Neural Comput & Applic (2014) 24:431–439
DOI 10.1007/s00521-012-1239-0

ORIGINAL ARTICLE

The method of solution of equations with coefficients that contain measurement errors, using artificial neural network

Konrad Zajkowski

Received: 5 May 2012 / Accepted: 17 October 2012 / Published online: 2 November 2012

Abstract This paper presents an algorithm for solving N-equations of N-unknowns. This algorithm allows to determine the solution in a situation where coefficients A_i in equations are burdened with measurement errors. For some values of A_i (where $i = 1,\ldots, N$), there is no inverse function of input equations. In this case, it is impossible to determine the solution of equations of classical methods.

Keywords Artificial neural network · Measurement errors · Induction motor model · Parameter identification

1 Introduction

Mathematical models that describe electric dependencies in the receiver tested are built from discrete components. For a full description of such a model, it is required to identify the parameters x_i of these elements (see Eq. 1). Most frequently, this identification is carried out indirectly through the measurements of electrical quantities A_i on the object tested [5, 8]. The parameters sought are determined from the mathematical relations (1) that describe the object.

$$\begin{cases} A_1 = f_1(y_1, y_2, \ldots, y_i, \ldots, y_N) \\ A_2 = f_2(y_1, y_2, \ldots, y_i, \ldots, y_N) \\ \vdots \\ A_i = f_i(y_1, y_2, \ldots, y_i, \ldots, y_N) \\ \vdots \\ A_N = f_N(y_1, y_2, \ldots, y_i, \ldots, y_N) \end{cases} \quad (1)$$

K. Zajkowski (✉)
Division of Electrotechnics and Electronics,
Technical University of Koszalin, 15-17 Raclawicka St.,
75-620 Koszalin, Poland
e-mail: konrad.zajkowski@tu.koszalin.pl

where f_i certain functions depending on the model, A_i the values measured, y_i parameters that describe the model.

The classic method to solve Eq. (1) consists in determining inverse functions (2). Measurement inaccuracies that are contained in A_i are transferred to parameters y_i to be determined.

$$\begin{cases} y_1 = g_1(A_1, A_2, \ldots, A_N) \\ y_2 = g_2(A_1, A_2, \ldots, A_N) \\ \vdots \\ y_N = g_N(A_1, A_2, \ldots, A_N) \end{cases} \quad (2)$$

In some cases, the determination of Eq. (2) may not be possible [14, 16]. This means that for adopted coefficients A_i, there are no inverse functions g_i. Eq. (2) are determined for the values of environment A_i and which contain measurement errors. In this case, approximate solutions are sought which satisfy Relation (3).

$$|A_i - f_i(y_1, y_2, \ldots, y_N)| \approx 0 \quad (3)$$

The solution will be close to coefficients A_i.

2 An example of a model for identification

The analysis covered a single phase on an induction motor. The purpose of the analysis is to determine current–voltage dependences on the terminals of one motor phase. These relationships can be determined from the model that consists of serially connected elements: R_s, L_s i e_s (Fig. 1).

Coefficients R_S, L_S, E_m, φ_{es}, α that are being sought represent many of the phenomena that occur in the motor and the system that is driven. For example, the inertia of the rotor and the system driven will affect e_s, and the angular velocity will exert an influence on mutual inductances, which are described with L_S. When searching for

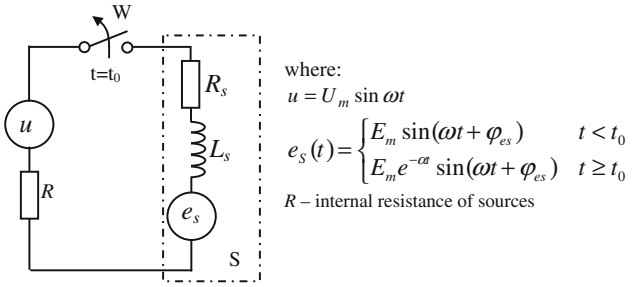


Fig. 1 Electric model of the motor and the power source, where $u = U_m \sin \omega t$, $e_S(t) = \begin{cases} E_m \sin(\omega t + \phi_{es}) & t < t_0 \\ E_m e^{-\alpha t} \sin(\omega t + \phi_{es}) & t \geq t_0 \end{cases}$, R internal resistance of sources

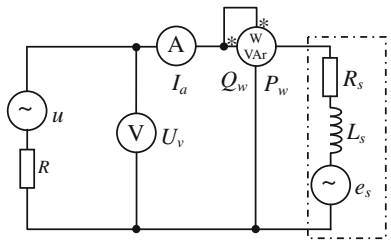


Fig. 2 Measuring circuit for parametric identification

the parameters of the model, the fact is also important that these factors cannot be determined with the engine being stopped. This means that the R_S does not reflect the winding resistance and L_S does not reflect their inductance. The parameters of the model are defined for a constant load on the machine shaft and for constant rotations. When changing the load, the parameters of the model change, as well.

In this situation, the parameters that are being determined cannot in any way be unified. They should be determined for a specific drive train (the motor and the machine driven). These parameters can vary considerably for the same engine with different mechanical properties of the system driven.

