EVALUATION OF NEURAL NETWORKS AND FUZZY LOGIC TECHNIQUES APPLIED TO THE CONTROL OF ELECTRICAL MACHINES

Jaime Fonseca, Lic. El. Eng., MSc.
Department of Industrial Electronics, Minho University, Braga, Portugal.
e-mail: jaime.fonseca@dei.uminho.pt

João L. Afonso, Lic. El. Eng., MSc.
e-mail: lafonso@dei.uminho.pt

Dr. Júlio S. Martins, Lic. El. Eng., Phd.
e-mail: jmartins@dei.uminho.pt

Prof. Carlos Alberto Couto, Lic. El. Eng., MSc, Phd.
e-mail: ccouto@dei.uminho.pt

ABSTRACT

This paper reports the work that is being developed in the evaluation of neural networks and fuzzy logic techniques applied to the control of electrical machines. An overview of a CACSD (Computer Aided Control System Design) tool – Matlab/Simulink is also presented. This tool was used to design and simulate a control system for an induction motor based on the referred techniques.

1 INTRODUCTION

Artificial Neural Networks (ANN) and Fuzzy Logic are no longer terms just used by specialists. Many people are aware of these new techniques, which are now being used in many everyday applications.

An ANN is a network of many very simple processors ("units"), each possibly having a small amount of local memory. The units are connected by unidirectional communication channels ("connections"), which carry numeric data. Most neural networks have some sort of "training" rule whereby the weights of connections are adjusted on the basis of presented patterns. In other words, neural networks "learn" from examples and exhibit some structural capability for generalization.

Fuzzy logic has emerged as a profitable tool for the controlling of complex industrial processes, as well as for household and entertainment electronics, diagnosis systems and other expert systems. Fuzzy logic is a superset of conventional (boolean) logic that has been extended to handle the concept of partial truth - truth values between "completely true" and "completely false". It was introduced by Dr. Lofti Zadeh of UC/Berkeley in 1960's as a mean to model the uncertainty of natural language. Fuzzy logic techniques attempt to simulate human thought processes, even in technical environments. In doing so, the fuzzy-logic approach allows the designer to handle efficiently very complex closed-loop control problems, reducing engineering time and costs.

Until now, in industrial automation, ANN and fuzzy logic were mainly used for relatively slow processes. ANN can be used in fast processes with multiprocessor architectures (which so far are not very much used in industry).

This paper reports work that is being done, trying to apply these techniques to faster events, namely, the control of electrical machines. Attention will be focused on the use of software development tools that support neural and fuzzy design, rather than in the details of induction motor control, that will be used as an example.

2. DEVELOPMENT TOOLS

The development tools play an important role when designing neuro and fuzzy systems, specially fuzzy controllers. For a number of reasons, a designer cannot connect the controller being developed directly to the controlled system until late in the development process. In early design stages, he must instead use simulation to design and tune the fuzzy controller.

Matlab is an integrated technical computing environment that combines numeric computation, advanced graphics and visualization, and a high-level programming language. It is a natural environment for analysis, algorithm prototyping, and application development. Matlab also features a family of application-specific solutions called *toolboxes* [8, 9]. Toolboxes are libraries of Matlab functions that customize Matlab to solve particular classes of problems, like fuzzy logic and neural network control. Simulink is built on top of the Matlab, and is an interactive environment for modelling, analyzing and simulating a wide variety of dynamic systems.

3 INDUCTION MOTOR SLIP CONTROL

Figure 1 presents a conventional slip control scheme for an inverter fed induction motor [10] which is used for low performance variable speed drives. Traditionally, the speed error (ω_e) is input to a PI controller that sets the motor slip frequency (ω_r). Stator frequency (ω_s) is obtained by adding slip frequency to rotor speed (ω), and stator voltage (U_s) is set accordingly to a pre-defined $U_s/\omega_s \approx constant$ law, so that motor flux is kept at its nominal value. Slip frequency must be limited, setting (indirectly) a limit to both peak torque and stator current.

