Abstract—Support vector machines (SVMs) have been very successful in pattern classification and function approximation problems for crisp data. In this paper, we incorporate the concept of fuzzy set theory into the support vector regression machine. The parameters to be estimated in the SVM regression, such as the components within the weight vector and the bias term, are set to be the fuzzy numbers. This integration preserves the benefits of SVM regression model and fuzzy regression model and has been attempted to treat fuzzy nonlinear regression analysis. In contrast to previous fuzzy nonlinear regression models, the proposed algorithm is a model-free method in the sense that we do not have to assume the underlying model function. By using different kernel functions, we can construct different learning machines with arbitrary types of nonlinear regression functions. Moreover, the proposed method can achieve automatic accuracy control in the fuzzy regression analysis task. The upper bound on number of errors is controlled by the user-predefined parameters. Experimental results are then presented that indicate the performance of the proposed approach.

Index Terms—Fuzzy modeling, fuzzy regression, quadratic programming, support vector machines (SVMs), support vector regression machines.

I. INTRODUCTION

In many real-world applications, available information is often uncertain, imprecise, and incomplete and thus usually is represented by fuzzy sets or a generalization of interval data. For handling interval data, fuzzy regression analysis is an important tool and has been successfully applied in different applications such as market forecasting [19] and system identification [24]. Fuzzy regression, first developed by Tanaka et al. [30] in a linear system, is based on the extension principle. In the experiments that followed this pioneering effort, Tanaka et al. [31] used fuzzy input experimental data to build fuzzy regression models. The process was explained in more detail by Dubois and Prade in [16]. A technique for linear least squares fitting of fuzzy variables was developed by Diamond [11], [12] giving the solution to an analog of the normal equation of classical least squares. A collection of relevant papers dealing with several approaches to fuzzy regression analysis can be found in [23]. In contrast to the fuzzy linear regression, there have been only a few articles on fuzzy nonlinear regression [2], [3], [6]. They usually assume the underlying model functions and treat the estimation procedures of some particular models such as linear, polynomial, exponential, and logarithmic.

The support vector machines (SVMs), developed at AT&T Bell Laboratories by Vapnik et al. [9], [17], [35], have been very successful in pattern classification and function estimation problems for crisp data. They are based on the idea of structural risk minimization, which shows that the generalization error is bounded by the sum of the training error and a term depending on the Vapnik-Chervonenkis dimension. By minimizing this bound, high generalization performance can be achieved. A comprehensive tutorial on SVM classifier has been published by Burges [4]. Excellent performances were also obtained in the function estimation and time-series prediction applications [14], [27], [36]. Hong et al. [20], [21] first introduced the use of SVM for multivariate fuzzy linear and nonlinear regression models. The fuzzy support vector regression machine proposed by Hong [20] was achieved by solving a quadratic programming with box constraints. Jeng et al. [22] also applied the SVM to the interval regression analysis. They proposed a two-step approach for the interval regression by constructing two radial basis function (RBF) networks, each identified the lower side and upper side of data interval, respectively. The initial structure of the RBF network is obtained by SVM learning approach. Consequently, a traditional backpropagation learning algorithm is used to adjust the network.

In this paper, we incorporate the concept of fuzzy set theory into the SVM regression model. The parameters to be identified in the SVM regression model, such as the components of weight vector and bias term, and the desired outputs in training samples are set to be the fuzzy numbers. This integration preserves the benefits of SVM regression and fuzzy regression, where the Vapnik-Chervonenkis theory (also known as VC theory) [35] characterizes the properties of learning machines, which enable them to generalize well in the unseen data, whereas the fuzzy set theory might be very useful for finding a fuzzy structure in an evaluation system. The proposed fuzzy SVM regression analysis was achieved by solving a convex quadratic programming with linear constraints; in other words, it has a unique solution. In addition, the proposed algorithm here is model-free method in the sense that we do not have to assume the underlying model function. By the choice of different kernel functions, we obtain different architectures of the nonlinear regression functions, such as polynomial regression functions, RBF regression functions. This model-free method turns out to be a promising method that has been attempted to treat fuzzy nonlinear regression analysis. Moreover, the proposed method can
achieve automatic accuracy control in the fuzzy regression analysis task. The number of errors in the obtained regression model is bounded by the user-predefined parameters.

