CONTROL OF PERMANENT–MAGNET SYNCHONOUS MOTORS USING NEURAL NETWORKS

Ing. Jiří KLÍMA, Doctoral Degree Programme (3)
Dept. of Power Electrical and Electronic Engineering, FEEC, BUT
E–mail: jiri.klima@compas.cz

Supervised by: Prof. Jiří Skalický

ABSTRACT

The neural networks may be used for various purposes, for example – object identification, system modelling and so on. This abstract deals with the problem of using the neural networks to control electrical drives. The subject of the article is the Neural network as the permanent- magnet synchronous motor speed controller, while the motor is vector controlled. The abstract states the results of computer simulations compared to normal PID controller behaviour.

1 INTRODUCTION

Nowadays, neural networks are being used in various fields of human activities. The drives and the Power electronics are no exception. Neural networks are there to be used to simulate drives, to observe the drive and so on. Neural networks, as controllers, may also be used to replace standard PID controllers.

This essay deals with the use of the neural network as a controller. The neural network is used as a controller of the permanent-magnet synchronous motor. The neural controller substitutes for the P-type speed controller with subordinated PI-type current controller of $\mathbf{i_q}$-component, while the $\mathbf{i_d}$ current component of the permanent-magnet synchronous motor is regulated onto its zero value with the help of PI-type current controller. This makes it possible to achieve high dynamics of the drive – so called Vector type control.

1.1 BRIEF DESCRIPTION OF NEURAL NETWORKS

Fundamental element of every neural network is, so called, neuron. Picture No. 1 shows its basic diagram.

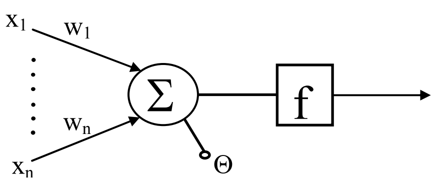

Fig. 1: *Neuron Basic Diagram*

where $x_1, ..., x_n$ - inputs
$w_1, ..., w_n$ – synaptic weights
f – usually non-linear neuron transfer function
Θ - neuron threshold value
y – output

Neuron consists of a summing block adding all input values weighed (multiplied) by weighing factor of the corresponding input. A threshold value is added to this sum in a summing block thus forming a value from which the neuron starts to respond – it outputs some value at the summing block output. The summing block is followed by the transfer function block. With regard to the concerned field, the tangential sigmoid is the best suitable transfer function.

The neural network is made of number of neurons, which are spaced in layers and interconnected.

For the neural network to give off required response, it is necessary to "teach" the network how to provide the required response. There are many training algorithms available. The choice of the algorithm again depends on the field of use and on the demands laid on the neural network. The "Back-propagation" algorithm appears to be the most convenient training algorithm. At the beginning of the training process, the weighing and threshold values are chosen randomly. On training, after completion of each training step, the network output values are compared to the required values and the weighing and the threshold values are adapted according to the Back-propagation training algorithm so long until the output values of the network equal the required ones, i.e. the difference between the both values is sufficiently small. Picture 2 illustratively shows the training procedure principle.

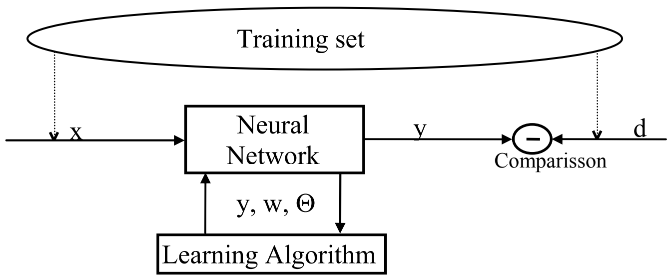

Fig. 2: *Back-propagation Training Procedure Basic Diagram*

2 SOLUTION

The solution to the problem is based on mathematical representation of the permanent-magnet synchronous motor. This can be described with the following set of differential equations:

$$u_d = r_S \cdot i_d + \frac{d\psi_d}{d\tau} - \omega \cdot \psi_q \tag{2-1}$$

$$u_q = r_S \cdot i_q + \frac{d\psi_q}{d\tau} + \omega \cdot \psi_d \tag{2-2}$$

$$\psi_d = x_d . i_d + \psi_f \tag{2-3}$$

$$\psi_q = x_q \cdot i_q \tag{2-4}$$

$$m = \psi_d \cdot i_q - \psi_q . i_d \tag{2-5}$$

$$m = \tau_m \cdot \frac{d\omega}{d\tau} + m_p \tag{2-6}$$

Meaning of individual symbols:

r_S – stator winding resistance

i_d – d-axis current component

i_q – q-axis current component

ω - motor angle speed

u_d – d-axis voltage component

u_q – q-axis voltage component

ψ_d – d-axis magnetic flux

ψ_q – q-axis magnetic flux

ψ_f – excitation permanent magnets magnetic flux (ψ_f=const)

x_d – d-axis winding reactance

x_q – q-axis winding reactance

m – machine torque

τ_m – mechanic time constant

m_p – load torque

There are quite a few principles of teaching the neural controller. This solution makes use of the **Model Reference Control** principle. This principle is on the picture 3. The principle is based on known **M** reference model of a closed control loop. This reference model is defined by input-output pairs of values $\{r(t), y^r(t)\}$. Furthermore, the **P** controlled system is known. The neural controller **C** is designed so that the y^p output values of the controlled system were equal to the $y^r(t)$ values of the reference model with required accuracy. The picture also shows the Reference model to be included in a cascade of the Neural controller. This connection is to enhance the training procedure, which was not used though.

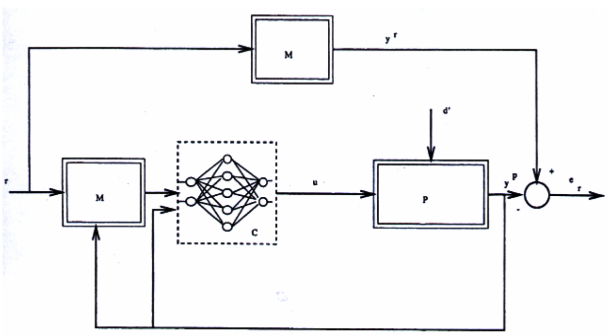

Fig. 3: *Structure of the Control Using Reference Model*

For the purpose of practical simulations the MatLab ver.6.1 environment including its associated Simulink and Neurotoobox tool were used.

Simulink was used to build the Reference Model – see Picture 4.

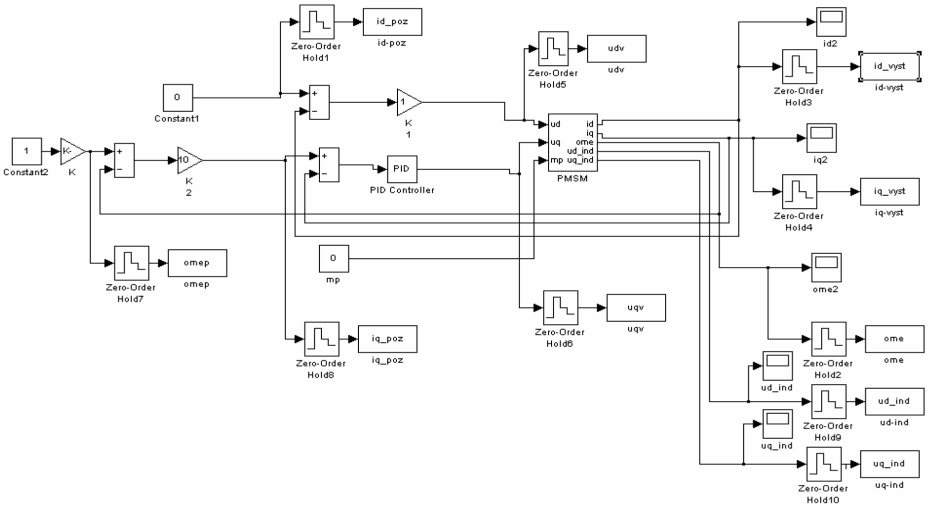

Fig. 4: *Reference model*

With the help of the above Reference model, the input-output pairs of required values were created to be used by the MatLab program. Additionally this program was used to train the neuron controller.

Having finished the teaching process within the Simulink tool, we can create the whole control loop of the neuron controller. Current $\mathbf{i_d}$ is regulated by the P-type controller.

3 RESULTS

The following pictures depict the results and differences of the controls of the permanent-magnet synchronous motor using neuron controller and the standard PID controller.

