Improving the Interpretability of TSK Fuzzy Models by Combining Global Learning and Local Learning

John Yen, *Senior Member, IEEE*, Liang Wang, *Member, IEEE*, and Charles Wayne Gillespie

***Abstract*— The fuzzy inference system proposed by Takagi, Sugeno, and Kang, known as the TSK model in fuzzy system literature, provides a powerful tool for modeling complex nonlinear systems. Unlike conventional modeling where a single model is used to describe the global behavior of a system, TSK modeling is essentially a *multimodel* approach in which simple submodels (typically linear models) are combined to describe the global behavior of the system. Most existing learning algorithms for identifying the TSK model are based on minimizing the square of the residual between the overall outputs of the real system and the identified model. Although these algorithms can generate a TSK model with good global performance (i.e., the model is capable of approximating the given system with arbitrary accuracy, provided that sufficient rules are used and sufficient training data are available), they cannot guarantee the resulting model to have a good local performance. Often, the submodels in the TSK model may exhibit an erratic local behavior, which is difficult to interpret. Since one of the important motivations of using the TSK model (also other fuzzy models) is to gain insights into the model, it is important to investigate the interpretability issue of the TSK model. In this paper, we propose a new learning algorithm that integrates global learning and local learning in a single algorithmic framework. This algorithm uses the idea of local weighed regression and local approximation in nonparametric statistics, but remains the component of global fitting in the existing learning algorithms. The algorithm is capable of adjusting its parameters based on the user's preference, generating models with good tradeoff in terms of global fitting and local interpretation. We illustrate the performance of the proposed algorithm using a motorcycle crash modeling example.**

***Index Terms*— Fuzzy modeling, fuzzy systems, learning algorithms, TSK model.**

I. Introduction

THE method of fuzzy inference proposed by Takagi, Sugeno and Kang [19], [21], which is known as the Takagi–Sugeno–Kang (TSK) model in fuzzy systems literature, has been one of the major topics in theoretical studies and practical applications of fuzzy modeling and control. The basic idea of this method is to decompose the input space into fuzzy regions and to approximate the system in every region by a simple model. The overall fuzzy model is thus considered as a combination of interconnected subsystems with simpler models.

Manuscript received September 2, 1997. This work was supported by National Science Foundation Young Investigator Award IRI-9257293.

The authors are with the Center for Fuzzy Logic, Robotics, and Intelligent Systems, Department of Computer Science, Texas A&M University, College Station, TX 77843 USA.

Publisher Item Identifier S 1063-6706(98)08262-9.

Typically, a TSK model consists of IF–THEN rules that have the form

$$R_i \text{ if } x_1 \text{ is } A_{i1} \text{ and } \cdots \text{ and } x_r \text{ is } A_{ir} \text{ then } y_i = b_{i0} + b_{i1}x_1 + \cdots + b_{ir}x_r \quad \text{for } i = 1, 2, \cdots, L \tag{1}$$

where L is the number of rules, x_i are input variables, y_i are local output variables, A_{ij} are fuzzy sets that are characterized by membership functions $A_{ij}(x_j)$, and b_{ij} are real-valued parameters. The overall output of the model is computed by

$$y = \frac{\sum_{i=1}^{L} \tau_i y_i}{\sum_{i=1}^{L} \tau_i} = \frac{\sum_{i=1}^{L} \tau_i (b_{i0} + b_{i1}x_1 + \cdots + b_{ir}x_r)}{\sum_{i=1}^{L} \tau_i} \tag{2}$$

where τ_i is the *firing strength* of rule R_i, which is defined as

$$\tau_i = A_{i1}(x_i) \times A_{i2}(x_2) \times \cdots \times A_{ir}(x_r). \tag{3}$$

The great advantage of the TSK model is its representative power; it is capable of describing a highly nonlinear system using a small number of rules. Moreover, since the output of the model has an explicit functional expression form (2), it is conventional to identify its parameters using some learning algorithms. Several commonly used fuzzy neuro systems such as adaptive neuro-fuzzy inference system (ANFIS) [10] and that in Takagi and Hayashi [20] have been constructed on the basis of the TSK model.

One major concern in building a TSK model is how well the model can approximate a real system. Most existing learning algorithms [1], [10], [21], [23] choose the parameters of the model to minimize the objective function J_G

$$J_G = \sum_{k=1}^{N} [d(k) - y(k)]^2 \tag{4}$$

where $d(k)$ is the output of the real system, $y(k)$ is the output of the identified model, and N is the number of training data. These algorithms are *global* algorithms in the sense that the parameters of the model are identified using the whole training data set in a single algorithmic operation. If sufficient rules and training data are used, the resulting model from these algorithms is guaranteed to converge to the real system.

While these global learning algorithms can lead to a TSK model with arbitrary approximation accuracy, they cannot always guarantee the model to be locally well behaved. To illustrate this, we use a three-rule TSK model to fit the target function

$$y = (x - 10)^2. \tag{5}$$

Suppose the input x varies in the range of $[0, 20]$. The input

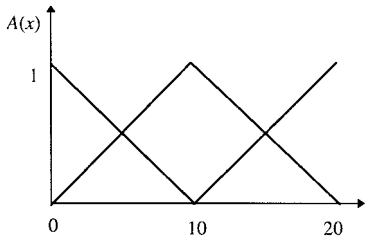

Fig. 1. Antecedent membership functions.

