A self-organizing neuro-fuzzy network based on first order effect sensitivity analysis

Cheng Chen\textsuperscript{a,}\textsuperscript{*}, Fei-Yue Wang\textsuperscript{a,b}

\textsuperscript{a} State Key Laboratory of Management and Control for Complex Systems, Chinese Academy of Sciences, Beijing 100190, PR China
\textsuperscript{b} Center for Military Computational Experiments and Parallel Systems, National University of Defense Technology, Changsha 410073, Hunan, PR China

\textbf{Article history:}
Received 17 April 2012
Received in revised form 9 January 2013
Accepted 9 February 2013
Communicated by A. Konar
Available online 14 March 2013

\textbf{Keywords:}
Neuro-fuzzy networks
First order effect
Sensitivity analysis
Systemic fluctuation

\textbf{Abstract}
As an effective method that can provide the information about the influence of inputs on the variation of output, variance based sensitivity analysis is widely used to determine the structure of neural networks. In the past, the global sensitivity analysis method for the total effect has been used for the structure learning of neural networks and various growing and pruning algorithms have been developed. In this paper, we find that neuro-fuzzy networks have the characteristics of additive models in which the first order effect index of the influence can provide the same comprehensive information as the total effect index, thus we only need to analyze the first order effects of the inputs to their output layers. Based on this observation, many low-cost effective methods for the first order effect global sensitivity can be used in for developing self-organizing neuro-fuzzy networks. Specifically, Random Balance Designs is employed here for sensitivity analysis. In addition, we also introduce the concept of systemic fluctuation of neuro-fuzzy networks to determine whether adjustment is needed for a network. This concept helps us to build a new procedure about the leaning of self-organizing neuro-fuzzy networks and to accelerate its speed of convergence in learning and organizing. Examples of simulations have demonstrated that our proposed method performs better than other existing procedures for self-organizing neuro-fuzzy networks, especially in learning of the network structure.

\section{Introduction}
Fuzzy control system has been applied in area of machine control widely and successfully. This kind of system is based on fuzzy logic, which always has a certain number of fuzzy rules. Hence, fuzzy control system can be understood by human operators well. In order to produce reasonable fuzzy rules, knowledge of experts is used as prior knowledge to determine both the scale of fuzzy rules and the parameters of each fuzzy rule. However, if the actual system cannot be covered by the knowledge of experts precisely, the production of fuzzy rules needs other approaches, like fuzzy neural network.

Neuro-fuzzy network is a hybrid method, which interprets the fuzzy control system into a special kind of neural network with learning capability \cite{1-3}. In order to learn reasonable parameters of each fuzzy rule, neural network can use supervised or unsupervised learning algorithms to find them \cite{4-6}. However, the learning about the scale of fuzzy rules needs the help of self-organizing neural network by growing and pruning of nodes online \cite{7}. Hence, the learning of neuro-fuzzy networks can be seen as a hierarchical learning \cite{8}, including coarse-grained learning and fine-grained learning. The fine-grained learning is learning about the parameters of network, including the parameters of the neurons and the weights in the network. The coarse-grained learning is the learning about the structure of network. Since the selection of the appropriate number of rules is difficult for designers and experts in technical domain, structure learning is one hot topic of neuro-fuzzy networks.

The self-organizing structure learning of neuro-fuzzy networks, which is also called the growing and pruning of network, can be grouped into two categories, i.e., clustering approach and sensitivity analysis.

\subsection{1.1. Clustering approach}
According to a clustering criterion, clustering approach determines the number of rules by partitioning of input-output space. In \cite{9}, clustering criterion is the firing strength with each incoming pattern to be the rule neurons. If the max firing strength of one pattern cannot exceed the critical value, SONFIN take this pattern to produce a new fuzzy rule. Similarly, a viewpoint that fuzzy system should have at least one fuzzy rule to ensure the match degree (or firing strength) is no less than a threshold, which called \(\epsilon\)-completeness, is used in \cite{10,11}. Hence, GD-FNN...
determines whether to add new rule neuron by the firing strength of coming pattern. In [8,12,13], the distance between incoming pattern and the centers of existing rule neurons is clustering criterion. When this distance exceeds a threshold, a new rule neuron is added. All of these different criteria are similar to the distance criterion of clustering. When the criterion is unsatisfied with the coming of a new pattern, a new cluster is built. These clustering approaches are mainly used for the growth of neural network. However, they can also be used for pruning, i.e., the pruning process can merge the closest neurons.

1.2. Sensitivity analysis

Sensitivity analysis techniques quantify the relevance of the component of network, which can be input neurons, hidden neurons or weights, as the influence that parameter perturbations have on a performance function [14]. The performance function can be the functional error of system [15–17] or the relevance of parameters or weights, as the influence that parameter perturbations have on the system as the criterion for structure learning will only pursue a low approximation error and miss some of the other available information of network, like the contribution of each hidden neuron to output. The generalization error of the system may be not ideal. Since structure selection of neuro-fuzzy networks is the coarse-grained learning, the variance based sensitivity analysis, which takes the influence of rule on the variance of output, can choose one more reasonable network and indeed provides us a new way. It has been firstly successfully used for the learning of neural network in [14]. The method of sensitivity analysis is based on a first-order Taylor expansion of the output function in the neural network, which is one local method of sensitivity analysis [20]. In [19], the authors use Extended Fourier Amplitude Sensitivity Test (EFAST) [21], which is one of the quickest means in global method of sensitivity analysis for total effect, to accomplish the work about variance based sensitivity analysis for the neural network with single hidden layer and prove its effectiveness. In the work of [18], EFAST is used in structure learning of self-organized neuro-fuzzy networks, and the theoretical convergence of fine-grained learning is given.

