DC MOTOR FAULT DIAGNOSIS BY MEANS OF ARTIFICIAL NEURAL NETWORKS

Krzysztof Patan, Józef Korbicz and Gracjan Głowacki
Institute of Control and Computation Engineering, University of Zielona Góra
ul. Podgórna 50, 65-246 Zielona Góra, Poland
k.patan@issi.uz.zgora.pl, j.korbicz@issi.uz.zgora.pl

Keywords: Neural networks, DC motor, modelling, density shaping, fault detection, fault isolation, fault identification.

Abstract: The paper deals with a model-based fault diagnosis for a DC motor realized using artificial neural networks. The considered process was modelled by using a neural network composed of dynamic neuron models. Decision making about possible faults was performed using statistical analysis of a residual. A neural network was applied to density shaping of a residual, and after that, assuming a significance level, a threshold was calculated. Moreover, to isolate faults a neural classifier was developed. The proposed approach was tested in a DC motor laboratory system at the nominal operating conditions as well as in the case of faults.

1 INTRODUCTION

Electrical motors play a very important role in the safe and efficient work of modern industrial plants and processes. An early diagnosis of abnormal and faulty states renders it possible to perform important preventing actions and it allows one to avoid heavy economic losses involved in stopped production, replacement of elements or parts (Patton et al., 2000). To keep an electrical machine in the best condition, a several techniques like fault monitoring or diagnosis should be implemented. Conventional DC motors are very popular, because they are reasonably cheap and easy to control. Unfortunately, their main drawback is the mechanical collector which has only a limited life spam. In addition, brush sparking can destroy the rotor coil, generate EMC problems and reduce the insulation resistance to an unacceptable limit (Moseler and Isermann, 2000). Moreover, in many cases electrical motors operate in the closed-loop control, and small faults often remain hidden by the control loop. Only if the whole device fails the failure becomes visible. Therefore, there is a need to detect and isolate faults as early as possible.

Recently, a great deal of attention has been paid to electrical motor fault diagnosis (Moseler and Isermann, 2000; Xiang-Qun and Zhang, 2000; Fuessel and Isermann, 2000). In general, elaborated solutions can be splitted into three categories: signal analysis methods, knowledge-based methods and model-based approaches (Xiang-Qun and Zhang, 2000; Korbicz et al., 2004). Methods based on signal analysis include vibration analysis, current analysis, etc. The main advantage of these approaches is that accurate modelling of a motor is avoided. However, these methods only use output signals of a motor, hence the influence of an input on an output is not considered. In turn, the frequency analysis is time-consuming, thus it is not proper for on-line fault diagnosis. In the case of vibration analysis there are serious problems with noise produced by environment and coupling of sensors to the motor (Xiang-Qun and Zhang, 2000).

Knowledge-based approaches are generally based on expert or qualitative reasoning (Zhang and Ellis, 1991). Several knowledge based fault diagnosis approaches have been proposed. These include the rule-based approaches where diagnostic rules can be formulated from process structure and unit functions, and the qualitative simulation-based approaches. The trouble with the use of such models is that accumulating experience and expressing it as knowledge rules is difficult and time-consuming. Therefore, development of a knowledge-based diagnosis system is generally effort demanding.

Model-based approaches include parameter estimation, state estimation, etc. This kind of methods can be effectively used to on-line diagnosis, but its disadvantage is that an accurate model of a motor is required (Korbicz et al., 2004). An alternative solution can be obtained through artificial intelligence, e.g. neural networks. The self-learning ability and property of modelling nonlinear systems allow one to employ neural networks to model complex, unknown and nonlinear dynamic processes (Frank and Köppen-

Seliger, 1997).

The paper is organized as follows. First, description of the considered DC motor is presented in Section 2. The model-based fault diagnosis concept is discussed in Section 3. The architecture details of the dynamic neural network used as a model of the DC motor is given in Section 4. Next, a density shaping technique used to fault detection and a neural classifier applied to isolation of faults are explained in Section 5. Section 6 reports experimental results.

