Eindhoven University of Technology

MASTER

Modelling and identification of single- and double-cage induction machines

Schilperoort, S.M.

Award date:
1994

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

7247

Faculteit der Elektrotechniek
Vakgroep Elektromechanica en Vermogenselektronica

Afstudeerverslag

MODELLING AND IDENTIFICATION OF SINGLE- AND DOUBLE-CAGE INDUCTION MACHINES

EMV 94-21 S.M. Schilperoort

Hoogleraar: Prof. Dr. Ir. A.J.A. Vandenput

Mentor: Dr. Ir. J.L. Duarte

Eindhoven, december 1994

De Faculteit der Elektrotechiek van de Technische Universiteit Eindhoven aanvaardt geen verantwoordelijkheid voor de inhoud van stage- en afstudeerverslagen

SUMMARY

This report aims at incorporating the influence of skin-effect into models, which are suited for numerical simulation of electrical machine dynamics. With a second-order description the skin-effect is approached. This leads to a double-cage machine model. Equations for the rotor circuit in rotor coordinates are derived, resulting in a linear rotor transfer function. The stator circuit is described in stator coordinates. Both circuits are linked by transforming current and flux from stator to rotor coordinates and vice-versa.

Another subject of study is machine parameter estimation. Two identification methods are studied: the two-flux-model (TFM) method and an Output-Error (OE) method. The TFM-method uses two independent set of equations. With the TFM-method only single-cage model parameters can be estimated using special load and test conditions.
The OE-method is based on the minimisation of the quadratic error between real output and model output. The switching harmonics of an inverter supply are used as excitation source to estimate the machine parameters. It is found that with this method it is possible to estimate double-cage machine parameters.

Both methods are applied to single and double-cage simulation models. For single-cage model estimation both methods lead to parameter values with deviation less than 1%. The OE-method is able to estimate all double-cage parameters with the same accuracy. The single-cage machine parameters generated by the TFM-method can also be used to approximate the dominant time constant of higher order machine models.

Finally, the methods are tested on a double-cage 11 kW SKA machine. When the OE-method is used, it appears that the estimated single-cage parameters are dependent of the pre-filter choice. This indicates that the chosen model order does not fit the real system (i.e. under-modelling). Applying a double-cage model results in an estimation that is independent of the chosen pre-filter.
To validate the estimated machine parameters, the simulation models are fed with the measured stator voltages and rotor speed. The estimated stator currents are compared with the measured currents. Results show that the OE-method describes the stator current more accurately than the TFM-method does.

SAMENVATTING

In dit rapport wordt ingegaan op het modelleren van het skin-effect in een dynamisch machinemodel. Voor de modellering van het skin-effect is een tweede orde benadering toegepast. Dit resulteerd in een dubbel-kooi machinemodel. Voor de model-vorming is het rotorcircuit beschreven in rotor-coördinaten waardoor een lineaire rotor overdrachts-functie ontstaat, waarin uitbreiding tot een hoger orde rotorcircuit relatief eenvoudig is. Het statorcircuit daarintegen wordt beschreven in statorcoördinaten. Beide circuits worden met elkaar gekoppeld door de flux en stroom te transformeren van rotor- naar statorcoördinaten en omgekeerd.

Een tweede studie-onderwerp in dit rapport is machine identificatie. Twee identificatie methodes zijn nader bekeken: de Twee-Flux-Model (TFM) methode en een Output-Error (OE) method. De TFM-methode gebruikt twee fluxmodellen met onafhankelijke machine-parameters. Met de TFM-methode kunnen echter alleen enkelkooi machine-parameters geschat worden. De OE-methode is gebaseerd op minimalisatie van de kwadratische fout tussen de geschatte procesuitgang en de werkelijke procesuitgang. Voor machine identificatie worden de harmonische componenten van de inverter-voeding gebruikt als excitatiebron.

Beide methoden zijn toegepast op een dynamische enkel- en dubbelkooi machinemodel. Toegepast op een enkelkooimodel resulteren beide methoden in een schatting van de enkelkooiparameters waarbij de afwijking kleiner is dan 1 %. Toegepast op een dubbel-kooi simulatiemodel resulteert de OE-methode in een schatting van alle dubbelkooi-parameters met dezelfde nauwkeurigheid. De TFM-methode kan gebruikt worden om hogere orde machine modellen te benaderen met een enkelkooi model.