The rest of this paper is organized as follows. A brief review of the theory of SVM regression is given in Section II. The fuzzy SVM regression is derived in Section III. Experiments are presented in Section IV. Some concluding remarks are given in Section V.

II. SVM REGRESSION MODEL

Suppose we are given a training data set \( \{(x_1, y_1), \ldots, (x_N, y_N)\} \subset \mathbb{R} \times R \), where \( \mathbb{R} \) denotes the space of input patterns, for instance, \( R^m \). In \( \varepsilon \)-SVM regression [14], [27], [35] the goal is to find a function \( f(x) \) that has at most \( \varepsilon \) deviation from the actually obtained targets \( y_i \) for all the training data. In other words, we do not care about errors as long as they are less than \( \varepsilon \) but will not accept any deviation larger than \( \varepsilon \). An \( \varepsilon \)-insensitive loss function

\[
|\varepsilon|_\varepsilon := \begin{cases} 
0, & \text{if } |\varepsilon| \leq \varepsilon \\
|\varepsilon| - \varepsilon, & \text{otherwise}
\end{cases}
\]

is used so that the error is penalized only if it is outside the \( \varepsilon \)-tube. Fig. 1 depicts this situation graphically.

To make the SVM regression nonlinear, this could be achieved by simply mapping the training patterns \( x_i \) by \( \Phi: \mathbb{R} \rightarrow F \) into some high-dimensional feature space \( F \). A best fitting function \( f(x) = (\mathbf{w} \cdot \Phi(x)) + b \) is estimated in that feature space \( F \). To avoid overfitting, one should add a capacity control term, which in the SVM case results to be \( \|\mathbf{w}\|^2 \).

Formally, we can write this problem as a convex optimization problem by requiring

\[
\begin{align*}
& \text{minimize}_{\mathbf{w}, b, \xi_1, \xi_2} & & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} (\xi_{1i} + \xi_{2i}) \\
& \text{subject to} & & y_i - (\mathbf{w} \cdot \Phi(x_i)) + b \leq \varepsilon - \xi_{1i} \\
& & & (\mathbf{w} \cdot \Phi(x_i)) + b - y_i \leq \varepsilon - \xi_{2i} \\
& & & \xi_{1i}, \xi_{2i} \geq 0 \quad \forall i.
\end{align*}
\]

The constant \( C > 0 \) determines the tradeoff between the complexity of \( f(x) \) and the amount up to which deviations larger than \( \varepsilon \) are tolerated. In short, minimizing the objective function given in (1) captures the main insight of statistical learning theory, stating that in order to obtain a small risk, we need to control both training error and model complexity, by explaining the data with a simple model [35].

Using the Lagrange multiplier method, this quadratic programming problem can be formulated as the Wolfe dual form shown in (2) at the bottom of the page, where \( \alpha_{1i}, \alpha_{2i} \) are the nonnegative Lagrange multipliers. Solving the above dual quadratic programming problem, we obtain the Lagrange multipliers \( \alpha_{1i} \) and \( \alpha_{2i} \), which give the weight vector \( \mathbf{w} \) as a linear combination of \( \Phi(x_i) \)

\[
\mathbf{w} = \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) \Phi(x_i),
\]

maximize \( \alpha_{1i}, \alpha_{2i} \)

\[
\begin{align*}
& \text{subject to} & & \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) = 0 \\
&& & \alpha_{1i}, \alpha_{2i} \in [0, C]
\end{align*}
\]

(2)

