Neural Comput & Applic (2017) 28:2017–2028
DOI 10.1007/s00521-015-2169-4

ORIGINAL ARTICLE

Interpolation neural network model of a manufactured wind turbine

José de Jesús Rubio[1]

Received: 17 August 2015 / Accepted: 21 December 2015 / Published online: 13 January 2016

Abstract In this paper, an interpolation neural network is introduced for the learning of a wind turbine behavior with incomplete data. The proposed hybrid method is the combination of an interpolation algorithm and a neural network. The interpolation algorithm is applied to estimate the missing data of all the variables; later, the neural network is employed to learn the output behavior. The proposed method avoids the requirement to know all the system data. Experiments show the effectiveness of the proposed technique.

Keywords Neural networks · Interpolation · Hybrid techniques · Wind turbine · Incomplete data

1 Introduction

The hybrid systems have been widely used in the learning of incomplete data for the applications of nonlinear modeling [21, 28], prediction [27], pattern recognition [33], classification [23, 25], control, fault detection, and diagnosis in industrial systems [7, 13], visual inspection [15], and cascaded systems [3].

There are many studies about hybrid systems for the learning of nonlinear behaviors. Despite the proposals, few researches have been carried out in the past to perform the learning of incomplete data.

✉ José de Jesús Rubio
jrubioa@ipn.mx; rubio.josedejesus@gmail.com

[1] Sección de Estudios de Posgrado e Investigación, ESIME Azcapotzalco, Instituto Politécnico Nacional, Av. de las Granjas no. 682, Col. Santa Catarina, 02250 Mexico, D.F., Mexico

On the other hand, there are other methods for the learning of nonlinear behaviors with incomplete data, but they use noise signals considering the design as a stochastic problem, it would be interesting to consider the design as a deterministic problem.

In this research, a hybrid algorithm as the combination of the stable neural network and interpolation algorithm is introduced for the learning of nonlinear systems with incomplete data where the design is considered as a deterministic problem. It consists in the following two stages.

First, the interpolation algorithm is used to obtain the missing data of all the variables in some nonlinear behavior. Figure 1 shows that the interpolation algorithm is applied to build the estimation of the variables denoted as $\widehat{x}_l(k)$ when only some points of the real variables denoted as $x_{lr}(k)$ are available.

Second, after the interpolation algorithm obtains the estimation of the variables, Fig. 2 shows that the interpolation neural network is employed to learn the output nonlinear behavior where the variables estimated by the interpolation algorithm denoted as $\widehat{x}_1(k) = \widehat{z}_1(k)$, $\widehat{x}_2(k) = \widehat{z}_2(k)$,..., $\widehat{x}_n(k) = \widehat{z}_n(k)$, $\widehat{x}_{n+1}(k) = \widehat{y}(k)$ are used instead of the real variables denoted as $x_{1,r}(k) = z_1(k)$, $x_{2,r}(k) = z_2(k)$, ..., $x_{n,r}(k) = z_n(k)$, $x_{n+1,r}(k) = y_r(k)$. $\widehat{y}(k)$ is the target output of the neural network. The inputs and output of the neural network are $\widehat{z}_1(k)$, $\widehat{z}_2(k)$,...,$\widehat{z}_n(k)$ and NN(k), respectively. The importance of the neural network is that while the interpolation algorithm only estimates the variables of the nonlinear behavior, the neural network learns the output behavior.

In remainder of this section, there will be the survey of related works. Finally, the organization of this paper will be mentioned.

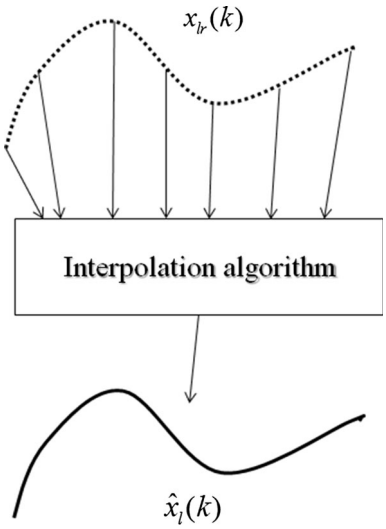

Fig. 1 Interpolation algorithm to estimate all the variables with incomplete data

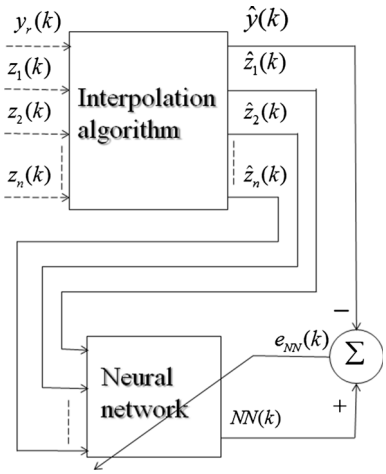

Fig. 2 Interpolation neural network for the learning

1.1 Related works

This subsection contains a survey of two kind of related works: (a) hybrid systems for the learning of nonlinear behaviors and (b) methods for the learning of behaviors with incomplete data.

There is some research about the learning with hybrid systems. In [1], a learning approach to train uninorm-based hybrid neural networks is suggested. In [2], four semi-supervised learning methods are discussed. A specific ensemble strategy is developed in [4]. In [5], an approach to the construction of classifiers from imbalanced datasets is described. A dynamic pattern recognition method is proposed in [9]. In [10] and [19], the use of evolving classifiers for the activity recognition is described. Hybrid and ensemble methods in machine learning are focused in [11]. In [12], a granular neural network framework for the evolving fuzzy system modeling is introduced. A novel hybrid active learning strategy is proposed in [14]. In [16], an enhanced version of the evolving participatory learning approach is developed. A class of hybrid fuzzy models is designed in [18]. A parsimonious network based on the fuzzy inference system is addressed in [20]. In [21], a novel dynamic parsimonious fuzzy neural network is considered. A holistic concept of a fully data-driven modeling tool is proposed in [22]. In [23], a novel evolving fuzzy-rule-based classifier is proposed. A novel meta-cognitive-based scaffolding classifier is considered in [24] . In [25], a novel interval type-2 fuzzy classifier is introduced. An evolving hybrid fuzzy neural network-based modeling approach is introduced in [26].

Otherwise, there is some research about the learning of nonlinear behaviors with incomplete data. In [6], kernel regression method is used for the modeling with incomplete data. The story of incomplete and redundant representation modeling is introduced in [8]. In [32], the authors propose a new model called sparse hidden Markov model. A novel sparse shape composition model is considered in [35]. In [36], a method is introduced for regression and classification problems.

1.2 Organization of the paper

The paper is structured as follows. In Sect. 2, the interpolation neural network is described. In Sect. 3, the interpolation neural network is employed for the modeling of two trajectories of the wind turbine behavior. Finally, in Sect. 4, the conclusion and future research are detailed.

2 Interpolation neural network

This section is divided in two subsection which consider the two stages of the proposed algorithm. (a) the interpolation algorithm is utilized to estimate the nonlinear behavior of all the variables with incomplete data. (b) The interpolation neural network is employed for the learning of the nonlinear behavior output with incomplete data.

2.1 Interpolation algorithm to estimate the incomplete data

The interpolation algorithm is described in this subsection as the first part of the proposed model. The algorithm proposed in this part is used to estimate the missing data of all the variables with incomplete data, i.e., the proposed algorithm is a multi-dimension approximator where all the variables are independently estimated.