The identification of the model consists in searching for E_m, φ_{es}, R_S, and L_S. These parameters can be determined on the receiver [6, 7, 9–11] by making measurements in the steady state (in the case of an induction motor: during operation with a constant load and a constant speed) in the system as shown below (Fig. 2):

According to the model adopted, we know that:

$$e_s = E_s \sqrt{2} \sin(\omega t + \phi_{es}). \tag{4}$$

In the field of complex numbers, the following can be written:

$$\underline{E}_s = E_s e^{j\phi_{es}}, \tag{5}$$

$$\underline{U} = U e^{j\phi_u}. \tag{6}$$

For one mesh, the voltage equation is as follows:

$$\underline{I}_a(R_s + R + jX_s) + \underline{E}_s - \underline{U} = 0, \tag{7}$$

where $X_S = \omega L_S$.

Next, by transforming (7), we determine current $\underline{I}_a$:

$$\underline{I}_a = \frac{\underline{U} - \underline{E}_s}{R_s + R + jX_s} \tag{8}$$

Voltmeter V measures the difference in the supply voltage and in the voltage drop across internal resistance R. Thus, in the field of complex numbers, there will be the following:

$$\underline{U}_V = \underline{U} - \underline{I}_a R. \tag{9}$$

From Eqs. (8) and (9), one can obtain the following:

$$\underline{U}_V = \frac{1}{R_s + R + jX_s} [\underline{U}(R_s + jX_s) + \underline{E}_s R]. \tag{10}$$

Knowing that the forces and the current are equal, respectively:

$$P_w = \mathrm{Re}(\underline{U}_v \underline{I}_a^*), \quad Q_w = \mathrm{Im}(\underline{U}_v \underline{I}_a^*), \quad \underline{I}_a^* = \frac{\underline{U}^* - \underline{E}_s^*}{R_s + R - jX_s}$$

We obtain the following equations:

$$\begin{cases} I_a = \left| \frac{\underline{U} - \underline{E}_s}{R_s + R + jX_s} \right| \\ U_v = \left| \frac{1}{R_s + R + jX_s} [\underline{U}(R_s + jX_s) + \underline{E}_s R] \right| \\ P_w = \mathrm{Re}\left\{ \frac{[\underline{U}(R_s + jX_s) + \underline{E}_s R] \cdot [\underline{U}^* - \underline{E}_s^*]}{(R_s + R)^2 + X_s^2} \right\} \\ Q_w = \mathrm{Im}\left\{ \frac{[\underline{U}(R_s + jX_s) + \underline{E}_s R] \cdot [\underline{U}^* - \underline{E}_s^*]}{(R_s + R)^2 + X_s^2} \right\} \end{cases}. \tag{11}$$

Equation (11) is consistent with (1). The coefficients of the model of the receiver that are obtained from the above equations are not determinable for all the input parameters (U_V, I_a, P_W, Q_W). There are those areas that result from measurement inaccuracies where the system of Eq. (11) has no solutions.

It was found that these coefficients cannot be determined using the Newton's interpolation algorithm [15, 16]. There are no functions that are inverse to Eq. (11), either.

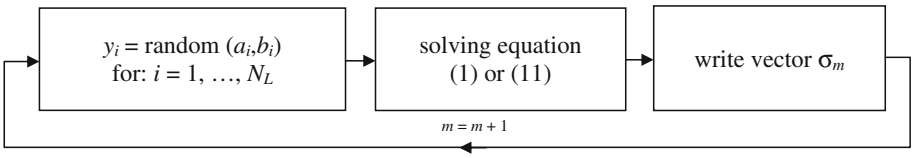
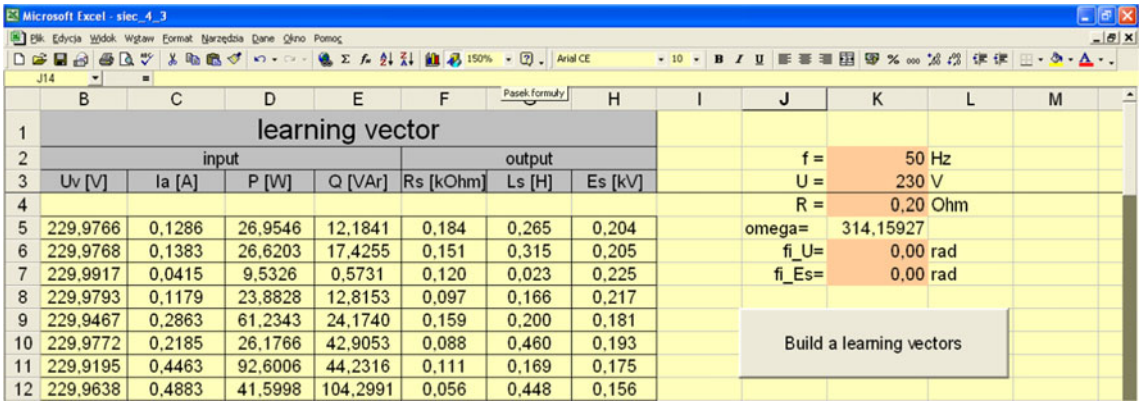


Fig. 3 Construction of learning vectors



	B	C	D	E	F	H	J	K	L	M
1	learning vector									
2	input				output		f =	50	Hz	
3	Uv [V]	Ia [A]	P [W]	Q [VAr]	Rs [kOhm]	Ls [H]	Es [kV]	U =	230	V
4							R =	0,20	Ohm	
5	229,9766	0,1286	26,9546	12,1841	0,184	0,265	0,204	omega=	314,15927	
6	229,9768	0,1383	26,6203	17,4255	0,151	0,315	0,205	fi_U=	0,00	rad
7	229,9917	0,0415	9,5326	0,5731	0,120	0,023	0,225	fi_Es=	0,00	rad
8	229,9793	0,1179	23,8828	12,8153	0,097	0,166	0,217			
9	229,9467	0,2863	61,2343	24,1740	0,159	0,200	0,181	Build a learning vectors		
10	229,9772	0,2185	26,1766	42,9053	0,088	0,460	0,193			
11	229,9195	0,4463	92,6006	44,2316	0,111	0,169	0,175			
12	229,9638	0,4883	41,5998	104,2991	0,056	0,448	0,156			

Fig. 4 The workbook that builds learning vectors

Process time constant $1/\alpha$ that is being sought, and which is mainly related to the inertia of the rotor and the system driven, can be determined experimentally by observing the course of voltage versus time at the motor terminals immediately after commutation.