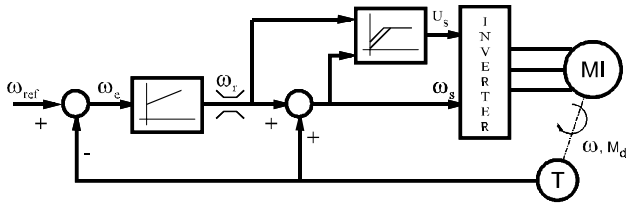


Figure 1. Slip control for an induction motor voltage-source inverter drive.

Keeping flux at nominal value allows the motor to develop its rated torque but does not guarantee that it operates with good efficiency. In fact it can be shown that, specially for small motors and light loads, motor efficiency can be improved by reducing motor flux [10, 11]. Therefore, the conventional slip control scheme can be improved, so that at steady state, stator voltage (and motor flux) is reduced to minimize motor losses. As stator frequency and rotor slip frequency are automatically adjusted by the speed control loop, the motor operating point is maintained (output power is constant). At the same time, an average motor input power can be computed so that voltage is decreased until minimum input power is found, in a efficiency optimization process.

The ideas presented so far have been simulated in the Matlab environment using both neural and fuzzy techniques, and compared with the results produced by conventional approaches.

3.1 The Fuzzy Logic Approach

Figure 2 presents the system block diagram in the Simulink environment. Note that together with a graphic interface, this tool provides an extensive block library, several integration algorithms, and allows the user to easily select other simulation parameters (Figure 3). It is also possible to generate ANSI *C* code from Simulink using the toolbox "Real Time Workshop".

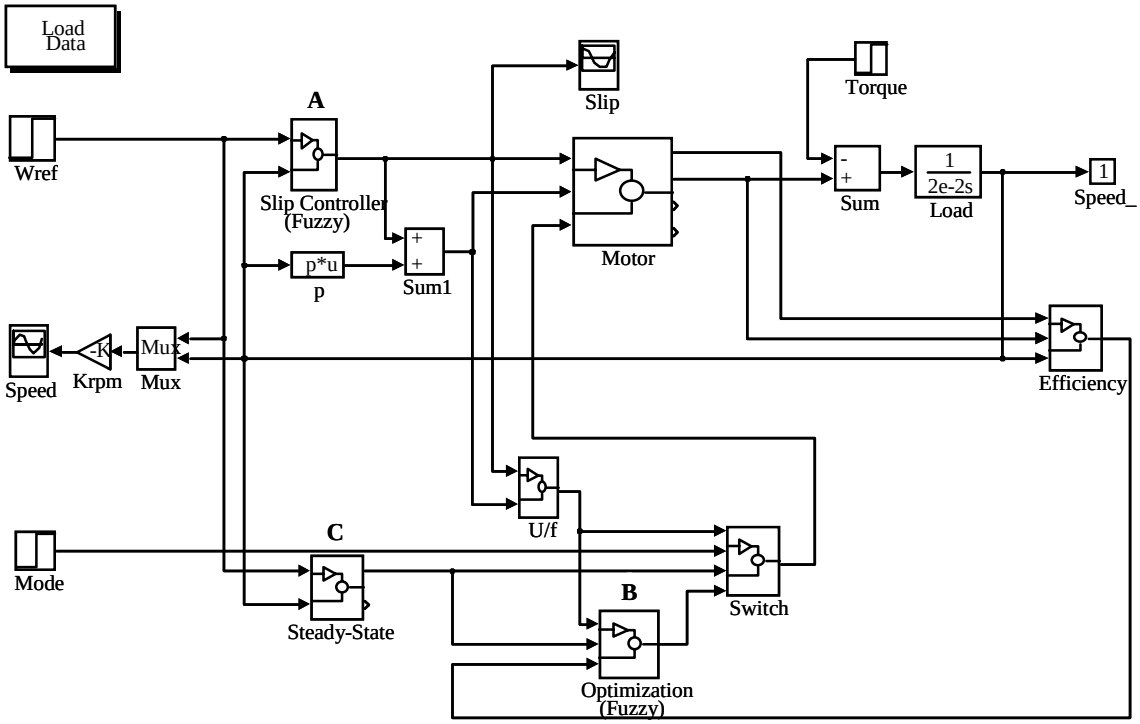


Figure 2. The fuzzy approach - system block diagram.

