Fault detection of the boiler unit using state space neural networks.

A. Czajkowski, K. Patan

Institute of Control and Computation Engineering, University of Zielona Góra, ul. Podgórna 50, 65-246 Zielona Góra; (e-mail: {A.Czajkowski, K.Patan}@issi.uz.zgora.pl).

Abstract: This paper deals with the general information about using state space neural network models for purpose of the fault detection on example of the boiler unit. The work presented deals with problems such as selecting proper threshold for compromising both fault sensitivity and early fault detection, designing proper neural network structure or calculating performance indices. All simulation data used in experiments are collected from simulator of the boiler unit implemented in Matlab/Simulink.

Keywords: nnsysid, state space model, dynamic system, neural network, fault detection.

1. INTRODUCTION

Recently it is observed an increasing development of the Fault Diagnosis (FD) methods for the Fault Tolerant Control (FTC) system design purposes. It is strictly connected to the advantages of the systems which can maintain current performance as close to the desirable one, and preserve stability conditions in the presence of faults. Faults and equipment failures directly affect the performance of the control system and can result in large economic losses and violation of the safety regulations.

During the fault tolerant control system design the basic problem is the early detection and identification of possible faults. The main objective here is to use approaches providing the acceptable behaviour of the control system in the case of a critical fault and to shutdown of the system safely. The paper focuses on the first stage of FTC system design which is the modelling of the system. In this work to construct the model of the system the so called State Space Neural Networks (SSNN) are applied. The SSNN model is then used to carry out the fault detection by analysing the residual. The state space model estimates the state vector of the system and in cases when states of the system are measureably available residuals can be calculated as differences between states and their estimates. The methodology presented in the paper is tested on the example of a boiler unit.

The paper is organized as follows. Section 2 presents a general description of the boiler unit and provides information about faulty scenarios considered. The state space neural networks are described in Section 3. Section 4 presents a fault detection algorithm, while experimental results are included in Section 5.

2. BOILER UNIT

The scheme of the boiler unit with process variables marked is presented in Fig.1. The whole system consists of the boiler, storage tank, control valve with positioner, pump and transducers to measure process variables. The specification of process variables is shown in Table 1. The objective of the control system is to keep a required level of the water in the boiler. The control system uses the classical PID controller.

The reference signal is defined as follows:

\[ r(t) = \begin{cases} 
0 & \text{for } t < 240 \\
\frac{1}{10} \sin \left( \frac{2\pi t}{600} \right) + \frac{1}{2} & \text{otherwise}
\end{cases} \tag{1} \]

The boiler unit together with control system was implemented in Matlab/Simulink. Simulations are performed with sample time equal to 0.05. The model of the boiler unit makes it possible to generate a number of faulty situations.

Table 1. Specification of process variables.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Specification</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>CV</td>
<td>control value</td>
<td>0-100 %</td>
</tr>
<tr>
<td>dP</td>
<td>pressure difference on the valve V1</td>
<td>0-275 kPa</td>
</tr>
<tr>
<td>P</td>
<td>pressure before the valve V1</td>
<td>0-500 kPa</td>
</tr>
<tr>
<td>F1</td>
<td>flow (electromagnetic flowmeter)</td>
<td>0-5 m³/h</td>
</tr>
<tr>
<td>F2</td>
<td>flow (Vortex flowmeter)</td>
<td>0-5 m³/h</td>
</tr>
<tr>
<td>L</td>
<td>water level in the boiler</td>
<td>0-0.5 m</td>
</tr>
</tbody>
</table>

3. STATE SPACE INNOVATIONS FORM MODEL

A very important class of dynamic neural networks is the state space neural network. Let \( u(k) \in \mathbb{R}^n \) be the input vector, \( x(k) \in \mathbb{R}^q \) - the output of the hidden layer at time \( k \), and \( y(k) \in \mathbb{R}^m \) - the output vector. Then the state space representation of the neural model is described by the equations

\[ x(k+1) = f(x(k), u(k)) \tag{2} \]
\[ y(k) = Cx(k) \tag{3} \]

where \( f(\cdot) \) is a non-linear function characterizing the hidden layer, and \( C \) is a synaptic weights between hidden
and output neurons. For the space state model the outputs which are fed back are unknown during training. As a result, state space models can be trained only by minimizing the simulation error.