2.1.1 Description of the interpolation algorithm

Consider the functions $x_{lr}(k) = f(k_l) \in \Re$ with $l = 1, 2, , \ldots, n+1$ is the number of variables estimated with this algorithm, $k_l = 1, 2, \ldots, T$, T are the iterations number for the variables, $x_{lr}(k)$ are the output real data of the nonlinear behaviors. The approximation consists to find $\widehat{x}_l(k)$ such that they estimate the real variables with incomplete data $x_{lr}(k)$.

The slopes of $x_{lr}(k)$ denoted as $m_l(k)$ using the k_l and $x_{lr}(k)$ data of the nonlinear behavior are obtained as follows:

$$m_l(k) = \frac{x_{lr}(k) - x_{lr}(k-1)}{(k_l) - (k_l - 1)} \tag{1}$$

The nonlinear behaviors are divided in N_l intervals, each interval is generated by considering the following inequality:

$$|(|m_l(k)| - |m_l(k-1)|)| \geq h_l \tag{2}$$

where h_l is a small selected threshold parameter, consider that the signals taken from k_l for each of the N_l intervals are denoted by j. Figure 3 shows the approximation of the nonlinear behaviors using the interpolation algorithm.

The Eq. (3) describes the approximation of the nonlinear behaviors using the proposed interpolation algorithm [30]:

$$\widehat{x}_l(k) = (1 - \lambda_l(k)) \cdot x_{l,i,j}(k) + \lambda_l(k) \cdot x_{l,f,j}(k) \tag{3}$$

where $x_{l,i,j}(k)$ are the initial values of $x_{lr}(k)$ in the interval j, $x_{l,f,j}(k)$ are the final values of $x_{lr}(k)$ in the interval j, k_l are the variant iterations inside of the interval j, $\lambda_l(k)$ are the variant-in-time parameters of the interval j, $\lambda_l(k)$ are given as follows:

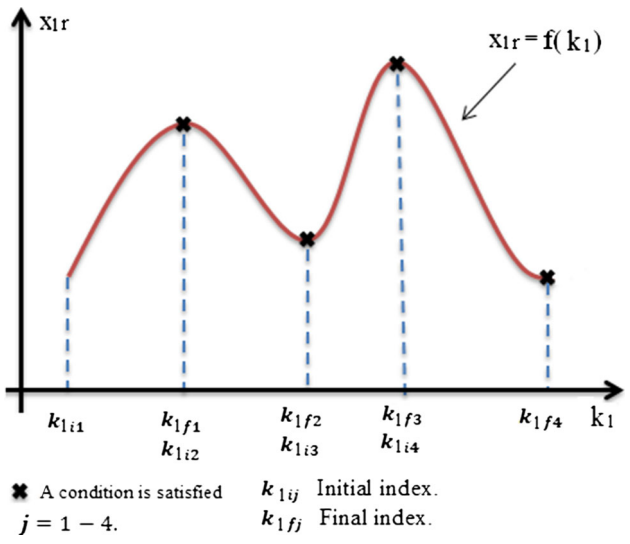

Fig. 3 Interpolation algorithm

$$\lambda_l(k) = \frac{k_l - k_{l,i,j}}{k_{l,f,j} - k_{l,i,j}} \tag{4}$$

where $k_{l,i,j}$ are the initial values of $\lambda_l(k)$ in the interval j, and $k_{l,f,j}$ are the final values of $\lambda_l(k)$ in the interval j.

It is known that $k_{l,i,j} \leq k_l \leq k_{l,f,j}$ for each interval j; consequently, $0 \leq \lambda_l(k) \leq 1$, and $\lambda_l(k)$ always increases. The variant parameters $\lambda_l(k)$ are important in the proposed interpolation algorithm because $\widehat{x}_l(k)$ are the approximations of $x_{lr}(k)$ from the initial points to the final points for each interval j. The interpolation algorithm for the approximation of nonlinear behaviors is as follows:

1. Obtain the slope of $x_{lr}(k)$ denoted as $m_l(k)$ using the k_l and $x_{lr}(k)$ data of the nonlinear behaviors using the Eq. (1), and select the threshold parameters h_l.
2. Obtain the elements number in the intervals N_l with Eq. (2).
3. The intervals are denoted by j.
4. For each interval j, obtain $\lambda_l(k)$ with Eq. (4).
5. For each interval j, obtain $\widehat{x}_l(k)$ as the approximations of $x_{lr}(k)$ using Eq. (3).

2.1.2 Boundedness of the interpolation algorithm

In this section, the variables of the interpolation algorithm will be guaranteed to be bounded. Substituting (4) into (3) of the interpolation algorithm gives:

$$\widehat{x}_l(k) = \left(1 - \frac{k_l - k_{l,i,j}}{k_{l,f,j} - k_{l,i,j}}\right) \cdot x_{l,i,j}(k) + \frac{k_l - k_{l,i,j}}{k_{l,f,j} - k_{l,i,j}} \cdot x_{l,f,j}(k) \tag{5}$$

(5) can be rewritten as follows:

$$\widehat{x}_l(k) = x_{l,i,j}(k) + \frac{k_l - k_{l,i,j}}{k_{l,f,j} - k_{l,i,j}} \left(x_{l,f,j}(k) - x_{l,i,j}(k)\right) \tag{6}$$

Theorem 1 *The outputs $\widehat{x}_l(k)$ of the interpolation algorithm (3)–(4), (6), are guaranteed to be bounded by $x_{l,i,j}(k)$ and by $x_{l,f,j}(k)$ for all the intervals j.*

Proof The proof is given by two parts. (a) If $x_{l,i,j}(k) \leq x_{l,f,j}(k)$, then $x_{l,f,j}(k) - x_{l,i,j}(k) \geq 0$, using (6), and the fact $k_{l,i,j} \leq k_l \leq k_{l,f,j}$, it gives:

$$\begin{aligned} & x_{l,i,j}(k) + \frac{k_{l,i,j} - k_{l,i,j}}{k_{l,f,j} - k_{l,i,j}} \left(x_{l,f,j}(k) - x_{l,i,j}(k)\right) \\ & \leq x_{l,i,j}(k) + \frac{k_l - k_{l,i,j}}{k_{l,f,j} - k_{l,i,j}} \left(x_{l,f,j}(k) - x_{l,i,j}(k)\right) \\ & \leq x_{l,i,j}(k) + \frac{k_{l,f,j} - k_{l,i,j}}{k_{l,f,j} - k_{l,i,j}} \left(x_{l,f,j}(k) - x_{l,i,j}(k)\right) \end{aligned} \tag{7}$$

Inequality (7) gives $x_{l,i,j}(k) \leq \widehat{x}_l(k) \leq x_{l,f,j}(k)$. (b) If $x_{l,f,j}(k) \leq x_{l,i,j}(k)$, then $x_{l,f,j}(k) - x_{l,i,j}(k) \leq 0$, using (6), and the fact $k_{l,i,j} \leq k_l \leq k_{l,f,j}$, it gives:

$$x_{l,i,j}(k)+\frac{k_{l,f,j}-k_{l,i,j}}{k_{l,f,j}-k_{l,i,j}}\left(x_{l,f,j}(k)-x_{l,i,j}(k)\right)$$
$$\leq x_{l,i,j}(k)+\frac{k_{l}-k_{l,i,j}}{k_{l,f,j}-k_{l,i,j}}\left(x_{l,f,j}(k)-x_{l,i,j}(k)\right)$$
$$\leq x_{l,i,j}(k)+\frac{k_{l,i,j}-k_{l,i,j}}{k_{l,f,j}-k_{l,i,j}}\left(x_{l,f,j}(k)-x_{l,i,j}(k)\right) \quad (8)$$

