SOLUTION OF NONLINEAR SISO SYSTEM THROUGH DIFFERENT PARADIGMS

Rubem Vasconcelos Pacelli[1]
[1]Federal University of Ceará, Forlateza – Ceará, Brazilian
e-mail: rubem070@alu.ufc.br

***Abstract* – This paper shows different methods for solving a nonlinear regression problem. The same dataset with 250 observations of a nonlinear SISO (single input, single output) system is used for the following methods: least-squares regressor by parts, kth order polynomial regressor, Mamdani fuzzy system, and 0-order Takagi-Sugeno fuzzy system. All models are analyzed via R^2 for different hyperparameters (polynomial order, number of intervals, etc...). The best solutions' residues and scatter plots are analyzed and discussed.**

***Keywords* – Nonlinear regression problem, Mamdani fuzzy system, Takagi-Sugeno fuzzy system, polynomial regressor, regression by parts.**

I. Least-Squares by parts

The Lest-Squares (LS) method is an approximation technique for overdetermined systems that aims to minimize the squared value of the residuals. Such systems are characterized by having more equations than variables and are easily found in practice [1].

Consider the example of a discrete-time SISO system, where $x_n, y_n \in \mathbb{R}$ are, respectively, their input and output values at the instant $n \in \{1, 2, ..., N\}$. At each instant, one has an equation where the input and output are related through a set of $K+1$ parameters, $\boldsymbol{\theta} \in \mathbb{R}^{K+1}$. Mathematically, one can define the output variable as

$$y_n = f(x_n; \boldsymbol{\theta}) \tag{1}$$

When $f(\cdot)$ is a linear function, the LS problem is commonly called Ordinary Least-Square (OLS). The least-squares method aims to minimize the following cost function

$$J\left(\hat{\boldsymbol{\theta}}\right) = \sum_{n=1}^{N} e_n^2, \tag{2}$$

where $\hat{y}_n$ and $\hat{\boldsymbol{\theta}}$ are the estimates of y_n and $\boldsymbol{\theta}$, respectively, and $e_n = y_n - \hat{y}_n$ is the residual signal. Although the OLS method has no optimality associated with it, various practical problems, such as regression analysis, can be solved via OLS since no probabilistic assumptions need to be made about the data [1].

The linear regressor output of the SISO system can be expressed as

$$\hat{y}_n = f(x_n; \hat{\boldsymbol{\theta}}) = \hat{a}x_n + \hat{b}, \tag{3}$$

where $\hat{\boldsymbol{\theta}} = \begin{bmatrix} \hat{a} & \hat{b} \end{bmatrix}^\mathsf{T}$. By using the Equation (3), we can rewrite te cost function as

$$J\left(\hat{\boldsymbol{\theta}}\right) = \sum_{n=1}^{N} \left(y_n - \hat{a}x - \hat{b}\right)^2. \tag{4}$$

The Equation (4) describes a convex function whose surface is a hyperparaboloid. The minimum value of the cost function corresponds to the set of coefficients sought [2]. In other words, we can state that

$$\exists\ \hat{\boldsymbol{\theta}}^\star \in \mathbb{R}^{K+1} \mid J\left(\hat{\boldsymbol{\theta}}^\star\right) < J\left(\hat{\boldsymbol{\theta}}\right)\ \ \forall\ \hat{\boldsymbol{\theta}} \neq \hat{\boldsymbol{\theta}}^\star \tag{5}$$

