2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)
July 6-11, 2014, Beijing, China

Non-linear Variable Structure Regression (VSR) and its Application in Time-Series Forecasting

Mohammad M. Korjani and Jerry M. Mendel
Ming Hsieh Department of Electrical Engineering
University of Southern California
Los Angeles, CA, USA
Korjani@ usc.edu, mendel@sipi.usc.edu

***Abstract*— Variable Structure Regression (VSR) is a new kind of non-linear regression model, which simultaneously determines the exact mathematical structure of non-linear regressors and how many regressors there are, thereby freeing the end user from trial and error time-consuming studies to determine these. The results are based on an iterative procedure for optimizing parameters and automatically identifying the structure of the VSR model. A novel feature of this new model is it not only uses a linguistic term for a variable but it also uses the complement of that term. It also provides the end user with a physical understanding of the regressors. A Monte Carlo study shows the practical accuracy of VSR model on the classical Gas Furnace time-series prediction problem. VSR ranked #1 compared to five other methods.**

Keywords*— *fuzzy rule-based systems; fuzzy sets; linguistic terms; non-linear regression; quantum particle swarm optimization

I. INTRODUCTION

Regression models are very widely used in just about all science, engineering and non-engineering real-world applications (e.g., behavioral science [8], biostatistics [42], business [14], econometrics [18], financial engineering [32], insurance [15], medicine [41], petroleum engineering [27], etc. [11], [20]). A typical linear regression model has the following structure [26]:

$$y(x_1, x_2, ..., x_p) = \beta_0 + \sum_{v=1}^{p} \beta_v x_v \qquad (1)$$

where β_0, β_v are the *regression coefficients*, and bias β_0 is a constant that does not depend on any of the variables (including such a term is a standard practice in regression models).

Assuming that an output has a linear dependence on its variables is often too simplistic for many real-world applications; hence, the following *nonlinear regression model* is often used:

$$y(x_1, x_2, ..., x_p) = \beta_0 + \sum_{v=1}^{R_S} \beta_v \varphi_v(x_1, x_2, ..., x_p) \qquad (2)$$

in which the *regressors* $\varphi_v(x_1, x_2, ..., x_p)$ are nonlinear functions of $x_1, x_2, ..., x_p$. These nonlinear functions are often also called *basis functions*, so in this paper we use the terms "regressors" and "basis functions" interchangeably. Many choices have been made in the past for the basis functions, e.g., polynomials (orthogonal and non-orthogonal), trigonometric, Gaussian, radial, fuzzy, etc.

There are many methods for non-linear regression, e.g., logistic regression [31] kernel-based regression [34], neural networks [16], rule-based fuzzy logic systems [22], etc.; however, there are four major challenges to implementing (2): 1) Choosing the variables; 2) Choosing the nonlinear structures of the regressors; 3) Choosing how many terms to include in (2), namely R_S ; and, 4) Optimizing the parameters that complete the description of the model.

For Challenge 1, how to choose the variables is crucial to the success of any regression model. In this paper we assume that the user has already established the variables that affect the outcome, using methods already available for doing this (e.g., [28]). *Our focus in this paper is on Challenges 2-4*.

For Challenge 2, in real-world applications the nonlinear structures of the regressors are usually not known ahead of time, and are therefore chosen either as products of the variables (e.g., two at a time, three at a time, etc.), or in other more complicated ways (e.g., trigonometric-, exponential-, logarithmic-functions, etc.). Sometimes a deep knowledge about the application provides justifications for the choices made for the nonlinear terms; however, often one does not have such deep knowledge, and a lot of time is spent, using trial and error, trying to establish the nonlinear dependencies. We shall demonstrate below that *Variable Structure Regression (VSR) establishes the exact nonlinear structure for each of the* R_S *regressors in (2) automatically*.

For Challenge 3, how to choose R_S is also usually done by trial and error, and this can be very tedious to do. We shall also demonstrate below that *VSR establishes* R_S *automatically*.

For Challenge 4, in addition to the regression coefficients that appear in (2), each regressor in VSR will be a parametric function of variables, and numerical values must be specified for all such parameters. Usually, one does not know how to specify such numerical values ahead of time. Instead, as we will explain below, *VSR follows the now common practice of determining numerical values for all such parameters, as well as for the regression coefficients in (2) by using some given data and one or more optimization methods that make the regression model optimally fit that data*.

As a result of our solutions to Challenges 2-4, the nonlinear regression model in (2) will have a variable structure, which is why we have called this kind of regression VSR. Exactly what we mean by a "variable structure" is deferred until Sections V, because the variability of the structure in (2) occurs in two different ways in VSR.

This study was funded by the Center of Excellence for Research and Academic Training on Interactive Smart Oilfield Technologies (CiSoft); CiSoft is a joint University of Southern California–Chevron initiative.

The rest of this paper is organized as follows: Section II deals with measured data and explains how the data are preprocessed; Section III explains how the antecedents of rules as well as the number of rules are determined simultaneously from the data; Section IV explains how rules are established and how the formula for the VSR model in (2) derives from them; Section V explains how all of the parameters in the VSR model as well as its structure are optimized; Section VI presents the experimental results; and Section VII provides conclusions, the strengths of VSR and future works.

II. PREPROCESSING

The variables and output that are used in the VSR model must be chosen by the end-user; this is done before the start of the VSR procedure. Assume a data pair is $(\mathbf{x}(t), y(t))$ where $y(t)$ is the output for that $\mathbf{x}(t)$. Each data pair is treated as a "case" and index t denotes a data case. In a time-series forecasting application the data cases have a natural temporal ordering. We assume that *N data pairs* are available, and refer to the collection of these data pairs as S_{Cases}, where:

$$S_{Cases} = \{(\mathbf{x}(t), y(t))\}_{t=1}^{N} \tag{3}$$

Assume that N data pairs are divided into three data sets one of which is used for *training*, one is used for *validation* and the last one is used for *testing*. More specifically:

- N_{trn} of the N data cases form the *training set*, S_{Cases}^{trn}, where

$$S_{Cases}^{trn} = \{\mathbf{x}_{trn}(t) : y_{trn}(t)\}_{t=1}^{N_{trn}} \tag{4}$$

- N_{val} from the N input-output data pairs form the *validation set*, S_{Cases}^{val}, where

$$S_{Cases}^{val} = \{\mathbf{x}_{val}(t) : y_{val}(t)\}_{t=1}^{N_{val}} \tag{5}$$

- N_{test} from the N input-output data pairs form the *testing set*, S_{Cases}^{test}, where

$$S_{Cases}^{test} = \{\mathbf{x}_{test}(t) : y_{test}(t)\}_{t=1}^{N_{test}} \tag{6}$$

We randomly select 10% of data for testing. The remaining 90% of data are divided into training and validation folds. VSR uses a *5-fold cross-validation* [10] in which the learning data are randomly partitioned into five data sets, each containing 20% of the remaining data, after which four of the five folds are used for training and one fold is used for validation (this is repeated five times). The training set is used to optimize the parameters of (2), the validation set is used to stop the training, and the testing set is used to evaluate the overall performance of the optimized model; this is fully explained in Section V.