Inequality (8) gives $x_{l,f,j}(k)\leq\widehat{x}_{l}(k)\leq x_{l,i,j}(k)$. Since (a) If $x_{l,i,j}(k)\leq x_{l,f,j}(k)$, then $x_{l,i,j}(k)\leq\widehat{x}_{l}(k)\leq x_{l,f,j}(k)$, and (b) If $x_{l,f,j}(k)\leq x_{l,i,j}(k)$, then $x_{l,f,j}(k)\leq\widehat{x}_{l}(k)\leq x_{l,i,j}(k)$, in all the cases $\widehat{x}_{l}(k)$ are bounded by $x_{l,i,j}(k)$ and by $x_{l,f,j}(k)$ as the Theorem claims. □

Remark 1 There are three differences between the interpolation algorithm introduced by [30] and that considered in this study. The first difference is that in [30], the interval number is obtained by the changes in the slopes sign, while in this study the interval number is determined by Eq. (2). The second difference is that in [30], the interpolation algorithm is applied only to estimate the nonlinear system output, while in this work the interpolation algorithm is used to estimate all the nonlinear system variables. The third difference is that in [30], the interpolation algorithm is considered alone, while in this research the interpolation algorithm is combined with a stable neural network.

2.2 Neural network to learn the nonlinear behavior with incomplete data

The neural network is described in this subsection as the second part of the proposed model. This subsection describes the algorithm proposed in this study for the modeling of a nonlinear behavior with incomplete data.

2.2.1 Description of the neural network

In this study, incomplete data are considered; consequently, the neural network of this paper is used to learn the nonlinear behavior using only the variables estimated with the interpolation model, not the real data variables, that is, the variables of the interpolation algorithm $\widehat{z}_1(k)$, $\widehat{z}_2(k)$,...,$\widehat{z}_n(k)$, $\widehat{y}(k)$, are used instead of the real variables with incomplete data $z_1(k)$, $z_2(k),\ldots,z_n(k)$, $y_r(k)$.

The stable backpropagation algorithm is employed with a new time-varying rate to guarantee its uniformly stability for online identification and its identification error converges to a small zone bounded by the uncertainty. The weights error is bounded by the initial weights error, i.e., overfitting and local optimum are eliminated in the mentioned algorithm [27, 28].

Stable backpropagation algorithm is as follows [27, 28]:

1. Obtain the output of the nonlinear system $y(k)$. Note that the nonlinear system may have the structure represented by Eq. (9); the parameter n is selected according to this nonlinear system.

$$\widehat{y}(k)=f[Z(k)] \quad (9)$$

where $Z(k)=[\widehat{z}_1(k)\ldots,\widehat{z}_i(k),\ldots,\widehat{z}_n(k)]^T\in\Re^{n\times 1}$ is the input vector, f is an unknown nonlinear function, $f\in C^{\infty}$, and $\widehat{y}(k)$, $\widehat{z}_1(k)$, $\widehat{z}_2(k)$,...,$\widehat{z}_n(k)$ are the outputs of the interpolation algorithm.

2. Select the following parameters; $V(1)$ and $W(1)$ as random numbers between 0 and 1; m as an integer number, and α_0 as a positive value smaller or equal to 1; obtain the output of the neural network NN(1) with Eq. (10). The interpolation neural network which learns the real output with incomplete data of the nonlinear behavior $y_r(k)$ is as follows:

$$\mathrm{NN}(k)=V(k)\Phi(k)=\sum_{j=1}^{m}V_j(k)\phi_j(k)$$
$$\Phi_k=\left[\phi_1(k),\ldots,\phi_j(k),\ldots,\phi_m(k)\right]^T$$
$$\phi_j(k)=\tanh\left(\sum_{i=1}^{n}W_{ij}(k)\widehat{z}_i(k)\right) \quad (10)$$

where $\widehat{z}_1(k)$, $\widehat{z}_2(k),\ldots,\widehat{z}_n(k)$ are the inputs estimation with the interpolation algorithm, $V_j(k+1)$ and $W_{ij}(k+1)$ are the weights of the hidden and output layer, respectively. m is the neuron number in the hidden layer. ϕ_j is the hyperbolic tangent function.

3. For each iteration k, obtain the output of the neural network NN(k) with Eq. (10), obtain the neural network error $e_{\mathrm{NN}}(k)$ with Eq. (11), and update the parameters $V_j(k+1)$ and $W_{ij}(k+1)$ with Eq. (12).

$$e_{\mathrm{NN}}(k)=\mathrm{NN}(k)-\widehat{y}(k) \quad (11)$$

$$V_j(k+1)=V_j(k)-\alpha(k)\phi_j(k)e_{\mathrm{NN}}(k)$$
$$W_{ij}(k+1)=W_{ij}(k)-\alpha(k)\sigma_{ij}(k)e_{\mathrm{NN}}(k) \quad (12)$$

where the new time-varying rate $\alpha(k)$ is:

$$\alpha(k)=\frac{\alpha_0}{2\left(\frac{1}{2}+\sum_{j=1}^{m}\phi_j^2(k)+\sum_{j=1}^{m}\sum_{i=1}^{n}\sigma_{ij}^2(k)\right)}$$

where $i=1,\ldots,n$, $j=1,\ldots,m$, $\sigma_{ij}(k)=V_j(k)\mathrm{sech}^2\left(\sum_{i=1}^{n}W_{ij}(k)z_i(k)\right)\widehat{z}_i(k)\in\Re$, α_0 is the constant learning speed, $\widehat{y}(k)$ is the output estimation with the interpolation algorithm, NN(k) is the output of the interpolation neural network, and $\widehat{z}_1(k)$, $\widehat{z}_2(k)$,...,$\widehat{z}_n(k)$, $\widehat{y}(k)$ are the outputs of the interpolation algorithm.

Remark 2 The hyperbolic tangent is used as the activation function in the proposed neural network because it considers positive and negative values, being it more complete than others as the sigmoid function which only considers positive values.

2.2.2 Stability analysis of the neural network

The following theorem guarantees that the interpolation neural network can approximate a nonlinear behavior.

Theorem 2 ([34]) *Suppose that the input universe of discourse U is a compact set in $\Re^n$. Then, for any given real continuous function $\sigma(k)$ on U, and arbitrary $\in > 0$, there exists an interpolation neural network* $\mathrm{NN}(k)$ *in the form* (10) *such that*:

$$\sup_{x\in U}|\mathrm{NN}(k) - \widehat{y}(k)| < \in \tag{13}$$

That is, the neural network $\mathrm{NN}(k)$ *is an approximator of the output of the interpolation algorithm* $\widehat{y}(k)$.

Proof See [34] for the proof. □

The following theorem gives the stability of the neural network model.