By calculating the derivative of $J\left(\hat{\boldsymbol{\theta}}\right)$ with respect to $\hat{a}$ and $\hat{b}$, we get

$$\frac{\partial J\left(\hat{\boldsymbol{\theta}}\right)}{\partial \hat{a}} = -2\sum_{n=1}^{N} x_n \left(\hat{y}_n - \hat{a}x_n - \hat{b}\right) = 0 \tag{6}$$

and

$$\frac{\partial J\left(\hat{\boldsymbol{\theta}}\right)}{\partial \hat{b}} = -2\sum_{n=1}^{N} \left(\hat{y}_n - \hat{a}x_n - \hat{b}\right) = 0, \tag{7}$$

respectively. The solution of this system of equations is given by

$$\hat{a} = \frac{\hat{\sigma}_{xy}}{\hat{\sigma}_x^2} \tag{8}$$

and

$$\hat{b} = \hat{\mu}_y - \hat{a}\hat{\mu}_x, \tag{9}$$

where $\hat{\mu}_\cdot$ and $\hat{\sigma}_\cdot^2$ are the sample mean and sample variance with respect to its subscript, respectively, and $\hat{\sigma}_{xy}$ is the estimate of the covariance of x_n and y_n.

Although the solution of the SISO OLS method is rather straightforward, many input-output relationships found in practice have nonlinearities. In these cases, one can resort to applying a transformation to the data in order to linearize the problem. Another approach is to utilize the OLS method in intervals where the scatter plot behaves approximately linear, yielding a set of linear curves with their respective parameters for each path. It is also possible to exploit other nonlinear regression methods, such as polynomial regression. The scatter plot of the dataset, shown in Figure 1, suggests that the input-output relationship is severely nonlinear. Nonetheless, there are intervals where the function can be approximated to a linear curve. A natural hyperparameter arises in this approach:

the number of intervals considered. The main trade-off is that the more intervals considered, the better the performance of the coefficient of determination tends to be. However, more parameters are needed to characterize the curve. The best solution is to solve the nonlinear problem with as few parameters as possible.

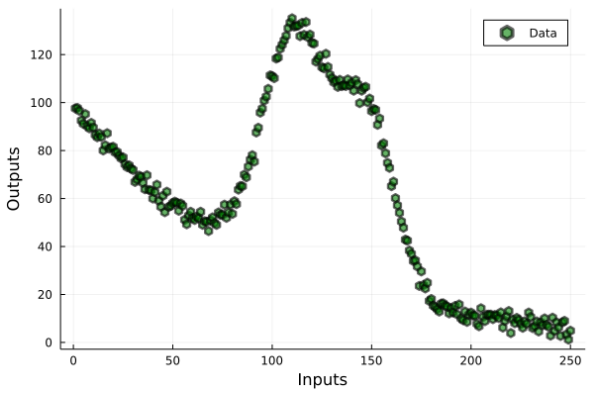


Fig. 1. Scatter plot of the dataset.

In this paper, the OLS algorithm by parts is implemented for different sets of curve intervals, $\{R_i\}_{i=1}^{I}$, where $I \in \{4,5,6\}$ and R_i is the ith interval. Since it is obtained a coefficient of determination, R^2, for each interval, the mean, μ_{R^2}, and the variance, $\sigma^2_{R^2}$, is analyzed for each configuration. The Figure 2 shows where the delimiters have been placed, in addition to the linear curves obtained by the OLS algorithm. The Algorithm 1 summarizes the behavior of the OLS algorithm by parts, and the Table I shows the performance for each value of I.

