Int. J. Engng Ed. Vol. 21, No. 1, pp. 49–54, 2005
Printed in Great Britain.

0949-149X/91 $3.00+0.00

Modeling and Simulation of the DC Motor Using Matlab and LabVIEW*

NICOLAE PATRASCOIU
Automatic and Industrial Information Department, University of Petrosani, Romania. E-mail: patrascoiu@upet.ro

One of the most used actuators in control systems is a direct current (DC) motor. The general output variable of this actuator can be angular speed or angular displacement motion, but, coupled with wheels or drums and cables, can provide translation motion. This paper proposes a state–space model of the DC motor built for constant flux and considering two inputs: supply voltage and resistive torque. The three states of the resulting model are represented by angular speed, angular displacement and current supply and any of these states can be an output variable for a simulation model. Consequently, the system's model has two inputs and three outputs. For the system's simulation a VI is built where the most important element is a Matlab script which contains the matrices A, B, C, D of the state-space model, the independent variable time and the Matlab simulation function lsim. The motor's parameters are given by digital controls on the panel so that these parameters can be interactively modified. To generate inputs, two CASE structures are used where the input variables can be set: impulse, step and ramp. Here it is also possible to set the signal amplitude and duration, either by knob or slide control. TRANSPOSE 2D ARRAY and INDEX ARRAY are used for setting the matrices' dimensions. The output signals are live display, either one by one or together, on the WAVEFORM GRAPH.

INTRODUCTION

STUDENTS CAN USE the model and the VI in the classroom to simulate the running of a DC motor and also to learn how to incorporate the latter into a control loop. To start with, based on a functionality structure of the DC motor and on the laws of physics and electricity that rule the variable magnetic flux density (separate excitation) motor's operation, students build a mathematical model. Afterwards, using the mathematical model in a series of simulation experiments in LabView, the students can observe the motor dynamics. For this, it is necessary to build the simulation VI and students must analyze the diagram bloc of the VI to ensure adequate functionality of the component subdiagrams.

For the input signal, it is possible to select various forms of supply voltage and/or resistive torque so that the above-mentioned outputs can be observed. The forms of the output can suggest the design of the control loop or the type of controller used. Also, for every input signal, students can observe the effect of varying the parameters of the DC motor over the outputs.

Functionality equations of the DC motor

We will consider a direct current (DC) electric motor with separate excitation, compensating winding and commutating pole. The structure of this machine is presented in Fig. 1, where:

v_S, i_S—supply voltage and current;
u_E, i_E—excitation voltage and current;
R, L—winding electric resistance and inductance;
$\varphi(t)$—excitation flux;
$e(t)$—back electromotive force;
$\omega(t)$—angular speed;
$m(t)$—electromagnetic torque;
$r_T(t)$—restoring torque.

If we apply the Kirchhoff voltage theorem to the supply circuit, we get:

$$v_S(t) - e(t) = R \cdot i_a(t) + L \cdot \frac{di_a(t)}{dt} \qquad (1)$$

The torque equilibrium equation on the axis of the motor is:

$$m(t) = m_T(t) + f_T(t) + r_T(t) \qquad (2)$$

where $m_T(t)$ is the motoring torque, which is dependent on the moment of inertia of the rotor, and $f_T(t)$ is the motor friction torque.

Now the torque equilibrium equation can be written:

$$m(t) = J \cdot \frac{d\omega(t)}{dt} + F \cdot \omega(t) + r_T(t) \qquad (3)$$

It is known that an electromagnetic torque is dependent on the excitation flux in excitation winding and on the supply current by the armature constant k_t which in SI units (which we will use here) is equal to the motor constant k_e ($k_t = k_e = k$):

* Accepted 14 July 2004.