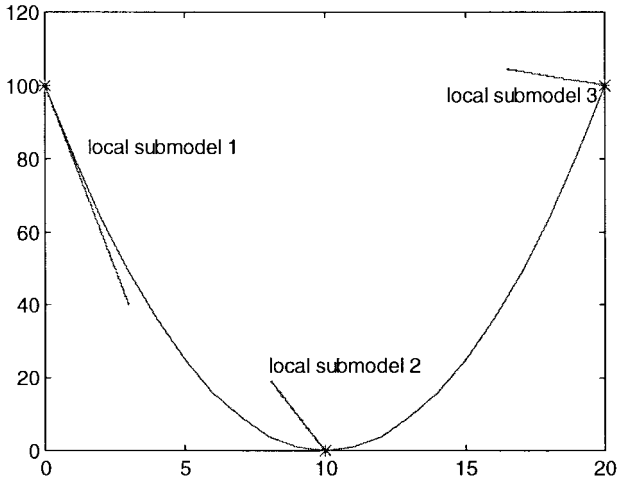

Fig. 2. TSK model with inappropriate local models.

space is partitioned into three fuzzy regions with membership functions shown in Fig. 1. In Fig. 2, we present a globally optimal TSK model. The local linear submodels in the model are indicated by the straight lines, with the centers of the triangular membership functions depicted by the small stars. This model fits the target function precisely, but the behaviors of the local submodels two and three are difficult to interpret. In general, a large number of TSK fuzzy models with identical global behaviors can exist, because shifting the hyperplanes associated with adjacent fuzzy partitions in a complementary manner to compensate for each other results in equivalent fuzzy models [25]. Since one of the important motivations of using fuzzy models is to gain insights into the local behavior of the models, it is important to address the interpretability issue of the TSK model.

Before presenting our results, it is helpful to discuss several existing methods that are relevant to this paper. *Locally weighted regression* (LWR) is a way of constructing regression surface through a multivariate smoothing procedure [2]. In such a scheme, the estimate of the regression surface at any value $\mathbf{x}_0$ of the input variables is obtained by local fitting of a polynomial function of degree d (d small) of the input variables. That is, LWR defines a neighborhood in the space of the input variables. Each point in the neighborhood is weighted according to its distance from $\mathbf{x}_0$; points close to $\mathbf{x}_0$ have large weight, and points far from $\mathbf{x}_0$ have small weight. The polynomial function of the input variables is fitted to the output variable using weighted least squares with these weights; the estimate of the regression surface is taken to be the value of this fitted function at $\mathbf{x}_0$.

LWR is a *smoother*.[1] Its purpose is to find the *trend* embedded in the input–output observations rather than seek the precise fit between the model and observations. Thus, the global behavior of the model is not its concern. Nevertheless, since LWR typically uses as many polynomial functions as data points, the resulting model usually has a good global behavior. Locally fitting a number of simple models to achieve a good global fitting is also the principle underlying the *local model networks* proposed in Murray–Smith and Johansen [15].

LWR requires a weight function and a specification of neighborhood size. The weight function in the original LWR is the *tricube* function: $w(x) = (1-x^3)^3$ for $0 \leq x < 1$ and 0 otherwise, but other form of weight functions such as Gaussian function can also be used [17]. The size of neighborhood (i.e., the number of data points in the neighborhood) should be not less than the degree d of the polynomial function for the weighted least-squares problem to have a unique solution. As the size of neighborhood increases, the estimate of regression surface becomes smoother.

The modeling philosophy of the LWR is quite similar to that of the TSK in the sense that both methods use the combinations of simple polynomial functions to construct a complex function. The main differences between the two methods lie in the determination of the parameters of polynomial functions and in the computation of the outputs of the model. In LWR, the parameters of each polynomial function are estimated using the data points in its neighborhood, and the output of the model at any point $\mathbf{x}_0$ is the output of a single polynomial function; in TSK, the parameters of each polynomial function are estimated using the data points in the whole input space and the output of the model at any point $\mathbf{x}_0$ is the weighted average of the outputs of all polynomial functions.

Another approach that is similar to the LWR is proposed in Farmer and Sidorowich [3] for predicting chaotic time series. This approach is called *local approximation* (LA) because it uses only nearby observations to make predictions. To be specific, let $\mathbf{x}(t)$ and $\mathbf{x}(t+T)$ denote the current observation and future observation, respectively. To predict $\mathbf{x}(t+T)$, LA first finds the k nearest neighbors of $\mathbf{x}(t)$, i.e., the k observations $\mathbf{x}(t')$ with $t' < t$ that minimize $||x(t) - x(t')||$, where $||\cdot||$ denotes an Euclidean norm. A polynomial function of degree d is then fitted through the k nearest neighbors, by ordinary least-squares algorithm. The only difference between LA and LWR is that LA uses an ordinary least-squares method to estimate the parameters of polynomial functions and no weight function is needed; LWR uses a weighted least-squares and, thus, requires a weight function. Several other methods that follow the same spirit of LA include *threshold models* [16], *codebook prediction* [18], and *competitive local linear models* [16].

[1] A smoother is a tool for summarizing the trend of an output measurement y as a function of one or more input measurements $x_1, x_2, \cdots, x_r$. It produces an estimate of the trend that is less variable than y itself; hence, the name of smoother. An important property of a smoother is its *nonparametric* nature: it does not assume a rigid form for the dependence of y on $x_1, x_2, \cdots, x_r$. For this reason, a smoother is often referred to as a tool for *nonparametric regression* [7].

In this paper, we develop a new learning algorithm to identify the TSK model. This algorithm is a *combined* algorithm in the sense that it integrates global learning and local learning in a single algorithmic framework. It uses the idea of local weighting and local approximating in LWR and LA, but remains the component of global fitting in the existing fuzzy model identification algorithms. The combination is done by weighting a global objective function and local objective function. This allows a user to adjust the algorithm based on his own preference, generalizing models with good tradeoff between global fitting and local interpretation. The detail of the combined algorithm is introduced in Section II.

Since the combined learning algorithm needs to construct linear regression models in different neighborhoods (fuzzy partitions), it is important to know the number and the position of these neighborhoods in the input space. In LWR and LA, the number of neighborhoods is typically taken as the number of data points and their position is the same as the position where the data points appear. In TSK, however, the neighborhoods are no longer directly relevant to the data points; as a result, the determination of the number and the position of the neighborhoods is not a trivial exercise. In Section III, we provide an orthogonal transformation based procedure to solve the problem. In Section IV, we apply the proposed combined learning algorithm to a simulated motor-cycle crash data set and compare its performance to the global learning algorithm and the local learning algorithm. Some concluding remarks are made in Section V.

