# ROBUST FAULT DIAGNOSIS IN CATALYTIC CRACKING CONVERTER USING ARTIFICIAL NEURAL NETWORKS

**Krzysztof Patan** [*,1]

[*] *Institute of Control and Computation Engineering*
*University of Zielona Góra*
*ul. Podgórna 50, 65-246 Zielona Góra, Poland*
*e-mail: k.patan@issi.uz.zgora.pl*

Abstract: The paper presents designing of a robust fault diagnosis system for a catalytic cracking process using artificial neural networks. Identification of the considered process is carried out by using recurrent neural networks. To achieve a robust fault diagnosis system, an uncertainty associated with the model is also taken into account. Neural version of the Model Error Modelling is used to deal with two main uncertainty sources: unmodelled dynamics and noise corrupting the data. The proposed approach is tested on the example of catalytic cracking converter at the nominal operations condition as well as in the case of faults.

Keywords: Identification, Neural Network, Fault diagnosis, Modelling errors, Uncertainty.

## 1. INTRODUCTION

Fault diagnosis becomes an important issue in modern control systems due to their increasing complexity. An early diagnosis of faults that might occur in the supervised process, renders it possible to perform important preventing actions. Moreover, it allows one to avoid heavy economic losses involved in stopped production, replacement of elements and parts, etc. The basic idea of model-based fault diagnosis is to generate signals that reflect inconsistencies between nominal and faulty system operation conditions. Such signals, called residuals, are usually calculated by using analytical methods such as observers (Chen and Patton, 1999), parameter estimation methods (Isermann, 1994) or parity equations (Gertler, 1999). Unfortunately, the common drawback of these approaches is that an accurate mathematical model of the diagnosed plant is required, so their application is limited. An alternative solution can be obtained through artificial intelligence, e.g. neural networks (Frank and Köppen-Seliger, 1997; Calado *et al.*, 2001).

The model-based fault diagnosis is built on a number of idealized assumptions. One of them is that model of the system is a faithful replica of a plant dynamics. Another one is that disturbances and noise acting upon the system are known. This is, of course, not possible in engineering practice. The robustness problem in fault diagnosis can be defined as the maximization of the detectability and isolability of faults and simultaneously the minimization of the uncontrolled effects such as disturbances, noise, changes in inputs and/or state, etc. (Chen and Patton, 1999).

In general, methods to achieve and analyze robustness can be divided into two groups. The first group of methods considers robustness as a

[1] The work was supported by the State Committee for Scientific Research in Poland (KBN) under the grant 4T11A01425 *Modelling and identification of nonlinear dynamic systems in robust diagnosis.*

part integrated into identification process. The second one identifies a model first without robustness considerations and then performing additional step by adjusting or compensating the originally constructed model. The paper describes an approach, which belongs to the second group of methods. First, the system is identified using a recurrent network. The architecture details are given in Section 2. Next, the uncertainty of the model is obtained by application of Model Error Modelling performed using feed-forward networks with delays. This technique is presented in Section 3. The proposed approach is tested on the example of catalytic cracking converter described in Section 4. Section 5 reports the experimental results.

## 2. DYNAMIC NEURAL NETWORKS

An artificial neural network used to model the system behaviour belongs to the class of so-called locally recurrent globally feed-forward networks. Its structure is similar to a multi-layer perceptron where neurons are organized in layers, but dynamic properties are achieved using neurons with internal feedbacks. One of the dynamic processing unit realization is a neuron model with the infinite impulse response filter (Ayoubi, 1994; Patan and Parisini, 2005). This structure is a generalized version of the neuron with an activation feedback model. The block structure of the $i$-th neuron considered is presented in the Fig. 1. Dynamics is introduced into the neuron in such a way that the neuron activation depends on its internal states. It is done by introducing a linear dynamic system – the IIR filter – into the neuron structure. Thus, the $i$-th neuron in the dynamic network reproduces the past signal value with two signals: the input $\boldsymbol{u}(k)$ and its output $y_i(k)$. The weighted sum of inputs is calculated according to the formula