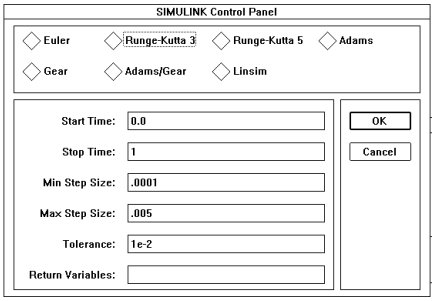


Figure 3. Simulation parameters dialog box.

Block *A* (Figure 2) is a masked block that implements the fuzzy slip controller. Block *B* is another masked fuzzy block, responsible for the motor efficiency optimization process. Block *C* decides if the motor runs in steady-state or in the transient regime, allowing switching from a "constant flux" to "flux reduction" law.

The "Fuzzy Logic Toolbox version 1.0" used to develop blocks *A* and *B* covers all steps of fuzzy logic design, from the project definition to the definition of membership functions and rules.

The first step is to define the structure of the controller by means of the "Fuzzy Inference System (FIS) Editor". The FIS Editor displays the control structure and permits to adjust the fuzzy inference functions, such as the defuzzification method (see Figure 4). The FIS Editor also allows the designer to directly access the "Membership Function Editor", the "Rule Editor", the "Rule Viewer" and the "Surface Viewer".

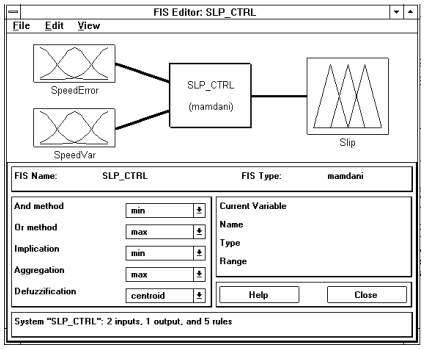


Figure 4. Structure of the induction motor fuzzy logic controller.

The second step of the controller design is the definition of the membership functions. The graphic interface of the development tool allows the designer to easily create the most suitable membership functions and the linguistic variable for the application. The slip controller (Figure 2), for instance, has two inputs (the speed error and the motor acceleration) and one output (the motor slip increment). Each of the inputs is described with three triangular membership functions: "Negative" (N), "Zero" (ZE) and "Positive" (P) (Figure 5). The output is described with five identical membership functions: "Negative Medium" (NM), "Negative Small" (NS), "Zero" (ZE), "Positive Small" (PS) and "Positive Medium" (PM).

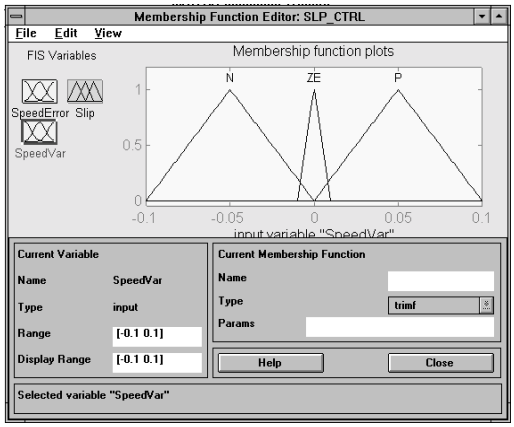


Figure 5. Membership functions for motor acceleration.

The third step is the definition of the rules with "Rule Editor" (Figure 6). It allows three different writing formats. The numbers in the parentheses represent weights that can be applied to each rule if desired (by default they are assumed to be one).

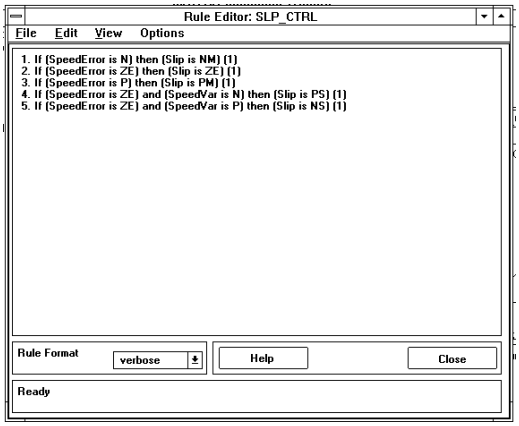


Figure 6. Rules used in the slip controller.