For the advantages of variance-based sensitivity analysis, we use it to determine the structure of our self-organizing neuro-fuzzy networks and optimize this approach in two ways. One way is using global sensitivity analysis for first order effect to simplify the process of sensitivity analysis based on the global sensitivity analysis for total effect. The other way is to rebuild a new learning produce to improve the overall performance of our self-organized neuro-fuzzy networks. We introduce the concept about the systemic fluctuation of neuro-fuzzy networks to determine whether needs to do sensitivity analysis of network and make a modification about the structure of neuro-fuzzy networks. Both two ways guarantee that we can find a reasonable structure of neuro-fuzzy networks quicker than the other variance-based sensitivity analysis methods. After that the advantages of parameter learning of other self-organizing neural networks are also used to realize our self-organizing neuro-fuzzy networks based on first order effect sensitivity analysis (NFN-FOESA). Hence, this paper is organized as follows. At first, we introduce the concept of Variance-based Sensitivity analysis in Section 2. Then, we find that the mathematical model of self-organizing neuro-fuzzy networks is one kind of additive model in Section 3. Hence, the sensitivity analysis of self-organizing neuro-fuzzy networks can be based on global sensitivity analysis for first order effect. After that the Random Balance Designs (RBD) approach [22] and its advantages are given in Section 4, which is used in our sensitivity analysis of neuro-fuzzy networks. In Section 5, we detail the whole learning procedure of NFN-FOESA and its characteristics. Comparing with other self-organizing neuro-fuzzy networks, some simulations are given to illustrate the advantages of NFN-FOESA in Section 6. Also one discussion about our method is provided. At last, the conclusion is given in Section 7.

2. Variance-based sensitivity analysis

Consider a system represented in the mathematical or computational model as follows:

\[ y = f(x_1, x_2, \ldots, x_k) \]  

where \( x_1, x_2, \ldots, x_k \) are the inputs or factors of the system and \( y \) is its output. For the purpose of sensitivity analysis, the function can be decomposed into Sobol’s high dimension model representation (HDMR) [23], i.e.

\[ f(x_1, x_2, \ldots, x_k) = f_0 + \sum_i f_i + \sum_{i<j} f_{ij} + \cdots + f_{12...k} \]

where each individual item is a function only of the factors in its index. This decomposition is not a series decomposition since it has \( 2^k \) terms.

With the assumption that inputs of the system are independent, Sobol proved that if each term in the decomposition above has zero mean, then all the terms of the decomposition are orthogonal in pairs. Therefore, these terms can be univocally calculated using the conditional expectations of model output \( y \) [24]. Thus

\[ f_0 = E(y) \]
\[ f_i = E(y|x_i) - f_0 \]
\[ f_{ij} = E(y|x_i, x_j) - E(y|x_i) - E(y|x_j) + f_0 \]

and so on.

As a consequence, in the analysis of variance based HDMR (ANOVA-HDMR), the following equation can be established:

\[ V(y) = \sum_i V_i + \sum_{i < j} V_{ij} + \cdots + V_{12...k} \]

where \( V_i = V(f_i) = V(E(y|x_i)), V_{ij} = V(f_{ij}) = V(E(y|x_i, x_j)) - V(E(y|x_i)) - V(E(y|x_j)), \) etc.

Normalizing the equation from both sides by the unconditional variance \( V(y) \), we obtain

\[ \sum_i S_i + \sum_{i < j} S_{ij} + \cdots + S_{12...k} = 1 \]

where \( S_i \) are the first-order sensitivity indexes, while \( S_{ij} \) to \( S_{12...k} \) are the high order sensitivity indexes.

For factor \( x_i \), its total sensitivity index \( S_{Ti} \) can be calculated by adding up all the sensitivity indexes involving this factor, i.e.

\[ S_{Ti} = S_i + \sum S_{ij} + \cdots + S_{12...k} \]

The total sensitivity index can provide comprehensive information about the influence of a specific factor on the variance of output \( y \), especially for a non-additive model. However, Eq. (6) implies that the computation of \( S_{Ti} \) might be quite complex in general.

As indicated by the previous works in the literature [19,18], equations listed above provided the foundation for variance-based sensitivity analysis for neural networks. In the following section of this paper, we show that for a class of neuro-fuzzy...
networks, only first order effect sensitivity analysis is required, this will significantly reduce the complexity of computation in sensitivity analysis.

3. Additive model of self-organizing fuzzy neural network

In variance-based sensitivity analysis, if the model of a system is an additive model, the first order sensitivity indexes can provide enough information to analyse the influence of factors on variation of output. Hence, the cost of sensitivity analysis of additive models can be reduced significantly. And the additive model \[25\] takes the form as follows:

\[y = \beta_0 + \sum_{i=1}^{k} f_i(x_i) + \varepsilon\]  \hspace{1cm} (7)

where \(x_i\) is the factors of model, \(k\) is the number of factors, \(y\) is the output of model, \(\beta_0\) is one constant, \(E(\varepsilon) = 0\) and \(\text{Var}(\varepsilon) = \sigma^2\). And \(\varepsilon\) can be seen as the white noise of the system.