For preprocessing, linguistic terms are assigned to each variable x_i $(i = 1,..., p)$, $x_i \in R^+$. We begin with just two linguistic terms (*Low* x_i and *High* x_i) for each variable in order to keep the number of term-parameters in (2) as small as possible, because all of these parameters are optimized and if not enough data pairs are available, then optimizing too large a number of term-parameters is problematic. Each linguistic term is modeled as a type-1 fuzzy set (T1 FS), and so a membership function (MF) has to be found for each term.

III. ESTABLISH ANTECEDENTS OF RULES AND THE NUMBER OF RULES

VSR simultaneously establishes the if-part (the *antecedent*) of a rule, as well as the number of rules, R_s. Each rule will later correspond to one basis function in (2). The antecedent of each rule contains one linguistic term or its complement for each of the p variables, and each of these linguistic terms is combined with the others by using the word "and" (e.g., A_1 and A_2 and ... and A_p). This interconnection is called a *causal combination*.

Note that in a traditional if-then rule the antecedents only use the terms and not their complements. In VSR, protection about being wrong for postulating a term is achieved by considering each term and its complement. This is different from Ruspini partition of fuzzy sets [33] where summation of MFs are equal to 1. These methods are similar only if one term is assigned to each variable; for more than one term per variable, each term is treated independently in VSR method. So antecedents of the rules contain all of the terms related to a variable.

To begin, 2^p *candidate causal combinations* (the 2 is due to both the term and its complement) are conceptually postulated. One does not know ahead of time which of the 2^p candidate causal combinations should actually be used as a compound antecedent in a rule. VSR prunes this large collection by using the MFs that were determined in Section II, as well as the MF (using fuzzy set mathematics) for "A_1 and A_2 and ... and A_p," and a simple test. The results of doing this are R_s *surviving causal combinations*.

Let S_F be the finite space of 2^p candidate causal combinations, F_j ($j = 1,..., 2^p$ and $i = 1,..., p$):

$$\begin{cases} \mathbf{S}_F = \{F_1,..., F_{2^p}\} \\ F_j = A_1^j \wedge A_2^j \wedge ... \wedge A_i^j \wedge ... \wedge A_p^j \\ A_i^j = C_i \text{ or } c_i \end{cases} \tag{7}$$

where $\wedge$ denotes conjunction (the "and" operator) and is modeled using *minimum* and c_i denotes the complement of C_i. Mendel and Korjani [25] have proven the somewhat surprising result that *for each case only* ***one*** *of the* 2^p *candidate causal combinations has a MF value that is > 0.5*. More importantly, they have provided a simple formula for establishing exactly which candidate causal combination that is. Their result is provided in the following:

Min-Max Theorem: Given p terms, C_1, C_2, …, C_p and their respective complements, c_1, c_2, …, c_p. Consider the 2^p candidate causal combinations $(j = 1,..., 2^p)$ $F_j = A_1^j \wedge ... \wedge A_i^j \wedge ... \wedge A_p^j$ where $A_i^j = C_i$ or c_i and $i = 1,..., p$. Let

$$\mu_{F_j}(t) = \min\{\mu_{A_1^j}(t),..., \mu_{A_i^j}(t),..., \mu_{A_p^j}(t)\},\ t = 1, 2,..., N_{trn} \tag{8}$$

where

$$\mu_{A_i^j}(t) = \mu_{C_i}(t) \text{ or } \mu_{c_i}(t) = 1 - \mu_{C_i}(t), \quad i = 1,...,p \tag{9}$$

Then for each t (case) there is only one j, $j^*(t)$, for which $\mu_{F_{j^*(t)}}(t) > 0.5$ and $\mu_{F_{j^*(t)}}(t)$ can be computed as:

$$\mu_{F_{j^*(t)}}(t) = \min\left\{\max\left(\mu_{C_1}(t), \mu_{c_1}(t)\right),...,\max\left(\mu_{C_p}(t), \mu_{c_p}(t)\right)\right\} \tag{10}$$

$F_{j^*(t)}(t)$ is determined from the right-hand side of (9), as:

$$\begin{aligned} F_{j^*}(t) &= arg\max\left(\mu_{C_1}(t), \mu_{c_1}(t)\right) \wedge .... \wedge arg\max\left(\mu_{C_p}(t), \mu_{c_p}(t)\right) \\ &= A_1^{j^*(t)} \wedge ... \wedge A_p^{j^*(t)} \end{aligned} \tag{11}$$

In (11), $arg\max\left(\mu_{C_i}(t), \mu_{c_i}(t)\right)$ denotes the winner of $\max\left(\mu_{C_i}(t), \mu_{c_i}(t)\right)$, namely C_i or c_i. ■

Not all of the N_{trn} winning causal combinations will be different, i.e. the same winner may frequently occurs for more than one case. Consequently, after the winning causal combination is found for each of the N_{trn} cases, the J uniquely different $F_{j^*}(t)$ are found; and, they are relabeled $F_{j'}$ $(j' = 1,...,J)$.

We are now ready to state how the R_S surviving causal combinations are actually computed:

1. Compute $F_{j^*}(t)$ using (11).
2. Find the J uniquely different $F_{j^*}(t)$ and re-label them $F_{j'}$ $(j' = 1,...,J)$.
3. Compute $t_{F_{j'}}$, where $(t = 1,...,N_{trn})$

$$t_{F_{j'}}(t) = \begin{cases} 1 & \text{if } F_{j'} = F_{j^*(t)}(t) \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

4. Compute $N_{F_{j'}}$, where

$$N_{F_{j'}} = \sum_{t=1}^{N_{trn}} t_{F_{j'}}(t) \tag{13}$$

5. Establish the R_S surviving causal combinations F_v^S $(v = 1,...,R_S)$, as:

$$F_v^S = \begin{cases} F_{j'}(j' \to v) & \text{if } N_{F_{j'}} \geq f \\ 0 & \text{if } N_{F_{j'}} < f \end{cases} \tag{14}$$

where $F_{j'}(j' \to v)$ means $F_{j'}$ is added to the set of surviving causal combinations as F_v^S, and v is the index of the surviving set.