2 AMIRA DR300 LABORATORY SYSTEM

The AMIRA DR300 laboratory system shown in Fig. 1 is used to control the rotational speed of a DC motor with a changing load. The considered laboratory object consists of five main elements: the DC motor M1, the DC motor M2, two digital increamental encoders and the clutch K. The input signal of the engine M1 is an armature current and the output one is the angular velocity. Available sensors for the output are an analog tachometer on optical sensor, which generates impulses that correspond to the rotations of the engine and a digital incremental encoder. The shaft of the motor M1 is connected with the identical motor M2 by the clutch K. The second motor M2 operates in the generator mode and its input signal is an armature current. Available measuremets of the plant are as follows:

- motor current I_m – the motor current of the DC motor M1,
- generator current I_g – the motor current of the DC motor M2,
- tachometer signal T,

and control signals:

- motor control signal C_m – the input of the motor M1,
- generator control signal C_g – the input of the motor M2.

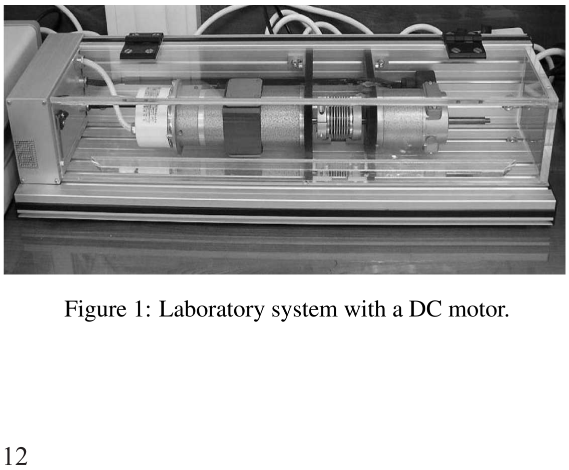

Figure 1: Laboratory system with a DC motor.

Table 1: Laboratory system technical data.

Variable	Value
Motor	
rated voltage	24 V
rated current	2 A
rated torque	0.096 Nm
rated speed	3000 rpm
voltage constant	6.27 mV/rpm
moment of inertia	17.7×10^{-6} Kgm^2
torque constant	0.06 Nm/A
resistance	3.13 Ω
Tachometer	
output voltage	5 mV/rpm
moment of interia	10.6×10^{-6} Kgm^2
Clutch	
moment of inertia	33×10^{-6} Kgm^2
Incremental encoder	
number of lines	1024
max. resolution	4096/R
moment of inertia	1.45×10^{-6} Kgm^2

The technical data of the laboratory system is presented in Table 1. The separately excited DC motor is governed by two differential equations. The electrical subsystem can be described by the equation:

$$u(t) = Ri(t) + L\frac{di(t)}{dt} + e(t) \qquad (1)$$

where $u(t)$ is the motor armature voltage, R – the armature coil resistance, $i(t)$ – the motor armature current, L – the motor coil inductance, and $e(t)$ – the induced electromotive force. The induced electromotive force is proportional to the angular velocity of the motor: $e(t) = K_e\omega(t)$, where K_e stands for the motor voltage constant and $\omega(t)$ – the angular velocity of the motor. In turn, the mechanical subsystem can be derived from the torque balance:

$$J\frac{d\omega(t)}{dt} = T_m(t) - B_m\omega(t) - T_l - T_f(\omega(t)) \qquad (2)$$

where J is the motor moment of inertia, T_m – the motor torque, B_m – the viscous friction torque coefficient, T_l – the load torque, and $T_f(\omega(t))$ – the friction torque. The motor torque $T_m(t)$ is proportional to the armature current: $T_m(t) = K_m i(t)$, where K_m stands for the motor torque constant. The friction torque can be considered as a function of angular velocity and it is assumed to be the sum of Stribeck, Coulumb and viscous components. The viscous friction torque opposes motion and it is proportional to the angular velocity. The Coulomb friction torque is constant at any angular velocity. The Stribeck friction is a nonlinear component occuring at low angular velocities.

The model (1)-(2) has a direct relation to the motor physical parameters, however, the relation between them is nonlinear.There are many nonlinear factors in the motor, e.g. the nonlinearity of the magnetization chracteristic of material, the effect of material reaction, the effect caused by eddy current in the magnet, the residual magnetism, the commutator characteristic, mechanical frictions (Xiang-Qun and Zhang, 2000). Summarizing, a DC motor is a nonlinear dynamic process.

