A Fuzzy System for Uncertain Data Modeling

Yogesh Mohan, Sukhvir Singh, Kishori Lal Bansal, Mohit Kumar

Abstract— This text presents a fuzzy rules based system for modeling the relationships between inputs and output data in the presence of uncertainties. The fuzzy system is designed by separating the uncertainties from the data using fuzzy filtering algorithms. A stochastic modeling of the uncertainties helps in designing the fuzzy system to approximate the uncertain relationships. The proposed fuzzy model offers the followings: ***1)*** predicts the output value for the given inputs assuming that there were no uncertainties in the input-output behavior; ***2)*** assesses the worst effect of uncertainties on the model-predicted output value via predicting upper and lower limits on the output; ***3)*** predicts the output value for the given inputs taking mathematically into account the underlying uncertainties (whose probabilistic-model was extracted from the data) in a sensible way. The paper illustrates through an example that the proposed fuzzy system is a useful modeling tool in presence of uncertainties.

Index Terms— Data Modeling, Fuzzy filtering, Parameters estimation, Uncertainties.

1 Introduction

THE use of fuzzy systems in data driven modeling is a widely studied topic [49], [44], [45], [43], [14], [48], [38], [12], [11], [3], [2], [1], [46], [47], [15], [50], [39], [22], [37] and [21]. A real-world modeling problem typically involves the uncertainties and the fuzzy systems based on fuzzy set theory [54], [55] are considered suitable tools for dealing with the uncertainties. The efforts has been made recently some efforts to develop fuzzy filtering based methods for a proper handling of the uncertainties typically involved in real-world modeling applications related to the life sciences [34], [18], [35], [36], [28], [19], [30], [29], [32], [33].

It is assumed that input variables $(x_1, x_2, \cdots, x_n)$ are related to the output variable y through a mapping $y = f(x)$; where $x = [x_1 x_2 \cdots x_n] \in R^n$ is the input vector and the modeling aim is to identify the unknown function f. The fuzzy modeling is based on the assumption that there exist an ideal set of model parameters w^* such that model output $M(x; w^*)$ to input x is an approximation of the output value y. However, it may not be possible, for a given type and structure of the model M, to identify perfectly the inputs-output relationships. The part of the input-output mappings that can't be modeled, for a given type and structure of the model, is what we refer to as the uncertainty. Mathematically, we have

$$y = M(x; w^*) + n; \quad (1)$$

where n is termed as disturbance or noise in system identification literature. However, we refer n, in context to real-world modeling applications; to as uncertainty to emphasize that the uncertainties regarding optimal choices of the model and errors in output data resulted in the additive disturbance in(1). For an illustration, the authors in [18], in context to subjective workload score modeling, explain the reasons giving rise to the uncertainty.

A robust (towards uncertainty n) identification of model parameters w^* using available input-output data pairs $\{x(j), y(j)\}_{j=0,1,\cdots}$ is obviously a straightforward approach to handle the uncertainty. Several robust methods of fuzzy identification have been developed [8], [51], [7], [53], [16], [13], [17], [25], [23], [27], [24], [20] and [26]. It is desired to build a data model capable of answering the following questions:

Q1. What is the output value associated to an input vector x assuming that there were no uncertainties in the input-output behavior? i.e. what is an estimate of quantity $M(x; w^*)$?

Q2. What is the worst effect of uncertainties on the model-predicted output value? That is, what are the estimates of the quantities $M(x; w_L^*)$ and $M(x; w_U^*)$ such that $M(x; w_L^*) \le y$ *and* $M(x; w_U^*) \ge y$, where w_L^* and w_U^* are as close to w^* as possible.

Q3. What is the output value associated to an input vector x given a priori knowledge regarding the statistical model of the uncertainties? That is, what is an estimate of output value y as per a given estimate criterion?

This study proposes a fuzzy model to answer the aforementioned questions. The question Q1 is answered by identifying the parameters w^* using recently developed fuzzy filtering algorithms [21], [22]. The question Q2 is addressed, as in [32], by solving two independent constrained estimation problems with the help of a recurrent neural network. To solve the problem of question Q3, we stochastically model the uncertainties (provided by the fuzzy filter) using finite-mixture models and utilize this information regarding uncertainties for identifying the structure and initial parameters of a data model. A fuzzy model, addressing the Q1 and Q3, has already been proposed (in context to real-world applications) in earlier

The work of M. Kumar was supported by the Center for Life Science Automation, Rostock. The software used in this study was developed by Y. Mohan and S. Singh as a part of their PhD works.

- ***Y. Mohan, S. Singh and K. L. Bansal***
  Department of Computer Science, Himachal Pradesh University, Shimla-171005, India Tel.:+91-94180-66088 E-mail: yogeshmohan.edu@gmail.com
- ***M. Kumar***
  Center for Life Science Automation, F.-Barnewitz-Str. 8, D-18119 Rostock, Germany Tel.:+49-381-4949956 E-mail: mohit.kumar@uni-rostock.de

works [18], [30]. In this text, we present the method of [18], [30] in a generalized framework of fuzzy filtering. Based on the results of research on fuzzy filtering algorithms mainly presented in [21], [22], this manuscript proposes a method of data modeling in presence of uncertainty that performs better than the standard neuro/fuzzy modeling techniques. The main contribution of this text is to present a modeling algorithm providing an intergrated framework to develop a data model based on different fuzzy filtering algorithms. Although the individual components of the proposed framework and the basic modeling approach have been previously applied for practical applications [18], [30], but we feel that the presentation of the fuzzy filtering based uncertain data modeling approach in a generalized manner is worth of fuzzy readership.

The paper is organized as follows. Section 2 presents some background of the method. Section 3 suggests a fuzzy system for data modeling. Section 4 deals with the estimation of parameters of proposed fuzzy system followed by an illustration of the method in section 5 and the concluding remarks in section 6.

2 PRELIMINARY

2.1 A Clustering Based Fuzzy Filter [34, 35, 28, 29, 30, 36, 18]

It is required to filter out the uncertainties from the data with applications to many real-world modeling problems [34, 35, 28, 29, 30, 36, 18]. A filter in, the context of our study, simply maps an input vector x to the quantity $y-n$ (called filtered output $y_f = y-n$) and thus separates uncertainty n from the output value y. The fuzzy filter of [34, 35, 28, 29, 30, 36, 18] has K number of fuzzy rules of following type:

If x belongs to a cluster having centre c_1 then $y_f = \alpha^1$

$\vdots$

If x belongs to a cluster having centre c_K then $y_f = \alpha^K$

where $c_i \in R^n$ is the centre of i^{th} cluster, and the values $\alpha^1, \cdots, \alpha^K$ are real numbers. Based on a clustering criterion, it was shown e.g. [18] that

$$y_f = \sum_{i=1}^{K} \alpha^i G_i\left(x, c_1, \cdots, c_K\right),$$

$$G_i\left(x, c_1, \cdots, c_K\right) = \frac{A_i\left(x, c_1, \cdots, c_K\right)}{\sum_{i=1}^{K} A_i\left(x, c_1, \cdots, c_K\right)},$$

$$A_i\left(x, c_1, \cdots, c_K\right) = \frac{A_{1i}^{\tilde{m}}}{2} + \frac{A_{2i}}{2}, \tilde{m} > 1,$$

Where A_{1i}, A_{1i} are given as

$$A_{1i} = \begin{cases} \dfrac{1}{\sum_{j=1}^{K}\left(\dfrac{\|x-c_i\|^2}{\|x-c_j\|^2}\right)^{\frac{1}{\tilde{m}-1}}} & x \in X \setminus \{c_j\}_{i=1,\cdots,K}, \\ 1 & x = c_i, \\ 0 & x \in \{c_j\}_{j=1,\cdots,K} \setminus \{c_i\} \end{cases}$$

$$A_{2i} = \exp\left(-\frac{\|x-c_i\|^2}{\delta_i}\right), \quad \delta_i = \min_j \|c_j - c_i\|^2.$$

With the notations:

$\alpha = \left[\alpha^1 \cdots \alpha^K\right] \in R^K$, $\theta = \left[c_1^T \cdots c_K^T\right]^T \in R^{Kn}$,

$G(x,\theta) = \left[G_1(x,\theta) \cdots G_K(x,\theta)\right] \in R^K$,

The output of fuzzy filter for an input x can be expressed as $y_f = G^T(x,\theta)\alpha$.

2.2 Finite Mixture Models

Assume that the 2-dimensional vector $z = \left[y_f \quad n\right]^T$ represents one particular outcome of a 2-dimensional random variable $Z \in R^2$ whose probability density function can be written as a mixture of the Gaussian distributions:

$$p(z) = \sum_{j=1}^{C} a_j\, p\left(z \middle| m_j, \Sigma_j\right), \text{ such that} \tag{2}$$

- The mixing probabilities $a_1, \cdots, a_C$ satisfy $a_j \geq 0$ and $\sum_{j=1}^{C} a_j = 1$,
- The parameters $m_j \in R^2, \Sigma_j$ (a 2x2 positive definite matrix) characterize fully the j^{th} Gaussian component: $p\left(z \middle| m_j, \Sigma_j\right) = \dfrac{\exp\left\{-\frac{1}{2}\left(z-m_j\right)^T \Sigma_j^{-1}\left(z-m_j\right)\right\}}{\sqrt{(2\pi)^2\left|\Sigma_j\right|}}$.

