

**KRZYSZTOF KLUSZCZYŃSKI DARIUSZ SPAŁEK**

**STEP-BY-STEP ANALYSIS  
OF INDUCTION MACHINES  
ALLOWING FOR SLOTTING**

UNDER THE AUSPICES OF ELECTRICAL ENGINEERING COMMITTEE  
OF POLISH ACADEMY OF SCIENCES

POLISH SOCIETY FOR THEORETICAL AND APPLIED ELECTRICAL ENGINEERING

UNDER THE AUSPICES OF  
ELECTRICAL ENGINEERING COMMITTEE OF  
POLISH ACADEMY OF SCIENCES

**STEP-BY-STEP ANALYSIS  
OF INDUCTION MACHINES  
ALLOWING FOR SLOTTING**

Krzysztof KLUSZCZYŃSKI

Dariusz SPAŁEK

POLISH SOCIETY FOR  
THEORETICAL AND APPLIED ELECTRICAL ENGINEERING



Warsaw, 2003

---

The reviewers:

Professor Andrzej Demenko, Poznań University of Technology, Poland

Professor Yoshiyuki Ishihara, Doshisha University, Kyoto, Japan

Professor Jan K. Sykulski, University of Southampton, United Kingdom

The editor PTETiS

The text and figures set by Dariusz Spałek

The cover designed by Dr Arch. Joanna Serdyńska

ISBN 83 – 915991 – 4 – 0

#### *Acknowledgement*

Special thanks are due to reviewers for their counsel and enriching remarks as well as to the members of Electrical Engineering Committee of Polish Academy of Science

Professor Kazimierz Zakrzewski (Chairman of Electrical Engineering Committee and Chairman of Electrical Machines and Transformers Section)

Professor Zbigniew Ciok (Past-Chairman of Electrical Engineering Committee)

Professor Andrzej Horodecki  
for their editorial support.

*The book has been published in three stages preceding the following conferences (at which the monograph was presented):*

*XVII Seminar on Electrical Engineering BSE, Istebna, December 2002,*

*XXXIX International Symposium on Electrical Machines SME,  
Gdańsk-Jurata, July 2003,*

*IV International Workshop on Research and Education in Mechatronics,  
Germany, Bochum, October 2003.*

*At the third stage the authors took into account the remarks and suggestions delivered by the readers and the participants of the above mentioned conferences (the edition completed).*

## CONTENTS

	pp.
<u>List of symbols</u>	<u>5</u>
<u>1. Introduction</u>	<u>7</u>
<u>2. Physical model of induction machine with discrete distribution of air-gap permeance</u>	<u>13</u>
2.1. Main assumptions.....	13
2.2. Air-gap permeance distribution and related equivalent magnetic circuits... <td style="text-align: right;">15</td>	15
2.3. Inductances.....	28
<u>3. Transient-state mathematical model of induction machine</u>	<u>37</u>
3.1. Concept of alternate step-by-step analysis.....	37
Procedure A.....	37
Procedure B.....	39
3.2. Computer simulation based on recurrent sequence relations.....	39
<u>4. Steady-state analysis of induction machine</u>	<u>48</u>
<u>5. Ability of mathematical model for describing parasitic reluctance torque resulting from slotting</u>	<u>53</u>
<u>Conclusions</u>	<u>62</u>
<u>References</u>	<u>63</u>

## APPENDIX

Computer simulation program for educational purposes.....	67
<u>Educational program listing – main unit</u>	<u>75</u>

---

## LIST OF SYMBOLS

$a_k$  - magnetic energy change related to Dirac impulse of electromagnetic torque  
 $T_e = a_k \delta(\vartheta - \vartheta_k),$

$B_s$  - magnetic flux density induced by stator coil,  
 $B_k$  - magnetic flux for  $k^{\text{th}}$  channel

$$\Delta\Phi_k = \frac{1}{2}B_k ld,$$

$c$  - integer number,

$d$  - inner diameter of stator,

$i_r$  - rotor current,

$i_s$  - stator current,

$i, j, k$  - numbers of phase circuits

for stator  $i, j, k = 1, \dots, Q_s$ ,

for rotor  $i, j, k = Q_s + 1, \dots, Q_s + Q_r$ ,

$\mathbf{I}$  - vector of stator and rotor currents,

$J$  - moment of inertia,

$k_r, k_s$  - supplementary coefficients (integer numbers),

$l_e$  - equivalent length of machine,

$m = m_s + m_r$  - total number of stator and rotor phases,

$m_s$  - number of stator phases,

$m_r$  - number of rotor phases,

$\mathbf{M}$  - matrix of inductances including elements corresponding to self, mutual and leakage inductances,

$M_{sr}$  - stator-rotor inductance,

$\mathbf{M}_{sr}$  - submatrix of stator-rotor inductances,

$n_s$  - synchronous speed for fundamental ( $v=1$ ) MMF space harmonic,

$n_{\text{syn}}$  - synchronous speed for reluctance torque ([rpm] or [rps]),

$N_s$  - number of stator turns per phase,

$N_r$  - number of rotor turns per phase,

$p$  - number of pole-pairs,

$\mathbf{R}$  - matrix of stator and rotor resistances ( $m \times m$ ),

$Q$  - number of steps for stator-rotor inductance v. rotor angle curve,

$Q_s$  - number of stator slots (teeth),

$Q_r$  - number of rotor slots (teeth),

$Q_L$  - number of channels at the lower region of coil,

$Q_U$  - number of channels at the upper region of coil,

$\mathbf{U}$  - vector of stator and rotor phase voltages,

$Y_r$  - pitch of rotor coil,

$Y_s$  - pitch of stator coil,

$T_e$  - electromagnetic torque,

$T_{e_{av}}$  - average value of electromagnetic torque,

$T_{e_{asyn}}$  - asynchronous torque,

$T_m$  - mechanical torque,

$\alpha$  - space coordinate along periphery of the air-gap,

$\alpha_k$  - angle position of  $k^{\text{th}}$  magnetic channel ( $k=1, \dots, Q_s+Q_r$ ),

$\delta(\vartheta - \vartheta_k)$  - Dirac impulse at the rotor angle  $\vartheta_k$ ,

$\delta_{ef}$  - effective air-gap width,

$\gamma_s$  - stator coil symmetry axis position angle,

$\Phi$  - magnetic flux passing through the air-gap,

$\Delta\Phi_L$  - magnetic flux passing through a single magnetic channel inn the lower region,

$\Delta\Phi_U$  - magnetic flux passing through a single magnetic channel inn the upper region,

$\Lambda$  - permeance of the air-gap,

$\Lambda_o$  - permeance of single magnetic channel,

$\mu_0$  - free-space permeability,

$v, v_\Theta$  - orders of space harmonic,

$\vartheta$  - rotor angle,

$\Omega_m$  - angular mechanical speed,

$\Omega_{syn}$  - synchronous speed for reluctance torque.

---

## 1. INTRODUCTION

Induction motors have a history of over one hundred years and their significance in electrical-drive systems is still increasing. Over all these past years, starting with steady-state equivalent circuit for the fundamental space harmonic, the theory of induction machines based on Fourier harmonics analysis has been developing.

In general, an induction machine (see Fig.1.1) as an electromechanical system is described by **m voltage equations** (related to electric circuits):

$$\mathbf{U}_{m \times 1} = \mathbf{R}_{m \times m} \cdot \mathbf{I}_{m \times 1} + \frac{d}{dt} \mathbf{M}(\vartheta) \cdot \mathbf{I}_{m \times 1}, \quad (1.1)$$

where **R** is matrix of resistances, **M** denotes matrix of inductances including elements corresponding to self, mutual and leakage inductances,  $\vartheta$  means rotor angle,  $m$  is total number of stator and rotor phases,

and **torque equation** (related to a mechanical system) which - in the simplest case - has the form:

$$J \frac{d\Omega_m}{dt} = T_e + T_m, \quad (1.2a)$$

$$\Omega_m = \frac{d\vartheta}{dt}, \quad (1.2b)$$

$$T_e = \frac{1}{2} \mathbf{I}^T \frac{\partial \mathbf{M}}{\partial \vartheta} \mathbf{I}, \quad (1.2c)$$

where  $T_e$ ,  $T_m$  mean electromagnetic and mechanical torque,  $\Omega_m$  means mechanical rotor speed,  $J$  denotes moment of inertia.

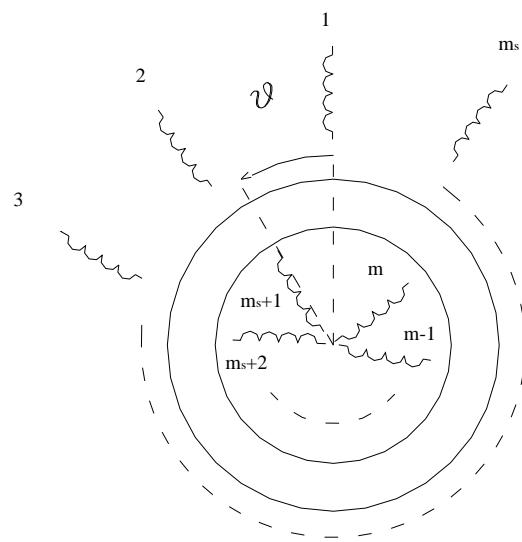


Fig.1.1. Electrical machine in which total number of stator and rotor phases is equal to  $m$ .

From mathematical point of view, even under the assumption of a non-saturated magnetic circuit which means that the magnetic circuit of the machine is linear, equations (1.1) and (1.2) constitute a non-linear system of  $(m+1)$  differential equations with time-varying coefficients ( $m$  is total number of stator and rotor phases). The reason for their non-linearity is the bilinear form in Eqn (1.2a). A block diagram corresponding to Eqns (1.1) and (1.2) is presented in Fig.1.2.

#### THE ELECTROMECHANICAL SYSTEM

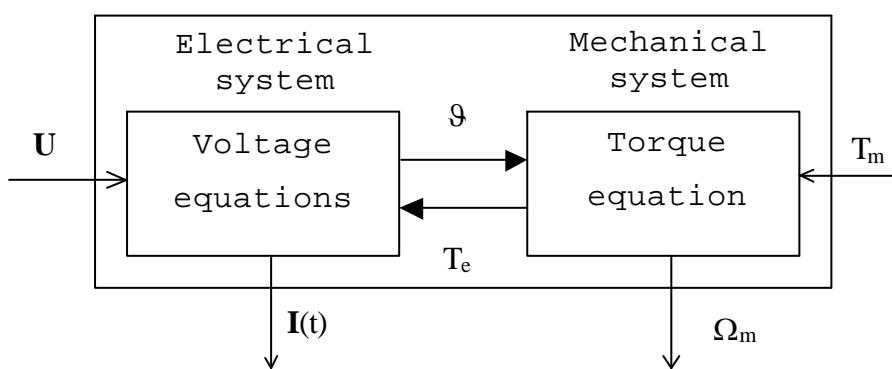
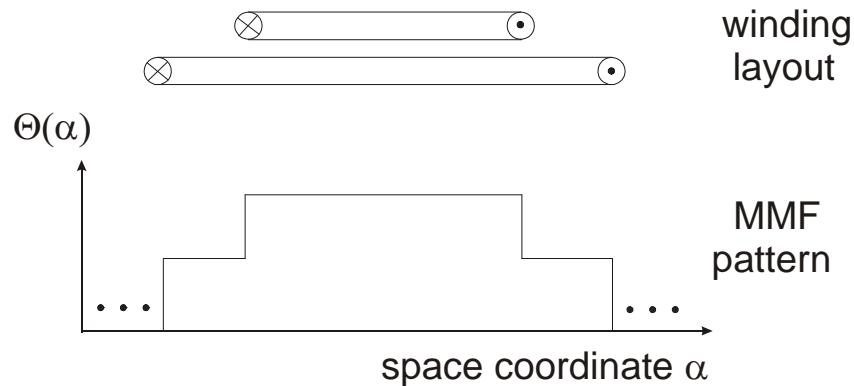


Fig.1.2. Electrical machine as electromechanical system

The bi-directional interaction between the electrical and mechanical systems is characterised by the two opposite arrows.

a)



b)

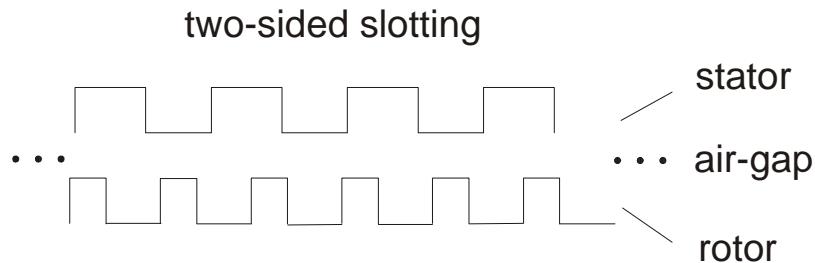
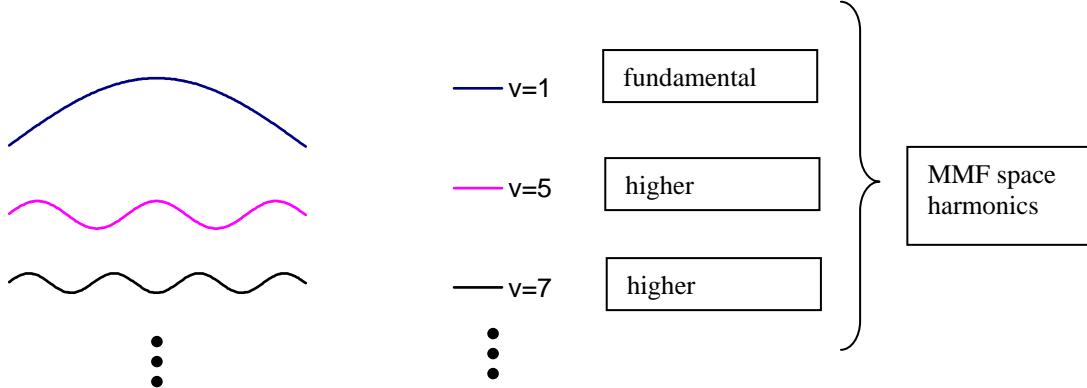


Fig.1.3. Electrical machine – idealized representation of windings and slotting: a) winding layout and MMF  $\Theta(\alpha)$  pattern, b) stator and rotor slotting.

Because of the non-sinusoidal space distribution of MMF  $\Theta(\alpha)$  produced by successive stator and rotor windings (Fig.1.3a) as well as because of the two-sided slotting (Fig.1.3b), magnetic field pattern in the air-gap of a machine and, as a consequence, the self and mutual inductances of an induction machine included in the matrix  $\mathbf{M}$  have a very complicated form. Usually, these non-sinusoidal functions (MMF space pattern and space distribution of air-gap permeance) are resolved in Fourier series, enabling the fundamental space harmonic ( $v=1$ ), higher MMF space harmonics ( $v=5,7,13,15,\dots$ ) and permeance space harmonics of the orders:  $cQ_s$  (stator harmonics),  $cQ_r$  (rotor harmonics),

$c|Q_s \pm Q_r|$  (differential harmonics), to be separated (Fig.1.4), where  $c$  denotes an integer number,  $Q_s$ ,  $Q_r$  are the stator and rotor numbers of slots, respectively.

a)



b)

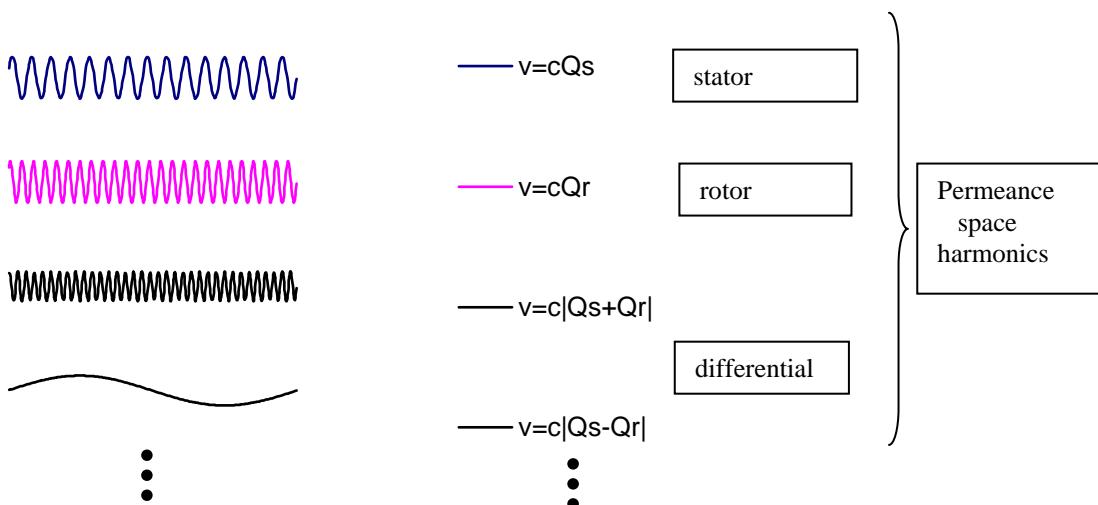


Fig.1.4. Fourier series for magnetic field space distribution: a) MMF space harmonics: fundamental  $v=1$  and higher harmonics  $v=5,7,\dots$  b) permeance harmonics of stator and rotor:  $v=cQ_s$ ,  $v=cQ_r$  and differential harmonics  $v=c|Q_s \pm Q_r|$ .

In many practical cases, it is sufficient to take into account only the fundamental space harmonic ( $v=1$ ) which allows the derivation of a very simple equivalent circuit with time-constant parameters describing properly the basic properties of the machine. Consideration of the so-called parasitic phenomena like parasitic torques (having asynchronous, synchronous or pulsating character), vibrations, magnetic noise etc. requires more complicated models allowing for higher MMF and permeance space harmonics (the so-called

polyharmonic models) which can only be solved numerically with the help of a computer. These polyharmonic models valid for transient and steady states, have been derived with the help of wide variety of different methods:

- revolving-field method [6],[13],[15],[38],[39],[53],
- tensor-based method [12], [35],
- non-transformational matrix analysis [36],[45],[46],
- classical and generalized symmetrical components [1],[4],
- instantaneous value symmetrical components [8],[40],[42],[43],[48],[49],
- $\alpha,\beta,0$  transformation (2-axis coordinates) [3],[5],[7],[11],[16],[17],[20], [21], [54],
- space vector method [50],[51],[54],
- spiral vector method [56],
- special time-dependent transformation [9],[10],[14],[42] etc.

Theoretically, differential equations corresponding to such polyharmonic models include an infinite number of coefficients associated with consecutive magnetic field space harmonics, making numerical solution of the equations very time-consuming and investigation of complex interactions among differential space harmonics very difficult. In practice, number of space harmonics is reduced by choosing the order of the highest space harmonic or by selecting the orders of space harmonics, but it must be borne in mind that this can influence unfavourably the accuracy of the solutions [8], [11], [14], [20], [39], [50].

Summing up, the complexity of a classical polyharmonical model of electrical machine based on harmonic analysis results from:

- the non-linear character of the differential equations,
- the time-varying values of their coefficients,
- the fact that the  $m$  voltage equations (1.1) and torque equation (1.2) constitute one system of  $(m+1)$  differential equations which must be solved as a whole,
- the large number of Fourier coefficients.

Some attempts of using another basic functions, instead of sinusoidal functions, e.g. seraphile functions [41] or Walsh functions [19] also did not bring about a major simplification of the analysis. Therefore, in some latest publications the Fourier harmonic analysis was abandoned and the quite

different non-harmonic approach was proposed. In [36] a new multiple coupled circuit model was derived by means of the so-called winding functions. The coefficients of differential equations are calculated directly from the geometry and winding layout of the machine without using Fourier series. Due to the nature of winding function method the stator-rotor inductance v. rotor angle curves are continuous piecewise-linear functions hence their derivatives are piecewise-constant functions.

The main purpose of this book is to present another new non-harmonic model which has been developed in many publications since 1994 [22] ÷ [33]. The model is derived under the assumption that the stator-rotor inductances as a function of rotor angle can be expressed by discontinuous piecewise-constant functions. As a consequence, the derivative of this curve is the sequence of Dirac impulses. Because of **separation of voltage equations** (1.1) and **torque equation** (1.2) (see the reason for the complexity of a classical model) such model has a remarkably simpler mathematical form than the classical polyharmonic models. Furthermore, the solution of this model - for both transient and steady states - can be expressed analytically in the form of recurrent sequence relations, impossible in the case of the classical model. The next advantage of the model is that the discretisation of the stator-rotor inductance v. rotor angle curve can be brought into relationship with the number of stator and rotor slots. Owing to that, the proposed model can be regarded as a model which takes into account two-sided slotting of the air-gap and enables to consider effects of parasitic reluctance torque resulting from such a kind of magnetic irregularities (Figs 1.3b and 1.4b).

The presented model is considered with reference to an induction machine but can be adopted for other types of AC machines e.g. reduction reluctance motors.

## 2. PHYSICAL MODEL OF INDUCTION MACHINE WITH DISCRETE DISTRIBUTION OF AIR-GAP PERMEANCE

### 2.1. MAIN ASSUMPTIONS

Let us consider a 3-phase induction machine. The most important assumption different from those usually made (e.g. linear magnetization curve, negligible skin effect in a rotor, leakage inductances treated as external ones, squirrel-cage rotor considered as multi-phase circuit with mesh currents) is that the permeance of the air-gap along the periphery of the machine has discrete space distribution. This means that main magnetic flux can pass continuously through the air-gap from stator to rotor only through a finite number of magnetic channels of infinitesimal width (Fig.2.1b), which will be marked henceforth in the cross-section of an electrical machine with black spots (Fig.2.1c).

Magnetic channels (marked in Fig. 2.2c with black spots) are spread regularly along the circumference of the machine and correspond to the axes of  $Q_s$  stator teeth and  $Q_r$  rotor teeth i.e. to the points corresponding to the maximum values of air-gap permeance related to stator and rotor side (Figs 2.1a and 2.2).

In order to take into account leakage fluxes, the additional external leakage inductances will be added to the inductances associated with the main flux passing through the air-gap.

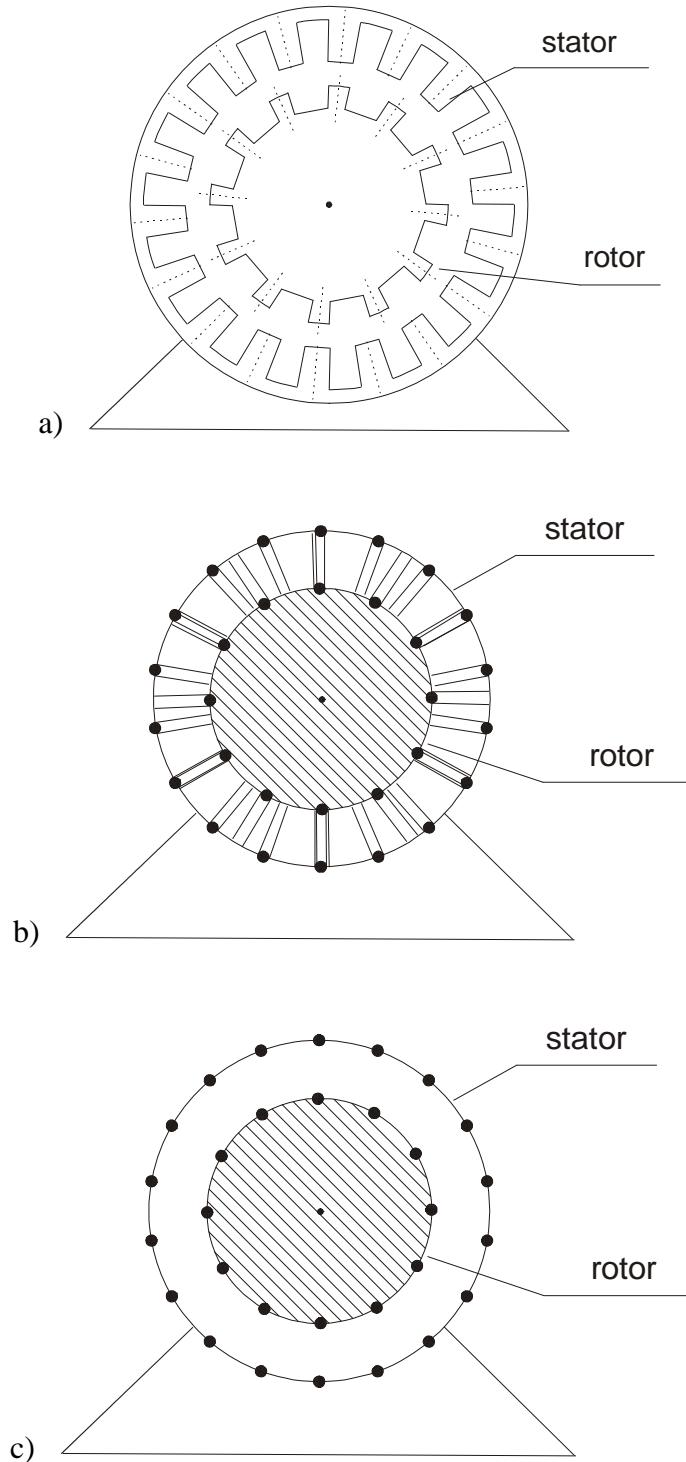


Fig 2.1.(a) Cross-section of electrical machine, (b) magnetic channels distributed along the periphery of air-gap, (c) simplified denotation

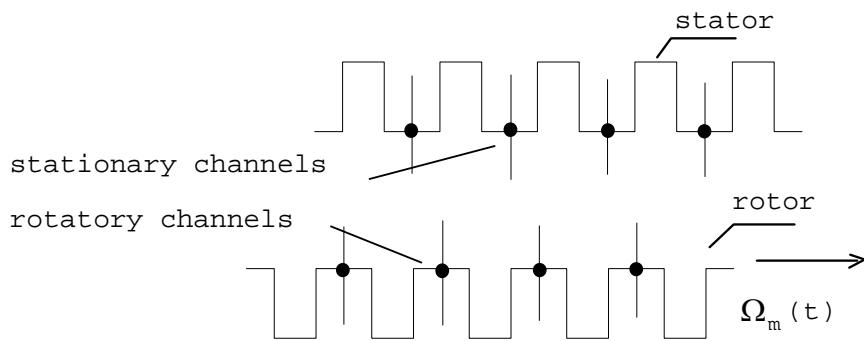


Fig 2.2. Developed circumference of the air-gap - idea of stationary and rotatory magnetic channels

## 2.2. AIR-GAP PERMEANCE DISTRIBUTION AND RELATED EQUIVALENT MAGNETIC CIRCUITS

There are  $Q_s$  magnetic channels fixed in the stator (stationary magnetic channels) and  $Q_r$  magnetic channels fixed in the rotor (rotatory magnetic channels) (Fig.2.2). Because of various constructional features of stator and rotor slots (width, depth, shape, openings etc.) permeance of magnetic channels  $\Lambda_{os}$  differ from permeance of rotatory channels  $\Lambda_{or}$  but the further consideration will be conducted under the simplifying assumption that

$$\Lambda_{os} = \Lambda_{or} = \Lambda_o$$

because the key purpose of the presented theory is to examine influence of the number of stator slots  $Q_s$  and rotor slots  $Q_r$  on the behaviour of the machine.

As far as MMF pattern is concerned, it is assumed, but at the introductory stage only, that the magnetic field is generated by concentrated full pitch coil as it is shown in Fig.2.4 ( $N_s$  – number of turns for full pitch stator coil,  $i_s$  – stator current) and that the number of pole-pairs  $p$  is equal to 1. It is worth reminding that the number of stator teeth  $Q_s$  in induction machines is always even but the number of rotor teeth  $Q_r$  can be either even or odd what results from the peculiar rules of choice of rotor slots. Because of that, it is necessary to consider separately two cases: even and odd number of rotatory magnetic channels  $Q_r$ .

If the number of rotor magnetic channels is even, the total number of stator and rotor teeth in the upper region (U) of the stator coil  $Q_U$  and in the lower region (L) of the stator coil  $Q_L$  are (in spite of the rotor motion) all the time constant (Fig.2.3 – compare with Fig.2.12):

$$Q_U = Q_L = \frac{Q_s + Q_r}{2}. \quad (2.1a)$$

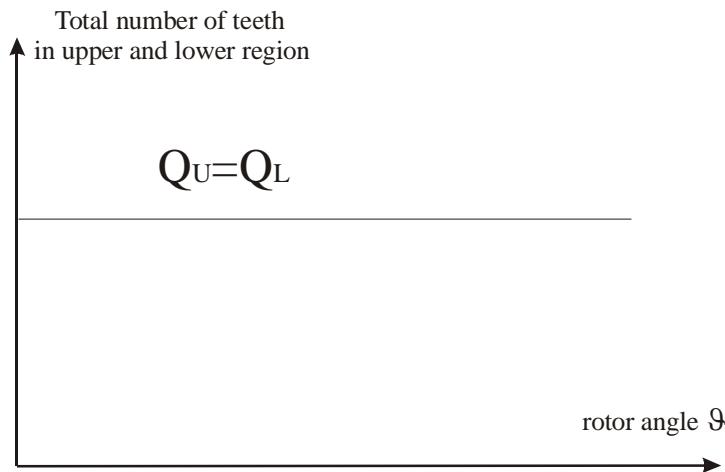


Fig 2.3. Total numbers of stator and rotor teeth in the upper region  $Q_U$  and lower region  $Q_L$  of the stator coil versus rotor angle  $\theta$  for the machine with even number of magnetic channels

Because of that magnetic flux passing through a single magnetic channel in the upper region  $\Delta\Phi_U$  and in the lower region  $\Delta\Phi_L$  (magnetic flux passing from stator to rotor through the air-gap is regarded as positive) are equal

$$\Delta\Phi = \Delta\Phi_U = \Delta\Phi_L. \quad (2.1b)$$

In consequence of that, the permeance of each stationary or rotatory magnetic channel has identical value  $\Lambda_o$  (Fig2.4b), which is independent of the rotor position and can be expressed as:

$$\Lambda_o = \frac{\pi\mu_0 l_e d}{\delta_{ef} (Q_s + Q_r)}, \quad (2.2)$$

where

$l_e$  - equivalent length of machine,  
 $d$  - inner diameter of stator,  
 $\delta_{ef}$  - effective width of the air-gap (regarding Carter factor and core saturation),  
 $\mu_0$  - free-space permeability,  
 $N_s$  - number of turns for full pitch stator coil,  
 $i_s$  - stator current.

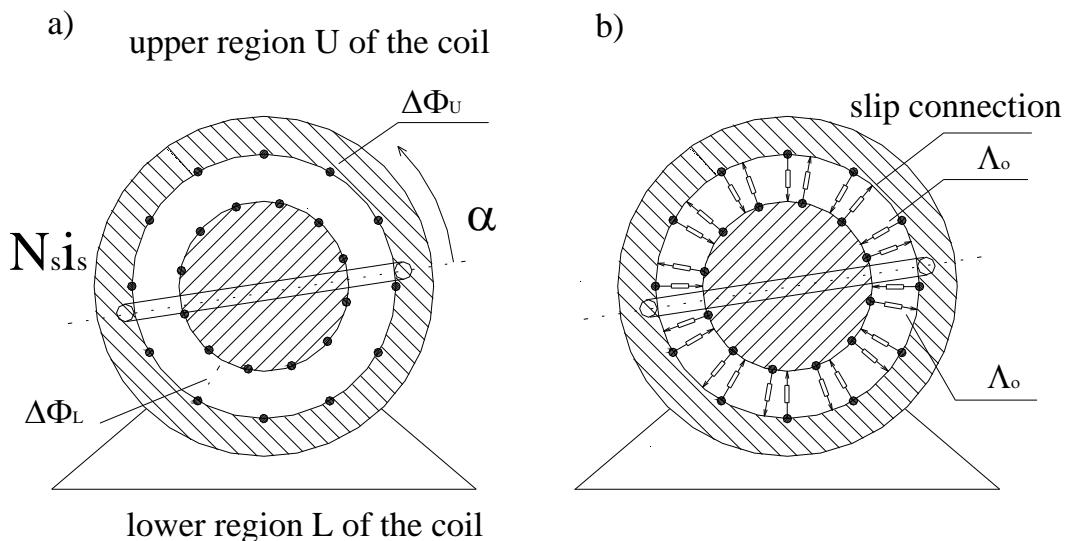


Fig.2.4. Electrical machine with discrete distribution of air-gap permeance:  $Q_s=Q_r=12$   
 (a) magnetic channels in the upper region U and lower region L of the stator coil  
 (b) magnetic circuit of the air-gap

The equivalent magnetic circuit of the air-gap is presented in Figs 2.4b and 2.5a. Magnetic fluxes passing through the magnetic channels in the upper region  $\Delta\Phi_U$  and in the lower region  $\Delta\Phi_L$  are the same (Eqn 2.1b), and can be calculated according to the given equivalent magnetic circuit of the air-gap (Fig.2.5a and Exps 2.1 and 2.2):

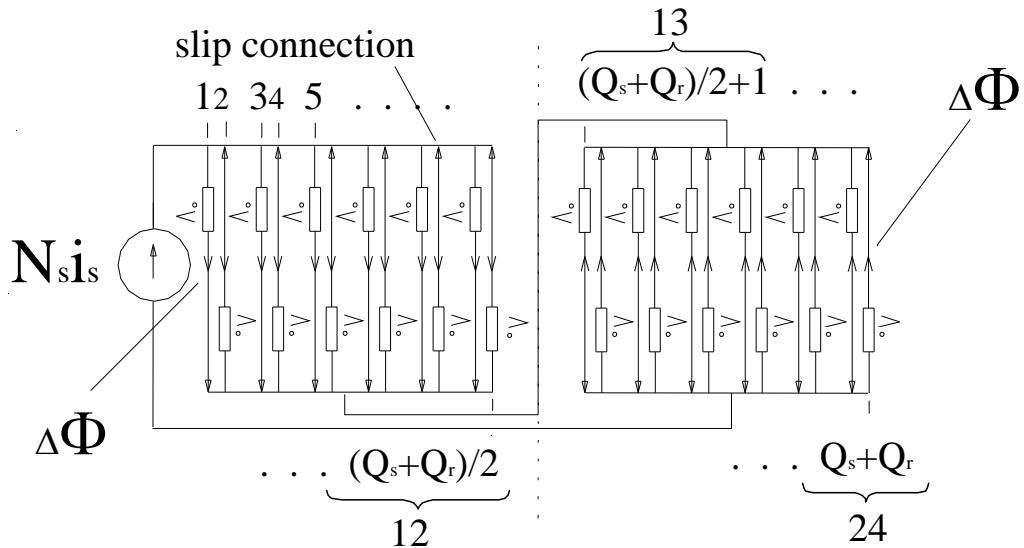
$$\Delta\Phi = \frac{1}{2} N_s i_s \Lambda_o = \frac{1}{2} N_s i_s \frac{\pi \mu_0 l_e d}{\delta_{ef} (Q_s + Q_r)}. \quad (2.3a)$$

Magnetic flux space distribution at certain instant of time along the circumference of the air-gap is illustrated in Fig 2.5b.