Fig. 1. The epsilon insensitive loss setting corresponding to a linear SV regression machine.
The training points \( x_i \) for which \((\alpha_{1i} - \alpha_{2i}) \neq 0\) are termed support vectors since only those points determine the final regression result among all training points. Knowing \( w \), we can subsequently determine the bias term \( b \) by exploiting the Karush–Kuhn–Tucker (KKT) conditions. Hence \( b \) can be computed as follows:

\[
b = y_i - \langle w \cdot \Phi(x_i) \rangle + \varepsilon \quad \text{for } \alpha_{1i} \in (0, C) \]
\[
b = y_i - \langle w \cdot \Phi(x_i) \rangle - \varepsilon \quad \text{for } \alpha_{2i} \in (0, C),
\]

A key property of the SVM is that the only quantities that one needs to compute are scalar products, of the form \( \langle \Phi(x) \cdot \Phi(y) \rangle \). It is therefore convenient to introduce the so-called kernel function \( k \):

\[
k(x, y) = \langle \Phi(x) \cdot \Phi(y) \rangle.
\]

The definition of kernel function \( k \) prevents the direct computation of inner production in the high-dimensional feature space, which is very time-consuming and makes the computation practical. Using this quantity, the solution of a SVM has the form

\[
f(x) = \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) \langle \Phi(x_i) \cdot \Phi(x) \rangle + b
\]
\[
= \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) k(x_i, x) + b.
\]

To motivate the new algorithm that we shall propose, note that the parameter \( \varepsilon \) can be useful if the desired accuracy of the approximation can be specified beforehand. The selection of a parameter \( \varepsilon \) may seriously affect the modeling performance. In addition, the \( \varepsilon \)-insensitive zone in the SVM regression model has a tube (or slab) shape. Namely, all training data points are equally treated during the training of SVM regression model and are penalized only if they are outside the \( \varepsilon \)-tube. In many real-world applications, however, the effects of the training points are different. We would require a learning machine that must estimate the precise training points correctly and would allow more errors on the imprecise training points. In the next section, we incorporate the fuzzy set theory into the SVM regression task. The proposed fuzzy SVM regression takes the imprecision of training points into consideration. The vagueness (or insensitivity) of the obtained regression model depends on the vagueness of the given training points.

### III. Fuzzy SVM Regression Model

In modeling some systems where available information is uncertain, we must deal with a fuzzy structure of the system considered. This structure is represented as a fuzzy function whose parameters are given by fuzzy sets. The fuzzy functions are defined by Zadeh’s extension principle [15], [25], [37], [38]. Basically, the fuzzy function provides an effective means of capturing the approximate, inexact nature of real world. In this section, we incorporate the concept of fuzzy set theory into the SVM regression model. The parameters to be identified in the SVM regression model, such as the components of weight vector and bias term, and the desired outputs in training samples are set to be the fuzzy numbers. For computational simplicity, we assume the fuzzy parameters to be identified are symmetric triangular fuzzy numbers.

#### A. The Quadratic Programming Problem

First, the components in weight vector and bias term used in the function regression model are fuzzy numbers. Given the fuzzy weight vector \( W = (w, c) \) and fuzzy bias term \( B = (b, d) \), \( W = (w, c) \) is the fuzzy weight vector, where each component within it \( W_i = (w_i, c_i) \) is symmetric triangular fuzzy numbers. It was denoted in the vector form of \( w = [w_1, \ldots, w_n] \) and \( c = [c_1, \ldots, c_n] \), which means “approximation \( w_\varepsilon \)” described by the center \( w \) and the width \( c \). Similarly, \( B = (b, d) \) is the fuzzy bias term, which means “approximation \( b_\varepsilon \)” described by the center \( b \) and the width \( d \). The fuzzy function

\[
Y = W_1 x_1 + \cdots W_n x_n + B = \langle W \cdot x \rangle + B
\]

is defined by the following membership function [31]:

\[
\mu_Y(y) = \begin{cases} 
1 - \frac{|y - \langle w \cdot x \rangle| + d}{c + d}, & \text{if } x \neq 0 \\
1, & \text{if } x = 0 \text{ and } y = 0 \\
0, & \text{if } x = 0 \text{ and } y \neq 0
\end{cases}
\]

where \( \mu_Y(y) = 0 \) when \( \langle c \cdot |x| \rangle + d \leq |y - (w \cdot x) + b| \).

