substituting Eq. (6) into the Lagrange function (5), the dual form of the optimization function maximizing with respect to Lagrange multiplier  $\alpha_i^* \geq 0, \alpha_i \geq 0, \gamma_i^* \geq 0, \gamma_i \geq 0$ , and  $C^* \geq 0$  becomes

$$\max_{\alpha, \alpha^*} [\mathbf{w}(\alpha, \alpha^*)] = \max_{\alpha, \alpha^*} \left[ \sum_{i=1}^n y_i (\alpha_i^* - \alpha_i^*) - \varepsilon \sum_{i=1}^n y_i (\alpha_i^* - \alpha_i) - \frac{1}{2} \sum_{i,j=1}^n (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)(\mathbf{x}_i \cdot \mathbf{x}_j) \right] \quad (7)$$

$$\text{subject to } \begin{cases} \sum_{i=1}^n (\alpha_i^* - \alpha_i) = 0 \\ 0 \leq \alpha_i^*, \alpha_i \leq 0 \end{cases}, \quad i = 1, \dots, n$$

where  $\alpha_i^*$  and  $\alpha_i$  are Lagrange multipliers and are calculated by a standard quadratic programming (QP) (Arora 1989, Bazaraa, *et al.* 1993), and  $\mathbf{x}_i \cdot \mathbf{x}_j$  is the inner product of two training patterns  $\mathbf{x}_i$  and  $\mathbf{x}_j$ . Finally, substituting Eq. (6a) into Eq. (1), the linear regression function can be expressed as

$$f(\mathbf{x}, \alpha^*, \alpha) = \sum_{i=1}^n (\alpha_i^* - \alpha_i)(\mathbf{x}_i \cdot \mathbf{x}) + b \quad (8)$$

where Lagrange multipliers are subject to constraints  $0 \leq \alpha_i^*, \alpha_i \leq C$ . The parameter vectors of the regression function  $\mathbf{w}$  and  $b$  are calculated as

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^n (\alpha_i^* - \alpha_i) \mathbf{x}_i \\ b &= -\frac{1}{2} \mathbf{w} \cdot (\mathbf{x}_r + \mathbf{x}_s) \end{aligned} \quad (9)$$

where  $\mathbf{x}_r$  and  $\mathbf{x}_s$  are any support vectors.

## 2.2. Nonlinear support vector regression

Since most real world problems are complex, the linear SVR has limited application. As an alternative, nonlinear SVR has been utilized. It is a simple combination of a mapping from the input data to a higher dimensional feature space and the linear SVR algorithm. The input training pattern  $\mathbf{x}_i$  is transformed into feature space ( $\phi(\mathbf{x}_i)$ ) by a nonlinear function (Aizerman, *et al.* 1964). Then, the optimization algorithm is applied in the same way as the linear SVR. Accordingly, the SVR for a nonlinear case is expressed as

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b \quad (10)$$

where  $\mathbf{w}$  and  $b$  are the parameter vectors of the function and  $\phi(\mathbf{x})$  is the mapping function from the input features to a higher dimensional feature space.

Fig. 1 shows the schematic diagram of the nonlinear SVR with the  $\varepsilon$ -insensitive loss function. Solid points are the support vectors obtained from the training data, which have the largest margin from the decision boundary. The plot on the right hand side shows the  $\varepsilon$ -insensitive loss function which has an error tolerance  $\varepsilon$ , and the lower and upper bounds calculated by a slack variable ( $\xi_i^*$ ,  $\xi_i$ ).

Following the same procedures as in Section 2.1, the nonlinear SVR can be expressed as

$$\begin{aligned} \max_{\alpha, \alpha^*} [W(\alpha, \alpha^*)] &= \max_{\alpha, \alpha^*} \left[ \sum_{i=1}^n y_i (\alpha_i^* - \alpha_i) - \varepsilon \sum_{i=1}^n (\alpha_i^* + \alpha_i) - \frac{1}{2} \sum_{i,j=1}^n (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \{\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)\} \right] \quad (11) \\ \text{subject to} &\begin{cases} \sum_{i=1}^n (\alpha_i^* - \alpha_i) = 0, & i = 1, \dots, n \\ 0 \leq \alpha_i^*, \alpha_i \leq 0 \end{cases} \end{aligned}$$

Inner product  $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$  in Eq. (11) requires complex computation in the feature space. Based on the Mercer's condition (Vapnik 1999b), this inner product can be replaced by a simple arithmetic computation in input space using a kernel function ( $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$ ). Therefore, Eq. (11) can be rewritten as

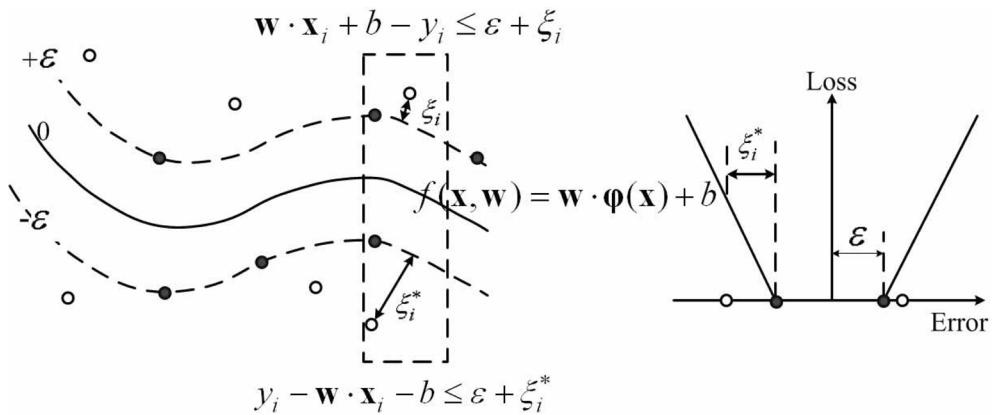


Fig. 1 Nonlinear SVR with  $\varepsilon$ -insensitive loss function (Yu, *et al.* 2006)

$$\max_{\alpha, \alpha^*} [W(\alpha, \alpha^*)] = \max_{\alpha, \alpha^*} \left[ \sum_{i=1}^n y_i (\alpha_i^* - \alpha_i) - \varepsilon \sum_{i=1}^n (\alpha_i^* + \alpha_i) - \frac{1}{2} \sum_{i,j=1}^n (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) \right] \quad (12)$$

subject to  $\begin{cases} \sum_{i=1}^n (\alpha_i^* - \alpha_i) = 0 \\ 0 \leq \alpha_i^*, \alpha_i \leq C \end{cases}, \quad i = 1, \dots, n$

There are various kernel functions such as linear, polynomial, radial basis function, sigmoid kernel, etc. In this study, the following four kernel functions are utilized.