All the above-presented exemplary figures (Figs 2.4a,b and 2.5a,b) are referred to the machine with the number of stator magnetic channels equal to the number of rotor magnetic channels:

$$Q_s = Q_r \text{ (the example in point: } Q_s = 12, Q_r = 12). \quad (2.3b)$$

a)



b)

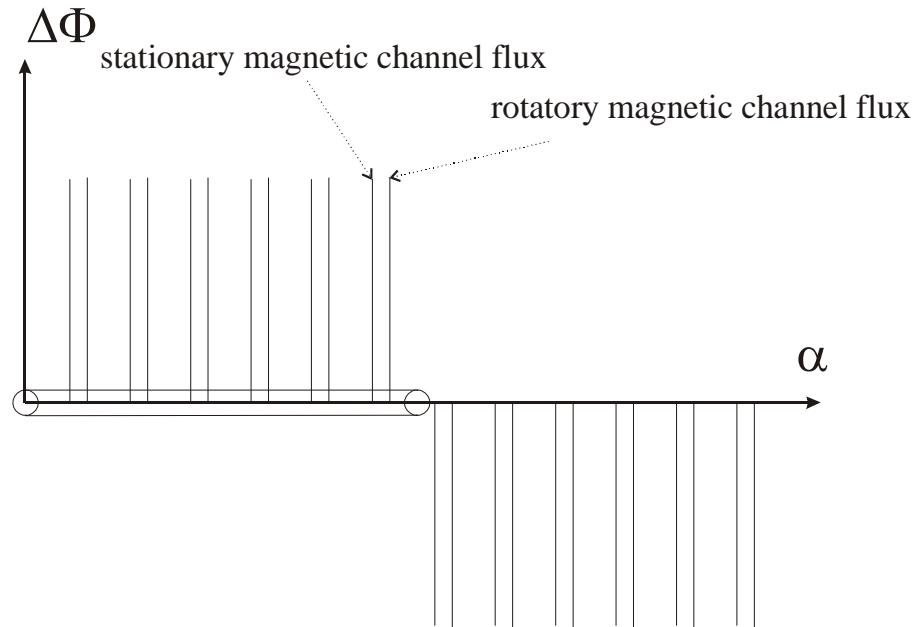


Fig.2.5. (a) Equivalent magnetic circuit of the air-gap corresponding to Fig.2.4b:  $Q_s=Q_r=12$  (b) magnetic flux space distribution along the circumference of the air-gap at certain instant of time

The general case of the machine having different numbers of stator and rotor magnetic channels:

$$Q_s \neq Q_r \text{ (the example in point: } Q_s = 18, Q_r = 12) \quad (2.3c)$$

is considered in Figs 2.6a,b and 2.7a,b.

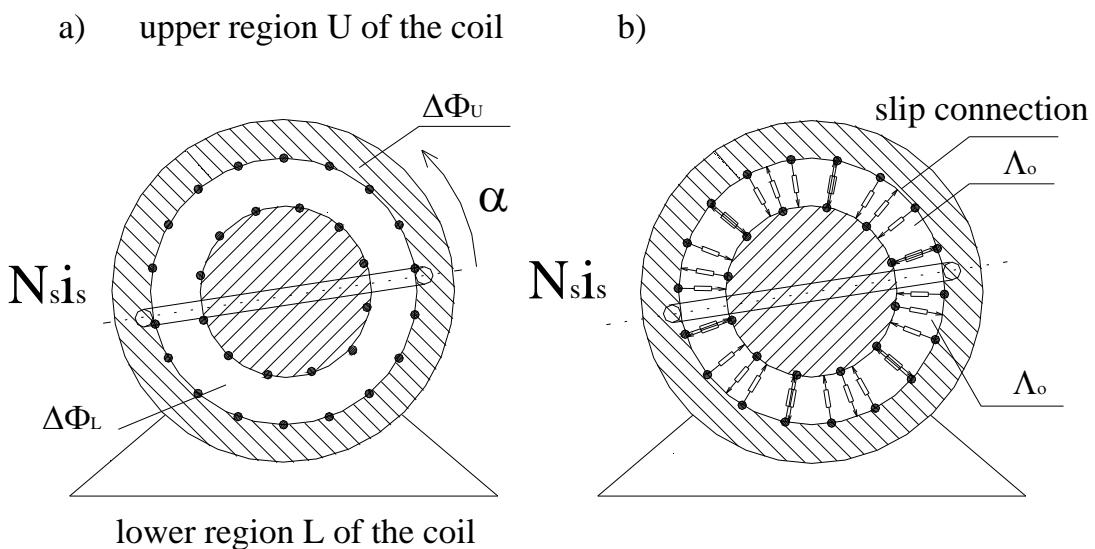


Fig.2.6. Electrical machine with discrete distribution of air-gap permeance:  
 $Q_s=18, Q_r=12$  (a) magnetic channels in the upper region U and lower region L of the stator coil, (b) magnetic circuit of the air-gap

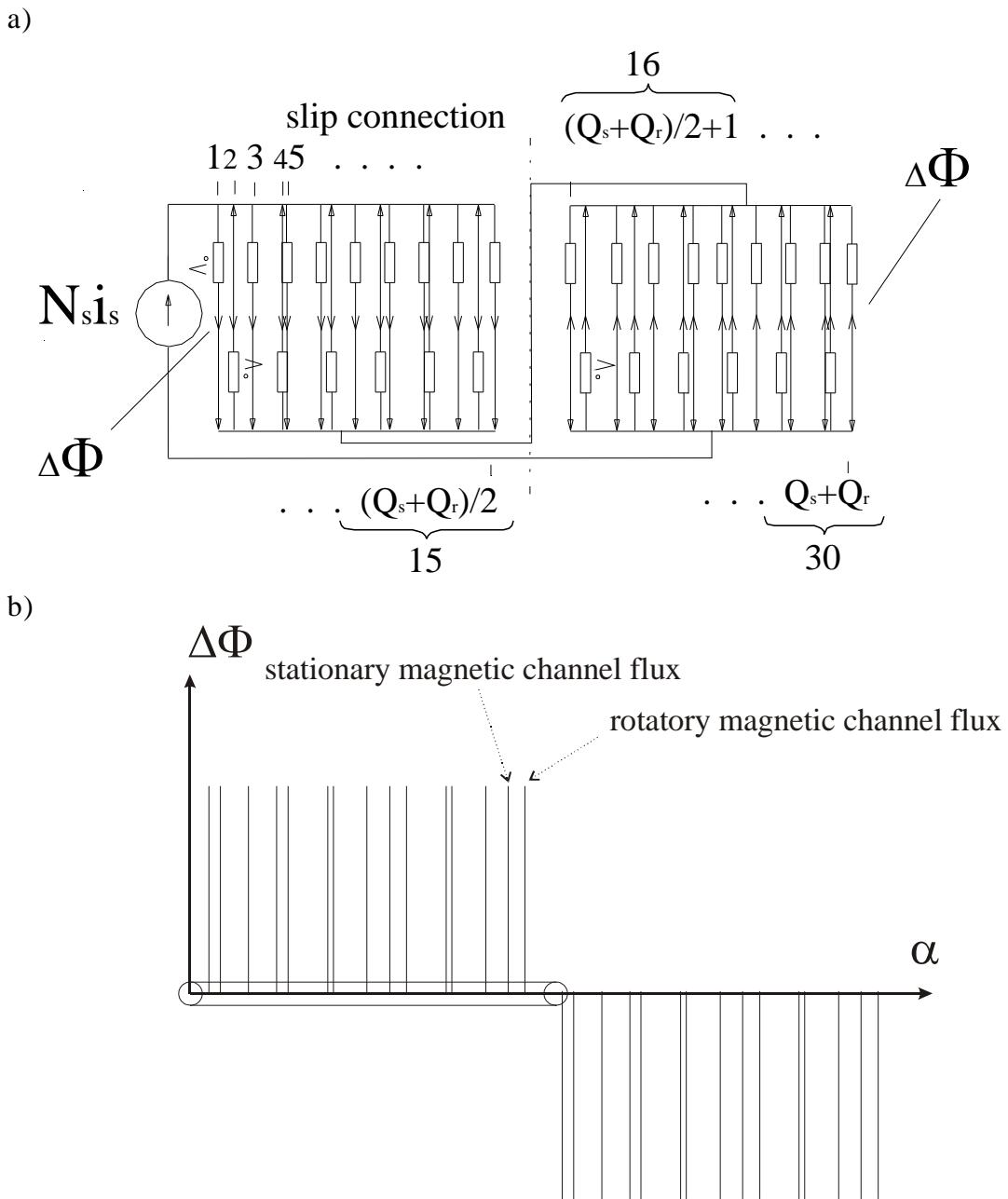


Fig.2.7. (a) Equivalent magnetic circuit of the air-gap corresponding to Fig. 2.6b:  $Q_s=18$ ,  $Q_r=12$  (b) magnetic flux distribution along the circumference of the air-gap

**If the number of rotor magnetic channels is odd** it is necessary to consider the two positions of a rotor: position I presented in Fig. 2.8a and position II presented in Fig.2.10 (the exemplary machine in point has the following numbers of magnetic channels:  $Q_s=12$ ,  $Q_r=11$ ).

## POSITION I

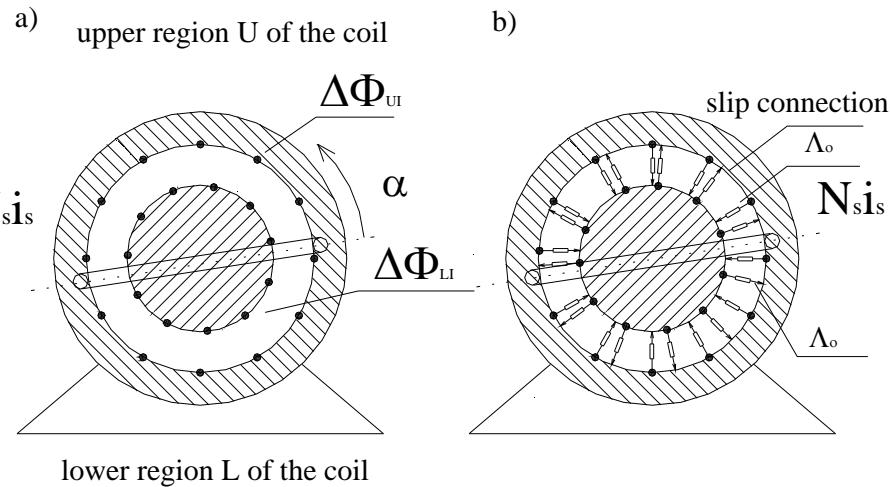


Fig.2.8. Electrical machine with discrete distribution of air-gap permeance in the position I:  $Q_s=12$ ,  $Q_r=11$  (a) magnetic channels in the upper and lower region (UI) of the stator coil (b) magnetic circuit of the air-gap.

For the rotor position I (Fig.2.8a) the number of magnetic channels in the upper region (UI) of the stator coil is equal to:

$$Q_{UI} = \frac{1}{2}(Q_s + Q_r + 1), \quad (2.4a)$$

and in the lower region (LI) equals:

$$Q_{LI} = \frac{1}{2}(Q_s + Q_r - 1), \quad (2.4b)$$

respectively.

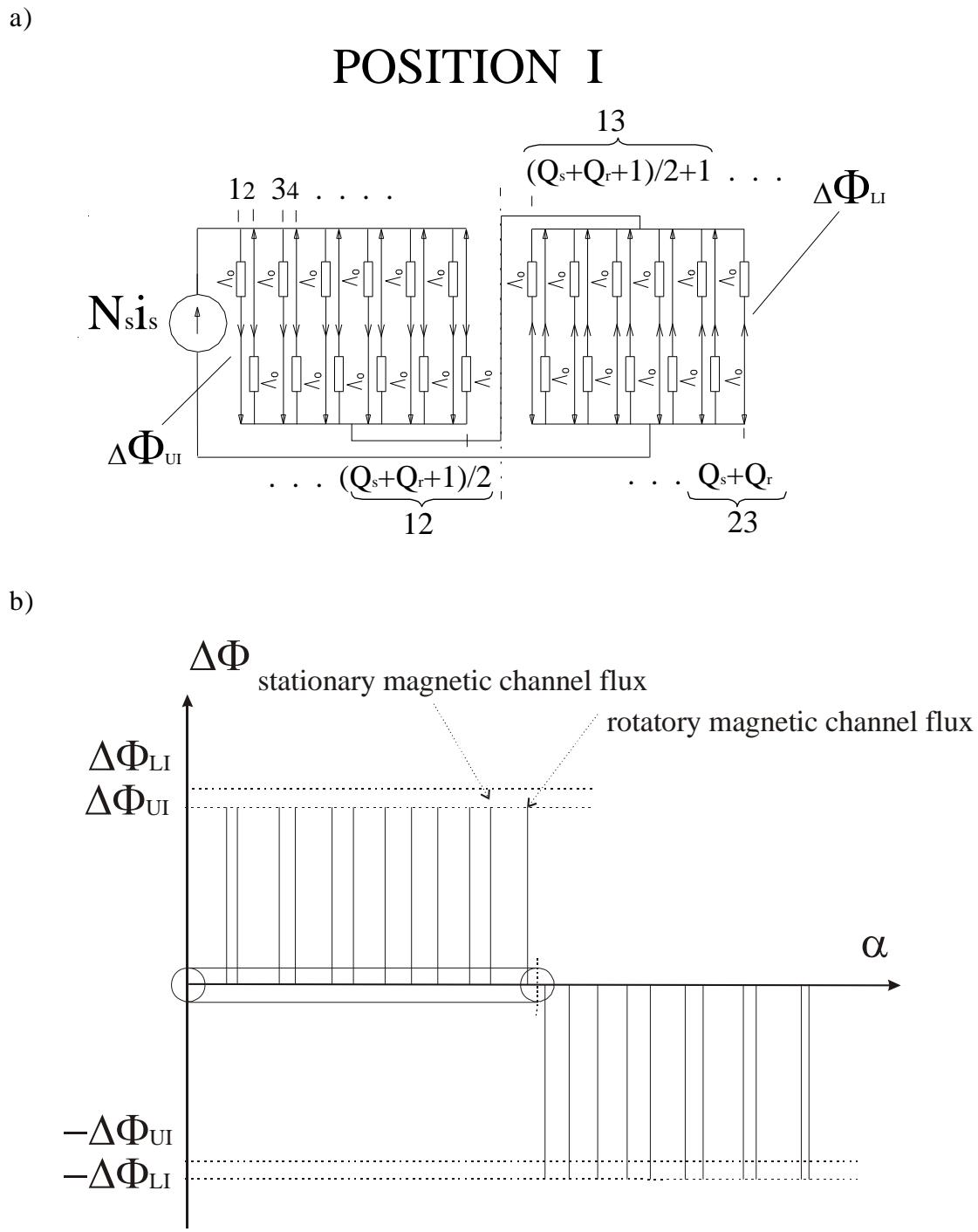


Fig.2.9. (a) Equivalent magnetic circuit of the air-gap corresponding to Fig 2.8b:  $Q_s=12$ ,  $Q_r=11$  (b) magnetic flux space distribution along the circumference of the air-gap at certain instant of time

## POSITION II

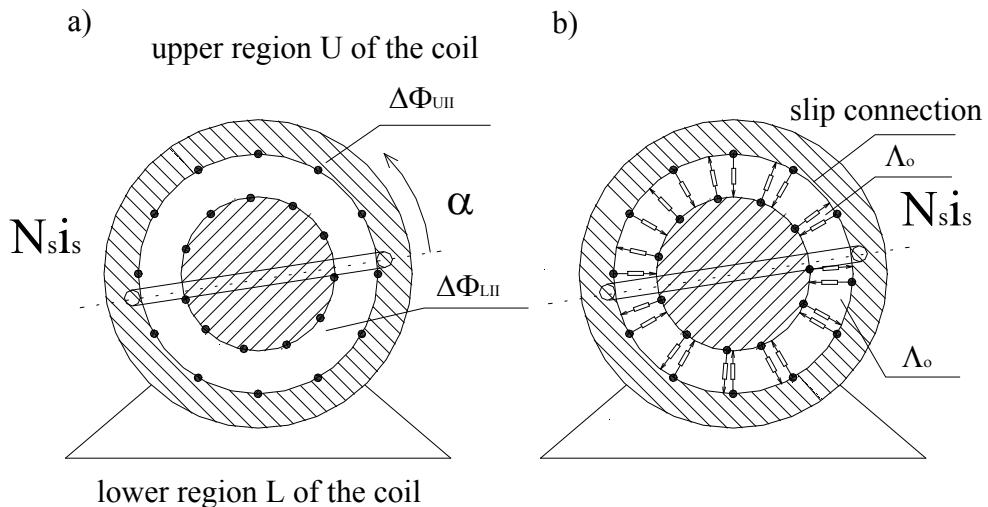


Fig.2.10. Electrical machine with discrete distribution of air-gap permeance at the certain instant of time for the position II:  $Q_s=12$ ,  $Q_r=11$   
 (a) magnetic channels in the upper region UII and lower region LII of the stator coil (b) magnetic circuit of the air-gap

For the rotor position II (Fig.2.10) the number of magnetic channels in the upper region (UII) of the stator coil is equal to:

$$Q_{U\text{II}} = \frac{1}{2}(Q_s + Q_r - 1), \quad (2.4c)$$

and in the lower region (LII) equals

$$Q_{L\text{II}} = \frac{1}{2}(Q_s + Q_r + 1), \quad (2.4d)$$

respectively.

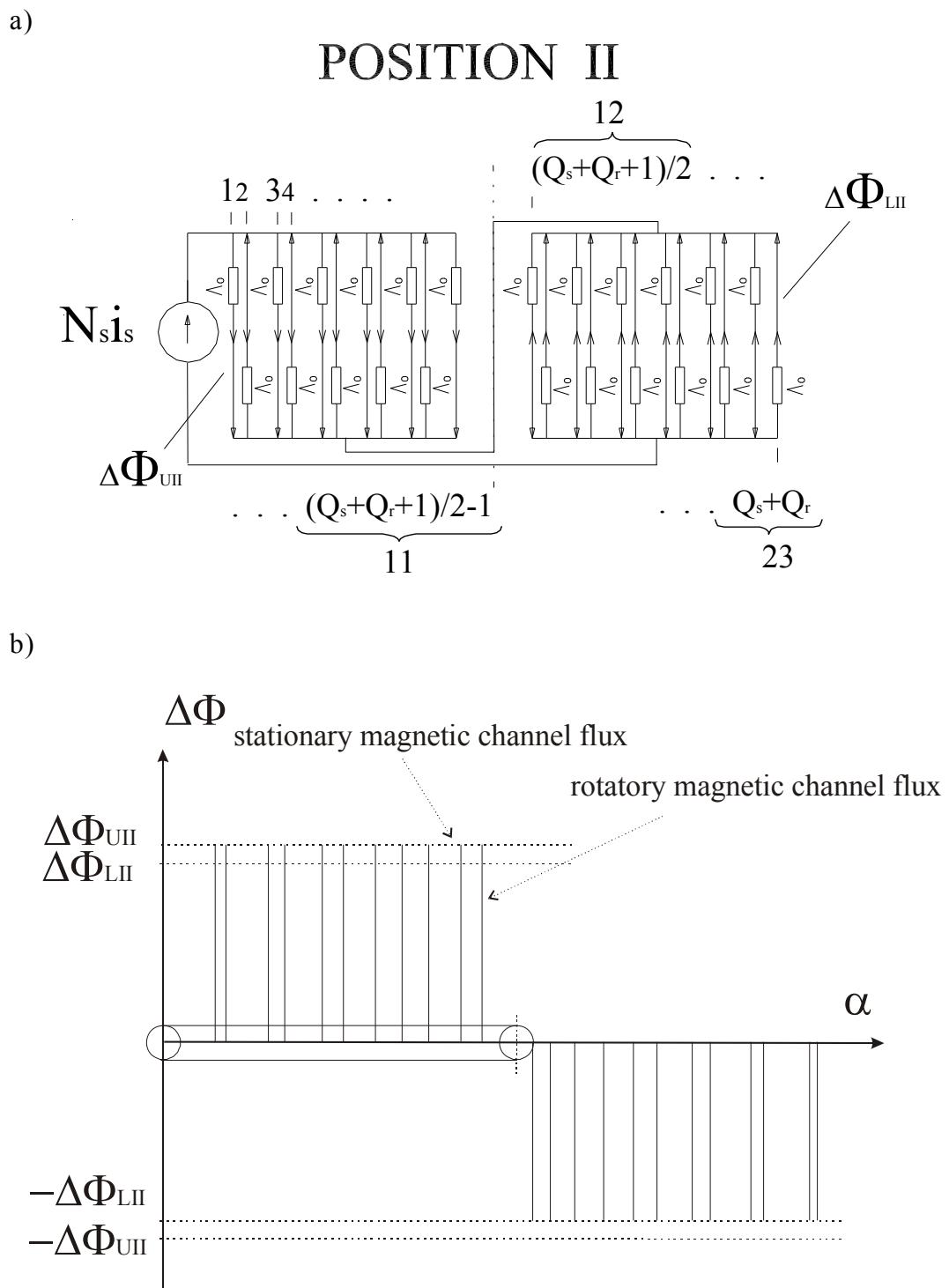


Fig.2.11. (a) Equivalent magnetic circuit of the air-gap corresponding to Fig 2.10b:  $Q_s=12$ ,  $Q_r=11$  (b) magnetic flux space distribution along the circumference of the air-gap at certain instant of time

Summing up, the number of magnetic channels in regions I and II depends on rotor angle  $\vartheta$  and varies in the way presented graphically in Fig.2.12.

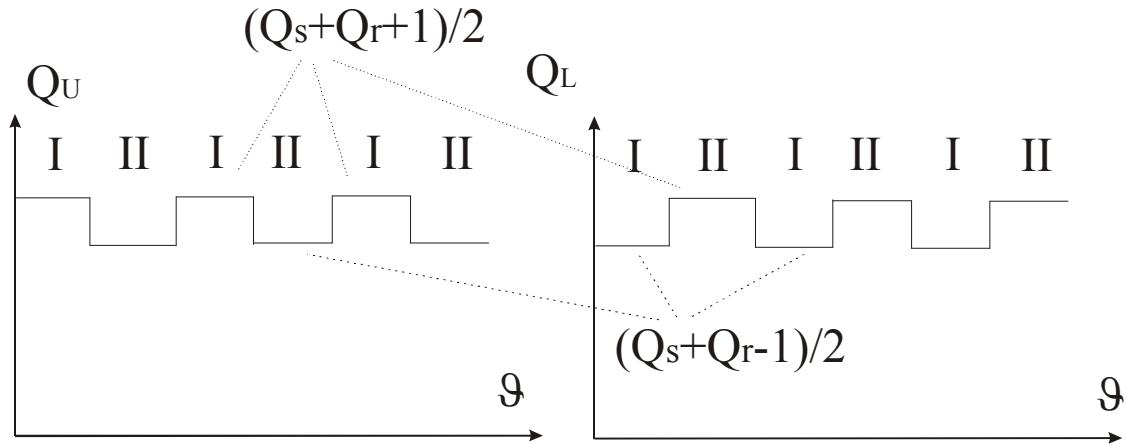


Fig.2.12. Number of magnetic channels in the upper and in the lower region of stator coil for rotor positions I and II versus rotor angle  $\vartheta$  ( $Q_s=12$ ,  $Q_r=11$ )

The equivalent magnetic circuits of the air-gap for the both rotor positions are presented in Figs 2.8b and 2.9a (for rotor position I) and in Figs 2.10b and 2.11a (for rotor position II). It results from these circuits and Fig.2.12 that the instantaneous values of the magnetic fluxes passing through magnetic channels in the upper region  $\Delta\Phi_{UI}$ ,  $\Delta\Phi_{UII}$  and in the lower region  $\Delta\Phi_{LI}$ ,  $\Delta\Phi_{LII}$  are different:

$$\Delta\Phi_{UI} = \Delta\Phi_{LII} = \frac{1}{2} N_s i_s \frac{\pi \mu_0 l_e d}{\delta_{ef}(Q_s + Q_r + 1)}, \quad (2.5)$$

$$\Delta\Phi_{LI} = \Delta\Phi_{UII} = \frac{1}{2} N_s i_s \frac{\pi \mu_0 l_e d}{\delta_{ef}(Q_s + Q_r - 1)}. \quad (2.6)$$

Magnetic flux space distribution along the circumference of the air-gap at certain instant of time for the both rotor positions are illustrated in Figs 2.9b and 2.11b.

An electrical machine with the pair of chorded coils can be considered in the same way, as shown in Figs 2.13 and 2.14a,b.

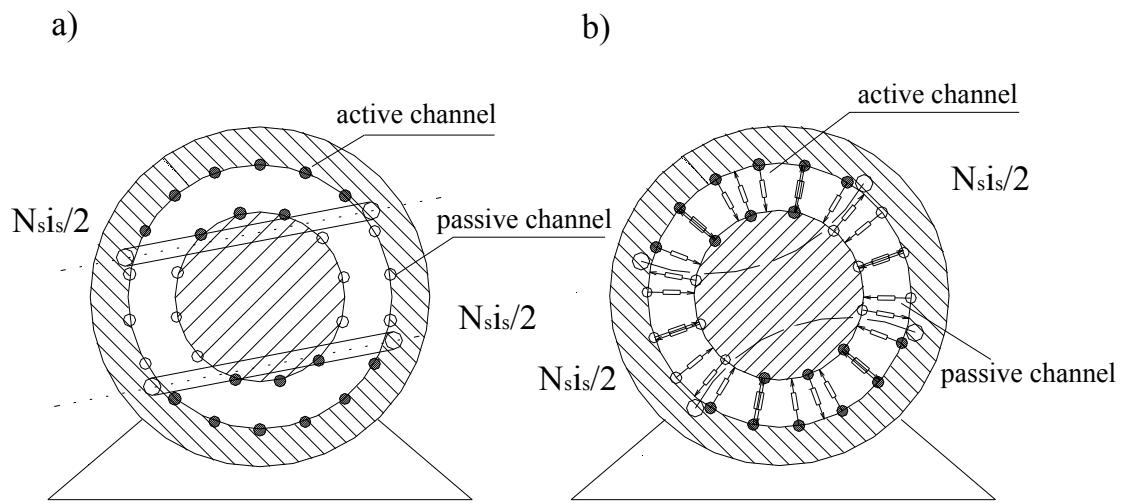
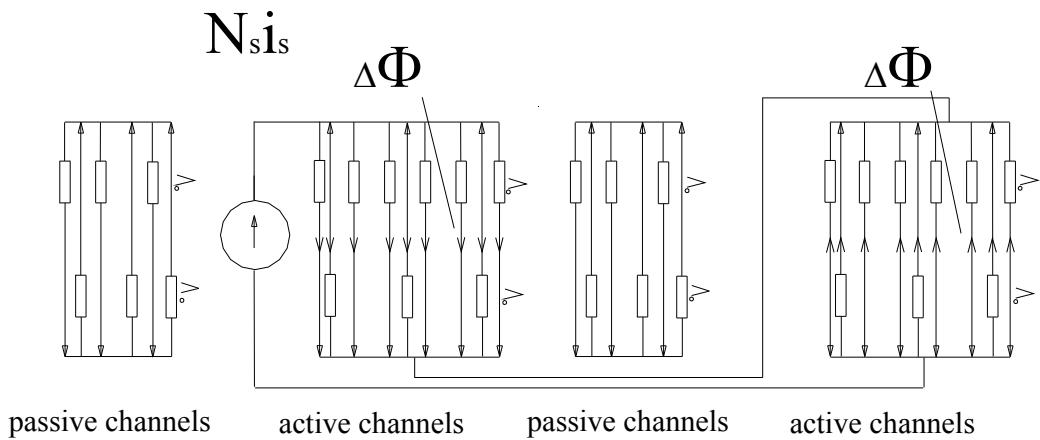


Fig.2.13. Electrical machine with discrete distribution of permeance:  
 $Q_s=18$ ,  $Q_r=12$  having the pair of charded coils: (a) active and passive  
channels (b) magnetic circuit of the air-gap

a)



b)

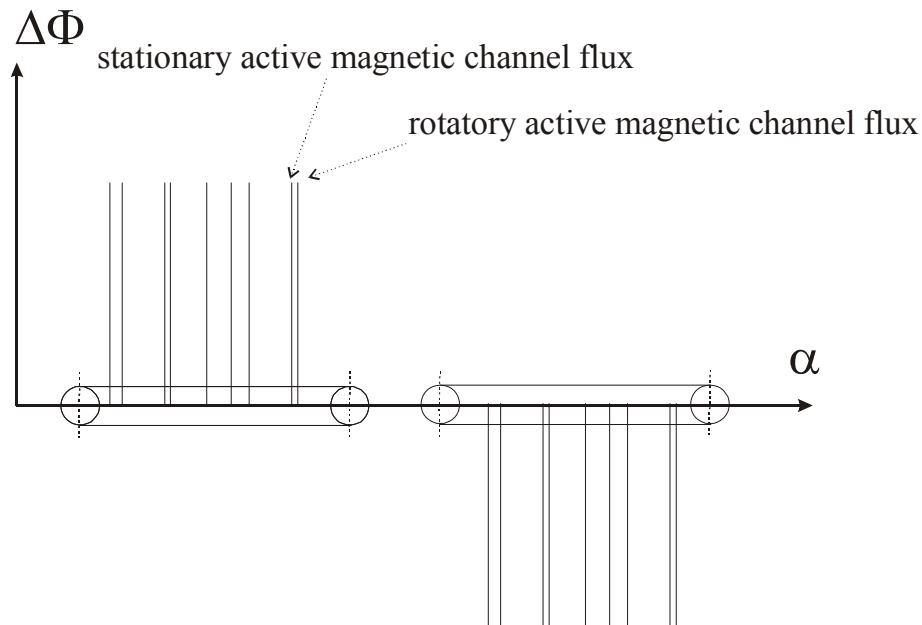


Fig.2.14. (a) Equivalent magnetic circuit of the air-gap corresponding to Fig 2.13b:  $Q_s=18$ ,  $Q_r=12$  (b) magnetic flux space distribution along the circumference of the air-gap at certain instant of time

Space distribution of magnetic flux produced by distributed winding consisting of greater number of coils having different pitches can be determined according to the principle of separation by summing magnetic fluxes generated by individual coils.

## 2.3. INDUCTANCES

Let us consider two coils: the first located on a stator and the second located on a rotor. The angle between the both coils is equal to the angle determining rotor position  $\vartheta$  (Fig.2.15). The mutual inductance between the stator and rotor coils (stator-rotor inductance) is given by the following definition:

$$M_{rs}(\vartheta) = \frac{\Psi_{rs}(\vartheta)}{i_s} = \frac{N_r \Phi_{rs}(\vartheta)}{i_s}, \quad (2.7)$$

where:

$\Psi_{rs}$  - magnetic flux linked with rotor coil and produced by stator coil,

$\Phi_{rs}$  - stator magnetic flux passing through rotor coil,

$i_s$  - current in stator coil,

$N_r$  - number of rotor turns.

Taking into account figures presenting magnetic flux space distribution along the developed circumference of a machine produced by a stator coil (Figs. 2.5b, 2.7b, 2.9b, 2.11b, 2.14b) and completing them with the sketch of the considered rotor coil shifted by the rotor angle  $\vartheta$ , one can easily determine the magnetic flux  $\Phi_{rs}$  by summing the fluxes of magnetic channels  $\Delta\Phi$  enclosed by the rotor coil i.e. located between the left side and right side of the full-pitch coil (Fig.2.15):

$$M_{rs}(\vartheta) = \frac{N_r}{i_s} \sum_{\substack{\text{right side of rotor coil} \\ \text{left side of rotor coil}}} \Delta\Phi, \quad (2.8)$$

or between left side and right side of the pair of chorded coils (Figs.2.16).

The above given formulae (2.8) leads to the algorithm for mutual inductance calculation. In general case space distribution of magnetic permeance of the air-gap can be presented in the following form

$$\Lambda(\alpha) = \sum_{k=1}^{Q_s+Q_r} \Lambda_o \delta(\alpha - \alpha_k), \quad (2.9)$$

where  $\delta(\alpha - \alpha_k)$  denotes Dirac impulse,  $\alpha_k$  is the angle position of  $k^{\text{th}}$  magnetic channel

$$\alpha_k = 2\pi(k-1)/Q_s, \quad \text{for stator } k=1, \dots, Q_s, \quad (2.10a)$$

$$\alpha_k = \theta + 2\pi(k-Q_s-1)/Q_r, \quad \text{for rotor } k=Q_s+1, \dots, Q_s+Q_r. \quad (2.10b)$$

Magnetic flux coupled with rotor coil can be determined by the general formulae

$$\Phi_{rs}(\theta) = \frac{1}{2} ld \int_{-\frac{\pi}{2}Y_r}^{\frac{\pi}{2}Y_r} B_s(\alpha - \gamma_s) d\alpha, \quad (2.11)$$

where

$B_s$  – magnetic flux density induced by stator current,

$\gamma_s$  – angle position of symmetry axis for stator coil,

$Y_r$  – pitch of rotor coil,

$d$  – rotor diameter,

$l$  – machine length.

Basing on the continuity principle for magnetic field (sourceless feature of magnetic field) the Eqn (2.11) can be rewritten in the following form

$$\Phi_{rs}(\theta) = \frac{1}{4p} ld \int_{-\pi}^{\pi} B_s a_r(\alpha - \gamma_r) d\alpha, \quad (2.12)$$

The auxiliary function  $a_r(\alpha)$  for rotor (or stator) winding coils was defined in the following way

$$a_r(\alpha) = \begin{cases} +1 & \text{if } \cos(p\alpha) > \cos(\pi Y_r / 2) \\ -1 & \text{if } \cos(p\alpha) < -\cos(\pi Y_r / 2) \\ 0 & \text{otherwise.} \end{cases} \quad (2.13a)$$

For rotor winding of squirrel-cage motor the auxiliary function takes the form

$$a_r(\alpha) = \begin{cases} +1 & \text{if } \cos(p\alpha) > \cos(\pi Y_r / 2) \\ -1 & \text{if otherwise,} \end{cases} \quad (2.13b)$$

where

$Y_r = 1/pQ_r$  - pitch factor for rotor of squirrel-cage motor.

The magnetic flux density  $B_s$  in the air-gap induced by stator currents has discrete distribution as follows

$$B_s(\alpha) = \sum_{k=1}^{Q_s+Q_r} B_k a_s (\alpha - \gamma_s) \delta(\alpha - \alpha_k), \quad (2.14)$$

where the magnetic flux density  $B_k$  for  $k^{\text{th}}$  channel has the value resulting from the condition

$$\Delta\Phi_k = \frac{1}{2} B_k l d \int_{\alpha_{k-}}^{\alpha_{k+}} \delta(\alpha - \alpha_k) d\alpha = \frac{1}{2} B_k l d. \quad (2.15)$$

The interval  $(\alpha_{k-}, \alpha_{k+})$  surrounds the angle position  $\alpha = \alpha_k$ .

Hence, the magnetic flux coupled with rotor coil equals

$$\Psi_{rs}(\vartheta) = \frac{1}{4p} N_r l d \sum_{k=1}^{Q_s+Q_r} \frac{\Delta\Phi_k}{\frac{1}{2} l d} \int_{-\pi}^{\pi} a_s(\alpha - \gamma_s) a_r(\alpha - \gamma_r - \vartheta) \delta(\alpha - \alpha_k) d\alpha, \quad (2.16)$$

and finally

$$\Psi_{rs}(\vartheta) = \frac{1}{2} N_r \sum_{k=1}^{Q_s+Q_r} \Delta\Phi_k a_s(\alpha_k - \gamma_s) a_r(\alpha_k - \gamma_r - \vartheta). \quad (2.17)$$

The equations (2.8) and (2.17) leads to the same curves for mutual inductances. The essence of the algorithms both (2.8) and (2.17) for the full-pitch coil and for the pair of charded coils is presented in Figs. 2.15 and 2.16 (under assumption of even number of rotor teeth), respectively.