We assume that y_f and n are independent, i.e., Σ_j is a diagonal matrix. If

$$m_j = \begin{bmatrix} m_j^1 \\ m_j^2 \end{bmatrix}, \Sigma_j = \begin{bmatrix} \Sigma_j^1 & 0 \\ 0 & \Sigma_j^2 \end{bmatrix}, \text{ then} \tag{3}$$

$p\left(z \middle| m_j, \Sigma_j\right) = p\left(y_f \middle| m_j^1, \Sigma_j^1\right) p\left(n \middle| m_j^2, \Sigma_j^2\right)$, where

$$p\left(y_f \middle| m_j^1, \Sigma_j^1\right) = \frac{\exp\left\{-\dfrac{\left(y_f - m_j^1\right)^2}{2\Sigma_j^1}\right\}}{\sqrt{2\pi\Sigma_j^1}}, \text{ and}$$

$$p\left(n\left|m_j^2,\Sigma_j^2\right.\right)=\frac{\exp\left\{-\frac{\left(n-m_j^2\right)^2}{2\Sigma_j^2}\right\}}{\sqrt{2\pi\Sigma_j^2}}, \quad (4)$$

Let us define a random variable $F_{x,n}$ as follows:

$F_{x,n}=j$, if for some data point x, the filtered value $y_f=G^T\left(x,\theta\right)\alpha$ is generated by the probabilistic model $p\left(y_f\left|m_j^1,\Sigma_j^1\right.\right)$ and the uncertainty value n is generated by the probabilistic model $p\left(n\left|m_j^2,\Sigma_j^2\right.\right)$. In other words, $F_{x,n}=j$, if the data point is generated by the j^{th} probabilistic model $p\left(z\left|m_j,\Sigma_j\right.\right)$.

2.3 The Grouping of Input Vectors

In many applications, there is a prior knowledge regarding the classification of input data such that the data belonging to a class either form a cluster or are labeled as of same type. For example, in any biomedical application, the data belonging to a particular patient can be grouped in a class. Similarly, in structure activity applications, the compounds with a similar chemical structure can be grouped in a class. Even if no priory knowledge is available, it is always possible to divide the input data into different types via performing clustering on the data.

Let $IG_1,IG_2,\cdots,IG_S$ denote the S different groups into which the input vectors are divided. Define a random variable $w_j^{i,k}\left(i=1,2,\cdots,S;\ j=1,2,\cdots,C;k=1,2,\cdots,K\right)$ such that distribution function of $w_j^{i,k}$ is given as

$$p\left(w_j^{i,k}\right)=p\left(y\left|x\in IG_i,F_{x,n}=j,y_f=\alpha^k\right.\right).$$

That is, $w_j^{i,k}$ takes its value equal to output value given that input x belongs to the group IG_i; y_f is generated by the probabilistic model $p\left(y_f\left|m_j^1,\Sigma_j^1\right.\right)$, uncertainty value n is generated by the probabilistic model $p\left(n\left|m_j^2,\Sigma_j^2\right.\right)$; and filtered value y_f is equal to α^k (i.e. equal to the consequent of k^{th} rule of the fuzzy filter). Since $y=y_f+n$, therefore

$$p\left(w_j^{i,k}\right)=\frac{\exp\left\{-\frac{\left(w_j^{i,k}-\alpha^k-m_j^2\right)^2}{2\Sigma_j^2}\right\}}{\sqrt{2\pi\Sigma_j^2}}.$$

Let us define a $K-$dimensional vector w_j^i as follows

$$w_j^i=\left[w_j^{i,1}\quad w_j^{i,2}\quad\cdots\quad w_j^{i,K}\right]^T\in R^K.$$

Now, we have

$$p\left(w_j^i\right)=\frac{\exp\left\{-\frac{\left(w_j^i-\overline{w}_j^i\right)^T\sigma_{w_j^i}^{-1}\left(w_j^i-\overline{w}_j^i\right)}{2}\right\}}{\sqrt{\left(2\pi\right)^K\left|\sigma_{w_j^i}\right|}}, \quad (5)$$

$\overline{w}_j^i=\alpha+m_j^2$ and $\sigma_{w_j^i}=\Sigma_j^2 I$.

2.4 A Cooperative Recurrent Neural Network for Solving Constrained L_1 Estimation Problem [52]

Lemma 1. *The constrained L_1 estimation problem*

$$\min_x\left\{\left\|Dx-d\right\|_1, Ax\le p\right\} \quad (6)$$

Can be solved by the *following cooperative recurrent neural network model:*

$$\frac{dx(t)}{dt}=\lambda\left\{F_1(t)-D^TF_2(t)-A^TF_3(t)\right\}, \quad (7)$$

$$\frac{ds(t)}{dt}=\lambda\left\{DF_1(t)+F_2(t)\right\}, \quad (8)$$

$$\frac{dz_{II}(t)}{dt}=\lambda\left\{AF_1(t)+F_3(t)\right\}, \quad (9)$$

where $\lambda>0$ is a designing constant and

$$F_1(t)=-D^Ts(t)-A^Tz_{II}(t),$$
$$F_2(t)=gx_1\left(s(t)+Dx(t)-d\right)-s(t),$$
$$F_3(t)=gx_2\left(z_{II}(t)+Ax(t)-p\right)-z_{II}(t).$$

Here, $x(t),s(t),z_{II}(t)$ are state vectors of suitable dimensions and

$$gx_1(a)=\begin{cases}-1, & a<-1\\ a, & -1\le a\le 1,\\ 1, & a>1\end{cases}$$

$$gx_2(a)=\begin{cases}a, & a\ge 0\\ 0, & a<0\end{cases}.$$

Proof: The result has been directly taken from [52]

Lemma 2. *The cooperative recurrent neural network* (7), (8), (9) *is stable in the sense of Lyapunov and is globally convergent to an optimal solution of the estimation problem* (6) *within a finite time.*

Proof: The result has been directly taken from [52].

3 A Fuzzy System for Data Modeling in Presence of Uncertainties

A model with the following K fuzzy rules is considered:

R_1) *If* x *belongs to a cluster having centre* c_1 *then*
filtered output value $y_f = \alpha^1$,
lower limit on output value $y_L = \alpha_L^1$,
upper limit on output value $y_U = \alpha_U^1$,
for $x \in IG_i$, *the output value*

$$y = \frac{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right) w_j^{j,1}}{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right)};$$

$\vdots$ $\vdots$

R_K) *If* x *belongs to a cluster having centre* c_K *then*
filtered output value $y_f = \alpha^K$,
lower limit on output value $y_L = \alpha_L^K$,
upper limit on output value $y_U = \alpha_U^K$,
for $x \in IG_i$, *the output value*

$$y = \frac{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right) w_j^{i,K}}{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right)}; \quad (10)$$

Such a fuzzy model has been successfully applied to real-world problem in [18], [30], [32]. Introduce the notations:

$\alpha_L = \left[\alpha_L^1 \alpha_L^2 \cdots \alpha_L^K\right]^T$, $\alpha_U = \left[\alpha_U^1 \alpha_U^2 \cdots \alpha_U^K\right]^T$.

By aggregating these rules, we have

$y_f = G^T(x,\theta)\alpha, y_L = G^T(x,\theta)\alpha_L, y_U = G^T(x,\theta)\alpha_U,$

$$y = G_1(x,\theta)\frac{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right) w_j^{i,1}}{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right)} + \cdots$$

$$\cdots + G_K(x,\theta)\frac{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right) w_j^{i,K}}{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right)}.$$

In other words,

$$y = \frac{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right) G^T(x,\theta) w_j^i}{\sum_{j=1}^{C} a_j p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right)}. \quad (11)$$

The motivation of considering such a mathematical expression for y in (10) is derived from the following fact.

Remark 1 The fuzzy rule based system (10) for any input x combines the outputs of C different local models $\left(M_1^i(x), \cdots, M_C^i(x)\right)$ valid in the predefined operating regions represented by fuzzy sets $\left(A_1(y_f), \cdots, A_C(y_f)\right)$ respectively based on the following fuzzy rule base:

For input x, if the filtered value $y_f = G^T(x,\theta)\alpha$ is A_1,
then $y = M_1^i(x)$; $[a_1]$

$\vdots$

For input x, if the filtered value $y_f = G^T(x,\theta)\alpha$ is A_C,
then $y = M_C^i(x)$; $[a_C]$ (12)

Here $M_j^i(x)$ and $A_j(y_f)$, for $j = 1, \cdots, C$, are defined as

$$M_j^i(x) = G^T(x,\theta) w_j^i, A_j(y_f) = p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right). \quad (13)$$

The value $a_j \in [0,1]$ is the weight of the rule that represents the belief in the accuracy of the j^{th} rule. To see the equivalence of (10) and (12), note that the weighted average of the output provided by each rule of (12) is equal to (11).

4 ESTIMATION OF FUZZY SYSTEM PARAMETERS

For a data driven construction of the fuzzy model (10), following parameters must be estimated using given input –output data pairs $\{x(j), y(j)\}_{j=0,1,\cdots,N}$.

