Short Contribution

AUTOMATED NONLINEAR SYSTEM MODELING WITH MULTIPLE FUZZY NEURAL NETWORKS AND KERNEL SMOOTHING

WEN YU
Departamento de Control Automatico, CINVESTAV-IPN
Av.IPNI 2508, Mexico D.F., 07360, Mexico

XIAOOU LI
Departamento de Computacion, CINVESTAV-IPN
Av.IPNI 2508, Mexico D.F., 07360, Mexico

This paper presents a novel identification approach using fuzzy neural networks. It focuses on structure and parameters uncertainties which have been widely explored in the literatures. The main contribution of this paper is that an integrated analytic framework is proposed for automated structure selection and parameter identification. A kernel smoothing technique is used to generate a model structure automatically in a fixed time interval. To cope with structural change, a hysteresis strategy is proposed to guarantee finite times switching and desired performance.

Keywords: Fuzzy neural networks; kernel smoothing; multiple models; modeling.

1. Introduction

Kernel smoothing (regression) is an estimation technique to find a regression function such that this function is the best-fit line at those data points. It can also be used for the sparse data interpolation. Kernel regression is different from linear regression or polynomial regression, it does not assume any underlying distribution (e.g. normal distribution) to estimate the regression function. That is why kernel regression is categorized as a non-parametric technique. The idea of kernel regression is to form a set of identical weighted functions, called local kernels, from observational data points. Each kernel is assigned a weight based on the distance between the data point and the regression line. The basis kernel function depends only on the radius (or width) between the local data point $x$ and a set of neighboring locations $X$. Kernel regression is a superset of local weighted regression, and is closely related to moving average (MA). $K$ nearest neighbor (KNN), radial basis function (RBF) neural network and support vector machine (SVM). The first successful kernel smoothing approach is the Nadaraya-Watson estimator, where the kernels were used as locally weighted average functions.

Both neural networks and fuzzy logic are universal estimators; they can approximate any nonlinear function to any prescribed accuracy, provided that sufficient hidden neurons and fuzzy rules are available. Recent results show that the fusion procedure of these two different technologies seems to be very effective for nonlinear systems identification. The combination of multiple models method and fuzzy neural networks is considered as a rational approach for modeling complex nonlinear systems. Static neural networks were used as multi-model identifier in Refs. 3 and 6, the switching algorithm was realized by a gating neural network. But the stability could not be guaranteed with this neural switching method. Failure detection based on multiple static neural networks was presented in Ref. 13. Adaptive critic neural networks applied several networks to realize adaptive control. Despite above multiple neural networks, there are few research works on automated structure selection.
In this paper, an integrated analytic framework is proposed for automated neural network structure selection, parameter identification and hysteresis network switching with guaranteed neural identification performance. We first use the kernel smoothing to select the structure of fuzzy neural networks automatically. It is different with our previous work in Ref. 18, where modified least square was used to generate the hidden layer of neural networks. After the membership functions are trained, a hysteresis switching strategy is developed to guarantee desired performance of the fuzzy neural identifiers.

2. Multiple Fuzzy Neural Networks for Nonlinear System Identification

The procedure of the multi-model algorithm given in this paper is shown in Fig. 1. We use a window length of \( N \) to collect data, then the kernel smoothing support method is applied to obtain an approximator in the time interval \([\tau_1, \tau_2]\). Since we can prove that the kernel smoothing model has the same form as a fuzzy system, the corresponding fuzzy rules are constructed. Then, the premise and consequent membership functions of these fuzzy rules are updated after \( k > \tau_2 \). If the plant changes at \( k = N + m \), by the criterion defined in Sec. 4, the fuzzy model is also changed. Now the kernel smoothing with the data window is applied again to generate a new model structure. The membership functions of the new fuzzy system are updated after \( k > N + m + N \).

We can see that our multi-model method is different with the concept of boosting in the area of data mining, they generate multiple models, and derive weights to combine these simple models into a single prediction model. However, we only use one model at each time, we do not combine them.