In spite of the fact that state space neural networks seem to be more promising than fully or partially neural networks, in practice a lot of difficulties can be encountered:

- Model states do not approach true process states;
- Wrong initial conditions can deteriorate the performance, especially when short data sets are used for training;
- Training can become unstable;
- The model after training can be unstable;

A very important property of the state space neural network is that it can approximate a wide class of non-linear dynamic systems.

Let consider such a nonlinear dynamic system governed by the following equation:

$$ x(k+1) = g(x(k), u(k)) + f(x(k), u(k)) $$

(4)

where $g$ is a process working at the normal operating conditions, $x(k)$ is the state vector $u(k)$ is the control input vector and $f$ represents a fault affecting the process. A fault function $f$ is a function of both the state and the input and in this way makes it possible to model a wide range of possible faults not only additive ones. Assuming that the system considered is working at the normal operating conditions without any fault occurrence then its model is governed by the following state equation:

$$
\begin{align*}
\dot{x}(k+1) &= \hat{g}(\hat{x}(k), u(k)) \\
\hat{g}(k+1) &= C\dot{x}(k+1)
\end{align*}
$$

(5)

where $\hat{y}$ is the output of the model and $C$ is the output matrix.

Creating state space model according to that equation should allow to easily detect the fault. In the case of skipping the fault function $f$ in the model when the faulty situation occurred should be possible an observation of discrepancy between the output of the system and the model. In that case in this paper were used the residuum of the simulated signal and modelled one.

In cases when states of the system are not measurably available, to calculate the residual signal one should compare the system and the model outputs according to (4):

$$ e(k+1) = y(k+1) - \hat{y}(k+1) $$

(6)

In the ideal case the residual should be equal to zero. In the case of a fault a significant change of the residual value should be observed.

It is necessary to mention that in cases when states of the system are measurably available the residual vector can be defined as a difference between system and model states.

$$ e(k+1) = x(k+1) - \hat{x}(k+1) $$

(7)

This representation is much more powerful than (4) and may be utilized in FTC systems to compensate the fault effect. In the nominal situation model should achieve the value of error close to 0. In case of fault it should be noticed that the $e(k+1)$ will significantly change.

3.1 The NNSYID toolbox

The NNSYSID toolbox (Neural Network SyStem Identification) was implemented by (lit) as a toolbox for the MATLAB environment. It provides the function NNSSIF which is an implementation of the state space neural network innovation form as follows:

$$
\begin{align*}
x(k+1) &= g(x(k), u(k), e(k)) \\
y(k+1) &= Cx(k+1)
\end{align*}
$$

(8)

When assuming the network is trained properly, in nominal situation $e(k)$ is equal 0 and makes possible to use the next function from the toolbox which is the NNSIMUL function. This function is described as it can simulate a neural network model of a dynamic system from a sequence of controls alone (not using observed outputs). The simulated output is compared to the observed output. For state space models the residuals in the function are set to 0. Which makes (8) exactly same as (5) and makes possible to use that function to the fault detection based on (6).

4. FAULT DETECTION

To evaluate residuals and to obtain information about faults, simple thresholding can be applied. If residuals are smaller than the threshold value, a process is considered to be healthy, otherwise it is faulty. For fault detection, the residual must meet the ideal condition being zero in the fault-free case and different from zero in the case of a fault. In practice, due to modelling uncertainty and measurement
noise, it is necessary to assign thresholds larger than zero in order to avoid false alarms. This operation causes a reduction in fault detection sensitivity. Therefore, the choice of the threshold is only a compromise between fault decision sensitivity and false alarm rate.

In order to select the threshold, in this paper method of \( \sigma \)-standard deviation is used. Assuming that the residual is an \( N(m, v) \) random variable thresholds are assigned to the values:

\[
T = m \pm \sigma v
\]  

where \( m \) is mean and \( v \) standard deviation of residuals values and \( \sigma \), in the most cases, is equal to 1, 2 or 3. The probability that a sample exceeds the threshold is equal to 0.15866 for \( \sigma = 1 \), 0.02275 for \( \sigma = 2 \) and 0.00135 for \( \sigma = 3 \).