4.1 Estimation of Filtering Parameters (α, θ)

Lemma 3. *A class of algorithms for estimating the parameters of Takagi-Sugeno type fuzzy filter recursively using input-output data pairs* $\{x(j), y(j)\}_{j=0,1,\cdots}$ *is given by the following recursions:*

$$\theta_j = \arg\min_{\theta}\left[\Psi_j(\theta)\right], \tag{14}$$

$$\alpha_j = \alpha_{j-1} + \frac{P_j G(x(j),\theta_j)\left[y(j) - G^T(x(j),\theta_j)\alpha_{j-1}\right]}{1 + G^T(x(j),\theta_j) P_j G(x(j),\theta_j)}, \tag{15}$$

$$\Psi_j(\theta) = -\frac{\left|y(j) - G^T(x(j),\theta)\alpha_{j-1}\right|^2}{1 + G^T(x(j),\theta) P_j G(x(j),\theta)} + \mu_\theta^{-1}\left\|\theta - \theta_{j-1}\right\|^2, \tag{16}$$

$$P_{j+1} = \left[P_j^{-1} + (1+\gamma) G(x(j),\theta_j) G^T(x(j),\theta_j)\right]^{-1},$$

for all $j = 0,1,\cdots$ *with* $\alpha_{-1} = 0, P_0 = \mu I$, *and* θ_{-1} *is an initial guess about antecedents. Here,* $\gamma \geq -1$ *is a scalar whose different choices, as illustrated in table 1, solve the different filtering problems.*

Table 1 The value of γ for the different filtering criteria

H^∞ - optimal like filtering criterion [22]	$\gamma = -1$
Risk-averse like filtering criterion [22]	$-1 \leq \gamma < 0$
Risk-seeking like filtering criterion [22]	$\gamma > 0$

Proof: The result has been directly taken from [22].

The positive constants μ_θ in (16) is the learning rate for θ (being the co-ordinates of clusters' centres in $n-$dimensional input space), if assumed as random variables, may have different variances depending upon the distribution functions of different inputs. Therefore, estimating the elements of θ with different learning rates makes a sense. To do this, define a diagonal matrix Σ (with positive entries on its main diagonal):

$$\Sigma = \begin{bmatrix} \mu_{\theta(1)} & 0 & \cdots & 0 \\ 0 & \mu_{\theta(2)} & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{\theta(K_n)} \end{bmatrix},$$

to reformulate (16) as

$$\Psi_j(\theta) = -\frac{\left|y(j) - G^T(x(j),\theta)\alpha_{j-1}\right|^2}{1 + G^T(x(j),\theta) P_j G(x(j),\theta)} + \left\|\Sigma^{1/2}(\theta - \theta_{j-1})\right\|^2, \tag{17}$$

Lemma 4 *The adaptive* $p-$*norm algorithms for estimating the parameters of Takagi-Sugeno type fuzzy filter recursively using input-output data pairs* $\{x(j), y(j)\}_{j=o,1,\cdots}$ *take a general form of*

$$\theta_j = \arg\min_{\theta}\left[E_j(\hat{\alpha}(\theta),\theta) + \mu_{\theta,j}^{-1} d_q(\theta,\theta_{j-1})\right], \tag{18}$$

$$\alpha_j = f^{-1}\left(f(\alpha_{j-1}) + \mu_j \phi\left(y(j) - G^T(x(j),\theta_j)\alpha_{j-1}\right) G(x(j),\theta_j)\right); \tag{19}$$

Here,

$$E_j(\alpha,\theta) = L_j(\alpha,\theta) + \mu_j^{-1} d_q(\alpha,\alpha_{j-1}),$$

$$\hat{\alpha}(\theta) = f^{-1}\left(f(\alpha_{j-1}) + \mu_j \phi\left(y(j) - G^T(x(j),\theta)\alpha_{j-1}\right) G(x(j),\theta)\right),$$

$$d_q(u,w) = \frac{1}{2}\|u\|_q^2 - \frac{1}{2}\|w\|_q^2 - (u-w)^T f(w),$$

Where $(\mu_j, \mu_{\theta,j})$ *are the learning rates for* (α,θ) *respectively,* f *(a* p *indexing for* f *is understood), as defined in* [10]*, is the bijective mapping* $f: R^K \to R^K$ *such that* $f = [f_1 \cdots f_K]^T$,

$$f_i(w) = \frac{sign(w_i)|w_i|^{q-1}}{\|w\|_q^{q-2}},$$

where $w = [w_1 \cdots w_K]^T \in R^K$, q *is dual to* p $(i.e.\ 1/p + 1/q = 1)$, *and* $\|.\|_q$ *denotes the* $q-$*norm.*

The different choices of loss term $L_j(\alpha,\theta)$ *lead to the different functional form of* ϕ *and thus different types of fuzzy filtering algorithms for any* $p(2 \leq p \leq \infty)$. ***Table 2*** *lists a few examples of fuzzy filtering algorithms.*

Table 2 A few examples of adaptive fuzzy filtering algorithms [21]

Algorithm	$L_j(\alpha,\theta)$	$\phi(e)$	$P_\phi(y,\bar{y})$
$A_{1,p}$	$\ln(\cosh(y(j)-G^T(x(j),\theta)\alpha))$	$\tanh(e)$	$\ln(\cosh(\bar{y}))-\ln(\cosh(y))-(\bar{y}-y)\tanh(y)$
$A_{2,p}$	$(1/2)\lvert y(j)-G^T(x(j),\theta)\alpha\rvert^2$	e	$(1/2)\lvert y-\bar{y}\rvert^2$
$A_{3,p}$	$(1/4)\lvert y(j)-G^T(x(j),\theta)\alpha\rvert^4$	e^3	$\frac{\bar{y}^4}{4}-\frac{y^4}{4}-\left(y^4-y\right)y^3$
$A_{4,p}$	$\frac{a}{2}\lvert y(j)-G^T(x(j),\theta)\alpha\rvert^2$ $+\frac{b}{4}\lvert y(j)-G^T(x(j),\theta)\alpha\rvert^4$	$ae+be^3$	$\frac{a}{2}\lvert y-\bar{y}\rvert^2+b\left[\frac{\bar{y}^4}{4}-\frac{y^4}{4}-\left(y^4-y\right)y^3\right]$
$A_{5,p}$	$\cosh(y(j)-G^T(x(j),\theta)\alpha)$	$\sinh(e)$	$\cosh(\bar{y})-\cosh(y)-(\bar{y}-y)\sinh(y)$

The filtering algorithms, with a learning rate of

$$\mu_j=\frac{2P_\phi\left(y(j),G^T\alpha_{j-1}\right)}{\phi\left(y(j)-G^T\alpha_{j-1}\right)(p-1)\left[\phi(y(j))-\phi\left(G^T\alpha_{j-1}\right)\right]\|G\|_p^2}, \tag{20}$$

Where $G=G\left(x(j),\theta_j\right)$ *and* $P_\phi(y,\bar{y})=\int_y^{\bar{y}}\left(\phi(r)-\phi(y)\right)dr$, *achieves a stability and robustness against disturbances in some sense.*

Proof: The result has been directly taken from [21].

For a standard algorithm for computing θ_j numerically based on (18), define

$$k^q_{\theta,\theta_{j-1}}=\begin{cases}\dfrac{2d_q\left(\theta,\theta_{j-1}\right)}{\left\|\theta-\theta_{j-1}\right\|^2}, & if \quad \theta\neq\theta_{j-1}\\ 1, & if \quad \theta=\theta_{j-1}\end{cases}$$

to express (18) as

$$\theta_j=\arg\min_\theta\left[E_j\left(\hat{\alpha}(\theta),\theta\right)+\frac{\mu^{-1}_{\theta,j}k^q_{\theta,\theta_{j-1}}}{2}\left\|\left(\theta-\theta_{j-1}\right)\right\|^2\right]. \tag{21}$$

Choosing a time-invariant learning rate for θ in (21), i.e. $\mu_{\theta,j}=\mu_\theta$, and estimating the elements of vector θ with different learning rates as in (17), (21) finally becomes

$$\theta_j=\arg\min_\theta\left[E_j\left(\hat{\alpha}(\theta),\theta\right)+\frac{k^q_{\theta,\theta_{j-1}}}{2}\left\|\Sigma^{-1/2}\left(\theta-\theta_{j-1}\right)\right\|^2\right]. \tag{22}$$

Define vectors $r(\theta)$ and $r_q(\theta)$ as

$$r(\theta)=\begin{bmatrix}\left(\dfrac{\left[y(j)-G^T\left(x(j),\theta\right)\alpha_{j-1}\right]^2}{1+G^T\left(x(j),\theta\right)P_jG\left(x(j),\theta\right)}\right)^{1/2}\\ \Sigma^{-1/2}\left(\theta-\theta_{j-1}\right)\end{bmatrix}\in R^{K_n+1}, \tag{23}$$

$$r_q(\theta)=\begin{bmatrix}\sqrt{E_j\left(\hat{\alpha}(\theta),\theta\right)}\\ \left(\dfrac{k^q_{\theta,\theta_{j-1}}}{2}\right)^{1/2}\Sigma^{-1/2}\left(\theta-\theta_{j-1}\right)\end{bmatrix}\in R^{K_n+1}, \tag{24}$$

So that (14) and (18) can be formulated as

$$\theta_j=\begin{cases}\arg\min_\theta\left\|r(\theta)\right\|^2, & as \quad per \quad Lemma \quad 3\\ \arg\min_\theta\left\|r_q(\theta)\right\|^2, & as \quad per \quad Lemma \quad 4\end{cases} \tag{25}$$

An algorithm to estimate fuzzy filter parameters based on the filtering criteria of either *Lemma 3* or *Lemma 4* was provided in [31]. The algorithm has been repeated here in Appendix **A** for the sake of completion.