Self-organizing neuro-fuzzy networks always contain three layers: input part, rule-produce part and output part, which are shown in Fig. 1 (a multi input single output (MISO) neural network, for example). A description of each part of neuro-fuzzy networks is given as follows:

1. **The input part**: this part is responsible for the import of the factors of system. If the number of factors is \(k\), the number of neurons in the layer of this part is \(k\). For neuro-fuzzy networks, the weights between this layer and the rule-produce part are all set to 1.

2. **The rule-produce part**: this part is the most important part in self-organizing neuro-fuzzy networks, which is responsible for the production of fuzzy rules. Hence, we denote it as the rule-produce layer in this paper. This part always contains one or more sub-layers. Among the sub-layers of the rule-produce part, the function layer is indispensable. The neurons in the function layer take all the factors as input and use one kind of function for each factor as the membership function (MF) of fuzzy rule to calculate the membership degree of this factor. Then, the membership degrees of these factors are multiplied as the output of neuron. This output is the match degree about the combination of factors to this rule. Therefore, one neuron in the function layer represents one fuzzy rule. Fig. 2 illustrates the internal structure of one neuron in the function layer. In [19], the rule-produce part only contains function layer and the MF function used in layer is hyperbolic tangent (tanh) function. The neuro-fuzzy network is shown in Fig. 3(a).

However, in [18], the rule-produce layer contains more than one layer. In addition to function layer that takes radial basis function (RBF) as MF function, there is another layer, called as normalized layer. The neuro-fuzzy network is shown in Fig. 3(b). For normalized layer is not the producer of fuzzy rules and only used for the normalization about the output of rule-produce layer, the neurons of normalized layer are mirror to the neurons of function layer. If the rule is not fit for the system, the related neurons in the two sub-layers are pruned at the same time.
3. The output part: this part takes the outputs of rule-produce part as inputs, which are match degrees about the combination of factors to these rules. The detailed process in output part is shown in Fig. 4. The process can be mathematical modeled as follows:

\[ y = \sum_{j=1}^{p} c_{oj} I_j \]  

where \( p \) is the number of output of rule-produce part, which also can be seen as the number of produced fuzzy rules in this system. \( I_j \) is one output of rule-produce layer, which also can be seen as the membership degree about the combination of factors to rule \( j \). \( c_{oj} \) is the weight between output layer and rule-produce layer.

Comparing the Eq. (8) to the Eq. (7), we find that the mathematical model of output layer is one additive model. In Section 6, the neuro-fuzzy networks can be represented as additive model in the simulations.

4. Global sensitivity analysis for first order effect

Without considering neuro-fuzzy networks as one additive model, the SA of neuro-fuzzy networks is based on global sensitivity analysis for total effect, like Extended Fourier Amplitude Sensitivity Test (EFAST) [18, 19]. By considering neuro-fuzzy networks as one additive model, in this paper, the Random Balance Designs (RBD) [22] can be used to SA of neuro-fuzzy networks.

4.1. Random Balance Designs with neuro-fuzzy networks

In RBD, all the factors are sampled using the same frequency \( \omega_r \) which avoid the time-consuming process of selection of frequency in global sensitivity analysis for total effect. The parametric equations used in EFAST are given as follows:

\[ I_i(s_j) = G_i(\sin \omega_r s_j), \quad \forall i = 1, \ldots, N \]

where \( I_i \) is the \( i \)-th input factor of model, \( G_i \) are functions to be chosen by the analyst to get the desired probability density function for \( I_i \), \( s_j \) is the parametric variable varying in \((-\pi, \pi)\) which is sampled over its range using \( N \) points. For all frequency \( \omega_r \) set to the same frequency, like 1, we find that the parametric curve of each input only covers a subset of each input space, as shown in Fig. 5(a). Hence, a random permutation is applied to the sampling of each input to generate design points. The parametric equation is changed as follows:

\[ I_i(s_j) = G_i(\sin \omega_r s_j), \quad \forall i = 1, \ldots, N \]

where \( s_1, s_2, \ldots, s_N \) denotes the \( i \)-th random permutation of the \( N \) points. Then, in order to systematically explore the input factors’ space, if the range of variation of the factor \( I_i \) is \([a, b] \), the parametric representation of the form is given by:

\[ I_i(s_j) = \frac{b + a}{2} + \frac{b - a}{\pi} \arcsin(\sin \omega_r s_j) \]

The result of sampling in the input space of two factors is shown in Fig. 5(b). The ranges of variations of two factors are \([-1, 1]\).