The forth step of the controller design is the test and optimization of rules and membership functions. The "Fuzzy Logic Toolbox" provides two tools to perform these tasks: the "Rule Viewer" and the "Surface Viewer" (Figures 7 and 8).

Block *B* (for efficiency optimization) was implemented in a similar way. It has one input (efficiency variation) and one output (motor input voltage increment) described with five triangular membership functions: "Negative Medium" (NM), "Negative Small" (NS), "Zero" (ZE), "Positive Small" (PS) and "Positive Medium" (PM). The controller uses five rules.

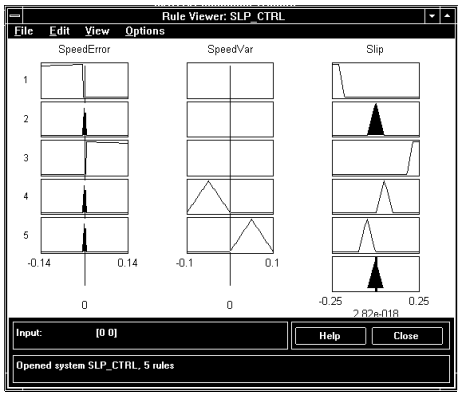


Figure 7. Rule Viewer.

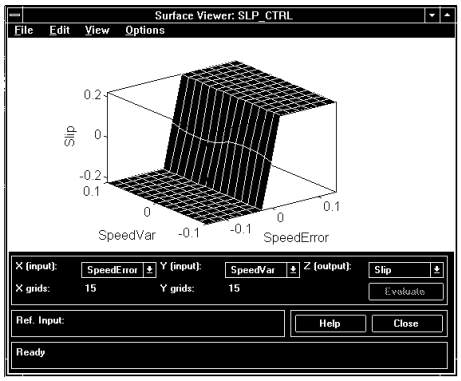


Figure 8. Surface Viewer.

In the development of both controllers each rule output was determined by "MIN-MAX" inference and the crisp output was generated by "centroid" (COG) defuzzification.

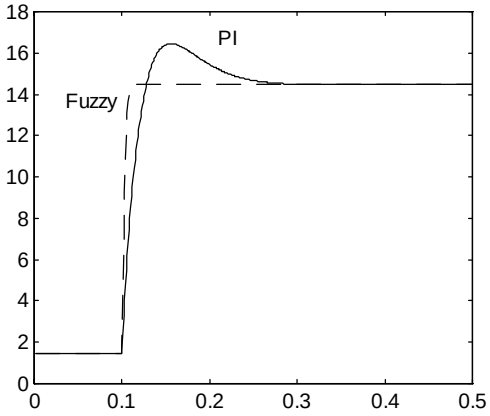
3.2 The Neural Network Approach

Attempts were made to implement the slip controller using an ANN feedforward neural net with two inputs (speed error and acceleration) and one output (slip speed increase). The network architecture had two layers, the first layer with five tan-sigmoid neurons and the second one with only one linear neuron. Soon was realized that it is difficult to find proper training vectors for this case and to obtain convergence with an acceptable error and a number of neurons compatible with real time computation.

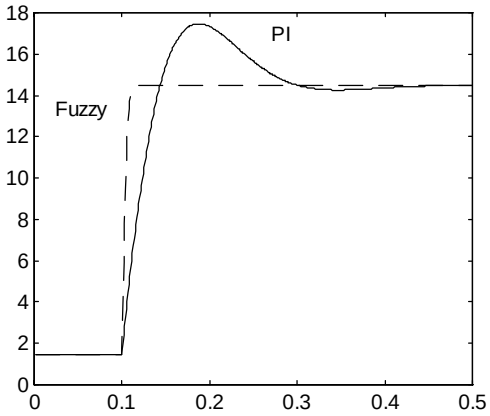
4 SIMULATION RESULTS

Figures 9 to 12 present simulation results for a small (1 hp) induction motor. The slip controller was implemented using both fuzzy techniques and a PI with antiwindup mechanism [12]. Note that it is not possible to tune a conventional PI controller because of slip limitation. Load is assumed to be pure inertia (unless otherwise stated, $J = 2.10^{-2}$ kg.m^2). The integration method used was the "Range-Kutta 3" and the sampling period for the slip controller was 20 ms. The sampling time for the efficiency optimization process was 0.1 s.