4.2 Estimation of Parameters (α_L,α_U)

Once fuzzy filter parameters $\left(\alpha^I,\theta^I\right)$ are identified using algorithm **2,** to estimate parameters (α_L,α_U) with data pairs $\left\{x(j),y(j)\right\}_{j=0,1,\cdots,N}$, define a matrix $A\in R^{(N+1)\times K}$ and a vector $p\in R^{N+1}$ as

$$A=\begin{bmatrix}G^T\left(x(0),\theta^I\right)\\ G^T\left(x(1),\theta^I\right)\\ \vdots\\ G^T\left(x(N),\theta^I\right)\end{bmatrix},P=\begin{bmatrix}y(0)\\ y(1)\\ \vdots\\ y(N)\end{bmatrix}.$$

Now, a lower and an upper fuzzy approximation to output data is provided by solving

$$\hat{\alpha}_L=\arg\min_{\alpha_L}\left\{\left\|\alpha_L-\alpha^I\right\|_1, A\alpha_L\le p\right\}, \tag{26}$$

$$\hat{\alpha}_U=\arg\min_{\alpha_U}\left\{\left\|\alpha_U-\alpha^I\right\|_1, -A\alpha_U\le -p\right\}. \tag{27}$$

Both (26) and (27) can be solved efficiently using **Lemma 1.**

4.3 Estimation of Parameters of finite mixture models

The uncertainties and filtered values associated to given input-output data are assessed by defining a set of 2-dimensional vectors $\{z_j\}_{j=0,1,\cdots,N}$ as

$$z_j = \begin{bmatrix} \hat{n}_j \\ f_f(j) \end{bmatrix} = \begin{bmatrix} y(j) - G^T(x(j),\theta^I)\alpha^I \\ G^T(x(j),\theta^I)\alpha^I \end{bmatrix}, \quad (28)$$

It is assumed that z_j represents one particular outcome of a 2-dimensional random variable $Z \in R^2$ whose probability density function can be expressed as (2). That is, a finite mixture model consisting of C different Gaussian components is fitted to the data $\{z_j\}_{j=0,1,\cdots,N}$.

Further, the covariance of each Gaussian component is chosen to be a diagonal matrix (3). "Expectation-Maximization (EM)" is the standard algorithm [41,42] used for the finite mixture modeling of data. In this study, we used the algorithm of [9] for estimating the parameters of the mixture (2). This algorithm is capable of automatically selecting the number of components C. The algorithm, unlike EM, is less sensitive to initialization and avoids the possibility of algorithm convergence to the boundary of the parameter space.

4.4 Estimation of Local Models' Parameters $\{w_j^i\}_{j=1,2,...,C}$

The $K-$dimensional vector w_j^i characterizes the linear parameters of j^{th} local model $M_j^i(x)$ for $x \in IG_i$, see(13). For an estimation of w_j^i, some input-output data pairs, say DP_j^i, must be chosen out of total $(N+1)$ pairs of data $\{x(j), y(j)\}_{j=0,1,\cdots,N}$. As seen in (12) that model $M_j^i(x)$ is associated to fuzzy set $A_j(y_f) = p(y_f | m_j^1, \Sigma_j^1)$, the data set DP_j^i is defined as

$$DP_j^i = \{(x,y), A_j(y_f = G^T(x,\theta^I)\alpha^I) \geq \varepsilon, x \in IG_i\}, \quad (29)$$

Where $0 \leq \varepsilon << 1$. DP_j^i contains all these input-output data pairs whose filtered output values belong to the fuzzy set A_j at least by a degree ε. To fit data DP_j^i through model $M_j^i(x) = G^T(x,\theta)w_j^i$, assume that

$$y = G^T(x,\theta^I)w_j^i + \upsilon, \ (x,\theta) \in DP_j^i,$$

where υ is Gaussian with mean 0 and variance Σ_j^2 (as the uncertainty associated to j^{th} probabilistic model has a variance equal to Σ_j^2, see (4)).

Given the data DP_j^i, distribution of υ (Gaussian with mean 0 and variance Σ_j^2), and distribution of w_j^i (5), the estimation of w_j^i based on MAP (maximum a posteriori) criterion:

$$\hat{w}_j^i = \arg\max_{w_j^i} p(w_j^i | DP_j^i) \quad (30)$$

is equivalent to estimating w_j^i using well known recursive least-squares (RLS) algorithm.

4.5 An Algorithm for data Modeling in Presence of Uncertainties.

Algorithm 1 finaly summarizes the different steps to identify a model of type (10) using given input-output data pairs. For a given input vector $x \in IG_i$, the parameters returned by algorithm 1, can be used to estimate

- fitered output $y_f = G^T\left(x,\theta^I\right)\alpha^I$,
- lower limit on output $y_L = G^T\left(x,\theta^I\right)\hat{\alpha}_L$,
- upper limit on output $y_U = G^T\left(x,\theta^I\right)\hat{\alpha}U$,
- output value

$$y = \frac{\sum_{j=1}^{C} a_j\, p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right) G^T\left(x,\theta^I\right)\hat{w}_j^i}{\sum_{j=1}^{C} a_j\, p\left(y_f \left| m_j^1, \Sigma_j^1\right.\right)}.$$

Algorithm 1 An algorithm for data modeling in presence of uncertainties.

Require: Training data pairs $\left\{x(j), y(j)\right\}_{j=0,1,\cdots,N}$ (input and output variables have been normalized to have zero means and unity variances).

1: Choose the number of maximum epochs $E_{\max}$. A clustering on the input-output data of training set, via Gaussian mixture models of [9] with a common diagonal covariance for all clusters, can be performed to choose the
   - Number of clusters (i.e. rules) K;
   - Initial guess about the clusters' centres θ_{-1};
   - Learning rate μ for α equal to the variance of the clusters on output dimensions;
   - Learning rates $\left(\mu_{\theta(1)}, \mu_{\theta(2)}, \cdots, \mu_{\theta(K_n)}\right)$ equal to the variances of different elements of θ_{-1}.

These choices are required at step 1 of algorithm 2.

2: Identify fuzzy filter parameters $\left(\alpha^I, \theta^I\right)$ using algorithm2.

3: Solve (26) and (27) to compute $\hat{\alpha}_L$ and $\hat{\alpha}_U$ using Lemma 1 (i.e. simulate the model (7), (8) and (9) taking some $\lambda > 0$ from $t = 0$ to the time that solution takes to converge).

4: Compute $\left\{z_j\right\}_{j=0,1,\cdots,N}$ as per (28) and estimate the parameters of the finite mixture model (2) using e.g. the algorithms of [9].

5: **for** $i = 1$ **to** S **do**

6: **for** $j = 1$ **to** C **do**

7: Obtain the data set DP_j^i as per(29). The parameter ε in (29) can be chosen in such a way that all data pairs for $x \in IG_i$, whose filtered output value y_f lies at $\pm 3\sqrt{\Sigma_j^1}$ from mean m_j^1, are included in DP_j^i.

8: Compute $\hat{w}_j^i$ based on (30) using recursive least-squares (RLS) algorithm. Perform several epochs of the RLS algorithm till either the parameters converge or the maximum number of epochs $E_{\max}$ is reached.

9: **end for**

10: **end for**

11. **return** $\left(\alpha^I, \theta^I\right)$; $\left(\hat{\alpha}_L, \hat{\alpha}_U\right)$; $\left\{a_j, m_j^1, \Sigma_j^1\right\}_{j=1,2,\cdots,C}$; $\left\{\hat{w}_j^i\right\}_{i=1,2,\cdots,S;\, j=1,2,\cdots,C}$.