The neural network was built in a VBA environment in EXCEL.

The script associated with the button in Fig. 4 determines the random values: $d_1 = R_s \in \langle 0.005 \div 0.2 \rangle \text{k}\Omega$, $d_2 = L_s \in \langle 0.005 \div 0.5 \rangle \text{H}$, $d_3 = E_s \in \langle 0.15 \div U \rangle \text{kV}$.

```
d(1) = Int(((200 - 5) * Rnd) + 5) / 1000     ' Rs [kOHm]= 0.005 - 0.200
d(2) = Int(((500 - 5) * Rnd) + 5) / 1000     ' Ls [H] = 0.005 - 0.500
d(3) = Int(((U - 150) * Rnd) + 150) / 1000   ' Es [kV]  = 0.150 - U
```

In [13], the authors proved that amplitude E_S can be equal to amplitude U. In this paper, it was also observed that frequency E_S is similar to the frequency of the mains voltage. It was also noted that phase shift φ_{es} is equal to 0.

In this model, it is assumed that the frequencies of both sources are identical. This assumption does not substantially affect the results of further simulations.

3 Construction of an artificial neural network

Coefficients E_m, R_S, and L_S can be determined from Eq. (11) using a neural network. The network input parameters $x_1 = U_v$ [V], $x_2 = I_a$ [A], $x_3 = P_w$ [W], $x_4 = Q_w$ [VAr] contain measurement errors. Due to the nature of the adopted activation function [1–3], the output neuron of the output layer must be within range $y \in (0, 1)$. The initial values were as follows: $y_1 = R_s$ [kΩ], $y_2 = L_s$ [H], $y_3 = E_s$ [kV].

Training of the network must be for those learning vectors $\sigma = \left[x_1, \ldots, x_{N_0}, \middle| , d_1^{(L)}, \ldots, d_{N_L}^{(L)}\right]$ that do not contain any measurement errors. Learning vectors are constructed from Eqs. (1) or (11) for random values y_1, y_2, y_3 that lie within the set of permissible changes, and which is limited with values a and b [4, 12].

The test vector is built according to Fig. 3 for values y_i that are not contained within the training set.

Further values U_v, I_a, P_w and Q_w are determined from Eq. (11).

After tests of several neural networks, a decision was made to build a neural network with topology (Fig. 5), with one hidden layer. The weights of neurons are determined by back propagation.

Individual neurons in the network are structured according to Fig. 6.

In the network being built, the following indications were accepted:

t	iteration step, $t = 1, 2, \ldots$
$y_i^{(k)}(t)$	ith output of the neuron $N_i^{(k)}$
$y_i^{(L)}(t)$	ith output of the network
k	network layer, $k = 1, \ldots, L$
L	network output layer, the number of network layers
i	neuron number in layer, $i = 1,\ldots, N_k$
$x_j^{(k)}(t)$	input signal in the kth layer
$x_j(t) = x_j^{(1)}(t)$	input of the network
j	number of the input signal in the kth layer, $j = 1,\ldots, N_{k-1}$
N_0	number of inputs to the network
N_k	number of neurons in the kth layer
N_L	number of neurons in the last layer
$s_i^{(k)}(t)$	neuron membrane potential $N_i^{(k)}$ in the kth layer

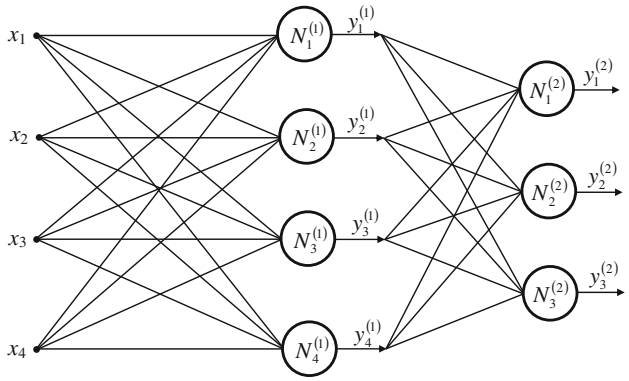


Fig. 5 Structure of the neural network

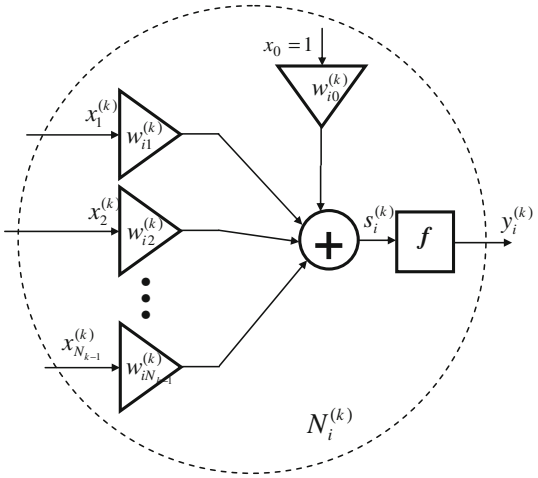


Fig. 6 Diagram of neuron $N_i^{(k)}$

$w_{i,j}^{(k)}(t)$ weight of the jth input of the ith neuron $N_i^{(k)}$ in the kth layer