$$s_i(k) = \boldsymbol{w}_i\boldsymbol{u}(k), \tag{1}$$

where $\boldsymbol{w}_i = [w_1^i, w_2^i, \ldots, w_n^i]$ denotes the input weights vector, $n$ is the number of inputs, and $\mathbf{u}(k) = [u_1(k), u_2(k), \ldots, u_n(k)]^T$ is the input vector ($T$ – transposition operator). The weights perform a similar role as in static feed-forward networks, namely, together with activation functions are responsible for approximation properties of the model. Then, the calculated sum $s_i(k)$ is passed to the IIR filter of the order $r$. Here, the filters under consideration are linear dynamic systems of different orders, e.g. the first or second order. The states of the $i$-th neuron in the network can be described by the following state equation:

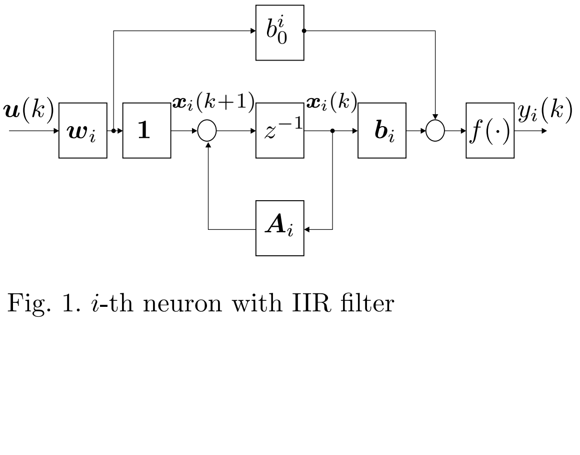

Fig. 1. $i$-th neuron with IIR filter

$$\boldsymbol{x}_i(k+1) = \boldsymbol{A}_i\boldsymbol{x}_i(k) + \mathbf{1}\boldsymbol{w}_i\boldsymbol{u}(k), \tag{2}$$

where the state vector $\boldsymbol{x}_i(k) = [x_1^i(k),\ x_2^i(k),\ \ldots,\ x_r^i(k)]^T$, $\mathbf{1} \in \mathbb{R}^r$ is the vector of ones and the state transition matrix $\boldsymbol{A}_i$ has a form

$$\boldsymbol{A}_i = \begin{bmatrix} -a_1^i & -a_2^i & \ldots & -a_{r-1}^i & -a_r^i \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{bmatrix}. \tag{3}$$

Finally, the neuron output is described by

$$y_i(k) = f\big(\boldsymbol{b}_i\boldsymbol{x}_i(k) + \boldsymbol{d}_i\boldsymbol{u}(k), \boldsymbol{g}_i\big) \tag{4}$$

where $f(\cdot)$ is a non-linear activation function, $\boldsymbol{b}_i = [b_1^i, b_2^i, \ldots, b_r^i]$ is the vector of feed-forward filter parameters, $\boldsymbol{d}_i = [b_0^i w_1, b_0^i w_2, \ldots, b_0^i w_n]$, $\boldsymbol{g}_i = [g_1^i, g_2^i]$ is the vector of the activation function parameters consisting two elements: $g_1$ and $g_2$, which are the bias and slope of the activation function, respectively. In the majority of cases, the neural activation function $f(\cdot)$ is chosen as a continous and differentiable non-linear hyperbolic tangent function

$$f(x, g_1, g_1) = \frac{1 - e^{-g_2(x-g_1)}}{1 + e^{-g_2(x-g_1)}}, \tag{5}$$

where $g_2 > 0$.

### *2.1 State-space representation of the network*

In this paper, a discrete-time dynamic network with $n$ time varying inputs and $m$ outputs is discussed. The description of such kind of dynamic network with $v$ hidden dynamic neurons, each containing an $r$-th order IIR filter, is given by the following non-linear system:

$$\begin{cases} \boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{W}\boldsymbol{u}(k) \\ \boldsymbol{y}(k) = \boldsymbol{C}\boldsymbol{f}(\boldsymbol{B}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{u}(k), \boldsymbol{G})^T \end{cases}, \tag{6}$$