1. Linear kernel function:

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}) = \mathbf{x}_i \cdot \mathbf{x} \quad (13a)$$

2. Polynomial kernel function:

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}) = (\mathbf{x}_i \cdot \mathbf{x} + 1)^d \quad (13b)$$

3. Gaussian Radial Basis function:

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}) = \exp \left[ -\frac{(\mathbf{x}_i - \mathbf{x}) \cdot (\mathbf{x}_i - \mathbf{x})}{2\sigma^2} \right] \quad (13c)$$

4. Exponential Gaussian Radial Basis function:

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}) = \exp \left[ -\frac{\sqrt{(\mathbf{x}_i - \mathbf{x}) \cdot (\mathbf{x}_i - \mathbf{x})}}{2\sigma} \right] \quad (13d)$$

where  $\mathbf{x}_i$  and  $\mathbf{x}$  are the training and test patterns, respectively,  $d$  is a dimension of the input vector, and  $\sigma$  is a the global basis function width. Finally, the nonlinear SVR function is expressed as

$$f(\mathbf{x}, \alpha^*, \alpha) = \sum_{i=1}^n (\alpha_i^* - \alpha_i) \mathbf{K}(\mathbf{x}_i, \mathbf{x}) + b \quad (14)$$

where

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x} &= \sum_{SVs} (\alpha_i^* - \alpha_i) \mathbf{K}(\mathbf{x}_i, \mathbf{x}) \\ b &= -\frac{1}{2} \sum_{SVs} (\alpha_i^* - \alpha_i) \mathbf{K}[(\mathbf{x}_r, \mathbf{x}_i) + \mathbf{K}(\mathbf{x}_s, \mathbf{x}_i)] \end{aligned} \quad (15)$$

where  $\mathbf{x}_r$  and  $\mathbf{x}_s$  are any support vectors,  $SVs$  is the number of support vectors, and Lagrange multipliers are subject to constraints  $0 \leq \alpha_i^*, \alpha_i \leq C$ .

### 3. Estimation of concrete strength using SVR

#### 3.1. Overview of estimation of concrete strength using SVR and NN

Most constructions use ready-mixed concrete prepared according to specified mix proportions. Ready-

Table 1 Material properties of concrete (Kim, et al. 2004)

Properties of material		Experiment data	
		Company A	Company B
Specific gravity	Cement	3.14	3.15
	Natural sand ( $s_1$ )	2.59	2.58
	Crushed sand ( $s_2$ )	2.51	-
	Coarse aggregate	2.64	2.63
Fineness modulus	Natural sand ( $s_1$ )	3.30	2.70
	Crushed sand ( $s_2$ )	2.25	-
	Coarse aggregate	6.53	6.60
Admixtures	Air-entraining admixtures	AE water-reducing (Standard)	

mixed concrete suppliers base their own mix proportions on codes, experiments, and experience as well. In this study, the SVR for estimating the concrete strength is applied using the actual mix proportion data provided by two concrete companies. The material properties of concrete are shown in Table 1.

Table 2 Samples of mix proportions of Company A for training

Specified strength (MPa)	Slump (cm)	W/C weight ratio (%)	Fine aggregate percentage (%)	Unit water content (kN/m <sup>3</sup> )	Unit cement content (kN/m <sup>3</sup> )	Unit fine aggregate content (kN/m <sup>3</sup> )		Unit coarse aggregate content (kN/m <sup>3</sup> )	Admixture (%)
						Natural sand ( $s_1$ )	Crushed sand ( $s_2$ )		
9.8	8	84.9	50.4	1.77	2.09	3.65	5.47	9.22	0.64
11.76	10	76.9	49.2	1.79	2.33	3.51	5.27	9.32	0.72
13.72	12	69.9	48.2	1.81	2.61	3.39	5.08	9.35	0.80
13.72	21	69.9	50.0	2.00	2.87	3.37	5.07	8.67	0.88
15.68	10	64.2	46.6	1.75	2.73	3.28	4.93	9.67	0.84
15.68	15	64.2	47.6	1.86	2.90	3.27	4.92	9.26	0.89
17.64	5	59.4	44.7	1.64	2.75	3.19	4.80	10.17	0.84
17.64	12	59.4	46.1	1.78	3.00	3.19	4.79	9.59	0.92
17.64	18	59.4	47.3	1.91	3.21	3.19	4.77	9.11	0.98
20.58	12	53.5	44.9	1.76	3.29	3.08	4.62	9.70	1.01
20.58	18	53.5	46.1	1.89	3.54	3.07	4.60	9.19	1.08
23.52	8	48.6	43.1	1.67	3.43	2.98	4.47	10.09	1.05
23.52	12	48.5	43.9	1.75	3.61	2.97	4.46	9.75	1.10
26.46	10	44.2	42.7	1.70	3.83	2.88	4.32	9.94	1.17
26.46	18	44.3	44.3	1.86	4.20	2.86	4.29	9.23	1.29
29.4	10	40.9	42.0	1.69	4.13	2.80	4.19	9.93	1.26
29.4	15	40.9	43.0	1.79	4.38	2.78	4.17	9.47	1.34
34.3	10	35.7	40.9	1.68	4.69	2.66	3.98	9.85	1.44
34.3	18	35.7	42.5	1.83	5.14	2.63	3.94	9.12	1.57
37.24	18	33.4	42.1	1.83	5.46	2.56	3.84	9.04	167
39.2	15	32.1	41.2	1.76	5.50	2.53	3.79	9.26	1.68