---

5 AN EXAMPLE

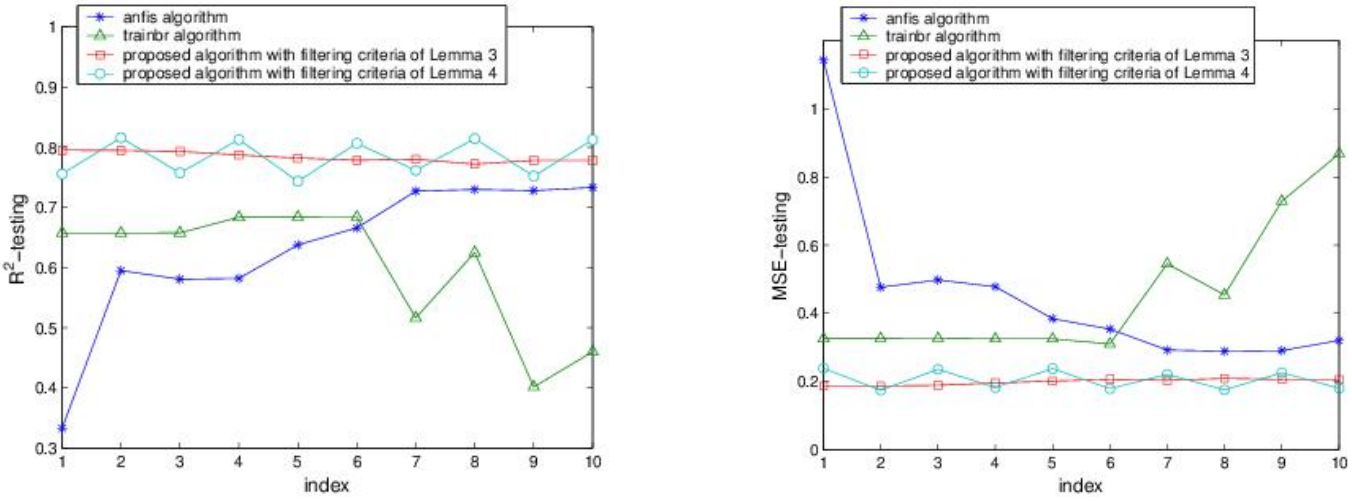
To illustrate the proposed approach, a problem of development of a model mapping 10-dimensional input vector to the output scalar is considered. Table 3 and 4 in B list the training data for an identification of the model. The testing data for the validation of the model are listed in table 5 of B. This data have been taken from a case study of [30] dealing with the modeling of biological activity of 3-phosphoinositide-dependent protein kinase-1 (PDK1) inhibitors. The input and output variables were normalized to have zero means and unity variances. The normalized data have been used to develop a model using some standard methods and the proposed method.

We start with the training of a Sugeno-type fuzzy model using a built-in training algorithm in the MATLAB Fuzzy Logic Toolbox ("anfis" command). The "anfis" algorithm is a standard fuzzy modeling technique that combines the least-square and backpropagation gradient descent methods to identify the parameters of the fuzzy model. The structures and initial parameters of the fuzzy models for "anfis" training were generated using subtractive clustering (MATLAB Fuzzy Logic Toolbox command "genfis2" with a "range of influence" of the cluster center for each input and output dimension varying from 1 to 3). The "range of influence" affects the structure and initial parameters of the generated model. Table 6 in C lists the performance of "anfis" algorithm in terms of coefficient of determination (R^2) and mean squared error (MSE) on training and testing data for different models (generated at different "range of influence"). The different values of "range of influence" were considered to search for an optimal structure of the fuzzy model. However, in case of our data, the "range of influence" couldn't be increased beyond 2.82 since this resulted in a fuzzy model of less than two rules (i.e. not a valid model for "anfis" training). The training of the model continued till 100 epochs.

Bayesian regularized neural networks are considered suitable for data modeling in presence of uncertainties, see e.g. [4], [5] and [6] in context of structure-activity modeling. Bayesian regularization provides an optimal level of regularization to the modeling problem by applying Bayes' theorem [40]. For a Bayesian regularized training consider a 3-layer feed-forward neural network with hyperbolic tangent sigmoid transfer function in the first and second layer while a linear transfer function in the output layer. The training algorithm has been run until the number of effective parameters has converged. Table 7 in C lists the performance of Bayesian regularization backpropagation learning algorithm (MATLAB Neural Network Toolbox command "trainbr") for different number of neurons in the layers of the network.

In tables 6, 7 a poor performance of "anfis" and "trainbr" algorithms on testing data (despite a good performance on training data) indicates that a large uncertainties exit in inputs-output relationships and therefore its modeling is a challenge. Tables 6, 7 provide a reference for comparing the performance of the proposed modeling method. Algorithm 1 was used for data modeling with filtering criteria of Lemma 3 and Lemma 4. First of all, input vectors were divided into different groups by performing clustering using Gaussian mixer models [9] on the total input data. The clustering method of [9] resulted in 25 different clusters (i.e. $S = 25$ and 25 input groups $IG_1, IG_2, \cdots, IG_{25}$). The numbers of maximum epochs $E_{\max}$ in algorithm 1 was taken equal to 10. At step 3 of algorithm 1, $\lambda = 100$ was taken and the simulation of model(7), (8) and (9) runs from $t = 0$ to $t = 10$. Tables 8 and 9 in C state the performance of the proposed method. The filtering algorithm of Lemma 3 for different values of γ was used in table 8. Each of five filtering algorithms of table 2 was used in table 9 for a smaller value of p (i.e. $p = 2$) as well as for a fairly large value of p (i.e. $p = 2\ln(K)$, where K is the number of rules in filter).

While modeling the data in presence of uncertainties, the model is typically overtrained leading to a loss of generalization performance as observed in **table** 6 and 7. A comparison among tables 6, 7, 8 and 9 **verifies** the effectiveness of the algorithm 1. The generalization performance of a model is assessed by its performance on the testing data. Figure 1 compares the generalization performance of the studied models. The better performance of the filtering based method is clearly observable from figure 1. Even the worst performance of algorithm 1 is better than the best performance of "anfis" and "trainbr" algorithms. For an illustration, figure 2 shows the model predicted value as well as the lower and upper limits when algorithm 1 was used with the filtering algorithm $A_{5,2\ln(K)}$.



(a) A comparison in terms of R^2 - testing (b) A comparison in terms of MSE - testing

Fig. 1 The generalization performance comparison of the proposed algorithm with "anfis" and "trainbr" algorithms.

Fig. 2 An illustration of data modeling using algorithm 1 with the filtering algorithm $A_{5,2\ln(K)}$.

6 Concluding Remarks

The authors believe that fuzzy filtering methods have much to offer in real-world modeling applications [34, 18, 35, 36, 28, 19, 30, and 29]. Our investigations with many modeling problems related to life science have shown that the standard neuro/fuzzy modeling techniques fail to the process behavior. This stydy outlines a fuzzy filtering based modeling technique that may be beneficial in the handling of uncertain processes. The proposed methos relies on the recently developed fuzzy filtering algorithms [21], [22] and stochastic modeling of the uncertainties (which are separated from data through the fuzzy filter) using finite mixture models.

The major limitation of our method is that it was a multi-step procedure involving different computational algorithms and therefore it was not easy to study the overall performance of the algorithm in an analytical manner.

Appendix A The Filtering Algorithm

Algorithm 2 Fuzzy filtering algorithm [31].

Require: Data pairs $\{x(j), y(j)\}_{j=0,1,\cdots,N}$.

1: Choose the number of rules K; initial guess about the antecedents θ_{-1}; positive learning rate μ for α; positive learning rates $(\mu_{\theta(1)}, \mu_{\theta(2)}, \cdots, \mu_{\theta(L)})$ for different elements of θ; number of maximum epochs $E_{\max}$.

2: Set data index $j = 0$; epoch count $EC = 0$; initial guess about consequents $\alpha_{-1} = 0$.

3: Choose a filtering criterion: either of Lemma 3 or of Lemma 4.

4: **while** $EC < EC_{\max}$ do

5: **while** $j \le N$ **do**

6: **if** filtering criterion of Lemma 3 **then**

7: Choose a value of $\gamma \ge -1$.

8: Define $r(\theta)$ as(23). Let $s^*(\theta)$ be the unique solution of following linear least squares problem:

$$s^*(\theta) = \arg\min_s \left[\left\| r(\theta) + r'(\theta) s \right\|^2 \right],$$

where $r'(\theta)$ is the Jacobian matrix of vector r with respect to θ, determined by the method of finite-differences. The Jacobian $r'(\theta)$ is a full rank matrix, since the main diagonal of the diagonal matrix Σ has positive entries.

9: Compute θ_j based on (25) using a Gauss-Newton like algorithm: $\theta_j = \theta_{j-1} + s^*(\theta_{j-1})$.

10: Compute α_j using (15).

11: **else** {filtering criterion of Lemma 4}

12: Choose an algorithm out of the 5 different algorithms $(A_{1,p}, A_{2,p}, A_{3,p}, A_{4,p}, A_{5,p})$ and a value of $p(2 \le p \le \infty)$.

13: Define $L_j(\alpha, \theta), \phi(e), P_\phi(y, \bar{y})$ as per table 2 depending upon the choice of algorithm.

14: Define $r_q(\theta)$ as (24) and let $s_q^*(\theta)$ be the unique solution of following linear least-squares problem:

$$s_q^*(\theta) = \arg\min_s \left[\left\| r(\theta) + r_q'(\theta) s \right\|^2 \right], \text{ where } r_q'(\theta) \text{ is the Jacobian matrix of vector } r_q \text{ with respect to } \theta.$$

15: Compute θ_j based on (25) using a Gauss-Newton like algorithm: $\theta_j = \theta_{j-1} + s_q^*(\theta_{j-1})$.

16: Compute α_j using (19) with the learning rate provided by (20).

17: **end if**

18: $j \leftarrow j + 1$

19: **end while**

20: $EC \leftarrow EC + 1; \alpha_{-1} = \alpha_N; \theta_{-1} = \theta_N; j = 0.$

21: **end while**

22: **return** identified fuzzy filter parameters $\alpha^I = \alpha_N$ and $\theta^I = \theta_N$.