II. Learning Algorithms

The task of learning algorithms is to estimate the parameters of the model, which include the parameters in the antecedent membership functions and the parameters in the linear regression equations. In order to lay down a foundation for the following derivation of learning algorithms, we rewrite (2) in a slightly different form

$$y = \sum_{i=1}^{L} w_i(b_{i0} + b_{i1}x_1 + \cdots + b_{ir}x_r) \tag{6}$$

where

$$w_i \equiv \frac{\tau_i}{\sum_{i=1}^{L} \tau_i} \tag{7}$$

is the *normalized* firing strength for rule i.

Equation (6) is nonlinear in the parameters and has to be solved using some nonlinear optimization algorithm such as the gradient descent algorithm [10]. However, if we fix the parameters in the antecedent membership functions (or equivalently the parameters in w_i) at the very beginning of model construction so that the only free parameters are those in the linear regression equations, then (6) is linear in the parameters. This is the case in the present paper where we predetermine the parameters of antecedent membership function using some heuristic technique.[2] The advantage of such a treatment is that the parameter identification problem is reduced to a simple linear optimization problem and, thus, can be solved using efficient linear learning algorithms [1], [23], [24].

A. Global Learning

Global learning determines the parameters of the model through minimizing the objective function J_G defined in (4), which can be rearranged into a simple matrix form

$$J_G = (\mathbf{d} - X\mathbf{b})^T(\mathbf{d} - X\mathbf{b}) \tag{8}$$

where

$$\mathbf{d} = [d(1) \quad d(2) \quad \cdots \quad d(N)]^T \in \Re^N \tag{9}$$

(See (10) and (11) at the bottom of the page.)

Because the parameters of antecedent membership functions are predetermined, the only unknown component in J_G is the parameter vector $\mathbf{b}$ whose elements are the parameters in the linear regression equations of the TSK model. The mostly commonly used method for computing $\mathbf{b}$ is the *recursive least-squares* (RLS) *algorithm* (see, e.g., [21]). The advantage of RLS is its on-line learning ability, but it requires a large number of iterations. In this paper, we use a computationally efficient, noniterative method to compute $\mathbf{b}$. The heart of the method is the *singular value decomposition* (SVD) [5]. In particular, applying SVD to X yields

$$X = U\Sigma V^T \tag{12}$$

where $U = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_N] \in \Re^{N\times N}$ and $V = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_{(r+1)\times L}] \in \Re^{[(r+1)\times L]\times[(r+1)\times L]}$ are orthogonal matrices $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \cdots, \sigma_{(r+1)\times L}) \in \Re^{N\times[(r+1)\times L]}$ is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{(r+1)\times L} \geq 0$. The diagonal elements of Σ are called the *singular values* of X. Substituting (12) into (8) and after simple algebraic manipulations, we can

[2] In this paper, we adopt Gaussian functions to express the antecedent membership functions in the TSK model. The centers of Gaussian functions are assumed to be uniformly distributed in the input space; the widths of Gaussian functions are determined using a nearest neighbors heuristic, as suggested in Moody and Darken [13].

$$X = \begin{bmatrix} w_1(1) & w_1(1)x_1(1) & \cdots & w_1(1)x_r(1) & \cdots & w_L(1) & w_L(1)x_1(1) & \cdots & w_L(1)x_r(1) \\ w_1(2) & w_1(2)x_1(2) & \cdots & w_1(2)x_r(2) & \cdots & w_L(2) & w_L(1)x_1(2) & \cdots & w_L(2)x_r(2) \\ & & & & \vdots & & & & \\ w_1(N) & w_1(N)x_1(N) & \cdots & w_1(N)x_r(N) & \cdots & w_L(N) & w_L(1)x_1(N) & \cdots & w_L(N)x_r(N) \end{bmatrix} \in \Re^{N\times[(r+1)\times L]} \tag{10}$$

$$\mathbf{b} = [b_{10} \quad b_{11} \quad \cdots \quad b_{1r} \quad \cdots \quad b_{L0} \quad b_{L1} \quad \cdots \quad b_{Lr}]^T \in \Re^{(r+1)\times L}. \tag{11}$$

obtain the smallest Euclidean norm solution of $\mathbf{b}$ as [5]

$$\mathbf{b} = \sum_{i=1}^{s} \frac{\mathbf{u}_i^T \mathbf{d}}{\sigma_i} \mathbf{v}_i \tag{13}$$

where s is the number of nonzero singular values in Σ.[3]

B. Local Learning

Local learning computes the parameters of the model through minimizing the objective function J_L defined by [25]

$$J_L = \sum_{i=1}^{L} \sum_{k=1}^{N} w_i(k)[d(k) - y_i(k)]^2. \tag{14}$$

In this objective function each fuzzy rule is encouraged to produce the whole of the output rather than a component. Moreover, notice that w_i, the normalized firing strength of the ith rule as defined in (7), has nonzero values only in a small region of the input space. As a result, each fuzzy rule acts like an independent model that is only related to a subset of training data. This is exactly the objective function form used in LWR except the tricube function (rather than w_i) is used there.[4]