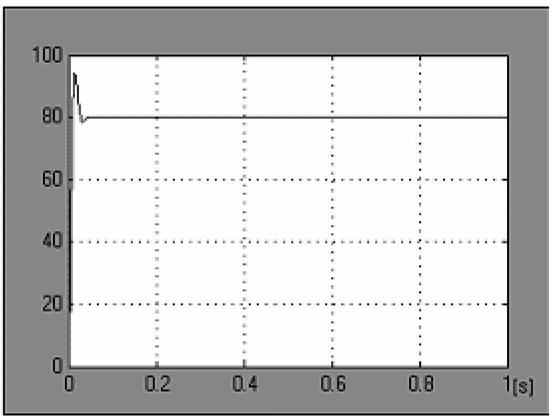

Fig. 5: *Speed – ref. Model*

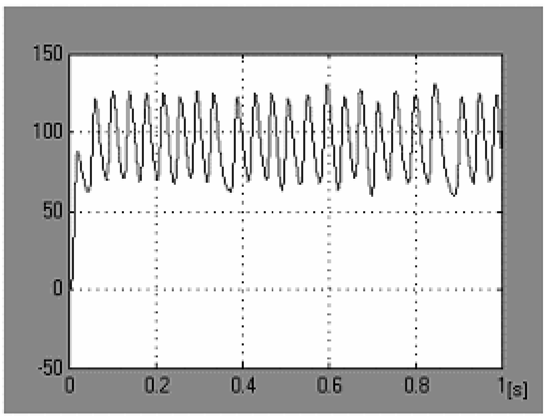

Fig. 6: *Speed – regulated by NN*

The difference between the PID controller and the neuron controller is apparent. There is a significant oscillating response of the permanent-magnet synchronous motor when controlled by the neuron controller. The neuron controller acts in discrete steps . These discrete steps in connection to the time constants of the permanent-magnet synchronous motor can give rise to such oscillations. MatLab environment itself, Simulink and numerical methods may be another reason for the oscillations. It is just a simulation, which can hardly cover all influences. It would, therefore, be necessary to perform the experiment with a real drive.

During the course of the simulations, there were other problems, that are not apparent from the solution nor from the results. One of the problems, there is, most of all, the design of the neural network – i.e. the number of neurons, transfer function, e.t.c This is, however, a general problem of all neural networks. Of course, there are certain rules, but the best design of the network is a question of the experiment.

Another problem is the training of the neural controller, which is time-consuming. This is a general problem of all neural networks again, because the result depends on the power of the computer, on the type of the training procedure etc. The training is also dependent on the input-output pairs obtained from the reference model. The biggest difficulty around this is the cross like feedback in the model of the permanent-magnet synchronous motor. If this feedback was open (which is impossible with the real motor), the teaching of the neural controller was easier and the response was better. Similar easier teaching was achieved on condition that the subordinated PI current controller of $\mathbf{i_q}$ current component in the reference model was replaced with a P-type controller.

The experiment has shown that the neural networks might be used as neural controllers, which are capable of substituting for standard PID controllers. This solution, however, has got several drawbacks, that need to be taken care of. It is also necessary to perform the experiment with a real drive, because the simulations can not account for all influences affecting the control loop.

REFERENCES

[1] Šnorek, M., Jiřina, M.: Neuronové sítě a neuropočítače. FEL ČVUT Praha 1996

[2] Šubrt, J.: Úvod do teorie elektrických střídavých pohonů. VUT FEI ÚPVE Brno 1996

[3] Novák, M.: Systémy napodobující myšlení (umělé neuronové sítě). Slaboproudý obzor- Příloha pro mladé inženýry, svazek 52, čísla 2 až 12; svazek 53, čísla 3-4 Praha 1991

[4] Pillay, P., Krishnan, R.: Modeling, simulation and analysis of permanent-magnet motor drives, Part I: The permanent-magnet synchronous motor drive. IEEE Trans. Ind. Appl., vol. 25, no.2, Mar./Apr. 1989

[5] The MATH WORKS Inc.: Neural Networks. User's Guide

[6] The MATH WORKS Inc.: Real-Time Workshop. User's Guide

[7] Skalický, J.: Neuronové sítě v reg. strukturách mechatronických soustav. EPVE 1997

[8] Klíma, J.: Diplomová práce. Brno 1999

[9] Gupta, M. M., Rao, H. D.: Neuro-Control Systems – theory and applications. A Selected Reprint Volume, IEEE Neural Networks Council, IEEE Press 1993, ISBN 0-7803-1041-1