In order to implement (14) threshold f has to be chosen. In our works, we chose $f = 1$. This choice is arbitrary and depends on an application and how many cases are available. Discussions on how to choose f are given in [23, Section 3.6], [30, Ch. 5, p. 107], [29, p. 197], and [13, p. 197].

IV. Establsih Rules and VSR Equations

The R_S surviving causal combinations lead to the following TSK rules [36], [38], [22, Ch. 13] ($v = 1,...,R_S$):

$$S_v: \text{IF } x_1 \text{ is } A_1^v \text{ ... and } x_p \text{ is } A_p^v, \text{ THEN } y_v(\mathbf{x}) = \beta_v \tag{15}$$

where the constants β_v have yet to be determined [they will be the regression coefficients that appear in (2)].

Note, again, that these rules are different from the usual kinds of rules that appear in a fuzzy logic rule-based system because in (15) a term may be the complement of a fuzzy set rather than just the fuzzy set.

The MF of the antecedents of each rule in (15) is $\mu_{F_v^S}(\mathbf{x})$, where:

$$\mu_{F_v^S}(\mathbf{x}) = \mu_{A_1^v}(x_1) \bigstar \mu_{A_2^v}(x_2) \bigstar ... \bigstar \mu_{A_p^v}(x_p) \tag{16}$$

Note that $\mu_{F_v^S}(\mathbf{x})$ is a highly nonlinear-function of the input variables because of the nonlinear dependence of each MF on its input variable [e.g., $\mu_{A_1^v}(x_1)$]. The other source of non-linearity comes from (16) where the MFs connected by the t-norms. The t-norm ★ in (16) is chosen as the product.

The formula for the *VSR* model begins with (15) and (16) and assumes that fired rules are aggregated using Center of Sets (COS) defuzzification, and is (see [22, Ch. 13, Section 13.2.1]):

$$\gamma(\mathbf{x}) = \frac{\sum_{v=1}^{R_S} \beta_v \mu_{F_v^S}(\mathbf{x})}{\sum_{v=1}^{R_S} \mu_{F_v^S}(\mathbf{x})} \tag{17}$$

which can also be written, as:

$$\gamma(\mathbf{x}) = \sum_{v=1}^{R_S} \beta_v \left[\frac{\mu_{F_v^S}(\mathbf{x})}{\sum_{v=1}^{R_S} \mu_{F_v^S}(\mathbf{x})}\right] \tag{18}$$

When a bias is added to (18), as is done in a regression model, then

$$y(\mathbf{x}) = \beta_0 + \gamma(\mathbf{x}) = \beta_0 + \sum_{v=1}^{R_S} \beta_v \left[\frac{\mu_{F_v^S}(\mathbf{x})}{\sum_{v=1}^{R_S} \mu_{F_v^S}(\mathbf{x})}\right] \tag{19}$$

(19) is now in the form of a *basis function expansion* [44], [22, Ch. 5] in which the basis functions[1], denoted $\varphi_v(\mathbf{x})$, are:

$$\varphi_v(\mathbf{x}) \equiv \frac{\mu_{F_v^S}(\mathbf{x})}{\sum_{v=1}^{R_S} \mu_{F_v^S}(\mathbf{x})} \tag{20}$$

Consequently, (19) can also be expressed as:

$$y(\mathbf{x}) = \beta_0 + \sum_{v=1}^{R_S} \beta_v \varphi_v(\mathbf{x}) \tag{21}$$

(21) is our *VSR* model, as stated in (2), except that now we have an explicit formula for the basis functions that is given by (20) along with (16).

V. Optimizing Parameters and Structure of VSR Model

Evolutionary algorithms constitute a class of search methods that can be used to optimize fuzzy rules, e.g., [1], [9], and [39]. Fazzolari et al. recently [12] review all evolutionary fuzzy systems. The objective of these methods is to find an

[1] These basis functions have also been called *fuzzy basis functions* (FBFs), and (19) has also been called a FBF expansion.

optimal set of rules as well as MFs by encoding both MFs and the rules in a chromosome and the optimizing an objective function using an evolutionary algorithm. The drawback of these methods is the dimension of the search space increases significantly when both rules and MFs are encoded, which makes it very complicated to find solutions. A solution to this is to use a conventional linguistic rule based system in order to reduce the search space [2]; however, these methods suffer from exponential rule explosion [49]. The VSR method only optimizes MFs and automatically identifies an associated set of rules to reduce the search space; and the search space is only related to MF parameters. Mendel and Korjani [24] show that the number of rules generated using the min-max theorem is very small; so rule explosion is not encountered.

In order to completely specify the VSR model in (21) we need to specify its structure as well as the numerical values for all of its parameters. The *structure* of (21) is established once we know R_s and the surviving causal combinations. The parameters in (21) are of two kinds, MF parameters that appear in the basis functions and the regression coefficients; both kinds of parameters need to be determined before (21) is completely specified. Recall, however, that both R_s and the surviving causal combinations depend on the MFs for each of the p variables; so, when the MF parameters change this may cause the structure of (21) to also change. Consequently, we will optimize both the structure of the VSR model as well as its parameters. This is summarized in the high-level flow chart in Fig. 1. Observe that the outer loop is devoted to *structure identification* and the inner loop is devoted to *parameter optimization*.

A. Parameter Optimization

There are different approaches for optimizing the parameters in the VSR model (e.g., see [22, Ch. 13, Section 13.2.4]). We determine MF parameters and regression coefficients iteratively by iterating between a linear optimization for the regression coefficient parameters and a nonlinear optimization for the MF parameters.

A.1 Optimizing the Regression Coefficients: The least squares (LS) method (e.g., [21 Lesson 3]) is used to find the regression coefficients, β_0 and β_v ($v = 1,...,R_S$) by using the training data. The training data are also used to compute the training error. In addition, the validation data are used to compute a validation error that is needed later to help find the overall optimized VSR model, as is explained below in Section C.