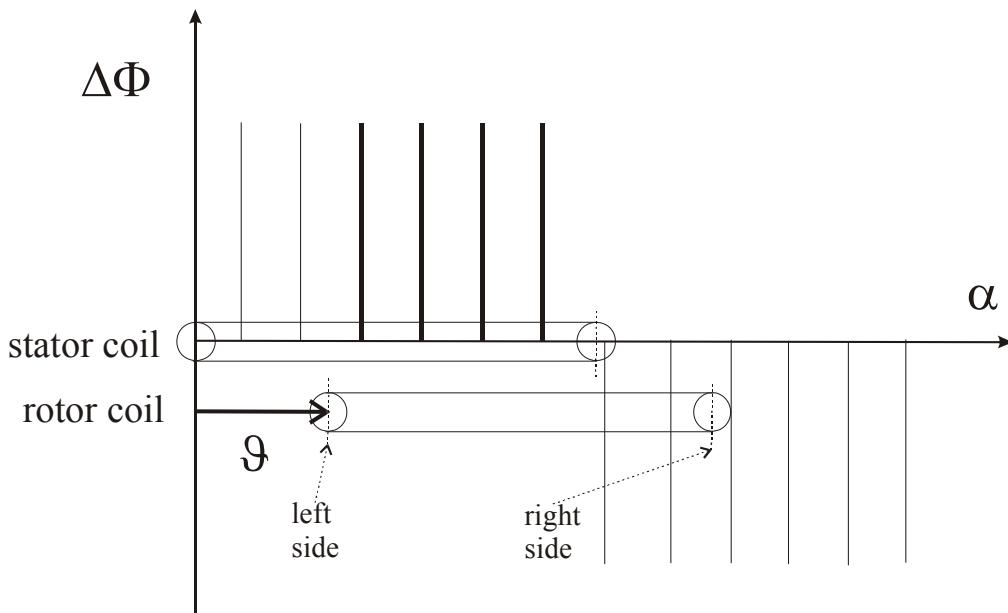


Fig.2.15. Procedure for mutual stator-rotor inductance calculation for full-pitch coils at the given rotor angle  $\theta$

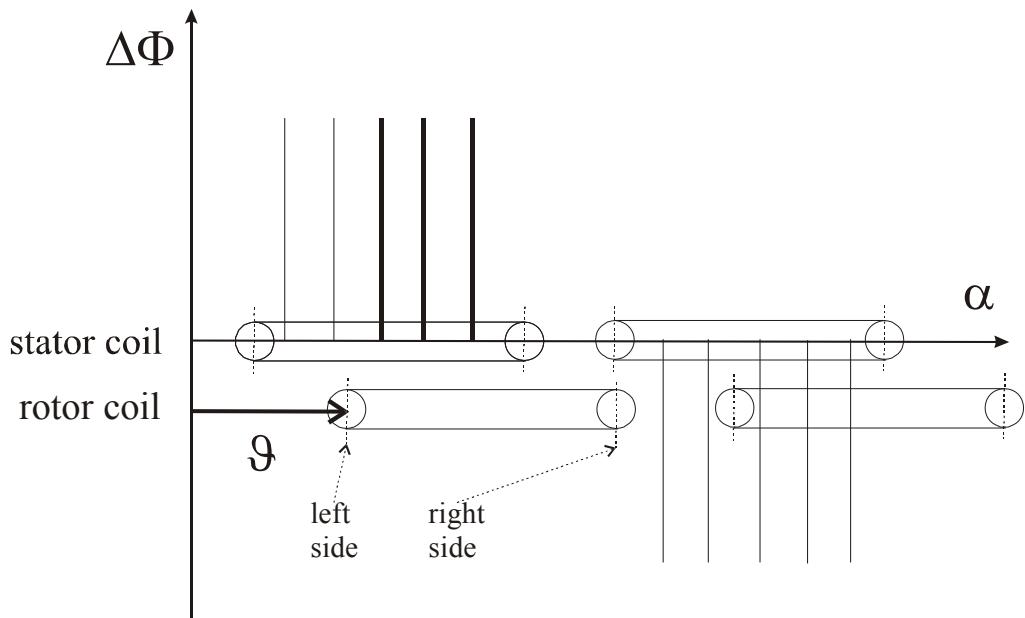


Fig.2.16. Procedure for mutual stator-rotor inductance calculation for pair of charded coils at the given rotor angle  $\theta$

The figures show clearly that in order to determine mutual inductance between stator and rotor coils for the chosen value of the rotor angle  $\theta$ , it is enough to sum the values of magnetic channel fluxes shown in Figs 2.15 and 2.16 with the help of bold lines.

Applying the above-described procedure for different rotor positions one can easily calculate mutual inductance between stator and rotor windings (stator-rotor inductance) as a function of rotor angle  $\theta$ . It results directly from the presented calculation that mutual inductance between stator and rotor coils is piecewise-constant function of rotor angle  $\theta$ . Some examples for the machines having different numbers of stator and rotor teeth with full-pitch coils or charded coils are presented in Figs. 2.17 - 2.19.

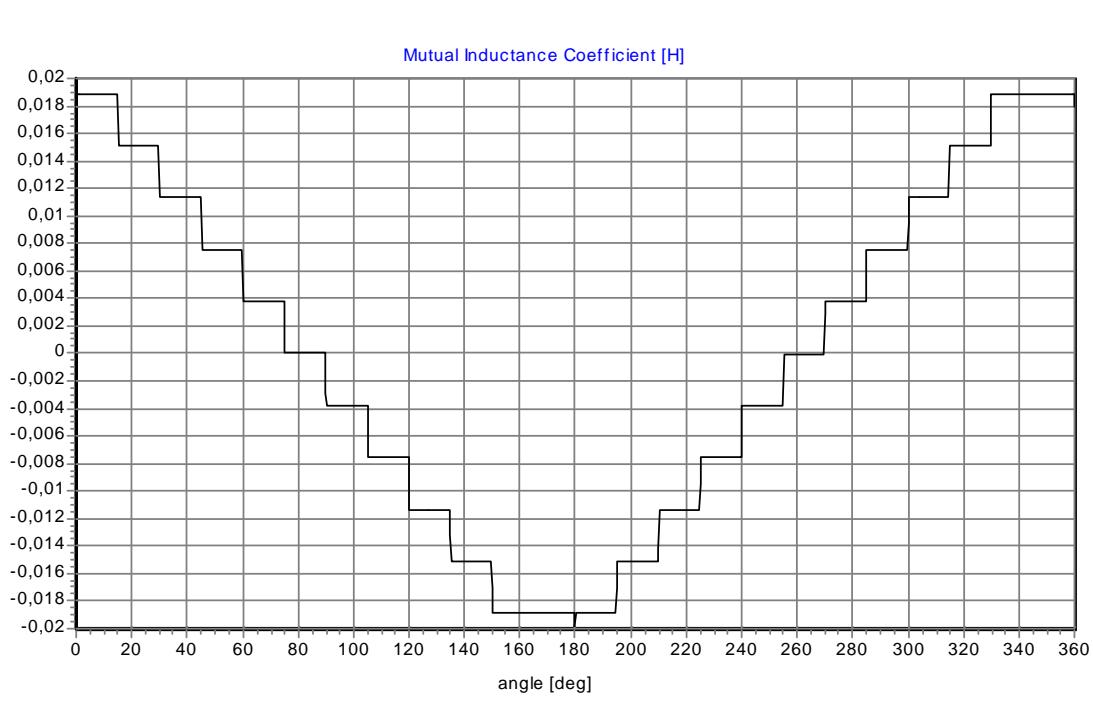


Fig.2.17. Stator-rotor inductance  $M_{sr}$  [H] v. rotor angle [deg] between two full-pitch coils (stator magnetic channels  $Q_s=12$ , rotor magnetic channels  $Q_r=12$ )

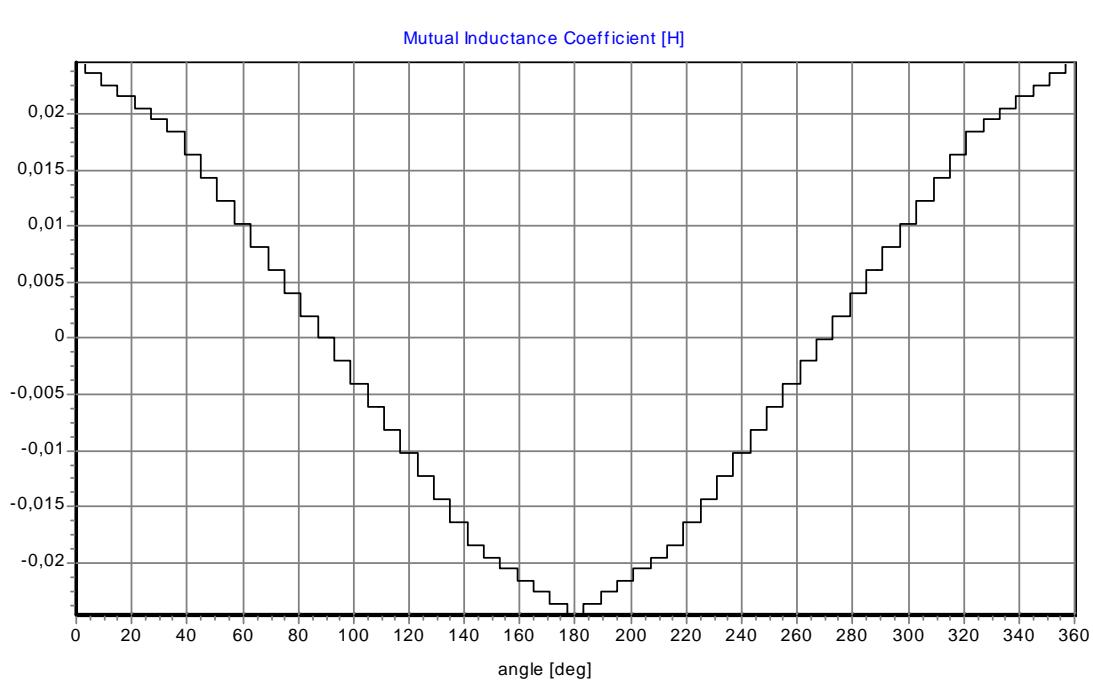


Fig.2.18. Stator-rotor inductance  $M_{sr}$  [H] v. rotor angle [deg] between two chorded coils (stator magnetic channels  $Q_s=12$ , rotor magnetic channels  $Q_r=12$ , stator pitch factor  $Y_s=5/6$ , rotor pitch factor  $Y_r=5/6$ )

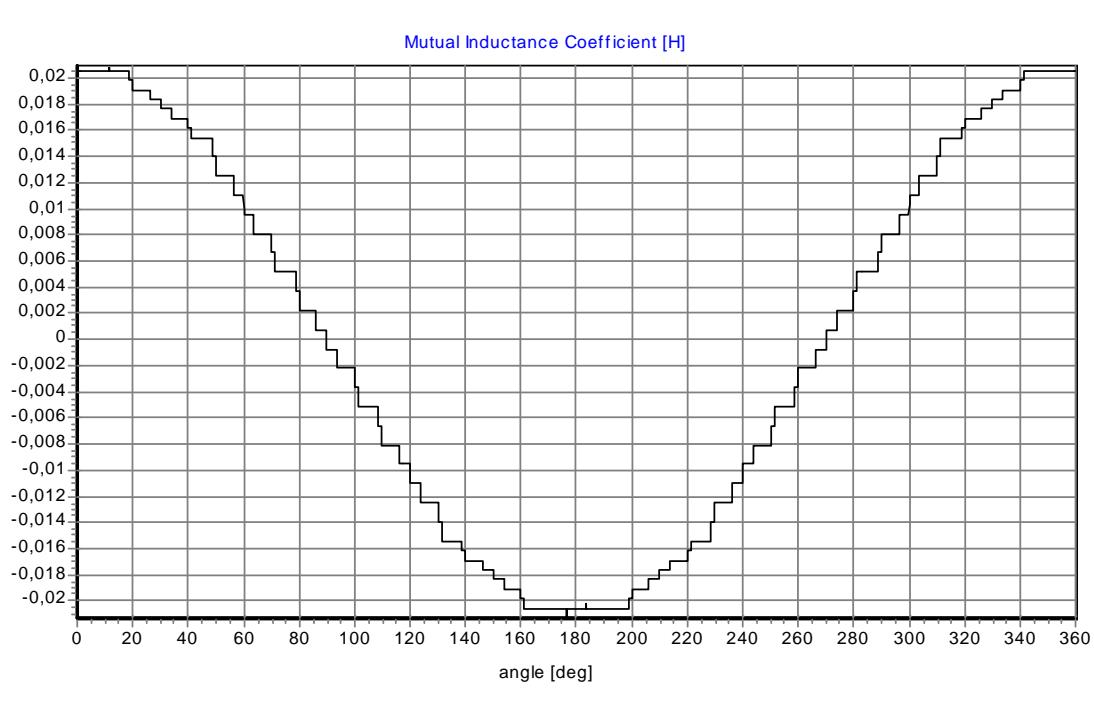


Fig.2.19. Stator-rotor inductance  $M_{sr}$  [H] v. rotor angle [deg] between two charded coils (stator magnetic channels  $Q_s=48$ , rotor magnetic channels  $Q_r=36$ , stator pitch factor  $Y_s=20/24$ , rotor pitch factor  $Y_r=12/18$ )

The stator-rotor inductances as a function of rotor angle  $M_{sr}(\theta)$  have the form of discontinuous piecewise-constant functions with various number of steps. The number of steps  $Q$  and their lengths  $\Delta\theta_k$  depend on the number of stator slots  $Q_s$ , rotor slots  $Q_r$  and on the winding layout. If  $Q_s \neq Q_r$  it can be shown that:

$$Q \leq \frac{Q_s Q_r}{|Q_s - Q_r|}. \quad (2.21)$$

Step changes of stator-rotor inductances occur at each rotor angle at which any coil of the rotor winding passes any stator magnetic channels. Let us denote such values of the rotor angle by:

$$\theta_1, \theta_2, \dots, \theta_k, \theta_{k+1}, \dots$$

and - corresponding to them - instants of time by:

$$t_1, t_2, \dots, t_k, t_{k+1}, \dots$$

As the results of the constant values of the stator-rotor inductances in open intervals  $(\vartheta_k, \vartheta_{k+1})$  for  $k = 0, 1, 2, \dots, Q, Q+1, \dots, 2Q, 2Q+1, \dots$  the submatrix of stator-rotor inductances  $\mathbf{M}_{sr}$  – referring to the machine presented in Fig.1 - is also constant in these intervals  $(\vartheta_k, \vartheta_{k+1})$ . Because of that fact, its form is changing from interval to interval, and the submatrix of stator-rotor inductances will be denoted in successive intervals  $(\vartheta_k, \vartheta_{k+1})$  by  $\mathbf{M}_{sr,k}$ . It results from the periodicity of the stator-rotor inductances that:

$$\mathbf{M}_{sr,k} = \mathbf{M}_{sr,k \bmod(Q)}. \quad (2.22a)$$

Basing on the reciprocal property of mutual inductances, we can write:

$$\mathbf{M}_{sr,k} = \mathbf{M}_{rs,k}^T. \quad (2.22b)$$

Finally, putting together all submatrices of inductances referring to the machine in Fig.1: stator-stator submatrix  $\mathbf{M}_{ss}$ , stator-rotor submatrix  $\mathbf{M}_{sr}$ , rotor-stator submatrix  $\mathbf{M}_{sr}^T$ , rotor-rotor submatrix  $\mathbf{M}_{rr}$  and leakage inductance matrix  $\mathbf{L}_\sigma = \text{diag}[L_{\sigma i}]$  (where  $L_{\sigma i}$  denotes leakage inductance of  $i^{\text{th}}$  winding) – and forming the matrix of inductances as follows:

$$\mathbf{M}_k = \text{diag}[L_{\sigma i}] + \begin{bmatrix} \mathbf{M}_{ss} & \mathbf{M}_{sr,k} \\ \mathbf{M}_{rs,k} & \mathbf{M}_{rr} \end{bmatrix}, \quad (2.23)$$

one gets set of  $Q$  inductance matrices  $\mathbf{M}_k$  (where:  $k=1,2,\dots,Q$ ) for consecutive rotor angle intervals  $(\vartheta_k, \vartheta_{k+1})$  which can be substituted in Eqns. (3.1), (3.5b). The relationships among the consecutive inductance matrices  $\mathbf{M}_k$  resulting from their periodicity are shown in graphical way in Fig.2.20.

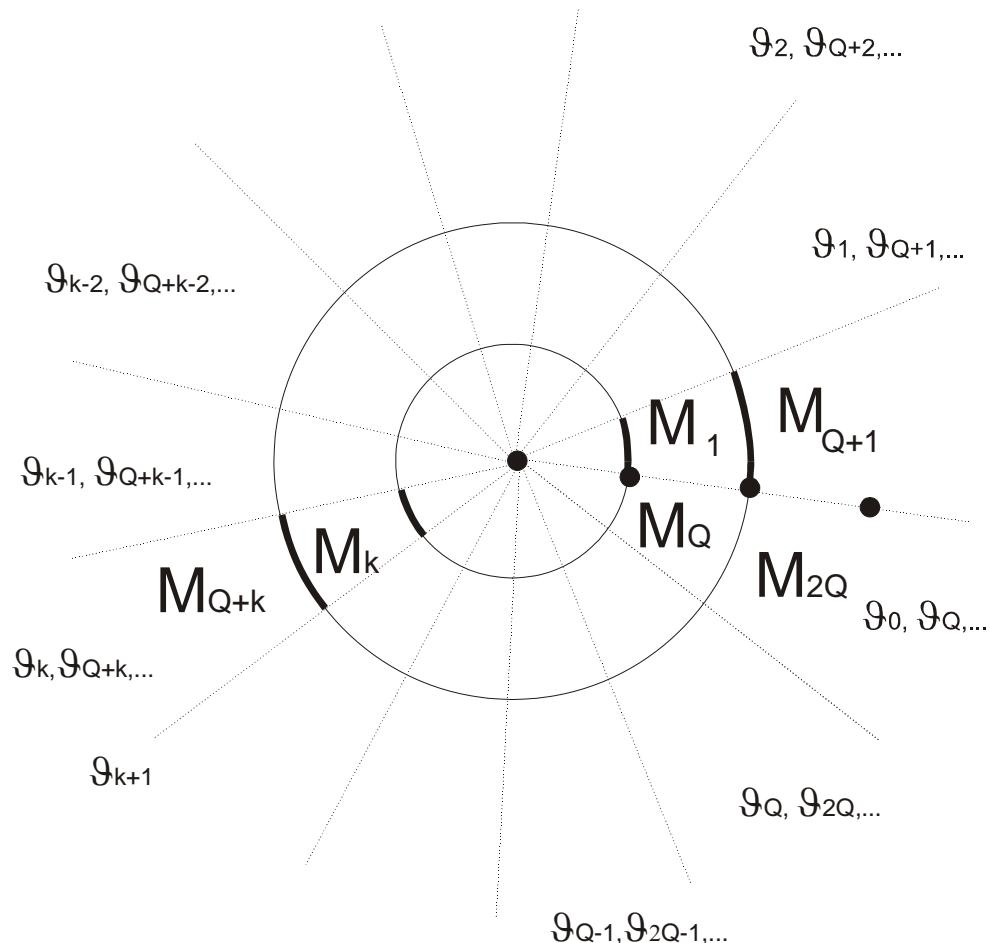


Fig.2.20. Relationships among matrices of inductances for consecutive rotor intervals

The possibility of piecewise-constant representation of stator-rotor inductances (Figs. 2.17-2.19) is of great importance for analysis of a machine and leads to a specific method of solving differential equations resulting from the fact that - according to Eqn (1.3) - electromagnetic torque is equal to zero in open intervals in which stator-rotor inductances are constant:  $T_e = 0$ , and has the form of Dirac impulse at endpoints of intervals corresponding to step discontinuities of inductances:

$$T_e = -\Delta W_{m,k} \delta(\vartheta - \vartheta_k), \quad (2.24)$$

where  $\delta(\vartheta - \vartheta_k)$  denotes the Dirac impulse at the rotor position  $\vartheta_k$ ,  $\Delta W_{m,k}$  is magnetic energy change given in the next chapter by Exp. (3.7).



### 3. TRANSIENT-STATE MATHEMATICAL MODEL OF INDUCTION MACHINE

#### 3.1. CONCEPT OF ALTERNATE STEP-BY-STEP ANALYSIS

The essence of the proposed method is that the dynamic state of a machine is analysed alternately step-by-step in open intervals  $(\vartheta_k, \vartheta_{k+1})$  according to the Procedure A - Fig.3.1 and at endpoints  $\vartheta_{k+1}$  of intervals according to the Procedure B - Fig.3.2 (where  $k=0,1,2,\dots$ ); both these procedures involve considerably simpler relationships than Eqns (1.1), (1.2), (1.3).

#### PROCEDURE A

For the intervals  $(\vartheta_k, \vartheta_{k+1})$  corresponding to time intervals  $(t_k, t_{k+1})$ :

- **voltage equations** are linear and include time-constant coefficients:

$$U = RI + M_k \frac{dI}{dt}, \quad (3.1)$$

*initial condition:  $I(t_k)$ ,*

- **torque equation** (because of relation  $T_e=0$ ) is reduced to the form:

$$J \frac{d\Omega_m}{dt} = T_m, \quad (3.2)$$

*initial condition:  $\Omega_m(+t_k)$ .*

The block diagram corresponding to Eqns (3.1) and (3.2) is presented in Fig.3.1. As may be seen, in open intervals  $(t_k, t_{k+1})$  the electrical system and the mechanical system do not interact, which means that the voltage-current

equations (3.1) and the torque equation (3.2) are autonomous. It is worth emphasizing that the speed variation is determined only by the value of external mechanical torque  $T_m$ :

$$\Omega_m(t) = \Omega_m(+t_k) + \frac{1}{J} \int_{t_k}^t T_m dt. \quad (3.3a)$$

In the case of constant mechanical torque:  $T_m = \text{const}$ , the change of speed is described by the linear function:

$$\Omega_m(t) = \Omega_m(+t_k) + \frac{T_m}{J} (t - t_k). \quad (3.3b)$$

If there is no load ( $T_m = 0$ ) the speed does not change in the considered interval of time and has a constant value all the time:

$$\Omega_m(t) = \Omega_m(+t_k). \quad (3.3c)$$

The value of time instant  $t_{k+1}$  or - in other words - the length of the time interval  $\Delta t_k = t_{k+1} - t_k$  results, in general, from the integral equation:

$$\Delta \vartheta_k = \vartheta_{k+1} - \vartheta_k = \int_{t_k}^{t_{k+1}} \Omega_m(t) dt, \quad (3.4a)$$

where  $\Omega_m(t)$  means the solution of the equation (3.2),  $\Delta \vartheta_k$  is the interval of rotor angle in which stator-rotor inductances are constant (see Figs 2.17-2.19).

Substituting Eqn (3.3a) in Eqn (3.4a) one obtains the equation:

$$\Delta \vartheta_k = \Omega_m(+t_k) \Delta t_k + \frac{1}{J} \int_{t_k}^{t_{k+1}} \int_{t_k}^t T_m dt, \quad (3.4b)$$

which leads to the algebraic quadratic equation for the most frequent case of constant load ( $T_m = \text{const}$ ) as follows:

$$\Delta \vartheta_k = \Omega_m(+t_k) \Delta t_k + \frac{T_m}{2J} (\Delta t_k)^2, \quad (3.4c)$$

where  $\Delta t_k = t_{k+1} - t_k$ .

## THE ELECTROMECHANICAL SYSTEM

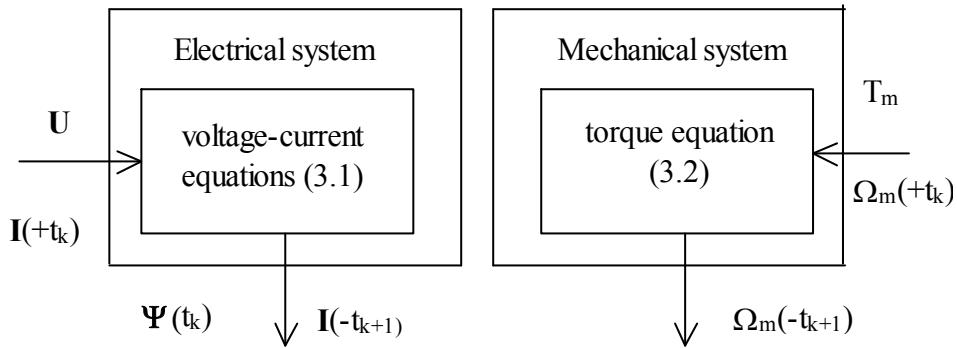


Fig.3.1. Block diagram of electrical machine for time intervals  $(t_k, t_{k+1})$

- Procedure A

## PROCEDURE B

For the endpoints  $\vartheta_{k+1}$  corresponding to instants of time  $t_{k+1}$ :

- magnetic fluxes coupling all windings do not change their values (continuity condition):

$$\Psi(-t_{k+1}) = \Psi(+t_{k+1}) = \Psi(t_{k+1}), \quad (3.5a)$$

hence:

$$\mathbf{M}_k \mathbf{I}(-t_{k+1}) = \mathbf{M}_{k+1} \mathbf{I}(+t_{k+1}), \quad (3.5b)$$

- continuity condition for flux linkages (3.5a) determines the values of current steps:

$$\Delta \mathbf{I}(t_{k+1}) = \mathbf{I}(+t_{k+1}) - \mathbf{I}(-t_{k+1}) = (\mathbf{M}_{k+1}^{-1} - \mathbf{M}_k^{-1}) \Psi(t_{k+1}), \quad (3.6)$$

- change of magnetic field energy stored in windings is equal to:

$$\Delta E_m(t_{k+1}) = \frac{1}{2} \Psi(t_{k+1})^T \Delta \mathbf{I}(t_{k+1}) = \Delta E_{m,k}, \quad (3.7)$$

- electromagnetic torque has the form of Dirac impulse:

$$T_e = -\Delta E_m(t_{k+1}) \delta(\vartheta - \vartheta_{k+1}), \quad (3.8)$$

- step change of speed results either from the torque equation:

$$J \frac{d\Omega_m}{dt} = -\Delta E_m(t_{k+1}) \delta(\vartheta - \vartheta_{k+1}), \quad (3.9)$$

or from the energy conservation law:

$$\Delta E_k + \Delta E_{m,k} = 0, \quad (3.10)$$

where  $\Delta E_k$  is the change of kinetic energy at time  $t_{k+1}$ .

According to the Eqn (3.10) mechanical speed change can be determined with the help of the following relation:

$$\Omega_m^2(+t_{k+1}) - \Omega_m^2(-t_{k+1}) = -\frac{1}{J} \Psi(t_{k+1})^T \Delta I(t_{k+1}). \quad (3.11)$$

The block diagram corresponding to Procedure B is presented in Fig.3.2.

### THE ELECTROMECHANICAL SYSTEM

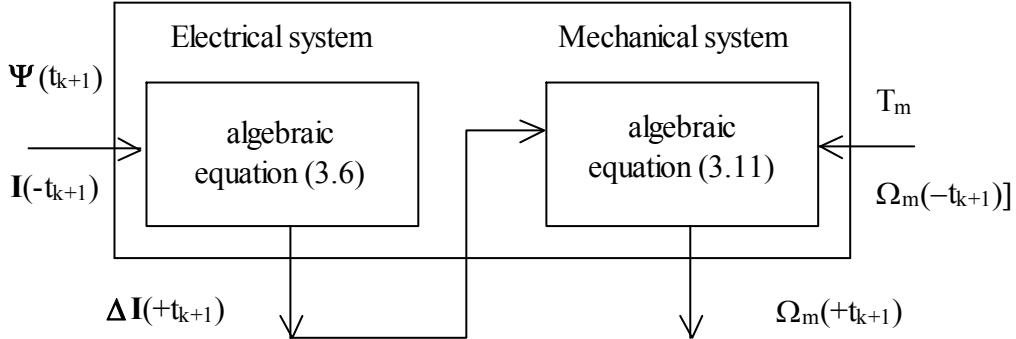


Fig.3.2. Block diagram of electrical machine for instants of time  $t_{k+1}$   
- Procedure B

As it may be seen, the interaction between the electrical and the mechanical systems has uni-directional character and is represented by only one arrow (instead of the two arrows shown in Fig.3.1).

Applying alternately Procedure A and Procedure B, we can analyse **step-by-step** the dynamic behaviour of a machine in an arbitrarily chosen interval of

time (Fig.3.3). This way of calculation is the reason for terming the method **step-by-step analysis of induction machines**.

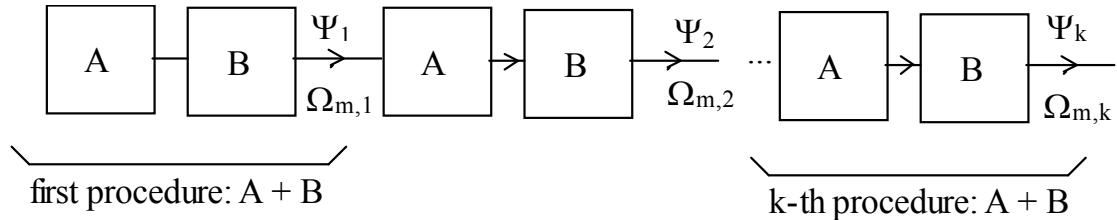


Fig.3.3. Idea of **step-by-step analysis**

### 3.2. COMPUTER SIMULATION BASED ON RECURRENT SEQUENCE RELATIONS

It is convenient to write the voltage equations (3.1) in terms of flux linkage variables:

$$\frac{d}{dt} \Psi = \mathbf{A}_k \Psi + \mathbf{U}, \quad (3.12)$$

where  $\Psi = \mathbf{M}_k \mathbf{I}$ ,  $\mathbf{A}_k = -\mathbf{R}\mathbf{M}_k^{-1}$ ,

which enable the solution of Eqn (3.12) to be expressed in each consecutive interval  $(t_k, t_{k+1})$  in the following analytical form:

$$\Psi(t) = \exp\{\mathbf{A}_k(t - t_k)\}\Psi(t_k) + \int_{t_k}^t \exp\{\mathbf{A}_k(t - \tau)\}\mathbf{U}(\tau) d\tau. \quad (3.13)$$

With the help of Eqn (3.13), the variables of flux linkages at the end of an arbitrary  $(k+1)$ -th interval can be expressed in relation to the values of the variables at the beginning of the previous  $k$ -th interval, as follows:

$$\Psi_{k+1} = E_{k+1} \Psi_k + F_{k+1}, \quad (3.14)$$

where

$$\Psi_k = \Psi(t_k) \quad (3.15a)$$

$$\Psi_{k+1} = \Psi(t_{k+1}), \quad (3.15b)$$

$$E_{k+1} = \exp\{A_k(t - t_k)\}, \quad (3.15c)$$

$$F_{k+1} = \int_{t_k}^{t_{k+1}} \exp\{A_k(t_{k+1} - \tau)\} U(\tau) d\tau. \quad (3.15d)$$

In the same manner, with the help of Eqns (3.11), (3.14) and (3.15), the right-hand value of speed  $\Omega_{m_{k+1}}^+ = \Omega_m(+t_{k+1})$  at the end of the (k+1)-th interval can be expressed in relation to the right-hand value of speed  $\Omega_{m_k}^+ = \Omega_m(+t_k)$  at the beginning of the previous k-th interval. For instance, in the case of constant mechanical torque ( $T_m = \text{const}$ ), one obtains:

$$\Omega_{m_{k+1}}^+ = \sqrt{(\Omega_{m_k}^-)^2 - \frac{1}{J} \Psi_{k+1}^T (\mathbf{M}_{k+1}^{-1} - \mathbf{M}_k^{-1}) \Psi_k}, \quad (3.16a)$$

$$\Omega_{m_{k+1}}^- = \Omega_{m_k}^+ + \frac{T_m}{J} (t_{k+1} - t_k), \quad (3.16b)$$

where the time instant  $t_{k+1}$  is determined in relation to the time instant  $t_k$  by Eqn (3.4b).

The foregoing recurrent equations (3.14), (3.15), (3.16a), (3.16b) can be regarded as an analytical discrete solution of the voltage and torque differential equations enabling the dynamic behaviour of a machine in an arbitrarily chosen period of time to be determined at the time instants:  $t_1, t_2, \dots, t_k, t_{k+1}, \dots$ .

The solution has the form of the following sequences:

$$\Psi_0, \Psi_1, \Psi_2, \dots, \Psi_{k-1}, \Psi_k, \Psi_{k+1}, \dots$$

$$\Omega_{m_0}, \Omega_{m_1}^-, \Omega_{m_1}^+, \Omega_{m_2}^-, \Omega_{m_2}^+, \dots, \Omega_{m_{k-1}}^-, \Omega_{m_{k-1}}^+, \Omega_{m_k}^-, \Omega_{m_k}^+, \Omega_{m_{k+1}}^-, \Omega_{m_{k+1}}^+, \dots$$

where  $\Psi_0, \Omega_{m_0}$  stand for the initial conditions for linkage fluxes and speed (at the beginning of the first interval).

It should be emphasized that the above given sequence relations can be quickly calculated with the help of a computer. This results from relatively low value of the number of steps  $Q$  as well as from periodicity of the stator-rotor inductances. It must be underlined that, because of the property Eqn (2.10a), the inductance inverse matrices  $\mathbf{M}_k^{-1}$  (for  $k = 1, 2, \dots, Q, Q+1, \dots, 2Q, 2Q+1, \dots$ ) included in Eqn (3.6) are calculated only once at the first turn of the rotor (only for  $k = 1, 2, \dots, Q$  see Eqn. 2.10a and Fig.2.20).

A computer program was elaborated for analysis of a 3-phase induction machine with given number of stator and rotor teeth (magnetic channels) according to the described method. The results of the computer simulations (starting characteristics: speed v. time, Dirac impulses of electromagnetic torque v. time) for exemplary motors having different number of magnetic channels are presented in Figs. 3.4. - 3.7.