With the sample of size \( N \), the model is evaluated by:

\[ Y(S_j) = f(I_1(s_{j1}), I_2(s_{j2}), \ldots, I_i(s_{jN})) \]

Hence, we get the model output of size \( N \). Then, according to the value of \( I_i(s_j) \) ranked in increasing order, \( Y(S_j) \) are reordered as \( Y(S_j)^b \). The sensitivity of \( Y \) to \( I_i \) is determined by the harmonic
content of the reorganized model output, which is quantified by its Fourier spectrum:

\[ F(w) = \sum_{j=1}^{N} Y(s_j)^2 \exp(-i\omega_j) \]  

evaluated at \( \omega = 1 \) and its higher harmonics (such as \( \omega = 2, \omega = 3, \ldots , \omega = M \)). \( M \) is the highest order. The estimate of the \( V_i \) of \( L_i \) is as follows:

\[ \hat{\mathcal{V}}_i = \mathcal{V}[\mathcal{E}|-] = \sum_{\omega = 1}^{M} F(\omega) \]  

Hence, the estimation about the first order effect of \( L_i \) to \( y \) can be computed as follows:

\[ \hat{\mathcal{S}}_i = \frac{\sum_{\omega = 1}^{M} F(\omega)}{E[\mathcal{V}^{-} - \mathcal{E}[\mathcal{V}^{-}]]^2} \]  

### 4.2. Advantages of RBD over EFAST

With respect to EFAST, RBD has two advantages.

#### 4.2.1. No need for algorithms to search for frequencies aiming at free of interference

In the EFAST algorithm, designers are needed to avoid the interference between the frequency of aiming factor and the frequencies of other factors. The aiming factor is the object of the SA of the related part of sampling set. With a system of multifactors, the algorithm of searching for sound frequencies needs to be executed repeatedly to the size of multifactors. Hence, if the size of the network of FNNs is large, this is indeed a time-consuming process.

#### 4.2.2. The size of sample set is much smaller than EFAST

The size of sample set of EFAST \( N \) is defined as follows:

\[ N = k(2M\omega_0 + 1) \]

where \( k \) is the number of factors needed for sensitivity analysis, \( M \) always set from 4 to 6, \( \omega_0 \) is the highest frequency assigned. However, the size of sample set of RBD only needs to meet the following condition: \( N \geq (2M\omega_0 + 1) \). Hence, the size of sample set of RBD is not affected by the size of the network of neuro-fuzzy networks. Moreover, in RBD, the frequency needs not to avoid interference. Therefore, the minimum sample size of RBD is very small.

According to the two advantages of RBD, we find that it is indeed a low-cost sensitivity analysis tool for neuro-fuzzy networks.

### 5. Learning algorithm of NFN-FOESA

The learning process of our proposed self-organizing fuzzy neural network has two phases: structure learning and parameter learning. In the phase of structure learning, with the introduction of the criterion about systemic performance fluctuation, one growing-and-pruning algorithm based on the global sensitivity analysis for first order effect is proposed. Then, we also detailed the methods used in the phase of parameter learning.

#### 5.1. Structure learning of NFN-FOESA

Our proposed self-organizing fuzzy neural network takes RBF as the function of rule-produce layer. Taking hyperbolic tangent (tanh) function or other nonlinear functions is also feasible. We only need to develop the corresponding strategy for the adjustment of the parameters in neurons. Before modification of the structure of network, we need to construct an initial network as the start point. When the first training pattern \( (X(1), T(1)) \) enters the neuro-fuzzy network, \( X(1) \) is the first input vector, and \( T(1) \) is the first desired output, the parameters of the first RBF neuron are defined as follows:

\[ C_1 = X(1), \quad \sigma_1 = \sigma_{ini} \]  

where \( C_1 \) and \( \sigma_1 \) are the center and the width of initial neuron, \( \sigma_{ini} \) is the predefined initial width.

For structure learning, which is a coarse-grained learning, only when the performance of neuro-fuzzy networks cannot be tolerated, the learning mechanism of structure will restart. In the existing papers, this situation is the appearance of a big performance error. However, the systemic performance fluctuation should also be taken into consideration. For example, the performance error of original network is tiny, like 0.005. When a new pattern comes, the performance error increases to 0.01, which is also a tiny error. However, this pattern should be paid attention to restart the adjustment of the network. The mathematical presentation of judgment about whether to modify the structure of network is defined as follows:

\[ E(t) = \frac{E(t)-E(t-1)}{E(t) > \eta} \]

where \( E(t) \) is the error of system at time \( t \), \( \xi_1 \) and \( \eta \) are the threshold of the error and fluctuation respectively. With adding the restriction of performance fluctuation, the definition of the situation that needs to modify the structure is more reasonable than other self-organized neuro-fuzzy networks, and the threshold of performance error can be appropriate to relax. Hence, the coming of a network in one reasonable stable structure is quicker than self-organized network with a strict threshold of performance error.

When the structure of neuro-fuzzy networks needs to be adjusted, we will first add the coming pattern at time \( t \) as one new rule-produce neuron. Then the global sensitivity analysis for the old network will be processed. The first order sensitivity indexes of the outputs of rule-produce layer at time \( t \) are defined as \( S_t = [S_t], \) \( i = 1, 2, \ldots , p \). If \( S_t \) is lower than a threshold \( \lambda_{down} \), the rule \( i \) will be not regarded as an important rule and corresponding rule-produce neuron will be pruned from the network. If \( S_t \) is larger than a threshold \( \lambda_{up} \), we will use the parameter of neuron \( i \) and the coming pattern at time \( t \) to grow a new rule-produce neuron in the network. The initial parameters of the new neurons are as follows:

\[ c_{new} = X(t), \quad \sigma_{new} = \gamma(C(t)-C_{nearest}) \]

\[ C_{grow} = \frac{C_i + X(t)}{2}, \quad \sigma_{grow} = \gamma(C_{grow}-C_{nearest}) \]  

where \( X(t) \) is the coming vector at time \( t \), \( C_{new} \) and \( \sigma_{new} \) are the center and the width of the new neuron produced by \( X(t) \), \( C_{grow} \) and \( \sigma_{grow} \) are the center and the width of the new neuron produced by neuron \( i \). \( C_{nearest} \) is the center of nearest neuron to new corresponding neuron in the network in the Euclidean distance, \( \gamma \) is the initial cover rate of the network.

