Engineering Applications of Artificial Intelligence 28 (2014) 52–63

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Fault detection and isolation for PEM fuel cell stack with independent RBF model

M.M. Kamal [a], D.W. Yu [b], D.L. Yu [a,*]

[a] Control Group, School of Engineering, Liverpool John Moores University, Liverpool, UK
[b] Department of Automation, Northwest University at Qinhuangdao, Qinhuangdao, China

ARTICLE INFO

Article history:
Received 18 December 2012
Received in revised form
30 September 2013
Accepted 7 October 2013
Available online 2 December 2013

Keywords:
Fault detection
Fault isolation
PEMFC
RBF networks
Independent model

ABSTRACT

Neural networks have been successfully used to model nonlinear dynamic systems. However, when a static neural network model is used in system fault detection and the model prediction error is used as the residual, the residual is insensitive to the fault if the neural network used is in dependent mode. This paper proposed the use of a radial basis function network in independent mode as the system model for fault detection, and it was found that the residual is sensitive to the fault. To enhance the signal to noise ratio of the detection the recursive orthogonal least squares algorithm is employed to train the network weights. Another radial basis function network is used to isolate fault using the information in the residual signal. The developed method is applied to a benchmark simulation model of the proton exchange membrane fuel cell stacks developed at the Michigan University. One component fault, one actuator fault and three sensor faults were simulated on the benchmark model. The simulation results show that the developed approach is able to detect and isolate the faults to a fault size of ±10% of nominal values. These results are promising and indicate the potential of the method to be applied to the real world of fuel cell stacks for dynamic monitoring and reliable operations.

1. Introduction

If faults occurred in a process plant, they will affect the productivity, quality, safety and performance of the control systems of the plant. Therefore, early detection of possible faults would minimize the downtime, increase the safety of plant operations, prevent damages to the equipment, minimize the operation cost and also the maintenance. Fault detection and isolation (FDI) for dynamic control systems of nonlinear plants has become an important and challenging task in many engineering applications and continues to be an active area of research in the control environment (Hwang et al., 2010).

In process engineering, most plants have nonlinear dynamics in nature (Basseville, 1988) and may also be multivariable and complex, the observer-based FDI methods (Frank, 1990; Isermann, 1984; Patton, 1994) would therefore be very difficult to apply to these systems. Researchers have been tuned to some other alternatives and one of which is the artificial intelligent method. For example, Frank and Koppen-Seliger (1997) studied fuzzy logic and neural network applications for fault diagnosis. They used a dependent neural network for residual generation and fuzzy logic for residual evaluation. Simeón et al. (2010) used classical multivariable statistical techniques for FDI of several manufacturing and process plants. Ng and Srinivasan (2010) used the multi-agent method in which principle component analysis (PCA), self-organizing map (SOP) and Bayesian network were combined to do FDI for the Tennessee Eastman process and also distillation unit to classify the temperature change and disturbance during start-up. Bayesian network was also used as a classifier to do fault diagnosis in the Tennessee Eastman process by Verron et al. (2010), where a fault database was constructed. Three kinds of fault are analyzed in terms of unit step and random signal excitation. Subrahmanya and Shin (2013) used recurrent neural network (RNN) to distinguish faults occurring in actuator, component and sensors. In recent years, Polycarpou and Helmicki (1995) proposed a framework to estimate faults occurring in the system dynamics, in which an estimator was added to the state space model. The error between the model and the plant outputs was used to update the estimator that is used to estimate the fault. The RBF network has been used as the estimator in their work and the Levenberg–Marquart algorithm by Chong and Zak (2013) as an updating algorithm. This method has been widely recommended for further research, but its drawback is that the difficulty of developing an accurate nonlinear state space model for the plant makes it difficult for real applications.

Patton et al. (1994) proposed an approach for detecting and isolating faults in a nonlinear dynamic process using neural networks. Firstly, a multi-layer perceptron network was trained to

[*] Corresponding author.
E-mail address: D.Yu@ljmu.ac.uk (D.L. Yu).

0952-1976/$ - see front matter Crown Copyright © 2013 Published by Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2013.10.002

predict the future system states, then the residual was generated using the differences between the actual and predicted states. Secondly, another neural network was used as a classifier to isolate faults from these state prediction errors. However, this method used the neural network model in its so-called dependent mode (see definition given in Narendra and Parthasarathy (1990)), i.e. the past state of the process was used as a part of the network input. In this way, the network model was trained using the nominal process data for state prediction. Then, when a fault occurred, the fault would cause the process state to be contaminated (the value was affected). This contaminated process state was then fed into the network model, the predicted state would tend to the real process state, and consequently the residual would tend to zero (insensitive to the fault) when the model mismatch and noise effects were omitted. In addition to the above analysis, this point was also affirmed by Yu et al. (1999), where the simulation data clearly disclosed that the neural network model of the dependent mode generated residual that was insensitive to the fault. Besides, the method described by Patton et al. (1994) was not practical for most nonlinear systems as some of the states may not be measurable, while the design of nonlinear state observer is very difficult.

The insensitive residual generated by static network of dependent mode can be avoided where the process sensor faults should be treated was proposed by Yu et al. (1999). In their work, a semi-independent MLP network was used with the predicted output by the network model used to replace process output in the network input. Then, the model predicted output was reset by the real process output on every period of time to reduce the growing model prediction error. This was called semi-independent mode. The residual generated in this way was sensitive to the fault but the prediction error is bigger and the residual is not smooth, and a filer had to be used to smoothen the residual. A higher threshold had to be used due to the increased model prediction error and therefore some faults with small amplitude would then not be detected.

In the field of PEMFC, most approaches for fault detection used a model-based approach which involved the comparison of the observed behavior of the process to a reference model. In the aspect of hydrogen safety and efficiency for PEMFC, Ingimundarson et al. (2008) and Lebbal and Lecoeuche (2009) have developed a computer simulation tool which can be used to detect and monitor faults in the hydrogen stations. Xue et al. (2006) proposed a model-based condition monitoring scheme that employs the Hotelling T^2 statistical analysis for fault detection of PEMFC. This model-based robust condition monitoring scheme can deal with the operating condition variation, various uncertainty in a fuel cell system and measurement noise.

FDI for the PEMFC systems is challenging due to its nonlinear nature. Thus, a method needs to be developed which can tackle the above problems in a simple and effective way. This is the motive of this paper. The novelty of this work lies in using the independent RBF network to model the fuel cell stacks, and generating the residual. By acquiring process data under different disturbances and with or without faults, the network model can be trained to make the residual of FDI monitoring system more sensitive to the faults and more robust to the disturbance. To enhance the model accuracy while reducing the false alarm rate, the recursive orthogonal least squares (ROLS) algorithm is employed to train the RBF model. For fault isolation, another RBF network is used to classify the different features of different faults on the residual vector. By setting appropriate number of hidden layer nodes, the clearness of the isolation can be maximized. The Michigan benchmark model is used as the benchmark to evaluate the proposed method with and without faults occurring in the process. The Michigan model has been modified to introduce one component fault, one actuator fault and three sensor faults. Simulation results approved the effectiveness of the method for detection and isolation of the faults with the fault size as small as ± 10% of their nominal values. The rest of paper is arranged as follows: Section 2 presents the dynamics of the PEMFC systems. The RBF neural network model is presented in Section 3. These are followed by fault detection in Section 4 and fault isolation in Section 5. Finally, conclusions are discussed in Section 6.

2. Proton exchange membrane fuel cell dynamics and faults

A fuel cell consists of two electrodes; a negative electrode (anode) and a positive electrode (cathode) separated by an electrolyte. Fuel cells convert the chemical energy of the hydrogen fuel (on the anode side) into electric energy while through a chemical reaction with oxygen (on the cathode side) produce water and heat as end product. Hydrogen atoms separate into protons and electrons once the chemical reaction happens. The electrons go through the load which contains a flow of electricity while the protons migrate through the electrolyte to the cathode side, where they reunite with oxygen to produce water and heat as shown in Fig. 1. To maintain the desired air supply, the air supply needs to replenish the air to maintain the oxygen partial pressure. The air supply system consists of an air compressor, an electric motor and pipes or manifolds between the components. The compressor not only achieves desired air flow but also increases air pressure which significantly improves the reaction at the membranes, and thus the overall efficiency and power density (Pukrushpan et al., 2004a).