Using the notations in (4) and (5), for the elements in our training and validation sets, (21) can be expressed for each of those data sets, as:

$$y_{trn}(t) = \beta_0 + \sum_{v=1}^{R_S} \beta_v \varphi_v(\mathbf{x}_{trn}(t)) \quad t = 1,...,N_{trn} \tag{22}$$

$$y_{val}(t) = \beta_0 + \sum_{v=1}^{R_S} \beta_v \varphi_v(\mathbf{x}_{val}(t)) \quad t = 1,...,N_{val} \tag{23}$$

Collecting the N_{trn} and N_{val} equations in (22) and (23), they can be expressed more compactly in vector-matrix format, as:

$$\mathbf{y}_{trn} = \mathbf{\Phi}_{trn}\boldsymbol{\beta} \tag{24}$$

$$\mathbf{y}_{val} = \mathbf{\Phi}_{val}\boldsymbol{\beta} \tag{25}$$

where

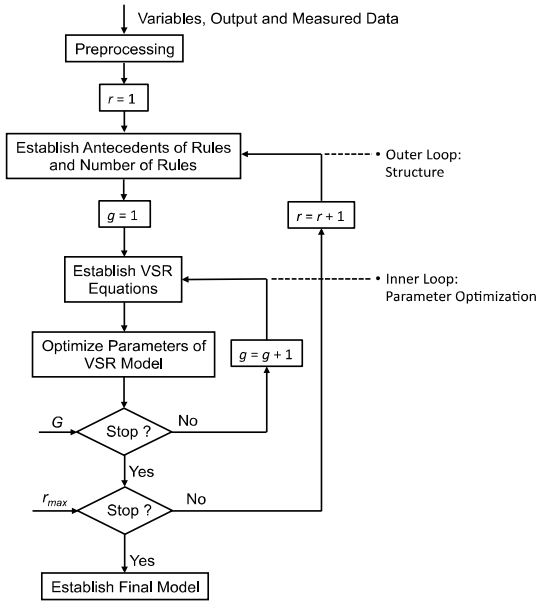

Figure 1. High-level flow-chart for the VSR model.

$$\mathbf{y}_{trn} = [y_{trn}(1),..., y_{trn}(N_{trn})]^T \tag{26}$$

$$\mathbf{y}_{val} = [y_{val}(1),..., y_{val}(N_{val})]^T \tag{27}$$

$$\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2,..., \beta_{R_S}]^T \tag{28}$$

$$\mathbf{\Phi}_{trn} = \begin{bmatrix} 1 & \varphi_1(\mathbf{x}_{trn}(1)) & \cdots & \varphi_{R_S}(\mathbf{x}_{trn}(1)) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \varphi_1(\mathbf{x}_{trn}(N_{trn})) & \cdots & \varphi_{R_S}(\mathbf{x}_{trn}(N_{trn})) \end{bmatrix} \tag{29}$$

Note that $\mathbf{\Phi}_{val}$ is constructed similar to (29) using $\mathbf{x}_{val}$. The least-squares optimized regression coefficients, $\boldsymbol{\beta}_{LS}$, obtained by minimizing $J(\boldsymbol{\beta}) = \|\mathbf{y}_{trn} - \mathbf{\Phi}_{trn}\boldsymbol{\beta}\|^2$, can be expressed, as [21, Lesson 3]:

$$\boldsymbol{\beta}_{LS} = (\mathbf{\Phi}_{trn}^T \mathbf{\Phi}_{trn})^{-1} \mathbf{\Phi}_{trn}^T \mathbf{y}_{trn} \tag{30}$$

We do not actually compute $\boldsymbol{\beta}_{LS}$ using (30), because to do so is well known to be fraught with numerical difficulties; instead, the *Singular Value Decomposition* (SVD) method is used (e.g., [21, Lesson 4]) because of its excellent numerical properties.

After $\boldsymbol{\beta}_{LS}$ is computed, the training and validation RMSEs are computed, as:

$$J_{trn} = \left(\|\mathbf{y}_{trn} - \mathbf{\Phi}_{trn}\boldsymbol{\beta}_{LS}\|^2 / N_{trn} \right)^{0.5} \tag{31}$$

$$J_{val} = \left(\|\mathbf{y}_{val} - \mathbf{\Phi}_{val}\boldsymbol{\beta}_{LS}\|^2 / N_{val} \right)^{0.5} \tag{32}$$

A.2 Optimizing MF Parameters: In this step, the MF parameters are optimized; however, as we have mentioned above, when the MF parameters change, the basis functions in (20) must also be changed, after which the LS optimized regression coefficients must also be changed. Consequently, there is a natural iteration (G times) between optimizing the regression coefficients and optimizing the MF parameters.

In this step one uses the same surviving R_S causal combinations (i.e., the same *structure* of the VSR model) that were found in Section IV; however, because the MFs are changed in the present step from those that were used in Section IV, the structure of the rules must be modified for the optimized MFs.

In order to optimize the MF parameters one must first choose a parametric model for each MF. We use piecewise linear MF models and Quantum Particle Swarm Optimization (QPSO) (e.g., [46]-[51]) as our MF parameter optimization method. Any other swarm optimization procedure can be used instead of QPSO.

In this paper, we only use one term per variable (e.g., right MFs, $\mu_H(x_i)$) to reduce MF parameters. Note that if more than one term is used for each variable then each term is treated independently; so the antecedent of VSR rules contains all terms (or their complement) assigned to variables.

For a right-shoulder MF (Fig. 2), the MF model is $(i = 1,..., p)$:

$$\mu_H(x_i) = \begin{cases} 0 & x_i \le a_i \\ \dfrac{x_i - a_i}{b_i - a_i} & a_i < x_i < b_i \\ 1 & x_i \ge b_i \end{cases} \tag{33}$$

This MF is described by two parameters, a_i and b_i.

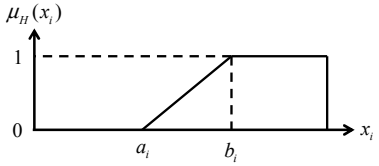

Figure 2. Piecewise-linear right-shoulder MF.

QPSO (with M particles, we chose $M = 1000$) is used to optimize the MF parameters by minimizing the objective function $J_{trn}(\boldsymbol{\theta}_m) = \left\| \mathbf{y}_{trn} - \Phi_{trn}(\boldsymbol{\theta}_m)\boldsymbol{\beta}_{LS} \right\|^2$. The MF parameters that have been collected into vector $\boldsymbol{\theta}_m$ are in the matrix $\Phi_{trn}(\boldsymbol{\theta}_m)$ and are initialized randomly. A description of our QPSO algorithm is given in [46]-[51].