**Algorithm 1.** Framework of ensemble learning for our system.

**Require:**
- The training error threshold, \( \xi_1 \) and \( \xi_2 \);
- The systemic fluctuation threshold, \( \eta \);
- The influence threshold, \( \lambda_{up} \) and \( \lambda_{down} \);

**Ensure:**
1: Construct the initial network by the first coming vector \( X(1) \);
2: When new pattern \( X(t) \) is coming, calculate the output \( Y(t) \) and the performance of FNN.

If training error of network \( E(t) > \xi_1 \) or a big fluctuation \( \frac{E(t) - E(t-1)}{E(t-1)} > \eta \) happens, go to Step 3;
else, go to Step 4;
3: Add the coming vector \( \mathbf{X}(t) \) as one new rule-produce neuron, and process the global sensitivity analysis about the first order index of influence about \( I(t) \) in the network; if \( S_i > \lambda_{\text{up}} \), a new rule-produce neuron will grow by \( \mathbf{X}(t) \) and \( C_i \); if \( S_i < \lambda_{\text{down}} \), the \( i \)-th rule-produce neuron will be prune from the network.

Then, go to Step 4;

4: If raining error \( E(t) > \varsigma_2 \), widths and centers-learning is processed, and then use LLS algorithm to learning the weight.

If raining error \( E(t) < \varsigma_2 \), only use the RLS algorithm to learning the weight.

Then, go to Step 2;

5.2. Parameter learning of NFN-FOESA

With an adjusted structure of network, parameter learning of self-organizing fuzzy neural network also has two phases. One phase is the parameter learning of the antecedents in rules, which is also called the phase of widths-and centers-learning. In the other phase, which is called the phase of weight-learning, parameters of the consequents in rules are learned. After the number of rules in the system is determined, when the approximation error of system is bigger than a threshold, both two phases are processed. We use the Backpropagation algorithm in [14] to optimize the widths and centers of MFs in antecedents, which are the centers and widths of the rule-produce neurons in rule-produce layer. This method used is defined as follows:

\[
C_{ij}(t+1) = C_{ij}(t+1) + \kappa_c \Delta C_{ij}(t)
\]

\[
\sigma_{ij}(t+1) = \sigma_{ij}(t+1) + \kappa_s \Delta \sigma_{ij}(t)
\]

(21)

where \( C_{ij}(t+1) \) and \( \sigma_{ij}(t+1) \) are the center and width of the \( i \)-th MF of the \( j \)-th RBF neuron at time \( t+1 \), respectively, \( \kappa_c \) and \( \kappa_s \) are the learning rates of center and width, respectively. In [14], \( \Delta C_{ij}(t) \) and \( \Delta \sigma_{ij}(t) \) are calculated as follows:

\[
\Delta C_{ij}(t) = \frac{\partial E(t)}{\partial C_{ij}}
\]

\[
\Delta \sigma_{ij}(t) = \frac{\partial E(t)}{\partial \sigma_{ij}}
\]

(22)

The phase of weight-learning in this situation of larger approximation error, linear least-squares (LLS) method [8] is used. Our FNNs can be mathematically modeled as follows:

\[
D(t) = Y(t) + E(t) = W(t)H(t) + E(t)
\]

(23)

where

\[
H(t) = [I^T(1), \ldots, I^T(T)] \in \mathbb{R}^{1 \times p}
\]

\[
l(t) = [l_1(t), \ldots, l_T(t)] \in \mathbb{R}^{1 \times p}
\]

\[
W(t) = [w_{11}(t), \ldots, w_{1p}(t)] \in \mathbb{R}^{1 \times p}
\]

\[
E(t) = [e(1), \ldots, e(T)] \in \mathbb{R}^{1 \times p}
\]

\[
D(t) = [d(1), \ldots, d(T)] \in \mathbb{R}^{1 \times p}
\]

\[
Y(t) = [y(1), \ldots, y(T)] \in \mathbb{R}^{1 \times p}
\]

(24)

For minimizing the error energy \( E^2(t)E(t) \), an optimal of weight vector is obtained as follows:

\[
w(t) = T(t)I(t)^{-1} = T(t)(I(t)^T(t)I(t))^{-1}I(t)
\]

(25)

When the approximation error of system is lower than a threshold \( \varsigma_2 \), in order to avoid problem of overfitting, only the phase of weight-learning is processed. Without widths-and centers-learning, \( H(t-1) \) will not change and only \( I(t) \) need to be calculated. Hence, the recursive least-squares (RLS) algorithm [26] is applied to accelerate the process of weight-learning. The algorithm is given as follows:

\[
M(t-1) = (H(t-1)^T(t)H(t-1))^{-1}
\]

\[
M(t) = M(t-1) + \frac{M(t-1)l(t)^T(t)M(t-1)}{1 + l(t)^T(t)M(t-1)l(t)}
\]

\[
W(t) = W(t-1) + M(t)l(t)(d(t) - I(t)^{-1}l(t)w(t-1))
\]

(26)

5.3. Discussion of NFN-FOESA

The main steps of NFN-FOESA are given in Algorithm 1. The convergence of this method is guaranteed by the phase of parameter learning. For using RLS to realize the weight learning, our self-organizing learning algorithm is theoretically convergent, which has been given in [18]. Therefore, with a good capacity about searching reasonable rules, the method will have a faster convergence. The following advantages of this method should be pointed out:

1. As one variance-based grown-prune method, the changes of structure occur when the neuro-fuzzy network is unable to handle information, not when the performance of network reaches a minimum or overfitting begins. The concept of systemic fluctuation helps us to define the situation that needs a modification of structure of network precisely. Hence, we can find the appropriate rules quickly.