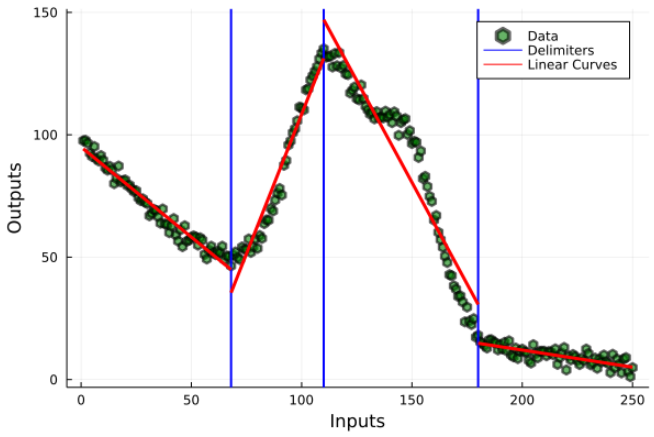
Algorithm 1: OLS algorithm by parts

Input: $\{(x_n, y_n)\}_{n=1}^{N}$
for $I \in \{4,5,6\}$ **do**
  for $i \in \{1,2,...I\}$ **do**
    for $(x_n, y_n) \in R_i$ **do**
      $N_i \leftarrow$ number of samples that belongs to R_i
      $\hat{\mu}_x \leftarrow \frac{1}{N_i}\sum x_n$
      $\hat{\mu}_y \leftarrow \frac{1}{N_i}\sum y_n$
      $\hat{\sigma}_{xy} \leftarrow \frac{1}{N_i}\sum x_n y_n - \hat{\mu}_x\hat{\mu}_y$
      $\hat{\sigma}_x^2 \leftarrow \frac{1}{N_i}\sum x_n^2 - \hat{\mu}_x^2$
      $\hat{a} \leftarrow \frac{\hat{\sigma}_{xy}}{\hat{\sigma}_x^2}$
      $\hat{b} \leftarrow \hat{\mu}_y - \hat{a}\hat{\mu}_x$

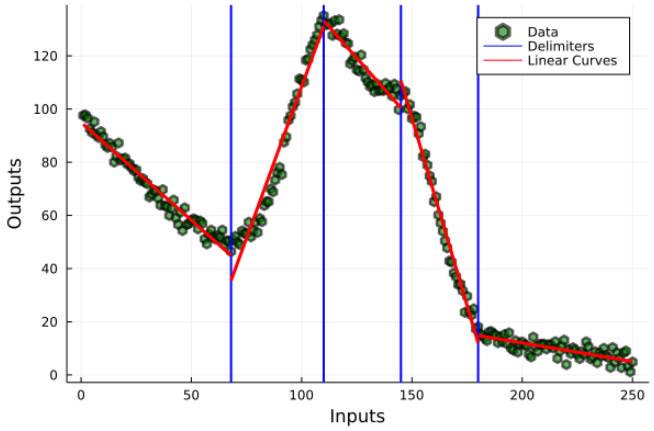
TABLE I
OLS by parts performance - R^2

I	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$	μ_{R^2}	$\sigma^2_{R^2}$
4	0.962	0.952	0.911	0.627	NaN	NaN	0.863	0.018
5	**0.962**	**0.952**	**0.895**	**0.985**	**0.627**	**NaN**	**0.884**	**0.017**
6	0.962	0.952	0.876	0.314	0.985	0.627	0.786	0.059

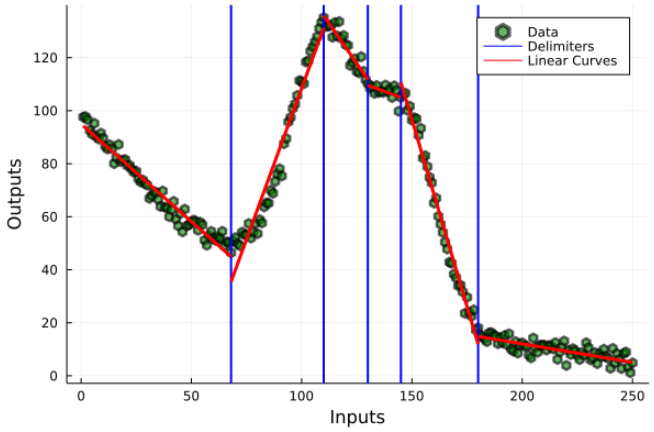
The best solution is found for $I = 5$, highlighted in the Table I. Although it is expected to achieve better performance as it is increased the number of intervals, it is possible to notice by the Figure 2c that the placement of the delimiters does not



(a) $I = 4$



(b) $I = 5$



(c) $I = 6$

Fig. 2. The OLS method by parts.