Als laatste zijn de methoden getest op een 11kW SKA dubbelkooi machine. De resultaten zijn beoordeeld aan de hand van het vergelijken van gemeten en gesimuleerde stromen, waarbij de gemeten spanningen en rotorsnelheid aangeboden worden aan de geschatte simulatiemodellen. Hieruit volgt dat het geschatte dubbelkooi machinemodel bij gebruik van de OE-methode resulteerd in een betere benadering van de statorstroom dan het geschatte enkelkooi model bij gebruik van de TFM-methode.

LIST OF SYMBOLS

Machine parameters

L_h	Main air gap inductance.
L_m	Main inductance.
$L_{\sigma s}$	Leakage inductance of the stator.
$L_{\sigma k}$	Leakage inductance of the rotor.
L_σ	Leakage inductance.
$L_{\sigma 2}$	Leakage inductance of the inner cage.
σ	Relative leakage factor.
L_{12}	Mutual inductance between inner and outer cage
L_{sk}, L_{ks}	Mutual inductance between stator and rotor-cage.
R_{k0}	Per phase rotor resistance of a 3 phase machine.
R_k	Per phase cage-resistance.
R_{k1}, R_{k2}	Resistance of outer and inner cage.
R_s	Stator resistance.
k	Transformation ratio.
p	Number of pole pairs.
m_j	Number of stator/rotor phases $j = \{s, k\}$
N_j	Number of stator/rotor windings $j = \{s, k\}$
J	Momentum of inertia.
M_{el}	Electric torque.
M_{load}	Load torque.
$slip$	Slip [%].
i_s^w	Stator current component in the direct axis of the flux-reference.
i_s^b	Stator current component in the quadrature axis of the flux-reference.

Inverter parameters

U_{dc}	DC-voltage on the input of the converter.
U_{rec}	Rectified three-phase voltage.
u_{ref}	reference voltage.
u_{qN}	Voltage between phase q and negative inverter terminal.
u_{q0}	Voltage between phase q and the star-node of the machine.
u_{tri}	Triangular carrier wave.
m_i	Modulation index.
n_p	Pulse number.
ω_{tri}	Triangle-carrier wave frequency

Greek variables

φ^s	Angle with respect to the stator axis.
$\dot{\varphi}^s$	Angular frequency with respect to the stator axis.
α_s^s	Stator voltage angle with respect to the stator axis
$\dot{\alpha}_s^s, \omega_s$	Stator voltage angular frequency
ρ^s	Angle between rotor- and stator coordinates
$\dot{\rho}^s, \omega_m$	Mechanical angular frequency.
Ψ	Flux.

Identification parameters

${}_{(u)}\hat{\bar{\Psi}}_k'$	Estimated cage flux of the U/I-model.
${}_{(i)}\hat{\bar{\Psi}}_k'$	Estimated cage flux of the I / φ -model.
u	System input.
v	Coloured output noise.
$\hat{y}$	Predicted output.
ε	Output error.
D	Parameter set.
f_s	Sample frequency [Hz].
T_s	Sample time [s].
f_{nyq}	Nyquist frequency [Hz].
ω_{nyq}	Angular Nyquist frequency [rad/s].
$G_0(\omega)$	Real process.
q	Shift operator.
$G(q)$	Process model.
$H(q)$	Noise model.
$L(q)$	Low-pass pre-filter.
$B(q)$	OE numerator polynomial.
$F(q)$	OE denominator polynomial.
$b_1, b_2, \ldots b_{n_b}$	Parameters of $B(q)$.
$f_1, f_2, \ldots f_{n_f}$	Parameters of $F(q)$.
θ	Parameter vector.
ϕ	Regression vector.
N	Number of data points.
N_0	Number of samples within one fundamental period.
N_p	Number of periods in the data set.
m	Number of samples in N_p periods
Δf	Distance between two observable frequencies [Hz].
N_{noise}	Desired signal to noise ratio.
U_{offset}	Offset voltage.