2. The sensitivity analysis of Step 3 is simpler than other variance-based growing and pruning methods.

3. Widths-and centers-learning is only selectively processed. This will help us to avoid overfitting and accelerate the parameter learning of self-organizing fuzzy neural network.

All these advantages will bring a self-organizing fuzzy neural network with faster speed and faster convergence. These characters will be shown in the following simulations.

6. Simulations

Three examples are discussed in this paper to demonstrate the effectiveness of the proposed algorithm. They include nonlinear system identification, the Mackey–Glass chaotic time-series prediction problem and traffic flow forecasting problem. The results of the first two problems are compared with GP-FNN [18], DFN [8], FAOSPFNN [13], and GD-FNN [10]. The results of traffic flow forecasting problem, which is a real world problem, are compared with the Bayesian approach [27] and the ELM approach [28]. Before the presentation and discussion about these three examples, we briefly introduce the evaluation criteria, which include predictive precision and convergence. The predictive precisions of different approaches are measured by using root-mean-square error (RMSE) and mean arithmetic error (MAE), which are defined as follows:

\[
\text{RMSE}(t) = \left[ \frac{1}{T-t_0} \sum_{t=t_0}^{T} (y(t) - \hat{y}(t))^2 \right]^{1/2}
\]

(27)

where \( \text{RMSE}(t) \) is the RMSE of online self-organizing FNN at time \( t \), \( t_0 \) is the start time of the system. The \( y(t) \) and \( \hat{y}(t) \) are the actual output and desired output of FNN at time \( t \)

\[
\text{MAE}(t) = \frac{1}{T-t_0} \sum_{t=t_0}^{T} |y(t) - \hat{y}(t)|
\]

(28)

where \( \text{MAE}(t) \) is the MAE of online self-organizing FNN at time \( t \).

The convergence of self-organized neuro-fuzzy networks has two levels, which are the convergence of structure and performance error. The convergence of structure means that the neural
networks acquire a stable structure from learning or the fuzzy systems find out a reasonable set of rules. A fast convergence of structure means that the self-organized neuro-fuzzy networks have a wonderful capability about the extraction of rules from limited samples. We use the coming time index of the first stable structure of network to show the speed of the convergence of structure, which is determined by the quality of structure learning. The time index begin at zero and plus one with the coming of every new sample. The convergence of performance error reflects the overall performance about the ability of learning of self-organized neuro-fuzzy networks, which involve the phase of structure learning and parameter learning. It is a comprehensive index, including the precision of convergence and the speed of convergence. The precision of convergence can be reflected in the predictive precision. And we choose the CPU time as the index of the speed of convergence. The learning time spent on each example is different, hence, the CPU time also reflect the computing burden of different self-organized neuro-fuzzy networks.

6.1. Nonlinear dynamic system identification

Nonlinear dynamic system identification, which is a classical benchmark problem, is used in several papers [18,8,13,10]. The nonlinear dynamic system is defined as follows:

\[ y(t+1) = \frac{y(t+y(t)+2.5)}{1+y(t)+y(t-1)} + u(t) \]  

(29)

where \( y(0) = 0, y(1) = 0, u(t) = \sin(2\pi t/25) \). And the model of this nonlinear dynamic system can be expressed by

\[ y(t+1) = f(y(t), y(t-1)) + u(t) \]  

(30)

This model takes three parameters as inputs of system, which are \( y(t), y(t-1) \) and \( u(t) \). The \( y(t+1) \) is taken as the output of neuro-fuzzy network. A set of samples are produced by Eq. (29), whose size is 500. The parameters of NFN-FOESA are shown as follows: \( \xi_1 = 0.1, \xi_2 = 0.01, \eta = 1, \lambda_{up} = 0.4, \lambda_{down} = 0.025 \).

The predictive process of NFN-FOESA on samples is using the learned neuro-fuzzy network at time \( (t-1) \) to predict the value of output at time \( t \). Hence, dividing the whole sample into training and testing sampling sets is unnecessary. In addition, in GP-FNN, we find that the modification about the network of neuro-fuzzy network is always happened at the testing period. This means that the rules learned by training sampling set are not suitable for testing sampling set. Therefore, dividing the sampling set into two parts is not useful for online self-organizing neuro-fuzzy network. Hence, we use \( \text{RMSE}(t) \) and \( \text{MAE}(t) \) as the criterion of performance of online self-organizing neuro-fuzzy networks. At first, the total sensitivity index and first order sensitivity index of rules in NFN-FOESA of the whole sample set are given in Fig. 6. We can find that the total sensitivity indexes and first order sensitivity indexes of these rules are coinciding. Hence, using first order global sensitivity indexes of rules in neuro-fuzzy networks can

Fig. 6. The total sensitivity indexes and first order sensitivity indexes of rules in NFN-FOESA, which are represented by \( \circ \) and \( \bullet \) respectively.