$d_i^{(L)}(t)$ ith reference signal output from the learning vector

$\varepsilon_i^{(L)}(t)$ error of the ith network output, $\varepsilon_i^{(L)}(t) = d_i^{(L)}(t) - y_i^{(L)}(t)$

η network learning rate

$Q(t)$ error at the output of the network for one reference vector

$Q^*(t)$ error at the output of the network for the entire epoch

The output of neuron $N_i^{(k)}$ (Fig. 6) at time t is described with the following relation:

$$y_i^{(k)}(t) = f\left(s_i^{(k)}(t)\right) \tag{12}$$

while membrane potential $s_i^{(k)}$ is equal to:

$$s_i^{(k)}(t) = \sum_{j=0}^{N_{k-1}} w_{i,j}^{(k)}(t) \cdot x_j^{(k)}(t) \tag{13}$$

The input neuron for $k = 1$ layer is equal to network inputs $x_j(t) = x_j^{(1)}(t)$. Each layer has one input $x_0^{(k)}(t) = 1$. Other inputs are the outputs of the previous layer.

$$x_j^k(t) = \begin{cases} x_j(t) & \text{for } k = 1 \\ y_j^{k-1}(t) & \text{for } k = 2, \ldots, L \\ 1 & \text{for } j = 0, \quad k = 1, \ldots, L \end{cases} \tag{14}$$

The error at the output of the network for one learning vector σ is:

$$Q(t) = \sum_{i=1}^{N_L} \left(\varepsilon_i^L(t)\right)^2 = \sum_{i=1}^{N_L} \left(d_i^L(t) - y_i^L(t)\right)^2 \tag{15}$$

The weights of the individual neuron inputs are determined from the steepest descent rule:

$$w(t+1) = w(t) - \eta \cdot \mathbf{g}(w(t)) \tag{16}$$

where $\mathbf{g}(w(t)) = \left[\frac{\partial Q(t)}{\partial w_1(t)}, \frac{\partial Q(t)}{\partial w_2(t)}, \frac{\partial Q(t)}{\partial w_3(t)}, \ldots, \frac{\partial Q(t)}{\partial w_n(t)}\right]^T$ is the vector gradient.

From Eq. (16), for any weight in any layer, the following is obtained:

$$\begin{aligned} w_{i,j}^{(k)}(t+1) &= w_{i,j}^{(k)}(t) - \eta \frac{\partial Q(t)}{\partial w_{i,j}^{(k)}(t)} \\ &= w_{i,j}^{(k)}(t) + 2\eta \delta_i^{(k)}(t) x_j^{(k)}(t) \end{aligned} \tag{17}$$

Parameter $\delta_i^{(k)}(t)$ is determined differently than for the output layer and the hidden layer:

$$\delta_i^{(k)}(t) = \begin{cases} \varepsilon_i^{(L)}(t) \cdot f'\left(s_i^{(L)}(t)\right) & \text{for } k = L \\ f'\left(s_i^{(k)}(t)\right) \cdot \sum_{m=1}^{N_{k+1}} \delta_m^{(k+1)}(t) w_{i,m}^{(k+1)}(t) & \text{for } k \neq L \end{cases} \tag{18}$$

where $\varepsilon_i^{(L)}(t) = d_i^{(L)}(t) - y_i^{(L)}(t)$.

Network training is carried out by an incremental updating of weights, that is, each time after the entry of a successive learning vector, responses are determined and the weights are modified. The simulation is continued until the total output error for entire epoch $Q^*(t)$ is smaller than the accepted set $Q_{\min}$.

$$Q^*(t) = \sum_{m=1}^{M} Q_m(t) \leq Q_{\text{Min}} \tag{19}$$

where M is the number of learning vectors in the epoch.

The neuron activation function was adopted as a continuous unipolar function of the signum type:

$$f(s_i(t)) = \frac{1}{1 + e^{-\beta \cdot s_i(t)}} \tag{20}$$

where β is the steepness factor.

With low values of coefficient β, the function is usually mild. By increasing β, the plot becomes steeper until the threshold course is obtained.