### 4.1 Selecting the threshold

Based on residuals collected from simulation which are shown in Fig.2, mean \( m \) and standard deviation \( v \) is calculated as follows:

\[
m = \frac{1}{N} \sum_{i=1}^{N} r_i = 0.0016
\]  

\[
v = \frac{1}{N-1} \sum_{i=1}^{N} (r_i - m)^2 = 0.0093
\]

Fig. 2. Simulation residuals used for selecting of threshold.

Table 2. Thresholds for different \( \sigma \) values.

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>threshold range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.0077 ÷ 0.0109</td>
</tr>
<tr>
<td>2</td>
<td>-0.0170 ÷ 0.0202</td>
</tr>
<tr>
<td>3</td>
<td>-0.0263 ÷ 0.0295</td>
</tr>
</tbody>
</table>

### 5. EXPERIMENTS

The experiments were divided into two parts. The first group of experiments was aimed on creating the optimal structure of the neural network model. For the training purposes the training set based on the data generated from the simulator of the boiler unit implemented in Simulink Fig. 3. Optimal structure of the state space neural network model was created. In the experiments the best performance was achieved for the network with 5 neurons in the hidden layer with hyperbolic tangent activation function. Designed model has 3 output states (third order model). The neural network model structure is presented on Fig.3. The modelling efficiency for the training data is shown in Fig.4.

Fig. 3. Optimal structure of the state space neural network model

Fig. 4. Simulation of model with training data.

#### 5.1 Fault detection

With well trained neural network model it was possible to start the experiments with the different faults situations. In simulation the fault was introduced to boiler on the 600th time sample and stopped on the 1000th. The boiler unit is in the nominal state on the 500th time sample. The desired water level in the boiler unit is set on the value 0.25, and the PID controller is adjusting the \( CV \) to satisfy this. Simulated were 4 single-faults. As representative were chosen 2 which simulations of system and model outputs are presented on the Fig.5 and Fig.6. There can be easily noticed that on the fault occurrence in both situations system’s behaviour differs from trained model. For purpose of fault detection residuals are calculated (6). Next step is calculating performance indices for different
values of $\varsigma$ used in selecting thresholds. In this paper are used detection time $t_d$, false detection rate $r_{fd}$ and true detection rate $r_{td}$. Results are shown in Table 3.

Table 3. Performance indices for fault examples.

<table>
<thead>
<tr>
<th>fault</th>
<th>$\varsigma$</th>
<th>$t_d$</th>
<th>$r_{fd}$</th>
<th>$r_{td}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>1</td>
<td>13</td>
<td>0.0768</td>
<td>0.9700</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>21</td>
<td>0</td>
<td>0.9500</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>29</td>
<td>0</td>
<td>0.9300</td>
</tr>
<tr>
<td>f2</td>
<td>1</td>
<td>27</td>
<td>0.0768</td>
<td>0.9350</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>42</td>
<td>0</td>
<td>0.8975</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>57</td>
<td>0</td>
<td>0.8600</td>
</tr>
<tr>
<td>f3</td>
<td>1</td>
<td>6</td>
<td>0.0768</td>
<td>0.9875</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6</td>
<td>0</td>
<td>0.9875</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>6</td>
<td>0</td>
<td>0.9600</td>
</tr>
<tr>
<td>f4</td>
<td>1</td>
<td>18</td>
<td>0.0768</td>
<td>0.9575</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>30</td>
<td>0</td>
<td>0.9275</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>42</td>
<td>0</td>
<td>0.8975</td>
</tr>
</tbody>
</table>

Examples of crossing different thresholds are shown on the Fig.7 and Fig.8.

6. CONCLUSION

As was shown the neural network based state space innovations form model of a dynamic system can be used to the fault detection very well. In the both presented faulty situation model behaved as expected and with use of the threshold made possible to discover the fault. Al tough there was a lot of simplifying conditions of research (as : only one input in form of control value, static desired water level in storage tank, only one fault in experiment) it shows the direction of the future work. Next step in researches should be building the scenario of possibly realistic work of the boiler, and simulation of many faults at once. If this kind of the model would also allow to detect the fault within acceptable time limit, it could be very useful to build the fault tolerant control based on the approach of approximating the fault function as shown in (4).
REFERENCES