Fig. 7. Dynamic structure process of the NFN-FOESA.

provide all the information about the influence of rules on the variation of output. Then, the dynamic structure process of the NFN-FOESA is given in Fig. 7. A self-organizing neuro-fuzzy network can be adaptive with dynamic system by modifying its structure online. As the learned rules can handle this dynamic system, the structure learning will terminate. If self-organizing neuro-fuzzy network only has a limited ability of structure learning, the process of structure learning will occur repeatedly. We find that the first coming time of stable structure of NFN-FOESA is much quicker than some other methods. In GP-FNN, which uses the variance-based sensitivity analysis as NFN-FOESA, this process goes through over 250 samples. With the help of the concept of systemic fluctuation, NFN-FOESA can set a large control this process in less than 50 samples as FAOSPFNN. Different from GP-FNN, after the coming of first stable structure, there is not another occurrence of the modification of structure. It makes up for the deficiency that the structure of network is not as compact as GP-FNN. However, Fig. 7 only illustrates that NFN-FOESA can quickly find the rules of neuro-fuzzy network and these rules can handle the dynamic system. The fitness of these rules to dynamic system needs to be proved by the criterion of RMSE and MAE. The predictive accuracy of NFN-FOESA is given in Figs. 8 and 9. From these two figures, after 200 samples, the RMSE is nearly to 0.02. Hence, NFN-FOESA is a self-organizing neuro-fuzzy network with a faster convergence than other methods. At last, Fig. 10 illustrates that the NFN-FOESA achieves the good approximation and generalization performance for this nonlinear dynamic system. And in Table 1, we can see the comparison between different self-organizing neuro-fuzzy networks on different parameters of performance. With the only one whose RMSE is under 0.02 (where 0.02 = 0.01), the approximation and generalization performance of NFN-FOESA is the best of all in this problem. Hence, the whole learning mechanism stop with the coming of the last half of samples as shown in Fig. 9. And this leads to an increasing in the tail of RMSE(t). The learned structure of network is not the most compact one. However, we have showed that the fitness of rules to this system is satisfactory in Figs. 8 and 9. Hence, the learned structure of our method is a reasonable one. With the RBD and fast convergence, the speed of Fig. 8. The predictive accuracy (RMSE) of the first half of samples by NFN-FOESA.

Fig. 9. The predictive accuracy (RMSE) of the last half of samples by NFN-FOESA.

Table 1

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>No. of final neurons</th>
<th>CPU time (s)</th>
<th>Predictive accuracy</th>
<th>Time index</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>RMSE$^b$</td>
<td>MAE$^b$</td>
</tr>
<tr>
<td>NFN-FOESA</td>
<td>9</td>
<td>25.11</td>
<td>0.0098</td>
<td>0.0070</td>
</tr>
<tr>
<td>GP-FNN</td>
<td>7</td>
<td>26.36</td>
<td>0.0105</td>
<td>0.0079</td>
</tr>
<tr>
<td>DFNN</td>
<td>7</td>
<td>27.03</td>
<td>0.0137</td>
<td>0.0094</td>
</tr>
<tr>
<td>FAOSPFNN</td>
<td>7</td>
<td>12.08</td>
<td>0.0230</td>
<td>0.0179</td>
</tr>
<tr>
<td>GD-FNN</td>
<td>8</td>
<td>30.14</td>
<td>0.0108</td>
<td>0.0081</td>
</tr>
</tbody>
</table>

$^a$ The simulation A is begun with one initial neuron.

$^b$ RMSE and MAE are based on the total sample set, which are RMSE(500) and MAE(500).

$^c$ The coming time index of the first stable structure of network.

Fig. 10. The desired and predicted outputs by NFN-FOESA.

NFN-FOESA is faster than GP-FNN. However, the parameter learning of nine rules makes this advantage limited.

6.2. Mackey–Glass time-series prediction

Mackey–Glass time-series prediction is also a classical benchmark problem widely used in [18,8,13,10]. The mathematics model is given as follows:

\[ x(t + 1) = (1 - a)x(t) + \frac{bx(t - \tau)}{1 + x^\eta(t - \tau)} \]  
(31)

where the parameter \( a = 0.1, b = 0.2, \tau = 17, \) and the initial condition \( x(0) = 0.3. \) The problem is to predict \( x(t + h) \) from a past time series \( x(t), x(t - \Delta t), \ldots, x(t - (n - 1)\Delta t). \) As in [12], setting \( h = \Delta t = 6 \) and \( n = 4, \) the predict model is shown as follows:

\[ x(t + \Delta t) = f(x(t), x(t - 6), x(t - 12), x(t - 18)) \]  
(32)

The sample set produced by Eq. (31) has 1000 samples. The parameters of NFN-FOESA are shown as follows: \( \zeta_1 = 0.1, \zeta_2 = 0.011, \eta = 1, \lambda_{up} = 0.4, \lambda_{down} = 0.025. \)

In Fig. 11, the feasibility of the global sensitivity analysis for first order effect is proved again. From Figs. 12 and 13, we can find...
that the characters of NFN-FOESA in the simulation A also exist in the simulation B. At the same time, we can find that the final structure of NFN-FOESA is the same compact as GP-FNN in the Mackey–Glass time-series prediction in Table 2. Hence, the speed advantage of NFN-FOESA to GP-FNN is clearer than Simulation A. Also the coming of first stable structure of neuro-fuzzy network of NFN-FOESA is much faster than other self-organizing neuro-fuzzy networks (Fig. 14).