Then, we deal with fuzzy desired output in the regression task. The given output data, denoted by \( \tilde{Y}_i = (y_i, e_i) \), are also fuzzy numbers, where \( y_i \) is a center and \( e_i \) is the width. The membership function of \( \tilde{Y}_i \) is given by

\[
\mu_{\tilde{Y}_i}(y) = 1 - \frac{|y - y_i|}{e_i}.
\]

To formulate a fuzzy linear regression model, the following are assumed to hold.

1) The data can be represented by a fuzzy linear model

\[
Y^{*}_i = \langle W^{*} \cdot x_i \rangle + B^{*}.
\]

Given \( x_i \), \( Y^{*}_i \) can be obtained as

\[
\mu_{Y^{*}_i}(y) = 1 - \frac{|y - \langle w \cdot x_i \rangle + b|}{c + d}.
\]

2) The degree of the fitting of the estimated fuzzy linear model \( Y^{h}_i = \langle W^{h} \cdot x_i \rangle + B^{h} \) to the given data \( \tilde{Y}_i = (y_i, e_i) \) is measured by the following index \( h_i \), which maximizes \( h \) subject to \( Y_i^{h} \subset Y_i^{*h} \), where

\[
Y_i^{h} = \{ y | \mu_{Y_i}(y) \geq h \}
\]
\[
Y_i^{*h} = \{ y | \mu_{Y^{*}_i}(y) \geq h \}
\]

which are \( h \)-level sets. The degree of fitting of the linear model to all data \( \tilde{Y}_1, \ldots, \tilde{Y}_N \) is defined by \( \min_j \bar{f}_{ij} \).

3) The vagueness of the fuzzy linear model is defined by

\[
\frac{1}{2} ||c||^2 + d.
\]

4) The capacity control term of the fuzzy linear model is defined by \( ||w||^2 \).

The regression problem is explained as obtaining the fuzzy weight vector \( W^{*} = (w, c) \) and the fuzzy bias term \( B^{*} = (b, d) \), such that the fitting degree between the estimated output
μY_i∗(y) and desired output μY_i∗(y) is more than a given constant H for all i, where H is chosen as the degree of the fitting of the fuzzy linear model by the decision-maker. The value h_i can be obtained from

$$h_i = 1 - \frac{\|k_i - (\langle w \cdot x_i \rangle + b)\|}{(c \cdot |x_i| + d) - e_i}$$

(4)

which is equal to the early work by Tanaka [31] and is illustrated in Fig. 2.

Our regression task here is therefore to minimize

$$J = \frac{1}{2}\|w\|^2 + K \left( \frac{1}{2}\|c\|^2 + d \right)$$

subject to

$$h_i \geq H - \xi_i$$

for all i = 1, ..., N

(5)

where 2\|w\|^2 is the term that characterizes the model complexity, the minimization of \(2\|w\|^2\) can be understood in the context of regularization operators [28] and \((1/2)\|c\|^2 + d\) is the term that characterizes the vagueness of the model. More vagueness in the fuzzy regression model means more inexactness in the regression result. K is a tradeoff parameter chosen by the decision-maker. The value of H determines the low bound for the degree of fitting of the fuzzy linear model and \(h_i\) is the degree of fitting of the estimated fuzzy linear model \(Y_i^∗ = \langle w^∗ \cdot x_i \rangle + B^∗\) to the given fuzzy desired output data \(Y_i = (y_i, e_i)\). If \(\xi_i\) are sets of surplus variables that measure the amount of variation of the constraints for each point where \(P\) is a fixed penalty parameter chosen by the user, a larger \(P\) corresponds to assigning a higher penalty to errors. Fig. 3 depicts this situation graphically.