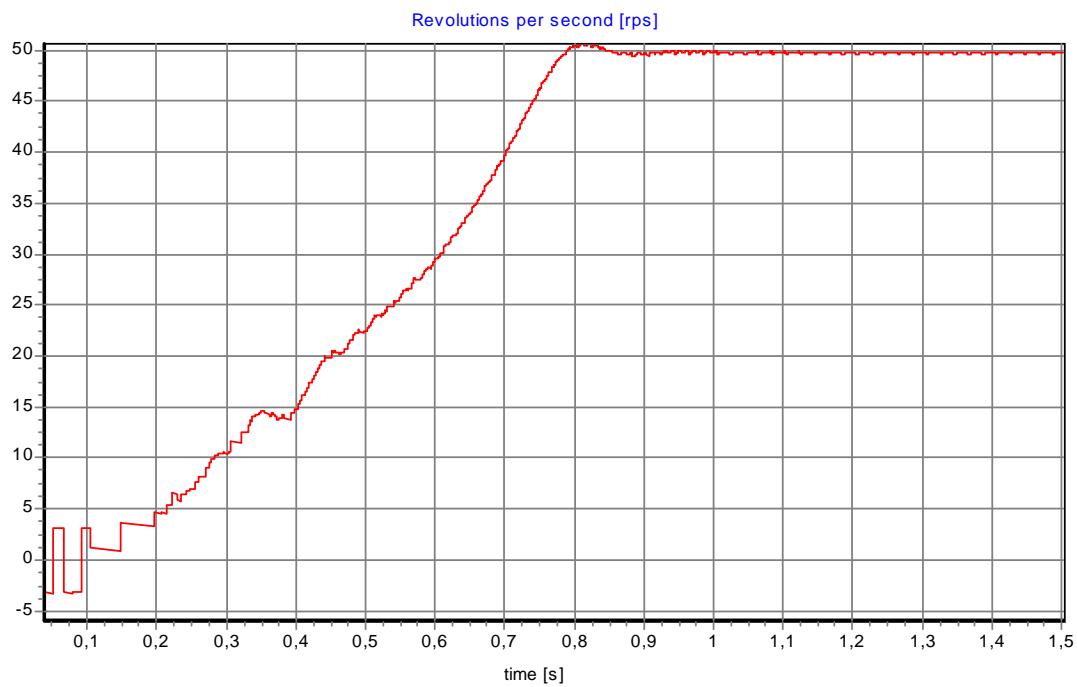


Fig.3.4. Speed  $n$  [rps] v. time  $t$  [s] curve for non-loaded machine (stator magnetic channels  $Q_s=12$ , rotor magnetic channels  $Q_r=12$ ,  $Y_s=5/6$ ,  $Y_r=1$ )

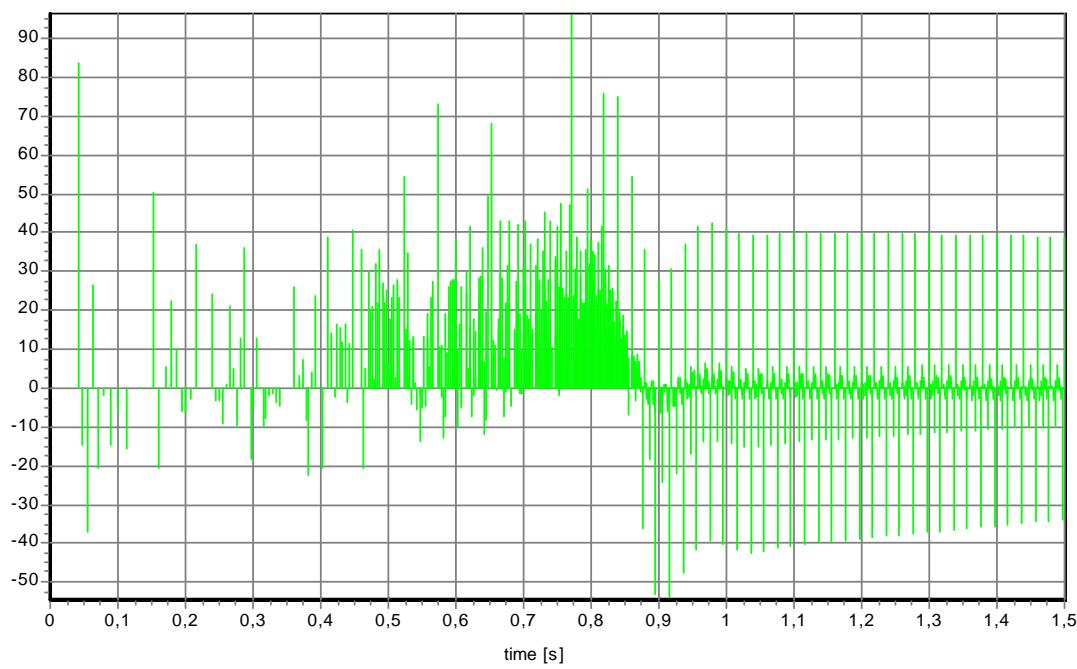


Fig.3.5. Dirac impulses of electromagnetic torque [Nm] v. time  $t$  [s] curve for non-loaded machine (stator magnetic channels  $Q_s=12$ , rotor magnetic channels  $Q_r=12$ ,  $Y_s = 5/6$ ,  $Y_r = 1$ )

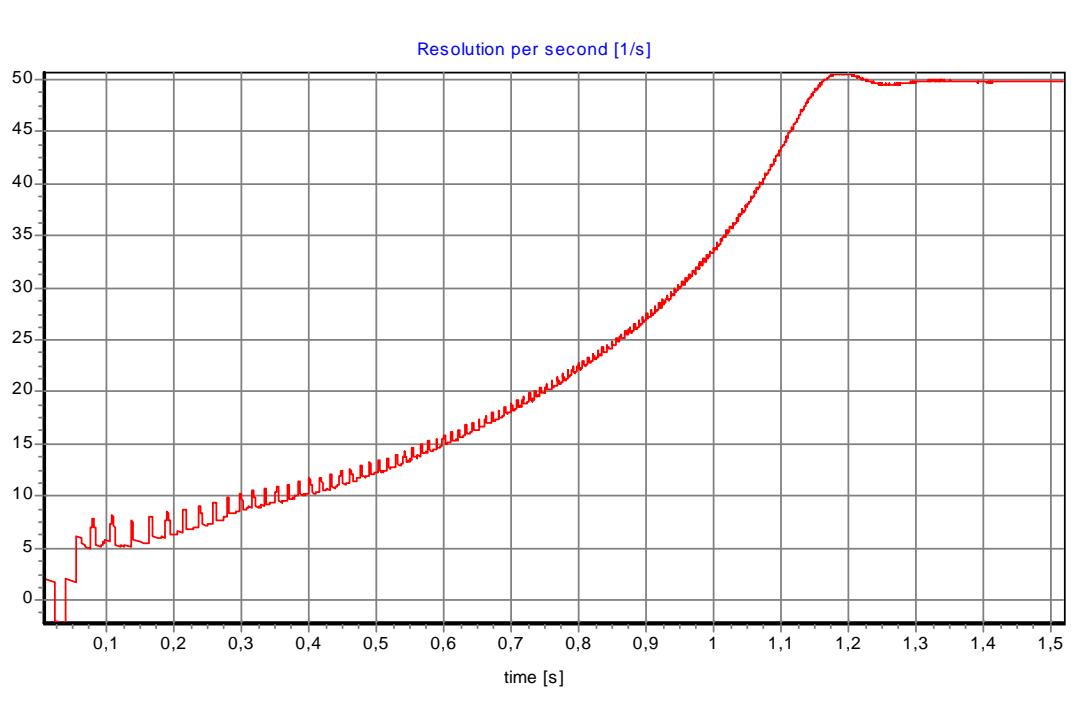


Fig.3.6. Speed  $n$  [rps] v. time  $t$  [s] curve for loaded machine (stator magnetic channels  $Q_s=18$ , rotor magnetic channels  $Q_r=12$ ,  $Y_s=5/6$ ,  $Y_r=1$ )

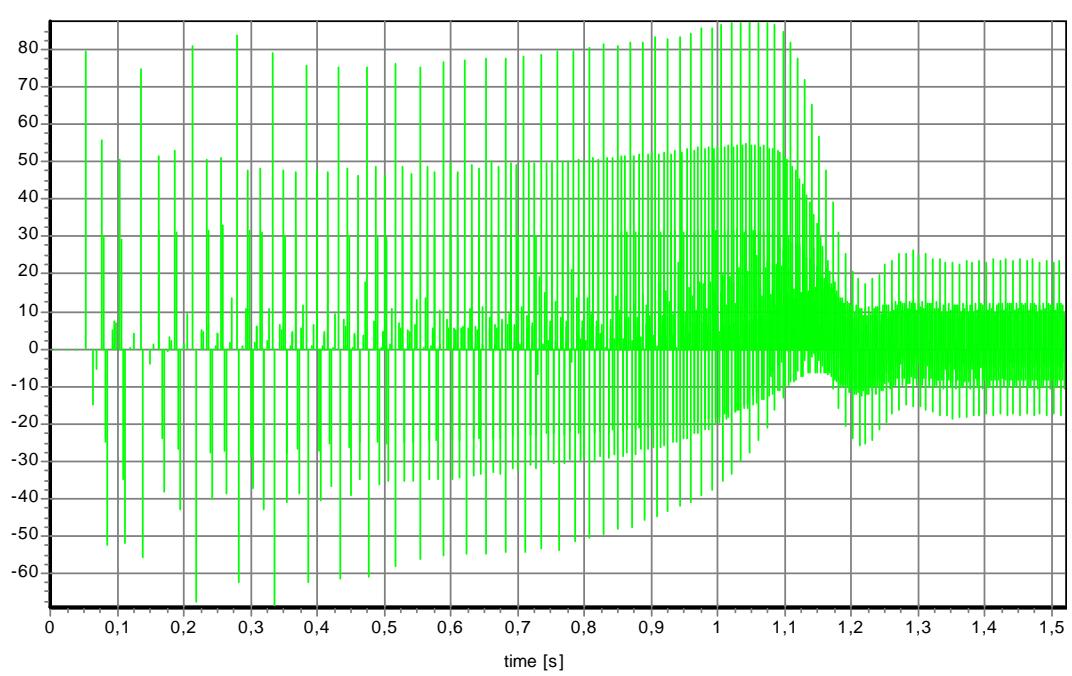


Fig.3.7. Dirac impulses of electromagnetic torque [Nm] v. time  $t$  [s] curve for loaded machine (stator magnetic channels  $Q_s=12$ , rotor magnetic channels  $Q_r=12$ ,  $Y_s=5/6$ ,  $Y_r=1$ )

It is very convenient to introduce the average value of electromagnetic torque for consecutive round angle  $2\pi$  which can be defined in the following way:

$$T_{e_{av}} = \frac{1}{2\pi} (\Delta E_{m,k+1} + \Delta E_{m,k+2} + \dots + \Delta E_{m,k+i} + \dots + \Delta E_{m,k+Q}), \quad (3.16c)$$

where the changes of magnetic field energy are equals to:

$$\Delta E_{m,k} = \frac{1}{2} \Psi_k^T (\mathbf{M}_k^{-1} - \mathbf{M}_{k-1}^{-1}) \Psi_k, \quad (3.16d)$$

or in the shortened form:

$$T_{e_{av}} = \frac{1}{2\pi} \sum_{i=1}^Q \Delta E_{m,k+i}. \quad (3.16e)$$

The above-given definition is very useful and enables us to describe the behaviour of the machine in the manner similar to this known from conventional analysis based on continuous functions. The average value of electromagnetic torque (expressed by Exps 3.16c,d) v. time curve and the average values of electromagnetic torque v. speed trajectory for the exemplary motor are presented in Figs 3.8 and 3.9, respectively.

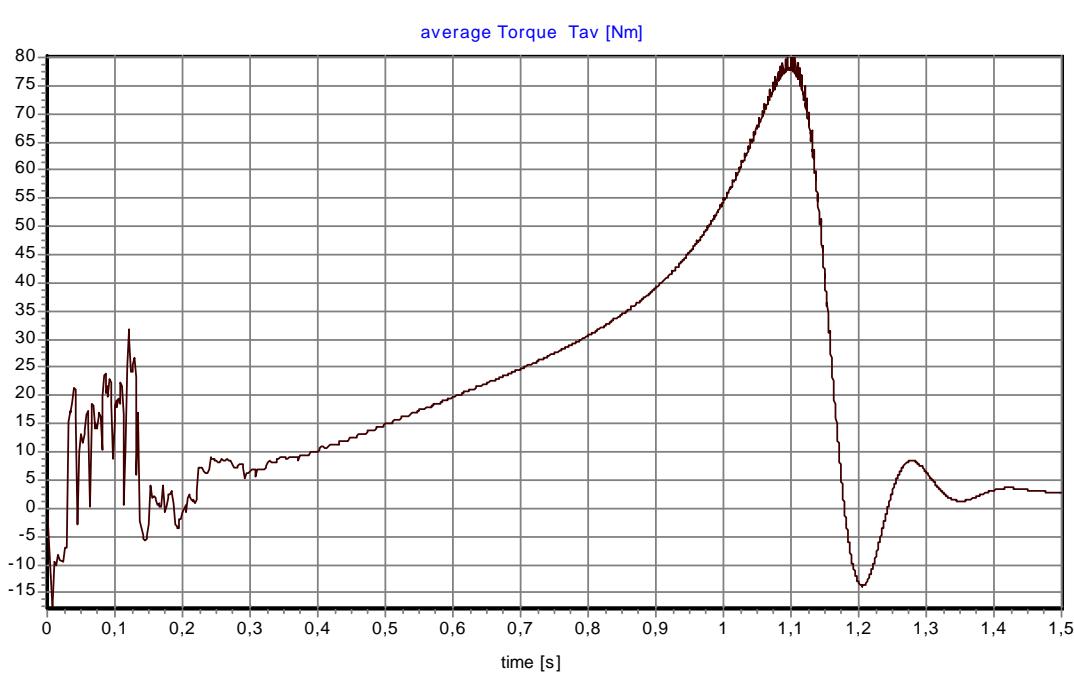


Fig.3.8. Average value of electromagnetic torque  $T_{e_{av}}$  [Nm] v. time  $t$  [s] curve at transient state for motor ( $Q_s = 48$ ,  $Q_r = 36$ ,  $Y_s = 20/24$ ,  $Y_r = 12/18$ )

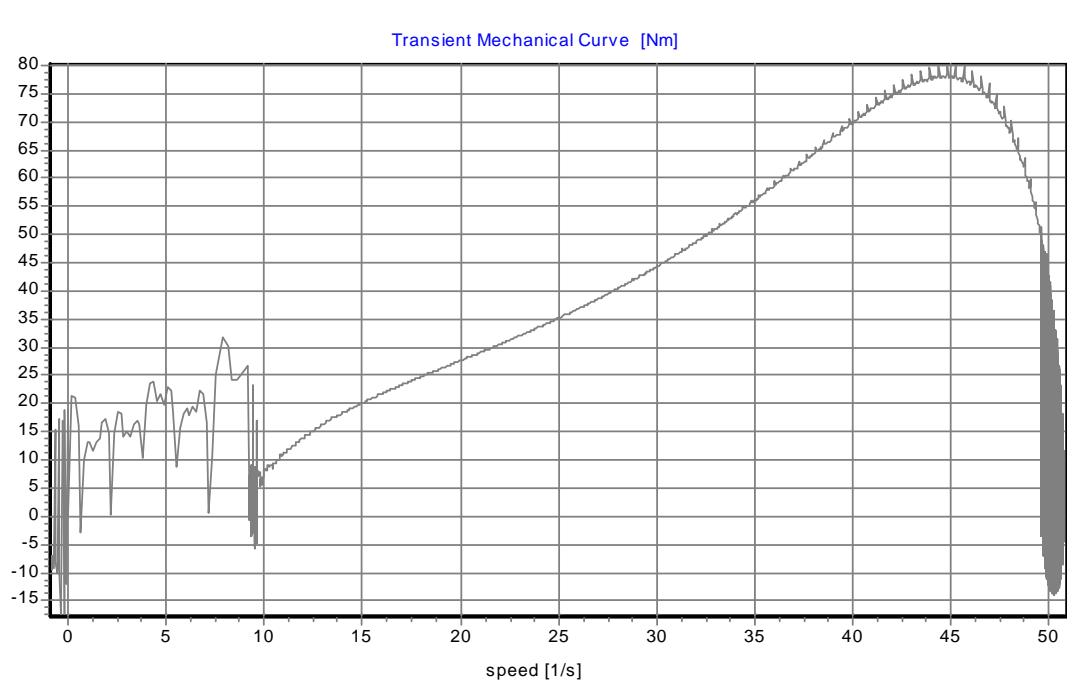


Fig.3.9.Average value of electromagnetic torque  $T_{e_{av}}$  [Nm] v. speed  $n$  [rps]  
trajectory at transient state for motor ( $Q_s=48$ ,  $Q_r =36$ ,  $Y_s=20/24$ ,  $Y_r=12/18$ )

## 4. STEADY-STATE ANALYSIS OF INDUCTION MACHINE

Let us consider the electromagnetic transient state under the condition:

$$\Omega_m = \text{const} . \quad (4.1)$$

The condition (4.1) enables us to determine in advance the length of all consecutive time intervals according to the following expression:

$$\Delta t_k = t_{k+1} - t_k = \frac{\vartheta_{k+1} - \vartheta_k}{\Omega_m} = \frac{\Delta \vartheta_k}{\Omega_m} . \quad (4.2)$$

The recurrent sequence relation (4.2) enables the value of the flux linkage at the end of an arbitrary  $(k+1)$ -th interval to be expressed in relation to the initial values of the flux linkage at the beginning of the first interval  $(t_0, t_1)$ :

$$\Psi_{k+1} = E_{k+1} E_k \cdot \dots \cdot E_2 E_1 \Psi_0 + \sum_{i=1}^{k+1} E_{k+1} E_k \cdot \dots \cdot E_{i+1} F_i . \quad (4.3)$$

If the value of flux linkage repeats after the round angle  $2\pi$  (i.e. after  $Q$  intervals):

$$\Psi_k = \Psi_{k+Q} , \quad (4.4)$$

the electromagnetic transient state will change to steady state.

Theoretically, this condition can be fulfilled only after infinite number of intervals ( $k \rightarrow \infty$ ) but with an assumed accuracy  $\varepsilon$  - sufficient from a technical point of view - one can regard (4.4) as satisfied after  $N(\varepsilon)$  intervals determined by the inequality:

$$\|\Psi_{N+Q} - \Psi_N\| \leq \varepsilon . \quad (4.5a)$$

The consecutive values corresponding to the round angle  $2\pi$  calculated after  $N$  time intervals can be treated as a steady-state solution:

$$\Psi_N, \Psi_{N+1}, \Psi_{N+2}, \dots, \Psi_{N+i}, \dots, \Psi_{N+Q} .$$

According to (4.5a), for each value  $\Psi_{N+i}$  the following approximate equality is satisfied:

$$\Psi_{N+i} - \Psi_{N+i+Q} = 0 . \quad (4.5b)$$

Substituting the values  $\Psi_{N+i}$  and  $\Psi_{N+i+Q}$  determined by the recurrent sequence equation (4.3) into Eqn (4.5b) one obtains:

$$\Psi_{N+i} = -\{\mathbf{H} - \mathbf{1}\}^{-1} \mathbf{H}_i \{\mathbf{H} - \mathbf{1}\} \Psi_0 + \sum_{k=1}^{N+i} \mathbf{E}_{k+1} \mathbf{E}_k \cdot \dots \cdot \mathbf{E}_{N+i} \mathbf{F}_k , \quad (4.6)$$

where  $\mathbf{H} = \mathbf{E}_{N+1} \mathbf{E}_{N+2} \cdot \dots \cdot \mathbf{E}_{N+Q}$ ,  $\mathbf{H}_i = \mathbf{E}_1 \mathbf{E}_2 \cdot \dots \cdot \mathbf{E}_{N+i}$ .

If the initial condition is equal to zero  $\Psi_0 = 0$ , Eqn (4.6) is reduced to the simple formula:

$$\Psi_{N+i} = \sum_{k=1}^{N+i} \mathbf{E}_{k+1} \mathbf{E}_k \cdot \dots \cdot \mathbf{E}_{N+i} \mathbf{F}_k . \quad (4.7)$$

The average value of electromagnetic torque at steady state for the round angle  $2\pi$  is determined by the following sum (see Fig.4.1):

$$T_{e_{av}} = \frac{1}{2\pi} (\Delta E_{m,N+1} + \Delta E_{m,N+2} + \dots + \Delta E_{m,N+i} + \dots + \Delta E_{m,N+Q}) , \quad (4.8)$$

where

$$\Delta E_{m,N+i} = \frac{1}{2} \Psi_{N+i}^T (\mathbf{M}_{N+i}^{-1} - \mathbf{M}_{N+i-1}^{-1}) \Psi_{N+i} . \quad (4.9)$$

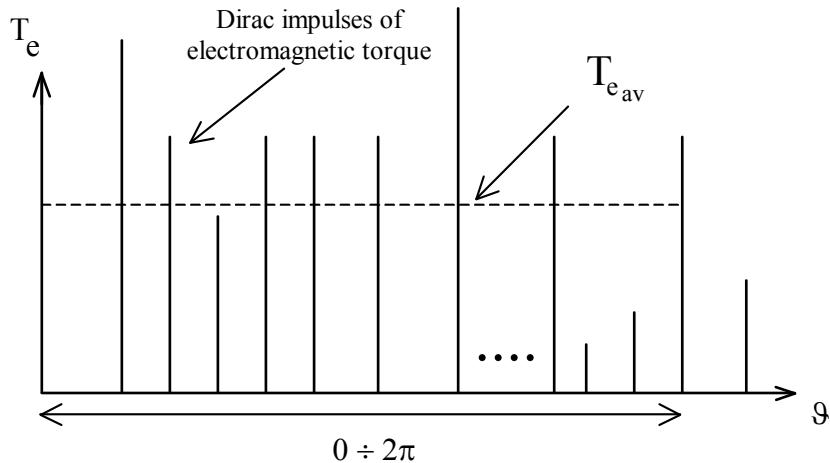


Fig.4.1. Graphical interpretation of average value of electromagnetic torque  $T_{e_{av}}$

Equation (4.8) enables to calculate the average value of the electromagnetic torque  $T_{e_{av}}$  for each arbitrarily chosen speed  $\Omega_m$  at steady state and obtain, finally, the steady state torque-speed curve:  $T_{e_{av}} = f(\Omega_m)$ . Torque-

speed curves for exemplary induction machines - calculated according to the given relations - are presented in Figs. 4.2 and 4.3. They are presented against a background of torque-speed curve calculated according to the standard model taking into account only the fundamental MMF space harmonic and derived under the assumption of the unslotted smooth air-gap (smooth line). In the stable operating region (between the speed corresponding to the maximum torque and synchronous speed) both curves are in very good agreement.

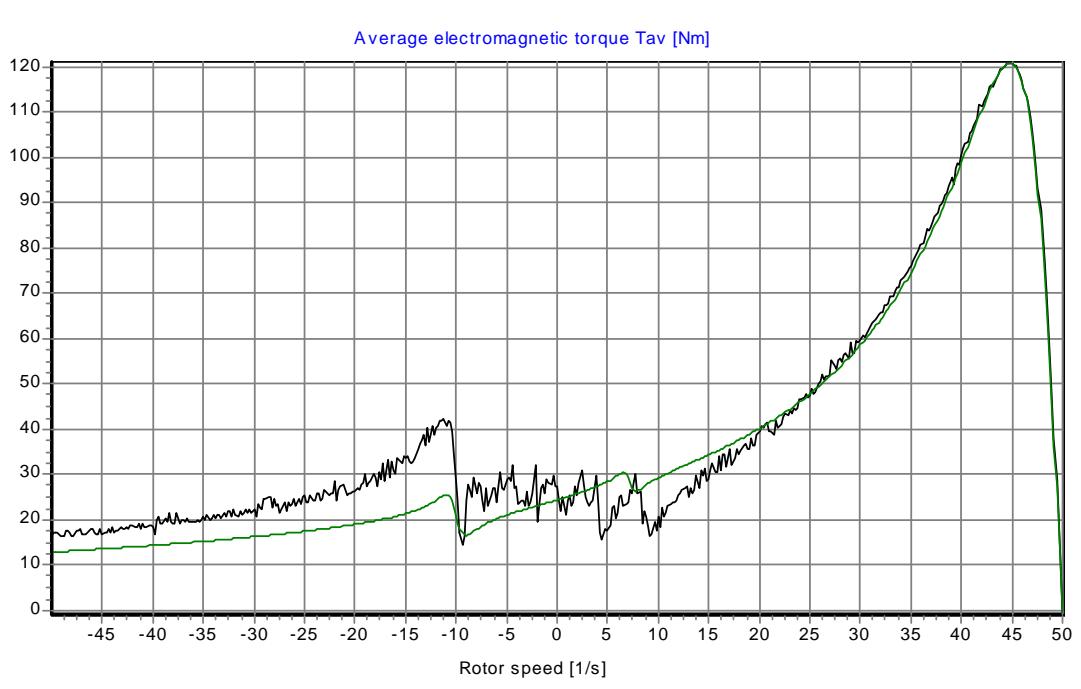


Fig.4.2. Steady-state torque-speed curve ( $T_{e_{av}}$  [Nm] v.  $n$  [rps]) for machine  
( $Q_s=18$ ,  $Q_r=12$ ,  $Y_s=5/6$ ,  $Y_r=1$ )

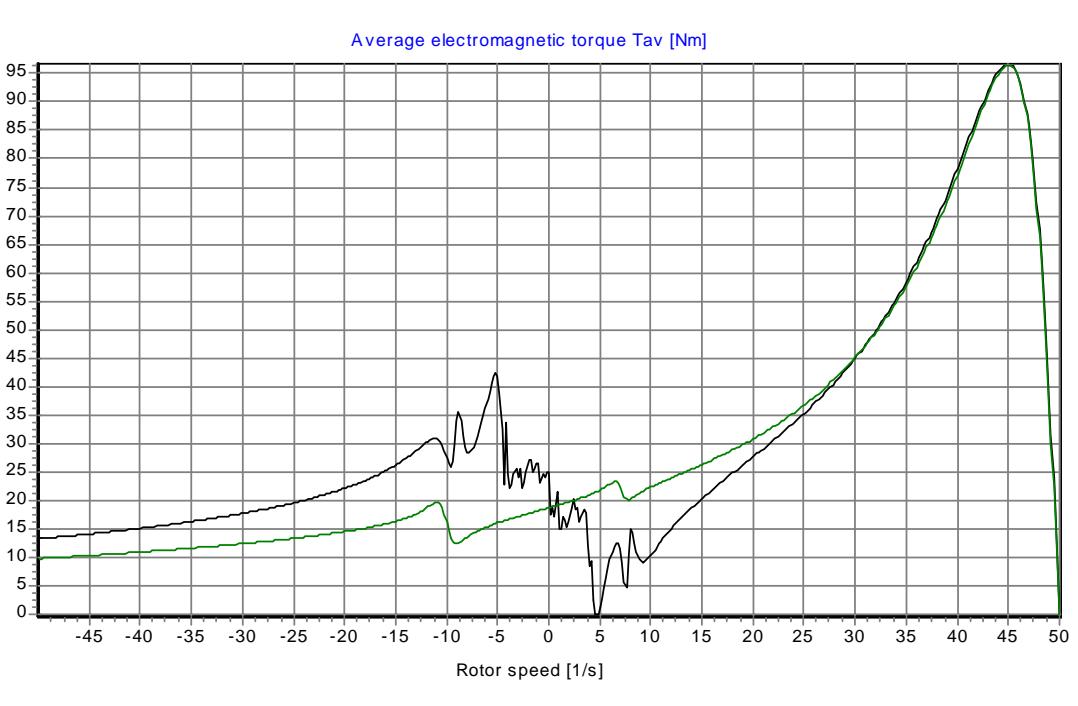


Fig.4.3. Steady-state torque-speed curve ( $T_{e_{av}}$  [Nm] v.  $n$  [rps]) for machine ( $Q_s=48$ ,  $Q_r=36$ ,  $Y_s=20/24$ ,  $Y_r=12/18$ )

It can be regarded as one of the proofs that the proposed method works well because, as known, in this region parasitic torques resulting from slotting and higher MMF space harmonics do not distort the torque-speed curve related to the fundamental space harmonic. The ability of the model for describing parasitic reluctance torques resulting from slotting has been tested experimentally and the results of this verification are presented in the next chapter.



## 5. ABILITY OF MATHEMATICAL MODEL FOR DESCRIBING PARASITIC RELUCTANCE TORQUE RESULTING FROM SLOTTING

The correctness of the proposed method for describing reluctance torque resulting from two-sided slotting was verified experimentally with the help of the especially-manufactured 3-phase induction squirrel-cage motor (number of pole-pairs  $p = 2$ , number of stator slots  $Q_s = 36$ , number of rotor slots  $Q_r = 28$ , typical two-layer stator winding) in which the rotor was not completed with cast aluminium cage and remained with non-filled slots (i.e. without winding).

In such a motor the only reasons for developed electromagnetic torque are:

- irregularity of the air-gap i.e. two-sided slotting which leads to generating parasitic torque having nature of pulsating torque and changing into constant reluctance torque at certain numbers of speeds which are termed synchronous speeds,
- interaction between stator currents and eddy-currents induced in the iron mass of a rotor.

In order to increase the value of reluctance torque the rotor was manufactured with non-skewed slots (but with normal openings).

The problem of producing reluctance torque associated with slotting has been considered in details in [42] as well as the procedure enabling us to determine synchronous speeds for reluctance torque as the function of slot numbers and supply frequency is described. The above-mentioned procedure is based on Fourier analysis and on developing MMF pattern of windings  $\Theta(\alpha)$  and space distribution of air-gap permeance  $\Lambda$  in Fourier series. Denoting the orders of MMF space harmonics by  $v_\Theta$ , one can find all the synchronous speeds related to the different combination of MMF and permeance space harmonics - in the motor with wound stator and non-wound rotor - according to the following expression:

$$\Omega_{\text{syn}} = \frac{k_r Q_r \pm k_s Q_s}{k_r Q_r} \frac{2\pi f_s}{v_\Theta}, \quad (5.1)$$

where supplementary coefficients  $k_s$  and  $k_r$  are integer numbers ( $k_s, k_r = 1, 2, 3, 4\dots$ ) determining orders of working permeance space harmonics and  $f_s$  means supply frequency.

The majority of arising reluctance torque have negligible value of magnitudes. As regards predominant torque they are connected with MMF space harmonics of orders:

$$p, 5p, 7p, \dots$$

and permeance space harmonics of the orders:

$$|k_s Q_s \pm k_r Q_r|$$

(the most significant harmonic is of order  $Q_s - Q_r$ ).

In the investigated machine ( $f_s=50$  Hz,  $p=2$ ,  $n_s=2\pi f_s/p = 1500$  rpm) the predominant torque are produced by MMF space harmonics of the orders:  $v_\Theta = p = 2$  and  $v_\Theta = 5p = 10$  - and permeance space harmonics corresponding to the following values of supplementary coefficients:  $k_s = 1$ ,  $k_r = 3$ .

As a consequence we obtain two values of synchronous speeds:

$$\Omega_{\text{syn}}^I = \frac{3 \cdot 28 + 1 \cdot 36}{3 \cdot 28} \frac{2\pi f_s}{10} = \frac{1}{7} 2\pi f_s \quad \Rightarrow \quad n_{\text{syn}}^I = \frac{1}{7} n_s \cong 214,3 \text{ rpm},$$

$$\Omega_{\text{syn}}^{II} = \frac{3 \cdot 28 - 1 \cdot 36}{3 \cdot 28} \frac{2\pi f_s}{2} = \frac{2}{7} 2\pi f_s \quad \Rightarrow \quad n_{\text{syn}}^{II} = \frac{2}{7} n_s \cong 428,6 \text{ rpm}.$$

The theoretical results were verified experimentally [42]. When the voltage was applied to the terminals of the investigated motor (manufactured without rotor cage), its speed began to increase slowly but gradually as a result of acting induction torque developed by stator currents and rotor eddy-currents. The motor was accelerating just to the first synchronous speed equal to:  $n_{\text{syn}}^I = (1/7) \cdot n_s$  and was pulled into synchronism.

Under the no-load condition the motor was remaining in equilibrium at the constant speed  $n_{\text{syn}}^I = (1/7) \cdot n_s \cong 214,3$  rpm.

Then, continuing the experiment its operation was disturbed with the help of hand in such a way that it caused a small increment of speed. It was a reason for further

gradual acceleration of the rotor. The motor was speeding up and stopped after reaching the velocity equal to the second synchronous speed  $n_{\text{syn}}^{\text{II}} = (2/7) \cdot n_s \cong 428,6 \text{ rpm}$ .

The above-described experiment confirms clearly that in the investigated motor the two-sided slotting is responsible for producing two components of reluctance torque: the first pulling the rotor in synchronism at the speed equalled to:

$$n_{\text{syn}}^{\text{I}} = (1/7) \cdot n_s \cong 214,3 \text{ rpm},$$

and the second - pulling the rotor in synchronism at the speed equalled to:

$$n_{\text{syn}}^{\text{II}} = (2/7) \cdot n_s \cong 428,6 \text{ rpm}.$$

It is necessary to underline that in both cases the values of reluctance torque was small and only slightly greater than the mechanical losses. The reason for that are small values of stator and rotor slot openings (commonly-applied in typical squirrel-cage motors).

The above experimentally-tested phenomena was modelled with the help of the transient-state mathematical model of an induction machine with discrete space distribution of air-gap permeance along the periphery (described in chapter 3.1). The elaborated physical model has 36 stator magnetic channels ( $Q_s=36$ ) and 28 rotor magnetic channels ( $Q_r=28$ ). It was assumed that the rotor resistance is equal to the equivalent iron-loss resistance resulting from presence and action of rotor eddy currents. This equivalent iron-loss resistance in the rotor ensures asynchronous starting of the considered motor which does not posses the cage winding. The results of computer simulation which follow the results obtained in the described laboratory experiment are presented in Fig.5.1.

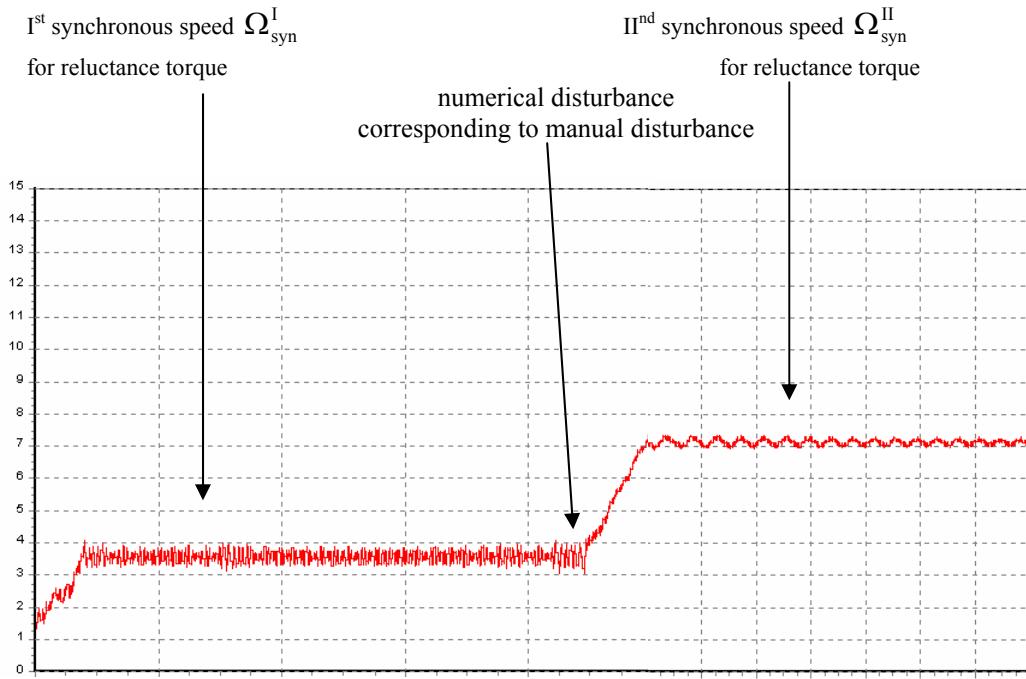


Fig.5.1. Speed v. time for the considered motor (result of computer simulation)

The next interesting computer simulations are carried out with the help of transient-state mathematical model of the machine at the constant given speed. In this numerical experiment the rotor speed was fixed at the five different values: equalled to the synchronous speed  $\Omega_{\text{syn}}^I$ , slightly greater than the synchronous speeds  $\Omega_{\text{syn}}^I$  (two cases) and slightly less than the synchronous speed  $\Omega_{\text{syn}}^I$  (two cases).

The above described experiments were carried out for the both reluctance torque i.e. related to the speeds  $\Omega_{\text{syn}}^I$  (left-hand column in the Table I) and  $\Omega_{\text{syn}}^{II}$  (right-hand column in the Table I).

Table I. Given constant rotor speeds for numerical experiments at transient-states

	I <sup>st</sup> reluctance torque related to $\Omega_{\text{syn}}^I$	II <sup>nd</sup> reluctance torque related to $\Omega_{\text{syn}}^{II}$
Synchronous speed	$n_{\text{syn}}^I = (1/7) \cdot n_s \cong 214,3 \text{ rpm}$ (Fig.5.2)	$n_{\text{syn}}^{II} = (2/7) \cdot n_s \cong 428,6 \text{ rpm}$ (Fig.5.7)
Greater speeds	$n_{m+}^I = n_{\text{syn}}^I \cdot (1+0,0024)$ (Fig.5.3) $n_{m+}^I = n_{\text{syn}}^I \cdot (1+0,0136)$ (Fig.5.4)	$n_{m+}^{II} = n_{\text{syn}}^{II} \cdot (1+0,005)$ (Fig.5.8) $n_{m+}^{II} = n_{\text{syn}}^{II} \cdot (1+0,01)$ (Fig.5.9)
Lower speeds	$n_{m-}^I = n_{\text{syn}}^I \cdot (1-0,0024)$ (Fig.5.5) $n_{m-}^I = n_{\text{syn}}^I \cdot (1-0,0134)$ (Fig.5.6)	$n_{m-}^{II} = n_{\text{syn}}^{II} \cdot (1-0,005)$ (Fig.5.10) $n_{m-}^{II} = n_{\text{syn}}^{II} \cdot (1-0,01)$ (Fig.5.11)

At the velocity equal to the first synchronous speed  $n_{\text{syn}}^I = (1/7) \cdot n_s \cong 214,3 \text{ rpm}$  (Fig.5.2) the resultant constant torque is the sum of reluctance torque resulting from slotting and the asynchronous torque  $T_{e_{\text{asyn}}}$  connected with rotor eddy currents.