where $N = v \times r$ represents the number of states, $\boldsymbol{x} \in \mathbb{R}^N$ is the neural state vector, $\boldsymbol{u} \in \mathbb{R}^n$, $\boldsymbol{y} \in \mathbb{R}^m$ are the input and output vectors, repectively, $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ represents the diagonal weight matrix $\boldsymbol{A} = \mathrm{diag}(\boldsymbol{A}_1, \ldots, \boldsymbol{A}_v)$ associated with the neural state, $\boldsymbol{W} \in \mathbb{R}^{N\times n}$ and $\boldsymbol{C} \in \mathbb{R}^{m\times v}$ are the input and output weights matrices, respectively, $\boldsymbol{B} \in \mathbb{R}^{v\times N}$ is the block diagonal matrix associated with the feed-forward filter parameters, $\boldsymbol{D} \in \mathbb{R}^{v\times n}$ is the transfer matrix, $\boldsymbol{G} \in \mathbb{R}^{v\times 2}$ denotes the activation function parameters, and $\boldsymbol{f} :\ \mathbb{R}^v \rightarrow$

$\mathbb{R}^v$ is a non-linear vector-valued function. The forms of the matrices $\boldsymbol{B}$, $\boldsymbol{D}$ and $\boldsymbol{G}$ are given below:

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{b_1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{b_2} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{b_v} \end{bmatrix} \tag{7}$$

and

$$\boldsymbol{D} = \begin{bmatrix} b_0^1\boldsymbol{w}_1 \\ b_0^2\boldsymbol{w}_2 \\ \vdots \\ b_0^v\boldsymbol{w}_v \end{bmatrix} \quad \boldsymbol{G} = \begin{bmatrix} g_1^1 & g_2^1 \\ g_1^2 & g_2^2 \\ \vdots & \vdots \\ g_1^v & g_2^v \end{bmatrix}. \tag{8}$$

The presented structure can be viewed as a network with a single hidden layer containing $v$ dynamic neurons as processing elements and an output layer with linear static elements. The LRGF networks possess many advantages over classical recurrent networks (Tsoi and Back, 1994; Nelles, 2001). One of them is that keeping stability of a network during training is much easier than in the case of recurrent architectures. The interested reader is referred to (Patan, 2004*a*; Patan, 2004*b*).

### *2.2 Training of the network*

All unknown network parameters can be represented by a vector $\boldsymbol{\theta}$ composed of elements of matrices $\boldsymbol{A}$, $\boldsymbol{W}$, $\boldsymbol{C}$, $\boldsymbol{B}$, $\boldsymbol{D}$ and $\boldsymbol{G}$ described in details in Section 2.1. The main objective of training is to adjust the elements of the vector $\boldsymbol{\theta}$ in such a way as to minimize some loss (cost) function:

$$\boldsymbol{\theta}^\star = \arg\min_{\theta \in C} J(\boldsymbol{\theta}) \tag{9}$$

where $\boldsymbol{\theta}^\star$ is the optimal network parameter vector, $J : \mathbb{R}^p \rightarrow \mathbb{R}$ represents some loss function to be minimized, $p$ is the dimension of the vector $\boldsymbol{\theta}$, and $C \subseteq \mathbb{R}^p$ is the set of admissible parameters constituted by constraints. To minimize (9) one can use gradient based (Patan and Korbicz, 2004) or stochastic approximation (Patan and Parisini, 2005) methods. For training, which guarantees stability of the model, the reader is referred to (Patan, 2004*a*; Patan, 2004*b*).

## 3. NEURAL MODEL ERROR MODELLING

The robust identification procedure should deliver not only a model of a given process, but also a reliable estimate of the uncertainty associated with the model. Two main philosophies exist in the literature. The first group of approaches, called bounding error methods, relies on the assumption that identification error is unknown but bounded. In this framework, the robustness is hardly integrated with the identification process. A somewhat different approach is to first identify the process without robustness considerations and then consider robustness as an additional step. This usually leads to least squares estimation and prediction error methods.

