2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Modeling via on-line clustering and fuzzy support vector machines for nonlinear system

Julio César Tovar*, Wen Yu, Floriberto Ortiz, Carlos Román Mariaca, José de Jesús Rubio

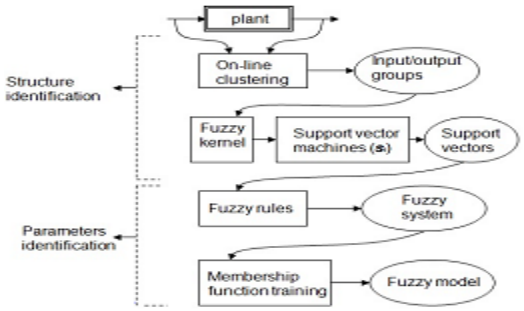
Abstract—**This paper describes a novel non-linear modeling approach by on-line clustering, fuzzy rules and fuzzy support vector machines. Structure identification is realized by on-line clustering method and support vector machines, and the rules are generated automatically. Time-varying learning rates are applied for updating the membership functions of the fuzzy rules. Finally, tue upper bounds of modeling errors are proven..**

I. Introduction

Both neural networks and fuzzy logic are universal estimators. The process of fuzzy rule extraction for nonlinear systems modeling is called structure identification. A common method is to partition the input and the output data, it is also called fuzzy grid [1]. Most of structure identification approaches are based on off-line data clustering, such as fuzzy C-means clustering [2], mountain clustering [2], and subtractive clustering [3]. These approaches require that the data is ready before the modeling. There are a few of on-line clustering methods in the literature. A combination of on-line clustering and genetic algorithm for fuzzy systems is proposed in [4]. In [5] the input space was automatically partitioned into fuzzy subsets by adaptive resonance theory. On-line clustering with a recursively calculated spatial proximity measure was given in [6]. There is one weakness for the above on-line clustering methods: the partitioning of the input (precondition) and the output (consequent) do not take into account time mark. They use all data to train each rule. In this paper, a novel on-line clustering approach is proposed. The time relationship in the input and the output spaces is considered.

Besides clustering approaches, fuzzy rule extraction can also be realized by neural networks method [1], genetic algorithms [7], SVD-QR [8] and support vector machine (SVM) technique [9]. SVM was first used for solving pattern classification problem. Vapnik defined it as the structure risk minimization which minimizes the upper bound of the modeling error. The basic idea of SVM modeling is to map the inputs into higher dimensional feature space, then solve quadratic programming (QP) with an appropriate cost function [10]. There is one important property in SVM: the solution vector is sparse. Only the non-zero solutions which are called support vectors are useful for the model. In this paper, we use the support vectors to extract the fuzzy rules in each group after the on-line clustering.

This paper is organized as follows. Firstly, we use an on-line clustering method itch divides the input and the output data into several clusters in the same temporal interval, so the structure of fuzzy systems is automatically established. Second, fuzzy SVM is applied to generate support vector in each cluster. With these support vectors, fuzzy rules are constructed. Here we use two fuzzy techniques to modify the standard SVM, the kernel is changed as fuzzy membership function and a fuzzy membership function and a fuzzy factor is added to the performance index of SVM, and the corresponding fuzzy systems are made. The fuzzy SVM provides an adaptive local representation for SVM, and this takes advantages of some properties of a fuzzy system, such as adaptive learning and economic structure. The scheme of the modeling via on-line clustering and fuzzy support vector machines proposed in this paper is shown in Figure [1].



II. On-line clustering for the input/output data

The following state-space discrete-time smooth nonlinear

Figure 1. The scheme of the modeling via on-line clustering and fuzzy support vector machines.

system with fuzzy rules lease check with your editor on whether to submit your manuscript by hard copy or electronically for review

$$x(k+1) = f[x(k), u(k)], \qquad y(k) = h[x(k)] \qquad (1)$$

where $u(k) \in R^m$ is input vector, $x(k) \in R^n$ is state vector, and $y(k) \in R^p$ is output vector. f and h are general smooth functions. (1) can be rewritten as:

Julio César Tovar is with Engineering Communications and Electronic, Automatic Control Department, Notional Institute Polytechnic, Unidad Profesional Adolfo López Mateos, Col. Lindavista, Del. Gustavo A. Madero, México, D.F. C.P. 07738, Mexico Cyty, Mexico (e-mail: jctovar@ipn.mx).