After G generations of QPSO the one model that has the smallest validation error is found and saved, i.e.:

$$J_{val}\left(\boldsymbol{\theta}_m^*\right) = \left\| \mathbf{y}_{val} - \Phi_{val}\left(\boldsymbol{\theta}_m^*\right)\boldsymbol{\beta}_{LS}\left(\boldsymbol{\theta}_m^*\right) \right\|^2 \tag{34}$$

$$\begin{aligned} \boldsymbol{\theta}_m^* &= arg \min_{\substack{g=1,...,G \\ m=1,...,M}} J_{val}\left(\boldsymbol{\theta}_m(g)\right) \\ &= arg \min_{\substack{g=1,...,G \\ m=1,...,M}} \left\| \mathbf{y}_{val} - \Phi_{val}(\boldsymbol{\theta}_m(g))\boldsymbol{\beta}_{LS}(\boldsymbol{\theta}_m(g)) \right\|^2 \end{aligned} \tag{35}$$

The value $\boldsymbol{\theta}_m^*$ establishes the parameters [$\varphi_v\left(\mathbf{x} \mid \boldsymbol{\theta}_m^*\right)$ and $\boldsymbol{\beta}_{LS}(\boldsymbol{\theta}_m^*)$] for the winning model, and that model is expressed as:

$$y(\mathbf{x} \mid \boldsymbol{\theta}_m^*) = \boldsymbol{\beta}_{LS,0}\left(\boldsymbol{\theta}_m^*\right) + \sum_{v=1}^{R_S} \boldsymbol{\beta}_{LS,v}\left(\boldsymbol{\theta}_m^*\right)\varphi_v\left(\mathbf{x} \mid \boldsymbol{\theta}_m^*\right) \tag{36}$$

B. Structure Identification

After G generations of parameter optimization have been completed, one passes from the Fig. 1 inner loop to the outer-loop structure-optimization stopping rule. Until that stopping rule is satisfied, the antecedents of the rules as well as their number are re-established by using the five-step procedure that is described in Section III. Now, however, the MFs that are used in that procedure use the MF parameters that are in $\boldsymbol{\theta}_m^*$.

Structure identification is performed a pre-specified ($r_{\max}$) number of times (we chose $r_{\max} = 100$ iterations; see Fig. 1) or until the same set of rules appears in any one of the $r_{\max}$ structure identification iterations[2].

The $r_{\max}$ iterations of the structure identification outer loop lead to $r_{\max}$ models $\{y^{(r)}(\mathbf{x})\}_{r=1}^{r_{\max}}$, each obtained as explained for (34)-(36) and described by

$$\left\{ \begin{matrix} \left\{ \varphi_v^{(r)}\left(\mathbf{x} \mid \boldsymbol{\theta}_m^*\left(g(r)\right)\right),\ \boldsymbol{\beta}_{LS,v}^{(r)}\left(\boldsymbol{\theta}_m^*\left(g(r)\right)\right) \right\}_{v=1}^{R_S(r)} \\ ,\ J_{val}^{(r)}\left(\boldsymbol{\theta}_m^*\left(g(r)\right)\right) \end{matrix} \right\}_{r=1}^{r_{\max}} \tag{37}$$

One reason for referring to (22) as a "variable-structure" model is because of the structure identification that occurs during the complete design of the VSR model during whose iterations the structure of the model changes.

C. Establishing the Final Model

Our final model is the model in (38) that has the smallest validation error, namely:

$$\begin{aligned} y^*(\mathbf{x}) &= \boldsymbol{\beta}_{LS,0}^{(r^*)}\left(\boldsymbol{\theta}_m^*\left(g(r^*)\right)\right) \\ &\quad + \sum_{v=1}^{R_S(r^*)} \boldsymbol{\beta}_{LS,v}^{(r^*)}\left(\boldsymbol{\theta}_m^*\left(g(r^*)\right)\right)\varphi_v^{(r^*)}\left(\mathbf{x} \mid \boldsymbol{\theta}_m^*\left(g(r^*)\right)\right) \end{aligned} \tag{38}$$

$$r^* = arg \min_{r=1,...,r_{\max}} J_{val}^{(r)}\left(\boldsymbol{\theta}_m^*\left(g(r^*)\right)\right) \tag{39}$$

D. Test on Testing Data

MF values for the testing data are computed from (33)-(34). Then the FBFs of the testing data are obtained similar to (29) using $\mathbf{x}_{test}$. The testing error is

$$J_{test}\left(\boldsymbol{\theta}_m^*\right) = \left(\left\| \mathbf{y}_{test} - \Phi_{test}\left(\boldsymbol{\theta}_m^*\right)\boldsymbol{\beta}_{LS}\left(\boldsymbol{\theta}_m^*\right) \right\|^2 \Big/ N_{test} \right)^{0.5} \tag{40}$$

VI. Experimental Results

We have applied VSR to the Gas Furnace time-series prediction problem for which the gas furnace data was recorded from a combustion process of a methane–air mixture. The gas flowing into the furnace, $u(t)$, and the CO_2 concentration in outlet gas, $o(t)$, were measured and the data were sampled every 9 secs. to generate 296 pairs of input-output measurements. Box and Jenkins [4] used a time-series

[2] When a set of rules re-appears, QPSO will converge to the same set of results if the number of particles and iterations of QPSO are large enough. In our applications of VSR we chose 1000 particles and 200 iterations of QPSO and have observed the re-appearance of a set of rules many times.

based approach to develop a model based on this data set, which has been referred to as the linear model. Many other authors have used this data set as a benchmark problem, e.g., [7], [35], [37], [40], [43] and have approached this forecasting and identification problem in many ways. Unfortunately authors do not present their results in a consistent and fully-explained manner, which makes it impossible to compare new results with all past results. We used the same approach that is explained in [7] and compared our results with methods presented in [7], i.e., [5], [6] and [17].

We used $u(t), u(t+1), u(t+2), o(t), o(t+1),$ and $o(t+2)$ to predict $o(t+3)$ for which there were 293 cases S_{Cases} $(t = 1,...,293)$:

$$S_{Cases} = \{\mathbf{x}(t), y(t)\}_{t=1}^{293} \tag{41}$$

$$\mathbf{x}(t) = \left[u(t), u(t+1), u(t+2), o(t), o(t+1), o(t+2)\right] \tag{42}$$

$$y(t) = o(t+3) \tag{43}$$

We randomly selected 28 cases for testing. To obtain the five data sets that were used in five-fold cross-validation [10], we randomly sampled the remaining 265 cases into five subsets (folds). In each of the five folds we used four of the data subsets (212 cases) for training and the remaining data subset (53 cases) for validation. In the sequel we illustrate the entire VSR procedure only for the first fold.

To begin, we applied LM-FCM for two clusters to all 212 cases from the first fold and obtained two shoulder MFs that are complements of one another. Because our causal conditions are time-advanced versions of $u(t)$ and $o(t)$, we used the same MFs for the time-advanced versions of $u(t)$ and $o(t)$ as their initial MFs. Fig. 3 shows the initial MFs of $u(t)$ and $o(t)$ extracted by means of LM-FCM.