Theorem 3 *The interpolation neural network* (10), (11), *and* (12) *applied for the identification of the nonlinear system* (9) *is uniformly stable and the upper bound of the average identification error $e_p^2(k)$ satisfies*:

$$\limsup_{T\to\infty}\frac{1}{T}\sum_{k=2}^{T}e_p^2(k) \le \alpha_0\overline{\mu}^2 \tag{14}$$

where $e_p^2(k) = \frac{\alpha(k-1)}{2}e^2(k-1)$, $0<\alpha_0\le 1\in\Re$ and $0<\alpha(k)\in\Re$ are defined in (12), $e(k)$ is defined in (11), $\mu(k) = y(k) - \sum_{j=1}^{M}V_j^*\phi_j^*$ is an uncertainty, $\overline{\mu}$ is the upper bound of the uncertainty $\mu(k)$, $|\mu(k)|<\overline{\mu}$, $\phi_j^* = \tanh(\sum_{i=1}^{N}W_{ij}^*x_i(k))$, V_j^* and W_{ij}^* are unknown weights such that the uncertainty $\mu(k)$ is minimized.

Proof See [27, 28] for the proof. □

The following theorem proves that the weights of the interpolation neural network are bounded.

Theorem 4 *When the average error $e_p^2(k)$ is bigger than the uncertainty $\alpha_0\overline{\mu}^2$, the weights error is bounded by the initial weights error as follows:*

$$\begin{aligned} e_p^2(k+1) &\ge \alpha_0\overline{\mu}^2 \\ &\Longrightarrow \sum_{j=1}^{M}\widetilde{V}_j^2(k+1) + \sum_{j=1}^{M}\sum_{i=1}^{N}\widetilde{W}_{ij}^2(k+1) \\ &\le \sum_{j=1}^{M}\widetilde{V}_j^2(1) + \sum_{j=1}^{M}\sum_{i=1}^{N}\widetilde{W}_{ij}^2(1) \end{aligned} \tag{15}$$

where $i=1,\ldots,N$, $j=1,\ldots,M$, $\widetilde{V}_j(k)$ and $\widetilde{W}_{ij}(k)$ is the weights error, $\widetilde{V}_j(1)$ and $\widetilde{W}_{ij}(1)$ is the initial weights error, $e_p^2(k+1) = \frac{\alpha(k)}{2}e^2(k)$, $V_j(k+1)$, $W_{ij}(k+1)$, $0<\alpha_0\le 1\in\Re$, and $0<\alpha(k)\in\Re$ are defined in (12), *$e(k)$ is defined in* (11), *$\overline{\mu}$ is the upper bound of the uncertainty $\mu(k)$, $|\mu(k)|<\overline{\mu}$.*

Proof See [27, 28] for the proof. □

Remark 3 There are two conditions for applying this algorithm for nonlinear systems: the first one is that the nonlinear system may have the form described by (9), and the second one is that the uncertainty $\mu(k)$ may be bounded.

Remark 4 The value of the parameter $\overline{\mu}$ used for the stability of the algorithm is unimportant, because this parameter is not used in the algorithm. The bound of $\mu(k)$ is needed to guarantee the stability of the algorithm, but it is not used in the backpropagation algorithm (10), (11), (12).

Remark 5 There is one important difference between the stable neural network of [27, 28] and the considered in this study. It is that in [27, 28], the stable neural network is alone used for the learning of short data, while, in this research, the stable neural network is combined with the interpolation algorithm for the learning of nonlinear systems with incomplete data.

Remark 6 The fuzzy slopes model of [29] has two differences with the interpolation neural network of this research: (1) the fuzzy slopes model uses a fuzzy inference system, while the interpolation neural network employs the stable neural network, obtaining an advantage in the proposed method because a stable algorithm guarantees that all the variables will remain bounded; (2) the fuzzy slopes model only considers the output with incomplete data, while the interpolation neural network considers all the variables with incomplete data, obtaining an advantage in the introduced technique because it is a generalization of the previous.

3 Experimental results

The interpolation neural network is compared with the fuzzy slopes model of [29] for the learning of the wind turbine behavior with incomplete data. The objective is that the interpolation neural network output NN of (1)–(4), (10)–(12) must be nearer with the real output of the wind turbine y_r than the fuzzy slopes model output.

Figure 4 shows the prototype of the manufactured wind turbine with a rotatory tower which is considered for this

Fig. 4 Prototype of the manufactured wind turbine

Table 1 Parameters of the prototype

Parameter	Value	Parameter	Value
l_{c2}	0.5 m	R_e	30 Ω
m_2	0.5 kg	k_m	0.09 Wb
k_{b2}	1×10^{-6} kgm^2/s^2	k_{b1}	1×10^{-6} kgm^2/s^2
b_{b2}	1×10^{-1} kgm^{2}rad/s	b_{b1}	1×10^{-1} kgm^2 rad/s
k_2	0.45 Vs/rad	k_1	0.0045 Vs/rad
R_2	6.96 Ω	R_1	18 Ω
L_2	6.031×10^{-1}H	L_1	6.031×10^{-1}H
R	l_{c2} m	V_ω	5 m/s
ρ	1.225 kg/m^3	β	0.5 rad
g	9.81 m/s^2		

study. This prototype has three blades with a rotatory tower which does not use a gear box. Important research about wind turbines is presented in [7, 31], and [33]. Table 1 shows the parameters of the prototype. The parameters m_2 and l_{c2} are obtained from the wind turbine blades. The parameters R_1, L_1, and k_1, are obtained from the tower motor. The parameters k_2, R_2, R_e, and L_2, are obtained from the wind turbine generator. The parameters R, ρ, V_ω, and β, are obtained from [31].

1×10^{-5} are considered as the initial conditions for the plant states $x_1 = i_2$, $x_2 = \theta_2$, $x_3 = \dot{\theta}_2$, $x_4 = i_1$, $x_5 = \theta_1$, and $x_6 = \dot{\theta}_1$. u_1 is the force of the air received by the three blades in kgm^2 rad/s^2, u_2 is the motor armature voltage in V, θ_1 is the angular position of the tower motor in rad, θ_2 is the angular position of a wind turbine blade in rad, i_1 is the motor armature current of the tower in A, i_2 is the generator armature current in A, and y is the output voltage generated by the wind turbine in V. An electronic circuit and a microcontroller board of Arduino are used to digitalize and to send the obtained signals to a personal computer. Figure 5 shows the electronic circuit of the acquisition system. Figure 6 shows one program designed to save the real data of the electric current, electric voltage, blades position, and tower position using the Mathlab software. Figure 7 shows the real electronic circuit to save the real data of the electric voltage, electric current, blades position, and tower position.

The interpolation neural network learns the behavior considering real data of the inputs and states of the wind turbine behavior, the eight inputs for the nonlinear behavior are denoted as $z_1(k) = u_{1r}$, $z_2(k) = u_{2r}$, $z_3(k) = x_{1r}$, $z_4(k) = x_{2r}$, $z_5(k) = x_{3r}$, $z_6(k) = x_{4r}$, $z_7(k) = x_{5r}$, and $z_8(k) = x_{6r}$, and the target output is denoted as $\widehat{y}(k) = y$. The root mean square error is used for the comparison results [17, 28, 30]:

$$\text{RMSE} = \left(\frac{1}{T} \sum_{k=1}^{T} e^2(k) \right)^{\frac{1}{2}} \tag{16}$$

where T is the iterations number, and $e(k) = e_{FS}(k)$ is the error of the fuzzy slopes model, or $e(k) = e_{\text{NN}}(k)$ is the error of the interpolation neural network of (11).