It is necessary to emphasize that the value of resultant constant torque in Fig. 5.2 depends on given constant rotor angle.

At the speeds greater (Figs 5.3 and 5.4) and less (Figs 5.5 and 5.6) than the synchronous speed  $n_{\text{syn}}^I$  the torque is the sum of pulsating torque (whose period depends on the number of stator slots and difference between the synchronous speed and the rotor speed:  $n_{\text{syn}}^I - n_m$ ) and asynchronous torque  $T_{e_{\text{asyn}}}$ . The analogous results are achieved for the second synchronous speed  $n_{\text{syn}}^{II} = 428,6 \text{ rpm}$  (Figs 5.7, 5.8 and 5.9, 5.10 and 5.11). The average value of torque calculated due to Figs 5.3 - 5.6 and Figs 5.8 - 5.10 are equal to:

$$T_{e_{\text{av}}} = 9,5 \text{ Nm} \quad \text{for the speed } n_{\text{syn}}^I = (2/7) \cdot n_s \cong 214,3 \text{ rpm, and}$$

$$T_{e_{\text{av}}} = 7,6 \text{ Nm} \quad \text{for the speed } n_{\text{syn}}^{II} = (2/7) \cdot n_s \cong 428,6 \text{ rpm.}$$

Henceforth, the values of reluctance torque can be easily determined according to these results. The difference between the steady-state value of total electromagnetic

torque  $T_{e_{av}}$  (read off from Fig.5.2 and Fig.5.7) and asynchronous torque  $T_{e_{asyn}}$  (evaluated with the help of Figs 5.3 - 5.6 and 5.8 - 5.11 as average values) equals to reluctance torque.

Namely, for the speed  $n_{syn}^I = (1/7) \cdot n_s \cong 214,3$  rpm the first reluctance torque equals:

$$70,0 \text{ Nm} - 9,5 \text{ Nm} = \mathbf{60,5 \text{ Nm}},$$

and for the speed  $n_{syn}^{II} = (2/7) \cdot n_s \cong 428,6$  rpm the synchronous torque equals:

$$13,1 \text{ Nm} - 7,6 \text{ Nm} = \mathbf{5,5 \text{ Nm}},$$

respectively.

Changing gradually the given constant rotor angle (what causes the change of average value of resultant constant torque in Fig.5.2 and Fig. 5.7), and repeating above-presented computer simulations one can easily determine reluctance torque – rotor angle curve for both synchronous speeds. They are shown in Fig.5.12 and Fig.5.13, respectively. The figures enable to find the maximum values of both reluctance torque:

$$T_{e_{synch,max}}^I = 74,4 \text{ Nm} \quad \text{for the speed} \quad n_{syn}^I = (1/7) \cdot n_s \cong 214,3 \text{ rpm},$$

$$T_{e_{synch,max}}^{II} = 14,1 \text{ Nm} \quad \text{for the speed} \quad n_{syn}^{II} = (2/7) \cdot n_s \cong 428,6 \text{ rpm}.$$

It results from Figs 5.12 and 5.13 that the period of the synchronous torque v. rotor angle curves is equal for  $n_{syn}^I$  rotor slot pitch:

$$\frac{2\pi}{Q_r},$$

and for  $n_{syn}^{II}$  equals double rotor slot pitch:

$$2 \frac{2\pi}{Q_r}.$$

It is worth reminding that the reluctance torque – rotor angle curve is calculated by analogy to the synchronous torque – load angle curve which is well-known in the theory of synchronous machine (in the considered case rotor angle corresponds to load angle)

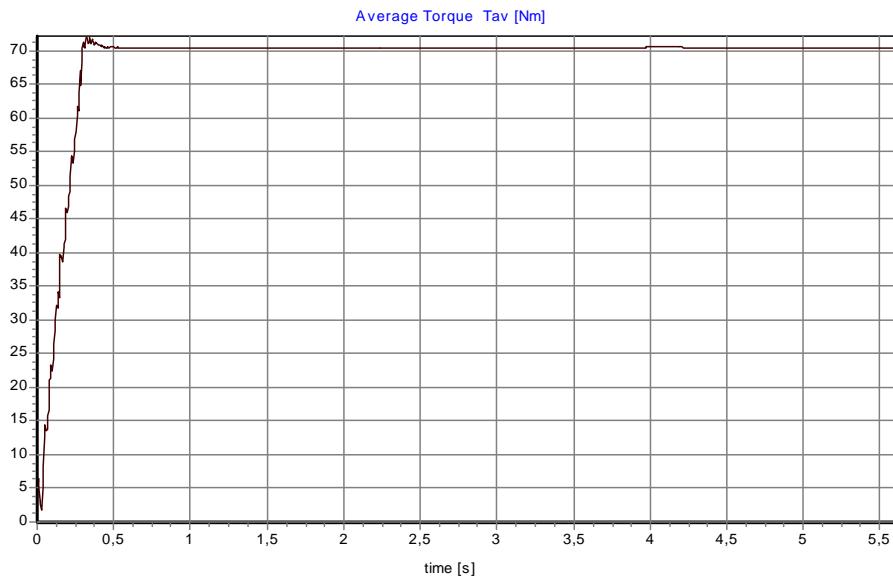


Fig.5.2. Torque v. time curve at the fixed speed  $n_{\text{syn}}^{\text{I}} = (1/7) \cdot n_1$   
 $= 214,2857143 \text{ rpm} = 3,571435 \text{ rps}$

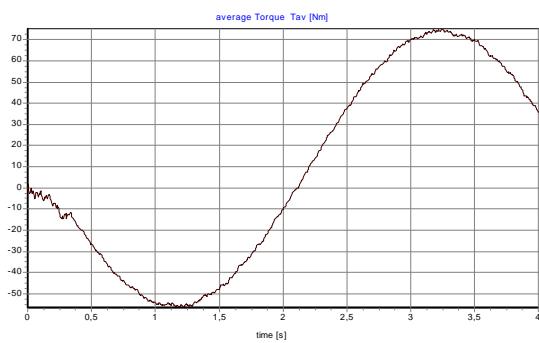


Fig.5.3. Torque v. time curve at the speed  $n_{m+}^{\text{I}} = 3,58 \text{ rps}$

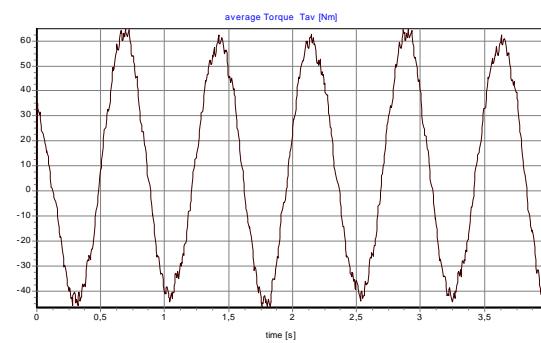


Fig.5.4. Torque v. time curve at the speed  $n_{m+}^{\text{I}} = 3,62 \text{ rps}$

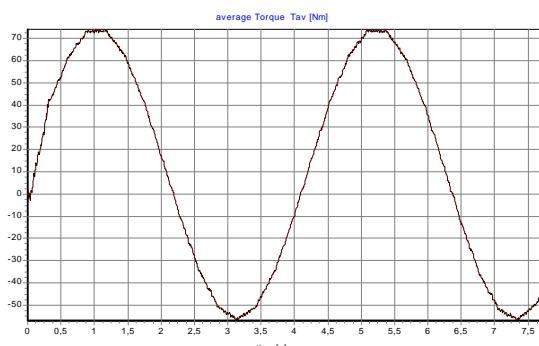


Fig.5.5. Torque v. time curve at the speed  $n_{m-}^{\text{I}} = 3,56 \text{ rps}$

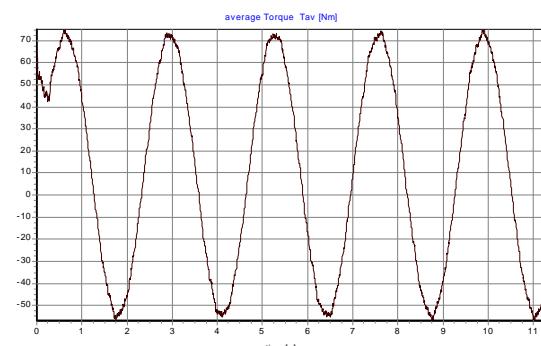


Fig.5.6. Torque v. time curve at the speed  $n_{m-}^{\text{I}} = 3,55 \text{ rps}$

## Experimental verification

---

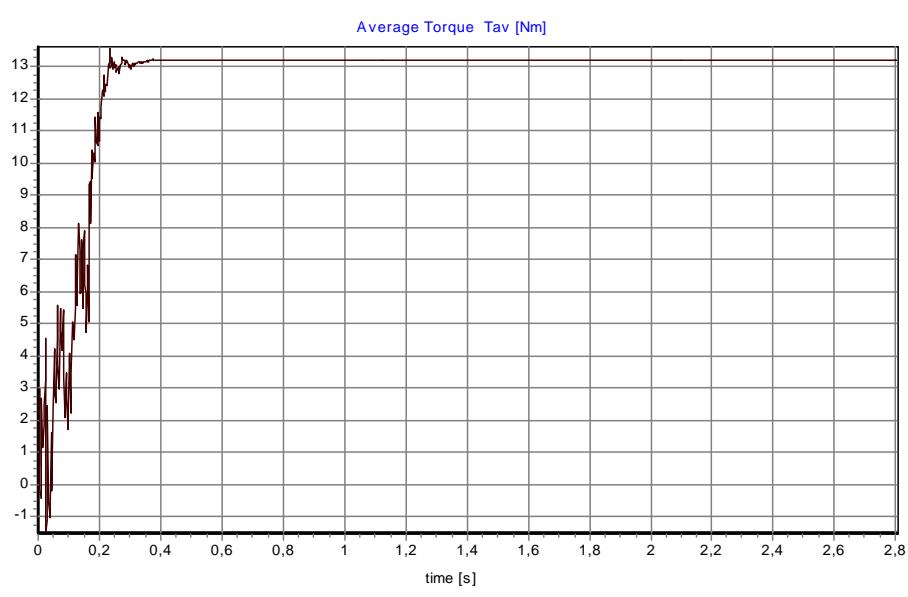


Fig.5.7. Torque v. time curve at the fixed speed  $n_{\text{syn}}^{\text{II}} = (2/7) \cdot n_1$   
 $= 428,5714286 \text{ rpm} = 7,1428571 \text{ rps}$

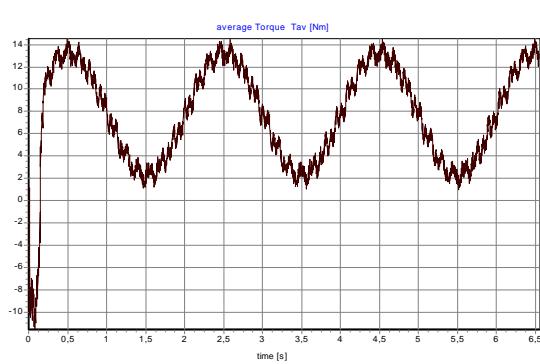


Fig.5.8.Torque v. time curve at the speed  $n_{m+}^{\text{II}} = 7,18 \text{ rps}$

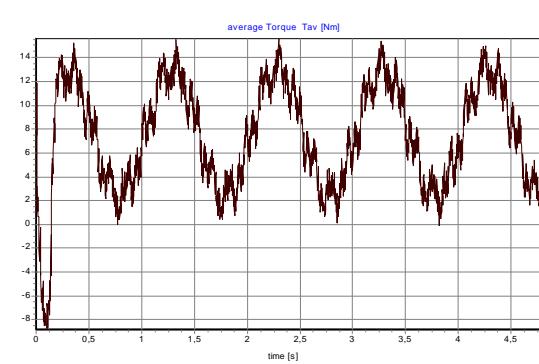


Fig.5.9.Torque v. time curve at the speed  $n_{m+}^{\text{II}} = 7,21 \text{ rps}$

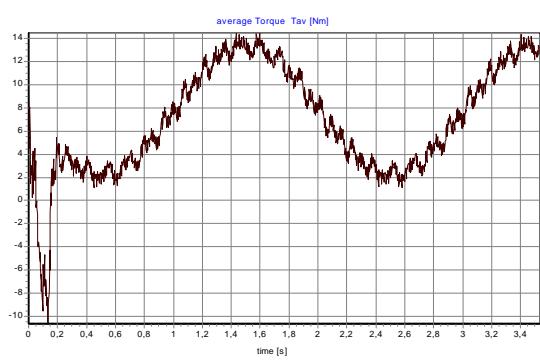


Fig 5.10. Torque v. time curve at the speed  $n_{m-}^{\text{II}} = 7,11 \text{ rps}$

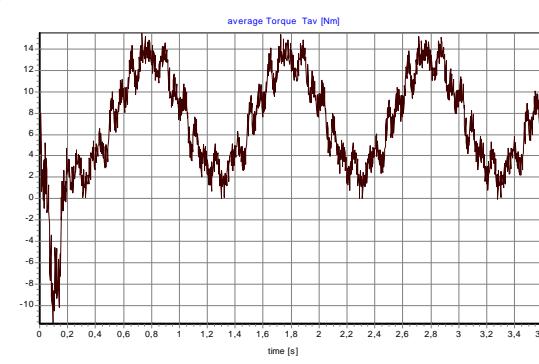


Fig 5.11. Torque v. time curve at the speed  $n_{m-}^{\text{II}} = 7,07 \text{ rps}$

the point corresponding to Figs. 5.2 ÷ 5.6

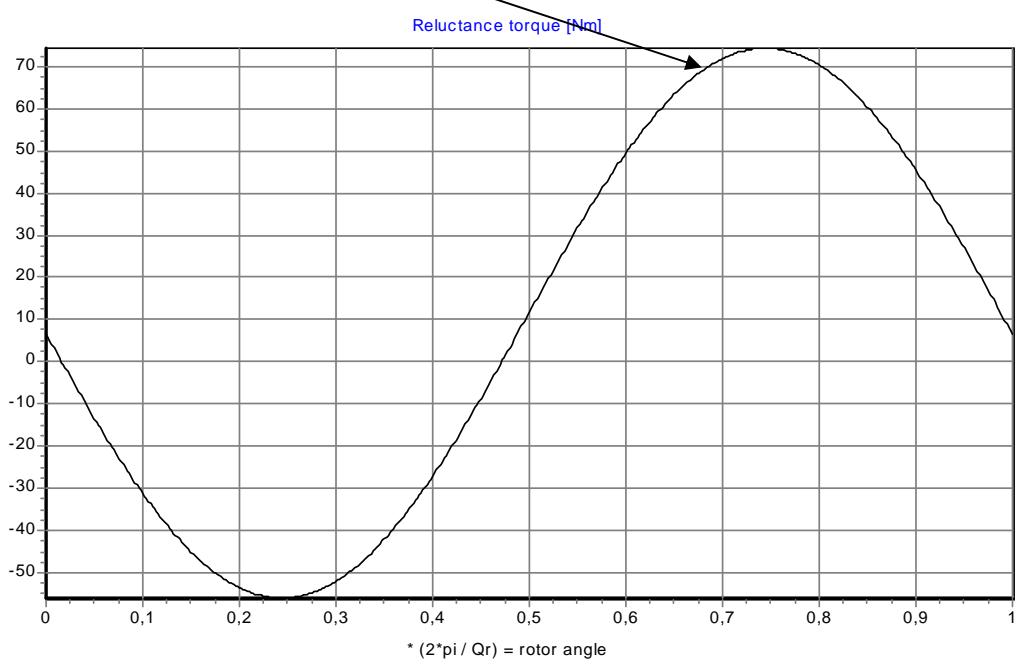


Fig.5.12. Reluctance torque v. rotor angle at the given constant speed  $n_{ms}^I = (1/7) \cdot n_1$   
 $= 214,2857143$  rpm

the point corresponding to Figs. 5.7 ÷ 5.11

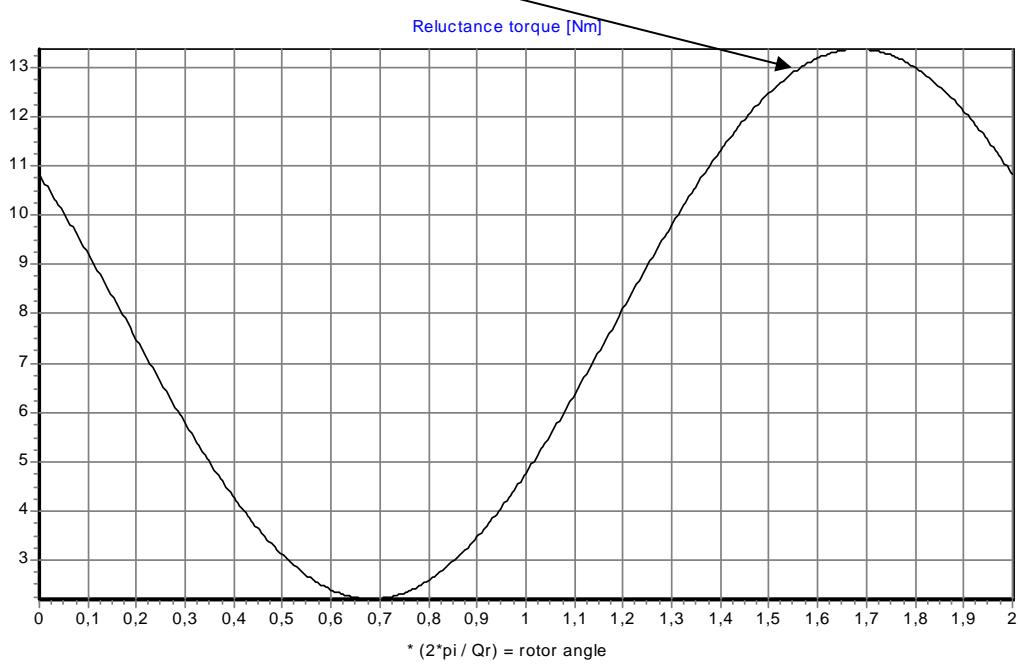


Fig.5.13. Reluctance torque v. rotor angle at the given constant speed  $n_{ms}^{II} = (2/7) \cdot n_1$   
 $= 428,5714286$  rpm

## CONCLUSIONS

The results of computer simulations and their comparison with the results of laboratory experiments enable to state that the proposed non-harmonic mathematical model and related to it **step-by-step analysis** takes into account effects resulting from slotting and allows to consider the influence of the number of stator and rotor slots on the dynamic behaviour of the machine as well as on steady-state torque-speed curves.

The model is helpful in determining basic properties of reluctance torque components and particularly enables to determine their synchronous speeds, approximate magnitudes, periods of reluctance torque v. rotor angle curves as well as magnitudes of pulsating torque at the given speeds. The information about parasitic torque resulting from slotting is hidden in the number of steps, as well as height and length of steps of the stator-rotor inductance curves.

The approximation of stator-rotor inductances based on piecewise-constant functions simplifies the mathematical model of an electrical machine which consists of the system of linear differential equations with time-constant coefficients and some algebraic equations. This model is convenient in computer calculations for both transient and steady states.

## REFERENCES

1. Alwash I., Ikhwan S.: *Generalized approach of the analysis of asymmetrical three-phase induction motors.* Proc. IEE, Power Appl. Vol.142, No2, 1995.
2. Antosik P., Mikusiński J., Sikorski R.: *Theory of distributions. The sequential approach.* Amsterdam. Elsevier Scientific Publishing Company, 1973.
3. Barton T.H., Dunfield J.C.: *Polyphase to the axis transformation for real windings.* Proc. IEE, Vol. 87, No 5, 1968, pp. 1342-1346.
4. Barton T.H., Poloujadoff M.A.: *A generalized symmetrical components transformation for cylindrical electrical machines.* IEEE PAS 91, 1972, pp.1781-1986.
5. Boldea I., Nasar S.A.: *Electric machine dynamic.* Macmillan Publishing Company, 1986.
6. Brown J.E., Jha C.S.: *Generalized rotating field theory of polyphase induction motors and its relationship to the symmetrical components theory.* IEEE Proc.109, 1962, pp.59-69.
7. Consoli A., Lipo T.A.: *Orthogonal axis models for asymmetrically connected induction machines.* IEEE Trans, Vol.PAS-101, 1982, pp.4518-4526.
8. Drozdowski P., Sobczyk T.J.: *On mathematical model of squirrel-cage induction motors.* Archiwum Elektrotechniki 70, 1987, pp.371-382.
9. Drozdowski P.: *Model matematyczny silnika klatkowego uwzględniający wyższe harmoniczne, sprowadzalny do równań różniczkowych o stałych współczynnikach (Mathematical model of a squirrel cage motor accounting for higher harmonics reducible to the differential equations with constant coefficients – in Polish).* Elektrotechnika, Kwartalnik AGH. Tom3, z.1. Kraków 1984, ss.15-35.
10. Drozdowski P.: *Analiza pracy silnika asynchronicznego klatkowego jako autotransformatora (The analysis of a squirrel cage induction as an autotransformer - in Polish).* Doctoral Thesis, AGH, Kraków 1984.
11. Fudech H.R., Ong C.M.: *Modelling and analysis of induction machines containing space harmonics.* Part I, II, III. IEEE PAS 102, 1983, pp.1608-1628.
12. Gibbs W.J.: *Tensors in Electrical Machine Theory.* London, Chapman and Hall Ltd, 1952.
13. Heller V., Hamata V.: *Harmonic field effects in induction machines.* Amsterdam, Elsevier 1977.
14. Hommes E., Paap G.C.: *The analysis of the 3-phase squirrel-cage induction motor with space harmonics: Part1. Equations developed by new time-dependent transformation. Part2. The influence of space harmonics on the transient behaviour.* Archiv für Elektrotechnik, No 4, 1984, pp.217-236.

15. Jha C.S., Sreenivasa Murthy S.: *Generalized rotating-field theory of wound-rotor induction machines having asymmetry in stator and/or rotor windings.* Proc. IEE 120, 1973, pp.867-873.
16. Kluszczyński K.: *Graphical method of choice of the number of slots in asynchronous machine.* Proc. of Int. Conference on Electrical Machines ICEM'88, Piza 1988.
17. Kluszczyński K., Miksiewicz R.: *Squirrel-cage motor with additional ring in rotor.* Electric Machines and Power Systems 21, 1993.
18. Kluszczyński K., Johr M.: *Simplified representation of doubly-slotted varying air-gap of electrical machine.* Proc. of Int. Conference on Electrical Machines ICEM'96, Vigo 1996, pp. 187-192.
19. Kluszczyński K., Kurzyński M.: *Walsh functions - will they find an application in the analysis and measurements of magnetic field of electrical machines ?* Proc. of XXXII Int. Symposium on Electrical Machines. Cracow, Poland, 1996.
20. Kluszczyński K., Miksiewicz R.: *Momenty pasożytnicze w indukcyjnych silnikach klatkowych (Parasitic torques in squirrel-cage motors – in Polish).* PTETiS, Monograph of Polish Academy of Sciences, Section of Electrical Machines and Transformers, Warsaw-Gliwice, 1993.
21. Kluszczyński K., Miksiewicz R.: *Synchronous parasitic torques in asymmetrically fed three-phase squirrel-cage motor.* Electric Machines and Power Systems, 24, 1996.
22. Kluszczyński K., Spałek D.: *Nowa koncepcja analizy pracy maszyny elektrycznej (New conception of analysis of electrical machines – in Polish).* XIV Seminar on Fundamentals of Electrotechnics and Circuit Theory SPETO, Proc. Vol.II, 1991, pp.279-286.
23. Kluszczyński K., Spałek D.: *Non-harmonic analysis of asynchronous machine based on model with discrete space distribution of magnetic permeance along periphery of air-gap.* Proc. of Int. Conference on Electrical Machines ICEM'94, Vol. II Paris, France, September 1994, pp.585-590.
24. Kluszczyński K.: *Schematy blokowe dla analizy maszyn elektrycznych opartej na funkcjach odcinkami stałych. (Block diagrams for analysis of electrical machine based on piecewise-constant functions – in Polish)* XVII Seminar on Fundamentals of Electrotechnics and Circuit Theory SPETO, Proc. Vol.II, 1994, pp.107-112.
25. Kluszczyński K., Spałek D.: *Non-harmonic analysis of electrical machine based on model with discrete space distribution of magnetic permeance along periphery of air-gap.* The Digests of 4-th Japanese-Polish Joint Seminar on Electromagnetic Phenomena Applied to Technology, Oita, Japan, June 1995, pp.42-45.
26. Kluszczyński K., Spałek D.: *Koncepcja nieharmonicznej analizy maszyn asynchronicznych (A conception of nonharmonic analysis of electrical machines – in Polish).* Przegląd Elektrotechniczny Vol.71, No 5, 1995, pp.113-120.
27. Kluszczyński K., Spałek D.: *Nowa koncepcja analizy pracy maszyny asynchronicznej (New conception of analysis of asynchronous machine – in Polish).* Archiwum Elektrotechniki. Tom XLIII, Vol.3, 1994, pp. 441-453.
28. Kluszczyński K., Spałek D.: *Non-harmonic analysis of an asynchronous machine based on a model with discrete space distribution of magnetic permeance along the periphery of the air-gap.* Archives of Electrical Engineering, Vol. No 1-2, pp.31-49, 1999.

29. Kluszczyński K., Spałek D.: *Computer sequence analysis of asynchronous machine. Part I – Mathematical model of asynchronous machine with discrete air-gap permeance.* Proceedings of Conference ZKwE 2002.
30. Kluszczyński K., Spałek D.: *Computer sequence analysis of asynchronous machine. Part II – Solutions and results of computer analysis.* Proceedings of Conference ZKwE 2002.
31. Kluszczyński K., Spałek D.: *Computer program for step-by-step analysis of asynchronous machine. XXVI Seminar on Fundamentals of Electrotechnics and Circuit Theory IC-SPETO,* Gliwice-Niedzica Proc. Vol. II, pp.505-508, 2003.
32. Kluszczyński K., Spałek D.: *Dyskretna analiza maszyn indukcyjnych wynikającej z żłobkowania (Discrete analysis of induction machine adequate to its slotting).* XXXIX Międzynarodowe Sympozjum Maszyn Elektrycznych'03 s. 23, Gdańsk-Jurata 2003.
33. Kluszczyński K., Spałek D.: *Unconventional mathematical approach to analysis of mechatronics components.* 4<sup>th</sup> International Workshop on Mechatronics, Bochum, 2003
34. Krause P. C.: *Analysis of electric machinery.* McGraw-Hill Book Company, 1986.
35. Kron G.: *The application of tensors to the analysis of rotating electrical machinery.* General electric review, 1942 (as a book).
36. Luo X., Liao Y., Toliyat H., El-Antably A., Lipo T.A.: *Multiple coupled circuit modelling of induction machines.* USA, 1995 (Manuscript by courtesy of prof. T.A. Lipo).
37. Nasar S.A.: *Electromagnetic energy conversion in n,m-winding double cylindrical structures in the presence of space harmonics.* IEEE Trans. PAS 87, 1968, pp.1099-1106.
38. Oberretl K.: *The harmonic field theory of the induction motor.* Archiv für Elektrotechnik, Vol.49, 1965.
39. Oberretl K.: *Field-harmonic theory of slip-ring motor taking multiple armature reaction into account.* Proc. IEE, No 7, 1970, pp.1667-1671.
40. Oberretl K.: *Parasitäre synchrone Dreh- und Pendelmomente in Asynchronmotoren, Einfluss von Ausgleichsvorgängen und Eisensättigung.* Part I, II. Archiv für Elektrotechnik, 1994, Vol.77.
41. Rausch H.: *Ermittlung des Verhaltens von Induktionsmaschinen unter Analytischer Formulierung des Luftspaltfeldes ohne wellenmäßige Darstellung.* Doctoral Thesis, Kaiserlautern 1980.
42. Sobczyk T.J., Sobczyk K.: *On searching for a transformation matrix to equations of AC machines.* Proc. of Int. Conference on Electrical Machines ICEM'88, Pisa (Italy), pp.183-186.
43. Sobczyk T.J., Weinreb K.: *Analysis of currents of electromagnetic torque in steady-state of induction squirrel-cage motor with asymmetric windings.* Archiv für Elektrotechnik, 71, pp.245-256.
44. Stepina J.: *Matrix calculation of inductances for the general theory of electrical machines.* Electric Machines and Power Systems, 11, 1986.
45. Stepina J.: *Non-transformational matrix analysis of electrical machinery.* Electric Machines and Electromechanics, Vol.4, 1979, pp.255-268.
46. Stepina J.: *Matrix analysis of space harmonics of asymmetrical stator windings.* Proc. IEE Vol.134, No 4, 1987.
47. Szymański D.: *Użłobkowanie stojana i wirnika maszyny elektrycznej jako przyczyna: odkształcenia pola magnetycznego w szczelinie powietrznej oraz*

- generowania dodatkowych momentów elektromagnetycznych (*Stator and rotor slotting of electrical machine as the reason for distortion of magnetic field in the air-gap and additional electromagnetic torques*) Doctoral Thesis, Silesian University of Technology, Gliwice, 2001.
- 48. Taegen F., Hommes E.: *Das allgemeine Gleichungssystem des Käufigläufermotors unter Berücksichtigung der Oberfelder*. Archiv für Elektrotechnik, 52, 1972, pp.98-105.
  - 49. Taegen F., Hommes E.: *Die Theorie des Käufigläufermotors unter Berücksichtigung der Ständer- und Läufernutung*. Archiv für Elektrotechnik, 56, 1974, pp.331-339.
  - 50. Van der Merve F.S.: *The analysis of an electric machine with a smooth air-gap allowing MMF harmonics. Part1. The basic space vector component machine equations. Application of the basis equations to the steady state and transient conditions*. Archiv für Elektrotechnik, 58, 1976, pp.283-303.
  - 51. Van der Merve F.S.: *Reference frame and transformations for rotating machines with smooth air-gap and MMF harmonics*. Archiv für Elektrotechnik, 60, 1978, pp.181-191.
  - 52. Van der Merve F.S.: *The basis voltage and torque equations for squirrel-cage induction motors allowing for all MMF and permeance harmonics*. Archiv für Elektrotechnik, 64, 1982, pp.251-261.
  - 53. Vas P., Brown J.E., Shirley A.: *The application of N-phase generalized rotating field theory of induction machines with arbitrary stator windings connection*. IEEE Trans. PAS 103, 1984 pp.1270-1276.
  - 54. Willems J.L.: *Space harmonics in unified electrical machine theory*. Proc. IEE, Vol.118, 1971, pp.1408-1412.
  - 55. White D., WoodsonH.: *Electromechanical energy conversion*. John Wiley & Sons, New York, 1959.
  - 56. Yamamura S.: *Spiral vector theory of AC motors*. Proc. of Int. Conference on Electrical Machines ICEM'90, Cambridge, MIT, USA, 1990.

## APPENDIX

### COMPUTER SIMULATION PROGRAM FOR EDUCATIONAL PURPOSES

The computer simulation program has been elaborated for educational purposes in order to illustrate the main features and results of step-by-step analysis of the chosen simplified  $m_s$ -phase induction machine with  $m_r$ -phase rotor. The stator and rotor consist of coils symmetrically shifted for angles  $2\pi/m_s$  and  $2\pi/m_r$ , respectively. It is possible to choose full-pitch coils as well as chorded coils with different values of turns:  $N_s$  and  $N_r$  and values of pitch:  $Y_s$  and  $Y_r$  – and the arbitrary value of stator and rotor magnetic channels:  $Q_s$  and  $Q_r$ .

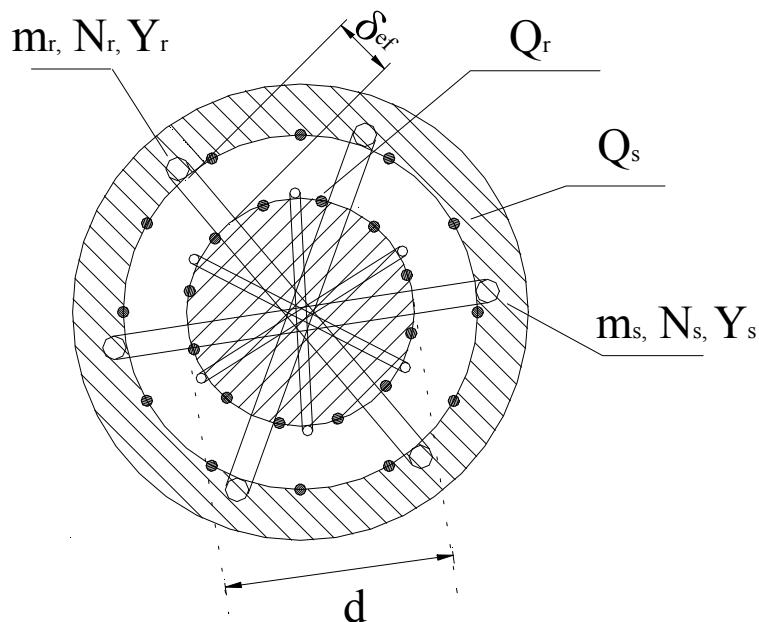


Fig.6.1. Parameters of induction motor to be chosen

The program ‘Elmech\_piecewise\_const.exe’ included in this book is compiled as a self-executing program. The activation by double-click on the

main form area opens the windows-like menu which enables to choose the main topics of program activity including:

1. Choice of the parameters for the induction motor (SI unit system)
2. Write/read for own motor data set.
3. Determination of number of steps Q and plotting the stator-rotor inductance versus rotor angle curve.
4. Solution of state equations at transient-state (for either given constant torque or for given constant speed), determination of steady-state torque v. speed curve and reluctance torque v. rotor angle curve (for either rotor slot pitch or  $2\pi/\text{GCD}(Q_s, Q_r)$  angle).
5. Saving the curve presented in the chart.
6. Finishing the program activity.

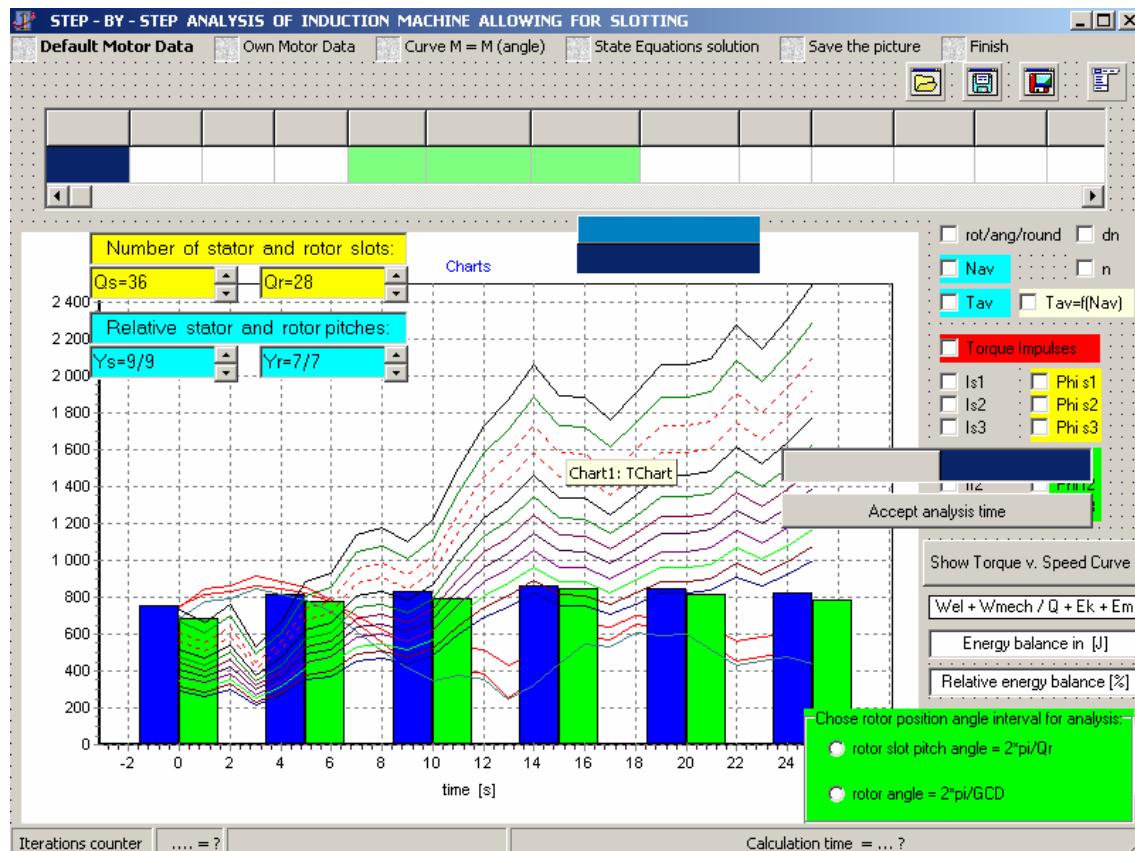


Fig 6.2. Main form and components of the educational program

1. **Default Motor Data** (SI unit system) – this submenu enables to choose supply and load conditions as well as induction machine parameters e.g.  $m_s$ ,  $m_r$   $Q_s$ ,  $Q_r$ ,  $Y_s$ ,  $Y_r$ ,  $N_s$ ,  $N_r$ .