3 MODEL-BASED FAULT DIAGNOSIS

Model-based approaches generally utilise results from the field of control theory and are based on parameter estimation or state estimation (Patton et al., 2000; Korbicz et al., 2004). When using this approach, it is essential to have quite accurate models of a process. Generally, fault diagnosis procedure consists of two separate stages: residual generation and residual evaluation. The residual generation process is based on a comparison between the output of the system and the output of the constructed model. As a result, the difference or so-called residual **r** is expected to be near zero under normal operating conditions, but on the occurrence of a fault, a deviation from zero should appear. Unfortunately, designing mathematical models for complex nonlinear systems can be difficult or even impossible. For the model-based approach, the neural network replaces the analytical model that describes the process under normal operating conditions (Frank and Köppen-Seliger, 1997). First, the network has to be trained for this task. Learning data can be collected directly from the process, if possible or from a simulation model that is as realistic as possible. Training process can be carried out off-line or on-line (it depends on availability of data). After finishing the training, a neural network is ready for on-line residual generation. In order to be able to capture dynamic behaviour of the system, a neural network should have dynamic properties, e.g. it should be a recurrent network (Korbicz et al., 2004; Patan and Parisini, 2005). In turn, the residual evaluation block transforms the residual **r** into the fault vector **f** in order to determine the decision about faults, their location and size, and time of fault occurence.

In general, there are three phases in the diagnostic process: detection, isolation and identification of faults (Patton et al., 2000; Korbicz et al., 2004). In practice, however, the identification phase appears rarely and sometimes it is incorporated into fault isolation. Thus, from practical point of view, the diagnostic process consists of two phases only: fault detection and isolation. The main objective of fault detection is to make a decision whether a fault occur or not. The fault isolation should give an information about fault location. Neural networks can be very usefull in designing the residual evaluation block. In the following sections some solutions for realization fault detection and isolation are discussed.

4 DYNAMIC NEURAL NETWORK

Let us consider the neural network described by the following formulae:

$$\varphi_j^1(k) = \sum_{i=1}^{n} w_{ji}^1 u_i(k) \tag{3}$$

$$z_j^1(k) = \sum_{i=0}^{r_1} b_{ji}^1 \varphi_j^1(k-i) - \sum_{i=1}^{r_1} a_{ji}^1 z_j^1(k-i) \tag{4}$$

$$y_j^1(k) = f(z_j^1(k)) \tag{5}$$

$$\varphi_j^2(k) = \sum_{i=1}^{v_1} w_{ji}^2 y_i^1(k) + \sum_{i=1}^{n} w_{ji}^u u_i(k) \tag{6}$$

$$y_j^2(k) = -\sum_{i=1}^{r_2} a_{ji}^2 y_j^1(k-i) \tag{7}$$

$$y_j(k) = \sum_{i=1}^{v_2} w_{ji}^3 y_j^2(k) \tag{8}$$

where w_{ji}^1, $j = 1, \ldots, v_1$ are the input weights, w_{ji}^2, $j = 1, \ldots, v_2$ – the weights between the output and the first hidden layers, w_{ji}^3, $j = 1, \ldots, m$ – the weights between the output and the second hidden layers, w_{ji}^u, $j = 1, \ldots, v_2$ – the weights between output and input layers, $\varphi_j^1(k)$ and $\varphi_j^2(k)$ are the weighted sums of inputs of the j-th neuron in the hidden and output layers, respectively, $u_i(k)$ are the external inputs to the network, $z_j^1(k)$ is the activation of the j-th neuron in the first hidden layer, a_{ji}^1 and a_{ji}^2 are feedback filter parameters of the j-th neuron in the first and second hidden layers, respectively, b_{ji}^1 are the feed-forward filter parameters of the j-th neuron of the first hidden layer, $f(\cdot)$ – nonlinear activation function, $y_j^1(k)$ and $y_j^2(k)$ – the outputs of the j-th neuron of the first and second hidden layers, respectively, and finally $y_j(k)$, $j = 1, \ldots, v_2$ are the network ouputs. The structure of the network is presented in Fig. 2. The first hidden layer consists of v_1 neurons with infinite impulse response filters of the order r_1 and nonlinear activation functions. The second hidden layer consists of v_2 neurons with finite impulse response filters of the order r_2 and linear activation functions. Neurons of this layer receive excitation not only from the neu-

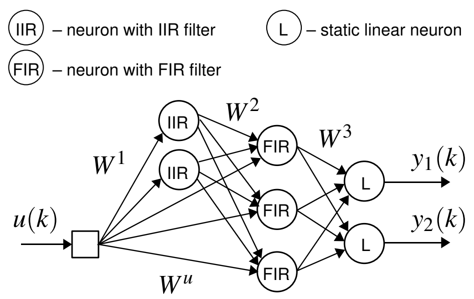

Figure 2: The cascade structure of the dynamic neural network.