3.1 Experiment 1

Experiment 1 considers the first movement of the wind turbine described as follows: (1) from 0 to 2 s, both inputs are fed; consequently, the tower moves far of the maximum air intake, the generator current is decreased, and the wind turbine blades stop moving; (2) from 2 to 4 s, both inputs are not fed; consequently, current is not generated, and both the tower and wind turbine blades do no move; (3) from 4 to 6 s, both inputs are fed, but the air intake is positive and tower voltage is negative; consequently, the tower returns to the maximum air intake, the generator current is increased, and the wind turbine blades move; (4) from 6 to 8 s, both inputs are not fed; consequently, current is not generated, and the tower and wind turbine blades do no move. The described behavior is repeated three times for the learning and once for the testing; consequently, 8412 data are used for the training and 2804 data are used for the testing.

The fuzzy slopes model is used with parameters $n = 8$, $m = 4$, $v_i(1) = \text{rand}$, $c_{ij}(1) = \text{rand}$, $\sigma_{ij}(1) = 10\text{rand}$, $h = 1 \times 10^{-7}$, rand is a random number between 0 and 1.

The interpolation neural network of (1)–(4), (10)–(12) is used with parameters $n = 8$, $m = 4$, $\alpha_0 = 0.5$, $V_j(1) = \text{rand}$, $W_{ij}(1) = \text{rand}$, $h = 1 \times 10^{-7}$, rand is a random number between 0 and 1.

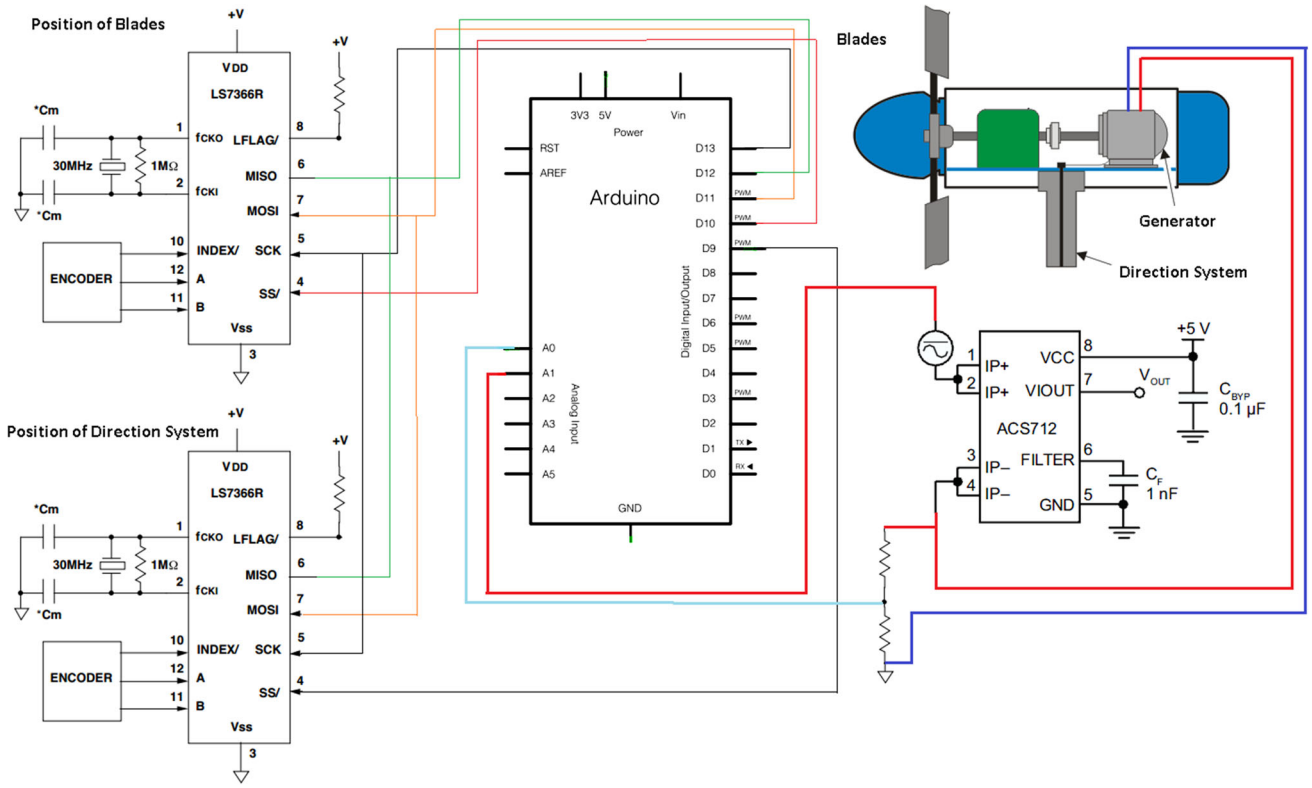

Fig. 5 Electronic circuit of the acquisition system

Figure 8 shows the incomplete data for the states of the wind turbine behavior. Figure 9 shows the modeling of the wind turbine behavior using the fuzzy slopes model and interpolation neural network for the training. Figure 10 shows the modeling of the wind turbine behavior using the fuzzy slopes model and interpolation neural network for the testing. Table 2 shows the root mean square error for the fuzzy slopes model and interpolation neural network.

The iterations number is shown instead of the time in seconds to guarantee that in this research incomplete data are employed. From Fig. 9 and Table 2, it is shown that the interpolation neural network is the best for the training of the wind turbine behavior because the RMSE of the above algorithm is the smallest one. The training could be used for online designs such as the control, prediction, or fault detection. From Fig. 10 and Table 2, it is shown that the interpolation neural network is the best for the testing of the wind turbine behavior because the RMSE of the above algorithm is the smallest one. The testing could be used for offline designs such as the pattern recognition or classification.

3.2 Experiment 2

Experiment 2 considers the second movement of the wind turbine described as follows: (1) from 0 to 2 s, the input air is fed and the tower input is not fed; consequently, the tower remains in the maximum air intake, the generator current is maximum, and the wind turbine blades have motion; (2) from 2 to 4 s, the air is not fed and the tower input is fed; consequently, current is not generated, the tower moves far of the maximum air intake, and the wind turbine blades do not have motion; (3) from 4 s to 6 s, the air is fed and the tower input is not fed; consequently, the tower does not move, the generator current is minimum, and the wind turbine blades almost do not move; (4) from 6 s to 8 s, the air is not fed and the tower input is fed with a negative voltage; consequently, current is not generated, the tower returns to the maximum air intake, and the wind turbine blades do not have motion. The described behavior is repeated three times for the learning and once for the testing; consequently, 8412 data are used for the training and 2804 data are used for the testing.

The fuzzy slopes model is used with parameters $n = 8$, $m = 4$, $v_i(1) = \text{rand}$, $c_{ij}(1) = \text{rand}$, $\sigma_{ij}(1) = 10\text{rand}$, $h = 1 \times 10^{-7}$, rand is a random number between 0 and 1.

The interpolation neural network of (1)–(4), (10)–(12) is used with parameters $n = 8$, $m = 4$, $\alpha_0 = 0.5$, $V_j(1) = \text{rand}$, $W_{ij}(1) = \text{rand}$, $h = 5 \times 10^{-8}$, rand is a random number between 0 and 1.