The data strings present the default (demonstration) data, but it is also possible to change them into one's own data. In the computer program the data are denoted as follows:

**f1** - supply frequency [Hz],

**p** - number of pole-pair,

**U** - phase voltage [V],

**Tm** - mechanical torque [Nm],

**No** - initial rotor speed – can be expressed either as [rps] or [rpm],

*This program enables to calculate the values of speed in either [rps] or [rpm] after clicking the date schedule (Delphi StringGrid).*

**init/rot/ang=0÷1** – initial rotor angle:

init/rot/ang = 0 corresponds with rotor position at which stator-rotor inductance is maximum,

init/rot/ang = 1 for the whole circumference is described by the value,

**L1o, L2o** - stator and rotor leakage inductance, respectively [mH]

**L2c** - rotor leakage ring section inductance [mH],

**R1, R2** - stator and rotor resistance, respectively [ $\Omega$ ]

**R2c** - rotor ring section resistance [ $\Omega$ ],

**J** - moment of inertia [ $\text{kg}\cdot\text{m}^2$ ],

**Ns, Nr** - stator and rotor coils number, respectively,

**l<sub>e</sub>** - equivalent length of machine [m],

**d** - inner diameter of stator [m],

**delta** - air-gap width [mm],

**ms, mr** - stator and rotor phase numbers, respectively,

**Qs, Qr** - stator and rotor slot number, respectively

**Ys, Yr** - stator and rotor pitch factor, respectively (for squirrel cage rotor

**Yr = 1/pQr**).

2. **Own Motor Data** (SI unit system) – this submenu enables to save and then read own parameters set for motor basing on text files structure.

**3. Number of M(angle) steps** – it activates (on click) the procedure which calculates the number of the step changes for stator-rotor inductance.

**4. Curve  $M = M(\text{angle})$**  – it enables (on click) to show stator-rotor inductance versus rotor angle curve in the form of

**Rough plot**, (for  $Q_s \cdot Q_r$  points; shortcut F2, the submenus “Number of stator/rotor slots” do not disappear)

**Fine plot**, (for  $10 \cdot Q_s \cdot Q_r$  points, shortcut F3, the submenus “Number of stator/rotor slots” vanishes).

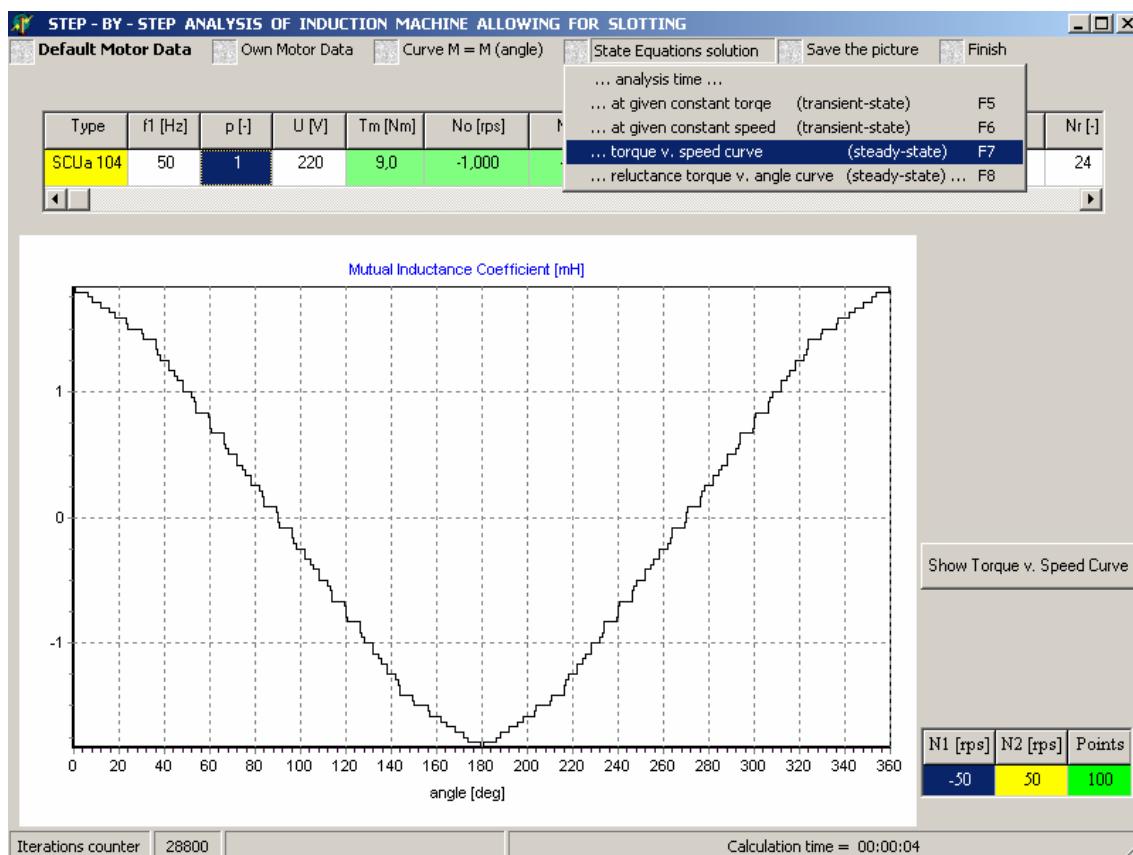


Fig.6.3. Stator-rotor inductance versus rotor angle

**5. State Equations solution ...** - it enables to choose either **transient-state** behaviour **at given constant torque** or **at given constant speed** or determination of steady-state torque v. speed curve or reluctance torque v. rotor angle curve.

There are four submenu positions for:

**... time for analysis change ...**,

**...at given constant torque (transient-state)** (shortcut F5),

- ...at given constant speed (transient-state) (shortcut F6),
- ...torque v. speed curve (shortcut F7),
- ...reluctance torque v. rotor angle curve (shortcut F8).

The ... **analysis time** ... displays the default analysis time and enables to change them in the form given below

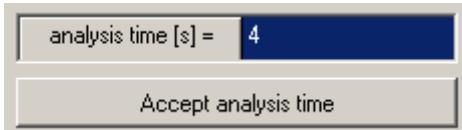


Fig 6.4. Submenu **analysis time**

The ...**at given constant torque (transient-state)** or ...**at given constant speed (transient-state)** submenus start analysis of transient-state. Double click on the main form (grey surface) the ...**at given constant torque (transient-state)** option is activated, too.

This option ...**torque v. speed curve** carries calculation series of torque-speed curve by completing steady-state both torque and speed values as a final results of transient state analyses (calculation time for particular step is given by ... **time for analysis** ... ).

There is also possible to collect values of reluctance torque at given synchronous speed (e.g.  $n_{syn}^1$  - at synchronous speed can be detected by shortcut F6) v. rotor angle by:

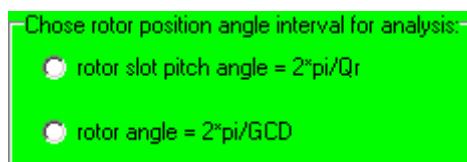
#### **...reluctance torque v. rotor angle curve.**

That submenus completes steady states values of torques at different rotor position angles for constant rotor speed (calculation time for particular step is given by ... **time for analysis** ... ).

There could be chosen one of two intervals for rotor position angle:

rotor slot pitch angle or

$2\pi/\text{GCD}(Q_s, Q_r)$  angle where  $\text{GCD}(Q_s, Q_r)$  means the greatest common divisor for stator and rotor slots numbers  $Q_s$  and  $Q_r$ .



Completing the calculations the checkboxes for choosing the electrical or mechanical variables to present them as curves.

The checkboxes enable to show:

**rot/ang/round** - rotor angle expressed in rounds ( $2\pi$  corresponds to 1 round) versus time,

**n** - rotor speed [rps],

**dn** - sequence of step-change values of rotor speed [rps],

**Tav** - average value of torque [Nm],

**Nav** - average value of speed [rps],

**Tav = f(Nav)** - torque versus speed curve,

**Is1,..., Is3, Ir1,..., Ir3** - stator and rotor currents, respectively [A],

**Phis1,..., Phis3, Phir1,..., Phir3** - stator and rotor fluxes, respectively [Wb].

The program checks the correctness of calculation by means of the given below equation which describes energy balance of the induction motor

$$W_{el} + W_{mech} = \Delta E_k + \Delta E_m + Q,$$

where  $W_{el}$  means the work of electric sources [J]

$$W_{el} = \sum_{k=1}^K \frac{1}{4} [\mathbf{U}(t_k) + \mathbf{U}(t_{k+1})]^T [\mathbf{I}(t_k) + \mathbf{I}(t_{k+1})],$$

$K$  denotes the integer number that corresponds to the analysis time,

$W_{mech}$  - work of mechanical forces [J]

$$W_{mech} = - \sum_{k=1}^K T_m (\vartheta_{k+1} - \vartheta_k),$$

$\Delta E_k$  - change of kinetic energy [J]

$$\Delta E_k = \frac{1}{2} J \Omega_{m,k}^2 - \frac{1}{2} J \Omega_{m,0}^2,$$

$\Delta E_m$  - change of magnetic energy [J]

$$\Delta E_{m,k} = \sum_{k=1}^K \frac{1}{2} \Psi(t_k)^T \cdot \Delta I(t_k),$$

$Q$  - Joule losses of heat energy [J]

$$Q = \sum_{k=1}^K \frac{1}{4} [\mathbf{I}(t_k) + \mathbf{I}(t_{k+1})]^T \cdot \mathbf{R} \cdot [\mathbf{I}(t_k) + \mathbf{I}(t_{k+1})].$$

The results are presented in two edit-boxes in the form of energy equality in Joules and in relative - percentage form for the error defined as follows

$$\text{Error} = \frac{(W_{\text{el}} + W_{\text{mech}}) - (Q + \Delta E_k + \Delta E_m)}{W_{\text{el}} + W_{\text{mech}}} \cdot 100\%.$$

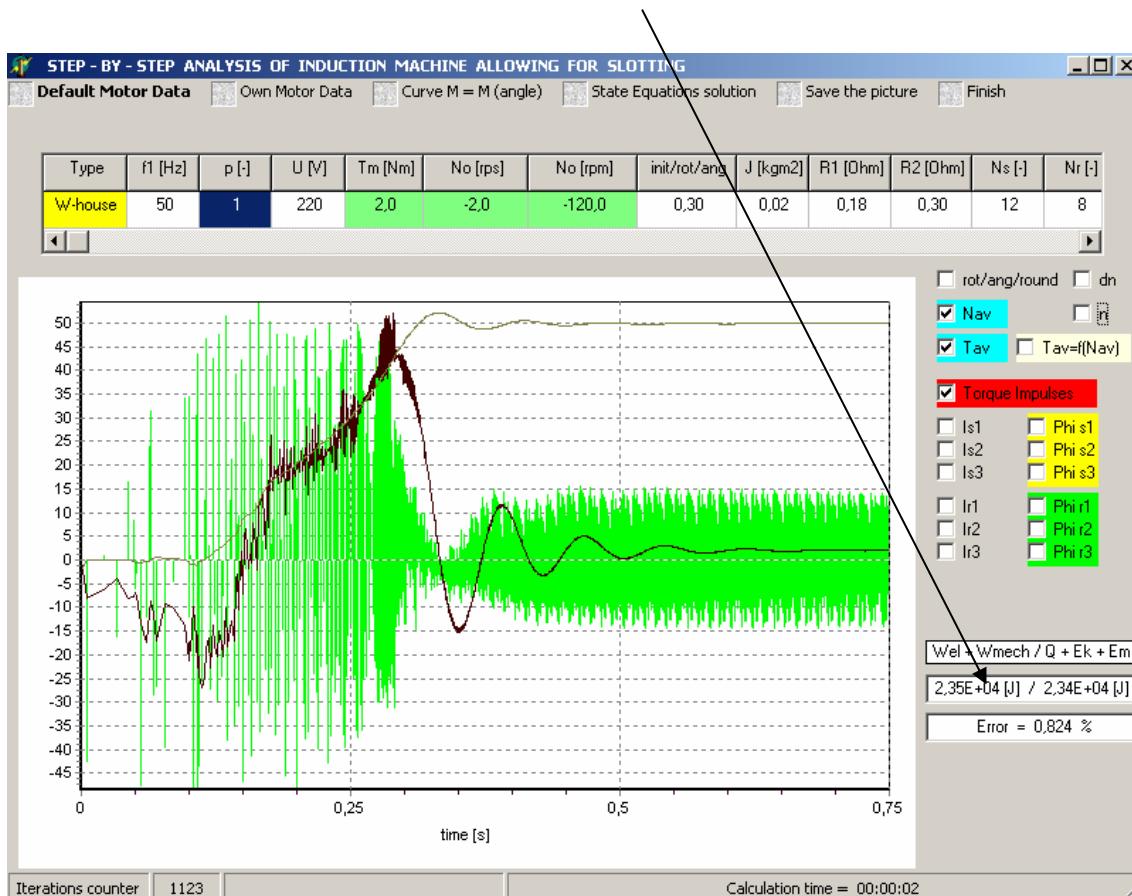


Fig.6.5. Transient-state at given constant torque – result presentation

The ...torque v. speed curve submenu enables to set the endpoints for the calculated curve (N1 – startpoint, N2 – endpoint). The cell Points introduces into the program the number of points for the considered curve.

6. **Save the picture** – it opens the save-picture-dialog (click on the question: **where ?** ).
7. **Finish** – it finishes the program work (shortcut Del).

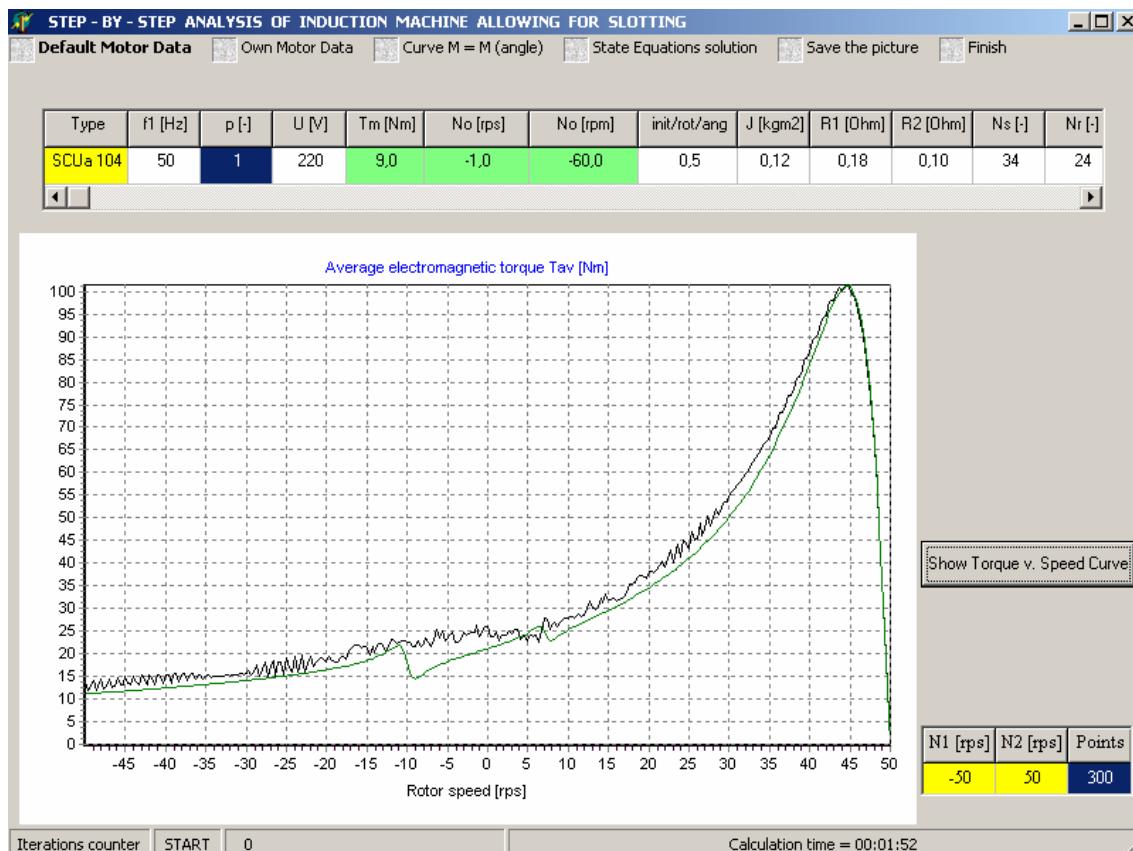


Fig.6.5. Steady-state torque v. speed curve – result presentation

## THE EDUCATIONAL PROGRAM LISTING – MAIN UNIT

```

unit Unit_State_Equations;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, Buttons, Unit_Dzes, Grids,
  TeEngine, Series, ExtCtrls, TeeProcs, Chart, Menus, ExtDlgs, Mask, ComCtrls, Aligrid;//Db, DBTables;
const density=100; mi=4*pi*1e-7; p2=1.4142135623730950488016887242097 {=sqrt(2)};
      p3=1.73205080756887729352744634150587; {=sqrt(3)}
type
  TForm1 = class(TForm)
    Num_of_Interv_Grid: TStringGrid; SavePictureDialog1: TSavePictureDialog;
    Chart1: TChart; MainMenu1: TMainMenu; MotorData: TStringAlignGrid; MCurve1: TMenuItem;
    MotorSZU72a: TMenuItem; Motor1812: TMenuItem; Motor1212: TMenuItem; MotorSCUA104: TMenuItem;
    Indukta3628: TMenuItem;
    Westinghouse1820:TMenuItem; SaveStateEqns1: TMenuItem; End1: TMenuItem; SaveChart1: TMenuItem;
    WhereSave1: TMenuItem; ForcedTorque1: TMenuItem; ForcedSpeed1: TMenuItem; TorqueSpeedCurve1:
    CheckBox1: TCheckBox; CheckBox6: TCheckBox; CheckBox11: TCheckBox; CheckBox17: TCheckBox;
    CheckBox2: TCheckBox; CheckBox7: TCheckBox; CheckBox12: TCheckBox; CheckBox18: TCheckBox;
    CheckBox3: TCheckBox; CheckBox8: TCheckBox; CheckBox13: TCheckBox; CheckBox19: TCheckBox;
    CheckBox4: TCheckBox; CheckBox9: TCheckBox; CheckBox14: TCheckBox;
    CheckBox5: TCheckBox; CheckBox10:TCheckBox; CheckBox15: TCheckBox;
    Series4: TBarSeries;           CheckBox16: TCheckBox;
    Series2: TLineSeries;   Series12: TLineSeries; Series13: TLineSeries;
    Series5: TBarSeries;   Series14: TLineSeries; Series3: TLineSeries;
    Series6: TFastLineSeries; Series15: TLineSeries; Series21: TFastLineSeries;
    Series7: TFastLineSeries; Series16: TLineSeries; Series22: TFastLineSeries;
    Series8: TFastLineSeries; Series17: TLineSeries; Series23: TFastLineSeries;
    Series9: TFastLineSeries; Series18: TLineSeries; Series24: TLineSeries;
    Series10:TFastLineSeries; Series19: TLineSeries; Series25: TFastLineSeries;
    Series11:TFastLineSeries; Series1: TLineSeries; Series20: TFastLineSeries;
    Edit1: TEdit; MaskEdit1: TMaskEdit; StatusBar1: TStatusBar; StaticText1: TStaticText; Determin_T_n: TButton;
    String_T_n_Curve: TStringAlignGrid; roughplot1: TMenuItem; fineplot1: TMenuItem; MaskEdit2: TMaskEdit;
    UpDown2: TUpDown; MaskEdit3: TMaskEdit; UpDown1: TUpDown; MaskEdit4: TMaskEdit; UpDown3:
    TUpDown; MaskEdit5: TMaskEdit; UpDown4: TUpDown; TorqueAngleCurve1: TMenuItem; MaskEdit6:
    TMaskEdit; MaskEdit7: TMaskEdit; Number1: TMenuItem; Motor1: TMenuItem; OwnMotorData1: TMenuItem;
    where1: TMenuItem; SaveDialog1: TSaveDialog; Readdatafrom1: TMenuItem; OpenDialog1: TOpenDialog;
    calculationtime1: TMenuItem; StringGrid1: TStringGrid; BitBtn1: TBitBtn; RG1: TRadioGroup; Rb1:
    TRadioButton; Rb2: TRadioButton;

    Procedure Data0(S: TObject); Procedure Data1(S: TObject); Procedure Data2(S: TObject); Procedure Data3(S:
    TObject); Procedure Data4(S: TObject); Procedure Data5(S: TObject); Procedure Koniec(S:TObject); procedure
  
```

## Computer simulation program for educational purposes

---

```
Curve_M(S:TObject); procedure Curve_M_fine(S:TObject); procedure SaveCurve(S:TObject); procedure
SaveOwnMotorData(S:TObject); procedure ReadOwnMotorData(S:TObject);
procedure Number_of_IntervalsClick(Sender: TObject); procedure ForcedTorque1Click(Sender: TObject);
procedure ForcedSpeed1Click(Sender: TObject);
Procedure Curve1(S:TObject);Procedure Curve2(S:TObject);Procedure Curve3(S:TObject);Procedure
Curve4(S:TObject);Procedure CurveIs1(S:TObject);Procedure CurveIs2(S:TObject);Procedure
CurveIs3(S:TObject);Procedure CurveIr1(S:TObject);Procedure CurveIr2(S:TObject);Procedure
CurveIr3(S:TObject); Procedure CurveFis1(S:TObject);Procedure CurveFis2(S:TObject);Procedure
CurveFis3(S:TObject);Procedure CurveFir1(S:TObject);Procedure CurveFir2(S:TObject);Procedure
CurveFir3(S:TObject);Procedure CurveTav(S:TObject);Procedure CurveNav(S:TObject);Procedure
CurveTavNav(S:TObject); procedure TorqueSpeedCurve1Click(Sender: TObject); procedure
Torque_Angle_Curve1Click(Sender: TObject); procedure Determin_T_nClick(Sender: TObject);
procedure UpDown1ChangingEx(Sender: TObject; var AllowChange: Boolean; NewValue: Smallint; Direction:
TUpDownDirection);procedure UpDown2ChangingEx(Sender: TObject; var AllowChange: Boolean; NewValue:
Smallint; Direction: TUpDownDirection);procedure UpDown3ChangingEx(Sender: TObject; var AllowChange:
Boolean; NewValue: Smallint; Direction: TUpDownDirection);procedure UpDown4ChangingEx(Sender: TObject;
var AllowChange: Boolean; NewValue: Smallint; Direction: TUpDownDirection);
procedure OnClickMotorData(S: TObject);
procedure calculationtime1Click(Sender: TObject); procedure BitBtn1Click(Sender: TObject);

private { Private declarations }
Procedure MUTUAL_INDCTANCE_CHART; Procedure MUTUAL_INDCTANCE_CHART_fine;
Procedure Data_Load; Procedure Solving_Differential_State_Eqns;
public { Public declarations }
Procedure Initial_Calculations; Procedure CheckBoxesNonVisible; Procedure CheckBoxesVisible; Procedure
CheckBoxesNonChecked; Procedure ChartSeriesNonVisible; Procedure ChartSeries2NonVisible; Procedure
ChartSeriesKlossNonVisible; Procedure ChartSeriesVisible; Procedure UpDownsNonVisible;
Procedure UpDownsVisible;
end;
VAR Form1:TForm1; Motor:TMotor; Ig:Integer; TC:TFloat; ResVect:TResult; XR:macierz_wynikow;

IMPLEMENTATION {$R *.DFM}
Procedure TForm1.Data0(S: TObject); {SZUA 72}
Begin Num_of_Interv_Grid.visible:=false; UpDownsVisible; TC:=1.5;
with MotorData do begin Cells[0,0]:='Type'; Cells[1,0]:='f1 [Hz]';Cells[2,0]:='p [-]'; Cells[3,0]:='U [V]';
Cells[4,0]:='Tm [Nm]'; Cells[5,0]:='No [rps]'; Cells[6,0]:='No [rpm]';Cells[13,0]:='L1o [mH]';
Cells[14,0]:='L2o [mH]'; Cells[8,0]:='J [kgm2]';Cells[11,0]:='Ns [-]';Cells[12,0]:='Nr [-]';Cells[15,0]:='le [m]';
Cells[16,0]:='d [m]'; Cells[17,0]:='delta[mm]'; Cells[19,0]:='ms >2'; Cells[20,0]:='mr >2';Cells[9,0]:='R1 [Ohm]';
Cells[10,0]:='R2 [Ohm]';Cells[21,0]:='Cage Rotor ? (Y/N)'; Cells[22,0]:='R2c [Ohm]';Cells[23,0]:='L2c
[mH]';Cells[7,0]:='init/rot/ang'; Cells[0,1]:='SZUA 72'; Cells[1,1]:='50'; Cells[2,1]:='1'; Cells[3,1]:='220';
Cells[4,1]:='3,0'; Cells[5,1]:='2,0'; Cells[6,1]:='120,0'; Cells[13,1]:='1,25'; Cells[14,1]:='1,25';
Cells[8,1]:='0,10'; Cells[11,1]:='68'; Cells[12,1]:='48'; Cells[15,1]:='0,15';
Cells[16,1]:='0,166';Cells[17,1]:='0,65'; Cells[19,1]:='3'; Cells[20,1]:='3'; Cells[9,1]:='0,18';
Cells[10,1]:='0,30'; Cells[21,1]:='N'; Cells[22,1]:='0,0'; Cells[23,1]:='0,0'; Cells[7,1]:='0,5'; end;
Motor.Qs:=48; MaskEdit2.Text:='Qs=48'; Motor.Qr:=36; MaskEdit3.Text:='Qr=36'; Motor.Ys:=20/24;
MaskEdit4.Text:='Ys=20/24'; Motor.Yr:=12/18; MaskEdit5.Text:='Yr=12/18'; Motor.ks:=8; Motor.kr:=6;
Motor.CageRotor:='N'; MaskEdit2.Update; MaskEdit3.Update; MaskEdit4.Update; MaskEdit5.Update;
MaskEdit6.visible:=true; MaskEdit7.visible:=true; Data_Load;
End;
Procedure TForm1.Data1(S: TObject); {SCUA 104}
```

```

Begin Num_of_Interv_Grid.visible:=false; UpDownsVisible; TC:=2;
with Motordata do begin Cells[0,0]:='Type'; Cells[1,0]:='f1 [Hz]';Cells[2,0]:='p [-]'; Cells[3,0]:='U [V]';
Cells[4,0]:='Tm [Nm]';Cells[5,0]:='No [rps]';Cells[6,0]:='No [rpm]';Cells[13,0]:='L1o [mH]';
Cells[14,0]:='L2o [mH]'; Cells[8,0]:='J [kgm2]';Cells[11,0]:='Ns [-]';Cells[12,0]:='Nr [-]';Cells[15,0]:='le [m]';
Cells[16,0]:='d [m]'; Cells[17,0]:='delta[mm]';{Cells[18,0]:='Rotor teeth number (A/B)?';} Cells[19,0]:='ms >2';
Cells[20,0]:='mr >2'; Cells[9,0]:='R1 [Ohm]';Cells[10,0]:='R2 [Ohm]';Cells[21,0]:='Cage Rotor ? (Y/N)';
Cells[22,0]:='R2c [Ohm]'; Cells[23,0]:='L2c [mH]';Cells[7,0]:='init/rot/ang'; Cells[0,1]:='SCUA 104';
Cells[1,1]:='50'; Cells[2,1]:='1'; Cells[3,1]:='220'; Cells[4,1]:='9,0'; Cells[5,1]:='-1,0';
Cells[6,1]:='-60,0'; Cells[13,1]:='1,25'; Cells[14,1]:='1,25'; Cells[8,1]:='0,12'; Cells[11,1]:='34';
Cells[12,1]:='24'; Cells[15,1]:='0,175'; Cells[16,1]:='0,166';Cells[17,1]:='0,65'; Cells[19,1]:='3';
Cells[20,1]:='3'; Cells[9,1]:='0,18'; Cells[10,1]:='0,10'; Cells[21,1]:='N'; Cells[22,1]:='0,0';
Cells[23,1]:='0,0'; Cells[7,1]:='0,5'; end;

Motor.Qs:=48; MaskEdit2.Text:='Qs=48'; Motor.Qr:=60; MaskEdit3.Text:='Qr=60'; Motor.Ys:=20/24;
MaskEdit4.Text:='Ys=20/24'; Motor.Yr:=30/30; MaskEdit5.Text:='Yr=30/30'; Motor.ks:=8; Motor.kr:=10;
Motor.CageRotor:='N'; MaskEdit2.Update; MaskEdit3.Update; MaskEdit4.Update; MaskEdit5.Update;
MaskEdit6.visible:=true; MaskEdit7.visible:=true; Data_Load;
End;

Procedure TForm1.Data2(S: TObject); {Model 12-12}
Begin Num_of_Interv_Grid.visible:=false; UpDownsVisible; TC:=1.5;
with MotorData do begin Cells[0,0]:='Type'; Cells[1,0]:='f1 [Hz]';Cells[2,0]:='p [-]';Cells[3,0]:='U [V]';
Cells[4,0]:='Tm [Nm]';Cells[5,0]:='No [rps]';Cells[6,0]:='No [rpm]';Cells[13,0]:='L1o [mH]';
Cells[14,0]:='L2o [mH]'; Cells[8,0]:='J [kgm2]';Cells[11,0]:='Ns [-]';Cells[12,0]:='Nr [-]';Cells[15,0]:='le [m]';
Cells[16,0]:='d [m]'; Cells[17,0]:='delta[mm]'; Cells[19,0]:='ms >2'; Cells[20,0]:='mr >2';Cells[9,0]:='R1 [Ohm]';
Cells[10,0]:='R2 [Ohm]'; Cells[21,0]:='Cage Rotor ? (Y/N)';Cells[22,0]:='R2c [Ohm]';Cells[23,0]:='L2c [mH]';
Cells[7,0]:='init/rot/ang'; Cells[0,1]:='M 12-12'; Cells[1,1]:='50'; Cells[2,1]:='1'; Cells[3,1]:='220';
Cells[4,1]:='4,0'; Cells[5,1]:='16,666666';Cells[6,1]:='1000,0'; Cells[13,1]:='1,25'; Cells[14,1]:='1,25';
Cells[8,1]:='0,10'; Cells[11,1]:='34'; Cells[12,1]:='24'; ;Cells[15,1]:='0,15';
Cells[16,1]:='0,166';Cells[17,1]:='0,65'; Cells[19,1]:='3'; Cells[20,1]:='3'; Cells[9,1]:='0,18';
Cells[10,1]:='0,10'; Cells[21,1]:='N'; Cells[22,1]:='0,0'; Cells[23,1]:='0,0'; Cells[7,1]:='0,5'; end;

Motor.Qs:=12; MaskEdit2.Text:='Qs=12'; Motor.Qr:=12; MaskEdit3.Text:='Qr=12'; Motor.Ys:=6/6;
MaskEdit4.Text:='Ys=6/6'; Motor.Yr:=6/6; MaskEdit5.Text:='Yr=6/6'; Motor.ks:=2; Motor.kr:=2;
Motor.CageRotor:='N'; MaskEdit2.Update; MaskEdit3.Update; MaskEdit4.Update; MaskEdit5.Update;
MaskEdit6.visible:=true; MaskEdit7.visible:=true; Data_Load;
End;

Procedure TForm1.Data3(S: TObject); {Model 18-12}
Begin Num_of_Interv_Grid.visible:=false; UpDownsVisible; TC:=1.5;
with MotorData do begin Cells[0,0]:='Type'; Cells[1,0]:='f1 [Hz]';Cells[2,0]:='p [-]';Cells[3,0]:='U [V]';
Cells[4,0]:='Tm [Nm]';Cells[5,0]:='No [rps]';Cells[6,0]:='No [rpm]';Cells[13,0]:='L1o [mH]';
Cells[14,0]:='L2o [mH]'; Cells[8,0]:='J [kgm2]';Cells[11,0]:='Ns [-]';Cells[12,0]:='Nr [-]';Cells[15,0]:='le [m]';
Cells[16,0]:='d [m]'; Cells[17,0]:='delta[mm]';{Cells[18,0]:='Rotor teeth number (A/B)?';} Cells[19,0]:='ms >2';
Cells[20,0]:='mr >2'; Cells[9,0]:='R1 [Ohm]';Cells[10,0]:='R2 [Ohm]';Cells[21,0]:='Cage Rotor ? (Y/N)';
Cells[22,0]:='R2c [Ohm]'; Cells[23,0]:='L2c [mH]';Cells[7,0]:='init/rot/ang'; Cells[0,1]:='M 18 -12';
Cells[1,1]:='50'; Cells[2,1]:='1'; Cells[3,1]:='220'; Cells[4,1]:='10,0'; Cells[5,1]:='-2,0';
Cells[6,1]:='-120,0'; Cells[13,1]:='1,25'; Cells[14,1]:='1,25'; Cells[8,1]:='0,12'; Cells[11,1]:='34';
Cells[12,1]:='24'; Cells[15,1]:='0,15'; Cells[16,1]:='0,166';Cells[17,1]:='0,65'; Cells[19,1]:='3';
Cells[20,1]:='3'; Cells[9,1]:='0,18'; Cells[10,1]:='0,10'; Cells[21,1]:='N'; Cells[22,1]:='0,0';
Cells[23,1]:='0,0'; Cells[7,1]:='0,5'; end;

Motor.Qs:=18; MaskEdit2.Text:='Qs=18'; Motor.Qr:=12; MaskEdit3.Text:='Qr=12'; Motor.Ys:=7/9;
MaskEdit4.Text:='Ys=7/9'; Motor.Yr:=6/6; MaskEdit5.Text:='Yr=6/6'; Motor.ks:=3; Motor.kr:=2;

