BULETINUL Universităţii Petrol – Gaze din Ploieşti	Vol. LVIII No. 2 bis / 2006	181-186	Seria Tehnică

Computational Models of the Asynchronous Machine by Using the Software of Matlab. First Order Model

Gabriela Petropol-Şerb, Aurel Câmpeanu

University of Craiova
e-mail: gpetropol@em.ucv.ro

Abstract

The goal of paper is to present the models of the asynchronous machine and their calculation by using the software of Matlab. Paper is a connection between the theory of the asynchronous machine and the data-processing programming. It was illustrate the effect of the approximations made to obtain the mathematical models and their influence in the tracing of the characteristics of operation.

Key words: *asynchronous machine, first order model, software of Matlab.*

Introduction

Induction machines are the most widely used of all electric motors. They are simple to build and rugged, offer reasonable asynchronous performance: a manageable torque-speed curve, stable operation under load, and generally satisfactory efficiency. The difficulty with using this machine in variable speed drives is that they are hard to control, since their torque speed relationships is complex and nonlinear. In this paper we develop models to control the asynchronous motor.

To start the analysis of this machine, assume that both the rotor and the stator can be described by balanced, three - phase windings. The two sets are coupled by mutual inductances, which are dependent on rotor position. To simulate the behavior of an asynchronous machine, we must adopt a physical model bringing into discussion the principal electro-mechanical characteristics. The equivalent schemas used in the study of the asynchronous machine are very important to establish the performances of the machine. Their complexity causes the approximations obtained.

The first order model of the asynchronous machine

The first order model is the 'steady state' model, which assumes that the electrical states reach steady state much more quickly than the mechanical state.

It is known from literature [1] that the generalized equivalent schema in transient regime is, without restrictions, like in figure 1.

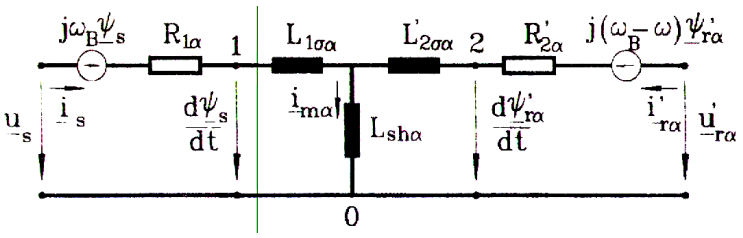

Fig.1. Generalized equivalent schema of an induction machine.

The parameters are those from the literature [1] and are given by:

$$R_{1\alpha} = R_1, L_{1\sigma\alpha} = L_{1\sigma} + (1-\alpha)L_{sh},$$
$$R'_{2\alpha} = \alpha^2 R'_2, L'_{2\sigma\alpha} = \alpha^2 L'_{2\sigma} + (\alpha^2 - \alpha)L_{sh},$$
$$L_{sh\alpha} = \alpha L_{sh}, L_{s\alpha} = L_{1\sigma\alpha} + L_{sh\alpha} = L_s, L'_{r\alpha} = L'_{2\sigma\alpha} + L_{sh\alpha} = \alpha L'_r$$

Considering different α, results that there are possible an infinity of equivalent schemas with representative phasors, which allowed the analyses of transient process.

In a referential which is solidar with the stator ($\omega_B = 0$) and for a short-circuiting rotor ($u'_{r\alpha} = 0$), is obtaining the equivalent schema from figure 2.

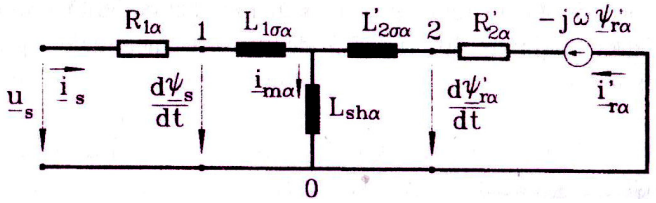

Fig. 2. Generalized equivalent schema at $\omega_B = 0$ and $u'_{r\alpha} = 0$.

The equations system for this schema is:

$$\begin{aligned}
&\underline{u}_s = R_{1\alpha}\underline{i}_s + sL_{1\sigma\alpha}\underline{i}_s + sL_{sh\alpha}\underline{i}_{m\alpha} \\
&0 = R'_{2\alpha}\underline{i}'_{r\alpha} + sL'_{2\sigma\alpha}\underline{i}'_{r\alpha} + sL_{sh\alpha}\underline{i}_{m\alpha} - j\omega\underline{\psi}'_{r\alpha} \\
&\underline{i}_{m\alpha} = \underline{i}_s + \underline{i}'_{r\alpha} \\
&\underline{u}_{e1\alpha} = -sL_{sh\alpha}\underline{i}_{m\alpha}
\end{aligned} \tag{1}$$