Table 3 Samples of mix proportions of Company B for training

Specified Strength (MPa)	Slump (cm)	W/C weight ratio (%)	Fine aggregate percentage (%)	Unit water content (kN/m <sup>3</sup> )	Unit cement content (kN/m <sup>3</sup> )	Unit fine aggregate content (kN/m <sup>3</sup> )		Unit coarse aggregate content (kN/m <sup>3</sup> )	Admixture (%)
						Natural sand ( $s_1$ )	Crushed sand ( $s_2$ )		
9.8	8	82.0	54.8	1.69	2.07	10.19	-	8.58	1.06
11.76	10	73.8	53.1	1.71	2.33	9.75	-	8.77	1.19
13.72	12	66.3	51.4	1.72	2.58	9.32	-	8.97	1.32
13.72	21	66.9	50.6	1.86	2.80	8.89	-	8.84	1.43
15.68	10	63.0	50.9	1.68	2.66	9.23	-	9.08	1.36
15.68	15	63.0	50.4	1.76	2.79	8.98	-	9.01	1.43
17.64	5	59.0	50.5	1.59	2.71	9.25	-	9.25	1.39
17.64	12	59.0	49.8	1.71	2.91	8.90	-	9.14	1.49
17.64	18	58.0	49.2	1.80	3.09	8.59	-	9.05	1.58
20.58	12	53.0	48.6	1.70	3.22	8.57	-	9.24	1.65
20.58	18	53.0	48.0	1.78	3.39	8.29	-	9.15	1.73
23.52	8	49.0	48.2	1.62	3.32	8.56	-	9.37	1.7
23.52	12	49.0	47.8	1.69	3.46	8.35	-	9.29	1.77
26.46	10	45.0	47.3	1.65	3.65	8.23	-	9.35	1.86
26.46	18	45.0	46.5	1.77	3.92	7.84	-	9.19	2.00
29.4	10	42.0	46.6	1.64	3.92	8.02	-	9.37	2.00
29.4	15	42.0	46.1	1.72	4.12	7.76	-	9.25	2.10
34.3	10	37.0	45.7	1.63	4.40	7.69	-	9.32	2.25
34.3	18	37.0	44.9	1.75	4.74	7.29	-	9.11	2.42
37.24	18	34.7	44.4	1.75	5.08	7.09	-	9.05	2.59
39.2	15	33.0	44.4	1.71	5.14	7.11	-	9.08	2.62

In all mixtures, normal Portland cement is used. The maximum size of aggregates is 25 mm. The range of compressive strengths in training patterns is defined from 9.8 to 39.2 MPa with a step size of 0.98 MPa (a total of 31 strengths). There are 7 independent samples with the slump values of 5, 8, 10, 12, 15, 18, and 21 cm in each group (strength). Therefore, companies A and B each have a total of 217 input and output pairs as training and test patterns. Five out of the 7 samples are randomly selected as training patterns, and the rest are used for testing, i.e., for each company, there are 155 training patterns and 62 test patterns in all.

Eight (for Company B) and nine (for Company A) parameters e.g. water-cement ratio, fine aggregate percentage, unit water content, unit cement content, unit fine aggregate content, unit coarse aggregate content, admixtures, and slump, are used as the input set for SVR; while the specified compressive strength is defined as the output to be estimated. Slump can also be used as the output to be estimated, but in this study, only the compressive strength is used for this purpose. A mixture of natural sand ( $s_1$ ) and crushed sand ( $s_2$ ) is used in Company A as fine aggregates, so Company A has an additional parameter for fine aggregate content. Examples of the specified concrete mix proportions of two companies for training are shown in Tables 2 and 3.

Fig. 2 shows the implementation procedures of SVR for strength estimation. First, all the input data of training and test patterns, and output data (concrete strengths) are normalized to 0.1~0.9 to

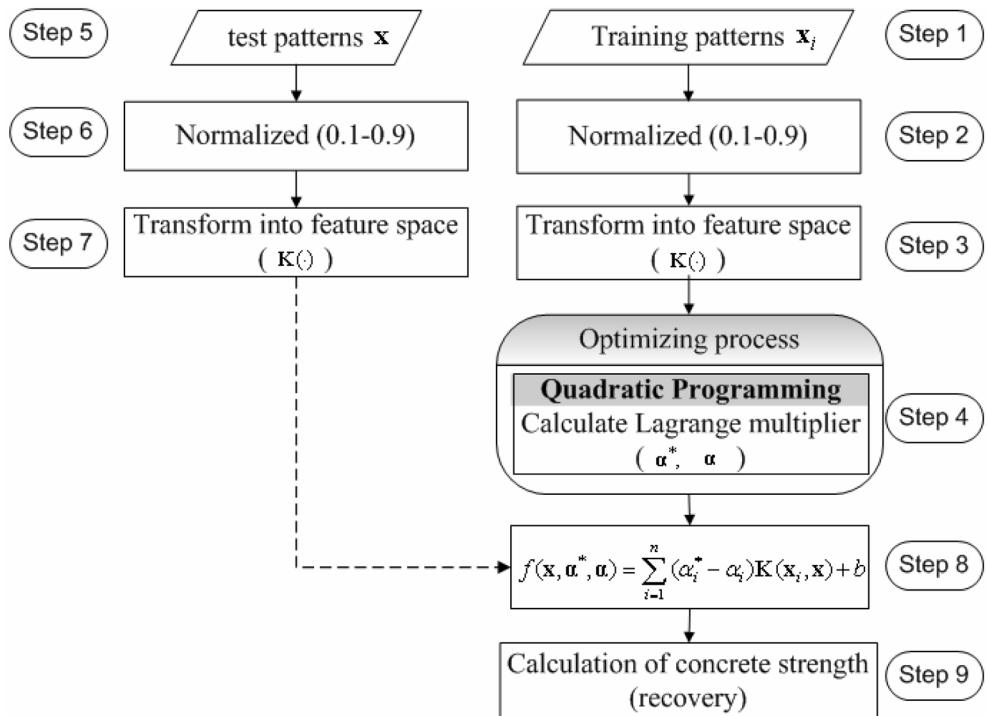


Fig. 2 Implementation procedure of nonlinear SVR

give an equal weighting factor before implementing the data (Steps 1 and 2). Then, training patterns are transformed into feature space using the kernel function ( $K(\cdot)$ ) (Step 3). The mapped training patterns and outputs are optimized by the quadratic programming, resulting in the Lagrange multipliers  $\alpha$  and  $\alpha^*$  (Step 4). After transforming the test patterns (Steps 5, 6 and 7), the normalized concrete strengths are calculated by Eq. (14) (Step 8), and are recovered by the scaling factor used in the data normalization stage (Step 9). An explanation of the step-by-step procedure of SVR applied to a simple example is provided in Appendix B.