Appendix B The Training and Testing Data Sets

Table 3 The training data

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y
1.5560	0.5820	0	0	0.7340	0	0.5250	3.3430	0.1720	0	5.5677
2.6820	0.5910	0	7.0000	0.9000	1.0000	0.6270	-0.4170	0.1710	2.2800	5.0540
2.7560	0.5920	0	56.0000	1.7180	0	0.5430	1.1090	0.1580	2.4220	7.4685
2.8830	0.5860	0	18.0000	2.6000	0	0.4440	2.4370	0.1760	2.1790	7.2218
1.9050	0.5760	0	0	0.5720	0	0.2870	2.1160	0.1980	1.4350	5.7447
2.4360	0.5770	0	0	0.8830	0	0.2800	1.2220	0.2030	1.6640	6.0000
2.5320	0.5940	0	22.0000	2.6090	0	0.4430	3.0900	0.1690	1.8530	7.3979
2.2320	0.5780	0	3.0000	1.7960	0	0.4200	3.0780	0.1770	1.4910	5.1278
3.1400	0.5830	0	4.0000	1.7090	0	0.5550	2.7950	0.1900	2.0540	6.0000
2.7120	0.5950	0	14.0000	1.5890	0	0.4870	2.4230	0.1700	1.9670	7.5376
3.1770	0.5860	1.0000	38.0000	1.5260	1.0000	0.5430	2.3190	0.1420	2.7630	7.4089
2.3670	0.6000	0	6.0000	0.8880	0	0.2990	2.4810	0.1970	1.8590	5.9586
2.5390	0.5810	0	14.0000	0.9230	0	0.3090	1.9720	0.1670	2.1190	6.1739
2.5100	0.5880	0	3.0000	2.4180	0	0.4110	3.3570	0.1820	1.7080	6.5850
2.7040	0.5990	0	14.0000	1.4530	0	0.4980	1.8470	0.1610	2.0580	7.8539
2.7360	0.5710	1.0000	0	1.1710	1.0000	0.4880	2.9360	0.1490	2.2150	7.9208
2.1910	0.5850	0	0	0.9680	0	0.2730	2.6020	0.1710	1.5850	4.5850
2.5480	0.5980	0	16.0000	1.1230	0	0.4490	1.5000	0.1950	2.0220	6.2757
2.5470	0.5820	0	8.0000	0.9630	0	0.3570	1.2370	0.1740	2.0890	7.2596
2.8800	0.5860	0	9.0000	2.6000	0	0.4050	2.2620	0.1670	2.0690	6.5229
2.3550	0.5910	0	6.0000	1.1730	0	0.3620	2.2950	0.1690	1.7470	7.0969
2.7660	0.5930	0	35.0000	1.8780	0	0.6160	1.0160	0.1630	2.3750	5.2676
2.5360	0.6050	0	14.0000	1.0430	0	0.3890	0.9440	0.1860	1.9670	6.5376
2.5350	0.5910	0	3.0000	2.6090	0	0.3600	4.1910	0.1600	1.7780	5.9281
2.5250	0.5820	0	0	0.8530	0	0.3460	1.4950	0.1680	1.4350	6.6990
2.3050	0.5860	0	0	0.8430	0	0.2540	2.6020	0.1710	1.5850	5.5686
2.3670	0.5940	0	6.0000	0.8880	0	0.2870	2.3350	0.1880	1.7470	6.5528
2.2170	0.5850	0	5.0000	0.9680	0	0.3090	1.8490	0.1740	1.6210	4.5850
2.3540	0.5860	0	6.0000	0.8430	0	0.3040	1.8490	0.1950	1.6210	6.4685
2.7480	0.6060	0	35.0000	1.7180	0	0.5690	-0.0930	0.1630	2.2980	5.4089
2.8260	0.5790	1.0000	28.0000	1.4090	1.0000	0.4090	0.9890	0.1500	2.4580	7.7696
2.3670	0.5940	0	6.0000	0.8880	0	0.3110	1.9940	0.1880	1.7470	6.1675
2.3710	0.5850	0	6.0000	1.0130	0	0.3390	2.6610	0.1820	1.7470	5.6383
2.2300	0.5860	0	0	0.8430	0	0.2770	1.3700	0.1830	1.5260	6.2366
2.5130	0.5890	0	3.0000	2.3780	0	0.4320	2.1240	0.1580	1.6560	6.1739
2.8230	0.5830	1.0000	8.0000	1.2090	1.0000	0.4140	2.5450	0.1690	2.0540	7.6778
2.8850	0.5790	1.0000	68.0000	1.4090	1.0000	0.4340	0.0610	0.1470	2.6330	7.5376
2.9170	0.5900	1.0000	34.0000	1.5610	1.0000	0.4820	1.5470	0.1640	2.3610	8.3010
2.8070	0.5840	1.0000	8.0000	1.6210	1.0000	0.4480	1.7590	0.1550	2.1040	8.0458
3.0560	0.5910	0	4.0000	1.6240	0	0.5310	4.0470	0.1720	2.1040	4.8539

Table 4 The training data continued from table 3

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y
2.9190	0.5800	1.0000	34.0000	1.6320	1.0000	0.4090	2.8090	0.1550	2.5010	8.3979
2.8270	0.5880	1.0000	8.0000	1.4810	1.0000	0.4690	2.7510	0.1920	2.2310	7.2924
2.7820	0.5870	1.0000	8.0000	1.2930	1.0000	0.4020	1.4730	0.1630	2.1380	8.3010
2.9850	0.5900	1.0000	32.0000	1.9260	1.0000	0.4930	1.2270	0.1780	2.6560	8.3979
2.3510	0.5860	0	6.0000	0.7630	0	0.3140	1.8490	0.1740	1.6210	6.0458
2.8040	0.5780	1.0000	8.0000	1.5090	1.0000	0.4570	1.3550	0.1640	2.0540	8.0000
2.9860	0.5850	1.0000	32.0000	1.7340	1.0000	0.4830	-0.2660	0.1560	2.6840	8.3010
2.9850	0.5850	1.0000	32.0000	1.7340	1.0000	0.4960	0.6910	0.1660	2.6280	8.2218
2.8260	0.5810	1.0000	28.0000	1.3830	1.0000	0.4230	0.0320	0.1660	2.4580	7.8539
2.8300	0.5700	1.0000	16.0000	1.4560	1.0000	0.5950	3.7040	0.1530	2.4410	8.2218
2.5860	0.5910	0	0	2.6090	0	0.3720	2.5740	0.1640	1.8250	5.6038
3.0390	0.5860	0	4.0000	1.5790	0	0.5620	3.5570	0.1640	2.1040	5.0458
2.7510	0.5920	0	35.0000	1.7180	0	0.5400	0.4610	0.1820	2.3370	5.2147
2.9190	0.5900	1.0000	60.0000	1.5610	1.0000	0.4960	2.1490	0.1860	2.4030	7.3468
2.5390	0.5890	0	14.0000	0.9230	0	0.3370	1.7210	0.1900	2.0100	6.5850
2.1770	0.5840	0	5.0000	0.8880	0	0.3600	1.8490	0.1740	1.6210	6.0000
2.9840	0.5840	1.0000	61.0000	1.7030	1.0000	0.4480	0.1420	0.1530	2.6320	8.0000
2.8260	0.5920	1.0000	48.0000	1.3380	1.0000	0.4750	0.6310	0.1630	2.4030	8.0458
2.9840	0.5850	1.0000	32.0000	1.7340	1.0000	0.5110	1.1140	0.1770	2.5140	8.3010
3.2700	0.6050	0	38.0000	4.0870	0	0.7720	3.4800	0.1440	2.3780	8.3010
2.8240	0.5880	1.0000	32.0000	1.4810	1.0000	0.4270	2.2780	0.1560	2.1590	8.2218
2.8230	0.5830	1.0000	8.0000	1.2090	1.0000	0.4360	1.3940	0.1700	2.0540	8.0969
2.8260	0.5830	1.0000	28.0000	1.4180	1.0000	0.4290	0.3730	0.1650	2.5010	7.5686
2.9840	0.5800	1.0000	32.0000	1.9340	1.0000	0.4570	1.7720	0.1540	2.6560	8.5229
2.9840	0.5840	1.0000	32.0000	1.7030	1.0000	0.4430	-0.1480	0.1630	2.5980	8.3010
2.9830	0.5870	1.0000	32.0000	1.7030	1.0000	0.4330	0.6000	0.1650	2.5680	8.0969