Model error modelling employs prediction error methods to identify a model from input-output data (Reinelt *et al.*, 2002). After that, one can estimate uncertainty of the model by analyzing residuals evaluated from the inputs. The uncertainty is a measure of unmodelled dynamics, noise and disturbances. Identification of residuals provides the so-called *model error model*. Designing procedure is described by the following steps:

(1) compute the residual $r = y - y_m$, where $y$ and $y_m$ are desired and model outputs, respectively
(2) collect the data $\{u_i, r_i\}_{i=1}^N$ and identify an error model using these data. This model constitutes an estimate of the error due to under modelling, and it is called model error model.
(3) construct a model along with uncertainty using both nominal and model error models.

The paper of (Reinelt *et al.*, 2002) proposes to carry out step 3 in the frequency domain adding frequency by frequency the model error to the nominal model. If the model error model is not falsified by the data, one can use statistical properties to calculate a confidence region. A confidence region form an uncertainty band around response of the model error model.

In this paper it is proposed to form uncertainty band in the time domain. First, the model error modelling is performed by using the well-known multi-layer perceptron with tapped delay lines, also known as the Neural Network ARX (NNARX) model (Norgard *et al.*, 2000). Response of this network is then used to form the uncertainty band in the following way: the upper band

$$r_u = y_m + y_e + t_\alpha v \tag{10}$$

and the lower band

$$r_l = y_m + y_e - t_\alpha v \tag{11}$$

where $y_e$ is the output of the error model on the input $\boldsymbol{u}$, $t_\alpha$ is $N(0,1)$ tabulated value assigned to $1-\alpha$ confidence level, $v$ is the standard deviation of $y_e$. It should be kept in mind that $y_e$ represents not only residual but also the structured uncertainty, disturbances, etc. Therefore, the uncertainty bands (10) and (11) should work well only assuming that signal $y_e$ has normal distribution. The centre of the uncertainty region is the signal $y_m + y_e \approx y$. Now, observing the system output $y$, one may take a decision whether the fault occurred or not. If the $y$ is inside the uncertainty region the system is healthy.

### *3.1 Fault detection sensitivity checking*

In order to check the fault detection sensitivity, a number of performance indexes have been applied.

(1) False detection rate defined as follows:

$$r_{fd} = \frac{\sum_i t^i_{fd}}{t_{from} - t_{on}} \qquad (12)$$

where $t^i_{fd}$ is the period of $i$-th false fault detection, $t_{on}$ is the benchmark start up time. This index is used to check the system in normal operation conditions. Its value shows a percentage of false alarms.

(2) True detection rate:

$$r_{td} = \frac{\sum_i t^i_{td}}{t_{hor} - t_{from}} \qquad (13)$$

where $t^i_{td}$ is the period of $i$-th true fault detection, $t_{hor}$ is the benchmark time horizon. This index is used in the case of faults and describes efficiency of fault detection.

(3) Detection time $t_{dt}$: period of time from the begin of fault start-up $t_{from}$ to the moment of fault detection.

## 4. CATALYTIC CRACKING

The Fluid Catalytic Cracking (FCC) converts heavy oil into lighter, more valuable fuel products and petrochemical feedstocks. The general scheme of catalytic cracking process is presented in Fig. 2 (Moro and Odloak, 1995). It consists of three main subsystems: reactor, riser and regenerator.

Finely sized solid catalyst continuously circulates in a closed loop between the reactor and regenerator. The reactor provides proper feed contacting time and temperature to achieve the desired level of conversion, and to disengage products from the spent catalyst. The regenerator restores catalytic activity of the coke-laden spent catalyst by combustion with air. It also provides heat of reaction and heat of feed vaporization by returning hot, freshly regenerated catalyst back to the reaction system. Hot regenerated catalyst flows to the base of the riser where it is contacted with heavier feed. Vaporized feed and catalyst travel up the riser where vapour phase catalytic reactions occur. The reacted vapour is rapidly disengaged from the spent catalyst in direct-coupled riser cyclones and is directly routed to product fractionation in order to discourage further thermal and catalytic cracking. In the product recovery system, reactor vapours are quenched and fractionated, yielding dry gas, LPG, naphtha, and middle distillate products.