More specifically, according to (4), our problem is to find out the fuzzy weight vector \(W^∗ = (w, c)\) and fuzzy bias term \(B^∗ = (b, d)\), which is the solution of the following quadratic programming problem:

$$J = \frac{1}{2}\|w\|^2 + K \left( \frac{1}{2}\|c\|^2 + d \right)$$

$$+ P \sum_{i=1}^{N} (\xi_{1i} + \xi_{2i})$$

(6)

subject to

$$(\langle w \cdot x_i \rangle + b) + (1 - H)(c \cdot |x_i| + d)$$

$$\geq y_i + (1 - H)e_i - \xi_{1i}$$

$$- (\langle w \cdot x_i \rangle + b) + (1 - H)(c \cdot |x_i| + d)$$

$$\geq -y_i + (1 - H)e_i - \xi_{2i}$$

and

$$d \geq 0, c_j \geq 0, \xi_{1j} , \xi_{2j} \geq 0, \text{ for } i = 1, ..., N, j = 1, ..., n.$$ (7)

Comparing Fig. 3 with Fig. 1, we can understand the difference between SVM regression model and fuzzy SVM regression model. The SVM regression model seeks a linear function that has at most \(\varepsilon\) deviation from the actually obtained targets \(y_i\) for all the training data, whereas the fuzzy SVM regression model seeks a fuzzy linear function with fuzzy parameters that has at least \(H\) fitting degree from the fuzzy desired targets \(Y_i\) for all the training data.

We can find the solution of this optimization problem given by (6) in dual variables by finding the saddle point of the Lagrangian

$$L = \frac{1}{2}\|w\|^2 + K \left( \frac{1}{2}\|c\|^2 + d \right)$$

$$+ P \sum_{i=1}^{N} (\xi_{1i} + \xi_{2i})$$

$$- \sum_{i=1}^{N} \alpha_{1i}(\langle w \cdot x_i \rangle + b) + (1 - H)(c \cdot |x_i| + d)$$

$$- y_i + (1 - H)e_i + \xi_{1i}$$

$$- \sum_{i=1}^{N} \alpha_{2i}(-\langle w \cdot x_i \rangle - b) + (1 - H)(c \cdot |x_i| + d)$$

$$+ y_i - (1 - H)e_i + \xi_{2i}$$

$$- \sum_{i=1}^{N} \rho_{1i}\xi_{1i} - \sum_{i=1}^{N} \rho_{2i}\xi_{2i} - \gamma d$$

(7)
where $\alpha_{1i}, \alpha_{2i}, \rho_{1i}, \rho_{2i}$, and $\gamma$ are the nonnegative Lagrange multipliers. Differentiating $L$ with respect to $w, c, b, d, \xi_{1i}$, and $\xi_{2i}$ and setting the results to zero, we obtain

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) x_i$$

(8)

$$\frac{\partial L}{\partial c} = 0 \Rightarrow c = \frac{1 - H}{K} \sum_{i=1}^{N} (\alpha_{1i} + \alpha_{2i}) |x_i|$$

(9)

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) = 0$$

(10)

$$\frac{\partial L}{\partial d} = 0 \Rightarrow \sum_{i=1}^{N} (\alpha_{1i} + \alpha_{2i}) = \frac{K - \gamma}{1 - H}$$

(11)

$$\frac{\partial L}{\partial \xi_{1i}} = 0 \Rightarrow \alpha_{1i} = P - \rho_{1i} \text{ and } \alpha_{1i} \leq P$$

(12)

$$\frac{\partial L}{\partial \xi_{2i}} = 0 \Rightarrow \alpha_{2i} = P - \rho_{2i} \text{ and } \alpha_{2i} \leq P.$$ 

(13)

Substituting (8)–(13) into (7), we obtain

$$L = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_{1i} - \alpha_{2i})(\alpha_{1j} - \alpha_{2j})(x_i \cdot x_j)$$

$$- \frac{(1 - H)^2}{2K} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_{1i} + \alpha_{2i})(\alpha_{1j} + \alpha_{2j})(|x_i| \cdot |x_j|)$$

$$+ \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) y_i + (1 - H) \sum_{i=1}^{N} (\alpha_{1i} + \alpha_{2i}) e_i.$$ 