![Fig. 1. Multi-model algorithm.](image)

Consider a nonlinear discrete-time plant represented by

\[
y(k) = f(x(k), \theta) + e(k)
\]

where

\[
x(k) = [y(k-1), y(k-2), \ldots, y(k-n_y), u(k-d), \ldots, u(k-d-n_u)]^T \in \mathbb{R}^{n+1}
\]

and \( f(\cdot) \) is an unknown nonlinear difference equation representing the plant dynamics, \( u(k) \) and \( y(k) \) are measurable scalar input and output, \( d \) is the time delay, \( \theta \) is an unknown parameter vector associated with a model structure, \( e(k) \) is a bounded observation noise, \( n_y \) and \( n_u \) are the lengths of output and input, \( n_y + n_u = n \), \( n \) is the plant order.

A generic fuzzy model is presented as a collection of fuzzy rules in the following form (Mamdani fuzzy model)

\[
R^j: \text{IF } x_1 \text{ is } A^j_1 \text{ and } x_2 \text{ is } A^j_2 \text{ and } \cdots x_n \text{ is } A^j_n \text{ THEN } \hat{y} \text{ is } B^j
\]

We use \( (j = 1, 2, \ldots, l) \) fuzzy IF-THEN rules to perform a mapping from the input linguistic vector \( x = [x_1, \ldots, x_n] \in \mathbb{R}^n \) to the output \( \hat{y} \). By using product inference, center-average and singleton fuzzifier, the output of the fuzzy logic system can be expressed as

\[
\hat{y} = \frac{\sum_{j=1}^{l} w_j \left( \prod_{i=1}^{n} \mu_{A^j_i}(x_i) \right)}{\sum_{j=1}^{l} \left( \prod_{i=1}^{n} \mu_{B^j_i} \right)}
\]

where \( \mu_{A^j_i} \) and \( \mu_{B^j_i} \) are the membership functions of the fuzzy sets \( A^j_i \) and \( B^j_i \), \( w_j \) is the point at which \( \mu_{B^j_i} = 1 \), \( W = [w_1, \ldots, w_l] \), \( \psi \) is \( l \)-dimension vector function, the element

\[
\psi_k = \frac{\prod_{i=1}^{n} \mu_{A^j_i}(x_i)}{\sum_{j=1}^{l} \left( \prod_{i=1}^{n} \mu_{B^j_i} \right)}
\]

If we use Takagi-Sugeno fuzzy model, the fuzzy rules are

\[
R^j: \text{IF } x_1 \text{ is } A_{1j} \text{ and } x_2 \text{ is } A_{2j} \text{ and } \cdots x_n \text{ is } A_{nj} \text{ THEN } \hat{y} = p^*_0 + p^*_1 x_1 + \cdots + p^*_n x_n
\]
Automated Nonlinear System Modeling with Multiple Fuzzy Neural Networks

where \( j = 1 \ldots l \). The output of the fuzzy logic system can be expressed as

\[
\hat{y} = \sum_{i=1}^{l} \left( p_{i0}^j + p_{i1}^j x_1 + \cdots + p_{ia}^j x_a \right) \psi_i
\]  

(7)

where \( \psi_i \) is defined as in (5). (7) can be also expressed in the form of the Mamdani-type (4),

\[
\hat{y} = W \psi [x(k)]
\]  

(8)

with \( W(k) = [p_0^1, \ldots, p_0^l, p_1^1, \ldots, p_1^l, \ldots, p_a^l] \).

Since the Takagi-Sugeno fuzzy model (7) has the same mathematical expression as the Mamdani fuzzy model (4), in this paper we only discuss Mamdani fuzzy system.

Generally the fuzzy neural networks (4) cannot match the given nonlinear system (1) exactly, the nonlinear system (plant) can be represented as

\[
y(k) = W_0 \phi(x(k)) + \varepsilon(k)
\]  

(9)

where \( \varepsilon(k) \) is defined as the modeling error. The identified nonlinear system can also be written as

\[
y(k) = W_0^\tau \phi(x(k)) + f(k)
\]  

(10)

where \( \hat{f}_k \) is the modeling error, \( W_0^\tau \) and \( W_0 \) are sets of known parameters chosen by the user. In general, \(|\hat{f}(k)| \geq |\varepsilon(k)|\).