2.1. Compressor model

The flow and temperature out of the compressor (W_{cp} and T_{cp}) depend on the compressor rotational speed ω_{cp}. A lumped rotational model is used to represent the dynamic behavior of the compressor (Pukrushpan et al., 2004a):

$$J_{cp}\frac{d\omega_{cp}(t)}{dt}=\tau_{cm}(t)-\tau_{cp}(t) \tag{1}$$

where $\tau_{cm}(v_{cm},\omega_{cp})$ is the compressor motor (CM) torque and τ_{cp} is the load torque. The compressor motor torque is calculated using a static motor equation:

$$\tau_{cm}=\eta_{cm}\frac{k_t}{R_{cm}}[V_{cm}(t)-k_v\omega_{cp}(t)] \tag{2}$$

where k_t, R_{cm} and k_v are motor constants and η_{cm} is the motor mechanical efficiency. The torque required to drive the compressor is calculated using the thermodynamic equation.

$$\tau_{cp}(t)=\frac{c_pT_{atm}(t)}{\eta_{cp}\omega_{cp}(t)}\left[\left(\frac{p_{sm}(t)}{p_{atm}}\right)^{(\gamma-1)/\gamma}-1\right]W_{cp}(t) \tag{3}$$

where γ is the ratio of the specific heats of air (=1.4), c_p is the constant pressure specific heat capacity of air (=1004 J kg^{-1} K^{-1}), η_{cp} is the motor compressor efficiency, p_{sm} is the pressure inside

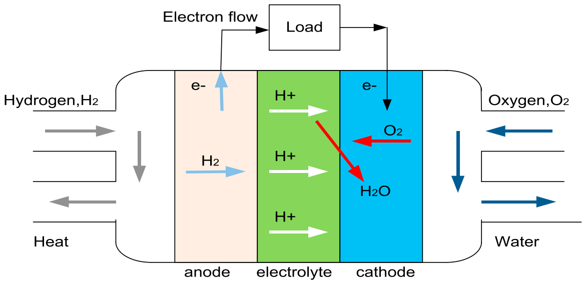

Fig. 1. PEMFC chemical reaction.

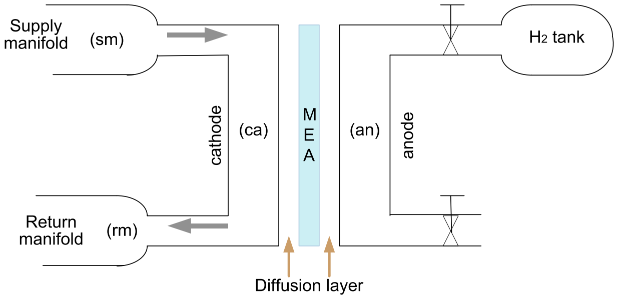

Fig. 2. The fuel cell supply system.

the supply manifold and p_{atm} and T_{atm} are the atmospheric pressure and temperature, respectively.

2.2. Supply manifold model

The cathode supply manifold (sm) includes pipe and stack manifold volumes between the compressor and the fuel cells as shown in Fig. 2. The supply manifold pressure, p_{sm}, is governed by mass continuity and energy conservation equations (Pukrushpan et al., 2004b):

$$\frac{dm_{sm}(t)}{dt} = W_{cp}(t) - W_{sm,out}(t) \tag{4}$$

$$\frac{dp_{sm}(t)}{dt} = \frac{\gamma R}{M_a^{atm} V_{sm}}[W_{cp}(t)T_{cp}(t) - W_{sm,out}(t)T_{sm}(t)] \tag{5}$$

where R is the universal gas constant, γ is the ratio of a specific heat capacities of air, M_a^{atm} is the molar mass of atmospheric air at ϕ_{atm}, V_{sm} is the manifold volume and $T_{sm} = p_{sm}V_{sm}M_a^{atm}/Rm_{sm}$ is the supply manifold gas temperature.

3. Radial basis function neural network modeling

3.1. RBF network structure

The nonlinear system can be modeled by the multivariable NARX model of the following form,

$$y(k) = f[y(k-1), \cdots, y(k-n_y), u(k-1-d), \cdots, u(k-n_u-d)] + e(k) \tag{6}$$

where $u \in \Re^m$, y, $e \in \Re^p$ are the process input, output and noise vectors respectively with m and p being the number of inputs and outputs, n_y and n_u are the maximum lags in the outputs and inputs respectively, d is a dead-time vector representing delayed time to different control variables, $f(*)$ is a vector-valued nonlinear function. Suppose that the RBF network model precisely model the system; the model can then be represented by

$$\hat{y}(k) = f[\hat{y}(k-1), \cdots, \hat{y}(k-n_y), u(k-1-d), \cdots, u(k-n_u-d)] + e(k) \tag{7}$$

A RBF neural network is a feed-forward network consisting of three layers which are the input layer, hidden layer and output layer. Each hidden node contains a center c_j, which is a cluster center on the input vector space, and calculates the euclidean distance between the center and the network input vector x defined by $\|x(t) - c_j(t)\|$ with x given as

$$x(k) = [y(k-1), \cdots, y(k-n_y), u(k-1-d), \cdots, u(k-n_u-d)] \tag{8}$$

then the output of the hidden layer node is a nonlinear function of the euclidean distance. In this work the Gaussian function is chosen as the nonlinear function.

$$\varphi_i = \exp\left(-\frac{\|x-c_i\|^2}{\sigma_i{}^2}\right), \quad i = 1, \ldots, n_h \tag{9}$$

where $c_i, \rho, x \in \Re^{n_h}$ are the center vector, width vector and input vector respectively. The network output is then sum of the weighted output of all hidden nodes and bias. The input–output mapping for the RBF networks (Yu et al., 1999) can then be described as

$$\hat{y} = \phi^T W \tag{10}$$

where $\hat{y} \in \Re^p$, $W \in \Re^{p \times n_h}$, $\phi \in \Re^{n_h}$, are the estimated output vector, weight matrix and hidden layer output vector respectively. The RBF network is chosen to model the fuel cell stack because it is capable of approximating any continuous function to a pre-defined precision provided it has enough hidden layer nodes (Yu et al., 1999). Also, the training process for the RBF network model is much faster compared with the back-propagation training because it uses the least squares algorithm.

3.2. The training algorithm

Training an RBF network is optimizing parameters including the hidden layer centers and the widths in the Gaussian functions and network weights, to achieve minimum model prediction error. In this work, the network centers are selected using the K-means clustering algorithm, so that the sum squared distance of each input data from the center of the data group, to which it belongs, is minimized. The width of the Gaussian functions are chosen using the ρ-nearest center algorithm (Yu et al., 1999), to achieve proper sampling of any input data by a few near centers. The weights between the hidden layer and the output are trained using the ROLS as it is a numerically robust algorithm. The detailed description of the three algorithms for the RBF network training can be found in Wang et al. (2006).

3.2.1. Recursive K-means algorithm

The s are set by the K-means clustering method whose objective is to minimize the sum squared distances from each input data to its closest center so that the data are adequately covered by the activation functions $\Phi_j(t)$. The K-means clustering method proceeds as follows:

(a) Randomly choose some input data to be the initial centers. The number of the centers is equal to the number of hidden nodes, n_h.

(b) Let $k(x)$ denote the index of the best-matching center for the input vector x. Find $k(x)$ at iteration t by minimizing the sum squared distances:

$$k(x) = \arg\min\left[\frac{1}{2}\sum_{k=1}^{nh}(x(t) - c_k(t))^2\right] \tag{11}$$

where $c_k(t)$ is the center of the kth activation function at iteration t.

(c) Update the centers of the activation functions by using the following rule:

$$c_k(t+1) = \begin{cases} c_k(t) + \alpha[x(t) - c_k(t)] & \text{if} \quad k = k(x) \\ c_k(t) & \text{otherwise} \end{cases} \tag{12}$$

(d) Increment t by 1 and go back to step 2. Continue the algorithm until no noticeable changes are observed in the centers c_k.

3.2.2. ρ-Nearest neighbours method

The widths are computed by the ρ-nearest neighbors method. The excitation of each node should overlap with other nodes (usually closest) so that a smooth interpolation surface between nodes is obtained. In this method, the widths for each hidden node are set as the average distance from the center to the ρ-nearest s as given by

$$\sigma_i = \frac{1}{P}\sum_{d=1}^{P} \| c_i(t) - c_d(t) \| \qquad i = 1, \dots, n_h \tag{13}$$

c_j is the ρ-nearest neighbor of c_i. For the non-linear function, the value of ρ depends on the type of problem encountered.