Z_N	Pre-processed data set of N complex data points.
α	Stepsize (Newton-Raphson method).
J	Loss-function to be minimised.
J', J''	First and second derivative of the loss-function with respect to θ.
χ	Gradient of the prediction error.

Space vectors

$\vec{i}_k^{\#}$	Transformed equivalent rotor current.
$\vec{i}_\mu^a$	Magnetising current vector in arbitrary coordinates, $a \in \{r, s\}$
$\vec{i}_k^a$	Cage current vector in arbitrary coordinates, $a \in \{r, s\}$
$\vec{i}_{k1}^r$	Outer cage current vector in rotor coordinates
$\vec{i}_{k2}^r$	Inner cage current vector in rotor coordinates
$\vec{i}_s^a$	Stator current vector in arbitrary coordinates, $a \in \{r, s\}$
$\vec{u}_s^s$	Stator voltage in stator coordinates
$\vec{\Psi}_k^a$	Cage-flux vector in arbitrary coordinates, $a \in \{r, s\}$
$\vec{\Psi}_k^r$	Cage-flux vector in rotor coordinates
$\vec{\Psi}_{k1}^r$	Outer cage-flux vector in rotor coordinates
$\vec{\Psi}_{k2}^r$	Inner cage-flux vector in rotor coordinates
$\vec{\Psi}_h^a$	Air-gap flux vector in arbitrary coordinates, $a \in \{r, s\}$
$\vec{\Psi}_s^s$	Stator flux vector in stator coordinates

Complex variables:

I^A	Current-vector represented as complex number with stator/rotor as reference $A = \{S, R\}$.
U^A	Voltage-vector represented as complex number $A = \{S, R\}$.
Ψ^A	Flux-vector represented as complex number $A = \{S, R\}$.
Z_k	Rotor impedance.

Superscripts: X is an arbitrary variable

X^s	Representation of X in stator coordinates.
X^r	Representation of X in rotor coordinats.
X^a	Representation of X in an arbitrary axis.
X^{R1}	Real part of the complex representation, equals rotor coordinate r1.
X^{R2}	Imaginary part of the complex representation, equals rotor coordinate r2.
$\dot{X}$	Derivative of variable X.

$\hat{X}$	Estimated variable or parameter.
X'	Transformed parameter or variable.
$X^{\perp}$	Inverse polynomial.

Subscripts: X is an arbitrary variable

X_s	Referring to the stator.
X_r	Referring to the rotor.
X_0	Related to the fundamental component.
X_σ	A leakage component.
X_m	Related to the mechanical section.
$X_{k,k1,k2}$	Refering to the cage.
X_{max}	Maximum value.

Operators

p	Laplace operator.
q	Schift operator.
$\Phi_X(\omega)$	Frequency spectrum of X

Abreviations

AC	Alternating current.
A/D	Analoque to Digital conversion.
DC	Direct current.
DSP	Digital Signal Processor.
IRTF	Ideal Rotating TransFormer.
OE	Output Error.
PE	Presistently Exciting.
PWM	Pulse Width Modulation.
SKA	Special cage asynchronuous machine
SCaSM	Single-cage simulation model.
DCaSM	Double-cage simulation model.
TFM	Two-Flux-Model.

Contents

1. INTRODUCTION

For many purposes, among them analysis and control design, people want to be able to describe electric machines in an understandable way. This means we want to describe some aspects of the real machine in an abstract way. For modelling it is therefore important to select only those characteristics which are necessary and sufficient for the desired purpose. Induction machines are widely used in all sorts of electro-mechanical drive systems because of their simple and robust structure. Due to the developments of modern control techniques, using power electronics and microprocessors, induction machines have now become a good alternative to DC-machines for variable speed applications. However, these control methods require an accurate machine model and therefore an estimation of the machine parameter is necessary.