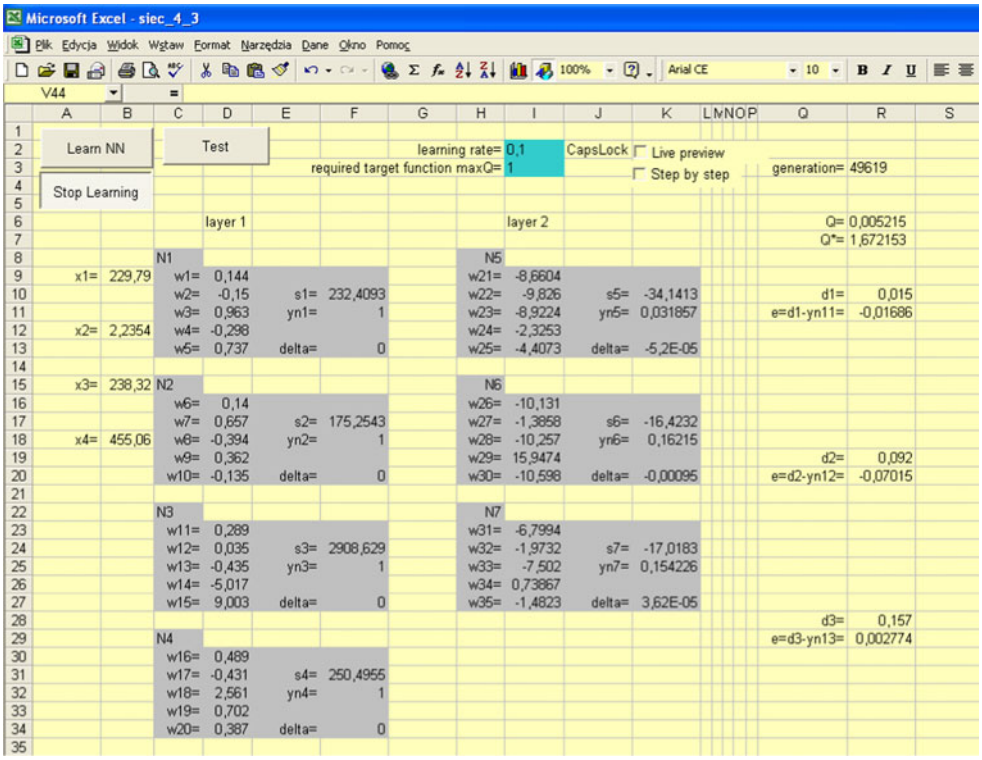


Fig. 7 Sheet for the visualization of the network operation

The derivative of the activation function is as follows:

$$f'(s_i(t)) = \frac{\beta \cdot e^{-\beta \cdot s_i(t)}}{\left(1 + e^{-\beta \cdot s_i(t)}\right)^2} = \beta \cdot f(s_i(t)) \cdot (1 - f(s_i(t))) \qquad (21)$$

The calculation sheet in Fig. 7 allows an observation of the characteristic values of the network tested. Starting of the network training produces a script written in VBA that executes in a loop of a neural network algorithm according to (12) ÷ (21) and the block diagram in Fig. 8.

The start of the algorithm is possible for the weights that are selected at random from range $\langle -1, 1 \rangle$ or the reading stored from the previous simulations (Fig. 9).

4 Learning of the network

The set of learning vectors that form one epoch consists of 200 elements. Owing to the ability to read and write data, it is possible to pause the simulation and to change its parameters during operation [4, 12].

Reading of the stored data allows a continuation of the previously stopped simulation. The window in Fig. 9 retrieves the values from the appropriate data sheet (Fig. 10).

The output values of the neurons (Fig. 8a) are determined by analyzing the neurons in layers starting from the input layer; the output layer comes last.

```
'Layer 1:
For i = 1 To Lneurons1 'All neurons on layer 1
   s(i) = w((i - 1) * Lweights1 + 1) 'weight w_i0 – bias (fig.6)
   For j = 2 To Lweights1 'All inputs in neuron N_i^1
         s(i) = s(i) + x(j - 1) * w((i - 1) * Lweights1 + j) 'neuron membrane potential
   Next j
   yn(i) = fx(beta1, s(i)) 'activation function
Next i
'Layer 2:
For i = Lneurons1 + 1 To Lneurons1 + Lneurons2 'All neurons on layer 2
   s(i) = w(Lweights1 * Lneurons1 + (i - Lneurons1 - 1) * Lweights2 + 1) 'weight w_i0
For j = 2 To Lweights2 'All inputs in neuron N_i^2
      s(i) = s(i) + yn(j - 1) * w(Lweights1 * Lneurons1 + (i - Lneurons1 - 1) * Lweights2 + j)  'neuron membrane potential
Next j
   yn(i) = fx(beta2, s(i)) 'activation function
Next i
```