From (10) we know that the modeling error \( \hat{f}_k \) depends on the structure of the fuzzy neural network. For many nonlinear processes, the operation conditions vary with time, and the operation environment is complicated. One model is not sufficient to describe the whole plant. Multiple models can then give a better modeling accuracy. Although a single fuzzy neural network (4) can identify any nonlinear process (black-box), the identification error can be big if the fuzzy network structure is not chosen appropriately. Generally speaking, we cannot find the optimal network structure representing the system (1) under all different operation conditions. A possible solution is to use several networks and select the best one by a proper switching algorithm.

We define \( I_1, \ldots, I_r \) as fuzzy neural identifiers, whose outputs are \( \hat{y}_1, \ldots, \hat{y}_r \). We use a switching policy to choose an identifier \( I_i \) such that a performance index of the identification error between the output of this neural network and the plant is minimized.

The multiple fuzzy neural identifiers are presented as

\[
I_r: \hat{y}_r(k) = W_r^\tau(k) \phi(x(k))
\]  

(11)

where \( \sigma = \{1, 2, \ldots, r\} \) and \( r \) is the total number of fuzzy neural identifiers.

The modeling objective of the multiple fuzzy neural networks is to find a suitable structure and a switching policy such that a performance index is minimized. In the next section we will propose an automatic selection method for the multiple fuzzy neural networks.

3. Automated Structure Selection Via Kernel Smoothing Method

At the time interval \( k \in [\tau_1, \tau_2] \), we assume the observation data \( [y(k), x(k)] \) come from the model

\[
f(x) = \phi(x) + \varepsilon
\]

where \( \phi(x) \) is an unknown smooth function, \( \varepsilon \) is a noise. A kernel smooth estimate \( \hat{\phi} \) is calculated from all data points, it is a weighted average

\[
\hat{\phi}(x) = N^{-1} \sum_{k=1}^{\tau_2} W_k(x(k)) y(k)
\]

where \( N = \tau_2 - \tau_1 + 1 \).

The weight sequence \( W_k \) can be described by a density function with a scale parameter which adjusts the size and the form of the weight near the data \( x(k) \). It is quite common to refer to this shape function as a kernel \( K[z] \). The kernel is a continuous, bounded and symmetric real function which integrates to one, i.e., \( \int f^2 dz = 1 \).

The weight sequence for the kernel smoothers is defined by

\[
W_k(x(k)) = K_k(x(k))/\hat{f}_k(x)
\]

where

\[
\hat{f}_k(x) = N^{-1} \sum_{k=1}^{\tau_2} K_k(x(k)), K_z < 0
\]

is a Gaussian function

\[
K_z(x(k)) = a \exp \left( -\frac{(x - x_k)^2}{2\sigma^2} \right), a > 0
\]  

(12)

Therefore the Nadaraya-Watson kernel regression\(^{8}\) for the data \( [y(k), x(k)] \) in \( k \in [\tau_1, \tau_2] \) is

\[
\hat{\phi}(x) = \frac{\sum_{k=1}^{\tau_2} K_k(x(k)) y(k)}{\sum_{k=1}^{\tau_2} K_k(x(k))}
\]  

(13)

Statistical analysis of the Nadaraya-Watson kernel regression is difficult because it depends on the ratio
of two random variables \( f_1 \) and \( K_0 \). In many important applications, such as signal processing and automatic control, a simple form of the kernel regression is used, which is easy to be analyzed statistically. In this form, all random variables have constant probability density functions. So the summations involving the kernel functions are equivalent to Monte Carlo integrations.

\[
K_0(x(k)) = \frac{N}{\tau_2}
\]

The Nadaraya-Watson kernel regression has a simple form, called the Priestley-Chao regression. The full width at half maximum method is then decided by the total number of the data, for example \( \sigma_{\text{max}} = \frac{N}{\tau_2} \). The width of Gaussian function is then

\[
\sigma = \frac{N}{20\sqrt{2\ln(2)}}
\]

where \( N \) is data length, see Fig. 1.

We use these Gaussian kernel functions as the membership functions of the fuzzy systems (4), i.e., the center points of each Gaussian function are \([x^*_j, y^*_j]\) in (4), \( x^*_j \) is center of the Gaussian function \( \mu_{A_j} \) and \( y^*_j \) is the center of the Gaussian function \( \mu_{B_j} \).