Similar to (8), J_L can be rearranged into a simple matrix form

$$\begin{aligned} J_L &= \sum_{i=1}^{L} (\mathbf{d} - X_i \mathbf{b}_i)^T W_i (\mathbf{d} - X_i \mathbf{b}_i) \\ &= \sum_{i=1}^{L} \left(W_i^{1/2}\mathbf{d} - W_i^{1/2} x_i \mathbf{b}_i\right)^T \left(W_i^{1/2}\mathbf{d} - W_i^{1/2} X_i \mathbf{b}_i\right) \end{aligned} \tag{15}$$

where

$$W_i = \begin{bmatrix} w_i(1) & & & \\ & w_i(2) & & \\ & & \ddots & \\ & & & w_i(N) \end{bmatrix} \in \Re^{N \times N} \quad i = 1, 2, \cdots, L \tag{16}$$

$$X_i = \begin{bmatrix} w_i(1) & w_i(1)x_1(1) & \cdots & w_i(1)x_r(1) \\ w_i(2) & w_i(2)x_1(2) & \cdots & w_i(2)x_r(2) \\ & & \vdots & \\ w_i(N) & w_i(N)x_1(N) & \cdots & w_i(N)x_r(N) \end{bmatrix} \in \Re^{N \times (r+1)}, \quad i = 1, 2, \cdots, L \tag{17}$$

$$\mathbf{b}_i = [b_{i0} \quad b_{i1} \quad \cdots \quad b_{ir}]^T \in \Re^{r+1}, \quad i = 1, 2, \cdots, L. \tag{18}$$

[3] In practice, the minimum Euclidean norm solution is usually approximated by substituting s with r in (13), where $r \leq s$ is the number of "large" singular values in Σ. The reason is that the presence of computer roundoff errors and data uncertainties often make the matrix X have "tiny" singular values. Setting these "tiny" singular values to zeros can effectively eliminate the effects of roundoff errors and uncertainties, and thus generates numerically reliable minimum Euclidean norm solution.

[4] A similar objective function has also been suggested in [9] under the framework of modular networks. For an interesting comparison between modular networks and TSK models, we refer the reader to [12].

Applying SVD to $W_i^{1/2} X_i$, we have

$$W_i^{1/2} X_i = U_i \Sigma_i V_i^T, \quad i = 1, 2, \cdots, L. \tag{19}$$

Then the smallest Euclidean norm solution of $\mathbf{b}_i$ is computed as

$$\mathbf{b}_i = \sum_{l=1}^{s_i} \frac{\mathbf{u}_l^T W_i^{1/2} \mathbf{d}}{\sigma_l} \mathbf{v}_l, \quad i = 1, 2, \cdots, L \tag{20}$$

where s_i is the number of nonzero singular values in Σ_i, u_l, and v_l are the lth column of U_i and V_i, respectively.

Notice that the parameters of each fuzzy rule have been computed independently using a subset of training data. Thus, the resulting model has a clear local implication. Also, the computation can be implemented in parallel and the computational load is essentially unaffected by the assumed number of fuzzy rules.

C. Combined Learning

Combined learning aims at striking a good tradeoff between the global approximation and the local interpretation of a TSK model; it chooses the parameters of the model through minimizing a combined objective function J_C

$$J_C = \alpha J_G + \beta J_L \tag{21}$$

where α and β are two positive constants satisfying

$$\alpha + \beta = 1. \tag{22}$$

Notice that the local objective function defined in (15) can be written in a more compact form; that is

$$J_L = (\mathbf{d}' - X'\mathbf{b})^T W (\mathbf{d}' - X'\mathbf{b}) \tag{23}$$

where

$$\mathbf{d}' = [\mathbf{d} \quad \mathbf{d} \quad \cdots \quad \mathbf{d}]^T \in \Re^{N \times N} \tag{24}$$

$$X' = \begin{bmatrix} X_1 & & & \\ & X_2 & & \\ & & \ddots & \\ & & & X_L \end{bmatrix} \in \Re^{(N \times N) \times [(r+1) \times L]} \tag{25}$$

$$W = \begin{bmatrix} W_1 & & & \\ & W_2 & & \\ & & \ddots & \\ & & & W_L \end{bmatrix} \in \Re^{(N \times N) \times (N \times N)} \tag{26}$$

$$\mathbf{b} = [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_L]^T \in \Re^{(r+1) \times L}. \tag{27}$$

Substituting (23) and (8) into (21) we get

$$J_C = \alpha(\mathbf{d} - X\mathbf{b})^T(\mathbf{d} - X\mathbf{b}) + \beta(\mathbf{d}' - X'\mathbf{b})^T W (\mathbf{d}' - X'\mathbf{b}). \tag{28}$$

Because $\mathbf{d}'$ and $\mathbf{d}$ as well as X and X' have different dimensions, we cannot directly apply the SVD (as before) to a single matrix to obtain the minimum Euclidean solution of $\mathbf{b}$. Instead, we differentiate J_C with respect to $\mathbf{b}$ and equate

the result to zero, which yields

$$(\alpha X^T X + \beta X'^T W X'^T)\mathbf{b} = (\alpha X^T \mathbf{d} + \beta X'^T W \mathbf{d}'). \quad (29)$$

Let $\tilde{X} \equiv \alpha X^T X + \beta X'^T W X'^T$, and $\tilde{\mathbf{d}} = \alpha X^T \mathbf{d} + \beta X'^T W \mathbf{d}'$. Then (29) becomes

$$\tilde{X}\mathbf{b} = \tilde{\mathbf{d}}. \quad (30)$$

Applying SVD to $\tilde{X}$ yields

$$\tilde{X} = \tilde{U}\tilde{\Sigma}\tilde{V}^T \quad (31)$$

where $\tilde{U}$, $\tilde{V}$, and $\tilde{\Sigma} \in \Re^{[(r+1)\times L]\times[(r+1)\times L]}$.

The minimum Euclidean norm solution of $\mathbf{b}$ is then computed as

$$\mathbf{b} = \sum_{i=1}^{s} \frac{\tilde{\mathbf{u}}_i^T \tilde{\mathbf{d}}}{\tilde{\sigma}_i} \tilde{\mathbf{v}}_i \quad (32)$$

where s is the number of nonzero singular values in $\tilde{\Sigma}$, $\tilde{u}_i$, and $\tilde{v}_i$ are the ith column of $\tilde{U}$ and $\tilde{V}$, respectively.