Figure 11 shows the incomplete data for the states of the wind turbine behavior. Figure 12 shows the modeling of

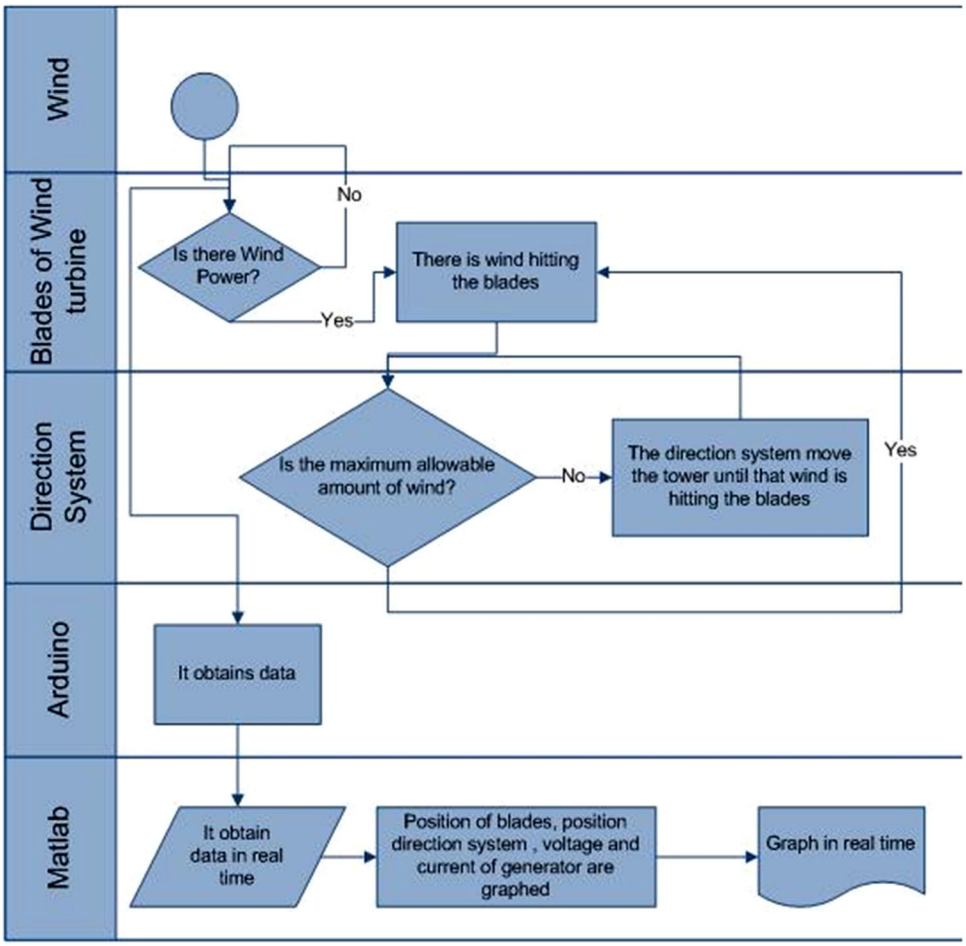

Fig. 6 Program to save the real data

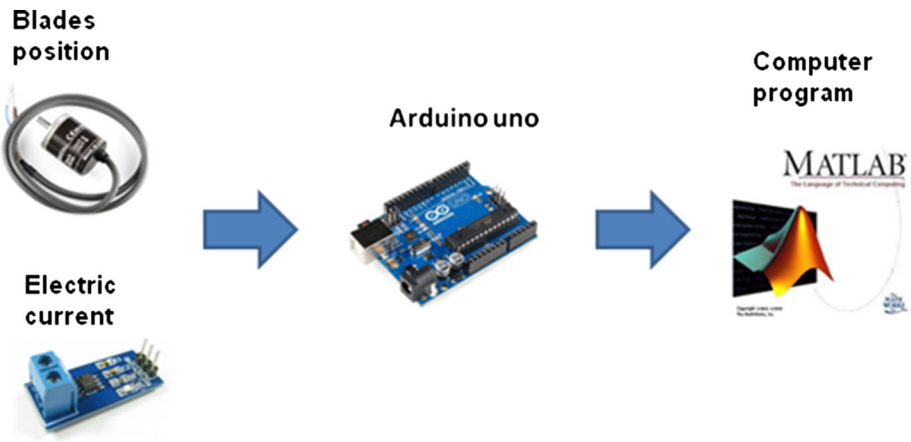

Fig. 7 Electronic circuit to save the real data

the wind turbine behavior using the fuzzy slopes model and interpolation neural network for the training. Figure 13 shows the modeling of the wind turbine behavior using the fuzzy slopes model and interpolation neural network for the testing. Table 3 shows the root mean square error for the fuzzy slopes model and interpolation neural network.

The iterations number is shown instead of the time in seconds to guarantee that in this research incomplete data are employed. From Fig. 12 and Table 3, it is shown that the interpolation neural network is the best for the training of the wind turbine behavior because the RMSE of the above algorithm is the smallest one. The training could be