So for the data set \([y(k), x(k)] \), \( k \in [1, N] \), we extract fuzzy product rules in the form of (13) or (4).

In the time interval \([t_1, t_2]\), we use the following automatic kernel smoother to construct fuzzy rules.

Step 1: Initialization: select \( \alpha \) and \( \sigma_{\text{max}} \) according to (16) and (15), then calculate the kernel according to (12).

Step 2: Solve the kernel smoothing problem according to (13), which has the same form as the fuzzy system (4).

Step 3: Membership function training: membership function training can be found in our previous papers.15,17

Step 4: Check phase: check if the network construction should be changed. If no, continuous training go to Step 3. Otherwise, terminate the training process, go to network construction phase (Step 2).

4. Multiple Fuzzy Neural Networks Selection

For system identification, the estimation error is caused by both structure uncertainty and parameter uncertainty. The structure uncertainty can be solved by switching in multiple fuzzy neural networks. The parameter uncertainty can be solved by membership functions updating. When the membership functions cannot make the identification error smaller, the model should be changed. When a new fuzzy neural network is obtained, parameter updating method should be applied to minimize the identification error until the parameters (membership functions) converge.

We define the performance index \( J_p(k) \) for \( p \)-th neural identifier as

\[
J(k) = (1 - \alpha)e^2(k) + \frac{\alpha}{N_p} \sum_{t=k-N_p}^{k} e^2(t)
\]

where \( \alpha \) is a design parameter for weighting both the short-term and long term errors, \( N_p \) is smoothing length of fuzzy neural identifier. This performance index is similar as Ref. 9.

\( N_p \) and \( \alpha \) can also be used to overcome the outlier problems. If there are many outliers \( N_p \) and \( \alpha \) should be increased, this means \( J_p(k) \) will be affected less by the outlier \( e(k) \), and affected more by the average error in the past.

The structure switching decision is directed by monitoring the performance index (17). In order to prevent an arbitrarily fast changing due to disturbances, a hysteresis decision algorithm is needed. In this paper, we change fuzzy neural network model only when the membership functions are almost convergent. We define the change function
as \((k > N)\)

\[
\omega(k) = \frac{1}{2} \left| \text{tr}(W(k)W(k)') - \frac{1}{N} \sum_{i=1}^{k} \text{tr}(W(i)W(i)') \right| \\
+ \frac{1}{2} \left| \text{tr}(V(k)V(k)') - \frac{1}{N} \sum_{i=1}^{k} \text{tr}(V(i)V(i)') \right|
\]

where \(V^T(k) = [\sigma_j(k), \sigma_j(k)]\). The hysteresis decision algorithm can be formulated as

\[
r = \begin{cases} 
  r + 1 & \text{if } J(k) > J(k - 1) + h \text{ and } \\
  r & \text{otherwise}
\end{cases}
\]

where \(h > 0\) is the hysteresis constant, \(L\) is the threshold for the weight change. \(r = r + 1\) means the structure of the r-th neural network is not suitable for the coming data, we should start a neural network, now \(\sigma = \{1, 2, \ldots, r, r + 1\}\), see (11). We choose \(L\) such that the switching algorithm (18) can work after the parameters do not significantly affect the fuzzy neural identification. The procedure for automatic selection of multiple fuzzy neural networks is shown in Fig. 2.

In general even for linear time invariant switched systems, the stability of each component-system in the switching loop does not guarantee the stability of entire switched system under arbitrary switching laws. Some conditions on switching policy are needed to stabilize the whole system. Figure 2 shows that our switching policy is very special: it can only switch from model \(r\) to model \(r + 1\) on the time axis. The switching is in a sequence. The stability analysis of this switching system can be found in Ref. 18.