rons of the hidden layer but also from the external inputs (Fig. 2). Finaly, the network output is produced by the linear output layer. The network structure (3)-(8) is not a strict feed-forward one. It has a cascade structure. Detailed analysis of this network and its approximation abilities are given in paper (Patan, 2007). Introduction of an additional weight matrix W^u renders it possible to obtain a system, in which the whole state vector is available from the neurons of the output layer. In this way, the proposed neural model can produce a state vector at its output and can serve as a nonlinear observer. This fact is of a crucial importance taking into account training of the neural network. Moreover, this neural structure has similar approximation abilities as a locally recurrent globally feedforward network with two hidden layers designed using nonlinear neuron models with IIR filters, while the number of adaptable parameters is significantly lower (Patan, 2007).

4.1 Network Training

All unknown network parameters can be represented by a vector θ. The objective of training is to find the optimal vector of parameters $\theta^\star$ by minimization of some loss (cost) function:

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

where $J : \mathbb{R}^p \to \mathbb{R}$ represents some loss function to be minimized, p is the dimension of the vector θ, and $C \subseteq \mathbb{R}^p$ is the set of admissible parameters constituted by constraints. To minimize (9) one can use the Adaptive Random Search algorithm (ARS) (Walter and Pronzato, 1996). Assuming that the sequence of solutions $\hat{\theta}_0, \hat{\theta}_1, \ldots, \hat{\theta}_k$ is already appointed, the next point $\hat{\theta}_{k+1}$ is calculated as follows (Walter and Pronzato, 1996):

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \mathbf{r}_k \tag{10}$$

where $\hat{\theta}_k$ is the estimate of the $\theta^\star$ at the k-th iteration, and $\mathbf{r}_k$ is the perturbation vector generated randomly according to the normal distribution $\mathcal{N}(0,\sigma)$. The new point $\hat{\theta}_{k+1}$ is accepted when the cost function $J(\hat{\theta}_{k+1})$ is lower than $J(\hat{\theta}_k)$ otherwise $\hat{\theta}_{k+1} = \hat{\theta}_k$. To start the optimization procedure, it is necessary to determine the initial point $\hat{\theta}_0$ and the variance σ. Let $\theta^\star$ be a global minimum to be located. When $\hat{\theta}_k$ is far from $\theta^\star$, $\mathbf{r}_k$ should have a large variance to allow large displacements, which are necessary to escape the local minima. On the other hand, when $\hat{\theta}_k$ is close $\theta^\star$, $\mathbf{r}_k$ should have a small variance to allow exact exploration of parameter space. The idea of the ARS is to alternate two phases: variance-selection and variance-exploitation (Walter and Pronzato, 1996). During the variance-selection phase, several successive values of σ are tried for a given number of iteration of the basic algorithm. The competing σ_i is rated by their performance in the basic algorithm in terms of cost reduction starting from the same initial point. Each σ_i is computed according to the formula:

$$\sigma_i = 10^{-i}\sigma_0, \quad \text{for} \quad i = 1, \ldots, 4 \tag{11}$$

and it is allowed for $100/i$ iterations to give more trails to larger variances. σ_0 is the initial variance and can be determined, e.g. as a spread of the parameters domain:

$$\sigma_0 = \theta_{max} - \theta_{min} \tag{12}$$

where θ_{max} and θ_{min} are the largest and lowest possible values of parameters, respectively. The best σ_i in terms the lowest value of the cost function is selected for the variance-exploitation phase. The best parameter set $\hat{\theta}_k$ and the variance σ_i are used in the variance-exploitation phase, whilst the algorithm (10) is run typically for one hundred iterations. The algorithm can be terminated when the maximum number of algorithm iteration n_{max} is reached or when the assumed accuracy J_{min} is obtained. Taking into account local minima, the algorithm can be stopped when σ_4 has been selected a given number of times. It means that algorithm got stuck in a local minimum and cannot escape its basin of attraction. Apart from its simplicity, the algorithm possesses the property of global convergence. Moreover, adaptive parameters of the algorithm, cause that a chance to get stuck in local minima is decreased.