A problem regarding the choice of two parameters ( $C$  and  $\varepsilon$ ) for SVR has been studied by several researchers (Cherkassky and Mulier 1998, Cherkassky and Ma 2004). The parameter  $C$  controls the smoothness or flatness of the approximation function. A large value of  $C$  indicates that the objective is only to minimize the empirical risk, which makes the learning machine more complex. On the other hand, a smaller  $C$  value may cause the errors to be excessively tolerated yielding a learning machine with poor approximation (Yu, *et al.* 2006). In this paper, an SVR model is constructed with  $C=10$  and  $\varepsilon=0.004$ , following the empirical values by Yu, *et al.* (2006).

In this study, the following are employed: Neural Network Toolbox in Matlab is utilized for comparative studies; a standard multi-layered feed-forward neural network with a back propagation algorithm is incorporated; the number of hidden layer is one; the number of neurons in the network is 9-5-1 for Company A and 8-5-1 for Company B; and the learning rate is 0.95. Most common combinations of transfer functions, the equations of which are given in Table 4, are utilized to investigate the general performance of NN. More details on the theoretical background of NN are described in the reference by Kim, *et al.* (2004).

Table 4 Transfer function and its equation used for NN

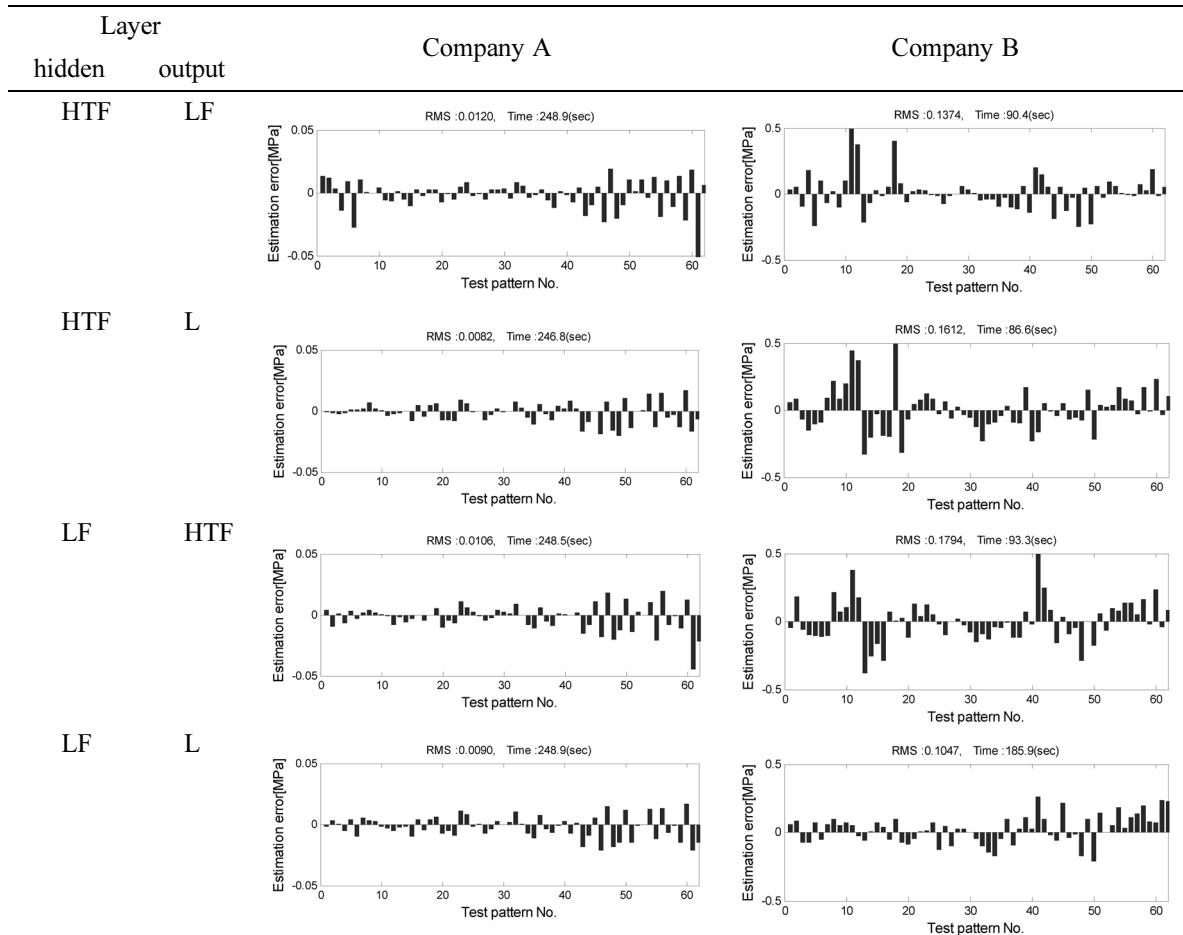
Transfer function	Equation
Linear function (L)	$f(v) = v$
Logistic function (LF)	$f(v) = \frac{1}{1 + \exp(-av)}$
Hyperbolic tangent function(HTF)	$f(v) = a \tanh(bv)$

\*Note:  $a$  and  $b$  are constants, and  $v$  represents local field.