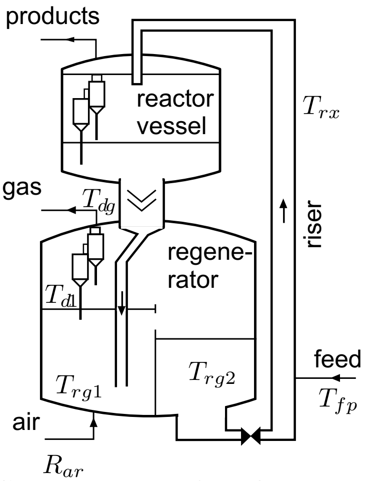

Fig. 2. General scheme of the fluid catalytic cracking converter

Table 1. Specification of the measurable process variables

| Variable | Description |
|---|---|
| $R_{ar}$ | air flowrate to regenerator $[ton/h]$ |
| $T_{ar}$ | air temperature $[C]$ |
| $T_{fp}$ | feed temperature at riser entrance $[C]$ |
| $T_{rx}$ | temperature of cracking mixture in riser $[C]$ |
| $T_{rg1}$ | temperature of dense phase at regenerator first stage $[C]$ |
| $T_{rg2}$ | temperature of dense phase at regenerator second stage $[C]$ |
| $T_{d1}$ | temperature of the regenerator first stage dilute phase $[C]$ |
| $T_{dg}$ | temperature of the general dilute phase $[C]$ |

The whole catalytic cracking process has been implemented in Simulink as a FCC benchmark according to the mathematical description presented in (Moro and Odloak, 1995). The benchmark is available on the website: `http://www.enq.ufrgs.br/recope/FCC`. The manipulated variables of a crucial importance are the flowrate of regenerated catalyst to the riser and the flowrate of combustion air to the regenerator beds. The available measurement variables are presented in Table 1. Taking into account the expert knowledge about the technological process one can design the following relations between variables:

- Temperature of cracking mixture

$$T_{rx} = s_1(T_{rg2}, T_{fp}) \qquad (14)$$

- Temperature of dense phase at regenerator first stage

$$T_{rg1} = s_2(T_{rg1}, T_{ar}, R_{ar}) \qquad (15)$$

- temperature of dense phase at regenerator second stage

$$T_{rg2} = s_3(T_{rg1}, T_{ar}, R_{ar}) \qquad (16)$$

- Temperature of the regenerator first stage dilute phase

$$T_{d1} = s_4(T_{rg1}) \qquad (17)$$

- Temperature of the general dilute phase

$$T_{dg} = s_5(T_{d1}) \qquad (18)$$

## 5. EXPERIMENTAL RESULTS

### *5.1 Process modelling*

In order to design fault diagnosis system for the FCC process, a neural network is used to describe the process at normal operation conditions. First, the network has to be trained for this task. Training data has been collected from the FCC benchmark. The model is designed applying the LRGF neural network presented in Section 2. The network has been trained using stochastic approximation method to mimic the behaviour of the temperature of cracking mixture 14. The neural model (6) has two inputs $T_{rg2}$ and $T_{fp}$, one output $T_{rx}$, and consists of three hidden neurons, with hyperbolic tangent activation function and second order IIR filter each.

### *5.2 Confidence bands*

Decision making is carried out using uncertainty bounds obtained by using model error modeling presented in Section 3. Many neural architectures of the NNARX type have been examined by the trial and error method. The best performing two-layered network consists of five hidden neurons and one output element, all of them with hyperbolic tangent activation function. Number of input and output delays was equal to 15. The conclusion is that to capture the residual dynamics, a high order model is required. To determine confidence bands, the 99% significance level was assumed ($\alpha = 0.01$). The uncertainty region (dashed lines) along with the output of the healthy system (solid line) is shown in Fig. 3. The false detection rate in this case was $r_{fd} = 0.0881$.