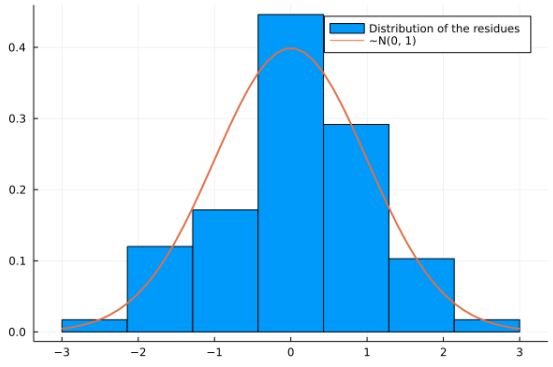


Fig. 3. Distribution of the residuals.

contribute for a good linear approximation (there are outliers for $i = 4$). This fact decreases R^2 and makes it worse when compared with the case where $I = 5$.

In addition to the regressor performance, one can analyze the distribution of the residuals. Let ξ_n be the normalized residual value, that is, $\xi_n = e_n/\sigma_e$. For a good placement of the intervals, the normalized residual distribution approximates to a zero-mean Gaussian distribution with unitary variance, i.e., $\xi_n \sim N(0,1)$. The Figure 3 shows the distribution (for $i = 1$) of the residuals along with the Gaussian distribution [3].

II. Polynomial Regression

The Polynomial regression is a nonlinear method which extends the concept of the SISO OLS algorithm. At the instant n, the input x_n is utilized to generate the tuple $(h_{n,1}, h_{n,2}, ..., h_{n,K})$, where $h_{n,k} = x_n^k$, $\forall\ k \in \{1,2,...,K\}$. Therefore, the SISO model is transformed into a MISO (Multiple Input, Single Output) model, where the kth input of the nth instant is x_n to the power k. Note that, albeit this model is not linear with relation to x_n, it is with relation to $h_{n,k}$, that is,

$$\begin{aligned} y_n &= f(x_n;\hat{\boldsymbol{\theta}}) = \mathbf{h}_n^{\mathsf{T}}\hat{\boldsymbol{\theta}} \\ &= \hat{\theta}_0 + \hat{\theta}_1 h_{n,1} + \hat{\theta}_2 h_{n,2} + \cdots + \hat{\theta}_K h_{n,K}, \end{aligned} \tag{10}$$

where $\hat{\boldsymbol{\theta}} = \begin{bmatrix}\hat{\theta}_0 & \hat{\theta}_0 & \cdots & \hat{\theta}_K\end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{K+1}$ and $\mathbf{h}_n = \begin{bmatrix}1 & h_{n,1} & h_{n,2} & \cdots & h_{n,K}\end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{K+1}$. The matricial notation for all N observations is given by

$$\hat{\mathbf{y}} = \mathbf{H}\hat{\boldsymbol{\theta}} \tag{11}$$

where $\mathbf{y} = \begin{bmatrix}y_1 & y_2 & \cdots & y_N\end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^N$ and

$$\mathbf{H} = \begin{bmatrix} 1 & h_{1,1} & h_{1,2} & \cdots & h_{1,K} \\ 1 & h_{2,1} & h_{2,2} & \cdots & h_{2,K} \\ \vdots & & & \ddots & \vdots \\ 1 & h_{N,1} & h_{N,2} & \cdots & h_{N,K} \end{bmatrix} \in \mathbb{R}^{N\times(K+1)} \tag{12}$$

The cost function is given by

$$\begin{aligned} J\left(\hat{\boldsymbol{\theta}}\right) &= \left(\mathbf{y} - \mathbf{H}\hat{\boldsymbol{\theta}}\right)^{\mathsf{T}}\left(\mathbf{y} - \mathbf{H}\hat{\boldsymbol{\theta}}\right) \\ &= \mathbf{y}^{\mathsf{T}}\mathbf{y} - 2\mathbf{y}^{\mathsf{T}}\mathbf{H}\hat{\boldsymbol{\theta}} + \hat{\boldsymbol{\theta}}^{\mathsf{T}}\mathbf{H}^{\mathsf{T}}\mathbf{H}\hat{\boldsymbol{\theta}} \end{aligned} \tag{13}$$