### 3.2. Estimation results by SVR and NN

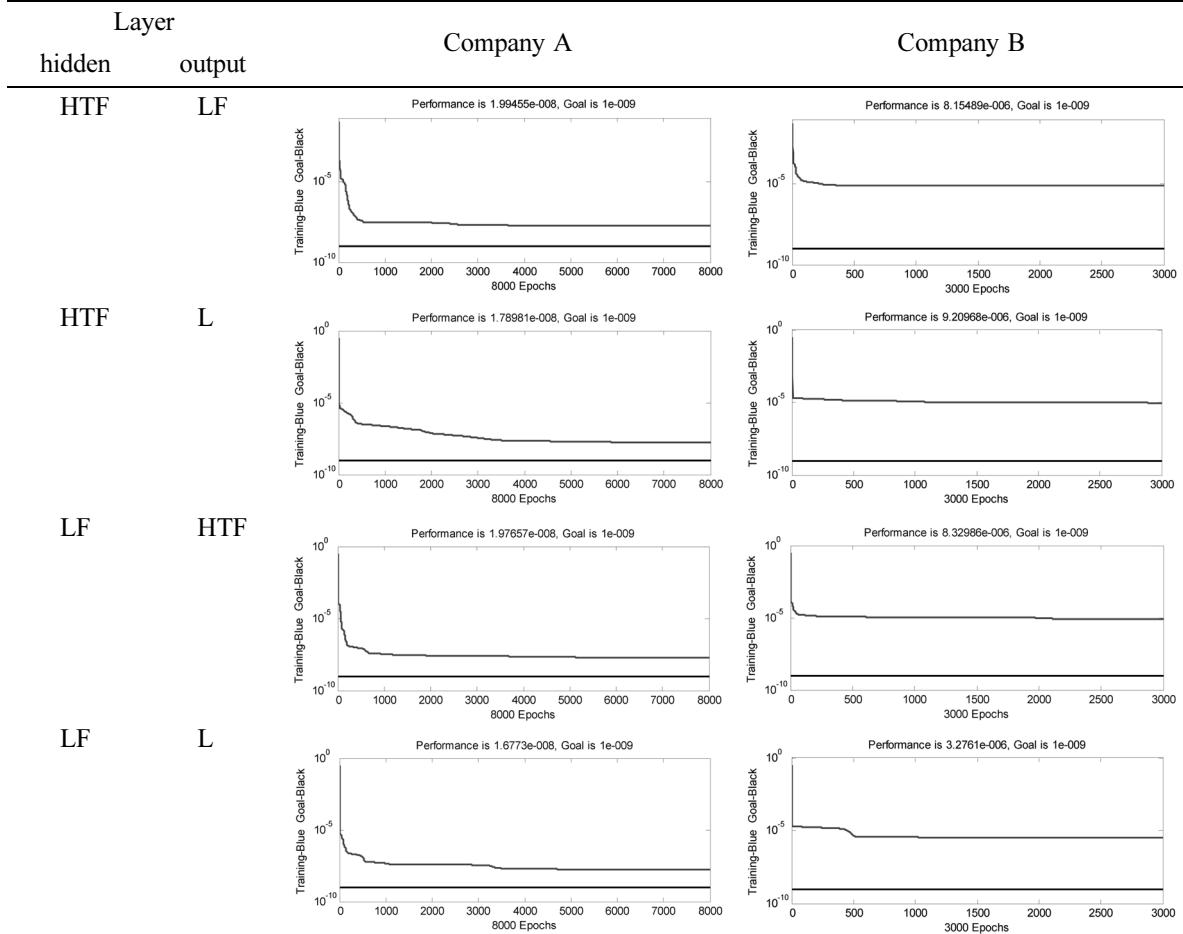
To investigate the performance of the SVR in estimation accuracy and computation time, the predicted results produced by several SVRs with different kernel functions described in Eq. (13) (where  $d$  and  $\sigma$  values are assumed to be 2 for simple kernel functions) are compared with those by

Table 5 Error of estimation results of NN according to transfer function



\*HTF: Hyperbolic tangent function, LF: Logistic function, L: Linear function

Table 6 Variation of MSE with training iteration of NN



\* HTF: Hyperbolic tangent function, LF: Logistic function, L: Linear function

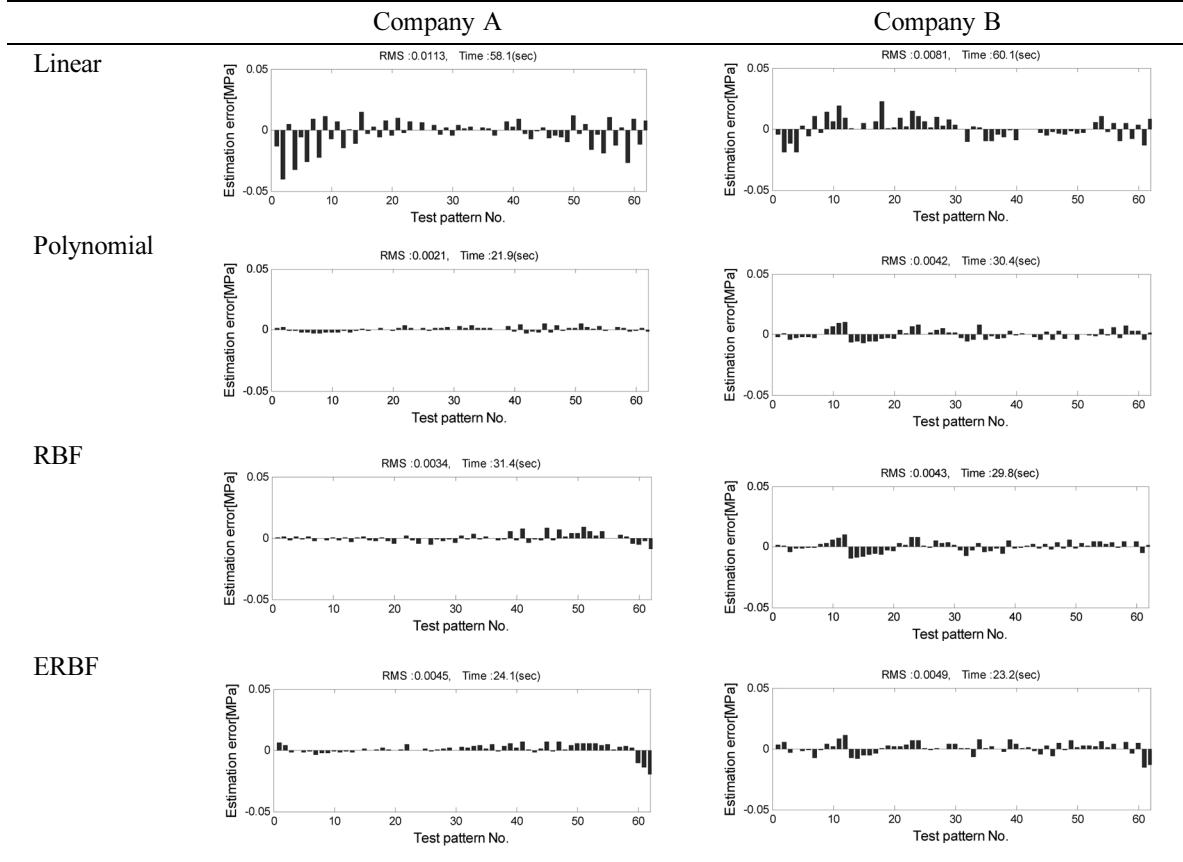
NNs with various transfer functions. This is because NN also has an advantage of being able to effectively consider various inputs without using complicated equations in predicting the concrete strength, in contrast to conventional regression analyses.