Table 5 The testing data

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y
2.6520	0.5880	0	3.0000	1.5390	0	0.3120	4.5780	0.1560	1.2750	6.3536
2.5260	0.5940	0	3.0000	2.6090	0	0.4390	2.6100	0.1900	1.7780	7.2218
2.5520	0.5940	0	3.0000	2.6090	0	0.4990	4.0210	0.1820	1.9570	5.6737
2.5270	0.6020	0	0	2.4890	0	0.5130	2.8320	0.1730	1.7420	6.2366
2.8840	0.5830	0	10.0000	2.6800	0	0.3940	2.3240	0.1510	2.0690	6.5376
2.8840	0.5830	0	10.0000	2.6800	0	0.3940	2.3240	0.1510	2.0690	6.5376
2.5450	0.5770	0	0	0.8830	0	0.3270	1.9950	0.1850	2.1920	6.0044
2.4920	0.5770	0	6.0000	0.8830	0	0.2860	2.1000	0.1880	1.7780	6.2366
2.3640	0.5940	0	6.0000	0.8430	0	0.3110	2.3350	0.2060	1.7470	6.5528
3.1520	0.5890	0	21.0000	1.9940	0	0.5820	3.4790	0.1640	2.3340	6.1249
2.5390	0.5890	0	7.0000	0.9230	0	0.3220	1.1180	0.1690	1.9460	6.9208
2.5350	0.5820	0	3.0000	2.6090	0	0.3720	3.3080	0.1610	1.7780	6.5686
3.0390	0.5790	0	4.0000	1.5790	0	0.5690	2.5890	0.1750	2.0010	5.3979
2.1420	0.5840	0	0	0.8880	0	0.3880	2.6020	0.1900	1.5850	6.2924
2.9840	0.5850	1.0000	32.0000	1.7340	1.0000	0.5110	1.1140	0.1960	2.5140	8.0000
2.5790	0.5880	0	0	2.6890	0	0.3420	2.6370	0.1810	1.8250	6.0132
2.5860	0.5880	0	3.0000	2.6890	0	0.3360	2.8110	0.1590	1.9630	6.6990
2.5470	0.5820	1.0000	8.0000	0.9630	1.0000	0.3870	1.0270	0.1670	2.0540	7.7447
2.8160	0.5860	1.0000	8.0000	1.5430	1.0000	0.4130	0.5790	0.1710	2.2580	8.0000
2.8310	0.5790	1.0000	28.0000	1.4090	1.0000	0.4200	1.2910	0.1700	2.5010	7.9586
2.8260	0.5830	1.0000	32.0000	1.5210	1.0000	0.4360	2.5290	0.1640	2.2530	7.7212
3.2950	0.5920	1.0000	56.0000	1.3380	1.0000	0.3650	1.7540	0.1630	2.6230	7.2441
2.9100	0.5960	1.0000	48.0000	1.3380	1.0000	0.4990	1.1170	0.1600	2.4630	8.0458
2.8250	0.5830	1.0000	8.0000	1.5210	1.0000	0.4190	1.6510	0.1730	2.1840	8.5229
2.9480	0.5850	1.0000	36.0000	1.6010	1.0000	0.4550	1.6650	0.1610	2.4580	8.0969
2.6970	0.6090	0	14.0000	1.3730	0	0.4510	1.3910	0.1730	2.0580	7.1739
2.7510	0.6060	0	56.0000	1.7180	0	0.6020	0.5090	0.1640	2.3430	5.3768
2.8240	0.5880	1.0000	8.0000	1.4810	1.0000	0.4020	3.0310	0.1860	2.1380	7.4318
3.1550	0.5890	1.0000	56.0000	1.2930	1.0000	0.3770	1.6080	0.1550	2.5730	7.4089
2.9950	0.5950	1.0000	28.0000	1.7480	1.0000	0.4750	0.4890	0.1570	2.5340	7.5086
3.2560	0.6030	0	19.0000	3.9220	0	0.7400	3.8200	0.1440	2.2950	8.1871
2.7940	0.5780	1.0000	8.0000	1.3730	1.0000	0.4420	1.9290	0.1600	2.1380	8.5229
2.9190	0.5830	1.0000	34.0000	1.6410	1.0000	0.4250	1.8920	0.1600	2.5010	8.3979

Appendix C The Performance Tables

Table 6 The performance of "anfis" training algorithm

Initial fuzzy model	R^2 - training	R^2 - testing	MSE-training	MSE-testing
"genfis2", "range of influence" = 1	0.9759	0.3328	0.0321	1.1430
"genfis2", "range of influence" = 1.2	0.9203	0.5950	0.1062	0.4764
"genfis2", "range of influence" = 1.4	0.9206	0.5806	0.1059	0.4976
"genfis2", "range of influence" = 1.6	0.9159	0.5820	0.1121	0.4779
"genfis2", "range of influence" = 1.8	0.8970	0.6376	0.1373	0.3838
"genfis2", "range of influence" = 2	0.8914	0.6658	0.1448	0.3537
"genfis2", "range of influence" = 2.2	0.8859	0.7269	0.1521	0.2922
"genfis2", "range of influence" = 2.4	0.8847	0.7296	0.1537	0.2887
"genfis2", "range of influence" = 2.6	0.8835	0.7278	0.1553	0.2906
"genfis2", "range of influence" = 2.82	0.8788	0.7331	0.1615	0.3201

Table 7 The performance of "trainbr" training algorithm

Number of neurons in first, second, third layer	R^2 - training	R^2 - testing	MSE-training	MSE-testing
2,4,1	0.7488	0.6573	0.3359	0.3255
2,8,1	0.7484	0.6574	0.3364	0.3261
2,12,1	0.7484	0.6576	0.3365	0.3262
3,4,1	0.9028	0.6835	0.1305	0.3246
3,8,1	0.9026	0.6837	0.1308	0.3246
3,12,1	0.8997	0.6837	0.1346	0.3098
4,4,1	0.9723	0.5162	0.0371	0.5452
4,8,1	0.9593	0.6248	0.0547	0.4538
4,12,1	0.9807	0.4015	0.0258	0.7292
5,4,1	0.9945	0.4598	0.0073	0.8684

Table 8 The performance of algorithm 1 with the filtering criteria of Lemma 3

γ	R^2 - training	R^2 - testing	MSE-training	MSE-testing
-1	0.9212	0.7952	0.1056	0.1868
-0.8	0.9210	0.7947	0.1059	0.1867
-0.6	0.9210	0.7931	0.1058	0.1883
-0.4	0.9234	0.7871	0.1027	0.1953
-0.2	0.9235	0.7819	0.1026	0.2007
0.01	0.9237	0.7783	0.1023	0.2059
0.3	0.9238	0.7798	0.1022	0.2032
0.5	0.9244	0.7719	0.1012	0.2088
0.7	0.9241	0.7781	0.1019	0.2055
0.9	0.9246	0.7780	0.1012	0.2056

Table 9 The performance of algorithm 1 with the filtering criteria of Lemma 4. Each of five filtering algorithms of table 2 is used for a smaller value of $p(i.e.\ p=2)$ as well as for a fairly large value of $p(i.e.\ p=2\ln(K),$ where K is the number of rules in the filter).

algorithm	R^2 - training	R^2 - testing	MSE-training	MSE-testing
$A_{1,2}$	0.9015	0.7556	0.1313	0.2381
$A_{1,2\ln(K)}$	0.9107	0.8157	0.1195	0.1739
$A_{2,2}$	0.9112	0.7572	0.1184	0.2355
$A_{2,2\ln(K)}$	0.9165	0.8129	0.1120	0.1821
$A_{3,2}$	0.9194	0.7434	0.1076	0.2376
$A_{3,2\ln(K)}$	0.9336	0.8065	0.0890	0.1788
$A_{4,2}$	0.9206	0.7615	0.1063	0.2207
$A_{4,2\ln(K)}$	0.9251	0.8143	0.1004	0.1750
$A_{5,2}$	0.9274	0.7520	0.0970	0.2253
$A_{5,2\ln(K)}$	0.9235	0.8125	0.1027	0.1803