Figure 9 shows the slip controller behavior in response to a sudden load change from 10% to 100% of the nominal torque, for two different load inertia values.



(a)

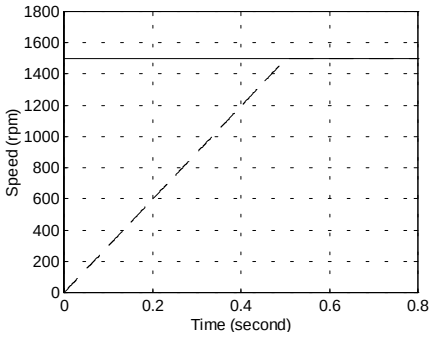


(b)

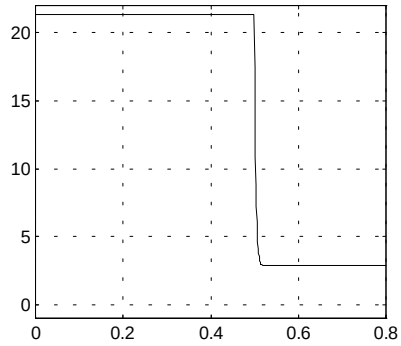
Figure 9. Torque step response:
(a) J = 2.10^{-2} kg.m^2; (b) J = 4.10^{-2} kg.m^2

As can be concluded from the graphics, while the PI dynamic response is highly dependent on load inertia (oscillations increase with this parameter), the fuzzy controller response remains almost unchanged.

Figures 10 and 11 present motor speed and slip during start-up when using, respectively the fuzzy and PI controllers. In this case the response of both controllers is identical.

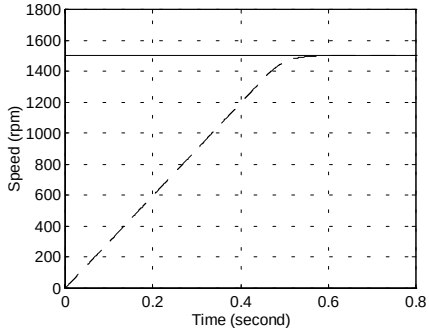


(a)

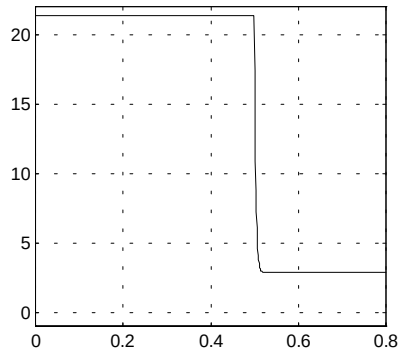


(b)

Figure 10. Motor start-up: (a) Speed; (b) Slip.



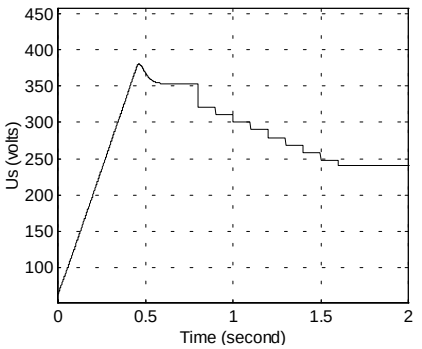
(a)



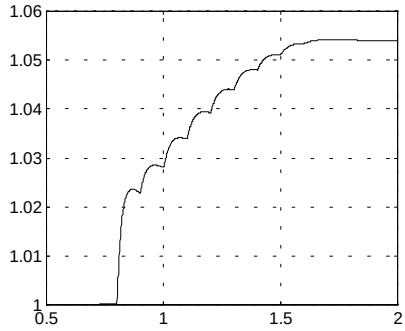
(b)

Figure 11. Motor startup: (a) Speed; (b) Slip.