The estimated results along with the corresponding computation time of SVRs and NNs for the two companies' data are shown in the Tables 5 and 7. Here, estimation errors are defined by the root mean square (RMS) errors as

$$e = \sqrt{\frac{1}{N} \sum_{i=1}^N (f_i - \bar{f})^2} \quad (16)$$

where  $N$  is the number of test patterns and  $f$  and  $\bar{f}$  denote the actual and predicted concrete strengths, respectively.

As for the estimation results by NN, the NN with Hyperbolic tangent function and Logistic function shows the best results in Company A; and the NN using Logistic and Linear transfer function shows the best results in Company B. The learning of NN is performed until the mean

Table 7 Error of estimation results of SVR according to kernel function ( $\sigma=2$ )

square error (MSE) no longer decreases. The maximum training epochs for Company A and B are 8000 and 3000, respectively. Table 6 shows the process of convergence of MSE for the two companies. In all cases, SVRs show more accurate estimation capability than NNs. In this example, the SVR with the polynomial kernel function shows the best results among the SVRs with four kernel functions. For the cases of Company B, the estimation errors by SVRs are found to be remarkably smaller than those by NNs. As for the computation time, SVRs take less time than NNs in all cases. It takes a longer time for the cases of Company A than the cases of Company B since the complexity of the prediction model is closely related to the number of inputs (wherein as mentioned earlier, Company A has an additional input parameter in the fine aggregate content). Though the computation time may increase with the maximum training epoch, the estimation performance may not be satisfied. When the maximum training epoch for Company B is increased to 8000, the estimation results are not improved much.

The above results are represented by the following reason: SVR utilizes structural risk minimization (SRM) principle which reduces an upper bound on the generalization error, while NN employs traditional empirical risk minimization (ERM) which reduces the training error. In other words, a global optimal solution can be found using SVR, while with NN there are none. Furthermore, SVR is less complex than NN in choosing parameters. The values of the SVR

parameters (e.g. kernel function, the global basis function width  $\sigma$ ,  $C$ , and  $\varepsilon$ ) can be determined easily (Cherkassky and Ma 2004, Yu *et al.* 2006). In NN however, results are defined by various parameters (e.g. the number of hidden layers and neurons, the transfer function, iteration of training, initial weight values), therefore requiring considerable efforts in order to construct a good NN model.

## 5. Conclusions

This paper presents a promising support vector regression (SVR) technique for predicting the compressive strength of concrete based on its mix proportion data. The performance of the proposed method is verified by comparing the predicted strengths using the several SVRs with different kernel functions with those by the neural networks (NNs) with various transfer functions. Both SVR and NN methods show good estimation results. However, estimation results by SVRs produce remarkably smaller estimation errors compared with those by NNs. Moreover, SVRs take lesser time than NNs for computations in all cases. From these results, it can be concluded that the SVR method can predict the compressive strength of concrete with higher estimation accuracy and in a shorter computation time.

It is expected that the present SVR method for predicting the concrete strength can contribute to the maintenance of concrete quality for optimal mixtures. As the database containing influential parameters on concrete strengths are well established over time, the SVR using the training data obtained from this will become more effective and the resulting predictions more reliable. Moreover, the SVR method can be utilized easily by field engineers since it does not require any procedure to determine the explicit forms as in the regression analysis, or any knowledge on the network's architecture as in the NN techniques.

## Acknowledgements

This work was supported by the Korea Science and Engineering Foundation (KOSEF) grant funded by Ministry of Science & Technology (No. R01-2006-000-10610-0(2007)) and by a grant (05RCB05-01) from Regionally Characterized Construction Technology Program funded by Ministry of Construction & Transportation of Korean government.

## Appendix A

In the case of the SVM, the distance  $d(\mathbf{w}, \mathbf{x})$  of a point  $\mathbf{x}$  from the hyperplane ( $\mathbf{w} \cdot \mathbf{x} + b = 0$ ) is

$$d(\mathbf{w}, \mathbf{x}) = \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\|\mathbf{w}\|} \quad (\text{A.1})$$

The optimal hyperplane is given by maximizing the margin,  $M$ , subject to the constraint of equation (A.2).

$$y_i[\mathbf{w} \cdot \mathbf{x} + b] \geq 1, \quad i = 1, \dots, n \quad (\text{A.2})$$

The margin is given by,

$$\begin{aligned} M &= \min_{\{\mathbf{x}:y=1\}} d(\mathbf{w}, \mathbf{x}) + \min_{\{\mathbf{x}:y=-1\}} d(\mathbf{w}, \mathbf{x}) = \min_{\{\mathbf{x}:y=1\}} \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\mathbf{w} \cdot \mathbf{w}} + \min_{\{\mathbf{x}:y=-1\}} \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\mathbf{w} \cdot \mathbf{w}} = \\ &= \frac{1}{\mathbf{w} \cdot \mathbf{w}} (\min_{\{\mathbf{x}:y=1\}} |\mathbf{w} \cdot \mathbf{x} + b| + \min_{\{\mathbf{x}:y=-1\}} |\mathbf{w} \cdot \mathbf{x} + b|) = \frac{2}{\mathbf{w} \cdot \mathbf{w}} \end{aligned} \quad (\text{A.3})$$

where  $|\mathbf{w} \cdot \mathbf{x} + b| = 1$  (Gunn 1998).