Therefore, the dual problem is

$$\text{maximize} \quad \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_{1i} - \alpha_{2i})(\alpha_{1j} - \alpha_{2j})(x_i \cdot x_j)$$

$$- \frac{(1 - H)^2}{2K} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_{1i} + \alpha_{2i})(\alpha_{1j} + \alpha_{2j})(|x_i| \cdot |x_j|)$$

$$+ \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) y_i + (1 - H) \sum_{i=1}^{N} (\alpha_{1i} + \alpha_{2i}) e_i,$$

subject to

$$\sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) = 0, \quad \sum_{i=1}^{N} (\alpha_{1i} + \alpha_{2i}) \leq \frac{K}{1 - H},$$

$$\alpha_{1i}, \alpha_{2i} \in [0, P].$$

(14)

As for the pair $(w, c, b, d)$, it is easy to find that

$$w = \sum_{i=1}^{N} (\alpha_{1i} - \alpha_{2i}) x_i$$

$$c = \frac{1 - H}{K} \sum_{i=1}^{N} (\alpha_{1i} + \alpha_{2i}) |x_i|.$$ 

While parameters $b$ and $d$ can be determined from the KKT conditions

$$\alpha_{1i} \left( \langle w \cdot x_i \rangle + b + (1 - H)(\langle c \cdot |x_i| \rangle + d) \right) = 0,$$

$$\alpha_{2i} \left( -\langle w \cdot x_i \rangle - b + (1 - H)(\langle c \cdot |x_i| \rangle + d) \right) = 0$$

(15)

$$P - \alpha_{1i} \xi_{1i} = 0,$$

$$P - \alpha_{2i} \xi_{2i} = 0.$$ 

(16)

(17)

(18)

For some $\alpha_{1i}, \alpha_{2j} \in (0, P)$, we have $\xi_{1i} = \xi_{2j} = 0$, and moreover the second factor in (15) and (16) has to vanish. Hence, $b$ and $d$ can be computed as shown in (19) and (20) at the bottom of the page, for some $\alpha_{1i}, \alpha_{2j} \in (0, P)$.

The fuzzy linear regression function is defined by the membership function shown in (21) at the bottom of the page, where $\mu_{\gamma^{*}}(y) = 0$ when

$$b = -\frac{1}{2} \left( \langle w \cdot x_i \rangle + \langle w \cdot x_j \rangle + (1 - H)(\langle c \cdot |x_i| \rangle - \langle c \cdot |x_j| \rangle) \right)$$

$$+ y_i - y_j - (1 - H)(e_i - e_j)$$

$$d = -\frac{1}{2(1 - H)} \left( \langle w \cdot x_i \rangle - \langle w \cdot x_j \rangle + (1 - H)(\langle c \cdot |x_i| \rangle + \langle c \cdot |x_j| \rangle) \right)$$

$$+ y_i + y_j - (1 - H)(e_i + e_j)$$

$$\mu_{\gamma^{*}}(y) = \begin{cases} 
1 - \frac{|w| \sum_{i=1}^{N} (\alpha_{1k} - \alpha_{2k})(|x_i| + |x_k|) + b|}{(1 - H) \sum_{i=1}^{N} (\alpha_{1i} + \alpha_{2i})(|x_i| + |x_k|) + d}, & \text{if } x \neq 0 \\
1, & \text{if } x = 0 \text{ and } y = 0 \\
0, & \text{if } x = 0 \text{ and } y \neq 0
\end{cases}$$

(19)

(20)

(21)
sum of squared distances of given outputs and the estimation center decreases as the dimension of the feature space increases. It is reasonable since as the dimension of the feature space increases, the learning machine has more capacity to construct a complex regression model, and hence the accuracy of the regression result is improved. In addition, values of $k_T$ also show a decreasing trend on the order of Table V(a)–(d), which tells that the sum of squared spreads decreases as the dimension of the feature space increases. It is indicative of the fact that the vagueness of the obtained fuzzy regression model decreases as the capacity of the learning machine increases. On the contrary, the numbers of support vectors show an increasing trend on the order of Table V(a)–(d), which tells that we need more support vectors to construct a more complex regression model.