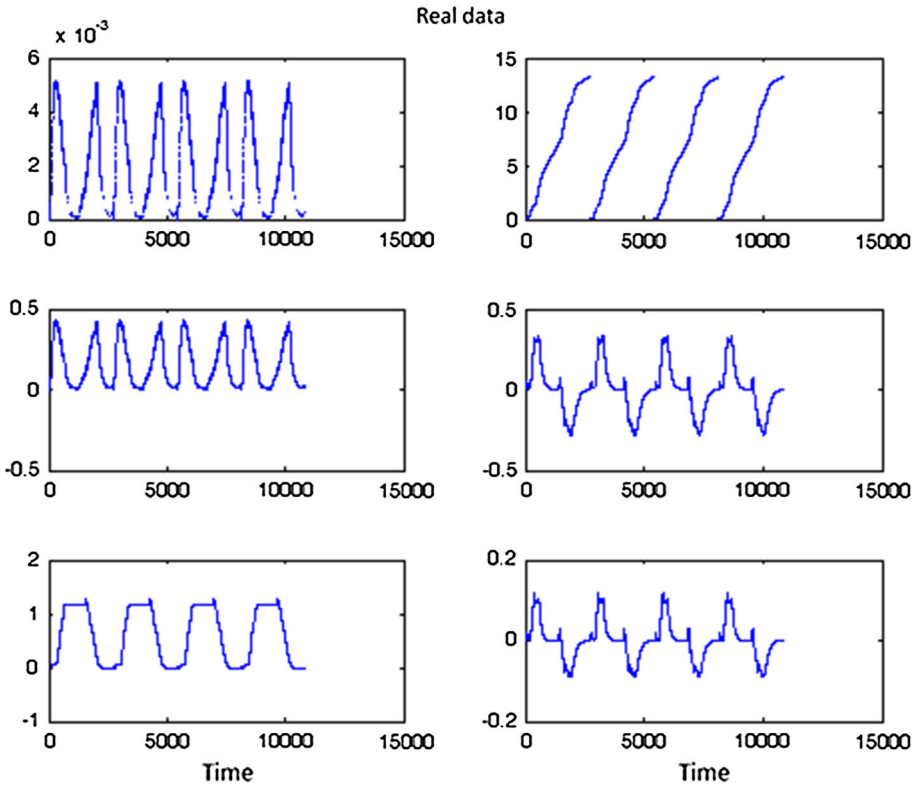

Fig. 8 Incomplete data for the experiment 1

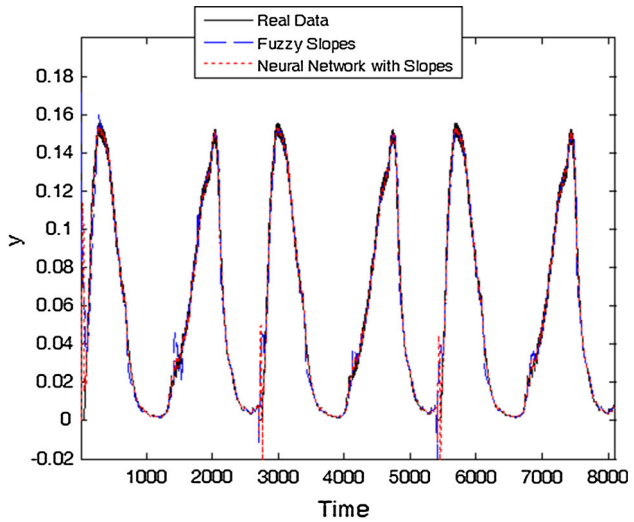

Fig. 9 Modeling for the training of the experiment 1

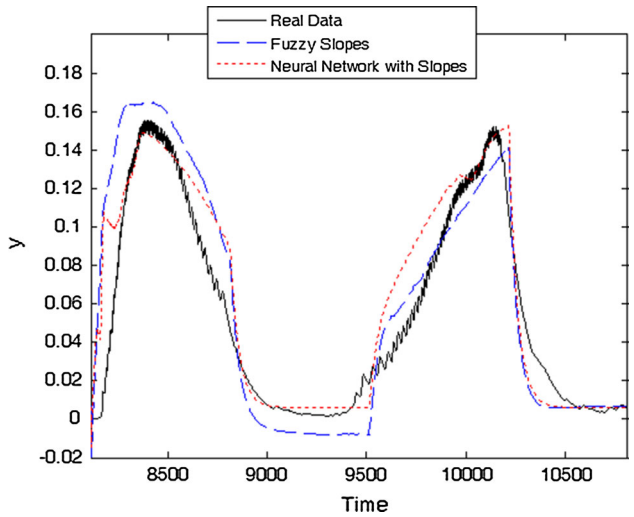

Fig. 10 Modeling for the testing of the experiment 1

used for online designs such as the control, prediction, or fault detection. From Fig. 13 and Table 3, it is shown that the interpolation neural network is the best for the testing of the wind turbine behavior because the RMSE of the above algorithm is the smallest one. The testing could be used for offline designs such as the pattern recognition or classification.

Table 2 Comparison of the errors

	RMSE for training	RMSE for testing
Fuzzy slopes model	0.0065	0.0086
Interpolation neural network	0.0049	0.0071

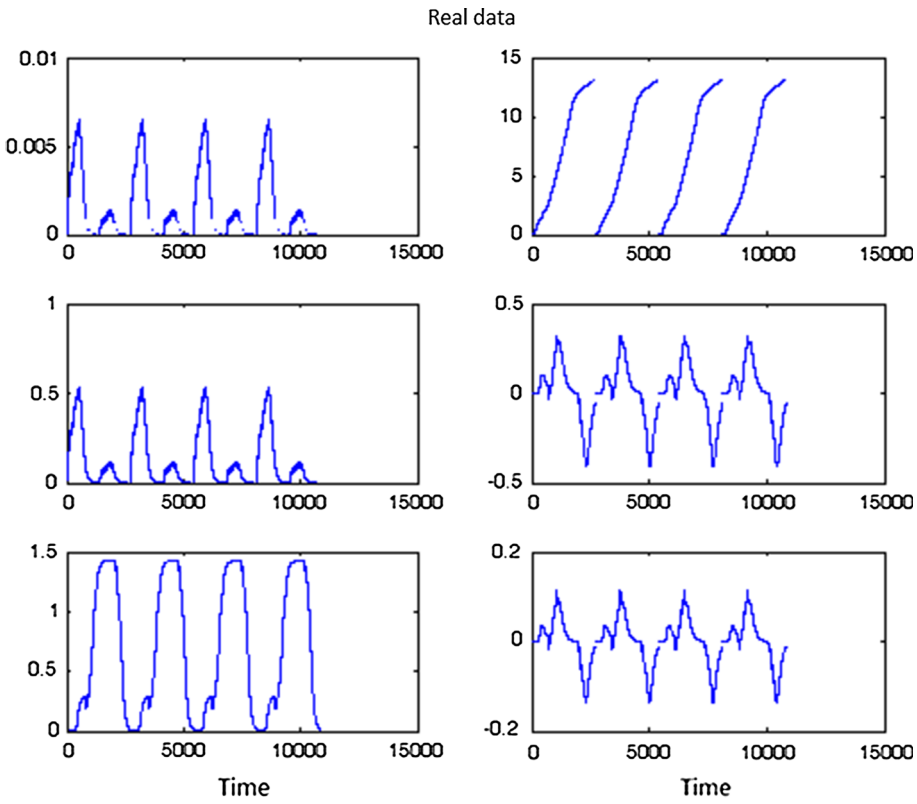

Fig. 11 Incomplete data for the experiment 2

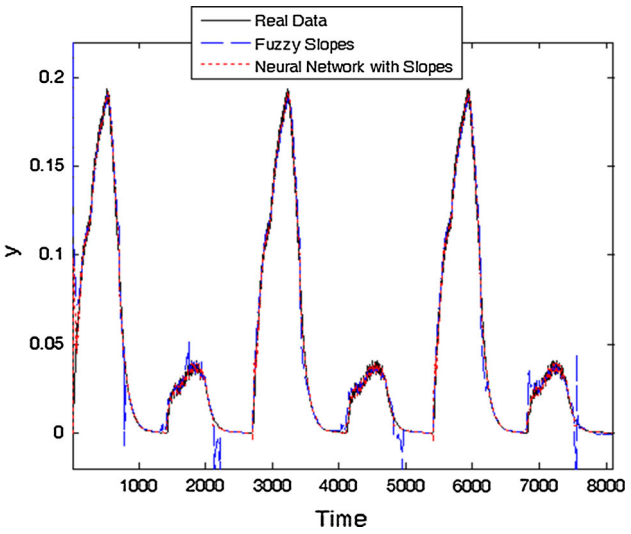

Fig. 12 Modeling for the training of the experiment 2

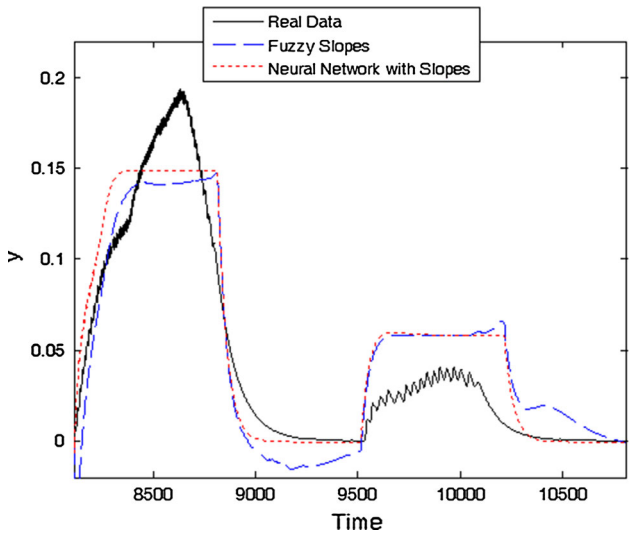

Fig. 13 Modeling for the testing of the experiment 2

Remark 7 Choosing an appropriate number of hidden neurons is important in the behavior, because too many neurons result in a complex system that may be unnecessary for the problem and it can cause overfitting [28], whereas too few neurons produce a less powerful system that may be

Table 3 Comparison of the errors

	RMSE for training	RMSE for testing
Fuzzy slopes model	0.0065	0.0086
Interpolation neural network	0.0035	0.0077

insufficient to achieve the objective. The number of hidden neurons is considered as a design parameter and it is determined based on the trial-error method.