Therefore, the optimization problem is one that minimizes

$$\Phi(\mathbf{w}) = \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) \quad (\text{A.4})$$

## Appendix B

The step-by-step procedures for linear and nonlinear SVR are explained using a simple regression problem. The only difference between linear and nonlinear SVR is that the latter uses the kernel function.

### Problem definition

Find an SVR function same as  $\mathbf{Y} = 2\mathbf{X}_1 + 3\mathbf{X}_2$  with two input parameters using the following samples.

	Input		Output
	$\mathbf{X}_1$	$\mathbf{X}_2$	$\mathbf{Y}$
Training samples	0.1	0.3	1.1
	0.2	0.5	1.9
	0.4	0.1	1.1
	0.5	0.6	2.8
Test sample	0.3	0.4	1.8

### The step-by-step procedure to find an SVR function

#### Step1: Prepare training patterns

Training patterns are shown in the problem definition.

#### Step2: Normalize training samples

Input and output data are normalized to have values between 0.1-0.9 using the following equations.

$$\bar{\mathbf{X}}_i = \left( \frac{\mathbf{X}_i - \min_j (X_i^{th})}{\max_j (X_i^{th}) - \min_j (X_i^{th})} \right) \times 0.8 + 0.1, \quad \bar{\mathbf{Y}} = \left( \frac{\mathbf{Y} - \min_j (Y_i^{th})}{\max_j (Y_i^{th}) - \min_j (Y_i^{th})} \right) \times 0.8 + 0.1$$

	Input		Output
	$\bar{\mathbf{X}}_1$	$\bar{\mathbf{X}}_2$	$\bar{\mathbf{Y}}$
Training samples	0.10	0.42	0.10
	0.30	0.74	0.48
	0.70	0.10	0.10
	0.90	0.90	0.90

*Step3: Transform training patterns into feature space*

For linear SVR,

$$\mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j) = \bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_j = \begin{bmatrix} 0.19 & 0.34 & 0.11 & 0.47 \\ 0.34 & 0.64 & 0.28 & 0.94 \\ 0.11 & 0.28 & 0.50 & 0.72 \\ 0.47 & 0.94 & 0.72 & 1.62 \end{bmatrix}$$

For nonlinear SVR using the following polynomial kernel function,

$$\mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j) = (\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}_j + 1)^2 = \begin{bmatrix} 1.41 & 1.80 & 1.24 & 2.16 \\ 1.80 & 2.68 & 1.65 & 3.75 \\ 1.24 & 1.65 & 2.25 & 2.96 \\ 2.16 & 3.75 & 2.96 & 6.86 \end{bmatrix}$$

**Step4: Sovle QP problem**

The object function for QP is described as

$$\min_{\gamma} f(r) = \frac{1}{2} \gamma^T \mathbf{H} \gamma + \mathbf{c}^T \gamma \quad (B.1)$$

$$\mathbf{LB} \leq \gamma \leq \mathbf{UB}$$

where the Lagrange multiplier,

$$\gamma = \begin{cases} \alpha^* \\ \alpha \end{cases}; \quad \mathbf{H} = \begin{bmatrix} \mathbf{K} & -\mathbf{K} \\ -\mathbf{K} & \mathbf{K} \end{bmatrix}; \quad \text{and } c = \begin{cases} \varepsilon - \mathbf{Y} \\ \varepsilon + \mathbf{Y} \end{cases}.$$

In this example,  $\gamma$  and  $c$  are  $8 \times 1$  column vectors;  $\alpha^*$  and  $\alpha$  are  $4 \times 1$  column vectors;  $\mathbf{K}$  and  $\mathbf{H}$  are  $4 \times 4$  and  $8 \times 8$  matrices, respectively;  $\varepsilon$  is a  $4 \times 1$  column vector of constant values,  $1 \times 10^{-5}$ ; and  $\mathbf{LB}$ , and  $\mathbf{UB}$  are  $8 \times 1$  column vectors of constant values, 0 and 10, respectively.

Solving the QP problem, the Lagrange multiplier,  $\gamma$ , is obtained as follows (Arora 1989, Bazaraa et al. 1993):

For linear SVR,

$$\alpha^* = \langle 0 \quad 0 \quad 0 \quad 1.25 \rangle^T, \alpha = \langle 0.368 \quad 0 \quad 0.882 \quad 0 \rangle^T,$$

$$\beta = \alpha^* - \alpha = \langle -0.368 \quad -0.882 \quad 0 \quad 1.25 \rangle^T$$

For nonlinear SVR,

$$\alpha^* = \langle 0 \quad 0.159 \quad 0 \quad 0.159 \rangle^T, \alpha = \langle -0.159 \quad 0 \quad 0.159 \quad 0 \rangle^T$$

$$\beta = \alpha^* - \alpha = \langle -0.159 \quad 0.159 \quad -0.159 \quad 0.159 \rangle^T$$

#### *Step 5: Prepare test pattern*

A test pattern is shown in the problem definition.