Wen Yu is with the Departamento de Control Automatico, CINVESTAV-IPN, Mexico City, Mexico

$$
\begin{aligned}
&y(k)=h[x(k)]=F_1[x(k)],\\
&y(k+1)=h[f[x(k),u(k)]]=F_2[x(k),u(k)]\\
&y(k+n\text{-}1)=F_n[x(k),u(k),\cdots u(k+n\text{-}2)]
\end{aligned}
\tag{2}
$$

denoting

$$
\begin{aligned}
&Y(k)=[y(k),y(k+1),\cdots y(k+n-1)]^T\\
&U(k)=[u(k),u(k+1),\cdots u(k+n-2)]^T\\
&y(k)=F[x(k),U(k)],F=[F_1\cdots F_n]^T
\end{aligned}
$$

Since (1) is a smooth nonlinear system, (2) can be expressed as *x(k+1)=g[Y(k+1),U(k+1)]*. This leads to the multivariable NARMAX model

$$
y(k)=h[x(k)]=\Psi[X(k)] \tag{3}
$$

where

$$
X(k)=[y(k-1),y(k-2),\cdots u(k-d),u(k-d-1),\cdots]^T \tag{4}
$$

Ψ(·) is an unknown nonlinear function representing the plan dynamics, *u(k)* and *y(k)* are measurable scalar input and output, d is time delay.

The objective of the structure identification is to partition the input and the output data *[y(k),x(k)]* of nonlinear system and extract fuzzy rules. Here $x(k)=[x_1(k),x_2(k),\cdots,x_n(k)]$, and to find many groups we need, or what is l for the following rule

$$
\begin{aligned}
&R^j: IF\ x_1(k)\ is\ A_1^j\ and\ x_2(k)\ is\ A_2^j\ and\ \cdots x_n(k)\ is\ A_n^j\\
&THEN\ y(k)\ is\ B^j, j=1\cdots l
\end{aligned}
$$

In this paper, the basic idea of the proposed on-line clustering scheme [11], is that the input and the output space partitioning are carried out in the same time index. If the distance from a point to the center is less than required length, the point is in this group. When a new data comes, the center and the group should be change according to the new data. We give the following algorithm. The Euclidean distance at time *K* is defined as

$$
\begin{aligned}
&d_{k,x}=\left(\sum_{i=1}^{n}\left[\frac{x_i(k)-\bar{x}_i^j}{x_{i,max}-x_{i,min}}\right]^2\right)^{1/2}\\
&d_{k,y}=\left|\frac{y(k)-\bar{y}^j}{y_{max}-y_{min}}\right|\\
&d_k=\alpha d_{k,x}+\beta d_{k,y}
\end{aligned}
\tag{5}
$$

where $x_{i,max}=max_k\{x_i(k)\}$, $x_{i,min}=min\{x_i(k)\}$, $y_{max}=max\{y(k)\}$, $y_{min}=min\{y(k)\}$, $x_i(k)$ and $y(k)$ are the centers of x_i and y at time k, α and β are positive factors, normally we can choose $\alpha=\beta=(1/2)$. For the *Group j*, the centers are updating as follows

$$
\begin{aligned}
&\bar{x}_i^j=\frac{1}{l_2^j-l1^j+1}\sum_{l=l_1^j}^{l_2^j}x_i(l)\\
&\bar{y}^j=\frac{1}{l_2^j-l1^j+1}\sum_{l=l_1^j}^{l_2^j}y(l)
\end{aligned}
\tag{6}
$$

where l_1^j the first number of the Group j is, l_2^j is the last number of the *Group j*. The length of *Group j* is $m^j=l_2^j-l_1^j+1$. The time interval *Group j* is $\left[l_1^j,l_2^j\right]$. The process of the structure identification can be formed as the following steps,

1. For the first data G_1, k=1. y(1), $x_i(1)$ are the centers of the first group, $\bar{x}_i^1=x(1)$, $\bar{y}^j=y(1)$, $l_1^j=l_2^j=1$.