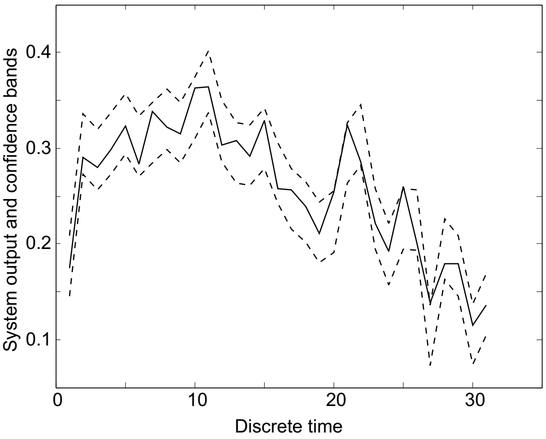

Fig. 3. Confidence bands and system output at the nominal operation conditions

### *5.3 Fault detection*

The FCC model makes it possible to generate a number of faulty scenarios and in this way to examine the efficiency of the proposed robust fault detection approach. Four fault scenarios are proposed according to (Sotomayor *et al.*, 2004):

(1) Scenario $f_1$ – 10% increasing of the catalyst density,
(2) Scenario $f_2$ – 15% decreasing of the weir constant of the first and second stages,
(3) Scenario $f_3$ – 10% decreasing of the $CO_2/CO$ ratio constant,
(4) Scenario $f_4$ – 5% increasing of catalyst reactor holdup.

These faulty scenarios have been implemented in Simulink/Matlab as an additional component to the mentioned FCC benchmark. The results of fault detection are presented in Fig. 4, and Table 2. In Fig. 4, the uncertainty bands are marked using grey lines, and system output with the black line.

Table 2. Performance indexes for the faulty scenarios.

| scenario | start-up time | $r_{td}$ | $t_{dt}$ |
|---|---|---|---|
| $f_1$ | 1900 | 0.6978 | 250 |
| $f_2$ | 1900 | 0.7184 | 370 |
| $f_3$ | 2020 | 0.6634 | 80 |
| $f_4$ | 2000 | 0.5294 | 55 |

## 6. CONCLUDING REMARKS

The paper presents the robust fault diagnosis realized by using artificial neural networks. The LRGF network is used to model the process at normal operation conditions and the NNARX is used to identify the error model (residual). The preliminary experiments show that the proposed method gives promising results. The worst results were obtained for scenario $f_4$ were true detection rate was equal to 0.5294. It means that during occurence of this faulty scenario about half of samples were inside of the uncertainty region giving wrong information about a system condition. The open problem here is to find a proper model error model. This problem seems to be much more difficult to solve that finding a model of the system.

## REFERENCES

Ayoubi, M. (1994). Fault diagnosis with dynamic neural structure and application to a turbocharger. In: *Proc. Int. Symp. Fault Detection Supervision and Safety for Technical Processes, SAFEPROCESS'94, Espoo, Finland.* Vol. 2. pp. 618–623.

Calado, J.M.F., J. Korbicz, K. Patan, R.J. Patton and J.M.G. Sa da Costa (2001). Soft computing approaches to fault diagnosis for dynamic systems. *European Journal of Control* **7**(2-3), 248–286.