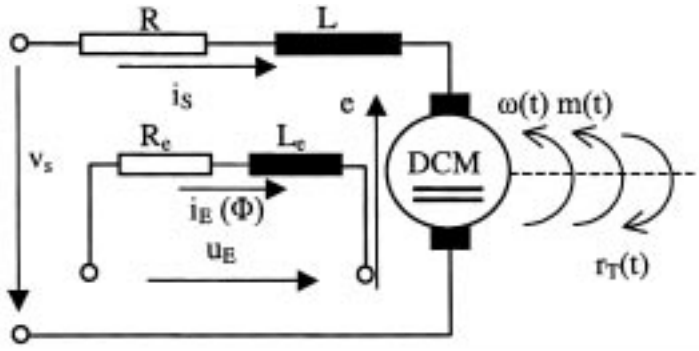

Fig. 1. DC motor functionality structure.

$$m(t) = k \cdot \varphi(t) \cdot i_S(t) \tag{4}$$

and the back electromotive force is dependent on the angular speed and on the excitation flux in excitation winding by motor constant k:

$$e(t) = k \cdot \varphi(t) \cdot \omega(t) \tag{5}$$

(1), (3), (4) and (5) are functionality analytical equations (i.e. a general mathematical model of the DC electrical motor).

If we consider the angular displacement $\alpha(t)$ instead of angular speed $\omega(t)$, a similar output variable is necessary to take account of the relationship between these:

$$\omega(t) = \frac{d\alpha(t)}{dt} \tag{6}$$

The DC motor angular speed control is achieved by two methods: constant flux and variable flux. In this paper we consider speed control by constant flux.

Constant flux simulation model

If the excitation flux is constant, insert the notation:

$$k \cdot \Phi = K_m \tag{7}$$

into the general mathematical model of the DC electrical motor. Now it can build the mathematical model in state–space form so that it is possible to use it in a Matlab simulation.

Substituting equations (4) and (5) and notation (7) in equations (1) and (3) results in a complete DC motor model by constant flux, respectively:

$$\begin{aligned} \frac{di_\alpha(t)}{dt} &= \frac{1}{L} \cdot v_S(t) - \frac{R}{L} \cdot i_S(t) - \frac{K_m}{L} \cdot \omega(t) \\ \frac{d\omega(t)}{dt} &= -\frac{F}{J} \cdot \omega(t) + \frac{K_m}{J} \cdot i_\alpha(t) - \frac{1}{J} \cdot m_L(t) \end{aligned} \tag{8}$$

To build the state–space model, bring into this mathematical model the input, state and output vectors:

- State vector $x(t)$, whose components are represented by supply current $i_S(t)$, angular displacement $\alpha(t)$ and angular speed $\omega(t)$:

$$x(t) = \begin{vmatrix} i_S(t) \\ \alpha(t) \\ \omega(t) \end{vmatrix} \tag{9}$$

- Input vector $u(t)$, whose components are repre-

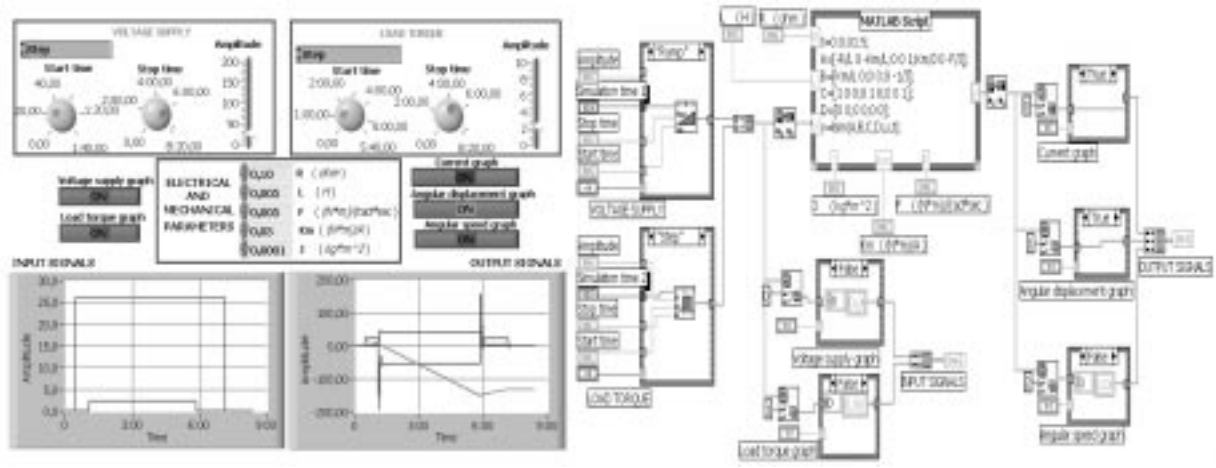

Fig. 2. Front panel and bloc diagram of the simmot.vi.