III. Partition of the Input Space

Since the combined learning algorithm needs to construct linear regression models in different fuzzy partitions, it is important to know the number and the position of these partitions in the input space. Usually, the number of fuzzy partitions also determines the number of fuzzy rules comprising the underlying model.

There are two approaches that can be used to form an appropriate fuzzy partition. The first approach starts with a small number of fuzzy partitions (typically one partition), and then adds more partitions based on some measure criterion [21]. The second approach goes to another direction; it begins with an oversized number of fuzzy partitions and then removes those redundant and less important fuzzy partitions [8], [14], [24], [26]. In this paper, the second approach is used.

To illustrate the approach, we recall that the antecedent of a fuzzy model results from the fuzzy partition of input space. Thus, we construct the following matrix, known as the *firing strength matrix*, whose entries are only related to the antecedent of the model

$$F = \begin{bmatrix} w_1(1) & w_2(1) & \cdots & w_L(1) \\ w_1(2) & w_2(2) & \cdots & w_L(2) \\ \vdots & \vdots & \cdots & \vdots \\ w_1(N) & w_2(N) & \cdots & w_L(N) \end{bmatrix}. \quad (33)$$

This is an N-by-L matrix, where N is the number of training data and L is the number of fuzzy partitions (rules). The entry w_i in this matrix is the normalized firing strength of the ith rule as defined in (7). Notice that each column of F corresponds to one of the fuzzy partitions. As a result, if one column of F is linearly dependent on other columns or consists of all zeros, then the partition associated with the column is redundant or less important and removing it will not affect the performance of the underlying model greatly. Mathematically, a linear dependent or zero column will make the matrix singular and, consequently, the matrix will have a zero singular value.[5] Thus, we may detect the number of redundant or less important fuzzy partitions by examining the singular values of the matrix F. In order to identify the position of those redundant or less important fuzzy partitions or, equivalently, pick out the columns that are linear dependent or full of zeros, we apply a numerically reliable orthogonal transformation method, known as SVD-QR with column pivoting algorithm to F. This algorithm was originally proposed in Golub *et al*. [4] to solve the subset selection problem in regression analysis and used by Kanjilal and Banerjee [11] to select hidden nodes in a feedforward neural network, and by Mouzouris and Mendel [14] to extract important fuzzy basis functions in a fuzzy model. A comparison of this algorithm with other orthogonal transformation based methods is given in Yen and Wang [26]. This algorithm is summarized as follows.

SVD-QR with column pivoting algorithm for selecting fuzzy partitions:

1) Compute the SVD of F: $F = U\Sigma V^T$ and save Σ and V.
2) Check the singular values in $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \cdots, \sigma_L)$, and determine the number of fuzzy partitions that are to be used to partition the input space as r where $r \leq \mathrm{rank}(F)$.
3) Partition V as $V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix}$, where $V_{11} \in \Re^{r\times r}$, and $V_{21} \in \Re^{r\times(L-r)}$; form $\bar{V}^T = [V_{11}^T \ V_{21}^T]$.
4) Apply *QR with column pivoting algorithm* [5] to $\bar{V}$ and get the *permutation matrix* $\Pi \in \Re^{L\times L}$: $Q^T[V_{11}^T \ V_{21}^T]\Pi = [R_{11} \ R_{12}]$ where $Q \in \Re^{r\times r}$ is orthogonal and $R_{11} \in \Re^{r\times r}$ is upper triangular. The position of the entries one's in the first r columns of Π indicates the position of the r most important fuzzy partitions in the input space.

IV. Results

In this section, we illustrate the above ideas using a simulated motorcycle crash data set taken from Hardle [6]. This data set, as presented in Fig. 3, consists of 133 accelerometer readings taken through time in an experiment on the efficacy of crash helmets. For various reasons, the time points are not regularly spaced, and there are multiple observations at some time points. In addition, the observations are all subject to error. Our interest here is to discern the general type of the underlying acceleration curve using a TSK model. The input of the model is the time, denoted by x, and the output of the model is the acceleration, denoted by y.

In order to build the model, we first partition the input space (i.e., the domain of time observations) using the Gaussian-type fuzzy membership functions defined by

$$A_i(x) = \exp\left(-\frac{(x-c_i)^2}{2\sigma_i^2}\right), \quad i = 1, 2, \cdots, L \quad (34)$$

[5] In practice, one column in F is seldom exactly linear dependent on other columns or consists of exactly zero elements. As a result, the relevant singular value is not zero but a "small" value. Even worse, the small singular value and other "large" singular values may not have a clear gap. This makes the determination of the number of redundant or less important fuzzy rules a nontrivial exercise. In this case, a certain degree of trial and error is unavoidable for the determination.

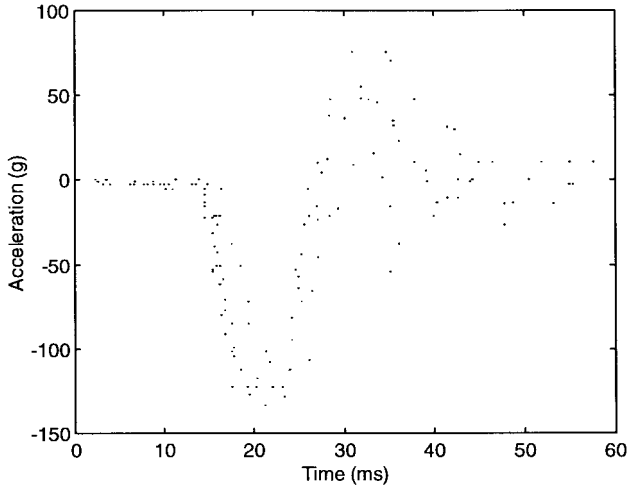

Fig. 3. The motorcycle data set.