```

## Computer simulation program for educational purposes

---

```
Motor.CageRotor:='N'; MaskEdit2.Update; MaskEdit3.Update; MaskEdit4.Update; MaskEdit5.Update;
MaskEdit6.visible:=true; MaskEdit7.visible:=true; Data_Load;
End;
// For Indukta Motor Nsynchr = 3,571435 [rps], 7,142857 [rps], 8,3333333 [rps]
Procedure TForm1.Data4(S: TObject); {Indukta 36-28, p=2}
Begin Num_of_Interv_Grid.visible:=false; UpDownsVisible; TC:=4;
with MotorData do begin Cells[0,0]:='Type'; Cells[1,0]:='f1 [Hz]'; Cells[2,0]:='p [-]'; Cells[3,0]:='U [V]';
Cells[4,0]:='Tm [Nm]'; Cells[5,0]:='No [rps]'; Cells[6,0]:='No [rpm]'; Cells[13,0]:='L1o [mH]'; Cells[14,0]:='L2o
[mH]'; Cells[8,0]:='J [kgm2]'; Cells[11,0]:='Ns [-]'; Cells[12,0]:='Nr [-]'; Cells[15,0]:='le [m]'; Cells[16,0]:='d [m]';
Cells[17,0]:='delta[mm]'; {Cells[18,0]:='Rotor teeth number (A/B)?';} Cells[19,0]:='ms >2'; Cells[20,0]:='mr >2';
Cells[9,0]:='R1 [Ohm]'; Cells[10,0]:='R2 [Ohm]'; Cells[21,0]:='Cage Rotor ? (Y/N)'; Cells[22,0]:='R2c [Ohm]';
Cells[23,0]:='L2c [mH]'; Cells[7,0]:='init/rot/ang'; Cells[0,1]:='Indukta'; Cells[1,1]:='50'; Cells[2,1]:='2';
Cells[3,1]:='220'; Cells[4,1]:='4,0'; Cells[5,1]:='1,0'; Cells[6,1]:='60,0'; Cells[13,1]:='1,20';
Cells[14,1]:='1,10'; Cells[8,1]:='0,90'; Cells[11,1]:='32'; Cells[12,1]:='8'; Cells[15,1]:='0,12';
Cells[16,1]:='0,100'; Cells[17,1]:='0,55'; Cells[19,1]:='3'; Cells[20,1]:='14'; Cells[9,1]:='0,12';
Cells[10,1]:='0,20'; Cells[21,1]:='Y'; Cells[22,1]:='0,4'; Cells[23,1]:='1,0'; Cells[7,1]:='0,00'; end;
Motor.Qs:=36; MaskEdit2.Text:='Qs=36'; Motor.Qr:=28; MaskEdit3.Text:='Qr=28'; Motor.Ys:=8/9;
MaskEdit4.Text:='Ys=8/9'; Motor.Yr:=1/14; MaskEdit5.Text:='Yr=1/14'; Motor.ks:=3; Motor.kr:=1;
Motor.CageRotor:='Y'; MaskEdit2.Update; MaskEdit3.Update; MaskEdit4.Update; MaskEdit5.Update;
MaskEdit6.visible:=true; MaskEdit7.visible:=true; Data_Load;
End;
// For Westinghouse Motor Nsynchr = 5,0 [rps]
Procedure TForm1.Data5(S: TObject); {Westinghouse 18-20, p=1}
Begin Num_of_Interv_Grid.visible:=false; UpDownsVisible; TC:=1.0;
with MotorData do begin Cells[0,0]:='Type'; Cells[1,0]:='f1 [Hz]'; Cells[2,0]:='p [-]'; Cells[3,0]:='U [V]';
Cells[4,0]:='Tm [Nm]'; Cells[5,0]:='No [rps]'; Cells[6,0]:='No [rpm]'; Cells[13,0]:='L1o [mH]'; Cells[14,0]:='L2o
[mH]'; Cells[8,0]:='J [kgm2]'; Cells[11,0]:='Ns [-]'; Cells[12,0]:='Nr [-]'; Cells[15,0]:='le [m]'; Cells[16,0]:='d [m]';
Cells[17,0]:='delta[mm]'; {Cells[18,0]:='Rotor teeth number (A/B)?';} Cells[19,0]:='ms >2'; Cells[20,0]:='mr >2';
Cells[9,0]:='R1 [Ohm]'; Cells[10,0]:='R2 [Ohm]'; Cells[21,0]:='Cage Rotor ? (Y/N)'; Cells[22,0]:='R2c [Ohm]';
Cells[23,0]:='L2c [mH]'; Cells[7,0]:='init/rot/ang'; Cells[0,1]:='W-house'; Cells[1,1]:='50';
Cells[2,1]:='1'; Cells[3,1]:='220'; Cells[4,1]:='2,0'; Cells[5,1]:='-2,0'; Cells[6,1]:='-120,0';
Cells[13,1]:='2,0'; Cells[14,1]:='2,0'; Cells[8,1]:='0,02'; Cells[11,1]:='12'; Cells[12,1]:='8'; ;
Cells[15,1]:='0,15'; Cells[16,1]:='0,166'; Cells[17,1]:='0,65'; Cells[19,1]:='3'; Cells[20,1]:='20';
Cells[9,1]:='0,18'; Cells[10,1]:='0,30'; Cells[21,1]:='Y'; Cells[22,1]:='0,39'; Cells[23,1]:='1,0';
Cells[7,1]:='0,30'; end;
Motor.Qs:=18; MaskEdit2.Text:='Qs=18'; Motor.Qr:=20; MaskEdit3.Text:='Qr=20'; Motor.Ys:=8/9;
MaskEdit4.Text:='Ys=8/9'; Motor.Yr:=1/20; MaskEdit5.Text:='Yr=1/20'; Motor.ks:=3; Motor.kr:=1;
Motor.CageRotor:='Y'; MaskEdit2.Update; MaskEdit3.Update; MaskEdit4.Update; MaskEdit5.Update;
MaskEdit6.visible:=true; MaskEdit7.visible:=true; Data_Load; End;
Procedure TForm1.Data_Load; var j:integer;
Begin MotorData.visible:=true; j:=Chart1.SeriesCount-1; while j>=0 do begin Chart1.Series[j].clear; j:=Pred(j);
end;
with Motor do begin {Record}
with MotorData do begin {rec}
f1:=StrToFloat(Cells[1,1]); p:=StrToInt(Cells[2,1]); U:=StrToFloat(Cells[3,1]);
Tm:=StrToFloat(Cells[4,1]); No:=StrToFloat(Cells[5,1]); No60:=StrToFloat(Cells[6,1]);
L1o:=StrToFloat(Cells[13,1])/1000; L2o:=StrToFloat(Cells[14,1])/1000; fio:=StrToFloat(Cells[7,1]);
fio:=Frac(fio); if fio < 0 then fio:=1+fio; Jo:=StrToFloat(Cells[8,1]); ms:=StrToInt(Cells[19,1]);
mr:=StrToInt(Cells[20,1]); Ns:=StrToInt(Cells[11,1]); Nr:=StrToInt(Cells[12,1]); l:=StrToFloat(Cells[15,1]); d:=Str

```

```

ToFloat(Cells[16,1]);delta:=StrToFloat(Cells[17,1])/1000;R1:=StrToFloat(Cells[9,1]);R2:=StrToFloat(Cells[10,1]);
;CageRotor:=UpperCase(Cells[21,1]);R2c:=StrToFloat(Cells[22,1]);L2c:=StrToFloat(Cells[23,1])/1000;
Repaint; Update; {rec}
end; {Record}
End;

procedure TForm1.OnClickMotorData(S: TObject); var N_old, N60_old:TFloat;
Begin N_old:=StrToFloat(MotorData.Cells[5,1]); N60_old:=StrToFloat(MotorData.Cells[6,1]);
if not(Motor.No=N_old) then begin Motor.No:=N_old; MotorData.Cells[5,1]:=FloatToStrF(N_old,ffFixed,7,3);
Motordata.Cells[6,1]:=FloatToStrF(60*N_old,ffFixed,7,3); end;
if not(Motor.No60=N60_old) then begin Motor.No60:=N60_old;
MotorData.Cells[6,1]:=FloatToStrF(N60_old,ffFixed,7,3);
MotorData.Cells[5,1]:=FloatToStrF(N60_old/60,ffFixed,7,3) end;
MotorData.Cells[21,1]:=UpperCase(MotorData.Cells[21,1]); if ( not(Motor.CageRotor='Y') and
not(Motor.CageRotor='N') ) then begin Motor.CageRotor:='N'; MotorData.Cells[21,1]:='N'; end;
MotorData.Update;
End; // stator and rotor slots numbers
procedure TForm1.UpDown1ChangingEx(Sender: TObject; var AllowChange: Boolean; NewValue: Smallint;
Direction: TUpDownDirection);
begin Motor.ms:=StrToInt(MotorData.Cells[19,1]); Motor.p:=StrToInt(MotorData.Cells[2,1]);
Motor.Qs:=2*Motor.ms*NewValue; MaskEdit2.Text:='Qs:='+IntToStr(Motor.Qs); Motor.ks:=NewValue end;
procedure TForm1.UpDown2ChangingEx(Sender: TObject; var AllowChange: Boolean; NewValue: Smallint;
Direction: TUpDownDirection);
begin Motor.mr:=StrToInt(MotorData.Cells[20,1]); Motor.p:=StrToInt(MotorData.Cells[2,1]); if
(Motor.CageRotor='N') then begin Motor.Qr:=2*Motor.mr*NewValue; Motor.kr:=NewValue end else begin
Motor.Qr:=NewValue; Motor.kr:=1;
Motor.mr:=StrToInt(MotorData.Cells[20,1]); Motor.p:=StrToInt(MotorData.Cells[2,1]);
if (Motor.CageRotor='N') then begin Motor.Yr:=NewValue/(Motor.Qr/2/Motor.p);
MaskEdit5.Text:='Yr:='+FloatToStr(NewValue)+'/'+FloatToStr(Motor.Qr/2/Motor.p); end;
if (Motor.CageRotor='Y') then begin NewValue:=1;
MaskEdit5.Text:='Yr:='+FloatToStr(NewValue)+'/'+FloatToStr(Motor.Qr/Motor.p); end;
end;
MaskEdit3.Text:='Qr:='+IntToStr(Motor.Qr); end;
// stator and rotor pitch factors - relative pitches Ys, Yr - teeth numbers per pair-pole
procedure TForm1.UpDown3ChangingEx(Sender: TObject; var AllowChange: Boolean; NewValue: Smallint;
Direction: TUpDownDirection);
begin Motor.ms:=StrToInt(MotorData.Cells[19,1]); Motor.p:=StrToInt(MotorData.Cells[2,1]);
Motor.Ys:=NewValue/(Motor.Qs/2/Motor.p);
MaskEdit4.Text:='Ys:='+FloatToStr(NewValue)+'/'+FloatToStr(Motor.Qs/2/Motor.p); end;
procedure TForm1.UpDown4ChangingEx(Sender: TObject; var AllowChange: Boolean; NewValue: Smallint;
Direction: TUpDownDirection);
begin Motor.mr:=StrToInt(MotorData.Cells[20,1]); Motor.p:=StrToInt(MotorData.Cells[2,1]);
if (Motor.CageRotor='N') then begin Motor.Yr:=NewValue/(Motor.Qr/2/Motor.p);
MaskEdit5.Text:='Yr:='+FloatToStr(NewValue)+'/'+FloatToStr(Motor.Qr/2/Motor.p); end;
if (Motor.CageRotor='Y') then begin NewValue:=1;
MaskEdit5.Text:='Yr:='+FloatToStr(NewValue)+'/'+FloatToStr(Motor.Qr/Motor.p); end; end;
Function mod1(D,T:integer):integer; var c:integer;
Begin
IF T<0 THEN HALT;
IF D>0 THEN BEGIN
  if D<=T then mod1:=D;

```

## Computer simulation program for educational purposes

---

```
if D>T then begin c:=D; repeat c:=c-T until (c<=T); mod1:=c; end;
      END;
IF D<=0 THEN BEGIN
  if D>-T then mod1:=T+D;
  if D<=-T then begin c:=D; repeat c:=c+T until (c>0); mod1:=c; end;
      END;
End;
function pr(s:State):State; begin if s=So then pr:=Sk else pr:=So; end;
function sgn(x:Tfloat):integer; begin if x > 0 then sgn:=-1; if x = 0 then sgn:= 0; if x < 0 then sgn:=-1; end;
Function Succ1(k,n:Integer):Integer; begin Result:=Succ(k); if (Result>n) then Result:=1; end;
Function Pred1(k,n:Integer):Integer; begin Result:=Pred(k); if (Result<1) then Result:=n; end;
{=====
Function MUTAL_IND_COEFF(i,j:integer;fi:Tfloat):Tfloat;
var k:integer; FiNi,FiNj,Fizk:Tfloat; argI,argJ,dZs,dZr,AI,AJ:Tfloat; cskS,cskW:Tfloat;
function a(k:integer;x:Tfloat):Tfloat; { csk=Ys=1.0 for full-pitch coil }
var csk:Tfloat; { csk=Ys<1.0 for chorded coil }
Begin { F } { csk=Yr=1/pQr for cage rotor }
  if k in [1..Motor.ms] then csk:=cskS;
  if k in [Motor.ms+1..Motor.ms+Motor.mr] then csk:=cskW;
  a:=0;
IF ( (k in [1..Motor.ms]) or (Motor.CageRotor='N') ) THEN
  BEGIN { for stator and rotor coiled }
    if cos(2*pi*x) > csk then a:=-0.5;
    if cos(2*pi*x) < -csk then a:=-0.5;
  END; { for stator and rotor coiled }
IF ( (Motor.CageRotor='Y') and (k in [Motor.ms+1..Motor.ms+Motor.mr]) ) THEN
  BEGIN { cage rotor }
    if cos(2*pi*x) > csk then a:=(1-Motor.Yr)*(Motor.Qr-1);
    if cos(2*pi*x) = csk then a:=0.5*(Motor.Qr*(1-Motor.Yr)-1); // average ;
    if cos(2*pi*x) < csk then a:=-Motor.Yr;
  END; { cage rotor }
End; { F }
Begin { PG } with Motor do begin {record}
  Mm:=mi*l*pi*d/(4*p*p*delta*(Qs+Qr)); Result:=0; dZs:=1/Qs; dZr:=1/Qr;
  cskS:=cos(pi*Ys/2); cskW:=cos(pi*Yr/2);
  if i in [1..ms] then FiNi:=(i-1)/ms;
  if i in [ms+1..ms+mr] then FiNi:=fi+(i-ms-1)/mr;
  if j in [1..ms] then FiNj:=(j-1)/ms;
  if j in [ms+1..ms+mr] then FiNj:=fi+(j-ms-1)/mr;
  FOR k:=1 to Qs+Qr do begin { k }
    if k in [1..Qs] then begin
      if odd(k) then Fizk:= k*dZs
      else Fizk:=(k-0.5)*dZs; end;
    if k in [Qs+1..Qs+Qr] then begin
      if odd(k) then Fizk:= fi+(k-Qs)*dZr
      else Fizk:= fi+(k-Qs-0.5)*dZr; end;
    argI:=Fizk-FiNi; ArgJ:=Fizk-FiNj;
    If not(p=1) then begin argI:=argI*p; argJ:=ArgJ*p; end;
    AI:=a(i,argI); AJ:=a(j,argJ);
    If not( (AI=0) or (AJ=0) ) Then Result:=Result+Mm*AI*AJ;
  End;
End;
```

```

        end; { k }
    end; { record }

End; { PG }

Function Step_Change_Number:Integer;
  const epsD=1e-6; Amount=180; var k,Io,Ik:integer; CHANGE:boolean; M1oR,M1R:Tfloat;
begin { P }

  Result:=0; CHANGE:=True; s:=So; M1oR:=0.25*Motor.Ns*Motor.Nr*MUTAL_IND_COEFF(1, Motor.ms+1,0);
  for k:=1 to Amount do begin { L_SK }

    M1R:=0.25*Motor.Ns*Motor.Nr*MUTAL_IND_COEFF(1, Motor.ms+1,k/(2*Amount));

    { 1st phase } IF (abs(M1R-M1oR) < epsD) THEN
      begin CHANGE:=False; Ik:=k-1; end
    ELSE begin {else}
      IF CHANGE=False THEN begin { 1 }
        Io:=k-1; s:=Sk;
        end; { 1 }
      CHANGE:=True; M1oR:=M1R;
      end; { else }

      IF ((s=Sk) and (CHANGE=False)) THEN
        begin { 2 } Result:=Result+1; Angle[Result]:=0.5*(Io+Ik)/(2*Amount); s:=So; end; { 2 }
      end; { L_SK }

    For k:=1 to Result do Angle[Result+k]:=1-Angle[Result-k+1];
    Result:=2*Result; Angle[0]:=Angle[Result]-1; Angle[Result+1]:=1+Angle[1];
  end; { P }

Procedure TForm1.MUTUAL_INDCTANCE_CHART; var k:integer;
BEGIN { MUTUAL_INDCTANCE_CHART_rough } Ilpkt:=Motor.Qs*Motor.Qr; Scale:=360/Ilpkt;
StatusBar1.Panels[1].Text:=IntToStr(Ilpkt); StatusBar1.Update; Edit1.Visible:=false; MaskEdit1.Visible:=false;
StaticText1.Visible:=false;

  Chart1.Series[0].clear; Chart1.Series[1].clear; CountResults:=7+3*(Motor.ms+Motor.mr);
  SetLength(ResVect,CountResults+1);
  for k:=0 to Ilpkt do begin { pg }

    ResVect[1]:=Scale*k; ResVect[2]:=MUTAL_IND_COEFF(1, Motor.ms+1,k/Ilpkt);
    ResVect[2]:=0.25*1000*Motor.Ns*Motor.Nr*ResVect[2]; // mH
    Chart1.Series[1].AddXY(ResVect[1],ResVect[2],"Chart1.Series[1].SeriesColor");
    Num_of_Interv_Grid.visible:=false;
    end; { pg }

  END; { MUTUAL_INDCTANCE_CHART_rough }

Procedure TForm1.MUTUAL_INDCTANCE_CHART_fine; var k:integer;
BEGIN { MUTUAL_INDCTANCE_CHART_fine } Ilpkt:=10*Motor.Qs*Motor.Qr; Scale:=360/Ilpkt;
StatusBar1.Panels[1].Text:=IntToStr(Ilpkt); StatusBar1.Update; Edit1.Visible:=false; MaskEdit1.Visible:=false;
StaticText1.Visible:=false;// Chart1.Animatedzoom:=false;

  Chart1.Series[0].clear; Chart1.Series[1].clear; CountResults:=7+3*(Motor.ms+Motor.mr);
  SetLength(ResVect,CountResults+1);
  for k:=0 to Ilpkt do begin { pg }

    ResVect[1]:=Scale*k; ResVect[2]:=MUTAL_IND_COEFF(1, Motor.ms+1,k/Ilpkt);
    ResVect[2]:=0.25*1000*Motor.Ns*Motor.Nr*ResVect[2]; // mH
    Chart1.Series[1].AddXY(ResVect[1],ResVect[2],"Chart1.Series[1].SeriesColor");
    Num_of_Interv_Grid.visible:=false;
    end; { pg }

  END; { MUTUAL_INDCTANCE_CHART_fine }
{=====

```

## Computer simulation program for educational purposes

---

```
Function MATRIX_L(fi:Tfloat):Matrix; var i,j,k:Integer; zi,zj:Tfloat;
Begin
  with Motor do Begin {with}
    SetLength(Result,ms+mr+1,ms+mr+1); // for i:=0 to Motor.ms+Motor.mr do
    SetLength(Result[i],Motor.ms+Motor.mr+1);
    for i:=1 to ms+mr do
      for j:=i to ms+mr do begin {M}
        if i in [1..ms] then zi:=Ns;
        if i in [ms+1..ms+mr] then zi:=Nr;
        if j in [1..ms] then zj:=Ns;
        if j in [ms+1..ms+mr] then zj:=Nr;
        Result[i,j]:=zi*zj*MUTAL_IND_COEFF(i,j,fi)//Result[i][j];
        if i=j then begin { 1 }
          if i in [1..ms] then Result[i,j]:=Result[i][j]+L1o;
        IF ( (CageRotor='Y') and (i in [ms+1..ms+mr]) ) THEN {Cage Rotor}
          begin k:=i-ms; Result[i,j]:=Result[i][j]+2*(L2o+L2c);
          Result[ms+k][ms+Succ1(k,mr)]:=Result[ms+k][ms+Succ1(k,mr)]-L2o;
          Result[ms+k][ms+Pred1(k,mr)]:=Result[ms+k][ms+Pred1(k,mr)]-L2o; end;
        IF ( (CageRotor='N') and (i in [ms+1..ms+mr]) ) THEN {Rotor Coiled}
          begin Result[i,j]:=Result[i][j]+L2o; end;
        end; { 1 }
        end; {M}
        for i:=1 to ms+mr do
          for j:=1 to i do Result[i,j]:=Result[j][i];
        End; {with}
      End;
{-----}
Function UNIT_MATRIX( K:BYTE):Matrix;
VAR I,J:BYTE;
BEGIN SetLength(Result,Motor.ms+Motor.mr+1,Motor.ms+Motor.mr+1);
  FOR I:=1 TO K DO
    FOR J:=1 TO K DO IF I=J THEN Result[I][J]:=1 ELSE Result[I][J]:=0
  END { UNIT_MATRIX };
FUNCTION RRMA( L1,L2:INTEGER; A,B:Matrix):Matrix;
CONST EPS=1E-300;
VAR I,J,K:INTEGER; S1,S,T:Tfloat;
BEGIN SetLength(Result,Motor.ms+Motor.mr+1,Motor.ms+Motor.mr+1);
  FOR I:=1 TO L1 DO
    BEGIN
      S:=0;
      FOR J:=1 TO L1 DO S:=S+A[I,J]*A[I,J];
      S1:=0.72134752045*LN(S)+1.0;
      IF (S<0.25) OR (S>1.0) THEN
        BEGIN
          IF S1>=0 THEN S:=EXP(-TRUNC(S1)*LN(2.0))
          ELSE IF S1-TRUNC(S1)=0.0 THEN S:=EXP(-TRUNC(S1)*LN(2.0))
          ELSE S:=EXP(-TRUNC(S1-1)*LN(2.0));
          FOR J:=1 TO L1 DO A[I,J]:=S*A[I,J];
          FOR J:=1 TO L2 DO B[I,J]:=S*B[I,J]
        END;
    END;
```

```

END{I};
FOR I:=1 TO L1 DO
BEGIN
T:=ABS(A[I,I]); K:=I;
FOR J:=I+1 TO L1 DO
IF ABS(A[J,I])>T THEN
BEGIN
T:=ABS(A[J,I]);K:=J
END;
IF T<EPS THEN
BEGIN
WRITELN;
WRITELN('DET A=0');HALT
END;
IF I=K THEN T:=A[I,I]
ELSE BEGIN
FOR J:=1 TO L2 DO
BEGIN
T:=B[K,J]; B[K,J]:=B[I,J]; B[I,J]:=T
END{J};
FOR J:=L1 DOWNTTO I DO
BEGIN
T:=A[K,J]; A[K,J]:=A[I,J]; A[I,J]:=T
END{J};
END;
T:=1/T; A[I,I]:=T;
FOR J:=I+1 TO L1 DO
BEGIN
S:=-A[J,I]*T;
FOR K:=1 TO L2 DO B[J,K]:=B[J,K]+B[I,K]*S;
FOR K:=I+1 TO L1 DO A[J,K]:=A[J,K]+A[I,K]*S;
END{J};
END;
FOR K:=1 TO L2 DO
FOR I:=L1 DOWNTTO 1 DO
BEGIN
T:=B[I,K];
FOR J:=I+1 TO L1 DO T:=T-A[I,J]*Result[J][K];
Result[I][K]:=T*A[I,I]
END
END { RRMA };
Function INVERSEMATRIX( K:BYTE; A:Matrix):Matrix; VAR A1:Matrix;
BEGIN A1:=UNIT_MATRIX(K); Result:=RRMA(K,K,A,A1) END; { INVERSEMATRIX }

Function MATRIX_R(k1,k2:Integer):Matrix;
VAR i:Integer;
begin { P } SetLength(Result,k1+k2+1,k1+k2+1);
for i:=1 to k1 do Result[i][i]:=Motor.R1;
IF (Motor.CageRotor='N') THEN { coiled rotor }
for i:=1 to k2 do Result[k1+i][k1+i]:=Motor.R2;

```

## Computer simulation program for educational purposes

---

```
IF (Motor.CageRotor='Y') THEN { cage rotor }
  for i:=1 to k2 do begin Result[k1+i][k1+i]:=2*(Motor.R2+Motor.R2c); Result[k1+i][k1+Succ1(i,k2)]:=-
  Motor.R2; Result[k1+i][k1+Pred1(i,k2)]:=-Motor.R2; end;
  end; { P }
{ ----- }

PROCEDURE DIFF_EQN SOLUTION(t,Dt:Tfloat; A:Matrix; WE:VectorCol; var WY:VectorCol;var
dWz,Wm0,dQ1,dQ2:Tfloat);
{ ----- }

{      Solving of differential equations set          }
{      Dx = A * x +  F          }
{ t - initial time, Dt - time interval, WE - initial conditions   }

var i,k,intervals:integer; Ti,T1:Tfloat;
ExpAt,A_1:Matrix; F,Fo,Fk,Xo,Xk,Q:VectorCol;

Procedure TRANSITION_FUN(ts:Tfloat; A:Matrix; var ExpAt:Matrix); var k,l,N:integer; NormA1,
NormAn:Tfloat; Ar,An:Matrix;
{ matrix exp(A*ts) calculation }
function NORM(A:Matrix; L1:byte):Tfloat;      VAR I,J:BYTE; RR,S:Tfloat;
{           --           }
{   NORM(A) = max >_ |A[I,J]|    }
{           I   J           }

Begin RR:=0;
FOR I:=1 TO L1 DO
BEGIN S:=0;
FOR J:=1 TO L1 DO S:=S+ABS(A[I,J]);
IF RR<S THEN RR:=S;
END; { I } NORM:=RR;
End;

Begin { transition f. } N:=Motor.ms+Motor.mr; SetLength(ExpAt,N+1,N+1); SetLength(Ar,N+1,N+1);
for k:=1 to N do
  for l:=1 to N do      begin { 1 }
    if k=l then begin Ar[k,l]:=1; ExpAt[k,l]:=1 end
    else begin Ar[k,l]:=0; ExpAt[k,l]:=0 end; end; { 1 }
mulMat(Ar,A,N,ts,An); NormA1:=Norm(An,N); K:=0;
repeat K:=K+1; mulMat(Ar,A,N,ts/k,An); addMat(ExpAt,An,N,ExpAt); Ar:=An; NormAn:= Norm(An,N) ;
until NormAn <= 1E-5*NormA1;
End; { transition f. }

Procedure Voltages(t:Tfloat; var F: VectorCol); var k,s:integer; Umax,faza:Tfloat;
{ supply voltages }
begin { p1 } Umax:=p2*Motor.U; s:=-1; for k:=2 to Motor.p do s:=-s; SetLength(F,Motor.ms+Motor.mr+1);
  for k:=1 to Motor.ms do
begin { 1 } faza:=2*pi*(Motor.fl*t+s*(k-1)/Motor.ms);
  if ((Motor.ms=2) and (k=2)) then faza:=2*pi*(Motor.fl*t+s*0.25);
  F[k]:=Umax*sin(faza); end; { 1 }
  for k:=1+Motor.ms to Motor.ms+Motor.mr do F[k]:=0;
end; { p1 }
BEGIN { PG } with Motor do begin {record}
  dQ1:=0; dQ2:=0; dWz:=0; Wm0:=0; Dt:=abs(Dt);
  intervals:=1+Trunc(density*Dt*f1); { depends on supply frequency f1 }
```

```

T1:=Dt/intervals; Xo:=WE; mulMatCol(Mo_1,Xo,ms+mr, 1, Curr[s]);
TRANSITION_FUN(T1,A,ExpAt); A_1:=INVERSEMATRIX(Motor.ms+Motor.mr,A);
for i:=1 to intervals do      begin { i }
  Ti:=(i-0.5)*T1; Voltages(t+Ti,F); mulMatCol(A_1,F,ms+mr, -1, Xk);
  subCol(Xo,Xk,ms+mr, Q); mulMatCol(ExpAt,Q,ms+mr, 1, WY); addCol(WY,Xk,ms+mr, WY);
  Curr[pr(s)]:=Curr[s]; mulMatCol(Mo_1,WY,ms+mr, 1, Curr[s]); Xo:=WY;
for k:=1 to ms      do
  dQ1:=dQ1 + 0.25*R[k][k]*T1*sqr( Curr[s][k]+Curr[pr(s)][k] );
  for k:=1 to mr do begin { dQ2 }
    IF (CageRotor='N') THEN { coiled rotor }
      dQ2:=dQ2 + 0.25*R[ms+k][ms+k]*T1*sqr( Curr[s][ms+k]+Curr[pr(s)][ms+k] );
    IF (CageRotor='Y') THEN { cage rotor }
      dQ2:=dQ2 + 0.25*R[ms+k][ms+k]      *T1*sqr( Curr[s][ms+k]+Curr[pr(s)][ms+k] )
        + 0.25*R[ms+k][ms+Succ1(k,mr)]*T1*( Curr[s][ms+k]+Curr[pr(s)][ms+k] )*
        Curr[s][ms+Succ1(k,mr)]+Curr[pr(s)][ms+Succ1(k,mr)] )
        + 0.25*R[ms+k][ms+Pred1(k,mr)]*T1*( Curr[s][ms+k]+Curr[pr(s)][ms+k] )*
        Curr[s][ms+Pred1(k,mr)]+Curr[pr(s)][ms+Pred1(k,mr)] );
    end; { dQ2 }
    Voltages(t+(i-1)*T1, Fo); Voltages(t+i*T1, Fk);
    for k:=1 to ms      do
      dWz:=dWz + 0.25*T1*(Fo[k]+Fk[k])*( Curr[s][k]+Curr[pr(s)][k] );
    for k:=1+ms to Motor.ms+Motor.mr do
      dWz:=dWz + 0.25*T1*(Fo[k]+Fk[k])*( Curr[s][k]+Curr[pr(s)][k] );
    end; { i }
  Wm0:=0; for k:=1 to Motor.ms+Motor.mr do Wm0:= Wm0+0.5*Flux[s][k]*Curr[s][k];
  end; { record}
END; { PG }

PROCEDURE Calc_IMPULS(Mo_1,M_1:Matrix; FI:VectorCol; var Impuls:Tfloat); var k:integer; dM_1:Matrix;
dI:VectorCol;
BEGIN { PG }      Impuls:=0;
  subMat(M_1,Mo_1,Motor.ms+Motor.mr, dM_1); mulMatCol(dM_1,FI, Motor.ms+Motor.mr, 1, dI);
  for k:=1 to Motor.ms+Motor.mr do begin { 1 }
{ ! } ResVect[7+(Motor.ms+Motor.mr)+k]:=dI[k]; Impuls:=Impuls - FI[k]*dI[k];
  end; { 1 }
  Impuls:=0.5*Impuls;
END; { PG }

PROCEDURE TForm1.Initial_Calculations; var j:Integer;
BEGIN { P0 }      StatusBar1.Panels[1].Text:='START'; StatusBar1.Panels[3].Text:='START';
  StatusBar1.Update;
with Motor do begin { RECORD Motor }    NPS:=Step_Change_Number;
  FOR Ig:=1 TO NPS DO begin
    M:=MATRIX_L( 0.5*(Angle[Ig]+Angle[Ig-1])); Mp_1[Ig]{bin}:=INVERSEMATRIX(ms+mr,M);
  end;
Chart1.visible:=true; for j:=0 to 23-4 { -4 in order to not clean torque-speed curves} do Chart1.Series[j].clear;
StatusBar1.Panels[3].Text:='Calculation time = ... ?'; ChartSeriesKlossNonVisible;
StatusBar1.Panels[2].Text:=' '; CountResults:=7+3*(Motor.ms+Motor.mr);
if (Balance='Y') then begin {then} Setlength(XR,MaxIntervals,CountResults+1); Edit1.visible:=true;
MaskEdit1.visible:=true; StaticText1.visible:=true; Edit1.Update; MaskEdit1.Update; StaticText1.Update; end
{then}      else begin Setlength(XR,0); end;