The goal in this report is to derive a model of an induction machine with significant skin-effect, which results in an accurate description of a real induction machine. In previous work many different machine models are developed; a lot of these models are based on single-cage machine properties. For dynamic modelling most of these machine models are described using one coordinate system (such as flux-coordinates or stator-coordinates). This approach always needs the standard machine equations to be transformed [Van '92]. In case of single-cage modelling this is not a real problem, but for higher order rotor-circuits like a double-cage this becomes very complicated and the equations become disorderly. In Chapter 2 a steady-state machine model will be derived, using a second order skin-effect approach, which equals a well known double-cage rotor model. This steady-state model is expanded to a dynamic double-cage model using the mechanical machine equations. However, once a machine model has been derived, it will only be useful if the parameters of this model can be found.
In Chapter 3 two identification methods are studied. One method is based on the state-space description of the single-cage induction machine. The parameters are tuned by minimising the error between two independent flux models. The second method is based on the minimisation of the error between real and estimated machine output, by means of a quadratic minimisation function. This so called output-error method is based on estimation of a linear in-output model and is therefore not directly applicable on the measured stator voltage and current, because they have no linear relation. By transforming the measured stator signals to rotor coordinates this non-linearity is cancelled out.
To be able to validate the estimation results of the two methods, the methods are tested on the single- and double-cage simulation models derived in Chapter 2. The results are discussed in Chapter 4.
The identification methods are applied to a real machine, which is discussed in Chapter 5. The used machine is a Heemaf 11kW SKA-machine. According to the measured steady-state speed/torque curve this machine has a significant skin-effect. The steady-state and dynamic behaviour of the estimated machine models is validated based on measured machine data.

2. ASYNCHRONOUS MACHINE MODELLING

2.1. INTRODUCTION

In recent years AC drives have become a good alternative to DC machines for variable speed applications. This is mainly due to the developments in power electronics. In particular asynchronous motor drives have the advantage of being robust and not having hindrance of commutators or brushes. These factors lead to a more reliable system, with less maintenance. These advantages also have a counter-side however: asynchronous motors are difficult to control. Induction motors described as a dynamic system are at least 6^{th} order systems [Velt '94], while a DC machine can be described as a 2^{nd} order system. However the use of field-oriented control for AC-machines gives us a tool to create a highly dynamic drive system. In the ideal case a field-oriented controlled induction motor becomes an easy to control DC-like machine. To create a good field-oriented control an accurate machine model is required. Accurate machine models can also be very useful to study:

- effects of the non-sinusoidal inverter supply on machine losses;
- effects of a particular mechanical load on the machine;
- new controllers.

In this chapter a transient simulation model for a double-cage/deep-bar machine is derived. First a locked-rotor equivalent circuit for a single-cage induction machine is derived. Then a locked-rotor circuit of a double-cage machine is studied. The locked-rotor circuits are then expanded to dynamic models using space-vector notation and the mechanical equations. Another approach for deriving a dynamic machine model is given by [Velt '94] using an ideal rotating transformer.

The following assumptions are made: the machine is a 3-phase asynchronous machine with one pole-pair; no saturation; both cages of the discussed double-cage have the same number of rotor bars; there is no influence of iron losses such as eddy current and hysteresis.

2.1.1. THE ASYNCHRONOUS MACHINE

All rotating electrical machines can be divided in a stationary part, the stator, and a rotating part, the rotor. Between the rotor and stator is a small air gap which is assumed to be narrow and uniform. The stationary part of the machine has three identical stator windings mechanically placed at an angle of 120°. These stator windings are considered to be sinusoidal distributed along the stator circumference.

In case of a wound-rotor the rotor construction is similar to that of the stator. Squirrel-cage rotors however are made up of two end rings and a number of rotor bars. This cage acts as a set of short-circuited windings. The symmetrical squirrel cage can therefore be seen as a small number of lumped windings assuming that the rotor material is linear and isotropic. If magnetic saturation is neglected, the rotor "windings" may be

represented as three identical, short-circuited, lumped windings, like the stator windings, positioned at an angle of 120 mechanical degrees, see Fig. 2.1.