2. If a new data *[y(t),x_i(t)]* comes, $l_2^j=l_2^j+1$, we use (5) and (6) to calculate d_k. If no any new data comes, go to 5.

3. If $d_k \le L$ then *[y(t),x_i(t)]* is in a new group G_j, go to 2.

4.If $d_k>L$ then *[y(t),x_i(t)]* is in a new group *j=j+1*, the center of the *Gj* is $\bar{x}_i^j=x(k)$, $\bar{y}^j=y(k)$, $l_1^j=l_2^j=k$, go to 2.

5.Check the distances between all centers $\bar{x}^j$, $\bar{y}^j$, If $\sum_{i=1}^{n}\left[\bar{x}_i^p-\bar{x}_i^q\right]^2+\left|\bar{y}^p-\bar{y}^q\right|\le L$, the two groups G_p and G_q are combine into one group.

The new idea of the on-line clustering of this paper is that the input-output spaces partitioning is carried out in the same temporal interval. There are two reasons. First, nonlinear system modeling is to find a suitable mapping between the input and the output. If we use fuzzy neural networks as models, the rules have the form as *"IF input is A THEN output is B"*. Only when the input and the output occur in the same time interval, they correspond to the nonlinear mapping. So our on-line clustering considers time factor. Second, we will propose an on-line modeling approach based on the on-line clustering. When a new group (or a new rule) is created, we can use the data in the corresponding time interval to train the rule. So clustering with temporal intervals will simplify parameter identification and make the on-line modeling easier.

There are three design parameters *α*, *β* and *L*. *α* and *β* can be regarded as the weights on the input and the output spaces. If the input dominates the dynamic property, we should increase *α* and decrease *β*. Usually we select *α=β=0.5* such that the input and the output are the same important. If we let *α=1*, *β=0*, it becomes the normal on-line clustering. L is the threshold of creating new rules, it is the

lowest possible value of similarity required to join two objects in one clusters. How to choose the user -defined threshold is a trade-off. If the threshold value L is too small, there will still be many groups present at the end, and many of them will be singletons. Conversely, if the threshold L is too large, many objects that are not very similar may end up in the same cluster. Since $d_k=\alpha d_{k,x}+\beta d_{k,y}$, $d_{max}=\alpha||x_{max}-x_{min}||+||y_{max}-y_{min}||\beta$. If we want the algorithm can partition several groups, we should let $L<d_{max}$, otherwise there is only one group.

III. Fuzzy Rules Extraction by Fuzzy Support Vector Machines

A SVM can separate the data with a hyper plane in maximum margin between two classes [12]. A possible formalization of this task is to design an estimating function $f:\mathbb{R}^n\rightarrow\{-1,+1\}$ use an independent generated couple of data according to an unknown distribution $P(X,y),(X_m,y_m)\in\mathbb{R}^n\times y$, $y=\{-1,+1\}$. If the training is detachable for the hyper plane, the function is chosen as $f(x)=(w\cdot x)+b$. The margin is defined as the minimum distance from a sample to the surface of resolution. The margin in turn we can measure for the longitude from vector w, such that the near points to the hyper plane satisfies $|(w\cdot x)+b|=1$.

In this paper we use SVM for function estimation. In order to find support vectors in the Group j, we use the input-output data $[x_k,y(k)]$, $k\in\left[l_1^j,l_2^j\right]$ to approximate a nonlinear function. Consider regression in the set of non-linear functions regression $y=f(x)$ can be estimated as the following form.

$$h(x_k)=w^T\varphi(x_k)+b$$

where w is the weight vector, $\phi(x_k)$ is a known non-linear function, b is a threshold, x_k is the input vector a time k, Kernel trick is defined as $K(x,x_k)=\phi(x_k)^T\phi(x_k)$, x is the input vector at any time.