3.2.3. The recursive orthogonal least squares algorithm

Training of the RBF network weights with the ROLS algorithm is as follows. Considering (10) at sample interval k for a set of N samples of input–output training data from $k-N+1$ to k, in other words a window going back in time N samples, we have

$$Y(k) = \hat{Y}(k) + E(k) = \Phi(k)W(k) + E(k) \tag{14}$$

where $Y \in \Re^{N \times p}$ is the desired output matrix, $\hat{Y} \in \Re^{N \times p}$ is the neural network output matrix, $\Phi \in \Re^{N \times n_h}$ is the hidden layer output matrix, and $E \in \Re^{N \times p}$ is the error matrix. Eq. (14) can be solved for $W(k)$ using the recursive MIMO least squares algorithm to minimize the following time-varying cost function,

$$J(k) = \left\| \begin{bmatrix} \sqrt{\lambda}Y(k-1) \\ - - - - \\ y^T(k) \end{bmatrix} - \begin{bmatrix} \sqrt{\lambda}\Phi(k-1) \\ - - - - \\ \phi^T(k) \end{bmatrix} W(k) \right\|_F \tag{15}$$

where the F-norm of a matrix is defined as $\|A\|_F^2 = \mathrm{trace}(A^T A)$ and $\lambda < 1$ is used to introduce exponential forgetting to the past data. It has been shown in Gomm and Yu (2000) that minimizing (15) is equivalent to minimizing the following cost function,

$$J(k) = \left\| \begin{bmatrix} \sqrt{\lambda}\widehat{Y}(k-1) \\ - - - - \\ y^T(k) \end{bmatrix} - \begin{bmatrix} \sqrt{\lambda}R(k-1) \\ - - - - \\ \phi^T(k) \end{bmatrix} W(k) \right\|_F \tag{16}$$

where R is an $n_h \times n_h$ upper triangular matrix, and $\widehat{Y}$ is computed by an orthogonal decomposition as follows,

$$\begin{bmatrix} \sqrt{\lambda}R(k-1) \\ - - - - \\ \phi^T(k) \end{bmatrix} = Q(k) \begin{bmatrix} R(k) \\ - - - \\ 0 \end{bmatrix} \begin{bmatrix} \widehat{Y}(k) \\ - - - \\ \eta^T(k) \end{bmatrix} = Q^T(k) \begin{bmatrix} \sqrt{\lambda}\widehat{Y}(k-1) \\ - - - - \\ y^T(k) \end{bmatrix}, \tag{17}$$

where Q is an orthogonal matrix. Combining (15) and (16) and considering that the F-norm is preserved by orthogonal transformation, the following equivalent cost function is obtained,

$$J(k) = \left\| \begin{bmatrix} \widehat{Y}(k) - R(k)W(k) \\ - - - - - - - - - \\ \eta^T(k) \end{bmatrix} \right\|_F \tag{18}$$

which allows the optimal solution of $W(k)$ to be solved straightforwardly from

$$R(k)W(k) = \widehat{Y}(k) \tag{19}$$

and leaves the residual at sample interval k as $\|\eta^T(k)\|_F$. Since $R(k)$ is an upper triangular matrix, $W(k)$ can be easily solved from (19) by backward substitution.

The decomposition in (17) can be achieved efficiently by applying given rotations to an augmented matrix to obtain the following transformation by Gomm and Yu (2000):

$$\begin{bmatrix} \sqrt{\lambda}R(k-1) & \sqrt{\lambda}\widehat{Y}(k-1) \\ \phi^T(k) & y^T(k) \end{bmatrix} \rightarrow \begin{bmatrix} R(k) & \widehat{Y}(k) \\ 0 & \eta^T(k) \end{bmatrix} \tag{20}$$

The procedure of the ROLS algorithm is therefore the following: for on-line training, calculate $\phi(k)$ at each sampling period to update the augmented matrix and compute the given rotations to realize the transformation in (20). Then solve $W(k)$ in (19) with $R(k)$ and $\widehat{Y}(k)$ obtained in (20). In this case, $W(k)$ is needed at each sample instant for prediction. Also, $\lambda < 1$ is needed to follow time-varying dynamics at the current time. For use in off-line mode, the given rotations can be computed to realize the transformation in (20) continuously to the end of training, then W is solved finally from (19). In this case, λ is set to 1. Initial values for $R(k)$ and $\widehat{Y}(k)$ in both cases can be assigned as $R(0) = \mu I$ and $\widehat{Y}(0) = 0$, where μ is a small positive number and I is a unity matrix with appropriate dimension.

3.3. Network modeling modes

There are two different modes of modeling a dynamic system using neural networks, one is the dependent mode and the other is the independent mode as defined by Narendra and Parthasarathy (1990). The structures of the two modes are displayed in Fig. 3 where Fig. 3(a) shows the block diagram of dependent mode (the process output $y(k)$ in the input vector $x(k)$ of the network in (7)) while in Fig. 3(b) shows the block diagram of independent mode (the model output $\hat{y}(k)$ in the input vector $x(k)$ of the network in (8)). From Fig. 3(a), it can be seen that in the dependant mode, the output of the plant is used as a part of network inputs. Therefore, the neural network model will be dependent on the plant and cannot run independently. When the dependent model runs alone, after predicting for one-step-ahead the plant output would not be available. Therefore, this mode of the model cannot do multi-step-ahead prediction and cannot run independently. The dependent mode is suitable to be trained for accurate one-step-ahead prediction, but one-step-ahead prediction is very limited for applications.

On the contrary, the past model output is used, instead of process output, as part of the model input in the independent

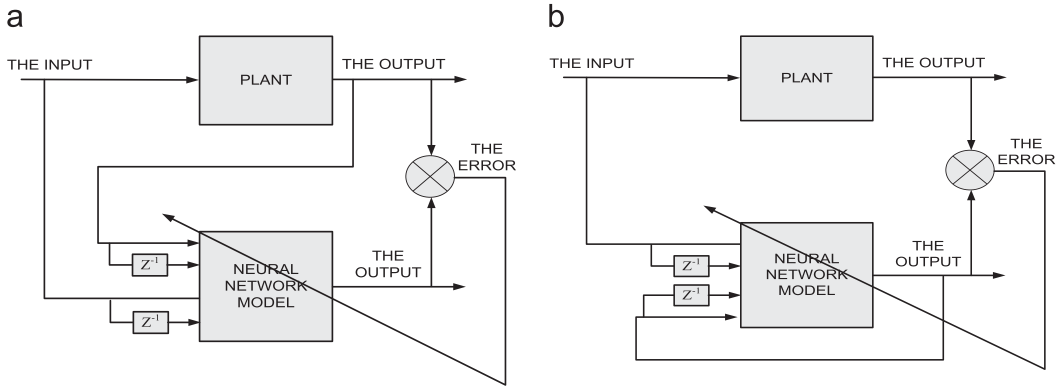

Fig. 3. The structure of the two modeling modes used in neural network modeling. (a) Dependent mode and (b) independent mode.

mode. Since the independent model can run independently of the process, it can be used to do multi-step-ahead prediction, and can also be used as a simulation model. The above features have been experienced by the authors in the past research (Yu et al., 1999). The model of the dependent mode can predict the process output for one-step-ahead only, while the independent model can predict for infinity steps as long as the input is available. When the two different modes of model are used for fault detection, the difference is significant. When a fault occurs to the plant and affects the plant output, the dependent model output will also be affected through the plant output being used as the model input; whilst the independent model will not be affected by the occurred fault as the model is independent from the plant. This can be clearly seen from the plant output and model prediction shown in Fig. 4(a) and (b).

Fig. 4 shows the results of the RBF network when faults occurred in the plant for stack voltage output. Here, a fault has been introduced at k=3000–3200. The same signal inputs have been used for both modes to compare the difference between these two. Fig. 4(a) shows the result of dependent mode when fault occurred at that particular times. As can be seen, when a fault occurs in the process, the process output is affected, at the same time due to the usage of the process out as the model input, the model prediction is also affected. Consequently, the error between the process and model output as the residual will not be sensitive to the occurrence of the fault. This is displayed in Fig. 4(a) with a fault occurring during sample instant k=3000–3200. The same output with the prediction by an independent model is displayed in Fig. 4(b). From the result, it is shown that when there is a fault in the plant, it will not affect the prediction of the independent network. Therefore, the modeling error of the independent model is very sensitive to the fault and the residual is big to indicate that a fault occurred in the plant.

3.4. The independent RBF network model

The structure of an independent RBF model under an open-loop condition for PEMFC dynamic systems proposed in this work can be referred to Fig. 5. Here, two inputs and three outputs of the process with their delayed values form the 13 inputs of the RBF model, while the three process outputs are the model outputs (Kamal and Yu, 2011). Such chosing of the input output orders are according to the training experience and by checking the process dynamics. The model prediction errors are generated for residual generation later.