Let $u(t) \equiv x_1$, $u(t+1) \equiv x_2$, $u(t+2) \equiv x_3$, $o(t) \equiv x_4$, $o(t+1) \equiv x_5$ and $o(t+2) \equiv x_6$. For six causal conditions there are $2^6 = 64$ causal combinations, and their MFs have to be evaluated for 212 cases. The min-max formulas in (9) and (10) were used to find the winning causal combination for each of the 212 cases. Only 22 causal combinations survived; they are given in Table I as well as the number of cases associated with each of them. These 22 surviving causal combinations were substituted into (21), (20) and (16) in order to establish the VSR equation, whose 23 regression coefficients were then computed via SVD, the results being: $\beta_0 = 45.7350$ and $\beta_1, ..., \beta_{22}$ whose values are given in the last column of Table I.

The QPSO-optimized MFs (which are complement of each other, for each term) are depicted in Fig. 4. Observe that these MFs are quite different from the LM-FCM MFs in Fig. 3.

After finding the optimized MFs and regression coefficients, the following training and validation objective function values were computed for the first complete iteration ($r = 1$) of the first fold, using (31) and (32): $J_{trn}(1) = 0.4236$ and $J_{val}(1) = 0.4528$.

This entire procedure was then repeated for $r = 2$. The optimized MFs from $r = 1$ were used to establish the structure of a new set of rules for $r = 2$, etc. When $r = 6$ the rules became the same as those from $r = 5$, so it was not necessary to go through all of the $r_{\max} = 100$ iterations in the outer loop. Fig. 5 shows surviving rules[3] for each iteration. Observe that the sets of rules change for the outer-loop iterations, which is why we call this method "Variable Structure" regression, and the original 22 rules have been reduced to 12.

Our final model was established by using (39) and (40). The final model with the smallest validation error occurs for $r^* = 5$; its rules are given in Table II, $\beta_0 = 23.5242$ and its regression coefficients $\beta_1, ..., \beta_{12}$ are given in the last column of that table. The final optimized MFs are depicted in Fig. 6; they are quite different looking from the LM-FCM MFs in Fig. 3 and fold 1 optimized MFs in Fig. 4. The final training, validation, and testing RMSEs are $J^*_{trn} = 0.3423$, $J^*_{val} = 0.3393$, and $J^*_{test} = 0.3985$.

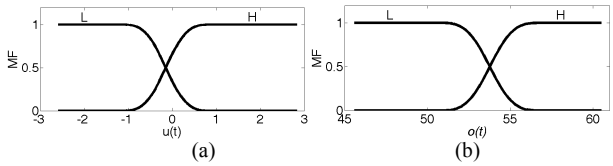

Figure 3. LM-FCM MFs (a) $u(t)$ and (b) $o(t)$.

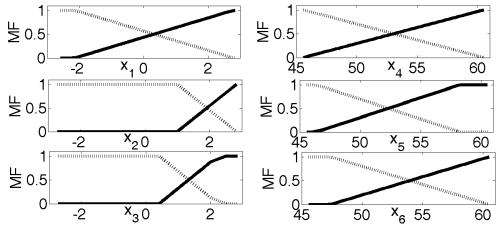

Figure 4. Fold 1 optimized MFs for outer-loop iteration #1.

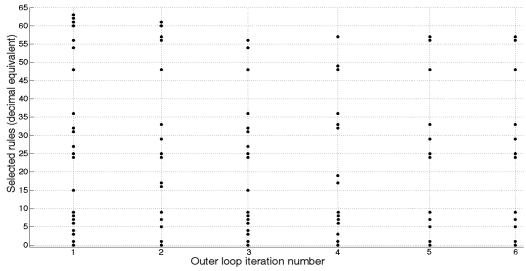

Figure 5. Decimal equivalent number of surviving rules.

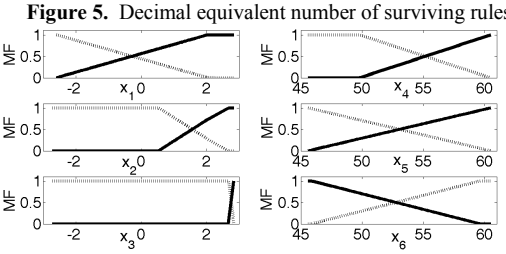

Figure 6. Fold 1 final optimized MFs.

[3] The antecedent of each rule contains a causal condition or its complement; hence, each rule can be represented as a binary number. Its corresponding decimal number is used to show the surviving rules in each outer-loop iteration.

Table I
Fold 1 First Iteration Surviving Causal Combinations and Regression Coefficients for the Gas Furnace Time-Series Prediction Problem[a]

Rule No.	H_1	H_2	H_3	H_4	H_5	H_6	No. of cases	Coef.
1	1	1	1	0	0	0	60	6.2752
2	0	0	0	1	0	1	30	-5.3301
3	0	0	0	0	0	0	24	-4.3126
4	0	0	0	1	1	1	15	17.776
5	0	0	0	0	0	1	15	0.7950
6	0	1	1	1	0	1	14	-1.3865
7	1	1	1	1	0	1	8	-33.458
8	0	1	1	0	0	0	6	-3.8284
9	0	1	1	1	1	1	5	9.2796
10	1	1	0	0	0	0	5	-0.8878
11	0	0	1	0	0	1	5	4.1604
12	1	1	1	1	0	0	5	16.507
13	0	1	1	0	0	1	4	0.3909
14	0	0	1	1	0	1	4	0.2358
15	1	1	1	0	0	1	3	-108.09
16	1	0	0	0	0	0	2	2.1477
17	0	0	0	1	0	0	2	1.3447
18	0	0	1	0	0	0	1	-8.7195
19	0	0	1	1	1	1	1	105.72
20	1	1	1	1	1	1	1	12.389
21	0	1	0	1	1	1	1	-9.7826
22	1	1	0	1	0	1	1	4.9738

[a] 1 represents the causal condition and 0 represents its complement.

Table II
Fold 1 Final Rules and Regression Coefficients For the Gas Furnace Time Series Prediction Problem[a]

Rule No.	H_1	H_2	H_3	H_4	H_5	H_6	No. of cases	Coef.
1	1	0	0	0	0	0	97	-2.3573
2	0	0	0	0	0	1	27	8.5824
3	1	0	0	0	0	1	25	11.6158
4	1	0	0	1	0	1	22	-1.1925
5	0	0	0	1	0	1	16	-20.323
6	0	0	0	1	1	1	10	5.0235
7	0	0	0	0	0	0	6	13.1022
8	1	0	0	1	1	1	3	13.5792
9	1	1	1	0	0	0	2	-0.2999
10	1	1	0	0	0	0	2	-2.0210
11	1	0	0	1	0	0	1	21.3691
12	1	0	1	0	0	0	1	6.9904

[a] 1 represents the causal condition and 0 represents complement of the causal condition.