A Kernel fuzzy

There are many possible choices for the kernel K(x,xk) we only require $K(x,x_k)$ satisfies the Mercer condition <cite>Cristianini</cite>. For example the linear kernel $K(x,x_k)=x_k^Tx$, the MLP kernel $K(x,x_k)=tanh(k_1x_k^Tx+k_2)$, k_1 and k_2 are constants. RBF kernel $K(x,x_k)=exp(-||x-x_k||^2/\sigma^2)$. In this paper, we use the fuzzy kernel, which is defined as

$$K(x,x_k)=\begin{cases}\prod_{i=1}^{M}u_i(x_k)\square u_i(x_i) & x_k \text{and } x \text{ are both in the } jth \text{ cluster}\\ 0 & \text{otherwise}\end{cases}$$

where M is total time, $u_i(x_k)$ is the membership function.

Let the training set be

$$S=\{(x_1,y(1)),(x_2,y(2)),\ldots,(x_v,y(M))\}$$

Assume the training samples are partitioned into l clusters. We can group the training samples into l clusters as follows:

$$\begin{aligned}&\text{cluster } 1 =\{(x_{k_1}^1,y_{k_1}^1),\ldots,(x_{k_1}^1,y_{k_1}^1)\}\\&\text{cluster } 2 =\{(x_1^2,y_1^2),\ldots,(x_{k_2}^2,y_{k2}^2)\}\\&\vdots\\&\text{cluster } l =\{(x_1^l,y_1^l),\ldots,(x_{k_l}^l,y_{k_l}^l)\}\end{aligned}\tag{7}$$

where k_g, $g=1,2,\ldots,l$ is the number of points belonging to the gth cluster, so that we have $\sum_{g=1}^{l}k_g=v$.

How to choose the membership function $u_i(x_k)$ is another problem. Gaussian function and triangle function are the most popular functions for the membership function of fuzzy systems. When $u_i(x_k)$ is Gaussian function, the kernel function is

$$K(x_k,x_j)=\varphi(x_k)\square\varphi(x_j)=\exp\left(\frac{\|x_k-x_j\|}{2\sigma^2}\right)\tag{8}$$

and the fuzzy kernel is

$$K_g=\begin{bmatrix}K(x_1^g,x_1^g) & K(x_1^g,x_2^g) & \cdots & K(x_1^g,x_g^g)\\ K(x_1^g,x_1^g) & K(x_2^g,x_2^g) & \ddots & \vdots\\ \vdots & \ddots & \ddots & K(x_{k_g-1}^g,x_{k_g}^g)\\ K(x_1^g,x_1^g) & \cdots & K(x_{k_g}^g,x_{k_g-1}^g) & K(x_{k_g}^g,x_{k_g}^g)\end{bmatrix}\in\square^{k_g\times k_g}\tag{9}$$

B Fuzzy support vector machines

In this paper, we introduce a fuzzy factor is to the above performance index [13]. Here $\sigma\le s_i\le 1$ where σ is a sufficient small positive number, s_i denotes the important degree of sample xi for learning the optimal hyper plane in SVM. We select si as bell shape function (7), see Figure 2

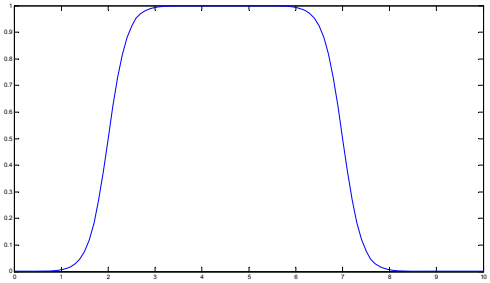


Figure 2. Generalized bell shape built-in membership function

$$s_i=f(t)=\frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}}\tag{10}$$

The construction of the optimization problem is formulated as FSVM

$$\begin{aligned}&\min\Phi(w,\xi,\mu)=\frac{1}{2}w^Tw+C\sum_{i=1}^{n}s_i^m\xi_i\\&sujeto\ \ y_i\left(w^Tx_i+b\right)\ge 1-\xi_i,\ i=1,2,\ldots,n\\&\xi_i\ge 0,\ i=1,2,\ldots,n\end{aligned}\tag{11}$$

where m defuzzyfied influence of the lack of the term defuzzufied in the cost function. Now the Lagrangian is

$$Q(w,b,\xi,\alpha,\beta,\mu)=\frac{1}{2}w^T w+C\sum_{i=1}^{n}s_i^m\xi_i \\ -\sum_{i=1}^{n}\alpha_i\left[y_i\left(w^T x_i+b\right)-1+\xi_i\right]-\sum_{i=1}^{n}\beta\xi_i \tag{12}$$