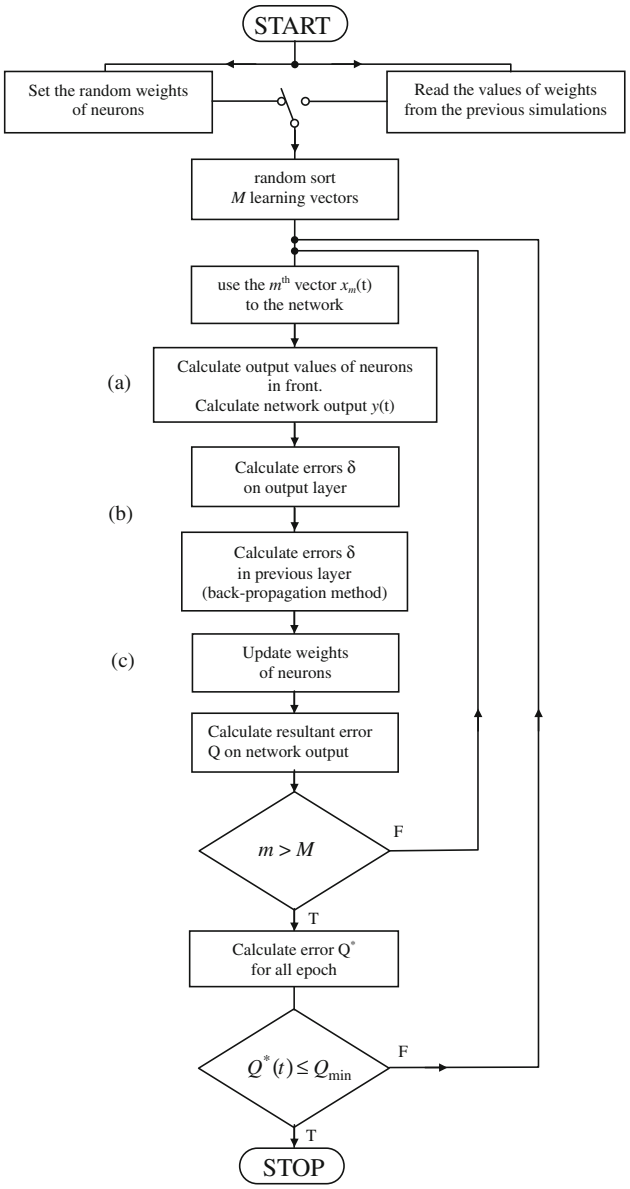


Fig. 8 Block diagram of the network learning algorithm

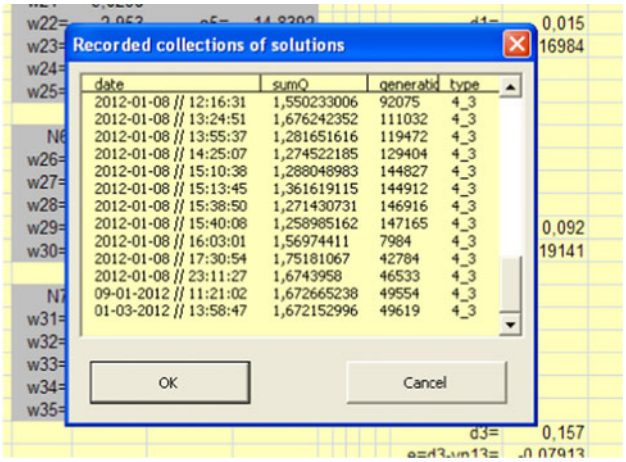


Fig. 9 The reading window of the recorded data

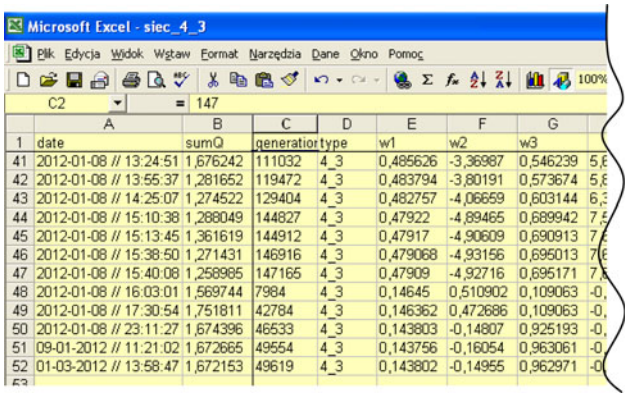


Fig. 10 Sheet with the saved results of the simulation

For all the layers, the steepness factor β (20) of activation function fx was assumed as equal to 0.1.

The determination of value $\delta_i^{(k)}(t)$ (Fig. 8b) shall be in accordance with Formula (18). This determination takes place starting from the output layer; the input layer comes last.!

```
'determination of delta on layer 2:
    For i = 1 To Loutputs
        e(i) = d(i) - yn(Lneurons1 + i)  'determine value of   e = d - y
    Next i
    For i = Lneurons1 + 1 To Lneurons
        delta(i) = e(i - Lneurons1) * fxP(beta2, s(i))  'determine value of
    Next i
'determination of delta on layer 1:
    For i = 1 To Lneurons1
        delta_propag = 0
        For j = 1 To Lneurons2 'propagating delta of the upper layer:
            delta_propag = delta_propag + delta(j + Lneurons1) * w(Lneurons1 * Lweights1 + (j - 1) * Lweights2 + i + 1)
        Next j
        delta(i) = fxP(beta1, s(i)) * delta_propag
    Next i
```

Correction of the values of weights (Fig. 8c) is carried out according to Relation (17).

```
'correction weights for layer 2 - the last:
  For i = Lneurons1 + 1 To Lneurons  'neurons in layer 2:
    k = Lweights1 * Lneurons1 + (i - Lneurons1 - 1) * Lweights2 + 1
    w(k) = w(k) + 2 * mi * delta(i) 'weight for x0
    For j = 2 To Lweights2 'input neurons
      k = Lweights1 * Lneurons1 + (i - Lneurons1 - 1) * Lweights2 + j
      w(k) = w(k) + 2 * mi * delta(i) * yn(j - 1)     'weights other inputs
    Next j
  Next i
'correction weights for layer 1:
  For i = 1 To Lneurons1 'neurons in layer 1:
    k = (i - 1) * Lweights1 + 1
    w(k) = w(k) + 2 * mi * delta(i) 'weight for x0
    For j = 2 To Lweights1 'input neurons
      k = (i - 1) * Lweights1 + j
      w(k) = w(k) + 2 * mi * delta(i) * x(j - 1)     'weights other inputs
    Next j
  Next i
```

Network learning factor η from Formula (17) was adopted on the first stage of the simulation as being constant and equal to 0.1. After an analysis of ca. 70,000 epochs, the value of target function $Q^*(t)$, which was calculated in accordance with Formula (19), began to oscillate on the level of 1.42. A decrease in $Q^*(t)$ occurred only after a reduction in network learning rate η. The correct procedure for the network training should provide for an ability to change this ratio during the analysis (Fig. 11).

Oscillations around the optimal solution are manifested with a momentary increase in the value of $Q^*(t)$.

$$Q^*(t) \geq Q^*(t-1) \tag{22}$$

Once the required value of $Q^*(t)$ from Eq. (19) has been reached, the network test is performed (Fig. 12).