By differentiating the Equation (13) with respect to $\hat{\boldsymbol{\theta}}$ and setting its result to zero, we get

$$\begin{aligned} &\frac{\partial J\left(\hat{\boldsymbol{\theta}}\right)}{\partial \hat{\boldsymbol{\theta}}} = -2\mathbf{H}^{\mathsf{T}}\mathbf{y} + 2\mathbf{H}^{\mathsf{T}}\mathbf{H}\hat{\boldsymbol{\theta}} = 0 \\ &\therefore \hat{\boldsymbol{\theta}} = \mathbf{H}^{\dagger}\mathbf{y}, \end{aligned} \tag{14}$$

where $\mathbf{H}^{\dagger} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}$ is the pseudoinverse of $\mathbf{H}$, also called left inverse since $\mathbf{H}$ is a tall matrix ($N > K$). The fact that $\mathbf{H}$ is full rank ($rank(\mathbf{H}) = K+1$) guarantees the invertibility of $\mathbf{H}^T\mathbf{H}$ [4]. The procedure of the Kth-order polynomial regression is summarized in the Algorithm 2, where the hyperparameter K is analyzed for the set $\{5,6,7\}$.

Algorithm 2: Kth-order polynomial regressor.

Input: $\{(x_n, y_n)\}_{n=1}^N$
for $K \in \{5,6,7\}$ **do**
  $\mathbf{y} \leftarrow$ generated from $\{y_n\}_{n=1}^N$
  $\mathbf{H} \leftarrow$ generated from $\{x_n\}_{n=1}^N$
  $\mathbf{H}^{\dagger} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}$ /* The pseudoinverse */
  $\hat{\boldsymbol{\theta}} = \mathbf{H}^{\dagger}\mathbf{y}$

The Figure 4 shows the curve fitting for the Kth-order polynomial regressor. As shown in the Table II, the 6th-oder polynomial curve reached the best result. For this case, the model achieved the Pearson correlation coefficient between $\mathbf{y}$ and $\hat{\mathbf{y}}$ of 97.84%.

TABLE II
Kth-order polynomial regressor - R^2

$K=5$	$K=6$	$K=7$
0.85	**0.957**	0.566

III. Mamdani fuzzy system

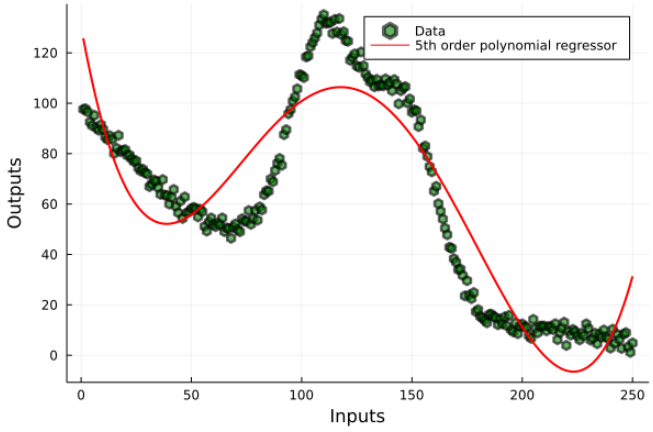
The fuzzy system consists of a continuum-valued logic in which the truth value may be in the interval between 0 and 1. This contrasts with the so-called ordinary sets, that has only two truth values: 0 and 1. This approach enables to model the vagueness or uncertainty encountered in many real problems.

The fuzzy model maps the input variable domain to the $[0,1]$ codomain through a membership function, This process is commonly referred as fuzzification. Based on a set of rules, a fuzzy inference system (FIS) uses the truth values of the fuzzified input to infer how much the antecedents impact the consequent. The fuzzy set obtained by the inference is then converted to a crisp value that outputs the system. This process of obtaining a crisp value from the output fuzzy set is commonly referred as defuzzification.