5 NEURAL NETWORKS BASED DECISION MAKING

5.1 Fault Detection

In many cases, the residual evaluation is based on the assumption that the underlying data is normally distributed (Walter and Pronzato, 1996). This weak assumption can cause inaccurate decision making and a number of false alarms can be considerable. Thus, in order to select a threshold in a proper way, the distribution of the residual should be discovered or transformation of the residual to another known distribution should be performed. A transformation of a random vector **x** of an arbitrary distribution to a new random vector **y** of different distribution can be realized by maximazing the output entropy of the neural network (Haykin, 1999; Roth and Baram, 1996; Bell and Sejnowski, 1995). Let us consider a situation when a single input is passed through a transforming function $f(x)$ to give an output y, where $f(x)$ is monotonically increasing continuous function satisfying $\lim_{x\to+\infty} f(x) = 1$ and $\lim_{x\to-\infty} f(x) = 0$. The probability density function of y satisfies

$$p_y(y) = \frac{p_x(x)}{|\partial y/\partial x|} \tag{13}$$

The entropy of the output $h(y)$ is given by

$$h(y) = -E\{\log\ p_y(y)\} \tag{14}$$

where $E\{\cdot\}$ stands for the expected value. Substituting (13) into (14) it gives

$$h(y) = h(x) + E\left\{\log\left|\frac{\partial y}{\partial x}\right|\right\} \tag{15}$$

The first term on the right can be considered to be unaffected by alternations in parameters of $f(x)$. Therefore, to maximize the entropy of y one need to take into account the second term only.

Let us define the divergence between two density functions as follows

$$D(p_x(x), q_x(x)) = E\left\{\log\frac{p_x(x)}{q_x(x)}\right\} \tag{16}$$

finally one obtains

$$h(y) = -D(p_x(x), q_x(x)) \tag{17}$$

where $q_x(x) = |\partial(y)/\partial(x)|$. The divergence between true density of x ($p_x(x)$) and an arbitrary one $q_x(x)$ is minimized when entropy of y is maximized. The input probability density function is then approximated by $|\partial y/\partial x|$. The simple and elegant way to adjust network parameters in order to maximize the entropy of y was given in (Bell and Sejnowski, 1995). They used the on-line version of stochastic gradient ascent rule

$$\Delta w = \frac{\partial h(y)}{\partial w} = \frac{\partial}{\partial w}\left(\log\left|\frac{\partial y}{\partial x}\right|\right) = \left(\frac{\partial y}{\partial x}\right)^{-1}\frac{\partial}{\partial w}\left(\frac{\partial y}{\partial x}\right) \tag{18}$$

Considering the logistic transfer function of the form:

$$y = \frac{1}{1+\exp(-u)}, \quad u = wx + b \tag{19}$$

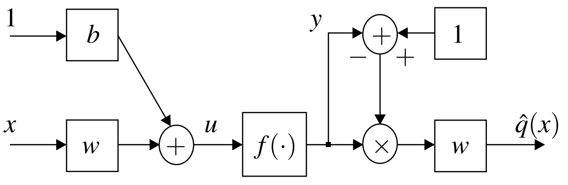

Figure 3: Neural network for density calculation.

where w is the input weight and b is the bias weight, and applying (18) to (19) finally one obtains

$$\Delta w = \frac{1}{w} + x(1 + 2y) \tag{20}$$

Using a similar reasoning, a rule for the bias weight parameter can be derived

$$\Delta b = 1 - 2y \tag{21}$$

After training, the estimated probability density function can be calculated using scheme shown in Fig. 3. In this case, an output threshold of the density shaping scheme (Fig. 3) can be easily calculated by the formula:

$$T_q = |0.5\alpha(1 - 0.5\alpha)w| \tag{22}$$

where α is the confidence level.

5.2 Fault Isolation

Transformation of the residual **r** into the fault vector **f** can be seen as a classification problem. For fault isolation, it means that each pattern of the symptom vector **r** is assigned to one of the classes of system behaviour $\{f_0, f_1, f_2, \dots, f_n\}$. To perform fault isolation, the well-known static multi-layer perceptron network can be used. In fact, the neural classifier should map a relation of the form $\Psi : \mathbb{R}^n \to \mathbb{R}^m : \Psi(\mathbf{r}) = \mathbf{f}$.