where α_i and β_i are nonnegative Lagrange multipliers. Differentiating Q with respect to w, β and ξi, and equating the results to zero yields the following three optimality conditions

$$\frac{\partial Q(w,b,\xi,\alpha,\beta,\mu)}{\partial w}=w-\sum_{i=1}^{n}\alpha_i y_i x_i=0 \\ \frac{\partial Q(w,b,\xi,\alpha,\beta,\mu)}{\partial b}=\sum_{i=1}^{n}\alpha_i y_i=0 \\ \frac{\partial Q(w,b,\xi,\alpha,\beta,\mu)}{\partial \xi}=C\mu_i^m-\alpha_i-\beta_i=0 \tag{13}$$

Each group has a diffuse pattern. Using the idea Takagi-Sugeno model [8], we can combine the local models in each group within a global model. The fuzzy rules have the form

$$R^j: IF\ x_{l_1^j}\le x_1\le x_{l_2^j}\ and\ \dots\ x_{l_1^j}\le x_n\le x_{l_2^j} \\ THEN\ y(k)=f_i\left[X(k)\right]$$

where $j = 1 \cdots p$, p is the number of cluster group online. Membership functions for x_i. The final fuzzy model

$$\hat{y}=\frac{\sum_{i=1}^{p}f_i\left[X(k)\right]\left[\prod_{j=1}^{n}\left(\mu_{A_i^j}\right)\right]}{\sum_{i=1}^{p}\left[\prod_{j=1}^{n}\left(\mu_{A_i^j}\right)\right]} \tag{14}$$

C.Membership functions training

After the fuzzy system is obtained by support vector machine, we use the input-output data [y(k),x(k)], k $\in \left[l_1^j, l_2^j\right]$ to train the membership functions Aij (i=1…n) and Bj, i.e., the parameter identification of the membership functions are performed in the corresponding time interval found in the structure identification. We discuss two cases: consequence membership functions training and premise membership functions training.

First we assume the premise membership functions $A_{1i}...A_{ni}$ are given by prior knowledge, i.e.,

$$\varphi_i=\prod_{i=1}^{n}\mu_{A_i^j}\Big/\sum_{j=1}^{sv_j}\left[\prod_{i=1}^{n}\mu_{A_i^j}\right]$$

is known [2] and [15].

In this paper we are only interested in open-loop identification, we assume that the plant (1) is bounde-input and bounded-output (BIBO) stable, i.e., *y(k)* and *u(k)* en (1) are bounded . The following theorem gives a stable gradient descent algorithm for fuzzy neural modeling.

Theorem 1. *If we use the fuzzy system (14) to identify nonlinear plant (1) in group j, the following gradient descent algorithm with a time-varying learning rate can make identification error e(k) bounded*

$$W(k+1)=W(k)-\eta_k e(k)\Phi^T\left[X(k)\right] \tag{15}$$

where the scalar ηk=(η/(1+||Φ[x(k)]||²)), 0<η≤1. The normalized identification error

$$e_N(k)=\frac{e(k)}{1+\max_k\left(\left\|\Phi\left[X(k)\right]\right\|^2\right)} \tag{16}$$

satisfies the following average performance

$$\limsup_{T\to\infty}\frac{1}{T}\sum_{k=1}^{T}\left\|e_N(k)\right\|^2\le\bar{\mu} \tag{17}$$

where $\bar{\mu}$ =maxk}[||μ||²].

Remark 1. *Generally, a fuzzy system cannot match a nonlinear system exactly. The parameters of the fuzzy system will not converge to its optimal values. The online identification proposed in this paper is to force the output of the fuzzy system to follow the output of the plant. Although the parameters cannot converge to their optimal values, (17) shows that the normalized identification error will converge to a ball radius* $\bar{\mu}$. *If the fuzzy system (18) can match the non-linear plant (1) exactly (μ(k)=0), i.e., we can find the best membership function* $\mu_{A_i^j}$ *and* W^* *such that the nonlinear system can be written as* $y(k)=W^*\Phi\left[\mu_{A_{ji}}\right]$.