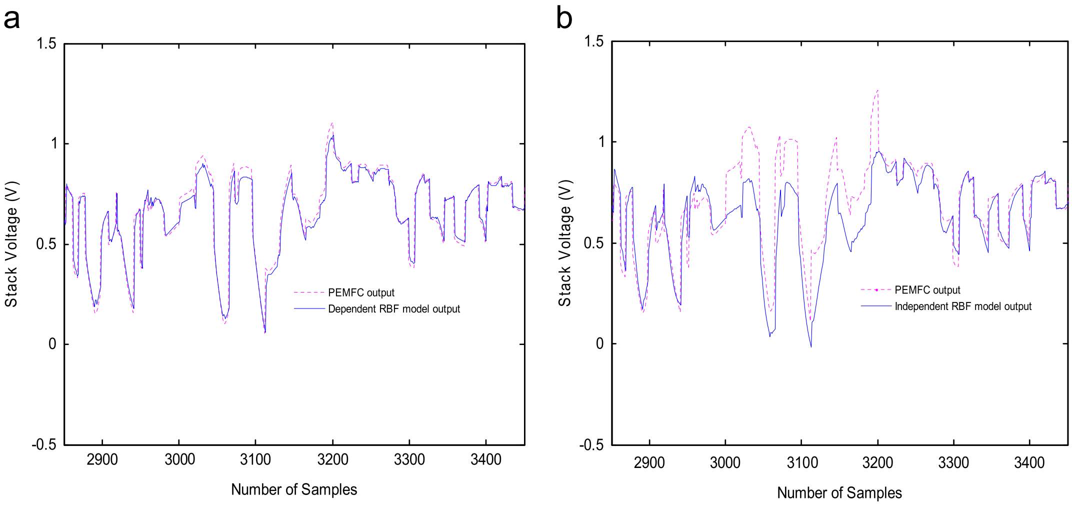

Fig. 4. Stack voltage predictions by (a) dependent mode and (b) independent mode.

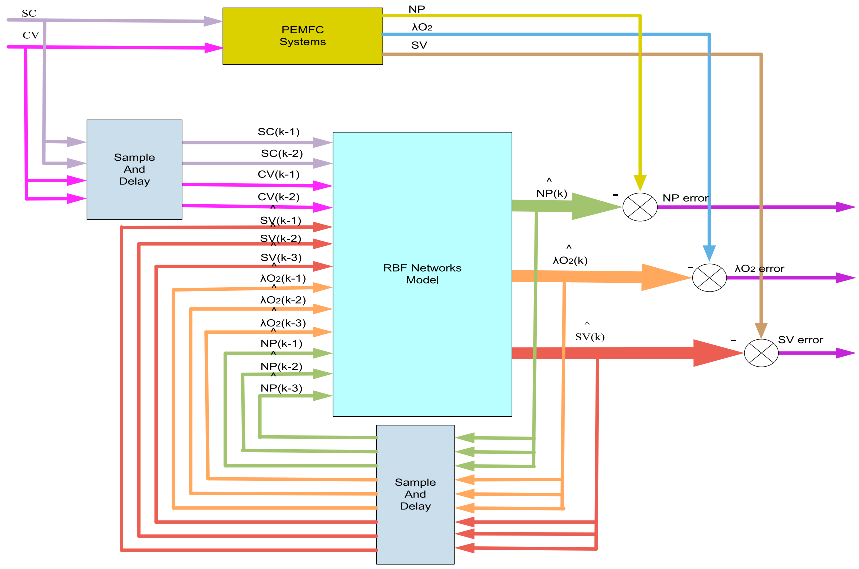

Fig. 5. The structure of an independent RBF network.

3.5. Data collection and scaling

At initial stage, a set of random amplitude signals (RAS) have been generated and used as the system inputs, the stack current SC and the compressor motor voltage CV. These signals ranging from 100 to 300 A for the stack current and from 100 to 235 V for the compressor motor voltage. Six thousand samples have been acquired when the sample period was set as 0.1 s considering the time constant of the main dynamics. Fig. 6 shows the example of RAS excitation signals of compressor voltage used as input in this work. The signals are generated randomly to cover the whole range of frequencies and entire operating space of amplitude in the PEMFC system. To ensure equal attention to all the variables, the input and output signals are normalized in the range of zero to one using the following linear normalization.

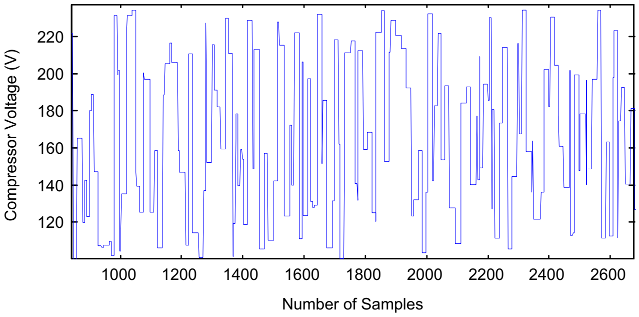

Fig. 6. The RAS excitation signals of compressor voltage.

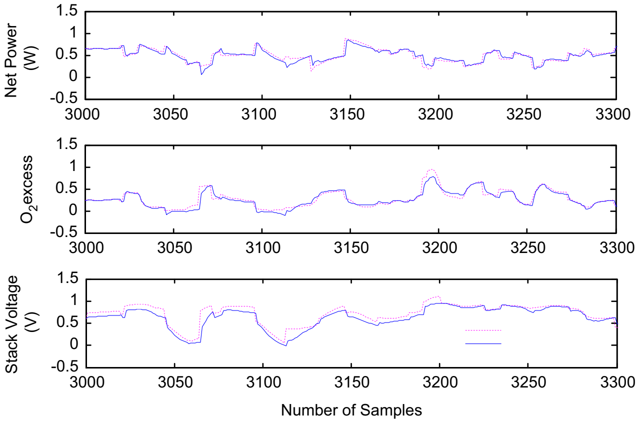

Fig. 7. The actual and estimated outputs of RBF model during testing.

$$u_{scale}(k) = \frac{u(k) - u_{min}}{u_{max} - u_{min}} \tag{21}$$

Training has been tried several times to determine the optimal number of hidden layer nodes. Finally using 22 hidden layer nodes was found to give minimal model prediction error in the measurement index of mean square error (MSE) defined in the following equation.

$$e_{mse} = \frac{1}{N}\sum_{j=1}^{N}[y(j) - \hat{y}(j)]^2 \tag{22}$$

where y and $\hat{y}$ are the PEMFC system output and the RBF model prediction respectively.

The first 2000 samples have been used to determine s using the K-means clustering method, and width using the ρ-nearest distance method. Then, the weights were trained using the RLS algorithm, where the input data to the RBF model was formed as shown in Fig. 5. As the past model output, rather than the plant output, are used as part of the model input in the independent modeling mode, the BLS method cannot be used while the RLS algorithm was used in this work. After training, the model was validated using 200 samples that have not been used in the training. The model prediction error for the normalized test data was also measured with MSE and their values for net power are

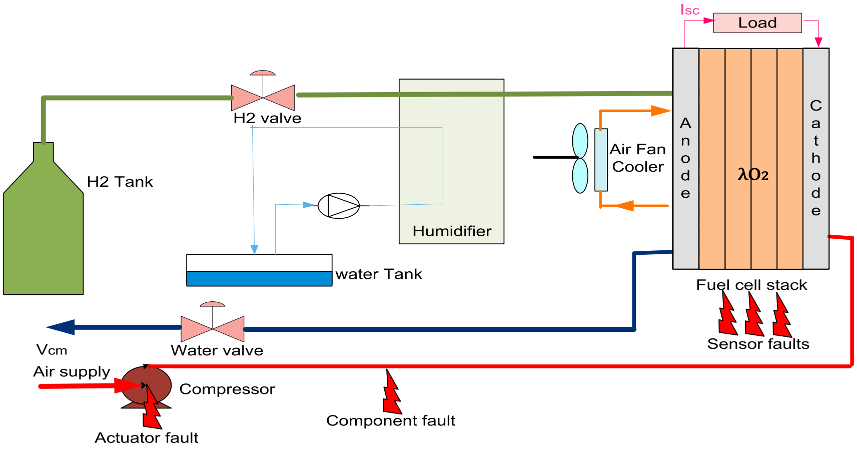

Fig. 8. The schematic of PEMFC systems with five types of faults.

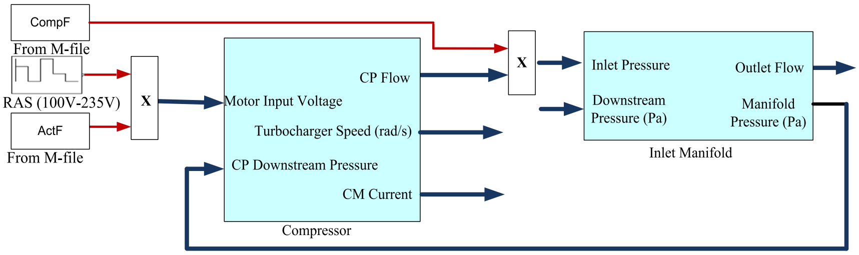

Fig. 9. The simulink block of component and actuator faults.

0.0025, 0.0018 for λO_2 and 0.0029 for stack voltage . The simulation results of the fuel cell stack outputs and the RBF model predictions are shown in Fig. 7 for 300 data samples. These results show the actual and estimated outputs during the testing process when there is a fault occurring in the process. In this period of time only one fault, the third sensor fault, sensorSV occurred for $k=3000$–3200. In the lower figure an observable difference between plant stack voltage and the predicted stack voltage for the samples 3000–3200 but not for 3200–3300 can be seen. It shows that the sensor fault directly influences the performance of stack voltage. However, this fault does not affect the result of net power and λO_2 because this fault is from sensorSV faults. It also observed that the modeling error signals are with sharp spikes caused by step change of load current disturbance and noise. These could be got rid of by applying low-pass filtering.