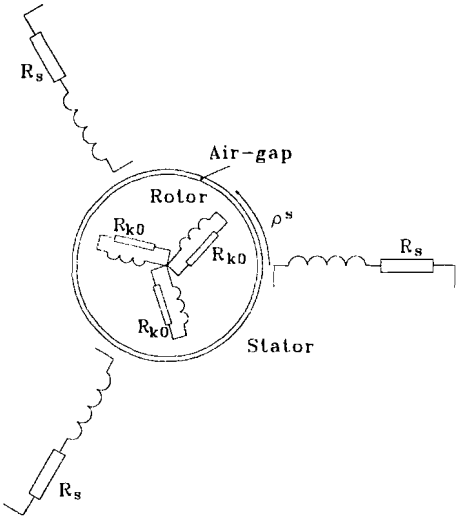


Figure 2.1: schematic layout of a 3-phase asynchronous machine.

In normal operation the rotor rotates slower than the applied stator field, which causes an induced voltage in the shorted windings. This voltage will generate a current flow through the rotor windings (bars). The magnetic field produced by the stator penetrates the rotor winding and, therefore, torque will be generated. If the rotor speed equals the rotation speed of the stator field (synchronous speed), no voltage is induced and consequently no torque is generated. The relative difference between the stator field rotation speed $\dot{\varphi}^s = 2\pi f_0$ and the mechanical rotor speed $\dot{\rho}^s$, is defined as *slip*:

$$slip = \frac{\dot{\varphi}^s - \dot{\rho}^s}{\dot{\varphi}^s} \tag{2-1}$$

For medium power induction machines the slip at rated torque is about 1...3 %.

2.1.2. SQUIRREL CAGE

To simplify the modelling, a symmetrical 3-phase machine can be described by an equivalent model with just two orthogonal windings. A 3-phase system will produce a rotating field composed of the three phase components with an angle of 120° to each other, see Fig. 2.2. The resulting vector component can be described in a two-dimensional plane without loss of information by means of rotation.

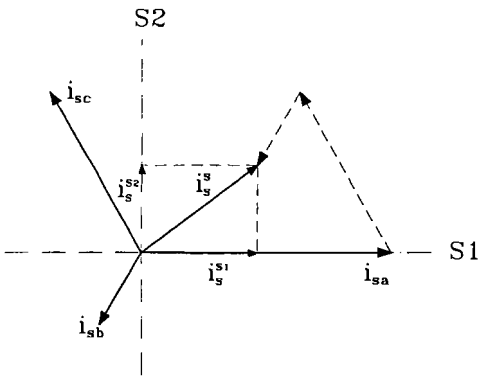


Figure 2.2: vector diagram of a three phase system

Throughout this work the two-coordinate system will be used with the stator coordinates s1 an s2 and the rotor coordinates r1 and r2. The transformation is discussed in Appendix B.

To use this so-called *semi 4-phase* model for a squirrel cage machine, it has to be treated like a wound rotor machine; the squirrel-cage machine with unequal mutual inductance's must be transformed to a wound rotor induction machine with equal mutual inductance's without losing any machine dynamics. In general the flux linkage, in case of a single-cage rotor, between stator- and rotor side can be written in space vector notation:

$$\bar{\Psi}_s = L_s \bar{i}_s + L_{ks} \bar{i}_k \tag{2-2}$$

$$\bar{\Psi}_k = L_k \bar{i}_k + L_{sk} \bar{i}_s \tag{2-3}$$

with L_s the inductance of the stator, L_k the inductance of the rotor and L_{ks}, L_{sk} the mutual inductance between rotor and stator circuit. By defining an equivalent rotor current

$$\bar{i}_k^{\#} = \bar{i}_k \frac{L_{ks}}{L_{sk}} \tag{2-4}$$

and substituting this equivalent rotor current in Eq. (2-2 and 2-3) yields:

$$\bar{\Psi}_s = L_s \bar{i}_s + L_{sk} \bar{i}_k^{\#} \tag{2-5}$$

$$\bar{\Psi}_k = L_{\sigma sk} \bar{i}_s + L_k^{\#} \bar{i}_k^{\#} \tag{2-6}$$

with :

$$L_k^{\#} = \frac{L_{sk}}{L_{ks}} L_k \tag{2-7}$$

This new rotor current $\bar{i}_k^{\#}$ is associated with a non-physical equivalent circuit with equal mutual inductances. Therefore the non-physical rotor circuit has the same construction as the stator winding. This results in the equivalent locked-rotor transformer circuit as depicted in Fig. 2.3.