We performed a double Monte-Carlo simulation where for each of 15 randomly chosen test sets a 5-fold cross-validation was performed 30 times. The average number of rules of the VSR method for the Gas Furnace prediction problem is 15.23. The number of generated fuzzy rules in the fuzzy rule base based on the Wang and Mendel method [45] is 72 when four isosceles triangular fuzzy sets were assigned to each variable. Note that the number of rules from the VSR method is considerably smaller than the number of rules from the Wang and Mendel method.

The average RMSE after the 15×30 double Monte-Carlo simulations of the five-fold cross-validation method ($15\times30\times5=2250$ RMSEs) obtained by the VSR method is 0.3310 ± 0.0753 whereas the average FRI RMSE is 0.7787 ± 0.0031. The standard deviation (STD) is larger for VSR than for FRI; however, because the average RMSE for VSR is much smaller than of FRI this is not very important.

Table III summarizes the standard deviation and average RMSE for all 2250 data sets. Table IV shows that the VSR method obtained the smallest average RMSE than the other methods reported on in [5], [6], [7] and [17].

Table III
Average RMSE and STD for Different Subsets of Data

	Training	Validation	Testing	All
Mean	0.3136	0.3277	0.4066	0.3310
STD	0.0605	0.0925	0.2205	0.0753

Table IV
A Comparison of the Average RMSE for Different Methods[a]

	HS [17]	CCL [5]	T1 CK [6]	FRI [7]	IT2 FRI [7]	VSR
Gas Furnace	1.2640	2.0914	1.2688	0.8573	0.7787	**0.3310**

[a] The numbers for the first five methods were taken from Table VI in [6].

VII. CONCLUSIONS

VSR automatically finds the number of terms (R_S+1) and establishes the mathematical structures of each of the terms in the nonlinear regression model (2), thereby freeing the end-user from time-consuming trial and error studies to determine this. Each term in the VSR model has a linguistic interpretation, because each term in the final optimized VSR model is associated with one linguistic if-then rule, thereby providing the end user with a physical understanding for each term. The importance of each rule can be determined automatically, by examining the number of cases that support each rule, thereby helping the end user to better understand the model.

The VSR model, just as all prior rule based models, acts like a multitude of models, because the number of its regressors that are actually activated by a set of measured variables changes automatically, i.e. different groups of regressors in the VSR model (2) are activated depending upon what the numerical values are for the variables. Therefore, a variable-structure model is very different from a traditional regression model in which all of the terms in (2) are always activated and contribute to the final answer regardless of the numerical values of the input variables.

Although the application of the VSR model to the Gas Furnace time-series prediction problem has demonstrated that it provides the best results as compared to five other methods still more research is needed on different aspects of the VSR method. Many extensions are possible:

1. From T1 to Interval Type-2 (IT2) to General Type-2 (GT2)
2. More than one term per variable (refer to [32] for theoretical results that demonstrates the impossibility of certain causal combinations to even exist)
3. Same kind of preprocessing can be used for a VS Classifier (VSC); the nonlinear discriminant functions could have the same structure as (21); but now all parameters are optimized to minimize classification errors.

ACKNOWLEDGMENT

The authors wish to acknowledge their interactions with Mark Rodighiero during the course of this research and the development of Fig. 5. They would also thank Prof. Iraj Ershaghi (USC Director of CiSoft) for his continued support of the VSR research program.

REFERENCES

[1] R. Alcala, J. Alcala-Fdez, and F. Herrera, "A proposal for the genetic lateral tuning of linguistic fuzzy systems and its interaction with rule

selection," *IEEE Trans. on Fuzzy System*, vol. 15, no. 4, pp. 616–635, Aug. 2007.

[2] R. Alcala, M.J. Gacto, and F. Herrera, "A fast and scalable multiobjective genetic fuzzy System for linguistic fuzzy modeling in high-dimensional regression problems," *IEEE Trans. on Fuzzy Systems*, vol.19, no.4, pp.666-681, Aug. 2011.

[3] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer, Norwell, MA, 1981.

[4] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis: Forecasting and Control*, 4th ed., Wiley Interscience, NJ, 2008.

[5] Y. C. Chang, S. M. Chen, and C. J. Liau, "Fuzzy interpolative reasoning for sparse fuzzy rule-based systems based on the areas of fuzzy sets," *IEEE Trans. Fuzzy Syst.*, vol. 16, no. 5, pp. 1285–1301, Oct. 2008.

[6] S. M. Chen and Y. K. Ko, "Fuzzy interpolative reasoning for sparse fuzzy rule-based systems based on cutting and transformation techniques," *IEEE Trans. Fuzzy Syst.*, vol. 16, no. 6, pp. 1626–1648, Dec. 2008.

[7] S. Chen, Y. Chang, and J. Pan*, "*Fuzzy Rules Interpolation for Sparse Fuzzy Rule-Based Systems Based on Interval Type-2 Gaussian Fuzzy Sets and Genetic Algorithms*," IEEE Trans. On Fuzzy Syst.*, Vol. 21, No. 3, June 2013.

[8] J. Cohen, P. Cohen, S. G. West and L. S. Aiken, *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences (2nd ed.),* Lawrence Erlbaum Associates, Hillsdale, NJ, 2003.

[9] O. Cordon, F. Herrera, and P. Villar, "Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base," *IEEE Trans. on Fuzzy Systems*, vol. 9, no. 4, pp. 667–674, Aug. 2001.

[10] P. A. Devijver and J. Kittler, Pattern Recognition: *A Statistical Approach*, London, U.K.: Prentice–Hall, 1982.

[11] N. R. Draper and H. Smith, *Applied Regression Analysis (3rd ed.)*, John Wiley, New York, 1998.

[12] M. Fazzolari, R. Alcala, Y. Nojima, H. Ishibuchi, and F. Herrera, "A review of the application of multiobjective evolutionary fuzzy systems: current status and further directions," *IEEE Trans. on Fuzzy Systems*, vol.21, no.1, pp.45,65, Feb. 2013.

[13] P. Fiss, "Building better causal theories: a fuzzy set approach to typologies in organization research," *Academy of Management Journal*, vol. 54, pp. 393-420, 2011.