4. Fault detection

4.1. Simulating faults

The air flow and pressure are the key controlled components of a fuel cell stack for an efficient and dynamic performance of the fuel cell. Fuel cell is in risk of oxygen starvation during high current demand and fast load changes. Oxygen starvation is defined as the ratio of the partial pressure of reserving oxygen to that of used oxygen. It comes along with a drop in partial pressure of oxygen (Pukrushpan et al., 2004a). The oxygen ratio, λO_2, must be kept above a minimum limitation for normal operation. In this study, five faults are introduced to a known test-bench PEMFC based on the model developed by the Michigan University (Kamal and Yu, 2011). First one is an actuator fault, which is simulated by superimposing a ±10% change of the compressor motor voltage measurement. The second is the air leak in the supply manifold which is a typical component fault. The third to fifth are three sensor faults for the three outputs, which are simulated by +10% deviation superimposed to the net power, λO_2 and stack voltage output measurements. The PEMFC simulator was modified to include five possible fault scenarios which may occur during the normal operation of PEMFC systems as shown in Fig. 8 which shows the five faults introduced to the system.

Table 1
The faults data used in the PEMFC systems.

Items	Types of fault	Fault size (%)	Data range
Sample1	SensorNP	+10	1000–1200
	SensorO$_2$	+10	2000–2200
	SensorSV	+10	3000–3200
	Component	+10	4000–4200
	Actuator	+10	5000–5200
Sample2	SensorNP	+10	1000–1200
	SensorO$_2$	+10	2000–2200
	SensorSV	+10	3000–3200
	Component	−10	4000–4200
	Actuator	−10	5000–5200
Sample3	Component	+10	1000–1200
	Actuator	+10	2000–2200
	SensorNP	+10	3000–3200
	SensorSV	+10	4000–4200

4.1.1. Actuator fault

Mostly centrifugal compressor used in fuel cells is susceptible to surge and choke that limit the efficiency and performance of the compressor. The air flow must ensure that the partial pressure of oxygen does not fall below a critical level at the cathode. On the other hand, it must also minimize the parasitic losses of the air pressure. The compressor voltage will be changed if the compressor experiences surge and choke and affects the air flow in the supply manifold. The compressor motor performance is reduced by ±10% of the total compressor motor voltage from the sample intervals, $k=5000$–5200 to reflect the scenario of the fault which happens at the actuator part for sample1 and sample2 while for sampls3 at $k=2000$–2200.

4.1.2. Component fault

Air leakage in the supply manifold makes the pressure in the cathode decrease. Therefore, to collect the fuel cell stack data subjected to the air leak fault, Eq. (5) is modified to

$$\frac{dp_{sm}(t)}{dt} = \frac{\gamma R}{M_a^{atm} V_{sm}}[W_{cp}(t)T_{cp}(t) - W_{sm,out}(t)T_{sm}(t) - \Delta l] \tag{23}$$

where Δl is used to simulate the leakage from the air manifold, which is subtracted to increase the air outflow from the supply manifold. $\Delta l=0$ represents that there is no air leakage in the supply manifold. The air leakage is simulated by ±10% change of the pressure inside the supply manifold. The fault occurs at the sample intervals, $k=4000$–4200 for sample1 and sample2 while for sample3, $k=1000$–1200. Fig. 9 shows the simulink model where the component and actuator fault being injected to the fuel cell stack process to reflect that faults have occurred in the systems.

4.1.3. Sensor faults

Net power, λO_2 and stack voltage sensors are considered experiencing over-reading faults. The faulty sensor data used was the data from the collected data set, superimposed with a +10% change of the measured net power over the sample interval, $k=1000$–1200, a +10% change of the measured λO_2 over the sample intervals, $k=2000$–2200 and a +10% change of the measured stack voltage over the sample intervals, $k=3000$–3200 for sample1 and sample2 while for sample3 as in Table 1.

Table 1 summarizes the three samples used in this work. The purpose of doing the FDI under open-loop systems is to test the effectiveness of the RBF algorithm developed in this work before it can be implemented to the closed-loop systems. Table 1, shows the fault sizes of each faults and the range of faults being introduced. For consistency, 200 data samples are used to indicate the faults of component, actuator of ±10% and for three sensor faults with a fault size of +10% (Fig. 10).

4.2. Fault detection

The implementation of FDI is done in the MATLAB R2009a/ Simulink environment. In this work, a data set with 6000 samples

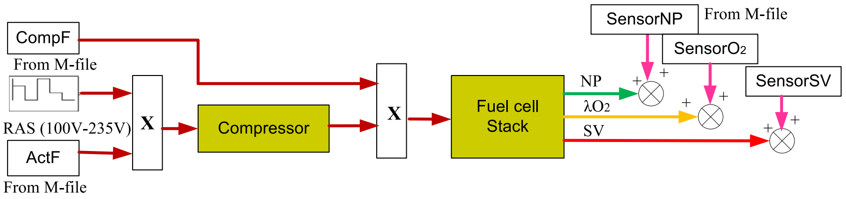

Fig. 10. The simulink block of sensors faults.

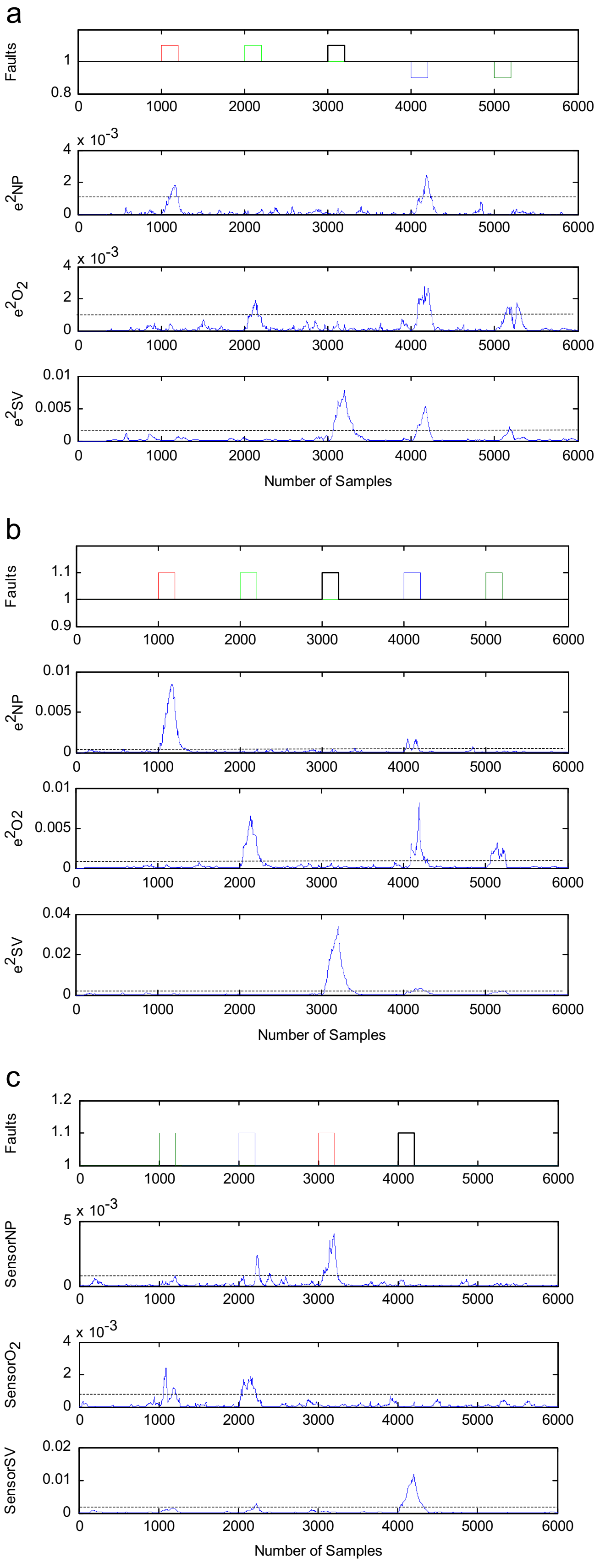

Fig. 11. Filtered model predicted errors for three samples.

is acquired from the plant when the five faults are simulated to the plant. The modeling errors obtained from the difference of plant and the RBF networks containing noisy signals are filtered.