In sinusoidal, symmetric regime:

$$\underline{u}_s = \underline{U}_1 e^{j\omega_1 t}, \underline{i}_s = \underline{I}_1 e^{j\omega_1 t}, \underline{i}_{m\alpha} = \underline{I}_{01} e^{j\omega_1 t}$$

and the general equations (1) became:

$$\begin{aligned}
&\underline{U}_1 = R_{1\alpha}\underline{I}_1 + j\omega_1 L_{1\sigma\alpha}\underline{I}_1 + j\omega_1 L_{sh\alpha}\underline{I}_{01\alpha} \\
&0 = R'_{2\alpha}\underline{I}'_{2\alpha} + j\omega_1 L'_{2\sigma\alpha}\underline{I}'_{2\alpha} + j\omega_1 L_{sh\alpha}\underline{I}_{01} - j\omega(L'_{r\alpha}\underline{I}'_{r\alpha} + L_{sh\alpha}\underline{I}_1)
\end{aligned} \tag{2}$$

Evidencing the slip, system (2) became:

$$\begin{aligned}
&\underline{U}_1 = \underline{Z}_{1\alpha}\underline{I}_1 - \underline{U}_{e1\alpha}\\
&\underline{U}_{e1\alpha} = -\underline{Z}_{1m\alpha}\underline{I}_{01\alpha}\\
&\underline{I}_{01\alpha} = \underline{I}_1 + \underline{I}'_{2\alpha}\\
&\underline{U}_{e1\alpha} = -\underline{Z}'_{2\alpha}\underline{I}'_{2\alpha}
\end{aligned} \tag{3}$$

where:

$$\begin{aligned}
&\underline{Z}_{1\alpha} = R_{1\alpha} + jX_{1\alpha} = R_{1\alpha} + j\omega_1 L_{1\sigma\alpha},\\
&\underline{Z}_{1m\alpha} = jX_{1m\alpha} = j\omega_1 L_{sh\alpha},\\
&\underline{Z}'_{2\alpha} = \frac{R'_{2\alpha}}{s} + jX'_{2\alpha} = \frac{R'_{2\alpha}}{s} + j\omega_1 L'_{2\sigma\alpha}
\end{aligned} \tag{4}$$

Equations (3) are descried by equivalent schema from figure 3.

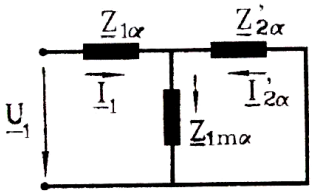

Fig. 3. Generalized equivalent schema for the stationery regime.

From the schema 3, after immediate calculus it is obtained:

$$\underline{Z}_{e1\alpha} = \underline{Z}_{e1} = \underline{Z}_1 + \frac{\underline{Z}'_2 \cdot \underline{Z}_{1m}}{\underline{Z}_{1m} + \underline{Z}'_2} \tag{5}$$

Terminal current is:

$$\underline{I}_1 = \frac{\underline{U}_1}{\underline{Z}_{e1}} \tag{6}$$

The expression of torque, used for the simulation in Matlab, is:

$$M = \frac{P_M}{\Omega_1} \tag{7}$$

The equation of motion is:

$$\frac{d\omega}{dt} = \frac{1}{J}(M - M_m) \tag{8}$$

Numerical method used to solve the mathematical model.

Numerical method used to solve the mathematical model is ode23, which is an implementation of an explicit Runge-Kutta (2,3).It is a one-step solver which solves initial value problems for

ordinary differential equations (ODEs). Its syntax is:

[T,Y] = solver(odefun,tspan,y0)

where:
odefun- is a function that evaluates the right-hand side of the differential equations. The ode23s solver can solve only equations with constant mass matrices;
tspan – is a vector specifying the interval of integration,
[t0,tf]. y0 – is a vector of initial conditions.

The default integration properties in the ODE solvers are selected to handle common problems. We can improve ODE solver performance by supplying the solvers with one or more property values in an options structure:

[t,y] = solver(odefun,tspan,y0,options)

The odeset function creates an options structure that we can pass as an argument to any of the ODE solvers. To create an options structure, odeset accepts property name/ property value pairs using the syntax:

options = odeset ('name1',value1,'name2',value2,...)

In the resulting structure, options, the named properties have the specified values. Any unspecified properties contain default values in the solvers. For all properties, it is sufficient to type only the leading characters that uniquely identify the property name.

At each step, the solver estimates the local error e in the 'i^{th}' component of the solution. This error must be less than or equal to the acceptable error, which is a function of the specified relative tolerance, RelTol, and the specified absolute tolerance, AbsTol. |e(i)| max(RelTol*abs(y(i)),AbsTol(i)).

For the absolute error tolerance, the scaling of the solution components is important: if |y| is somewhat smaller than AbsTol, the solver is not constrained to obtain any correct digits in y. It might have to solve a problem more than once to discover the scale of solution components. This means that we want RelTol correct digits in all solution components except those smaller than thresholds AbsTol(i). Even if we are not interested in a component y(i) when it is small, you may have to specify AbsTol(i) small enough to get some correct digits in y(i) so that we can accurately compute more interesting components.