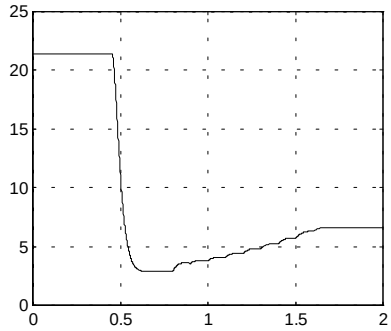
Figure 12 illustrates the efficiency optimization process, which starts, approximately 0.8 seconds after start-up. Torque is 20% of nominal value. Note that as voltage is decreased by properly adjusted steps by the fuzzy controller, efficiency is increased till a maximum is reached. An improvement of about 5.5% was achieved in this case.



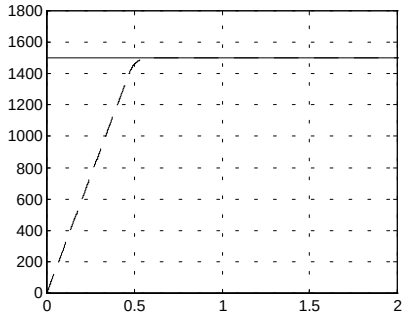
(a)



(b)



(c)



(d)

Figure 12. Efficiency optimization process: (a) Voltage; (b) Efficiency improvement; (c) Slip; (d) Speed.

5 CONCLUSIONS

In this paper an evaluation of neuro and fuzzy techniques applied to the control of an induction motor was presented. Matlab/Simulink was used for controller design and for simulation of the whole system.

Simulation results confirmed the superior dynamic performance of fuzzy logic control against more conventional schemes (PI controller), namely in terms of insensitivity to changes in model parameters.

The sampling period used with the slip fuzzy controller - 20 ms - will probably be compatible with real time control using standard hardware (a 16/32 bits microcontroller), which is the next goal. As an alternative there is already available specific hardware for the implementation of fuzzy controllers.

ANN techniques tested so far seem unsuitable for this case: results obtained are poor compared to both fuzzy and conventional techniques.

REFERENCES

[1] Robert E. Uhrig, *Introduction to Artificial Neural Networks*, IEEE IECON'95, Orlando, USA, 1995.

[2] Simon Haykin, Neural Networks - *A Comprehensive Foundation*, Macmillan Publishing Company, 1995.

[3] Jerry M. Mendel, *Fuzzy Logic Systems for Engineering: A Tutorial*, Proceedings of the IEEE, Vol. 83, No. 3, pp 345-377, 1995.

[4] David Brubaker, *Design and Simulate your Own Fuzzy Setpoint Controller*, EDN, pp 167-170, January 5, 1995.

[5] David Brubaker, *Simulation in Fuzzy-Controller Design*, EDN, pp 163-166, February 16, 1995.

[6] David Brubaker, *Fuzzy Setpoint Controller-Act 3*, EDN, pp 133-136, March 16, 1995.

[7] H. Chris Tseng and Victor H. Hwang, *Servocontroller Tunning with Fuzzy Logic*, IEEE Transaction on Control Systems Technology, Vol. 1, No. 4, pp 262-269, 1993.

[8] Howard Demuth and Mark Beale, *Neural Network Toolbox For Use with MATLAB*, The Math Works, Inc., 1993.

[9] J.-S. Roger Jang and Ned Gulley, *Fuzzy Logic Toolbox For Use with MATLAB*, The Math Works, Inc., 1995.

[10] J. S. Martins, *Controlo de Velocidade do Motor de Indução Trifásico*, PhD Thesis, Minho University (Portugal), 1993.

[11] C. Couto and J. S. Martins, *Control of a Voltage Source Inverter Fed Induction Motor with On-Line Efficiency Optimization*, IEEE ICIT'94, Guangzhou, China, 1994.

[12] Karl J. Astrom, *Computer Controlled Systems*, Prentice-Hall, 1990.