```

## Computer simulation program for educational purposes

---

```
R:=MATRIX_R(Motor.ms,Motor.mr); IF Forced='M' THEN Error0:=Error_T ELSE Error0:=Error_n;
SetLength(ResVect,CountResults+1); SetLength(Curr[So],ms+mr+1); SetLength(Curr[Sk],ms+mr+1);
SetLength(Flux[So],ms+mr+1); SetLength(Flux[Sk],ms+mr+1);
end; { RECORD Motor }
END; { P0 }

Procedure TForm1.Solving_Differential_State_Eqns;
VAR j,k, Entr:INTEGER; T1,T2:TFloat;
Begin {Solving_Differential_State_Eqns}
  with Motor do begin { RECORD Motor }
    Stopped:=false; pT:=0; Entr:=0;
    NFS:=0; s:=So; N[s]:=No; N[pr(s)]:=No; Nav:=N[s]; dn:=0; Em0:=0;
    Impuls[pr(s)]:=0; Impuls[s]:=0; Lobr:=fio; Tav:=0; Ek0:=2*pi*pi*Jo*sqr(No);
    IF N[s]>0 THEN BEGIN
      Ig:=0; repeat Ig:=Ig+1 until ( (fio>=Angle[Ig-1]) and (fio<Angle[Ig]) );
      DLobr:=Angle[Ig]-fio; END;
    IF N[s]<=0 THEN BEGIN
      Ig:=0; repeat Ig:=Ig+1 until ( (fio>Angle[Ig-1]) and (fio<=Angle[Ig]) );
      DLobr:=Angle[Ig-1]-fio; END;
    t:=0; Q1:=0; Q2:=0; Wz:=0; Wmech:=0; M_1:=Mp_1[mod1(Ig,NPS)];
    for k:=1 to ms+mr do Flux[s][k]:=0; mulMatCol(M_1,Flux[s],ms+mr,1, Curr[s]);
    for k:=1 to ms+mr do Em0:= Em0+0.5*Flux[s][k]*Curr[s][k];
    for k:=0 to MaxIntervals do Npr[k]:=0; wsk1:=0; wsk2:=0; WSK:='1';
    for k:=0 to MaxIntervals do Mom1[k]:=0; for k:=0 to MaxIntervals do Mom2[k]:=0;
    ResVect[1]:=t; ResVect[2]:=Lobr; ResVect[3]:=No; ResVect[4]:=dn; ResVect[5]:=Impuls[s];ResVect[6]:=Tav;
    ResVect[7]:=Nav;
    for k:=1 to ms+mr do begin ResVect[7+k]:=Curr[s][k]; ResVect[7+(Motor.ms+Motor.mr)+k]:=0;
    ResVect[7+2*(Motor.ms+Motor.mr)+k]:=Flux[s][k]; end; if (Store='Y') then for k:=1 to CountResults do
    XR[0][k]:=ResVect[k];
    REPEAT {----- Solving_Differential_State_Eqns -- BEGIN -----}
      s:=pr(s); NFS:=NFS+1;
      IF Forced='M' THEN Begin { M }
        W:= sqr(N[pr(s)]) - {sgn(N[pr(s)])}*{Tm*DLobr/(pi*Jo)};
        if W > 0 then N[s]:= sgn(N[pr(s)])*sqrt(W) else begin N[s]:=0; end;
        End; { M }
      IF Forced='P' THEN Begin { P } N[s]:=N[pr(s)] End; { P }
      dn:=N[s]-N[pr(s)]; if abs(N[pr(s)]+0.5*dn)=0 then begin Stopped:=true; dTime[s]:=dTime[pr(s)]; end
      else dTime[s]:=abs( DLobr/(N[pr(s)]+0.5*dn) );
      Mo_1:=M_1; Ig:=Ig+sgn(N[s]); Ig:=mod1(Ig,NPS); M_1:=Mp_1[Ig]; mulMat(R,Mo_1,Motor.ms+Motor.mr,-1,A);
      DIFF_EQN SOLUTION(t,dTime[s],A,Flux[pr(s)], Flux[s], dWz,Em0,dQ1,dQ2);
      Lobr:=Lobr+DLobr; t:=t+dTime[s]; Q1:=Q1+dQ1; Q2:=Q2+dQ2; Q:=Q1+Q2; Wz:=Wz+dWz;
      mulMatCol(M_1,Flux[s],ms+mr,1, Curr[s]);
      Em:=0; for k:=1 to Motor.ms+Motor.mr do Em:= Em+0.5*Flux[s][k]*Curr[s][k];
      Calc_Impuls(Mo_1,M_1,Flux[s],Impuls[s]);
      // currents changes and Impuls[s] are saved in ResVect[7+(Motor.ms+Motor.mr)+k]
      IF Forced='M' THEN Begin { M } W:=sqr(N[s])+Impuls[s]/(2*pi*pi*Jo);
      if W > 0 then begin N[s]:=sgn(N[s])*sqrt(W); end
      else begin Ig:=Ig-sgn(N[s]); N[s]:=-N[pr(s)]; Wz:=Wz-Impuls[s];
      { Enegry Impuls is consumed by source } end;
      End; { M }
      DLobr:= sgn(N[s])*(Angle[Ig]-Angle[Ig-1]);
```

```

IF Forced='P' THEN Begin { P } N[s]:=N[pr(s)]           End; { P }

IF Forced='M' THEN BEGIN { M } Wmech:=Wmech-2*pi*DLoBr*Tm;      END; { M }
IF Forced='P' THEN BEGIN { P } Wmech:=Wmech-2*pi*DLoBr*Tav {approx.} END; { P }

IF Forced='M' THEN BEGIN { M } Ek:=2*pi*pi*Jo*sqr(N[s])      END; { M }
IF Forced='P' THEN BEGIN { P } Ek:=Ek0                      END; { P }

if WSK='1' then begin { 1 }
    wsk1:=wsk1+1;  Mom1[wsk1-1]:=sgn(N[s])*Impuls[s];
    MomL:=0; for k:=0 to NPS-1 do MomL:=MomL+Mom1[k];
    if wsk1=NPS then begin wsk2:=0; WSK:='2'; end;
        end { 1 }
    else begin { 2 }
        wsk2:=wsk2+1;  Mom2[wsk2-1]:=sgn(N[s])*Impuls[s];
        MomL:=0; for k:=0 to NPS-1 do MomL:=MomL+Mom2[k];
        if wsk2=NPS then begin wsk1:=0; WSK:='1'; end;
            end; { 2 }
    MomTav[(NFS-1) mod NPS]:=sgn(N[s])*Impuls[s]; Tav:=0; for k:=0 to NPS-1 do Tav:=Tav+MomTav[k mod NPS];
    Tav:=0.5*Tav/pi;
    Npr[(NFS-1) mod NPS]:=N[s]-0.5*dn;          Nav:=0; for k:=0 to NPS-1 do Nav:=Nav+Npr[k mod NPS];
    Nav:=Nav/NPS;
    Error:=0; T1:=0; T2:=0;
    IF Forced='M' then for k:=0 to NPS-1 do Error:=Error+abs((Mom1[k]-Mom2[k])); Error:=0.5*Error/pi;
    IF Forced='P' then for k:=0 to NPS-1 do
        begin T1:=T1+Mom1[k]; T2:=T2+Mom2[k]; if not(T2=0) then Error:=abs((T1-T2)/T2) else Error:=10 end;

    IF ( (Error<1.5*Error0) and (Entr=0) ) THEN begin Tav1:=Tav; Tav2:=Tav; Entr:=1 end;
    IF Error<1.5*Error0 THEN begin if Tav<Tav1 then Tav1:=Tav; if Tav>Tav2 then Tav2:=Tav; end;
    if (Store='Y') then begin {store}
        ResVect[1]:=t; ResVect[2]:=LoBr; ResVect[3]:=N[s]; ResVect[4]:=dn; ResVect[5]:=Impuls[s]; ResVect[6]:=Tav;
        ResVect[7]:=Nav;
        for k:=1 to Motor.ms+Motor.mr do begin ResVect[7+k]:=Curr[s][k];
        ResVect[7+2*(Motor.ms+Motor.mr)+k]:=Flux[s][k]; end;
        for k:=1 to CountResults do XR[NFS][k]:=ResVect[k];
        StatusBar1.Panels[1].Text:=IntToStr(NFS); if frac(0.005*NFS)=0 then begin StatusBar1.Update; end;
            end; {store}
        if (Balance='Y') then begin {balance}
            if frac(0.002*NFS)=0 then begin {energy balance}
                Edit1.Text:=' '+FloatToStrF(Wz+Wmech,ffExponent,3,2)+' [J] / '+FloatToStrF(Q+(Ek-Ek0)+(Em-Em0),ffExponent,3,2)+' [J]';
                MaskEdit1.Text:='      Error = '+FloatToStrF(100*(Wz+Wmech-Q-(Em-Em0)-(Ek-Ek0))/(Wz+Wmech),ffGeneral,3,2)+' %';
                MaskEdit1.Update; Edit1.Update; end; {energy balance}
            end; {balance}
        UNTIL ( (Stopped=true) or (NFS=MaxIntervals-1) (***) or (t>TC) );
{----- Solving_Differential_State_Eqns -- THE END -----}
        if (Balance='Y') then begin {balance}
            Edit1.Text:=' '+FloatToStrF(Wz+Wmech,ffExponent,3,2)+' [J] / '+FloatToStrF(Q+(Em-Em0)+(Ek-Ek0),ffExponent,3,2)+' [J]';

```

## Computer simulation program for educational purposes

---

```
MaskEdit1.Text:='      Error = '+FloatToStrF(100*(Wz+Wmech-Q-(Em-Em0)-(Ek-Ek0))/(Wz+Wmech),ffGeneral,3,2)+' %';
MaskEdit1.Update; Edit1.Update;
end; {balance}
if Stopped=true then StatusBar1.Panels[2].Text:='The motor has been stopped';
end; { RECORD Motor }
End; { Solving_Differential_State_Eqns }

Procedure TForm1.Number_of_IntervalClick(Sender: TObject);
begin {Data_Load;} UpDownsNonVisible; Motor.NPS:=Step_Change_Number; with Num_of_Interv_Grid do
begin visible:=true; Cells[0,0]:='  Number of steps Q ='; Cells[0,1]:=          '+FloatToStr(Motor.NPS); end;
end;

PROCEDURE TForm1.ForcedTorque1Click(Sender: TObject); var T0:TTime;
begin CheckBoxesNonChecked; CheckBoxesVisible; ChartSeriesVisible; Chart1.Series[24].clear;
RG1.visible:=false; Rb2.visible:=false; Rb1.visible:=false; T0:=Time; Determin_T_n.visible:=false;
String_T_n_Curve.visible:=false; UpDownsNonVisible; {Data_Load;} UpDownsNonVisible; Forced:='M';
Balance:='Y'; Store:='Y';
Initial_Calculations; Solving_Differential_State_Eqns; Num_of_Interv_Grid.visible:=false;
CheckBox2.checked:=true; StatusBar1.Panels[3].Text:='Calculation time = '+TimeToStr(Time-T0);
Determin_T_n.Caption:='Show Torque v. Speed Curve'; end;

PROCEDURE TForm1.ForcedSpeed1Click(Sender: TObject); var T0:TTime;
begin CheckBoxesNonChecked; CheckBoxesVisible; ChartSeriesVisible; RG1.visible:=false; Rb2.visible:=false;
Rb1.visible:=false; Chart1.Series[24].clear; T0:=Time; Determin_T_n.visible:=false;
String_T_n_Curve.visible:=false; {Data_Load;} UpDownsNonVisible; Forced:='P'; Balance:='Y'; Store:='Y';
Initial_Calculations; Solving_Differential_State_Eqns;
Num_of_Interv_Grid.visible:=false; CheckBox17.checked:=true; CheckBox19.visible:=false;
StatusBar1.Panels[3].Text:='Calculation time = '+TimeToStr(Time-T0); Determin_T_n.Caption:='Show Torque v.
Speed Curve'; end;

PROCEDURE TForm1.TorqueSpeedCurve1Click(Sender: TObject); var ns1,ns2:String;
begin String_T_n_Curve.visible:=true; RG1.visible:=false; Rb2.visible:=false; Rb1.visible:=false; {Data_Load;}
UpDownsNonVisible; ns1:=FloatToStr(-Motor.f1/Motor.p); ns2:=FloatToStr(Motor.f1/Motor.p);
with String_T_n_Curve do begin Cells[0,0]:='N1 [rps]';Cells[1,0]:='N2 [rps]'; Cells[2,0]:='Points';
Cells[0,1]:=ns1; Cells[1,1]:=ns2; Cells[2,1]:='100'; end;
Edit1.visible:=false; MaskEdit1.visible:=false; StaticText1.visible:=false; Chart1.Series[24].clear;
CheckBoxesNonVisible; Num_of_Interv_Grid.visible:=false; Determin_T_n.visible:=true; Determin_T_n.Update;
Balance:='N'; Store:='N';
end;

PROCEDURE TForm1.Determin_T_nClick(Sender: TObject);
const eps=1e-4; var No,No60,N1,N2,pN,ns,s1,s5,s7,sk,nk,M1,M5,M7,Mk:TFloat; m,Iprz:Integer; T0:TTime;
Mh:array of TFloat;
BEGIN T0:=Time; {Data_Load;} UpDownsNonVisible; ns:=Motor.f1/Motor.p; CheckBoxesNonChecked;
RG1.visible:=false; Rb2.visible:=false; Rb1.visible:=false;
IF Determin_T_n.Caption='Show Torque v. Speed Curve' THEN Begin {if then}
Determin_T_n.Caption:='Calc. Torque v. Speed Curve'; Determin_T_n.Update;
ChartSeries2NonVisible; ChartSeriesNonVisible; Chart1.Series[20].active:=true; Chart1.Series[23].active:=true;
for m:=0 to Chart1.Series[20].Count-1 do if not(KLOSS[m].n=0) then begin
Chart1.Series[20].AddXY(KLOSS[m].n,KLOSS[m].T); Chart1.Update; end;
```

```

        End {if then}
        ELSE Begin {if else}
Forced:='P'; Store:='N'; No:=Motor.No; No60:=Motor.No60; Initial_Calculations; Determin_T_n.Caption:='Show
Torque v. Speed Curve'; Determin_T_n.Update;
with String_T_n_Curve do begin N1:=StrToFloat(Cells[0,1]);
N2:=StrToFloat(Cells[1,1]);Iprz:=StrToInt(Cells[2,1]); end;
pN:=(N2-N1)/Iprz; Chart1.Title.visible:=true; Chart1.Foot.visible:=true; Chart1.Series[20].clear;
Chart1.Series[21].clear; Chart1.Series[22].clear; Chart1.Series[23].clear; CheckBoxesNonChecked;
ChartSeriesVisible; with Chart1.Foot.Text do begin clear; Add('Rotor speed [rps]') end; with Chart1.Title.Text do
begin clear; Add('Average electromagnetic torque Tav [Nm]') end;
for m:=0 to Iprz do begin StatusBar1.Panels[2].Text:=' '+IntToStr(Iprz-m); StatusBar1.Update;
Motor.No:=N1+m*pN; if abs(Motor.No)<eps*ns then begin KLOSS[m].n:=Motor.No; KLOSS[m].T:=Tav;
KLOSS[m].T1:=Tav1; KLOSS[m].T2:=Tav2 end else begin KLOSS[m].n:=Motor.No;
Solving_Differential_State_Eqns; KLOSS[m].T:=Tav; KLOSS[m].T1:=Tav1; KLOSS[m].T2:=Tav2; end;
{ KLOSS } Chart1.Series[20].AddXY(KLOSS[m].n,KLOSS[m].T ); Chart1.Update;
// Chart1.Series[21].AddXY(KLOSS[m].n,KLOSS[m].T1);
Chart1.Series[22].AddXY(KLOSS[m].n,KLOSS[m].T2);
end;
{*} Mk:=0; for m:=0 to Iprz do begin {KLOSS} if (Kloss[m].T>Mk) then begin {if} Mk:=KLOSS[m].T;
nk:=KLOSS[m].n; end; {if} end; {KLOSS}
{*} sk:=(ns-nk)/ns; SetLength(Mh,Iprz+1);
{*} for m:=0 to Iprz do begin {K} s1:=(ns-KLOSS[m].n)/ns; s5:=((-ns/5)-KLOSS[m].n)/(-ns/5); s7:=((ns/7)-
KLOSS[m].n)/(ns/7);
if (abs(s1)<eps) then M1:=0 else M1:=2*Mk/(s1/sk+sk/s1); if (abs(s5)<eps) then M5:=0 else
M5:=2*Mk/(s5/sk+sk/s5)/25;
if (abs(s7)<eps) then M7:=0 else M7:=2*Mk/(s7/sk+sk/s7)/49; Mh[m]:=M1-M5+M7;
Chart1.Series[23].AddXY(KLOSS[m].n,Mh[m]);
Motor.No:=No; Motor.No60:=No60;
end; {K}
End; {if else}
StatusBar1.Panels[3].Text:='Calculation time = '+TimeToStr(Time-T0);
END;

PROCEDURE TForm1.Torque_Angle_Curve1Click(Sender: TObject);
const L=200; MD:array[0..45] of Integer =
(2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,107,109,113,127,131,137,139,1
49,151,157,163,167,173,179,181,191,193,197,199); R=Length(MD);
var T0:TTime; i,j,G,GCD,kg:Integer; tau,Ws,Wr:TFloat; ExpQs,ExpQr,Exp: array[0..R] of Integer;
FIMat,TsynchMat: array [0..L] of TFloat;
BEGIN CheckBoxesNonChecked; CheckBoxesNonVisible; ChartSeriesVisible; Edit1.visible:=false;
MaskEdit1.visible:=false; StaticText1.visible:=false; Chart1.Series[24].clear; Determin_T_n.visible:=false;
RG1.visible:=true; Rb2.visible:=true; Rb1.visible:=true;
IF ( (Rb1.checked=true) or (Rb2.checked=true) ) then BEGIN {IF}
RG1.visible:=false; Rb2.visible:=false; Rb1.visible:=false;
String_T_n_Curve.visible:=false; {Data_Load;}UpDownsNonVisible; Forced:='P'; Balance:='N'; Store:='N';
Initial_Calculations;
Chart1.Series[24].clear; with Chart1.Title.Text do begin clear; Add('Reluctance torque [Nm]') end;
if Rb1.checked=true then begin {Rb1} tau:=1/(Motor.Qr*L); kg:=Motor.Qr;
with Chart1.Foot.Text do begin clear; Add(' * (2*pi / Qr) = rotor angle
Qs='+IntToStr(Motor.Qs)'+', Qr='+IntToStr(Motor.Qr)) end;

```

```

        end; {Rb1}
    if Rb2.checked=true then begin {Rb2}
(* Greatest Common Divisor for Qs and Qr *)
    for i:=0 to R do begin ExpQs[i]:=0; ExpQr[i]:=0; Ws:=Motor.Qs; Wr:=Motor.Qr;
    while (Frac(Ws/MD[i])=0) do begin ExpQs[i]:=ExpQs[i]+1; Ws:=Ws/MD[i]; end;
    while (Frac(Wr/MD[i])=0) do begin ExpQr[i]:=ExpQr[i]+1; Wr:=Wr/MD[i]; end;
    Exp[i]:=ExpQs[i]; if (ExpQr[i]<ExpQs[i]) then Exp[i]:=ExpQr[i];
    end;
    GCD:=1; for i:=0 to R do begin G:=1;
    for j:=1 to Exp[i] do G:=G*MD[i]; if (Exp[i]>0) then GCD:=GCD*G;
(* Greatest Common Divisor *) end; tau:=1/(GCD*L); kg:=GCD;
    with Chart1.Foot.Text do begin clear; Add('
                                         * (2*pi / GCD) = rotor angle
Qs='+IntToStr(Motor.Qs)+', Qr='+IntToStr(Motor.Qr)+',
GCD('+IntToStr(Motor.Qs)+','+IntToStr(Motor.Qr)+')='+IntToStr(GCD)) end;
    end; {Rb2}

T0:=Time;
for i:=0 to L do begin
    fio:=i*tau; Solving_Differential_State_Eqns; FiMat[i]:=fio; TsynchMat[i]:=Tav;
Chart1.Series[24].AddXY(FiMat[i]*kg,TsynchMat[i],"Chart1.Series[24].SeriesColor");if frac(0.1*i)=0 then begin
Chart1.Repaint; Chart1.Update; end;
    StatusBar1.Panels[2].Text:=' '+IntToStr(L-i); StatusBar1.Repaint; end;
    StatusBar1.Panels[3].Text:='Calculation time = '+TimeToStr(Time-T0);
    END; {IF}
    Rb1.checked:=false; Rb2.checked:=false;
END;

Procedure TForm1.Curve1(S: TObject); var i:LongInt;
begin Chart1.Series[1].clear; if CheckBox1.checked=true then begin with Chart1.Foot.Text do begin clear;
Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Rotor angle round [-]') end;
CheckBox19.checked:=false; for i:=0 to NFS do
{Lobr} Chart1.Series[1].AddXY(XR[i][1],XR[i][2],"Chart1.Series[1].SeriesColor) end
else Chart1.Title.Text.clear; end;
Procedure TForm1.Curve2(S: TObject); var i:LongInt;
begin Chart1.Series[2].clear; if CheckBox2.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Revolutions per
second [rps]') end;
    Chart1.Series[2].AddXY(XR[0][1],XR[0][3]+XR[0][4],"Chart1.Series[2].SeriesColor);
    Chart1.Series[2].AddXY(XR[0][1],XR[0][3]+0      ,"Chart1.Series[2].SeriesColor);
    for i:=1 to NFS do begin
{n} Chart1.Series[2].AddXY(XR[i][1],XR[i-1][3]+XR[i][4],"Chart1.Series[2].SeriesColor);
    Chart1.Series[2].AddXY(XR[i][1],XR[i][3]+0      ,"Chart1.Series[2].SeriesColor) end; end
else Chart1.Title.Text.clear;
end;
Procedure TForm1.Curve3(S: TObject); var i:LongInt;
begin Chart1.Series[3].clear; if CheckBox3.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Change of rps
[rps]') end; for i:=0 to NFS do
{dn} Chart1.Series[3].AddXY(XR[i][1],XR[i][4],"Chart1.Series[3].SeriesColor) end
else Chart1.Title.Text.clear;
end;
Procedure TForm1.Curve4(S: TObject); var i:LongInt;

```

```

begin Chart1.Series[4].clear; if CheckBox4.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Torue impulses
[Nm]') end; for i:=0 to NFS do
  {ak} Chart1.Series[4].AddXY(XR[i][1],XR[i][5],"Chart1.Series[4].SeriesColor) end
    else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveTav(S: TObject); var i:LongInt;
begin Chart1.Series[5].clear; if CheckBox17.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Average Torque
Tav [Nm]') end; for i:=0 to NFS do
  {Tav} Chart1.Series[5].AddXY(XR[i][1],XR[i][6],"Chart1.Series[5].SeriesColor); end else
Chart1.Title.Text.clear; end;
Procedure TForm1.CurveNav(S: TObject); var i:LongInt;
begin Chart1.Series[6].clear; if CheckBox18.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Average Speed
Nav [rps] ') end; for i:=0 to NFS do
  {Nav} Chart1.Series[6].AddXY(XR[i][1],XR[i][7],"Chart1.Series[6].SeriesColor); end else
Chart1.Title.Text.clear; end;
Procedure TForm1.CurveTavNav(S: TObject); var i:LongInt;
begin Chart1.Series[7].clear; if CheckBox19.checked=true then begin CheckBoxesNonChecked; with
Chart1.Foot.Text do begin clear; Add('speed [rps]') end; with Chart1.Title.Text do begin clear; Add('Transient
Torque v. Speed Curve [Nm]') end; for i:=0 to NFS do
  {Mav=f(Nav)} Chart1.Series[7].AddXY(XR[i][7],XR[i][6],"Chart1.Series[7].SeriesColor); end
    else begin Chart1.Foot.Text.clear; Chart1.Title.Text.clear; end;
end;
{ --- THE CURVES FOR CURRENTS AND FLUXES ARE FOR THE THREE FIRST PHASES only !!!!!!!----}

Procedure TForm1.CurveIs1(S: TObject); var i:LongInt;
begin Chart1.Series[8].clear; if CheckBox5.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Stator current
Is1 [A]') end; for i:=0 to NFS do begin
  {Is1} Chart1.Series[8].AddXY(XR[i][1],XR[i][8]-XR[i][7+(Motor.ms+Motor.mr)+1],"Chart1.Series[8].SeriesColor);
Chart1.Series[8].SeriesColor);
    Chart1.Series[8].AddXY(XR[i][1],XR[i][8]      ,"Chart1.Series[8].SeriesColor) end; end
      else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveIs2(S: TObject); var i:LongInt;
begin Chart1.Series[9].clear; if CheckBox6.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Stator current
Is2 [A]') end; for i:=0 to NFS do begin
  {Is2} Chart1.Series[9].AddXY(XR[i][1],XR[i][9]-
XR[i][7+(Motor.ms+Motor.mr)+2],"Chart1.Series[9].SeriesColor);
    Chart1.Series[9].AddXY(XR[i][1],XR[i][9]      ,"Chart1.Series[9].SeriesColor);end; end
      else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveIs3(S: TObject); var i:LongInt;
begin Chart1.Series[10].clear; if CheckBox7.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Stator current
Is3 [A]') end; for i:=0 to NFS do begin
  {Is3} Chart1.Series[10].AddXY(XR[i][1],XR[i][10]-XR[i][7+(Motor.ms+Motor.mr)+3],"Chart1.Series[10].SeriesColor);
Chart1.Series[10].SeriesColor); Chart1.Series[10].AddXY(XR[i][1],XR[i][10]      ,"Chart1.Series[10].SeriesColor);
    Chart1.Series[10].SeriesColor);
end; end
      else Chart1.Title.Text.clear; end;

```

```
Procedure TForm1.CurveIr1(S: TObject); var i:LongInt;
begin Chart1.Series[11].clear; if CheckBox8.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Rotor current Ir1
[A]') end; for i:=0 to NFS do begin
{Ir1} Chart1.Series[11].AddXY(XR[i][1],XR[i][7+Motor.ms+1]-
XR[i][7+2*Motor.ms+Motor.mr+1],"Chart1.Series[11].SeriesColor");
Chart1.Series[11].AddXY(XR[i][1],XR[i][7+Motor.ms+1],"Chart1.Series[11].SeriesColor");end; end
else Chart1.Title.Text.clear; end;(**)
Procedure TForm1.CurveIr2(S: TObject); var i:LongInt;
begin Chart1.Series[12].clear; if CheckBox9.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Rotor current Ir2
[A]') end; for i:=0 to NFS do begin
{Ir2} Chart1.Series[12].AddXY(XR[i][1],XR[i][7+Motor.ms+2]-
XR[i][7+2*Motor.ms+Motor.mr+2],"Chart1.Series[12].SeriesColor");
Chart1.Series[12].AddXY(XR[i][1],XR[i][7+Motor.ms+2] ,",Chart1.Series[12].SeriesColor");end; end
else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveIr3(S: TObject); var i:LongInt;
begin Chart1.Series[13].clear; if CheckBox10.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Rotor current Ir3
[A]') end; for i:=0 to NFS do begin
{Ir3} Chart1.Series[13].AddXY(XR[i][1],XR[i][7+Motor.ms+3]-
XR[i][7+2*Motor.ms+Motor.mr+3],"Chart1.Series[13].SeriesColor");
Chart1.Series[13].AddXY(XR[i][1],XR[i][7+Motor.ms+3] ,",Chart1.Series[13].SeriesColor");end; end
else Chart1.Title.Text.clear; end;

PROCEDURE TForm1.CurveFis1(S: TObject); var i:LongInt;
begin Chart1.Series[14].clear; if CheckBox11.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Stator magnetic
flux Fis1 [Wb]') end; for i:=0 to NFS do
{Fis1} Chart1.Series[14].AddXY(XR[i][1],XR[i][7+2*(Motor.ms+Motor.mr)+1],"Chart1.Series[14].SeriesColor")
end
else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveFis2(S: TObject); var i:LongInt;
begin Chart1.Series[15].clear; if CheckBox12.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Stator magnetic
flux Fis2 [Wb]') end; for i:=0 to NFS do
{Fis2} Chart1.Series[15].AddXY(XR[i][1],XR[i][7+2*(Motor.ms+Motor.mr)+2],"Chart1.Series[15].SeriesColor")
end
else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveFis3(S: TObject); var i:LongInt;
begin Chart1.Series[16].clear; if CheckBox13.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Stator magnetic
flux Fis3 [Wb]') end; for i:=0 to NFS do
{Fis3} Chart1.Series[16].AddXY(XR[i][1],XR[i][7+2*(Motor.ms+Motor.mr)+3],"Chart1.Series[16].SeriesColor")
end
else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveFir1(S: TObject); var i:LongInt;
begin Chart1.Series[17].clear; if CheckBox14.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Rotor magnetic
flux Fir1 [Wb]') end; for i:=0 to NFS do
```

```

{Fir1}
Chart1.Series[17].AddXY(XR[i][1],XR[i][7+2*(Motor.ms+Motor.mr)+Motor.ms+1],"Chart1.Series[17].SeriesCol
or) end
else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveFir2(S: TObject); var i:LongInt;
begin Chart1.Series[18].clear; if CheckBox15.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Rotor magnetic
flux Fir2 [Wb]') end; for i:=0 to NFS do
{Fir2}
Chart1.Series[18].AddXY(XR[i][1],XR[i][7+2*(Motor.ms+Motor.mr)+Motor.ms+2],"Chart1.Series[18].SeriesCol
or) end else Chart1.Title.Text.clear; end;
Procedure TForm1.CurveFir3(S: TObject); var i:LongInt;
begin Chart1.Series[19].clear; if CheckBox16.checked=true then begin CheckBox19.checked:=false; with
Chart1.Foot.Text do begin clear; Add('time [s]') end; with Chart1.Title.Text do begin clear; Add('Rotor magnetic
flux Fis3 [Wb]') end; for i:=0 to NFS do
{Fir3}
Chart1.Series[19].AddXY(XR[i][1],XR[i][7+2*(Motor.ms+Motor.mr)+Motor.ms+3],"Chart1.Series[19].SeriesCol
or) end else Chart1.Title.Text.clear; end;
Procedure TForm1.CheckBoxesNonVisible;
begin CheckBox1.visible:=false;CheckBox2.visible:=false;CheckBox3.visible:=false;CheckBox4.visible:=false;
CheckBox5.visible:=false; CheckBox6.visible:=false;CheckBox7.visible:=false;
CheckBox8.visible:=false;CheckBox9.visible:=false;CheckBox10.visible:=false;CheckBox11.visible:=false;
CheckBox12.visible:=false;CheckBox13.visible:=false;CheckBox14.visible:=false;CheckBox15.visible:=false;
CheckBox16.visible:=false;CheckBox17.visible:=false; CheckBox18.visible:=false; CheckBox19.visible:=false;
end;
Procedure TForm1.CheckBoxesVisible;
begin CheckBox1.visible:=true;CheckBox2.visible:=true;CheckBox3.visible:=true;CheckBox4.visible:=true;
CheckBox5.visible:=true;CheckBox6.visible:=true;CheckBox7.visible:=true;

CheckBox8.visible:=true;CheckBox9.visible:=true;CheckBox10.visible:=true;CheckBox11.visible:=true;CheckBox
12.visible:=true;CheckBox13.visible:=true;CheckBox14.visible:=true;CheckBox15.visible:=true;CheckBox16.visi
ble:=true;CheckBox17.visible:=true; CheckBox18.visible:=true; CheckBox19.visible:=true; end;
Procedure TForm1.CheckBoxesNonChecked; // except the curve no. 19 and further curves
begin CheckBox1.checked:=false;CheckBox2.checked:=false;CheckBox3.checked:=false;
CheckBox4.checked:=false; CheckBox5.checked:=false;CheckBox6.checked:=false;CheckBox7.checked:=false;
CheckBox8.checked:=false;CheckBox9.checked:=false;CheckBox10.checked:=false;CheckBox11.checked:=false;
CheckBox12.checked:=false;CheckBox13.checked:=false;CheckBox14.checked:=false;CheckBox15.checked:=
false;CheckBox16.checked:=false;CheckBox17.checked:=false; CheckBox18.checked:=false; end;
Procedure TForm1.ChartSeriesVisible; var j:integer;
begin Chart1.visible:=true; j:=chart1.SeriesCount-1; while j>=0 do begin Chart1.Series[j].active:=true; j:=Pred(j)
end; end;
Procedure TForm1.ChartSeriesNonVisible; var j:integer;
begin Chart1.visible:=true; j:=chart1.SeriesCount-1; while j>=2 do begin Chart1.Series[j].active:=false;
j:=Pred(j) end; end;
Procedure TForm1.ChartSeries2NonVisible;
begin Chart1.visible:=true; Chart1.Series[0].active:=false; Chart1.Series[1].active:=false; end;
Procedure TForm1.ChartSeriesKlossNonVisible;
begin Chart1.visible:=true; Chart1.Series[20].active:=false; Chart1.Series[21].active:=false;
Chart1.Series[22].active:=false; Chart1.Series[23].active:=false; end;
Procedure TForm1.UpDownsNonVisible;

```

## Computer simulation program for educational purposes

---

```
begin UpDown1.visible:=false; UpDown2.visible:=false; UpDown3.visible:=false; UpDown4.visible:=false;
MaskEdit5.visible:=false; MaskEdit6.visible:=false; MaskEdit7.visible:=false; MaskEdit2.visible:=false;
MaskEdit3.visible:=false; MaskEdit4.visible:=false; end;
Procedure TForm1.UpDownsVisible;
begin UpDown1.visible:=true; UpDown2.visible:=true; UpDown3.visible:=true; UpDown4.visible:=true;
MaskEdit5.visible:=true; MaskEdit2.visible:=true; MaskEdit3.visible:=true; MaskEdit4.visible:=true; end;
Procedure TForm1.Koniec(S:TObject); begin close end;
Procedure TForm1.Curve_M(S:TObject); var T0:TFloat; begin T0:=Time;
StatusBar1.Panels[3].Text:='Calculation time = ... ?'; StatusBar1.Repaint; CheckBoxesNonChecked;
CheckBox19.checked:=false; CheckBoxesNonVisible; ChartSeriesNonVisible; Chart1.visible:=true;
Chart1.Series[0].clear; Chart1.Series[1].clear; Chart1.Repaint; with Chart1.Foot.Text do begin clear; Add('angle
[deg]') end; with Chart1.Title.Text do begin clear; Add('Mutual Inductance Coefficient [mH]') end; Data_Load;
MUTUAL_INDCTANCE_CHART; StatusBar1.Panels[3].Text:='Calculation time = '+TimeToStr(Time-T0); end;
Procedure TForm1.Curve_M_fine(S:TObject); var T0:TFloat; begin T0:=Time;
StatusBar1.Panels[3].Text:='Calculation time = ... ?'; Statusbar1.Repaint; CheckBoxesNonChecked;
CheckBox19.checked:=false; CheckBoxesNonVisible; ChartSeriesNonVisible; Chart1.visible:=true;
Chart1.Series[0].clear; Chart1.Series[1].clear; Chart1.Repaint; with Chart1.Foot.Text do begin clear; Add('angle
[deg]') end; with Chart1.Title.Text do begin clear; Add('Mutual Inductance Coefficient [mH]') end; Data_Load;
MUTUAL_INDCTANCE_CHART_fine; StatusBar1.Panels[3].Text:='Calculation time = '+TimeToStr(Time-T0);
end;
Procedure TForm1.SaveCurve(S:TObject); begin SavePictureDialog1.Execute;
Chart1.SaveToMetaFile(SavePictureDialog1.Filename); end;
Procedure TForm1.SaveOwnMotorData(S:TObject); var i:integer; F1: TextFile; Str:string;
Begin SaveDialog1.FileName :='* .txt'; if SaveDialog1.Execute then begin AssignFile(F1,
SaveDialog1.Filename); Rewrite(F1);
for i:=0 to 24 do begin Writeln(F1,MotorData.Cells[i,0]); Writeln(F1,MotorData.Cells[i,1]); end;
Writeln(F1,'Qs=');Writeln(F1,Motor.Qs); Writeln(F1,'Qr=');Writeln(F1,Motor.Qr);
Writeln(F1,Motor.ks);Writeln(F1,Motor.kr);
Str:=FloatToStrF(Motor.Ys,ffFixed,18,17);Writeln(F1,Str);Str:=FloatToStrF(Motor.Yr,ffFixed,18,17);
Writeln(F1,Str); CloseFile(F1); end; End;
Procedure TForm1.ReadOwnMotorData(S:TObject); var i:integer; F1:TextFile; Str:String;
Begin OpenDialog1.FileName :='* .txt'; if OpenDialog1.Execute then begin AssignFile(F1,
OpenDialog1.Filename); Reset(F1);
for i:=0 to 24 do begin Readln(F1,Str);MotorData.Cells[i,0]:=Str; Readln(F1,Str);MotorData.Cells[i,1]:=Str;
MotorData.Repaint; end;
Readln(F1,Str);Readln(F1,Str);Motor.Qs:=StrToInt(Str);MaskEdit2.visible:=true;MaskEdit2.Text:='Qs='+Str;
Readln(F1,Str);Readln(F1,Str);Motor.Qr:=StrToInt(Str);MaskEdit3.visible:=true;MaskEdit3.Text:='Qr='+Str;
Readln(F1,Str); Motor.ks:=StrToInt(Str);Readln(F1,Str);Motor.kr:=StrToInt(Str);
Readln(F1,Str);Motor.Ys:=StrToFloat(Str);MaskEdit4.visible:=true;MaskEdit4.Text:='Ys='+Str;
Readln(F1,Str); Motor.Yr:=StrToFloat(Str);MaskEdit5.visible:=true;MaskEdit5.Text:='Yr='+Str;
CloseFile(F1); end;
MaskEdit6.visible:=true;MaskEdit7.visible:=true;UpDown1.visible:=true;UpDown2.visible:=true;
UpDown3.visible:=true; UpDown4.visible:=true;
End;
Procedure TForm1.calculationtime1Click(Sender: TObject);
begin StringGrid1.Visible:=true; BitBtn1.Visible:=true; StringGrid1.Cells[1,0]:=FloatToStr(TC);
StringGrid1.Cells[0,0]:=' analysis time [s] ='; end;
Procedure TForm1.BitBtn1Click(Sender: TObject);
begin TC:=StrToFloat(StringGrid1.Cells[1,0]); StringGrid1.Visible:=False; BitBtn1.Visible:=False; end;
END.
```

## Summary

The results of computer simulations and their comparison with the results of laboratory experiments enable us to state that the proposed non-harmonic mathematical model and related to it **step-by-step analysis** takes into account effects resulting from slotting and allows to consider the influence of the number of stator and rotor slots on the dynamic behaviour of the machine as well as on steady-state torque-speed curves.

The model is helpful in determining basic properties of reluctance torque components and particularly enables us to determine their synchronous speeds, approximate magnitudes, periods of reluctance torque v. rotor angle curves as well as magnitudes of pulsating torque at the given speeds. The information about parasitic torque resulting from slotting is hidden in the number of steps, as well as height and length of steps of the stator-rotor inductance curves.

The approximation of stator-rotor inductances based on piecewise-constant functions simplifies the mathematical model of an electrical machine, which consists of a system of linear differential equations with time-constant coefficients and some algebraic equations. This model is convenient in computer calculations for both transient and steady states.

*The book is an attempt to use "the best of both worlds": circuit-analysis and field-analysis based methods – and is a sort of compromise between accuracy and computation speed. In particular, the technique seems to cope very well with parasitic torque.*

*The book can be recommended for those looking for new method and new possibilities in theory of electromechanical conversion. It is aimed at researchers and postgraduate students wishing to gain deeper insight into non-conventional techniques used in rotating electrical machine analysis and to widen theoretical background for further studies.*

*W prezentowanej metodzie jest wykorzystywany oryginalny jednowymiarowy model permeacyjny szczeliny powietrznej. Z uwagi na odciinkowo-liniowy charakter funkcji indukcyjności wzajemnych w funkcji kąta, opisującego położenie wirnika, równania stanu można rozwiązać „przedziałami analitycznie”, co stanowi bardzo istotną zaletę metody, różniącą ją od typowych ujęć obwodowo-polowych.*

**Professor  
J. K. Sykulski  
University of  
Southampton,  
United Kingdom**

**Professor  
Y. Ishihara  
Doshisha University,  
Japan**

**Professor  
A. Demenko  
Technical University  
of Poznań,  
Poland**

I S B N      8 3 - 9 1 5 9 9 1 - 4 - 0