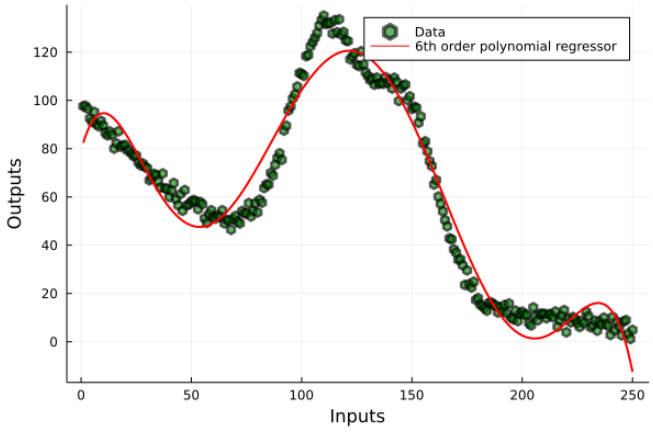
There are different possibilities to create a fuzzy system. A well-established fuzzy model the is the Mamdani system, whose set of rules can be generically written as

IF x_n IS A_1 AND x_n IS A_2... AND x_n IS A_K
THEN y_n IS B_i,

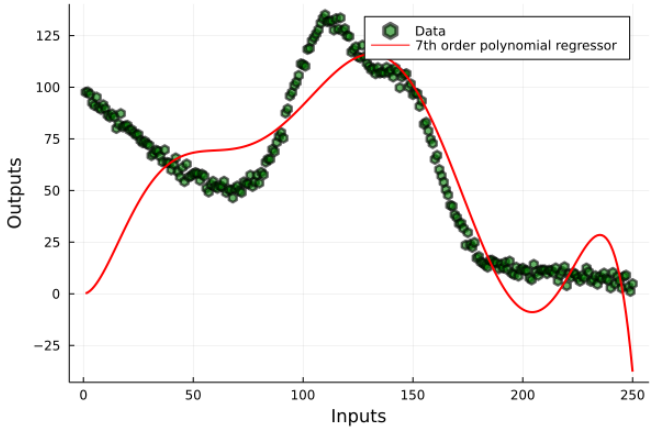
where B_i and $\{A_1, A_2, \cdots, A_K\}$ are the output and input fuzzy



(a) $K = 5$



(b) $K = 6$



(c) $K = 7$

Fig. 4. The Kth-order polynomial regressor.

sets for the ith rule, respectively, and "AND" is a fuzzy operator. For this problem, we consider $K \in \{2, 3\}$. The set of rules applied for $K = 2$ is given by

IF x_n IS LOW OR x_n IS HIGH
THEN y_n IS LOW,

IF x_n IS NOT LOW AND x_n IS NOT HIGH
THEN y_n IS HIGH,

and the set of rules applied for $K = 3$ is

IF x_n IS MEDIUM
THEN y_n IS HIGH,

IF x_n IS LOW
THEN y_n IS MEDIUM,

IF x_n IS HIGH
THEN y_n IS LOW.

The fuzzy sets cited in these rules (LOW and HIGH for $K = 2$, and LOW, MEDIUM, and HIGH for $K = 3$) are depicted in the Figure 5. The Gaussian shape of the fuzzy sets and their parameters (mean and variance) were defined experimentally in an attempt to obtain the best performance. Once the Gaussian function reaches its peak, its value it is held to maintain the true value at the extremes.

After the input and output fuzzification and the rule definitions, it is necessary to define an inference method. Among the different inference methods, the Modus Ponens, defined by

$$\mu^{(i)}_{A\to B}(x_n; y_n) = \min\left\{\mu^{(i)}_A(x_n), \mu^{(i)}_B(y_n)\right\}, \qquad (15)$$