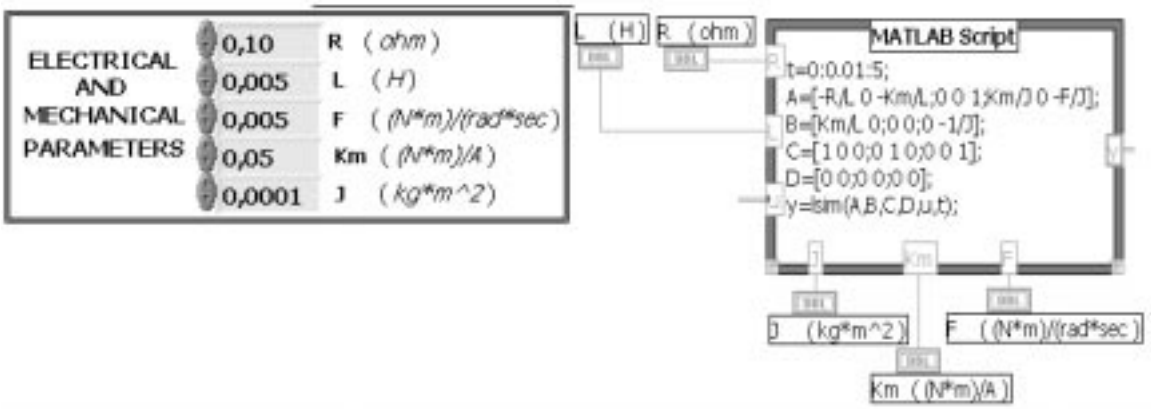

Fig. 3. Matlab script node and parameter controls.

sented by supply voltage $v_S(t)$ and load torque $m_L(t)$:

$$u(t) = \begin{vmatrix} v_S(t) \\ m_L(t) \end{vmatrix} \tag{10}$$

- Output vector $y(t)$, whose components we treat as the same as state vector components, so that it is possible to simulate these three physical quantities.

Using these vectors it is possible to write equations (8) in matrix form:

$$\begin{vmatrix} \dot{i}_S(t) \\ \dot{\alpha}(t) \\ \dot{\omega}(t) \end{vmatrix} = \begin{vmatrix} -\frac{R}{L} & 0 & -\frac{K_m}{L} \\ 0 & 0 & 1 \\ \frac{K_m}{J} & 0 & -\frac{F}{J} \end{vmatrix} \cdot \begin{vmatrix} i_S(t) \\ \alpha(t) \\ \omega(t) \end{vmatrix} + \begin{vmatrix} \frac{1}{L} & 0 \\ 0 & 0 \\ 0 & -\frac{1}{J} \end{vmatrix} \cdot \begin{vmatrix} v_S(t) \\ m_L(t) \end{vmatrix} \tag{11}$$

Using systems general equations, this form can be written in compact form:

$$\dot{x}(t) = A \cdot x(t) + B \cdot u(t) \tag{12}$$

where A and B are a costants matrix:

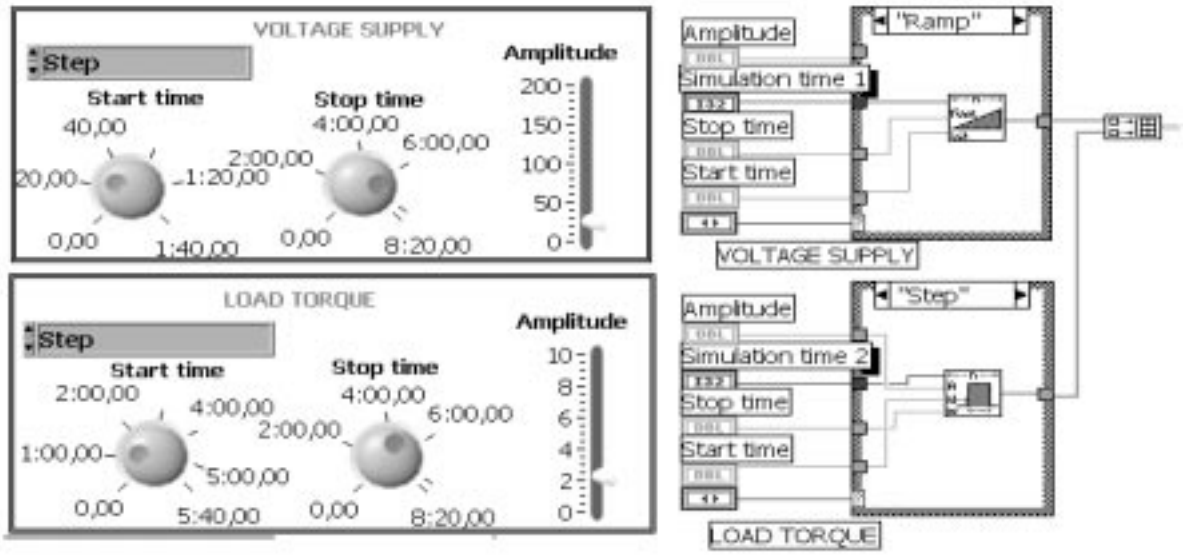

Fig. 4. Generation of input signals.

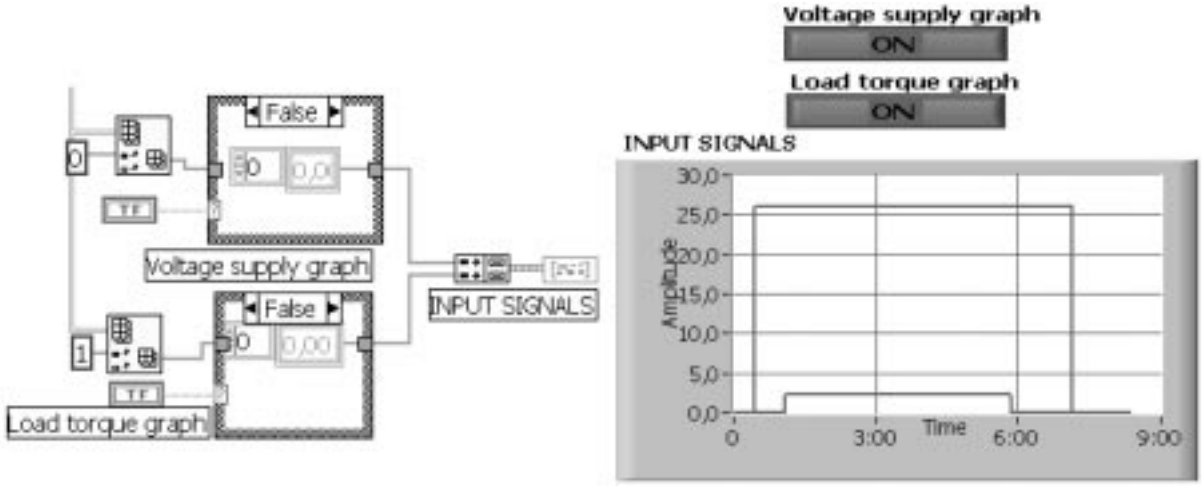

Fig. 5. Data representation of input signals.

$$A = \begin{vmatrix} -\frac{R}{L} & 0 & -\frac{K_m}{L} \\ 0 & 0 & 1 \\ \frac{K_m}{J} & 0 & -\frac{F}{J} \end{vmatrix}; \qquad B = \begin{vmatrix} \frac{1}{L} & 0 \\ 0 & 0 \\ 0 & -\frac{1}{J} \end{vmatrix} \tag{13}$$

If we add the output vector definition, the input–output equation can be written as the matrix:

$$y(t) = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} \cdot \begin{vmatrix} i_S(t) \\ \alpha(t) \\ \omega(t) \end{vmatrix} + \begin{vmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{vmatrix} \cdot \begin{vmatrix} v_S(t) \\ m_L(t) \end{vmatrix} \tag{14}$$

or, in compact form:

$$y(t) = C \cdot x(t) + D \cdot u(t) \tag{15}$$

where:

$$C = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}; \qquad D = \begin{vmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{vmatrix} \tag{16}$$

Now we have the matrix A, B, C, D and we can use the function *lsim*, which has the form:

$$y = lsim(A, B, C, D, v, t) \tag{17}$$

where vector t specifies the time samples for the simulation and consists of regularly spaced time samples t to simulate the DC motor as an LTI system.