Since $||e(k)||^2>0$, *the same learning law makes the*

$$\lim\left\|e(k)\right\|=0 \tag{18}$$

Remark 2. *The normalizing learning rate η in (15) is time varying to assure the stability of identification error. These learning rates are easier to be chosen than [17] without requiring any prior information, for example we may select*

η=1. Time varying learning rates can be found in some standard adaptive schemes [18]. But they need robust modifications to guarantee stability of the identification (15) is similar to the results of [19]. In this paper the algorithm is derived from stability analysis (or ISSLyapunov function), the algorithm of [19] was obtained from minimization of the cost function. We focus on the bound of identification error, [19] focused on convergence analysis. It is interested to see that the two different methods can get similar results.

Now we discuss the case of training both consequence and premise membership functions. The initial conditions are $c_{ji}(1) = x_i^*$, $w_j(1) = y_i^*$, $\sigma_{ji}(1)$ *is random in (0,1). Since the membership functions are Gaussian functions, the output of the fuzzy system can be expressed as*

$$\hat{y} = \frac{\sum_{i=1}^{sv_j} w_i \left[\prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}\right)^2}{\sigma_{ji}^2} \right) \right]}{\sum_{i=1}^{sv_j} \left[\prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}\right)^2}{\sigma_{ji}^2} \right) \right]} \tag{19}$$

Theorem 2. *If we use Mamdani-type fuzzy neural network (19) to identify nonlinear plant (1), the following backpropagation algorithm makes identification error e(k) bounded*

$$\hat{y} = \frac{\sum_{i=1}^{sv_j} \left(\sum_{k=0}^{n} p_k^i x_k \right) \left[\prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^{*2}} \right) \right]}{\sum_{i=1}^{sv_j} \left[\prod_{j=1}^{n} \exp\left(-\frac{\left(x_j - c_{ji}^*\right)^2}{\sigma_{ji}^{*2}} \right) \right]} \tag{20}$$

where c_{ji}^* *and* σ_{ji}^{*2} *are unknown parameters which minimizes μ unmodelled dynamics, and x0= 1.*

IV. SIMULATION

The engine operation at idle is a nonlinear process that is far from its optimal range. Because it does not require any large degree of instrumentation or external sensing capabilities, the idle speed control is also accessible and can be formulated as a benchmark problem in control society. The process of engine at idle has time delays that vary inversely with engine speed and is time-varying due to aging of components and environmental changes such as engine warm-up after a cold start. The measurement of system outputs occurs asynchronously with the calculation of control signals. We assume that the occurrence of plant disturbances, such as engagement of air conditioner compressor, shift from neutral to drive in automatic transmissions, application and release of electric loads, and power steering lock-up, are not directly measured. The dynamic engine model is a two-input and two-output system [20]

$$\begin{aligned}
&\dot{P} = k_p\left(\dot{m}_{ai} - \dot{m}_{ao}\right), \quad \dot{N}_p = k_n\left(T_i - T_L\right) \\
&\dot{m}_{ai} = \left(1 + k_{m1}\theta + k_{m1}\theta^2\right) g\left(P\right) \\
&\dot{m}_{ai} = -k_{m3}N_p - k_{m4}P + k_{m5}N_pP - k_{m6}N_pP^2
\end{aligned} \tag{21}$$

the parameters for an engine model with 1.6 liter, 4-cylinder fuel injected are:

$$g(P) = \begin{cases} 1 & P \geq 50.6625 \\ 0.097\sqrt{101.325P - P^2} & P \geq 50.6625 \end{cases}$$

$$T_L = \left(\frac{N}{263.17}\right)^2 + T_d,$$

$$m_{ao} = \dot{m}_{ao} \Big/ \left(120N\right),$$

$$T_i = -39.22 + 325024m_{ao} - 0.0112\delta^2 + 0.635\delta$$

$$+\frac{2\pi}{60}\left(0.0216 + 0.000675\delta\right)N_p - \left(\frac{2\pi}{60}\right)^2 0.000102N^2,$$