5. Simulation

The dynamic engine model is a two-input and two-output system:

\[
P = k_P \left( \begin{array}{c}
    m_\text{in} \\
    m_\text{in}
  \end{array} \right), \quad N_p = k_N(T_1 - T_2)
\]

\[
m_\text{in} = (1 + k_{\text{in}}\theta + k_{\text{out}}\theta^2)g(P)
\]

\[
m_\text{in} = -k_{\text{in}}N_p - k_{\text{out}}P + k_{\text{out}}N_pP + k_{\text{out}}N_pP^2
\]

(19)

The discrete-time model for \(P\) can be expressed as

\[
P(k + 1) = f_k[P(k), N_p(k), \theta(k), \delta(k), T_\text{in}(k)]
\]

In order to train the neural model, we select inputs as \(\delta = 30\sin(0.06\pi t)\), \(\theta\) is a square wave with amplitude 20, \(\theta(k) = 20\text{Square}(0.04k)\), \(T_\text{in}(k) = 10, x_\text{in} = [10, 500]^T\). The time interval is chosen as \(N = 300\).

The system (19) is a two inputs and two outputs MIMO system. In order to consider all cases, we use every measurable variables, \(P(k), N_p(k), \theta, \delta\) and \(T_\text{in}\) as input, and separate the two outputs as two systems. For manifold press \(P\), we use the following Mamdani fuzzy model

\[
R1: \text{IF } (P(k) \text{ is } A^P_1) \land (N_p(k) \text{ is } A^P_{n1}) \land (\theta \text{ is } A^\theta_1) \land (\delta \text{ is } A^\delta_1) \land (T_\text{in} \text{ is } A^T_{11}) \text{ THEN } P(k + 1) \text{ is } B^P_1
\]
For engine speed \( N_p \), we use the following Mamdani fuzzy model

\[
R^j: IP (P(k) \in C^j_0) \land (N_p(k) \in C^j_1) \land (\theta(k) \in C^j_2) \\
\land (\delta(k) \in C^j_3) \land (T_d(k) \in C^j_4) \text{ THEN} \\
N_p(k + 1) = D^j_p
\]

Since these two fuzzy systems are similar, in this paper we only discuss the manifold press. All of data are normalized such that are within the interval of \([-1, 1]\). We start from \( W(k) \in \mathbb{R}^{15 \times 5}, V(k) \in \mathbb{R}^{15 \times 5} \).

After \( k = 200 \), we use the kernel smoothing and obtain 4 fuzzy rules. Then the membership functions are updated. After \( k > 500 \), the plant is changed as \( g(P(k) = 0.8 \) with \( P < 50.6625 \). The hysteresis constant is selected \( h = 0.05 \), the threshold of weights change is \( L = 1.5 \). According to (17), the fuzzy model is changed. We repeat above procedure. \( \tau_1 = 500 \).

After \( k = 700, \tau_2 = 700, N = 200 \), fuzzy rules from kernel smoothing are obtained.

We compare the performances of our automated structure selection via kernel smoothing (KFNN) approach with normal neural networks identification (MLP),\(^3\) multiple RBF neural networks (MNN),\(^2, 3\) and multi-stage neural identification (MSNN).\(^4\) The multilayer neural networks as in Ref. 9 is \( H_{5.15.1} \) (one hidden layer with 15 nodes), and its learning rate of backpropagation is fixed as \( \eta = 0.05 \). For multiple neural networks,\(^3\) the two neural networks are chosen as \( H_{5.15.1} \) and \( H_{10.1} \), their learning rates are 0.05. The results are shown in Table 1, where the modeling error is calculated as RMS.

### Table 1. Comparisons of four identification methods.

<table>
<thead>
<tr>
<th>Networks</th>
<th>MLP</th>
<th>MNN</th>
<th>MSNN</th>
<th>KFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modeling error</td>
<td>0.34</td>
<td>0.62</td>
<td>0.31</td>
<td>0.59</td>
</tr>
<tr>
<td>Running steps</td>
<td>5000</td>
<td>1000</td>
<td>5000</td>
<td>1000</td>
</tr>
<tr>
<td>Hidden nodes rule number</td>
<td>15</td>
<td>15</td>
<td>10,15</td>
<td>15,10</td>
</tr>
</tbody>
</table>

Further works will be done on closed-loop identification with the automatic selection fuzzy neural networks.

### References

6. G. Lightbody and G. W. Irwin, Nonlinear control structures based on embedded neural system models,
Automated Nonlinear System Modeling with Multiple Fuzzy Neural Networks

435

IEEE Transactions on Neural Networks 8(3) (1997) 553–567.