Simulations and results

The theoretical aspects shown hereinbefore are applied for an asynchronous motor with the following rated values:

P_n = 4 kW; U_n = 220/380 V; m = 3; f = 50 Hz; p = 2; J = 0,024 kg m^2; R_s = 1,16 Ω; R'_r = 3,515 Ω; $L_{s\sigma}$ = 0,024 H; $L'_{r\sigma}$ = 0,0034 H; L_{sh} = 0,2986 H.

Using the software of Matlab we solve the differential equation (12) and plot the speed n(t), the torque M(t), and the current $I_1(t)$ for the asynchronous motors :

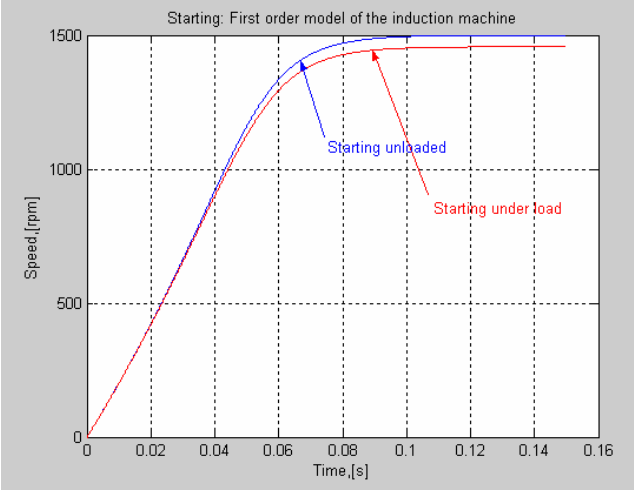

Fig. 4. Speed curve: n(t).

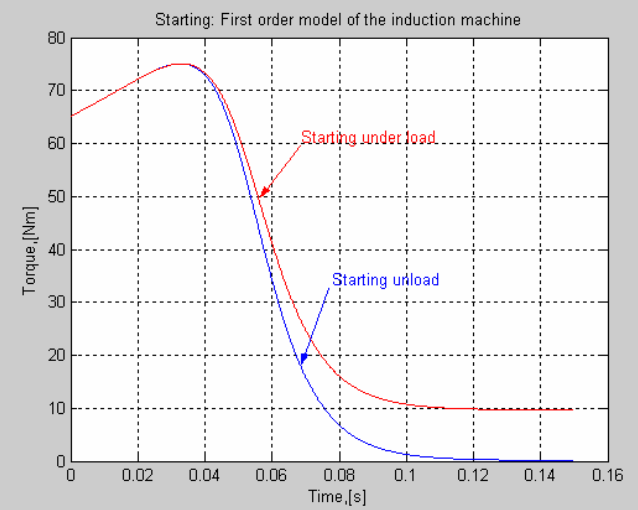

Fig. 5. Torque curve M(t).

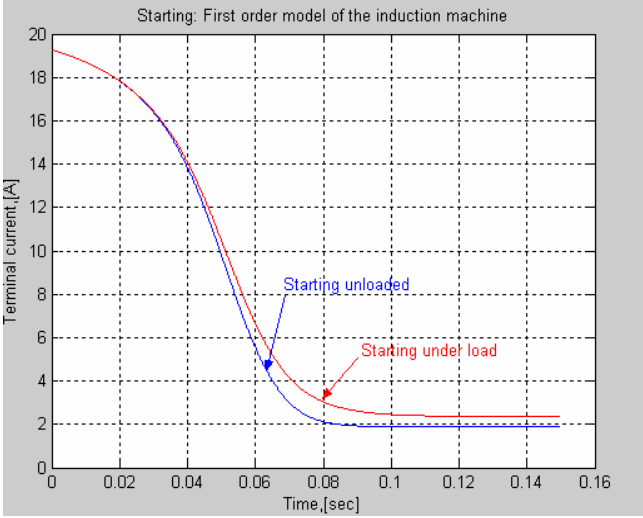

Fig. 6. Current curve $I_1(t)$.

The simulations were made for a load torque proportional to speed squared. Using this model we can change the different parameters of the machine to see its behavior. The errors introduced by this model are much greatest than the errors introduced by the other models of the machine.

Conclusions

The computational modeling of the induction machine offers much information about the behavior of this and let us to anticipate their operation in different situations.

References

1. Câmpeanu, A. - *Introducere în dinamica maşinilor electrice de curent alternativ*, Editura Academiei Române, Bucureşti,1998
2. Câmpeanu, A - *Masini electrice.Probleme fundamentale, speciale si de functionare optimala*. Editura Scrisul Romanesc, 1988.
3. Ghinea, M., Fireţeanu, V. - *MATLAB - calcul numeric, grafică, aplicaţii*., Teora, 1999
4. Using MATLAB, by The MathWorks, Inc.

Modelarea numerică a maşinii asincrone folosind programarea în Matlab. Modelul de ordinul întâi

Rezumat

Scopul acestei lucrări este de a prezenta modele ale maşinii asincrone şi calculul lor numeric folosind programarea în Matlab. Lucrarea este o conexiune între teoria maşinii asincrone şi programare. Este ilustrat efectul aproximărilor făcute pentru a obţine modelele matematice şi influenţa lor asupra trasării caracteristicilor de funcţionare.