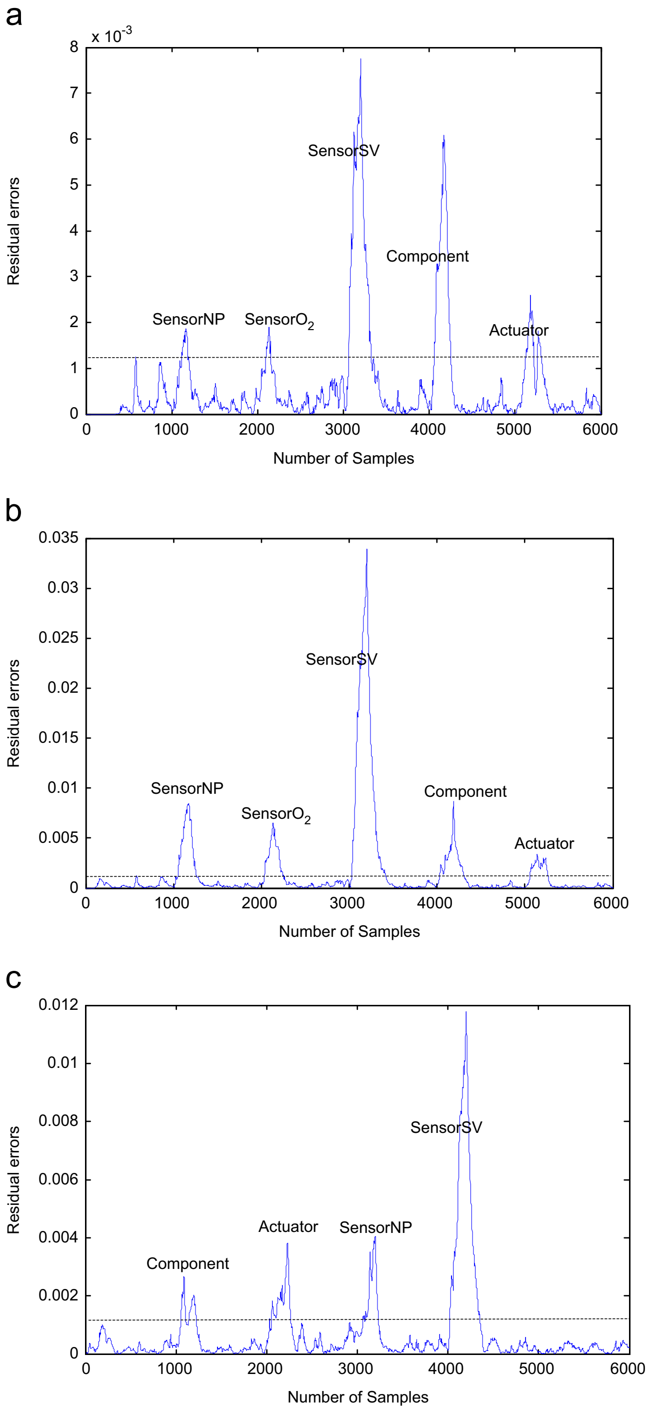

Fig. 12. The fault classification of residual generator faults.

Therefore, to enhance the signal-to-noise ratio the filtered modeling errors are squared. The three squared and filtered model prediction errors are displayed in Fig. 11, with the simulated faults on the top of the figure for easy observation. In Fig. 11 it can be observed that these signals can be used for fault detection, but cannot be used for fault isolation. From observation, the simulation results of sample1, sample2 and sample3 show that there are more than one faults in the output signals.

This can be clearly seen, for example, between the samples k=4000 and k=4200 for Fig. 11(a), that the filtered squared modeling error signals caused by the component fault can be seen in the three outputs of the plant. There are faults occurring in the plant because the signals are over the threshold setting. Again, by referring to Fig. 11(b), the filtered squared modeling error of λO_2 contains three types of faults due to sensor O_2, component and actuator faults. Therefore it is impossible to identify individual faults happening during the operation of the plant. From this point of view, an efficient fault isolation method is needed to tackle this problem.

While Fig. 11(c) shows if there are four faults that happened in the process. Similarly, all four faults can be detected after the threshold value setting. From observation based on Fig. 11(a)–(c), we did not obtain the same pattern of amplitude in the same fault location due to the excitation signals which are randomly generated. That is why the fault detection result in Fig. 11 is not the same.

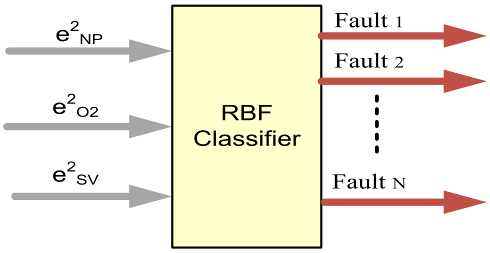

Fig. 13. The block diagram of fault isolation.

Table 2
The target matrix in training the RBF classifier.

Rows	Xo				
1000–1200	[1	0	0	0	0]
2000–2200	[0	1	0	0	0]
3000–3200	[0	0	1	0	0]
4000–4200	[0	0	0	1	0]
5000–5200	[0	0	0	0	1]

4.3. Residual generation

The residual signal is generated by combining the three prediction errors, so that the sensitivity of the residual to each fault can be significantly enhanced, and consequently the false alarm rate would be reduced. The residual in this work is defined as in Janik and Lobos (2006).

$$re = \sqrt{e_{NP}^2 + e_{\lambda O2}^2 + e_{SV}^2} \tag{24}$$

where e_{NP}, $e_{\lambda O_2}$ and e_{SV} are the filtered modeling error of net power, λO_2 and stack voltage, respectively. A threshold is set for the residual signal in Fig. 12. The level of the threshold is according to the modeling error for no fault condition and the noise level. The signal with faults is clearly been identified and is less influenced by a noise signal.

Fig. 12 shows the residual signal produced in simulation of the data set with the 6000 samples where the five faults were simulated for sample1 and sample2 while there are four faults for sample3. By setting a threshold, it can isolate the signals with no faults and with faults after the residual generator is applied. It is observed in Fig. 12 that all five faults of ±10% amplitude are clearly detected for all the three samples data sets. From Fig. 11, by applying Eq. (24) to all three data samples, shows that all five faults in sample1 and sample2 are detectable. Also, by using the same equation, four faults of sample3 can also be detectable. This

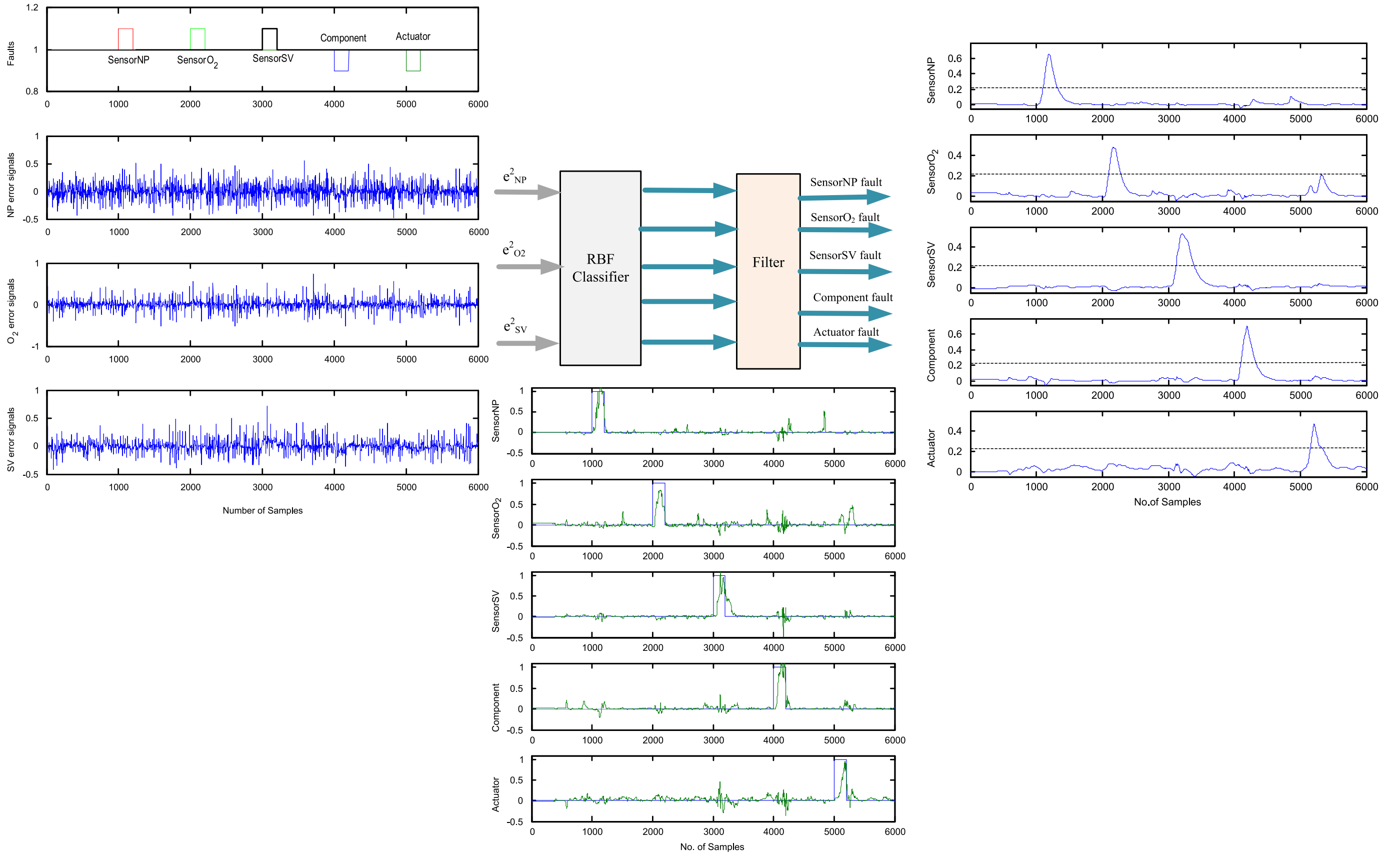

Fig. 14. The overall process of RBF classifier.

implies that the fault detection signal in the proposed method is sensitive to all simulated faults while is robust to the effects of plant dynamics and noise.