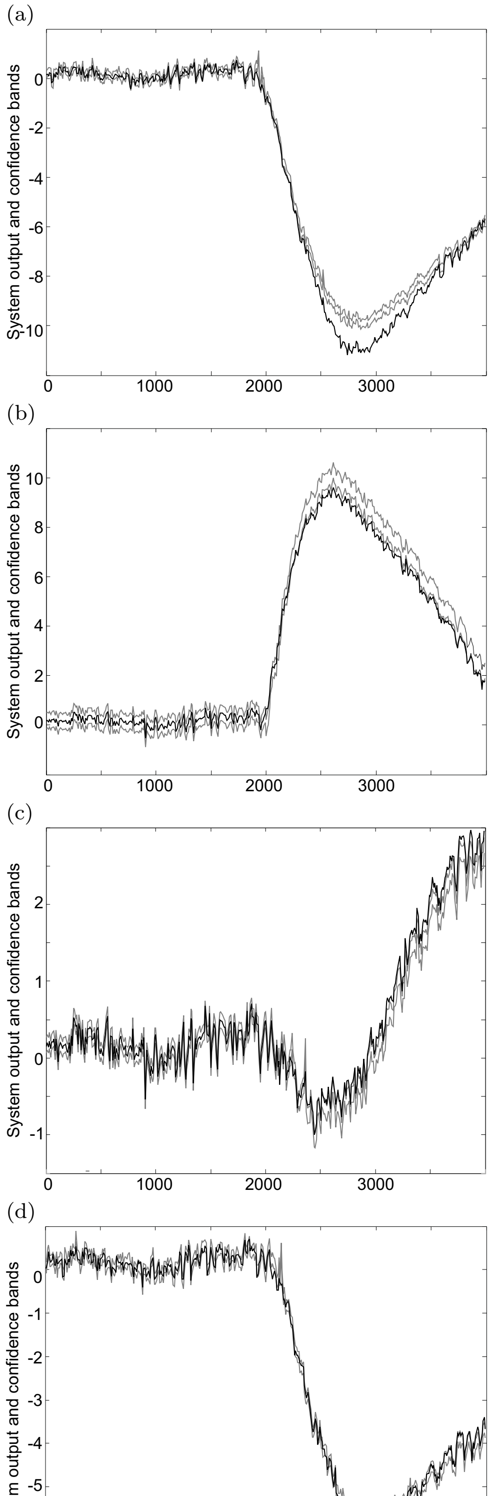

Fig. 4. Fault detection results: scenario $f_1$ (a), scenario $f_2$ (b), scenario $f_3$ (c), scenario $f_4$ (d)

.

Chen, J. and R. J. Patton (1999). *Robust Model-Based Fault Diagnosis for Dynamic Systems.* Kluwer Academic Publishers. Berlin.

Frank, P. M. and B. Köppen-Seliger (1997). New developments using AI in fault diagnosis. *Artificial Intelligence* **10**(1), 3–14.

Gertler, J. (1999). *Fault Detection and Diagnosis in Engineering Systems.* Marcel Dekker, Inc.. New York.

Isermann, R. (1994). Process fault detection and diagnosis methods. In: *Proc. IFAC Symposium SAFEPROCESS'94, Helsinki, Finland* (T. Ruokonen, Ed.). Vol. 2. Pergamon Press. Helsinki, Finland.

Moro, L. F. L. and D. Odloak (1995). Constrained multivariable control of fluid catalytic cracking converters. *Journal of Process Control* **5**(1), 29–39.

Nelles, O. (2001). *Nonlinear System Identification. From Classical Approaches to Neural Networks and Fuzzy Models.* Springer-Verlag. Berlin.

Norgard, M., O. Ravn, N.M. Poulsen and L.K. Hansen (2000). *Networks for Modelling and Control of Dynamic Systems.* Springer-Verlag. London.

Patan, K (2004*a*). Model-based actuator fault diagnosis using dynamic neural networks. In: *Proc. 10th IEEE Int. Conf. Methods and Models in Automation and Robotics, MMAR 2004. Międzyzdroje, Poland.* pp. 743–748.

Patan, K (2004*b*). Training of the dynamic neural networks via constrained optimization. In: *Proc. IEEE Int. Joint Conference on Neural Networks, IJCNN 2004, Budapest, Hungary.* published on CD-ROM.

Patan, K. and J. Korbicz (2004). Artificial neural networks in fault diagnosis. In: *Fault Diagnosis. Models, Artificial Intelligence, Applications* (J. Korbicz, J. M. Kościelny, Z. Kowalczuk and W. Cholewa, Eds.). Springer-Verlag. Berlin. pp. 330–380.

Patan, K. and T. Parisini (2005). Identification of neural dynamic models for fault detection and isolation: the case of a real sugar evaporation process. *Journal of Process Control* **15**, 67–79. in press, available on-line.

Reinelt, W., A. Garulli and L. Ljung (2002). Comparing different approaches to model error modeling in robust identification. *Automatica* **38**, 787–803.

Sotomayor, O. A. Z., D. Odloak, E. Alcorta-Garcia and P. Léon-Cantón (2004). Observer-based supervision and fault detection of a FCC unit model predictive control system. In: *Proc. 7th Int. Symp. Dynamic and Control of Process Systems, DYCOPS 7, Massachusetts, USA.*

Tsoi, A. Ch. and A. D. Back (1994). Locally recurrent globally feedforward networks: A critical review of architectures. *IEEE Transactions on Neural Networks* **5**, 229–239.