$$k_p = 42.40,\ k_N = 54.26,\ k_{m1} = 0.0907,$$

$$k_{m2} = 0.0998,\ k_{m3} = 0.0005968,\ k_{m4} = 0.0005341,$$

$$k_{m5} = 0.000001757,\ \tau = 45/\mathrm{N}_p.$$

The system outputs are manifold press P (kPa) and engine speed N_p (rpm). The control inputs are throttle angle θ (degree) and the spark advance δ (degree). Disturbances act to the engine in the form of unmeasured accessory torque T_d (N-m). The variable $\dot{m}_{ai}$ and $\dot{m}_{ao}$ refer to the mass air flow into and out of the manifold. m_{ao} is the air mass in the cylinder. The parameter τ is a dynamic transport time delay. The function $g(P)$ is a manifold pressure influence function. T_i is the engine's internally developed torque, T_L is the load torque.

Difference technique is used to get the discrete-time states of the system (21). (21) can be written as $x=f(x,u)$ with $x=[P,N_p]_T$, $u=[\theta,\delta,T_d]_T$. Let us define $s_1=(f(x_k,u_k)$, $s_2=f(x_k+s_1,u_k)$, $s_3=f(x_k+((s_1+s_2)/4),u_k)$. If $|((s_1-2*s_3+s_2)/3| \leq ((|x_k|)/(1000))$ or $|((s_1-2*s_3+s_2)/3)|<1$, then $x_{k+1}=x_k+(((s_1+4s_3+s_2))/6)$, $k=0,1,2,\ldots$. The discrete-time model for P is $P(k+1) = f_1[(P)\ k, N_P(k), \theta(k), \delta(k), T_d(k)]$.

In order to train the neural model, we select inputs as $\delta=30\sin(0.06k)$, θ is a square wave with amplitude 20, $\theta(k)=20\mathrm{Square}(0.04k)$, T_d is a constant, $T_d(k)=10$, $x_0=[10,500]^T$. The time interval the hidden layer selection N is chosen as $N=300$.

We use the following Mamdani fuzzy model, for *jth* rule. All of data are normalized such that are within the interval of *[-1,1]*. We start from $W(k) \in \Re^{1\times 15}, V(k) \in \Re^{15\times 5}$.

$$R^j : IF\ \left(P\ is\ A_p^j\right)\ and\ \left(N_p\ is\ A_N^j\right)\ and\ \left(\theta\ is\ A_\theta^j\right)\ and\ \left(\delta\ is\ A_\delta^j\right)$$
$$and\ \left(T_p\ is\ A_T^j\right)\ THEN\ P(k+1)\ is\ B_p^j$$

After $k=200$, we use support vector machine and obtain 4 support vectors. These support vectors are the initial elements of W and V.

After $k>500$, the plant is changed as $g(P)=0.8$ with $P<50.6625$. The hysteresis constant is selected $h=0.05$, the threshold of weig change is $L=1.5$. We repeat above procedure, $\tau_1=500$.

After $k=700$, $\tau_2=700$, $N=200$. 6 support vectors are obtained, so the number of the fuzzy rule is 6. The final training result is shown in Figure 3(a). The testing signals are changed as $\delta=10$, $\theta(k)=40\cos(0.05k)$, $T_d(k)=1$. The testing result is shown in Figure 3(b).

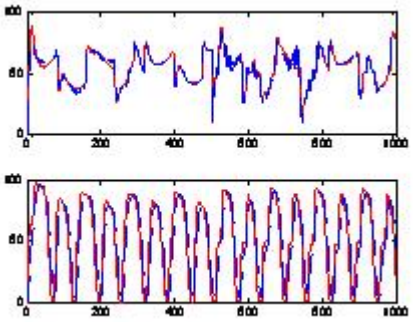


Figure 3. a)Testing signal, b) testing result

V. CONCLUSION

In this paper, we propose an efficient approach for nonlinear system modeling using fuzzy rules. Several techniques are combined for the new approach. First we propose an online clustering method which divides the input/output data into several clusters in a same temporal interval. Then we apply fuzzy support vector machines which generate support vectors in each cluster. With these support vectors, fuzzy rules are constructed and the corresponded fuzzy system is made. After the structure identification, a time-varying learning rate is applied for the parameters identification. The contributions of the paper are:

1. online clustering method and support vectors machines are used for the fuzzy rules extraction.
2. the upper bound of modeling error and stability are proved for the fuzzy modeling.