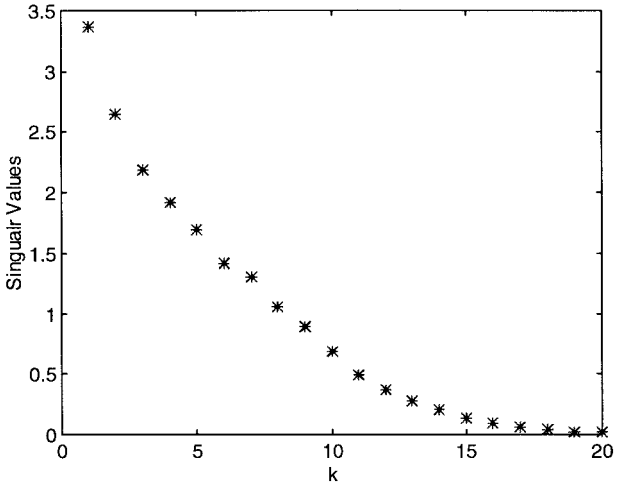

Fig. 4. Distribution of singular values of the 133 × 20 firing strength matrix.

where L is the number of fuzzy partitions (also the number of fuzzy rules). The centers, c_i, of the Gaussian membership functions are assumed to be regularly distributed in the interval [2, 58], and the widths σ_i are determined using a nearest neighbor heuristic suggested in Moody and Darken [13], that is

$$\sigma_i = \left[\frac{1}{p}\sum_{l=1}^{p}(c_i - c_l)^2\right]^{1/2}, \quad i = 1, 2, \cdots, L \tag{35}$$

where c_l $(l = 1, 2, \cdots, p)$ are the p (typically $p = 2$) nearest neighbors of the center c_i. In our experiment, we assume that all Gaussian membership functions have the same width σ, which is obtained by averaging σ_i in (35) over all L centers, that is

$$\sigma = \frac{1}{L}\sum_{i=1}^{L}\sigma_i. \tag{36}$$

Initially, we set the number of fuzzy partitions to 20. These fuzzy partitions are labeled as $1, 2, \cdots, 20$ to indicate their position in the input space. For each of the 133 input data points $x(k)$, $k = 1, 2, \cdots, 133$ to the partitions, we compute the normalized firing strengths using (7). A 133 × 20 firing strength matrix F is then formed. Applying SVD to F, the resulting singular values are shown in Fig. 4. There is no clear gap between the "large" singular values and the "small" singular values. Here we arbitrarily take the first eight singular values as the "large" singular values and the last 12 singular values as the "small" ones. As a result, we reduce the number of fuzzy partitions from 20 to 8. The position of the eight fuzzy partitions in the input space is identified as 6, 4, 10, 8, 2, 12, 15, 19 using the SVD-QR with column pivoting algorithm introduced in Section III. The order of the position also indicates the importance of the associated fuzzy partitions in the input space (also the importance of the associated fuzzy rules in the rule base). Fig. 5 shows the Gaussian membership functions associated with the initial 20 fuzzy partitions and the retained eight fuzzy partitions.

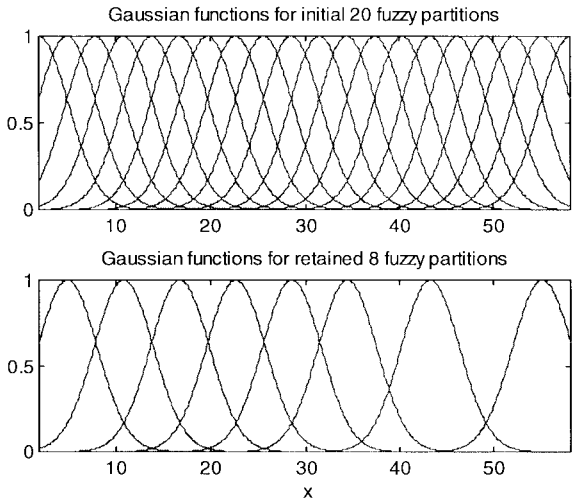

Fig. 5. Membership functions associated with the initial and retained fuzzy partitions.

Notice that the number of fuzzy partitions determines the number of fuzzy rules constituting the underlying TSK model. Thus, by retaining eight fuzzy partitions from the initial 20 fuzzy partitions, we build a TSK model with eight rules. The consequent parameters (i.e., those in the linear regression equations) of the model are computed using the three different learning algorithms introduced in Section II. The performance of the model are measured by the global and the *local mean-squares errors* (MSE's), which are computed, respectively, by

$$Global\_MSE = \frac{1}{N}\sum_{i=1}^{N}[d(k) - y(k)]^2 \tag{37}$$

and

$$Local\_MSE = \frac{1}{N}\sum_{i=1}^{L}\sum_{k=1}^{N}[d(k) - y_i(k)]^2. \tag{38}$$

Fig. 6–8 present the outputs of the resultant TSK model. The linear regression equations in the consequent part of the model

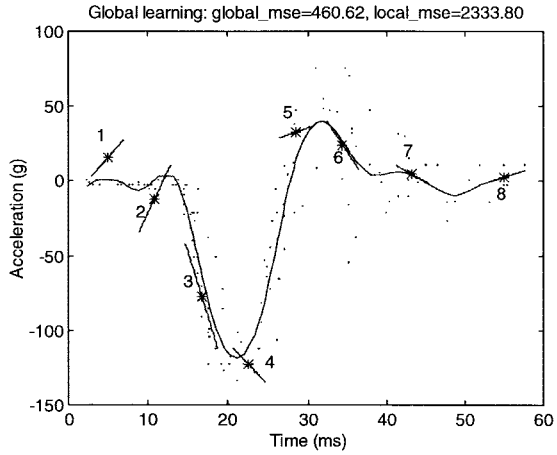

Fig. 6. Global and local outputs of TSK model using only global learning.