5. Fault isolation

The RBF network is well known for its powerful ability to classify components with different features from a mixed signal. Fault isolation in this study is implemented by adding another RBF network as a classifier (Kamal and Yu, 2012). The model prediction error vector obtained from the fault detection part is caused by the faults, and is a nonlinear function of the faults. As this vector is multi-dimension, three dimensions in this study, it will have different structures for different faults. The RBF classifier uses this feature to classify these faults. Based on this idea, the RBF classifier is designed with three inputs to receive the three elements of the filtered and squared model prediction error vector, and five outputs with each being dedicated to one fault. The structure of the RBF classifier is shown in Fig. 13, where e_{NP}^2, $e_{O_2}^2$ and e_{SV}^2 are the input signals. It was found in the experiments that the RBF classifier with 22 hidden layer nodes is most suitable for the classification task in this research.

This RBF classifier is a nonlinear static network. The network is trained with a set of data including 5 subsets. Each subset of data is collected when the system is subjected to one of the 5 simulated faults. Then the classifier is trained with its target vector arranged in the following way: for the data subset with the first fault occurrence, the target for the first output is “1”, while the target for the other outputs are “0”. For the data subset with the second fault occurrence, the target for the second output is “1”, while the target for the other outputs are “0”, and so on so forth, until the final faults. In this study, 6000 samples of data were collected with the first fault occurring during $k=1001$–1200, the second fault occurring during $k=2000$–2200, and so on. Then, the generated filtered and squared model prediction error vector from the fault detection part was used as the input data of the RBF classifier. Correspondingly, the target matrix X_0 has 6000 rows and 5 columns. The entries from the 1000th row to the 1200th row in the first column are “1”, while the other entries are “0”. The arrangement for the columns 2–5 is done in the same way. This is shown as in Table 2 for sample1 example arrangement to do fault isolation.

The target matrix in Table 2 was used in training the RBF classifier. The *s* and widths of the network were chosen using the *K*-means clustering algorithm and the *p*-nearest center algorithm. The weights were trained by using the RLS algorithm with the following data, $\mu=0.99999$, $w(0)=1.0\times10^{-6}\times U_{(nh\times3)}$, $P(0)=1.0\times10^{8}\times I_{(nh)}$; where I is an identity matrix and U is an ones matrix. The RBF networks model only used the three rows of the PEMFC outputs matrix which contain the values of net power, λO_2 and stack voltage. After training, a similar data set also with 6000 samples, with the same five faults simulated, was collected for sample1 and sample2 while 6000 samples with four faults for sample3. These data were applied to the fault detection part and then to the isolation part with the trained RBF classifier.

The filtered squared modeling errors of three outputs will be used by the RBF classifier in order to do fault isolation. These filtered squared modeling errors are trained as stated in Table 2 to differentiate the signals of no fault and with fault. The whole process of fault isolation can be referred to in Fig. 14 where from the modeling errors, the signals are squared before the classifier takes place. The RBF classifier will train the individual fault according to Table 2. Once the fault has been classified, the signal is filtered because the signal has a lot of disturbance in it. After the filtering process, a smoother signal is obtained and a threshold is set to indicate whether it is a false signal.

The five outputs of the classifier are displayed in Fig. 15 for sample1, sample2 and sample3. As can be seen in Fig. 15, the RBF classifier will be trained with ‘1’ if there is fault existing and ‘0’ if there is no fault. From the fault isolation signals in Fig. 15 it is observed that any fault isolation signal of the five is sensitive to the

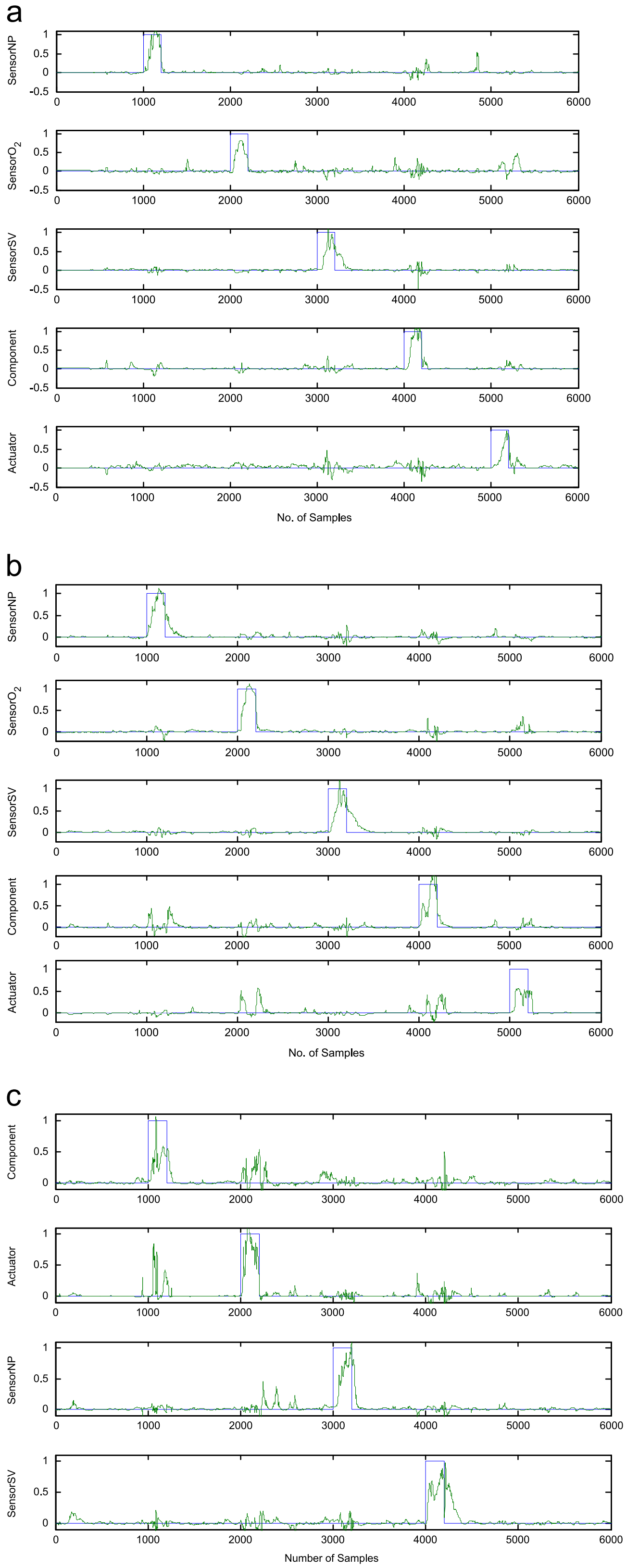

Fig. 15. The fault isolation for test data sets with faults.

fault represented by it, while is insensitive to all the other four faults. In this way, all the five considered faults have been clearly isolated. The RBF classifier successfully suppressed the corresponding output value for the no-fault-occurring period, while promoted the corresponding output value for the fault-occurring period. By comparing the curves in Fig. 11, it is not difficult to find out that the component fault caused a big spike in all the three model prediction errors in Fig. 11, which make the isolation of this fault difficult. When these signals are fed into the RBF classifier, only the fourth output (the corresponding output to the component fault) had a response to the component fault, while all the other four outputs did not have significant output values. Therefore, the component fault is clearly isolated from the others using this method. It is noticed in Fig. 15 that the fault isolation signals are very noisy and that would cause false alarm. Then, the RBF classifier outputs are filtered by a low-pass filter, and the filtered signal can clearly isolate the occurred fault and will not cause false alarm.