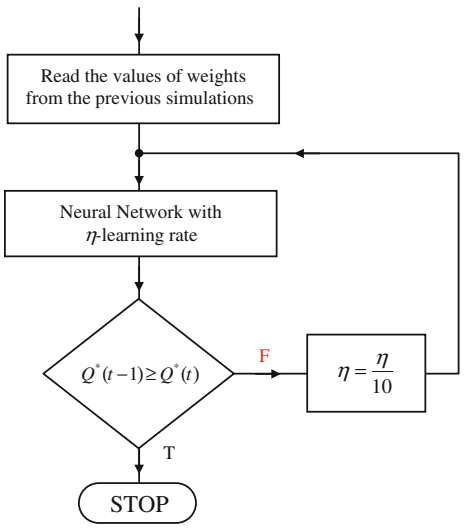


Fig. 11 The method to reduce the coefficient of network learning

5 Network test

The network test consists in determining the values of U_V, I_a, P_W, and Q_W from Relation (11). These values are then substituted into the neural network input, whose solution is R_S, L_S, and E_S. The window in Fig. 12 also allows a determination of the network's solution for a selected set of weights.

Table 1 illustrates the network test for randomly selected values of R_S, L_S, and E_S.

The relative error for all the output neurons for the randomly adopted input vectors is:

$$\delta_1 = \frac{y_1 - R_S}{R_S}, \quad \delta_2 = \frac{y_2 - L_S}{L_S}, \quad \delta_3 = \frac{y_3 - E_S}{E_S} \tag{23}$$

The total network error for the accepted values of R_S, L_S, and E_S are:

$$\delta = \left|\frac{y_1 - R_S}{R_S}\right| + \left|\frac{y_2 - L_S}{L_S}\right| + \left|\frac{y_3 - E_S}{E_S}\right|. \tag{24}$$

The percentage error made by the network is determined from the largest error (the top bar in the chart in Fig. 13), and it is equal to 26.8 %.

6 Conclusions

The large error value is shown for the input values that occur least frequently in the training set. An improved performance is possible by enlarging the training set or by reducing the range of acceptable changes of the values being sought.

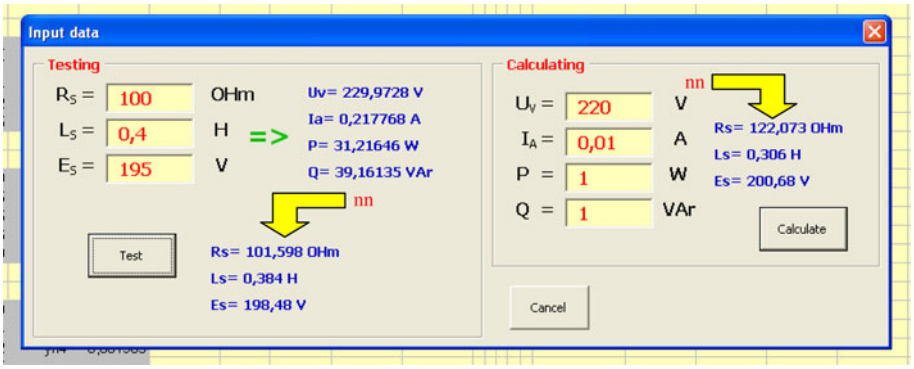


Fig. 12 The window for testing and results analysis

Table 1 Verification of the network error

Random values			Network input				Network output			Relative error			
			x_1	x_2	x_3	x_4	y_1	y_2	y_3	δ_1	δ_2	δ_3	δ
R_S [Ω]	L_S [H]	E_S [V]	U_V [V]	I_a [A]	P [W]	Q [VAr]	R'_S [Ω]	L'_S [H]	E'_S [V]	–	–	–	–
20	0.2	180	229.95	0.7576	53.21	165.88	19.721	0.210	168.626	−0.014	0.050	−0.063	0.127
151	0.315	193	229.97	0.2048	39.395	25.7897	147.083	0.354	192.849	−0.026	0.124	−0.001	0.151
150	0.09	197	229.96	0.2159	48.794	9.18694	152.107	0.093	197.019	0.014	0.033	0.0001	0.048
147	0.3	197	229.97	0.1888	36.563	23.415	147.413	0.352	192.976	0.003	0.173	−0.020	0.197
47	0.46	193	229.99	0.2434	17.368	53.211	41.373	0.409	195.946	−0.120	−0.111	0.015	0.246
150	0.09	193	229.95	0.2421	54.707	10.301	151.590	0.092	196.866	0.011	0.022	0.020	0.053
45	0.448	190	229.98	0.2706	19.015	59.256	41.182	0.408	195.771	−0.085	−0.089	0.030	0.205
145	0.4	190	229.97	0.2083	36.218	31.353	145.773	0.357	192.810	0.005	−0.108	0.015	0.128
50	0.03	160	229.73	1.3705	309.422	58.163	58.641	0.026	159.829	0.173	−0.133	−0.001	0.307
150	0.1	200	229.96	0.1955	44.0056	9.20584	152.394	0.093	197.103	0.016	−0.070	−0.014	0.100
140	0.317	189	229.96	0.2384	44.6931	31.755	146.133	0.354	192.544	0.044	0.117	0.019	0.179
40	0.317	195	229.98	0.3259	28.036	69.507	40.416	0.402	195.029	0.010	0.268	0.0002	0.279
135	0.4	195	229.97	0.1896	31.937	29.691	143.751	0.358	192.972	0.065	−0.105	−0.010	0.180
145	0.3	190	229.96	0.2311	44.568	28.936	146.443	0.353	192.636	0.010	0.177	0.014	0.201
140	0.4	190	229.97	0.2125	36.378	32.615	145.157	0.356	192.790	0.037	−0.110	0.015	0.162