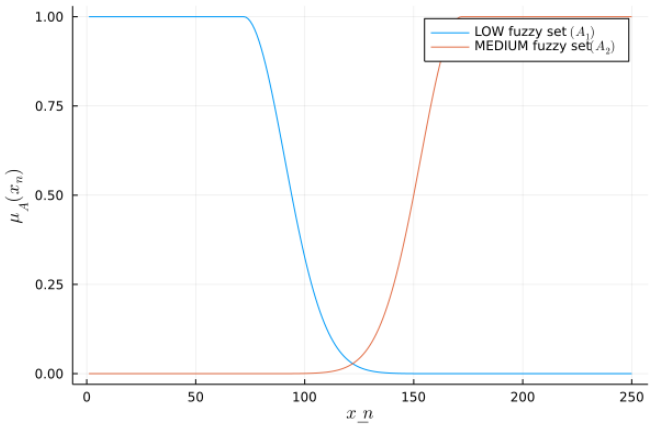
is widely adopted. In this equation, $\mu^{(i)}_{A\to B}(x_n; y_n)$ is the output fuzzy set of the ith rule and $\mu^{(i)}_{\cdot}$ is the membership function with respect to its subscript.

The output fuzzy sets of all rules are aggregated (with the $\vee$ operator) and the resulting fuzzy set is defuzzified to a crisp value using one of several methods described in the literature. A very popular method is the "centroid", which utilizes the center of mass of the output fuzzy set. The Algorithm 3 summarizes the procedure of the Mamdani fuzzy system.

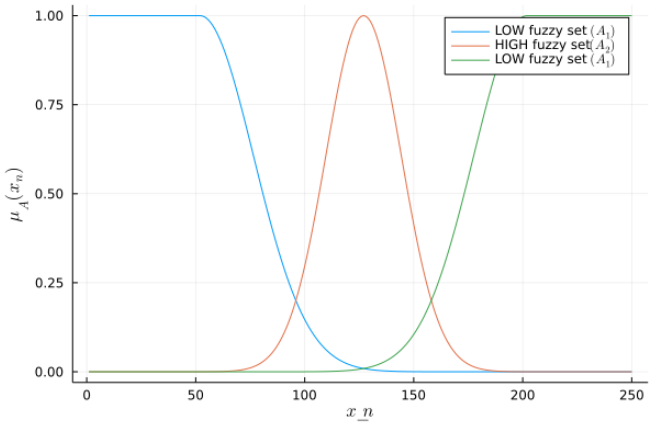
The solution of the regression problem using Mamdani fuzzy system for $K = 3$ is shown in Figure 6. The final model achieved a coefficient of determination of 0.91, and the Pearson correlation between the input and predicted value was 0.96.

Zero-order Takagi-Sugeno Fuzzy system

In 1985, Takagi and Sugeno introduced a fuzzy model where its consequence is a linear input-output relation [5]. The method is known as "nth-order Takagi-Sugeno model", where the linear function initially proposed in [5] is generalized to an nth-order polynomial function.



(a) $K = 2$



(b) $K = 3$

Fig. 5. Input fizzy sets. The output fuzzy set follows the same parameters, but with different means.

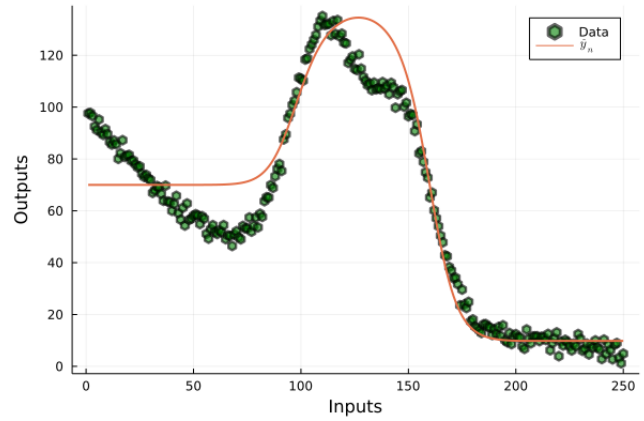


Fig. 6. The curve fit for the Mamdani fuzzy system.