4 Conclusion

In this paper, the interpolation neural network was introduced. The interpolation algorithm was applied to build an estimation of the nonlinear behaviors when only some points of the real behavior with incomplete data were available. After the interpolation algorithm obtained the estimation of the nonlinear behaviors, the neural network was employed to learn the output nonlinear behavior considering only the outputs of the interpolation model instead of the real data inputs and output. The importance of the neural network is that while the interpolation algorithm only estimates the nonlinear behaviors, the neural network learns the output behavior. The proposed interpolation neural network was compared with a fuzzy slopes model for the modeling of the wind turbine behavior, giving that the first algorithm provides higher accuracy compared to the other. The proposed technique could be used in control, prediction, pattern recognition, classification, or fault detection. As a future research, the proposed strategy will be used for the control design.

Acknowledgments The author is grateful with the editor and with the reviewers for their valuable comments and insightful suggestions, which can help to improve this research significantly. The author thanks the Secretaría de Investigación y Posgrado, Comisión de Operación y Fomento de Actividades Académicas, and Consejo Nacional de Ciencia y Tecnología for their help in this research.

References

1. Bordignon F, Gomide F (2014) Uninorm based evolving neural networks and approximation capabilities. Neurocomputing 127:13–20
2. Bouchachia A (2005) Learning with hybrid data. In: Proceedings of the fifth international conference on hybrid intelligent systems, pp 1–6
3. Bouchachia A (2010) An evolving classification cascade with self-learning. Evol Syst 1(3):143–160
4. Cernuda C, Lughofer E, Hintenaus P, Marzinger W, Reischer T, Pawliczek M, Kasberger J (2013) Hybrid adaptive calibration methods and ensemble strategy for prediction of cloud point in melamine resin production. Chemometr Intell Lab Syst 126:60–75
5. Chawla NV, Bowyer KW, Hall LO, Kegelmayr WP (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
6. Cruz-Vega I, Yu W (2010) Multiple fuzzy neural networks modeling with sparse data. Neurocomputing 73:2446–2453
7. Duviella E, Serir L, Sayed-Mouchaweh M (2013) An evolving classification approach for fault diagnosis and prognosis of a wind farm, conference on control and fault-tolerant systems (SysTol), pp 377–382
8. Elad M (2012) Sparse and redundant representation modeling-what next? IEEE Signal Process Lett 19(12):922–928
9. Hartert L, Sayed-Mouchaweh M (2014) Dynamic supervised classification method for online monitoring in non-stationary environments. Neurocomputing 126:118–131
10. Iglesias JA, Tiemblo A, Ledezma A, Sanchis A (2015) Web news mining in an evolving framework. Information fusion. doi:10.1016/j.inffus.2015.07.004
11. Kazienko P, Lughofer E, Trawinski B (2013) Hybrid and ensemble methods in machine learning J.UCS special issue. J Univers Comput Sci 19(4):457–461
12. Leite D, Costa P, Gomide F (2013) Evolving granular neural networks from fuzzy data streams. Neural Netw 38:1–16
13. Lemos A, Caminhas W, Gomide F (2013) Adaptive fault detection and diagnosis using an evolving fuzzy classifier. Inf Sci 220:64–85
14. Lughofer E (2012) Hybrid active learning for reducing the annotation effort of operators in classification systems. Pattern Recognit 45:884–896
15. Lughofer E, Weigl E, Heidl W, Eitzinger C, Radauer T (2015) Integrating new classes on the fly in evolving fuzzy classifier designs and its application in visual inspection. Appl Soft Comput 35:558–582
16. Maciel L, Gomide F, Ballini R (2014) Enhanced evolving participatory learning fuzzy modeling: an application for asset returns volatility forecasting. Evol Syst 5:75–88
17. Marques Silva A, Caminhas W, Lemos A, Gomide F (2014) A fast learning algorithm for evolving neo-fuzzy neuron. Appl Soft Comput 14(B):194–209
18. Nuñez A, De Schutter B, Saez D, Skrjanc I (2014) Hybrid-fuzzy modeling and identification. Appl Soft Comput 17:67–78
19. Ordoñez FJ, Iglesias JA, de Toledo P, Ledezma A, Sanchis A (2013) Online activity recognition using evolving classifiers. Expert Syst Appl 40:1248–1255
20. Pratama M, Anavatti SG, Angelov PP, Lughofer E (2014) PANFIS: a novel incremental learning machine. IEEE Trans Neural Netw Learn Syst 25(1):55–68
21. Pratama M, Er MJ, Li X, Oentaryo RJ, Lughofer E, Arifin I (2013) Data driven modeling based on dynamic parsimonious fuzzy neural network. Neurocomputing 110:18–28
22. Pratama M, Anavatti SG, Lughofer E (2014) GENEFIS: toward an effective localist network. IEEE Trans Fuzzy Syst 22(3):547–562
23. Pratama M, Anavatti SG, Er MJ, Lughofer ED (2015) pClass: an effective classifier for streaming examples. IEEE Trans Fuzzy Syst 23(2):369–386
24. Pratama M, Anavatti SG, Lu J (2015) Recurrent classifier based on an incremental meta-cognitive-based scaffolding algorithm. IEEE Trans Fuzzy Syst. doi:10.1109/TFUZZ.2015.2402683
25. Pratama M, Lu J, Zhang G (2015) Evolving type-2 fuzzy classifier. IEEE Trans Fuzzy Syst. doi:10.1109/TFUZZ.2015.2463732
26. Rosa R, Gomide F, Ballini R (2013) Evolving hybrid neural fuzzy network for system modeling and time series forecasting, 12th international conference on machine learning and applications, pp 1–6
27. Rubio JJ, Angelov P, Pacheco J (2011) An uniformly stable backpropagation algorithm to train a feedforward neural network. IEEE Trans Neural Netw 22(3):356–366
28. Rubio JJ (2014) Evolving intelligent algorithms for the modelling of brain and eye signals. Appl Soft Comput 14(B):259–268
29. Rubio JJ (2015) Fuzzy slopes model of nonlinear systems with sparse data. Soft Comput. doi:10.1007/s00500-014-1289-6

30. Rubio JJ, Vazquez DM, Mujica-Vargas D (2013) Acquisition system and approximation of brain signals. IET Sci Meas Technol 7(4):232–239
31. Rubio JJ, Soriano LA, Yu W (2014) Dynamic model of a wind turbine for the electric energy generation. Math Probl Eng 2014:1–8
32. Tao L, Elhamifar E, Khudanpur S, Hager GD, Vidal R (1012) Sparse hidden markov models for surgical gesture classification and skill evaluation. Lecture notes in artificial intelligence, pp 167–177
33. Toubakh H, Sayed-mouchaweh M, Duviella E (2013) Advanced pattern recognition approach for fault diagnosis of wind turbines. 12th international conference on machine learning and applications, pp 368–373
34. Wang LX (1997) A course in fuzzy systems and control. ISBN: 0-13-540882-2
35. Zhang S, Zhan Y, Dewan M, Huang J, Metaxas DN, Zhou XS (2012) Towards robust and effective shape modeling: sparse shape composition. Med Image Anal 16:265–277
36. Zhong LW, Kwok JT (2012) Efficient sparse modeling with automatic feature grouping. IEEE Trans Neural Netw Learn Syst 23(9):1436–1447