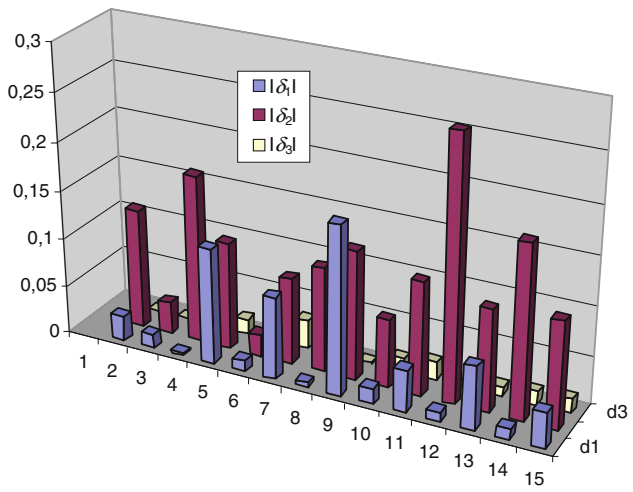


Fig. 13 Mistake made by the network

Owing to the method presented of the selection of the electrical model parameters from the values that are measured on the receiver, it is not required to build any complex physical and electrical dependences. The engineering method of voltage, current, and power measurement allows one to determine the parameters of the model for constant electrical and mechanical conditions in the engine. The method presented is particularly useful in situations where measurement errors make it impossible to solve Eq. (2).

Building of a network with the use of the VBA environment is relatively simple. It requires the knowledge of the language basics. An important advantage of this approach is the ability to build its own networks of any topology. The design loop iteration depends largely on how one defines those variables that describe the network.

In the present solution, the individual variables occupy adjacent bytes of the memory. A sample definition of the variable holding the weights of neurons is:

Public w(1 To Lweights)As Single

where Lweights is the number of weights of all neurons.This solution facilitates the construction of a loop program, but special attention is to be paid to assigning the weight number with the neuron number.

An alternative is to build one's own variable (using the opportunity to build one's own type of variables) that represents the neuron, and then group all the parameters that describe the type of the neuron in this variable. This approach will make the program more transparent, but there are problems in the construction of iterative loops. This will make the source code longer and will require more CPU load.

References

1. Duer S (2009) Artificial neural network-based technique for operation process control of a technical object. Def Sci J 59(3):305–313. http://publications.drdo.gov.in/ojs/index.php/dsj
2. Duer S (2010) Diagnostic system with an artificial neural network in diagnostics of an analogue technical object. Neural Comput Appl 19(1):55–60
3. Duer S (2010) Diagnostic system for the diagnosis of a reparable technical object, with the use of an artificial neural network of RBF type. Neural Comput Appl 19(5):691–700
4. Duer S, Duer R (2010) Diagnostic system with an artificial neural network which determines a diagnostic information for the servicing of a reparable technical object. Neural Comput Appl 19(5):755–766
5. Duer S (2010) Investigation of the operation process of a repairable technical object in an expert servicing system with an artificial neural network. Neural Comput Appl 19(5):767–774
6. Duer S (2011) Qualitative evaluation of the regeneration process of a technical object in a maintenance system with an artificial neural network. Neural Comput Appl 20(5):741–752
7. Duer S (2010) Expert knowledge base to support the maintenance of a radar system. Def Sci J 60(5):531–540. http://publications.drdo.gov.in/ojs/index.php/dsj
8. Duer S (2011) Modelling of the operation process of repairable technical objects with the use information from an artificial neural network. Expert Syst Appl 38:5867–5878. http://dx.doi.org/10.1016/j.eswa.2010.11.036
9. Duer S (2011) Assessment of the quality of decisions worked out by an artificial neural network which diagnoses a technical object. Neural Comput Appl. doi:10.1007/s00521-011-0725-0. http://www.springerlink.com/openurl.asp?genre=article&id
10. Duer S (2011) Examination of the reliability of a technical object after its regeneration in a maintenance system with an artificial neural network. Neural Comput Appl. doi:10.1007/s00521-011-0723-2. http://www.springerlink.com/openurl.asp?genre=article&id
11. Duer S (2011) Applications of an artificial intelligence for servicing of a technical object. Neural Comput Appl. doi:10.1007/s00521-011-0788-y
12. Duer S (2012) Artificial neural network in the control process of object's states basis for organization of a servicing system of a technical objects. Neural Comput Appl 21(1):153–160
13. Gacek Z (1999) Technika wysokich napiec. Izolacja wysokonapieciowa w elektroenergetyce. Przepiecia i ochrona przed przepieciami. Skrypt uczelniany nr 2137 wyd.III. Wydawnictwo Politechniki Slaskiej, Gliwice (in polish)
14. Zajkowski K (2004) Algorytm do rozwiazywania ukladow rownan z danymi obarczonymi bledami pomiarowymi. In: Conference on computer applications in electrical engineering, T.II, IEP Politechniki Poznanskiej, pp 499–502(in polish)
15. Zajkowski K (2009) Analysis of overvoltages on inductive system with varistor and capacitor. Poznan University of Technology, Academic Journals Issue 59 2009, Electrical Engineering, Publication by Poznan University of Technology, pp 87–97
16. Zajkowski K (2011) (monograph) Analiza stanu nieustalonego w obwodach rezystancyjno-indukcyjnych w aspekcie minimalizacji przepiec komutacyjnych. Wydawnictwo Uczelniane Politechniki Koszalinskiej, Koszalin (in polish)