Build the VI to simulate a DC motor in LabView

The control panel and bloc diagram of the VI that was used to simulate the DC motor are presented in Fig. 2.

The base element of the LabView program simmot.vi, which is used to simulate a working DC motor, is the Matlab script node. Through this, the matrix A, B, C, D of the motor state–space model and the simulation function are inserted into the LabView program (Fig. 3).

The motor construction parameters R, L, K_m, F and J also represent the model parameters and these are set using the controls on the front panel, which should be named with the same letters as the

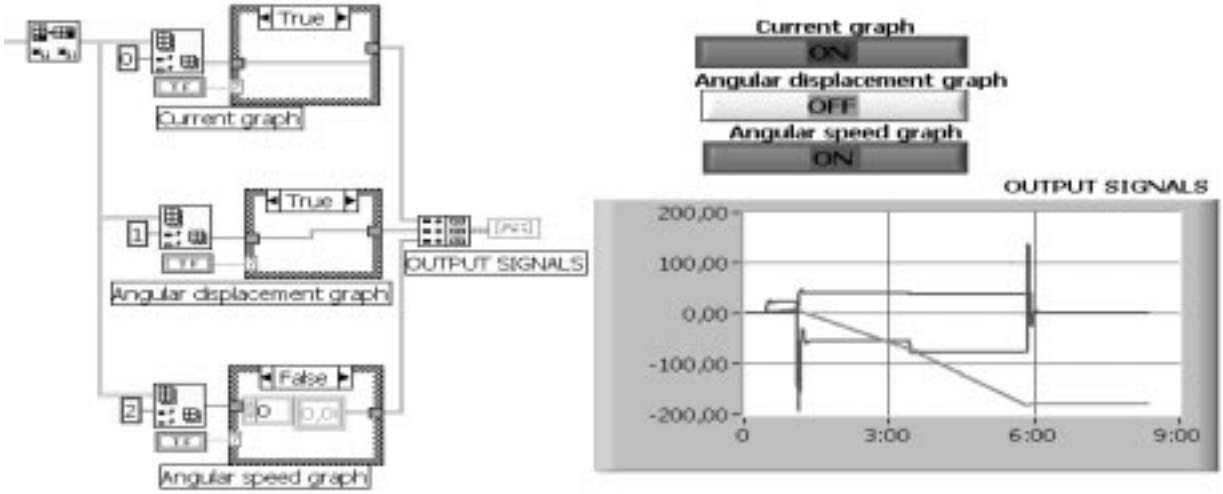

Fig. 6. Data representation of output signals.

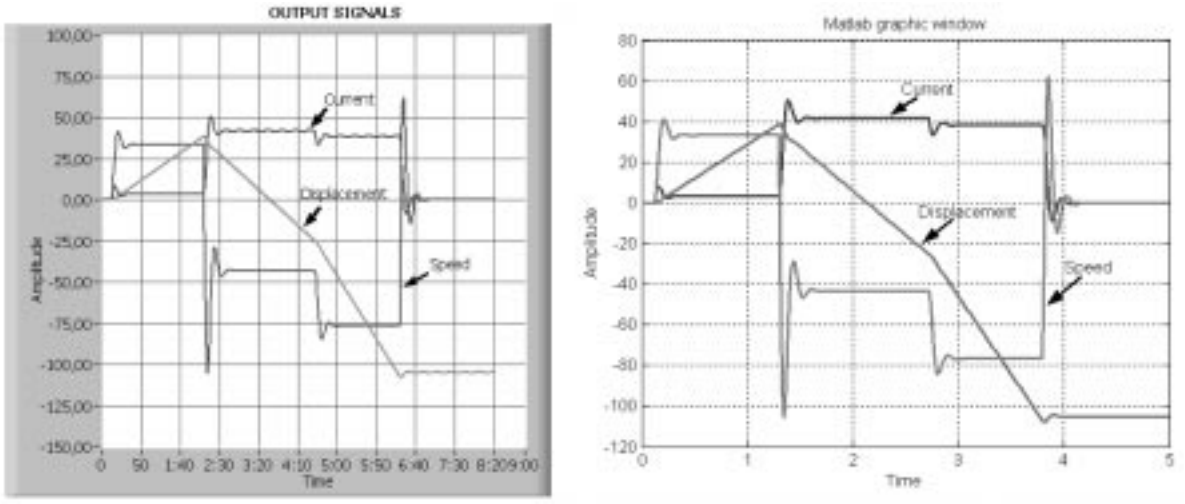

Fig. 7. Data representation of output signals in LabView and Matlab.

parameters. For the respective controls, the steps of values corresponding to the real values of these parameters are chosen. The values set by these controls represent the inputs for the script node with real data type.