[14] E. W. Frees, *Data Analysis Using Regression Models: The Business Perspective,* Prentice-Hall, Upper Saddle River, NJ, 1996.

[15] E. W. Frees, *Regression Modeling with Actuarial and Financial Applications*, Cambridge, Univ. Press, New York, 2009.

[16] S, Haykin, *Neural Networks and Learning Machines* (3rd ed.), Prentice Hall, Upper Saddleback River, NJ, 2008.

[17] Z. H. Huang and Q. Shen, "Fuzzy interpolative reasoning via scale and move transformations," *IEEE Trans. Fuzzy Syst.*, vol. 14, no. 2, pp. 340– 359, Apr. 2006.

[18] J. Kmenta, *Elements of Econometrics*, Macmillan, New York, 1971.

[19] M. M. Korjani and J. M. Mendel, "Fuzzy set qualitative comparative analysis (fsQCA): challenges and applications," *NAFIPS Conf.*, Berkeley, CA, Aug. 2012

[20] M. H. Kutner, C. Nachtsheim and J. Neter, *Applied Linear Regression Models(4th ed.),* McGraw-Hill, New York, 2004.

[21] J. M. Mendel, *Lessons in Estimation Theory for Signal Processing Communications and Control*, Prentice-Hall, Upper Saddleback River, NJ, 1995.

[22] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Prentice-Hall, Upper Saddleback River, NJ, 2001.

[23] J. M. Mendel and M. M. Korjani, "Charles Ragin's fuzzy set qualitative comparative analyses (fsQCA) used for linguistic summarization," *Information Sciences*, vol. 202, pp. 1-23, 2012.

[24] J. M. Mendel and M. M. Korjani, "On establishing nonlinear combinations of variables from small to big data for use in later processing," *submitted to Information Sciences*, 2014.

[25] J. M. Mendel and M. M. Korjani, "Theoretical aspects of fuzzy set qualitative comparative analysis (fsQCA)," *Information Sciences*, vol. 237, pp. 137-161, 2013.

[26] D. C. Montgomery, E. A. Peck and G. G. Vining, *Introduction to Linear Regression Analysis, 5th Ed.*, Wiley, Hoboken, NJ, 2013.

[27] I. B. Mutafar and R. Razali, "A study on prediction of output in oilfield using multiple linear regression," *Int'l. J. of Applied Science and Technology*, vol. 1, no. 4, pp. 107-113, July 2011.

[28] L. L. Nathans, F. L. Oswald and N. Kim, "Interpreting multiple linear regression: a guidebook of variable importance," in *Practical Assessment, Research & Evaluation*, vol. 17, vo. 9, April 2012.

[29] C. C. Ragin, *Redesigning Social Inquiry: Fuzzy Sets and Beyond*, Univ. of Chicago Press, Chicago, IL, 2008.

[30] B. Rihoux and C. C. Ragin (Eds.), *Configurational Comparative Methods: Qualitative Comparative Analysis (QCA) and Related Techniques*, SAGE, Los Angeles, CA, 2009.

[31] C. Ritz and J. C. Streibig, *Nonlinear Regression With R*, Springer, New York, 2008.

[32] D. Ruppert, *Statistics and Data Analysis for Financial Engineering*, Springer, New York, 2010.

[33] E.H. Ruspini, "A New Approach to Clustering", *Information and Control*, Vo1.15, pp.22-32, 1969.

[34] G. A. F Seber and C. J. Wild, *Nonlinear Regression,* Wiley, New York, 1989.

[35] H. H. Stokes, M. Hinich, "Detecting and modeling nonlinearity in the gas furnace data*," Journal of Computational Statistics*, vol. 26, pp. 77-93, 2011.

[36] M. Sugeno and G. T. Kang, "Structure identification of fuzzy models," *Fuzzy Sets and Systems*, vol. 28, pp.15-33, 1988.

[37] M. Sugeno, and T. Yasukawa, "A Fuzzy Logic Based Approach to Qualitative Modeling," *IEEE Trans. on Fuzzy Systems*, vol. 1, pp. 7-31, 1993.

[38] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 15, pp. 116-132, 1985.

[39] Y. Teng and W. Wang, "Constructing a user-friendly ga-based fuzzy system directly from numerical data," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 5, pp. 2060–2070, Oct. 2004.

[40] R. M. Tong, "Synthesis of Fuzzy Models For Industrial Processes: Some Recent Results," *Int. J. General Systems*, vol. 4, pp. 143-162, 1978.

[41] W. Vach, *Regression Models as a Tool in Medical Research*, CRC Press, 2012.

[42] E. Vittinghoff, D. V. Glidden, S. C. Shiboski and C. E. McCulloch, *Regression Methods in Biostatistics: Linear, Logistic, Survival, and Repeated Measures Models (2nd ed.)*, Springer, New York, 2011.

[43] L. Wang and R. Langari, "Complex Systems Modeling via Fuzzy Logic," *IEEE Trans. Syst. Man Cybern.*, vol. 26(1), pp.100-106, 1996.

[44] L.-X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation and orthogonal least squares learning," *IEEE Trans. on Neural Networks*, vol. 3, pp. 807-813, Sept. 1992.

[45] L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. Syst. Man Cybern.*, vol. 22, no. 6, pp. 1414–1426, Nov./Dec. 1992.

[46] X. Wang, Y. He, L. Dong and H. Zhao, "Particle swarm optimization for determining fuzzy measures from data," *Information Sciences*, vol. 181, pp. 4230-4252, Oct. 2011.

[47] M. Xi, J. Sun and W. Xu, "An improved quantum-behaved particle swarm optimization algorithm with weighted mean best position," *Applied Mathematics and Computation*, vol. 205, pp. 751-759, 2008.

[48] S. Yang, M. Wang and L. Jiao, "A quantum particle swarm optimization," *Congress on Evolutionary Computation,* vol. 1, pp. 320-324, 2004.

[49] J. Yaochu, "Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement," *IEEE Trans. on Fuzzy Systems*, vol.8, no.2, pp.212-221, Apr. 2000.

[50] D. Zhai, M. Hao, J.M. Mendel, "A non-singleton interval type-2 fuzzy logic system for universal image noise removal using quantum-behaved particle swarm optimization," *IEEE Trans. On Fuzzy Systems*, vol. 19, pp. 957-964, Jun. 2011.

[51] D. Zhai, M. Hao, J.M. Mendel, "Universal image noise removal filter based on type-2 fuzzy logic system and QPSO," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 20, pp. 207-232, 2012.