5.3 Fault Identification

When analytical equations of residuals are unknown, the fault identification consists in estimating the fault size and time of fault occurence on the basis of residual values. An elementary index of the residual size assigned to the fault size is the ratio of the residual value r_j to suitably assigned threshold value T_j. This threshold can be calculated using (22). In this way, the fault size can be represented as the mean value of such elementary indices for all residuals as follows:

$$s(f_k) = \frac{1}{N}\sum_{j:r_j\in R(f_k)}\frac{r_j}{T_j} \tag{23}$$

where $s(f_k)$ represents the size of the fault f_k, $R(f_k)$ – the set of residuals sensitive to the fault f_k, N – the size of the set $R(f_k)$.

5.4 Evaluation of the Fdi System

The benchmark zone is defined from the benchmark start-up time t_{on} to the benchmark time horizon t_{hor} (Fig. 4). Decisions before the benchmark start-up t_{on} and after the benchmark time horizon t_{hor} are out of interest. The time of fault start-up is represented by t_{from}. When a fault occurs in the system, a residual should deviate from the level assigned to the fault-free case (Fig. 4). The quality of the fault detection system can be evaluated using a number of performance indexes (Patan and Parisini, 2005):

- Time of fault detection t_{dt} – a period of time needed for detection of a fault measured from t_{from} to a permanent, true decision about a fault, as presented in Fig. 4. As one can see there, the first three true decisions are temporary ones and are not taken into account during determining t_{dt}.
- False detection rate r_{fd}:

$$r_{fd} = \frac{\sum_i t^i_{fd}}{t_{from} - t_{on}}, \tag{24}$$

  where t^i_{fd} is the period of i-th false fault detection. This index is used to check the system in the fault-free case. Its value shows a percentage of false alarms. In the ideal case (no false alarms) its value should be equal to 0.
- True detection rate r_{td}:

$$r_{td} = \frac{\sum_i t^i_{td}}{t_{hor} - t_{from}}, \tag{25}$$

  where t^i_{td} is the period of i-th true fault detection. This index is used in the case of faults and describes efficiency of fault detection. In the ideal case (fault detected immediately and surely) its value is equal to 1.

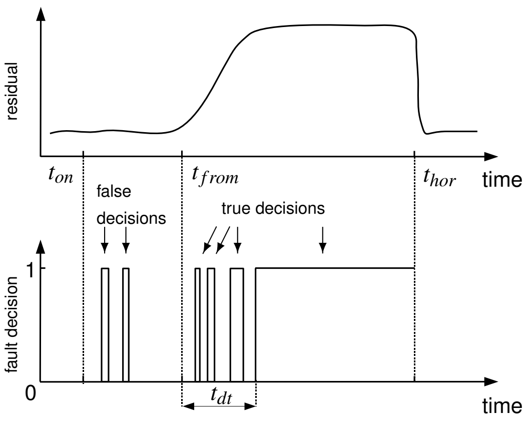

Figure 4: Illustration of performance indexes.

6 EXPERIMENTS

The motor described in Section 2 works is a closed loop control with the PI controller. It is assumed that load of the motor is equal to 0. The objective of the system control is to keep the rotational speed at the constant value equal to 2000. Additionally, it is assumed that the reference value is corrupted by additive noise.

6.1 Motor Modelling

A separately excited DC motor was modelled by using dynamic neural network presented in Section 4. The model of the motor was selected as follows:

$$T = f(C_m) \tag{26}$$

The following input signal was used in experiments:

$$C_m(k) = 3\sin(2\pi 0.017k) + 3\sin(2\pi 0.011k - \pi/7) + 3\sin(2\pi 0.003k + \pi/3) \tag{27}$$

The input signal (27) is persistantly exciting of the order 6. Using (27) a learning set containig 1000 samples was formed. The neural network model (3)-(8) had the following structure: one input, 3 IIR neurons with the first order filters and hyperbolic tangent activation functions, 6 FIR neurons with the first order filters and linear activation functions, and one linear output neuron. Training process was carried out for 100 steps using the ARS algorithm with initial variance $\sigma_0 = 0.1$. The outputs of the neural model and separately excited motor generated for another 1000 testing samples are depicted in Fig. 5. The efficiency of the neural model was also checked during the work of the motor in the closed-loop control. The results are presented in Fig. 6. After transitional oscilations, the neural model settled at a proper value. This gives a strong argument that neural model mimics the behaviour of the DC motor pretty well.

6.2 Fault Detection

Two types of faults were examined during experiments:

- f_{1_i} – tachometer faults were simulated by increasing/decreasing rotational speed by $\pm 25\%$, $\pm 10\%$ and $\pm 5\%$,
- f_{2_i} – mechanical faults were simulated by increasing/decreasing motor torque by $\pm 40\%$, $\pm 20\%$.