#### *Step 6: Normalize test samples*

Input parameters of test data are also normalized to have a value between 0.1-0.9 following the same transformation as described in Step 2.

$$[\bar{\mathbf{X}}_1^{(test)}, \bar{\mathbf{X}}_2^{(test)}] = [0.5 \quad 0.58]$$

#### *Step 7: Transform test patterns into feature space*

For linear SVR,

$$\mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}^{(test)}) = \bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}^{(test)} = [\bar{\mathbf{X}}_1^{(test)} \quad \bar{\mathbf{X}}_2^{(test)}][\bar{\mathbf{X}}_1 \quad \bar{\mathbf{X}}_2]^T$$

$$= [0.2936 \quad 0.5792 \quad 0.4080 \quad 0.9720]$$

For nonlinear SVR using the following polynomial kernel function,

$$\mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}^{(test)}) = (\bar{\mathbf{x}}_i \cdot \bar{\mathbf{x}}^{(test)} + 1)^2$$

$$= [(0.2936 + 1)^2 \quad (0.5792 + 1)^2 \quad (0.4080 + 1)^2 \quad (0.9720 + 1)^2]$$

$$= [1.67 \quad 2.49 \quad 1.98 \quad 3.89]$$

#### *Step 8: Calculate test output*

For linear SVR,

$$f(\bar{\mathbf{x}}, \alpha^*, \alpha) = \sum_{i=1}^n (\alpha_i^* - \alpha_i) \mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}^{(test)}) + b$$

$$= \mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}^{(test)}) \beta + b = 0.4294$$

where  $b$  calculated by Eq. (9) is -0.3176.

For nonlinear SVR using the following polynomial kernel function,

$$f(\bar{\mathbf{x}}, \alpha^*, \alpha) = \sum_{i=1}^n (\alpha_i^* - \alpha_i) \mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}^{(test)})$$

$$= \mathbf{K}(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}^{(test)}) \beta = 0.4336$$

#### *Step 9: Recover actual output value*

Recover the actual output value of the normalized output using the following equation.

For linear SVR,

$$Y = \left( \frac{0.4294 - 0.1}{0.8} \right) \times (2.8 - 1.1) + 1.1 = 1.8000$$

For nonlinear SVR using the following polynomial kernel function,

$$Y = \left( \frac{0.4336 - 0.1}{0.8} \right) \times (2.8 - 1.1) + 1.1 = 1.8088$$

The equation is  $Y=2X_1+3X_2$  converted to  $\bar{Y}=0.4706\bar{X}_1+0.8824\bar{X}_2-0.3176$  through data normalization. Using the Lagrange multipliers for linear SVR obtained in Step 4, the parameter vectors of the regression function  $w$  and  $b$  are calculated from Eq. (9) as

$$w = [0.4706 \quad 0.8824], \quad b = -0.3176$$

It is noteworthy that the calculated linear SVR function is exactly the same as the normalized equation.

## References

- Aizerman, M. A., Braverman, E. M. and Rozonoer, L. I. (1964), "Theoretical foundations of the potential function method in pattern recognition learning", *Automation and Remote Control*, **25**, 821-837.
- Arora, J. S. (1989), *Introduction to Optimum Design*, McGraw-Hill, New York.
- Bazaraa, M. S., Sherali, H. D. and Shetty, C. M. (1993), *Nonlinear Programming: Theory and Algorithms*, 2<sup>nd</sup> Ed., John Wiley & Sons, Inc., New York.
- Cherkassky, V. and Ma, Y. (2004), "Practical selection of SVM parameters and noise estimation for SVM regression", *Neural Network*, **17**, 113-126.
- Cherkassky, V. and Mulier, F. (1998), *Learning from data: Concepts, theory, and methods*, Wiley, New York.
- Gunn, S. R. (1998), "Support vector machines for classification and regression", Technical Report ISIS-1-98, University of Southampton.
- Ji, T. and Lin, X. J. (2006), "A mortar mix proportion design algorithm based on artificial neural networks", *Comput. Concrete*, **3**(5).
- Kim, J. I., Kim, D. K., Feng, M. Q. and Yazdani, F. (2004), "Application of neural networks for estimation of concrete strength", *J. Mater. Civ. Eng.*, ASCE, **16**(3), 257-264.
- Lee, S. C. (2003), "Prediction of concrete strength using artificial neural networks", *Eng. Struct.*, **25**, 849-857.
- Madan, A. (2005), "Vibration control of building structures using self-organizing and self-learning neural networks", *J. Sound Vib.*, **287**(4/5), 759-784.
- Mukherjee, S., Osuna, E. and F. Girosi. (1997), "Nonlinear prediction of chaotic time series using support vector machines", *Proceedings of IEEE NNSP'97 Amelia Island, FL*.
- Muller, K. R., Smola, A., Ratsch, G., Scholkopf, B., Kohlmorgen, J. and V. Vapnik. (1997) "Predicting time series with support vector machines", *Proceedings of ICNN'97, Lausanne*.
- Oztas, A., Pala, M., Ozbay, E., Kanca, E., Caglar, N. and Asghar Bhatti, M. (2006), "Predicting the compressive strength and slump of high strength concrete using neural network", *Construction and Building Materials*, **20**(9), 769-775.
- Popovics, S. (1998), "History of a mathematical model for strength development of Portland cement concrete", *ACI Mater. J.*, **95**(5), 593-600.
- Snell, L. M., Van Roekel, J. and Wallace, N. D. (1989), "Predicting early concrete strength", *Concrete International*, **11**(12), 43-47.
- Stitson, M. O., Weston, J. A. E., Gammerman, A , Vork, V. and Vapnik, V. (1996), "Theory of support vector

- machines”, Technical Report CSD-TR-96-17, Department of Computer Science, Royal Holloway College, University of London.
- Vapnik, V. (1995), *The Nature of Statistical Learning Theory*, Springer Berlag, New York.
- Vapnik, V. (1999a), *The Nature of Statistical Learning Theory*, 2<sup>nd</sup> edition, Springer Berlag, New York.
- Vapnik, V. (1999b), “An overview of statistical learning theory”, *IEEE Transactions on Neural Networks*, **10**(5), 988-999.
- Ye, Q., Huang, Q., Gao, W. and Zhao, D. (2005), “Fast and robust text detection in images and video frames”, *Image Vis. Comp.*, **23**(6), 565-576.
- Yu, P. S., Chen, S. T. and Chang, I. F (2006), “Support vector regression for the real-time flood stage forecasting”, *J. Hydrology*, **328**(3-4), 704-716.
- Zhang, J., Sato, T. and Iai, S (2006), “Support vector regression for on-line health monitoring of large-scale structures”, *Structural Safety*, **28**(4), 392-406.

CC