Because the input vector has two components for generating the right signals, two blocs are built on the front panel. These two blocs are represented in diagram bloc by a CASE structure with three subdiagrams, which generates the standard signals (i.e. IMPULSE, STEP and RAMP). The subdiagrams contain, respectively, Impulse Pattern.vi, Pulse Pattern.vi and Ramp Pattern.vi, and these can be adjusted by the corresponding controls. Also, the control panel can be used to set parameters such as amplitude, width (given by START TIME end STOP TIME) and delay (given by START TIME) using either a knob or a slide control (Fig. 4).

With the two arrays that contain the input pattern, the BUIL ARRAY node can be used to obtain a new array that has values arranged in rows that represent for a real system the input signals. For a graphic representation of these, it can use a graphic display WAVEFORM GRAPH type. The input vector requires a transpose array so that a vector can be obtained that is arranged in two columns and in order that it can use the TRANSPOSE 2D ARRAY node to rearrange the elements of the 2D array such that the 2D array [i, j] becomes a transposed array [j, i]. Similarly, the same node is used at the output of the Matlab script node because the output vector of this node must be in rows and must have three columns that correspond to the three output signals. The input and the output terminals of the Matlab script node corresponding to the input and output signals must be in Real Matrix type.

The input and output signals have different forms and different value ranges (Figs 5 and 6), so that a selection of these signals is necessary using the AutoScale option of the WAVEFORM GRAPH display. To do this, the INDEX ARRAY nodes are used for the input and the output. The right signal is selected to set the index input of these nodes using the corresponding button on the front panel, so as to display the same signal by its connection to the Matlab scrip input.

The right sequence of the simulation program is necessary in order that the number of samples of the vector t, which represents the simulation time in Matlab, is the same as the number of samples of the signal generation pattern. To do this in Matlab script, the vector t that specifies the time samples for the simulation should have $5/0.01 + 1 = 501$ components. It is also necessary for the number of samples of the Ramp Pattern to be 501. For Stop Time control, Relative time (seconds) in the Format & Precision option should be selected.

CONCLUSIONS

The Matlab lsim function simulates the (time) response of continuous or discrete linear systems to arbitrary inputs and lsim (sys, u, t) produces a plot of the time response of the LTI model sys to the input time history. If this function is incorporated into a Matlab simulation program, such as adding Matlab script to a LabView program called Virtual Instrument (VI), the input u can be produced by a different signal generation function in VI.

LabView starts Matlab and, if using the plot (t,y) instruction after the lsim instruction in Matlab script, a new Matlab window appears labeled Figure No. 1, which displays the graphical responses of the DC motor outputs. In Fig. 7, the similarity between the LabView and Matlab results can be observed.

By means of controls placed on the front panel of the VI, the model parameters and the input signal parameters can be set. For example, where the load torque has a significant value, as shown in

Fig. 6, negative values appear for displacement and speed outputs. This means that the motor axle starts spinning the other way round.

Also, because LabView is a data acquisition program, it is possible to make comparisons between data acquired from a real system and data obtained from a mathematical model of the system.

REFERENCES

1. A. Fransua, Masini si actionari electrice, Tehnica, Bucuresti (1986).
2. M. Ghinea, Matlab: Calcul numeric-grafica, Teora, Bucureşti (1997).
3. D. Matko and R. Karba, Simulation and Modelling of Continuous System, Prentice-Hall, New York (1992).
4. N. Patrascoiu, Modelarea şi simularea sistemelor, Focus, Petroşani (2001).
5. LabView User Manual, National Instruments.
6. MATLAB® Release 11. Documentation set (http://www.mathworks.com/support/).

Nicolae Patrascoiu is an Associate Professor in Engineering. He teaches data acquisition and control systems at the Automatic and Industrial Informatics Department at the University of Petrosani, Romania. He shows a particular interest in the links between data acquisition and modeling and simulation systems.