In result, a total of 10 faulty situation were investigated. Using the neural model of the process, a residual signal was generated. This signal was used to

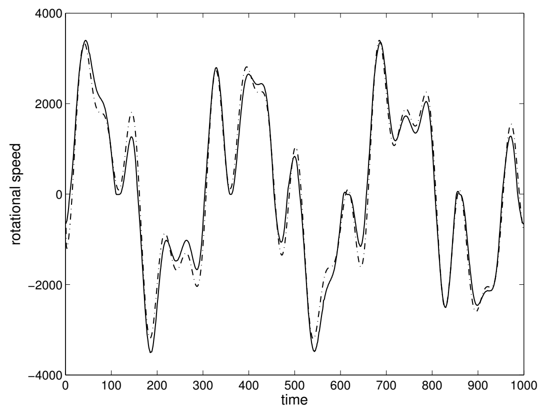

Figure 5: Responses of the motor (solid) and neural model (dash-dot) – open-loop control.

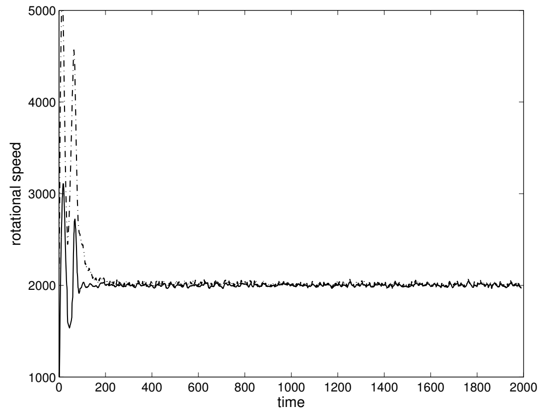

Figure 6: Responses of the motor (solid) and neural model (dash-dot) – closed-loop control.

train another neural network to approximate a probability density function of the residual. Training process was carried out on-line for 100000 steps using unsupervised learning described in Section 5.1. The final network parameters were: $w = -711,43$ and $b = -1.26$. Cut off values determined for significance level $\alpha = 0.05$ had values $x_l = -0.007$ and $x_r = 0,003$ and the threshold was equal to $T = 17,341$. In order to perform decision about faults, and to determine detection time t_{dt}, a time window with the length $n = 50$ was used. If during the following n time steps the residual exceeded the threshold then a fault was signalled. Application of time-window prevents the situation when a temporary true detection will signal a fault (see Fig. 4). The results of fault detection are presented in the second and third columns of Table 2. All faults were reliably detected except fault f_{1_6}. In this case, the changes simulated on tachometer sensor were poorly observed in the residual.

Table 2: Performance indices.

Fault	r_{td}	t_{dt}	R [%]	s
f_{1_1}	0.9995	2035	99	19.51
f_{1_2}	0.9995	2000	82	12.60
f_{1_3}	0.9988	2097	56	5.78
f_{1_4}	0.9395	2086	84	5.45
f_{1_5}	0.9923	2142	62	4.07
f_{1_6}	0.5245	undetected	54	1.48
f_{2_1}	0.9998	2005	100	52.69
f_{2_2}	1.0	2000	100	29,24
f_{2_3}	0.9997	2005	96	27.30
f_{2_4}	0.9993	2008	97	14.56

6.3 Fault Isolation

Fault isolation can be considered as a classification problem where a given residual value is assigned to one of the predefined classes of system behaviour. In the case considered here, there is only one residual signal and 10 different faulty scenarios. To perform fault isolation the well-known multilayer perceptron was used. The neural network had one input (residual signal) and 4 outputs (each class of system behaviour was coded using 4-bit representation). The learning set was formed using 100 samples per each faulty situation and 100 samples representing the fault-free case, then the size of the learning set was equal to 1100. As the well performing neural classifier, the network with 15 hyperbolic tangent neurons in the first hidden layer, 7 hyperbolic tangent neurons in the second hidden layer, and 4 sigmoidal output neurons was selected. The neural classifier was trained for 500 steps using the Levenberg-Marquardt method. Additionally, the real-valued response of the classifier was transformed to the binary one. The simple idea is to calculate a distance between the classifier output and each predefined class of system behaviour. As a result, a binary representation giving the shortest distance is selected as a classifier binary output. This transformation can be represented as follows:

$$j = \arg \min_i ||x - K_i||, \quad i = 1, \ldots, N_K \qquad (28)$$

where x is the real-valued output of the classifier, K_i – the binary representation of the i-th class, N_K – the number of predefined classes of system behaviour, and $||\cdot||$ – the Euclidean distance. Then, the binary representation of the classifier can be determined in the form $\bar{x} = K_j$. Recognition accuracy (R) results are presented in the fourth column of Table 2. The worst results 56% and 62% were obtained for the faults f_{1_3} and f_{1_6}, respectively. The fault f_{1_3} was frequently recognized as the fault f_{1_5}. In spite of misrecognizing, this fault was detected as a faulty situation. Quite different situation was observed for the classification of fault f_{1_6}. This fault, in majority of cases, was classified as the normal operating conditions, thus it cannot be either detected or isolated properly.

6.4 Fault Identification

In this work, the objective of fault identification was to estimate the size of detected and isolated faults. The sizes of faults were calculated using (23). The results are shown in the last column of Table 2. Analyzing results, one can observe that quite large values were obtained for the faults f_{2_1}, f_{2_2} and f_{2_3}. That is the reason that for these three faults true detection rate and recognition accuracy had maximum or close to maximum values. Another group is formed by the faults f_{1_1}, f_{1_2} and f_{2_4} possessing the similar values of the fault size. The third group of fault consists of f_{1_3}, f_{1_4}, and f_{1_5}. The fault sizes in these cases are distinctly smaller than in the cases already discussed. Also the detection time t_{dt} is relatively longer. In spite of the fact that f_{1_6} was not detected, the size of this fault was also calculated. As one can see the fault size in this case is very small, what explains the problems with its detection and isolation.

7 FINAL REMARKS

In the paper, the neural network based method for the fault detection and isolation of faults in a DC motor was proposed. Using the novel structure of dynamic neural network, quite accurate model of the motor was obtained what rendered it possible detection, isolation or even identification of faults. The approach was successfully tested on a number of faulty scenarios simulated in the real plant. The achieved results confirm usefullness and effectiveness of neural networks in designing fault detection and isolation systems. It should be pointed out that presented solution can be easily applied to on-line fault diagnosis.

ACKNOWLEDGEMENTS

This work was supported in part by the Ministry of Science and Higher Education in Poland under the grant *Artificial Neural Networks in Robust Diagnostic Systems*

G. Głowacki is a stypendist of the Integrated Regional Operational Programme (Measure 2.6: Regional innovation strategies and the transfer of knowledge) co-financed from the European Social Fund.

REFERENCES

Bell, A. J. and Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7:1129–1159.

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

Fuessel, D. and Isermann, R. (2000). Hierarchical motor diagnosis utilising structural knowledge and a self-learning neuro-fuzzy scheme. *IEEE Trans. Industrial Electronics*, 47:1070–1077.

Haykin, S. (1999). *Neural Networks. A comprehensive foundation, 2nd Edition*. Prentice-Hall, New Jersey.

Korbicz, J., Kościelny, J. M., Kowalczuk, Z., and Cholewa, W., editors (2004). *Fault Diagnosis. Models, Artificial Intelligence, Applications*. Springer-Verlag, Berlin.

Moseler, O. and Isermann, R. (2000). Application of model-based fault detection to a brushless dc motor. *IEEE Trans. Industrial Electronics*, 47:1015–1020.

Patan, K. (2007). Approximation ability of a class of locally recurrent globally feedforward neural networks. In *Proc. European Control Conference, ECC 2007, Kos, Greece*. accepted.

Patan, K. and Parisini, T. (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.

Patton, R. J., Frank, P. M., and Clark, R. (2000). *Issues of Fault Diagnosis for Dynamic Systems*. Springer-Verlag, Berlin.

Roth, Z. and Baram, Y. (1996). Multidimensional density shaping by sigmoids. *IEEE Trans. Neural Networks*, 7(5):1291–1298.

Walter, E. and Pronzato, L. (1996). *Identification of Parametric Models from Experimental Data*. Springer, London.

Xiang-Qun, L. and Zhang, H. Y. (2000). Fault detection and diagnosis of permanent-magnet dc motor based on parameter estimation and neural network. *IEEE Trans. Industrial Electronics*, 47:1021–1030.

Zhang, J., P. D. R. and Ellis, J. E. (1991). A self-learning fault diagnosis system. *Transactions of the Institute of Measurements and Control*, 13:29–35.