REFERENCES

1. Abonyi, J., Babuska, R., Szeifert, F.: Modified Gath-Geva Fuzzy Clustering for Identification of Takagi-Sugeno Fuzzy Models. IEEE Trans. on System, Man and Cybernetics, Part B pp. 612–621 (2002)
2. Babuska, R.: Fuzzy Modeling for Control. Kluwer Academic Publishers, Boston (1998)
3. Babuska, R., Verbruggen, H.: Constructing fuzzy models by product space clustering. In: H. Hellendoorn, D. Driankov (eds.) Fuzzy Model Identification: Selected Approaches, pp. 53–90. Springer, Berlin, Germany (1997)
4. Burden, F.R., Ford, M.G., Whitley, D.C., Winkler, D.A.: Use of automatic relevance determination in qsar studies using bayesian neural networks. J. Chem. Inf. Comput. Sci. **40**, 1423–1430 (2000)
5. Burden, F.R., Winkler, D.A.: New qsar methods applied to structure-activity mapping and combinatorial chemistry. J. Chem. Inf. Comput. Sci. **39**, 236–242 (1999)
6. Burden, F.R., Winkler, D.A.: Robust qsar models using bayesian regularized neural networks. J. Med. Chem. **42**, 3183–3187 (1999)
7. Burger, M., Engl, H., J.Haslinger, U.Bodenhofer: Regularized data-driven construction of fuzzy controllers. J. Inverse and Ill-posed Problems **10**, 319–344 (2002)
8. Chen, D.S., Jain, R.C.: A robust back propagation learning algorithm for function approximation. IEEE Trans. Neural Networks **5**, 467–479 (1994)
9. Figueiredo, M.A.T., Jain, A.K.: Unsupervised Learning of Finite Mixture Models. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(3), 381–396 (2002)
10. Gentile, C.: The robustness of the p-norm algorithms. Machine Learning **53**(3), 265–299 (2003)
11. Gonz'alez, A., P'erez, R.: Completeness and consistency conditions for learning fuzzy rules. Fuzzy Sets and Systems **96**, 37–51 (1998)
12. Herrera, F., Lozano, M., Verdegay, J.: Generating fuzzy rules from examples using genetic algorithms. In: Proc. 5th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'94), pp. 675–680. Paris, France (1994)
13. Hong, X., Harris, C.J., Chen, S.: Robust neurofuzzy rule base knowledge extraction and estimation using subspace decomposition combined with regularization and d-optimality. IEEE Trans. Syst., Man., Cybern. B **34**(1), 598–608 (2004)
14. Jang, J.S.R.: ANFIS: Adaptive-network-based fuzzy inference systems. IEEE Transactions on Systems, Man, and Cybernetics **23**, 665–685 (1993)
15. Jang, J.S.R., Sun, C.T., Mizutani, E.: Neuro-Fuzzy and Soft Computing; a Computational Approach to Learning and Machine Intelligence. Prentice-Hall, Upper Saddle River (1997)
16. Johansen, T.: Robust identification of takagi-sugeno-kang fuzzy models using regularization. In: Proc. IEEE conf. Fuzzy Systems, pp. 180–186. New Orleans, USA (1996)
17. Kim, J., Suga, Y., Won, S.: A new approach to fuzzy modeling of nonlinear dynamic systems with noise: Relevance vector learning mechanism. IEEE Trans. on Fuzzy Systems **14**(2), 222–231 (2006)
18. Kumar, M., Arndt, D., Kreuzfeld, S., Thurow, K., Stoll, N., Stoll, R.: Fuzzy techniques for subjective workload score modelling under uncertainties. IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics **38**(6), 1449–1464 (2008)
19. Kumar, M., Stoll, N., Kaber, D., Thurow, K., Stoll, R.: Fuzzy filtering for an intelligent interpretation of medical data. In: Proc. IEEE International Conference on Automation Science and Engineering (CASE 2007), pp. 225–230. Scottsdale, Arizona USA (2007)
20. Kumar, M., Stoll, N., Stoll, R.: An energy-gain bounding approach to robust fuzzy identification. Automatica **42**(5), 711–721 (2006)
21. Kumar, M., Stoll, N., Stoll, R.: Adaptive fuzzy filtering in a deterministic setting. IEEE Transactions on Fuzzy Systems **17**(4), 763–776 (2009)
22. Kumar, M., Stoll, N., Stoll, R.: On the estimation of parameters of takagi-sugeno fuzzy filters. IEEE Transactions on Fuzzy Systems **17**(1), 150–166 (2009)
23. Kumar, M., Stoll, R., Stoll, N.: Robust adaptive fuzzy identification of time-varying processes with uncertain data. Handling uncertainties in the physical fitness fuzzy approximation with real world medical data: An application. Fuzzy Optimization and Decision Making **2**(3), 243–259 (2003)
24. Kumar, M., Stoll, R., Stoll, N.: Robust adaptive identification of fuzzy systems with uncertain data. Fuzzy Optimization and Decision Making **3**(3), 195–216 (2004)
25. Kumar, M., Stoll, R., Stoll, N.: Robust solution to fuzzy identification problem with uncertain data by regularization. Fuzzy approximation to physical fitness with real world medical data: An application. Fuzzy Optimization and Decision Making **3**(1), 63–82 (2004)
26. Kumar, M., Stoll, R., Stoll, N.: Deterministic approach to robust adaptive learning of fuzzy models. IEEE Trans. Syst., Man., Cybern. B **36**(4), 767–780 (2006)
27. Kumar, M., Stoll, R., Stoll, N.: A robust design criterion for interpretable fuzzy models with uncertain data. IEEE Trans. on Fuzzy Systems **14**(2), 314–328 (2006)
28. Kumar, M., Thurow, K., Stoll, N., Stoll, R.: Fuzzy handling of uncertainties in modeling the inhibition of glycogen synthase kinase-3 by paullones. In: Proc. IEEE International Conference on Automation Science and Engineering (CASE 2007), pp. 237–242. Scottsdale, Arizona USA (2007)
29. Kumar, M., Thurow, K., Stoll, N., Stoll, R.: Robust fuzzy mappings for QSAR studies. European Journal of Medicinal Chemistry **42**, 675–685 (2007)
30. Kumar, M., Thurow, K., Stoll, N., Stoll, R.: A fuzzy system for modeling the structure-activity relationships in presence of uncertainties. In: Proc. IEEE International Conference on Automation Science and Engineering (CASE 2008), pp. 1025–1030. Washington DC, USA (2008)
31. Kumar, M., Thurow, K., Stoll, N., Stoll, R.: Fuzzy filtering: A mathematical theory and applications in life science. In: A.T. Azar (ed.) Fuzzy Systems, pp. 129–146. Intech, Olajnica, Croatia (2010)
32. Kumar, M., Weippert, M., Kreuzfeld, S., Stoll, N., Stoll, R.: A fuzzy filtering based system for maximal oxygen uptake prediction using heart rate variability analysis. In: Proc. IEEE International Conference on Automation Science and Engineering (CASE 2009), pp. 604–608. Bangalore, India (2009)
33. Kumar, M., Weippert, M., Stoll, N., Stoll, R.: A mixture of fuzzy filters applied to the analysis of heartbeat intervals. Fuzzy Optimization and Decision Making **9**(4), 383–412 (2010)
34. Kumar, M., Weippert, M., Vilbrandt, R., Kreuzfeld, S., Stoll, R.: Fuzzy evaluation of heart rate signals for mental stress assessment. IEEE Transactions on Fuzzy Systems **15**(5), 791–808 (2007)
35. Kumar, S., Kumar, M., Stoll, R., Kragl, U.: Handling uncertainties in toxicity modelling using a fuzzy filter. SAR and QSAR in Environmental Research **18**(7-8), 645–662 (2007)
36. Kumar, S., Kumar, M., Thurow, K., Stoll, R., Kragl, U.: Fuzzy filtering for robust bioconcentration factor modelling. Environmental Modelling & Software **24**(1), 44–53 (2009)

37. Lin, C.J., Chen, C.H., Lin, C.T.: Efficient self-evolving evolutionary learning for neuro-fuzzy inference systems. IEEE Trans. on Fuzzy Systems **16**(6), 1476–1490 (2008)
38. Liska, J., Melsheimer, S.S.: Complete design of fuzzy logic systems using genetic algorithms. In: Proc. of the 3rd IEEE Int. Conf. on Fuzzy Systems, pp. 1377–1382 (1994)
39. Lughofer, E.: FLEXFIS: A Robust Incremental Learning Approach for Evolving TS Fuzzy Models. IEEE Trans. on Fuzzy Systems **16**(6), 1393–1410 (2008)
40. MacKay, D.J.C.: Bayesian interpolation. Neural Computation **4**, 415–447 (1992)
41. McLachlan, G., Krishnan, T.: The EM Algorithm and Extensions. New York: John Wiley & Sons (1997)
42. McLachlan, G., Peel, D.: Finite Mixture Models. New York: John Wiley & Sons (2000)
43. Nauck, D., Kruse, R.: A neuro-fuzzy approach to obtain interpretable fuzzy systems for function approximation. In: Proc. IEEE International Conference on Fuzzy Systems 1998 (FUZZ-IEEE'98), pp. 1106–1111. Anchorage, AK (1998)
44. Nozaki, K., Ishibuchi, H., Tanaka, H.: A simple but powerful heuristic method for generating fuzzy rules from numerical data. Fuzzy Sets and Systems **86**, 251–270 (1997)
45. Shan, J.J., Fu, H.C.: A fuzzy neural network for rule acquiring on fuzzy control systems. Fuzzy Sets and Systems **71**, 345–357 (1995)
46. Simon, D.: Design and rule base reduction of a fuzzy filter for the estimation of motor currents. International Journal of Approximate Reasoning **25**, 145–167 (2000)
47. Simon, D.: Training fuzzy systems with the extended kalman filter. Fuzzy Sets and Systems **132**, 189–199 (2002)
48. Thrift, P.: Fuzzy logic synthesis with genetic algorithms. In: Proc. of the 4th Int. Conf. on Genetic Algorithms, pp. 509–513 (1991)
49. Wang, L.X., Mendel, J.M.: Generating fuzzy rules by learning from examples. IEEE Trans. on Systems, Man, and Cybernetics **22**(6), 1414–1427 (1992)
50. Wang, W., Vrbanek, J.: An evolving fuzzy predictor for industrial applications. IEEE Trans. on Fuzzy Systems **16**(6), 1439–1449 (2008)
51. Wang, W.Y., Lee, T.T., Liu, C.L., Wang, C.H.: Function approximation using fuzzy neural networks with robust learning algorithm. IEEE Trans. Syst., Man., Cybern. B **27**, 740–747 (1997)
52. Xia, Y., Kamel, M.S.: A cooperative recurrent neural network for solving *L*1 estimation problems with general linear constraints. Neural Comput. **20**(3), 844–872 (2008)
53. Yu, W., Li, X.: Fuzzy identification using fuzzy neural networks with stable learning algorithms. IEEE Trans. on Fuzzy Systems **12**(3), 411–420 (2004)
54. Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and decision processes. IEEE Trans. on Systems, Man, and Cybernetics **3**, 28–44 (1973)
55. Zadeh, L.A.: The role of fuzzy logic in the management of uncertainty in expert systems. Fuzzy Sets Systems **11**, 199–227 (1983)