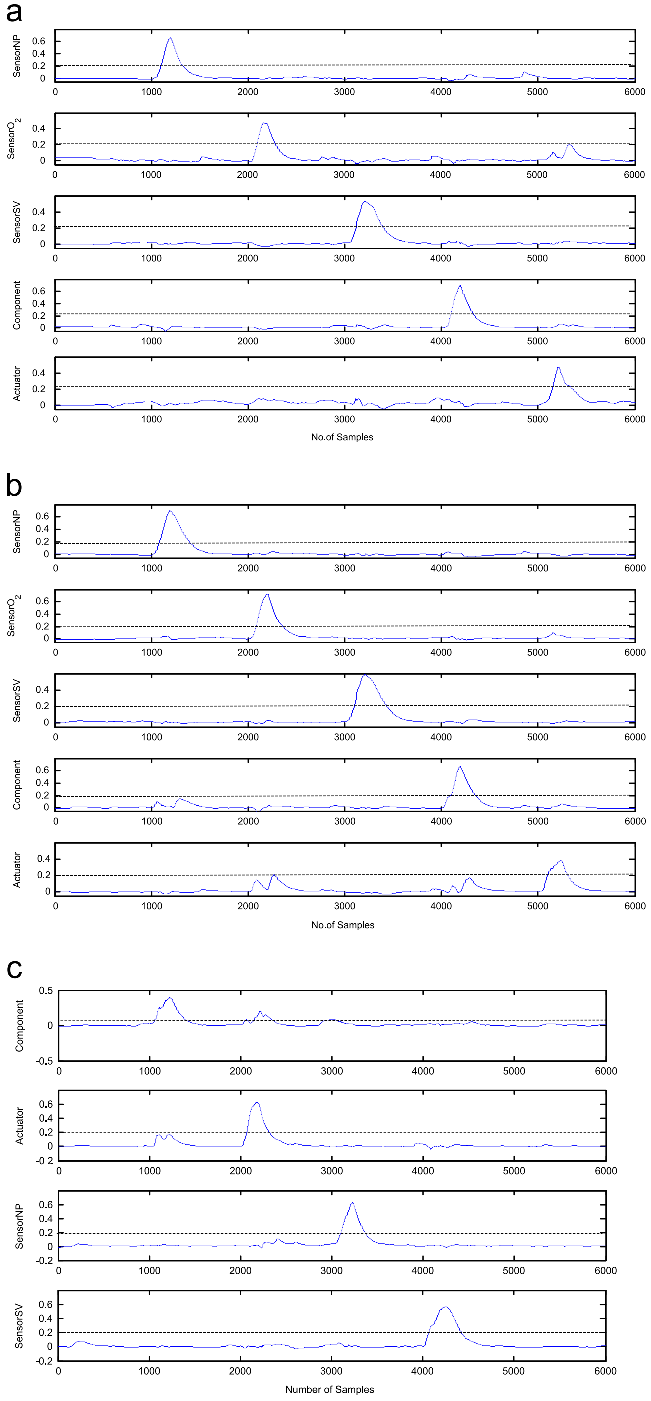

Fig. 16. The location of faults in the PEMFC systems.

Therefore, the RBF classifier outputs are filtered for these three data samples and the filtered signals are displayed in Fig. 16. It is obvious that the filtered fault isolation signals are much smoother and the robustness of the signal to modeling errors, interactions between variables and noise is greatly enhanced. It is important to isolate the malfunctioning devices in the systems for easy troubleshooting and maintenance purposes. By doing this step, the device can easily be replaced and any appropriate action can be taken quickly and therefore it can save time and increase productivity. Fig. 16 shows the final results of fault isolation for the three samples used in this work.

6. Conclusions

This paper presents a model-based FDI for PEMFC dynamic systems. The RBF network model can predict process output precisely when it is used in the independent mode. Then, by classifying the different features of different faults on the model prediction error vector, these faults are clearly isolated with a much higher signal-to-noise ratio. For the simulated actuator, component and sensor faults to the benchmark PEMFC model, it is evident that these faults are detected and isolated when the system in a transient state and under varying load disturbance. It is shown that all these faults with the amplitude of only ±10% of variable's nominal value can be clearly detected and isolated. Here, we considered the fault condition that single fault occurs at a time. But this result can be extended to the fault condition of multi-faults occurring simultaneously. The developed method has a big potential to be applied to real world fuel cell stacks. Also, the method is not limited to fuel cell systems, and can be applied to other multivariable nonlinear dynamic systems with some modifications.

References

Basseville, M., 1988. Detecting changes in signals and systems – a survey. Automatica 24 (3), 309–326.

Chong, E.K.P., Zak, S.H., 2013. An Introduction to Optimization. John Wiley & Sons, New Jersey.

Frank, P.M., 1990. Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy – a survey and some new results. Automatica 26 (3), 459–474.

Frank, P.M., Koppen-Seliger, B., 1997. Fuzzy logic and neural network applications to fault diagnosis. International Journal of Approximate Reasoning 16 (1), 67–88.

Gomm, J.B., Yu, D.L., 2000. Selecting radial basis function network centers with recursive orthogonal least squares training. IEEE Trans. Neural Networks 11 (2), 306–314.

Hwang, I., Kim, S., Kim, Y., Seah, C.E., 2010. A survey of fault detection, isolation and reconfiguration methods. IEEE Trans. Control Syst. Technol. 18 (3), 636–653.

Ingimundarson, A., Stefanopoulou, A.G., McKay, D., 2008. Model-based detection of hydrogen leak in a fuel cell stack. IEEE Trans. Control Syst. Technol. 16 (5), 1004–1012.

Isermann, R., 1984. Process fault detection based on modelling and estimation methods – a survey. Automatica 20 (4), 387–404.

Janik, P., Lobos, T., 2006. Automated classification of power-quality disturbances using SVM and RBF networks. IEEE Trans. Power Delivery 21 (3), 1663–1669.

Kamal, M.M., Yu, D.L., 2011. Model-based fault detection for proton exchange membrane fuel cell systems. Int. J. Eng. Sci. Technol. 3 (9), 1–15.

Kamal, M.M., Yu, D.L., 2012. Fault detection and isolation for PEMFC systems under closed-loop control. In: Proceedings of UKACC International Conference on Control 2012 Cardiff UK, 3–5 September 2012, pp. 976–981.

Lebbal, M.E., Lecoeuche, S., 2009. Identification and monitoring of PEM electrolyser based on dynamical modelling. J. Power Sour. 34 (14), 5992–5999.

Narendra, K.S., Parthasarathy, K., 1990. Identification and control of dynamical systems using neural networks. IEEE Trans. Neural Networks 1 (1), 4–27.

Ng, Y.S., Srinivasan, R., 2010. Multi-sgent based collaborative fault detection and identification in chemical processes. Eng. Appl. Artificial Intell. 23, 934–949.

Patton, R.J., 1994. Robust model-based fault diagnosis: The state of art. In: Proceeding of IFAC Symposium SAFEPROCESS'94, Helsinki, Finland.

Patton, R.J., Chen, J., Siew, T.M., 1994. Fault diagnosis in nonlinear dynamic systems via neural networks. In: International Conference on Control (2), pp. 1346–1351.

Polycarpou, M.M., Helmicki, A.J., 1995. Automated fault detection and accommodation: a learning systems approach. IEEE Trans. Syst. Man Cybern. 25 (11), 1447–1458.

Pukrushpan, J.T., Peng, H., Stefanopoulou, A.G., 2004a. Control-oriented modeling and analysis for automotive fuel cell systems. J. Dyn. Syst. Meas. Control 126, 14–26.

Pukrushpan, J.T., Stefanopoulou, A.G., Peng, H., 2004b. Control of fuel cell breathing. IEEE Control Syst. Mag. 24 (2), 30–46.

Simeón, E.A., Álvares, A.J., Gudwin, R.R., 2010. An expert system for fault diagnostics in condition based maintenance. In: ABCM Symposium Series in Mechatronics, vol. 4, pp. 304–313.

Subrahmanya, N., Shin, Y.C., 2013. A data-based framework for fault detection and diagnostics of non-linear systems with partial state measurement. Eng. Appl. Artificial Intell. 26 (1), 446–455.

Verron, S., Tiplica, T., Kobi, A., 2010. Fault diagnosis of industrial systems by conditional Gaussian network including a distance rejection criterion. Eng. Appl. Artificial Intell. 23 (7), 1229–1235.

Wang, S.W., Yu, D.L., Gomm, J.B., Page, G.F., Douglas, S.S., 2006. Adaptive neural network model based predictive control for air–fuel ratio of SI engine. Eng. Appl. Artificial Intell. 19 (2), 189–200.

Xue, X., Tang, J., Sammes, N., Ding, Y., 2006. Model-based condition monitoring of PEM fuel cell using Hotelling T^2 control limit. J. Power Sour. 162 (1), 388–399.

Yu, D.L., Gomm, J.B., Williams, D., 1999. Sensor fault diagnosis in a chemical process via RBF neural networks. Control Eng. Pract. 7 (1), 49–55.