6.3. Short-term traffic flow forecasting

As a dynamic system, status of urban traffic system changes quickly. For the complexity of traffic system, short-term traffic flow forecasting is a difficult problem and one hot topic in traffic research area. In order to handle its dynamic property, traffic flow forecasting system is indispensable to traffic management system. And fuzzy logic is also used in traffic flow forecasting [29]. In this paper, we use the field data from the PtMS of Tianhe area in Guangzhou as sample set [30]. The size of data is six days and the record interval of the vehicle flow rate is 5 min. The model of one intersection in the road network is shown in Fig. 15.

The information of traffic flow in upstream links at time \((t-1)\) is used to predict the traffic flow in downstream link at time \(t\). The mathematical model of traffic flow forecasting is shown as following:

\[
t_f(t) = G(t_f(t-1), t_f(t-2), t_f(t-3))
\]

where \(t_f(t)\) is the vehicle flow rate on link \(i\) at time \(t\).

We use the NFN-FOESA to predict traffic flow on the downstream link. The dynamic structure process of the NFN-FOESA in Fig. 16 shows that the structure is fixed at the end of the first day. This phenomenon means that NFN-FOESA can find the appropriate rules quickly. Since the evening of the third day had more traffic flow data over 800 veh/h than the evenings of two former days, the network indeed has an alteration. From Figs. 17 and 18, we can see the predictive performance of NFN-FOESA in traffic flow forecasting. At the same time, two algorithms are used to compare with NFN-FOESA. One is the Bayesian approach [27]. The number of Gaussian models in the Mixture Gaussian Model of Bayesian approach is optimized by the X-means clustering approach [31]. The other one is the ELM approach [28]. We also use X-means clustering approach to classify the traffic flow in each upstream link. Then we use the information of clustering to build a reasonable neural network. And the ELM approach is used to train the neural network. Bayesian approach and ELM approach are offline learning approaches, the training data set and the test data set are the same in this simulation. NFN-FOESA is an online learning approach, it uses the past data to predict the future data. However, in Table 3, the RMSE of Bayesian approach is higher than NFN-FOESA. And considering that the hidden neurons in the neural network of ELM approach is 27, which is more than NFN-FOESA, the RMSE of ELM is nearly equal to NFN-FOESA. Hence, the NFN-FOESA preforms better than the other methods.

6.4. Discussions

In this section, we briefly discuss two limitations of NFN-FOESA.

6.4.1. Balance between predictive precision and speed

In the fine grained learning of NFN-FOESA, we have both widths-and centers-learning and weight-learning to acquire a better predictive precision as GP-FNN. Hence, compared with
FAOSPFNN, which only has weight-learning by RLS algorithm, speed is sacrificed for predictive precision.

6.4.2. Balance between compact structure and fast convergence

In order to avoid the phenomenon that adjustment of structure of neuro-fuzzy networks in GP-FNN lasts a long time, NFN-FOESA only process global sensitivity analysis with the appearance of a large systemic error or a large systemic fluctuation. Hence, the structure of NFN-FOESA may be not as compact as GP-FNN, like in simulation A. However, the concept of systemic fluctuation indeed helps us find the appropriate rules quickly and accelerates the convergence of NFN-FOESA.

7. Conclusion

In this paper, the self-organizing neuro-fuzzy networks are represented as an additive model. We prove that the global sensitivity analysis for first order effect can be used to substitute the global sensitivity analysis for total effect in self-organizing neuro-fuzzy networks. Hence, a new self-organizing neuro-fuzzy network, called NFN-FOESA, is introduced. NFN-FOESA can learn appropriate rules without initialization of the number of neurons in the hidden layer and the parameters of them. Two benchmark problems in the function approximation area and one real word problem are used to demonstrate the effectiveness of NFN-FOESA. Besides the advantages of variance based self-organizing neuro-fuzzy networks, the results of simulations show that NFN-FOESA has the following advantages compared with other self-organizing neuro-fuzzy networks:

1. The adjustment of structure of neuro-fuzzy networks not only happens at an appearance of large systemic error, but also with the occurrence of large systemic fluctuation. Hence, we will find the reasonable structure quicker than other methods, like GP-FNN, DFNN, FAOSPFNN and GD-FNN. Moreover, with the application of global sensitivity analysis for first order effect, the time of computation is less than the method based on total sensitivity analysis, like GP-FNN.
2. The approximation and generalization performance of NFN-FOESA is better than other self-organizing neuro-fuzzy networks commonly used in the literature, like GP-FNN, DFNN, FAOSPFNN and GD-FNN.
3. NFN-FOESA is a potential tool to deal with dynamic complex problems, like short-term traffic flow forecasting.

Acknowledgments

This work is under the support of NSFC 71232006 and 61233001. We would like to express our appreciation to the other students contributed to this work in the State Key Laboratory of Management and Control for Complex Systems, Chinese Academy of Sciences.
References