REFERENCES

[1] J. S. Jang, ANFIS: Adaptive-network-based fuzzy inference system, IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, 665--685, 1993.

[2] S.Mitra and Y.Hayashi, Neuro--fuzzy rule generation: survey in soft computing framework, IEEE Transactions on Neural Networks, Vol.11, No.3, 748-769, 2000.

[3] S.L.Chiu, Fuzzy Model Identification based on cluster estimation, Journal of Intelligent and Fuzzy Systems, Vol.2, No.3, 1994.

[4] C.F.Juang, Combination of on-line clustering and Q-value based GA for reinforcement fuzzy system design, IEEE Transactions on Fuzzy Systems, Vol.13, No.3, 289- 302, 2005.

[5] S.G.Tzafestas and K.C.Zikidis, NeuroFAST: On-line neuro-fuzzy ART-based structure and parameter learning TSK model, IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol.31, No.5, 797-803, 2001.

[6] J. Wang, "Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication)," *IEEE J. Quantum Electron.*, submitted for publication.

[7] I.Rivals and L.Personnaz, Neural-network construction and selection in nonlinear modeling, IEEE Transactions on Neural Networks, Vol.14, No.4, 804-820, 2003.

[8] J.-H.Chiang, P.-Y.Hao, Suuport Vector Learning Mechanism for Fuzzy Rule-Based Modeling: A New Approach, IEEE Transactions on Fuzzy Systems, Vol. 12, No. 1, February 2004.

[9] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines: Cambridge Univ. Press, 2000.

[10] K.-R.Mueller, S.Mika, G.Rasch, K.Tsuda, B.Scholkopf. An Introduction to Kernel-Based Learning Algorithm. IEEE Transactions on Neural Networks, Vol. 12, No. 2, March 2001.

[11] Julio César Tovar and Wen Yu, Fuzzy neural modeling via clustering and support vector machines, 16th IEEE International Conference on Control Applications, Singapore, 1-3 October 2007.

[12] B.E. Boser, I.M. Guyon, and V.N. Vapnik, An training algorithm for optimal margin classifier, in Proc. 5th ACM Workshop Computational Learning Theory, Pittsburg, PA, pp 144-152, 1993.

[13] C-F. and Wang, S-D. (2002) Fuzzy support vector machines, IEEE Transactions on Neural Networks, March, Vol. 13, No. 2, pp.464--471.

[14] E.H. Mamdani, Application of fuzzy algorithms for control of simple dynamic plant, IEEE Proceedings-Control Theory and Applications, vol. 121, no. 12, 1585-1588, 1976.

[15] L.X.Wang, Adaptive Fuzzy Systems and Control, Englewood Cliffs NJ: Prentice-Hall, 1994.

[16] T. Takagi and M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. Syst., Man. and Cybern., vol. 1, pp. 116-132, Jan. 1985.

[17] J.S. (1993) ANFIS: adaptive network based fuzzy inference system, IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, pp.665--685.

[18] Robust Adaptive Control, Prentice-Hall, Inc, Upper Saddle River, NJ,P.A. and Sun, J. (1996).

[19] *D.P., Hanna, A.I. and Razaz, M. (2001) A normalized gradient descent algorithm for nonlinear adaptive filters using a gradient adaptive step size, IEEE Signal Processing Letters, Vol. 8, No. 11, pp.295--297.*

[20] *G. V. Puskorius and L. A. Feldkamp, Neurocontrol of nonlinear dynamics systems with kalman filter trained recurrent networks, IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 279--297, 1994.*

[21] K.S.Narendra and X.Cheng, Adaptive control of discrete-time systems using multiple models, IEEE Transactions on Automatic Control, vol. 45, no. 9, pp. 1669 --1686, Sptember 2000.

[22] B.Kikens and M. Karim, Process identification with multiple neural network models, Int. J. Control, vol. 72, pp. 576--590, 1999.

[23] K. Li, J. Peng, and E. W. Bai, A two-stage algorithm for identification of nonlinear dynamic systems, Automatica, vol. 42, no. 7, pp. 1189-1197, 2006.