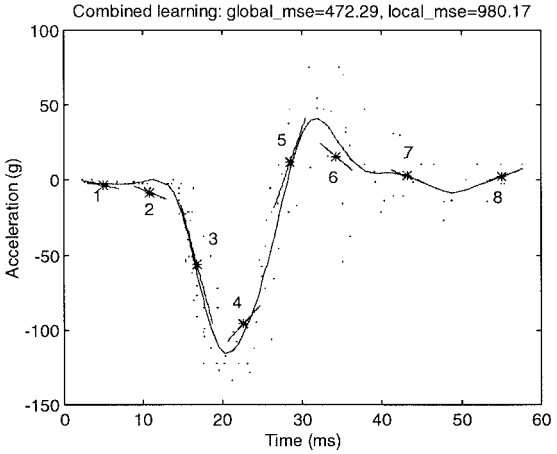

Fig. 8. Global and local outputs of TSK model using combined learning.

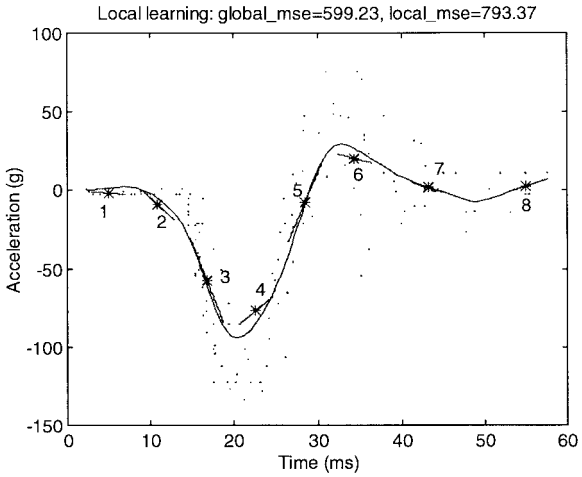

Fig. 7. Global and local outputs of TSK model using only local learning.

TABLE I
GLOBAL AND LOCAL PERFORMANCE OF THE TSK MODEL IDENTIFIED USING THE COMBINED LEARNING ALGORITHM WITH VARYING WEIGHTS

α	β	Global MSE	Local MSE
0.0	1.0	599.23	793.37
0.1	0.9	685.64	794.11
0.2	0.8	571.60	796.60
0.3	0.7	557.30	801.42
0.4	0.6	542.63	809.37
0.5	0.5	527.74	821.62
0.6	0.4	512.83	839.99
0.7	0.3	498.19	867.43
0.8	0.2	484.37	909.47
0.9	0.1	472.29	980.17
1.0	0.0	460.62	2333.80

are indicated by the straight lines, with the centers of fuzzy partitions depicted by the small stars. The model identified by the global learning algorithm (Fig. 6) has the best global performance but worst local performance. The submodels one, two, four, and five in the model exhibit an erratic local behavior and are hard to interpret. On the contrary, the model identified using the local learning algorithm (Fig. 7) shows the best local performance, while its global performance is the worst. A good tradeoff between global approximation and local interpretation has been achieved in the model identified by the combined learning algorithm with the weights $\alpha = 0.9$ and $\beta = 0.1$ (Fig. 8). Notice that the weights α and β in the combined learning algorithm can be adjusted by the user according to his own preference: if the user prefers to have a model with better global performance, then he should assign a larger value to α; if the user prefers to have a model with better local performance, then he should assign a larger value to β. In particular, the combined learning algorithm is reduced to the global one when $\alpha = 1$ and to the local one when $\beta = 1$. Table I presents the global MSE and local MSE of several TSK models with varying α and β values.

V. CONCLUSION

The TSK modeling methodology is essentially a multimodel approach in which simple submodels (where each submodel acts like a "local model") are coupled to describe the global behavior of the system. Since one of the important motivations of using the TSK model (also other fuzzy models) is to gain insights into the model, it is important to investigate the local interpretation issue of the TSK model. In this paper, we present a combined learning algorithm that is capable of generating a TSK model with both good global approximating ability and good local interpretation capacity. We illustrate the performance of the proposed algorithm using a motorcycle crash modeling example.

The singular value decomposition (SVD) has played a central role in forming the combined learning algorithm and the input space partitioning algorithm. However, the computation of SVD can become memory and time intensive for high-dimensional (i.e., a large number of input variables) and large data sets. Developing alternative computationally efficient procedures for combining global learning and local learning for high-dimensional/large data sets modeling problems is our current research focus.

REFERENCES

[1] S. L. Chiu, "Fuzzy model identification based on cluster estimation," *J. Intell. Fuzzy Syst.*, vol. 2, pp. 267–278, 1994.

[2] W. S. Cleveland, "Robust locally weighted regression and smoothing scatterplots," *J. Amer. Statistical Assoc.*, vol. 74, pp. 829–836, 1979.

[3] J. D. Farmer and J. Sidorowich, "Predicting chaotic time series," *Physical Rev. Lett.*, vol. 59, pp. 845–848, 1987.

[4] G. H. Golub, V. Klema, and G. W. Stewart, "Rank degeneracy and least squares problems," Tech. Rep. TR-456, Dept. Comput. Sci., Univ. Maryland, College Park, MD, 1976.

[5] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed. Baltimore, MD: John Hopkins Univ. Press, 1989.

[6] W. Hardle, *Applied Nonparametric Regression*. Cambridge, MA: Cambridge Univ. Press, 1990.

[7] T. J. Hastie and R. J. Tibshirani, *Generalized Additive Models*. London, U.K.: Chapman Hall, 1990.

[8] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Selecting fuzzy IF–THEN rules for classification problems using genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 260–270, Aug. 1995.

[9] R. A. Jacobs, M. J. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, pp. 79–87, 1991.

[10] J.-S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 665–685, May/June 1993.

[11] P. P. Kanjilal and D. N. Banerjee, "On the application of orthogonal transformation for the design and analysis of feedforward networks," *IEEE Trans. Neural Networks*, vol. 6, pp. 1061–1070, 1995.