Algorithm 3: Mamdani fuzzy model

Input: $\{(x_n, y_n)\}_{n=1}^{N}$

for $K \in \{2,3\}$ **do**

  for $n \in \{1,2,\cdots,N\}$ **do**

    for $i \in \{1,2,\cdots,I\}$ **do** /* for each rule */

      $\mu_A^{(i)}(x_n) \leftarrow$ Apply the ith rule for the set $\left\{\mu_{A_1}^{(i)}(x_n), \mu_{A_2}^{(i)}(x_n), \cdots, \mu_{A_K}^{(i)}(x_n)\right\}$

      $\mu_{A\to B}^{(i)}(x_n; y_n) \leftarrow \mu_A^{(i)}(x_n) \wedge \mu_B^{(i)}(y_n)$ /* The output fuzzy set of the ith rule (Modus Ponens). */

    $\hat{y} \leftarrow \text{centroid}\left(\mu_{A\to B}^{(1)}(x_n; y_n) \vee \mu_{A\to B}^{(2)}(x_n; y_n) \vee \cdots \vee \mu_{A\to B}^{(I)}\right)$ /* aggregation method and defuzzification */

The Takagi-Sugeno model basically follows the same approach described in the Section III., that is, it uses a dictionary[1] (a collection of fuzzy sets), a set of rules, and a inference method. However, instead of defining an output fuzzy set and defuzzifying it to obtain the crisp value, the output of the nth-order Takagi-Sugeno model is given by

$$y_n = \frac{\sum_{i=1}^{K} w_i f_i(x_n)}{\sum_{i=1}^{K} w_i}, \tag{16}$$

where

$$w_i = \mu_{A_1}^{(i)}(x_n) \wedge \mu_{A_2}^{(i)}(x_n) \wedge \cdots \wedge \mu_{A_K}^{(i)}(x_n) \tag{17}$$

is the result of the ith rule and $f_i(x_n)$ is the nth-order polynomial function. For this problem, the following set ot rules are defined:

IF x_n IS VERY LOW, THEN y_n IS HIGH

IF x_n IS LOW, THEN y_n IS MEDIUM

IF x_n IS MEDIUM, THEN y_n IS VERY HIGH

IF x_n IS HIGH, THEN y_n IS LOW.

Based on this set of rules and observing the Equation (17), it is easy to see that w_i is equal to the output of the membership function since there is only one input fuzzy set per rule. These input fuzzy sets (VERY LOW, LOW, MEDIUM, and HIGH) are shown in the Figure 7.

[1] Also called database.

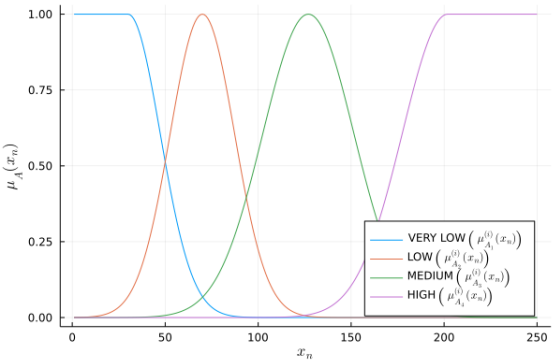


Fig. 7. Input fuzzy set of the Takagi-Sugeno model.

For a zero-order Takagi-Sugeno FIS, $f_i(x_n)$ is simply a constant. These values are given by

$$f_i(x_n) = \begin{cases} 1, \text{ if } i = 1 \\ 1, \text{ if } i = 2 \\ 1, \text{ if } i = 3 \\ 1, \text{ if } i = 4 \end{cases} \tag{18}$$

REFERENCES

[1] S. M. Kay, *Fundamentals of statistical signal processing: estimation theory*, Prentice-Hall, Inc., 1993.

[2] P. S. Diniz, *et al.*, *Adaptive filtering*, vol. 4, Springer, 1997.

[3] A. Leon-Garcia, *Probability and random processes for electrical engineering*, Pearson Education India, 1994.

[4] G. Strang, G. Strang, G. Strang, G. Strang, *Introduction to linear algebra*, vol. 3, Wellesley-Cambridge Press Wellesley, MA, 1993.

[5] T. Takagi, M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control", *IEEE transactions on systems, man, and cybernetics*, , no. 1, pp. 116–132, 1985.