Fig. 7(a)–(d) illustrates the obtained fuzzy regression models with different kernel functions. It can be noticed that Fig. 7(d) has more central tendency than Fig. 7(a)–(c) because the observations are all inside the estimated interval and the spreads of regression model in Fig. 7(d) are smaller than those in Fig. 7(a)–(c). It can also be noticed that the proposed algorithm is a model-free method. As described in Section III-C, we can actually use a larger class of kernels without destroying the mathematical formula of the quadratic programming problem given by (22). By choosing different kernels, we obtain different architectures of the nonlinear regression models.

V. CONCLUSION

In this paper, we introduced fuzzy regression analysis by support vector learning approach. The difference between the original SVM regression and the proposed fuzzy SVM regression is that the SVM approach seeks a linear function that has at most $\varepsilon$ deviation from the actually obtained targets $y_i$ for all the training data, whereas the proposed fuzzy SVM approach seeks a fuzzy linear function with fuzzy parameters that has at least $H$ fitting degree from the fuzzy desired targets $\tilde{Y}_i$ for all the training data. Incorporating the concept of fuzzy set theory into the SVM regression preserves the benefits of SVM regression and fuzzy regression, where the VC theory characterizes the properties of learning machines, which enable them to generalize well in the unseen data, whereas the fuzzy set theory might be very useful for finding a fuzzy structure in an evaluation system.

The main difference between our fuzzy SVM approach and the nonlinear fuzzy regression approaches by Buckley et al. [2], [3] and Celmins [6] is not crisp input-fuzzy output versus fuzzy input-fuzzy output but model-free versus model-dependent. By the choice of different kernels, we can construct different learning machines with arbitrary types of nonlinear regression functions in the input space. The model-free method turned out to be a promising method attempted to treat a fuzzy nonlinear regression task. Moreover, the proposed method can achieve automatic accuracy control in the fuzzy regression analysis task. The upper bound on number of errors is controlled by the user-predefined parameters.

In this paper, we deal with estimating fuzzy nonlinear regression model with crisp inputs and fuzzy output. In future work, we intend to devise algorithms for estimating fuzzy nonlinear regression model with fuzzy inputs and fuzzy output.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

REFERENCES


TABLE V

<table>
<thead>
<tr>
<th>Case</th>
<th>Kernel Function</th>
<th>Dimension of Feature Space</th>
<th>Central Tendency $\phi_Y$</th>
<th>Vagueness $k_T$</th>
<th>Num. of SVs</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Linear</td>
<td>3</td>
<td>41.9485</td>
<td>14038.0</td>
<td>4</td>
</tr>
<tr>
<td>(b)</td>
<td>Polynomial ($d=2$)</td>
<td>10</td>
<td>30.4739</td>
<td>5964.7</td>
<td>6</td>
</tr>
<tr>
<td>(c)</td>
<td>Polynomial ($d=4$)</td>
<td>35</td>
<td>22.7203</td>
<td>2604.2</td>
<td>8</td>
</tr>
<tr>
<td>(d)</td>
<td>RBF ($\tau=10$)</td>
<td>Infinite</td>
<td>21.6841</td>
<td>2250.0</td>
<td>12</td>
</tr>
</tbody>
</table>


**Pey-Yi Hao** received the B.Sc. degree in mathematics and the Ph.D. degree in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 1999 and 2003, respectively. He is currently an Associate Professor in the Department of Information Management, National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan. His research interests include fuzzy theory, neural networks, pattern recognition, support vector machines, and modeling of bioinformatics problems.

**Jung-Hsien Chiang** (M’92–SM’05) received the B.Sc. degree in electrical engineering from the National Taiwan Institute of Technology, Taiwan, R.O.C., and the M.S. and Ph.D. degrees in computer engineering from the University of Missouri, Columbia, in 1991 and 1995, respectively. He is currently a Professor in the Department of Computer Science and Information Engineering, National Cheng Kung University, Taiwan. Previously, he was a Researcher in the Computer and Communication Laboratory, Industrial Technology Research Institute, the largest information technology research institute in Taiwan. His current research interests include fuzzy modeling, neural networks, pattern clustering, and modeling of bioinformatics problems.