[12] R. Langari and L. Wang, "Fuzzy models, modular networks, and hybrid learning," *Fuzzy Sets Syst.*, vol. 79, pp. 141–150, 1996.

[13] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computat.*, vol. 1, pp. 281–294, 1989.

[14] G. C. Mouzouris and J. M. Mendel, "Designing fuzzy logic systems for uncertain environments using a singular-value-QR decomposition method," in *Proc. 5th IEEE Int. Conf. Fuzzy Syst.*, New Orleans, LA, Sept. 1996, pp. 295–301.

[15] R. Murray-Smith and T. A. Johansen, "Local learning in local model networks," in *Multiple Model Approaches to Modeling and Control*, R. Murray-Smith and T. A. Johansen, Eds.. London, U.K.: Taylor Francis, 1997, pp. 185–210.

[16] C. J. Pantaleon-Prieto, I. Santamaria-Caballero, and A. R. Figueiras-Vidal, "Competitive local linear modeling," *Signal Processing*, vol. 49, pp. 73–83, 1996.

[17] S. Schaal and C. G. Atkeson, "Assessing the quality of learned local models," in *Advances in Neural Information Processing Systems 6*, J. Cowen, G. Tesauro, and J. Alspector, Eds. San Francisco, CA: Morgan Kaufmann, 1994, pp. 35–42.

[18] A. C. Singer, G. W. Wornell, and A. V. Oppenheim, "Codebook prediction: A nonlinear signal modeling paradigm," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, San Francisco, CA, 1992, pp. 325–328.

[19] M. Sugeno and G. T. Kang, "Structure identification of fuzzy model," *Fuzzy Sets Syst.*, vol. 28, pp. 15–33, 1988.

[20] H. Takagi and I. Hayashi, "NN-driven fuzzy reasoning," *Int. J. Approximate Reasoning*, vol. 5, pp. 191–212, 1991.

[21] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. 15, pp. 116–132, Jan. 1985.

[22] H. Tong, *Threshold Models in Non-Linear Time Series Analysis*. New York: Springer-Verlag, 1983.

[23] L. Wang and R. Langari, "Complex systems modeling via fuzzy logic," *IEEE Trans. Syst., Man, Cybern.*, vol. 26, pp. 100–106, Feb. 1996.

[24] L. X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least squares learning," *IEEE Trans. Neural Networks*, vol. 3, pp. 807–814, Sept. 1992.

[25] J. Yen and W. Gillespie, "Integrating global and local evaluations for fuzzy model identification using genetic algorithms," in *Proc. 6th Int. Fuzzy Syst. Assoc. World Congress (IFSA'95)*, Sao Paolo, Brazil, 1995.

[26] J. Yen and L. Wang, "Simplification of fuzzy rule based systems using orthogonal transformation," in *Proc. 6th IEEE Int. Conf. Fuzzy Syst.*, Barcelona, Spain, July 1997, pp. 253–258.

John Yen (M'91–SM'97) received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, in 1980, the M.S. degree in computer science from the University of Santa Clara, Santa Clara, CA, in 1982, and the Ph.D. degree in computer science from the University of California, Berkeley, in 1986.

He is currently a Professor in the Department of Computer Science and the Director of the Center for Fuzzy Logic, Robotics, and Intelligent Systems, Texas A&M University, College Station, TX. Before joining Texas A&M in 1989 he had been conducting artificial intelligence research as a Research Scientist at Information Sciences Institute, University of Southern California, Marina del Rey, CA. He is an Associate Editor of three international journals on artificial intelligence and fuzzy logic. He has authored or coauthored over 100 technical journal and conference papers. Recently, he coauthored the textbook, *Fuzzy Logic: Intelligence, Control, and Information*. His research interests include intelligent agents, fuzzy logic, software engineering, and pattern recognition.

Dr. Yen is a Vice President of publications for the IEEE Neural Networks Council and the Secretary of International Fuzzy Systems Association (IFSA). He received the NSF Young Investigator Award in 1992, the K. S. Fu Award from NAFIPS in 1994, and the Dresser Industries Award from Texas A&M University in 1995.

Liang Wang (M'95) received the B.S. degree in electrical engineering and the M.S. degree in systems engineering from Tianjin University, Tianjin, China, in 1985 and 1988, respectively, and the Ph.D. degree in computer science (highest honors) from the Faculté Polytechnique de Mons, Mons, Belgium, in 1993.

He was selected by the Liaison Committee of the European Community to continue his research in Europe in 1991. He is now a Postdoctoral Research Associate in the Center for Fuzzy Logic, Robotics, and Intelligent Systems at Texas A&M University, College Station, TX. He has co-authored over 30 technical journal and conference papers. His current research interests include fuzzy logic, neural networks, computer simulations, and intelligent systems modeling and control.

Charles Wayne Gillespie received the B.S. degree in mathematics and computer science from Trinity University, San Antonio, TX, and the M.S. degree in computer science from Texas A&M University, College Station, TX in 1990 and 1993 respectively. He is currently working toward the Ph.D. degree at the Department of Computer Science, Texas A&M University.

From 1990 to 1991, he was a Texas A&M University Graduate Research Assistant working with Texas Instruments to use neural networks for problem detection in the assembly line. From 1991 to 1995 he was the recipient of a NASA Graduate Fellowship to pursue research in fuzzy logic, neural networks, genetic algorithms, and real-time control for fuzzy modeling of tether satellite operations. From 1996 to 1997 he was a Texas A&M University Graduate Assistant Researcher working with the Texas Transportation Institute to develop intelligent traffic control. Since 1997 he has been a Graduate Assistant Researcher pursuing studies in fuzzy modeling. His current research interests include fuzzy modeling, genetic algorithms, neural networks, and other artificial intelligence paradigms for control and modeling.