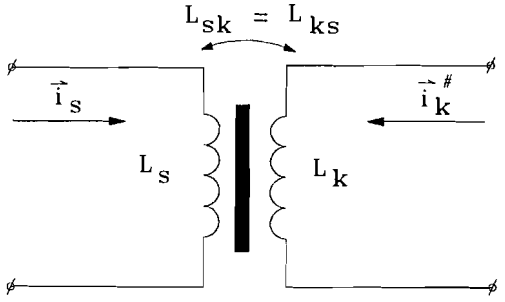


Figure 2.3: equivalent locked-rotor transformer circuit.

The mutual inductance between rotor and stator is defined as [Mach '90]:

$$L_{ij} = \frac{m_j}{2}\left(\Lambda \cdot N_i \cdot N_j\right) \qquad i,j=\{s,k\} \tag{2-8}$$

With $N_{s,k}$ is the number of stator/rotor windings and $m_{s,k}$ the number of phases. Λ is the magnetic permeability of the flux path through the machine. Thus with $m_s=3$, m_k will equal 3 and the rotor cage has changed to an equivalent three phase rotor. Both rotor and stator can be described by a three phase wound system, which enables us to use the equivalent *semi 4-phase* circuit.

2.2. LOCKED-ROTOR INDUCTION MACHINE MODELLING

2.2.1. SINGLE-CAGE

The starting point for this modelling concept is the space-vector description of an induction machine with locked-rotor:

$$\vec{\Psi}_s^s = L_h\left(\vec{i}_s^s + \vec{i}_k^s\right) + L_{\sigma s}\,\vec{i}_s^s \tag{2-9}$$

$$\vec{\Psi}_k^r = L_{\sigma k}\,\vec{i}_k^r + L_h\left(\vec{i}_s^r + \vec{i}_k^r\right) \tag{2-10}$$

$$\vec{u}_s^s = R_s\vec{i}_s^s + \dot{\vec{\Psi}}_s^s \tag{2-11}$$

$$\vec{u}_k^r = R_k\vec{i}_k^r + \dot{\vec{\Psi}}_k^r = 0 \tag{2-12}$$

In this case the rotor impedance and current are transformed to the stator side and the mutual inductance is represented by one main air-gap inductance L_h , as depicted in Fig.2.4. The current flowing trough L_h is called the magnetising current $\vec{i}_\mu$. The mutual air gap flux $\vec{\Psi}_h = L_h\vec{i}_\mu$, linked to both stator and rotor windings, is produced by this magnetising current .

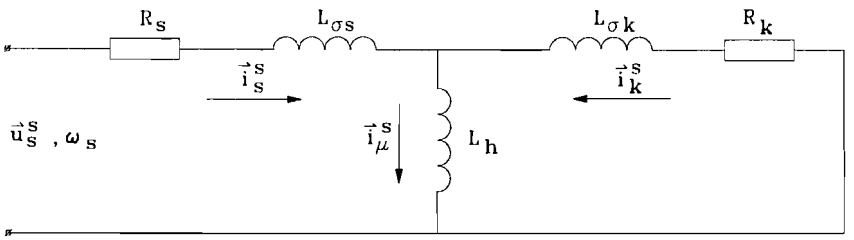


Figure 2.4: equivalent circuit for an induction machine with locked-rotor

The leakage inductances of stator and rotor windings are represented by $L_{\sigma s}$ and $L_{\sigma k}$. The stator resistance R_s and the rotor resistance R_k are the transformed values in the two-phase plane (Appendix B). It's clear that the equivalent scheme of Fig. 2.4 is analogue to a single phase transformer with a secondary resistive load.
To describe the machine with 3 inductors, as depicted in Fig. 2.4, an infinite number of different equivalent circuits can be found with the same impedance at the stator side and the same torque characteristics [Lag '92]. This is caused by the redundancy in the classical steady-state equivalent circuit. This can be seen with the aid of the Bond-graph method, in Fig. 2.5 the Bond-graph of the equivalent circuit in Fig. 2.4 is given. The bond-graph method uses a "1" for a series connection and "0" for a parallel connection, links between connections being called bonds (Appendix A).

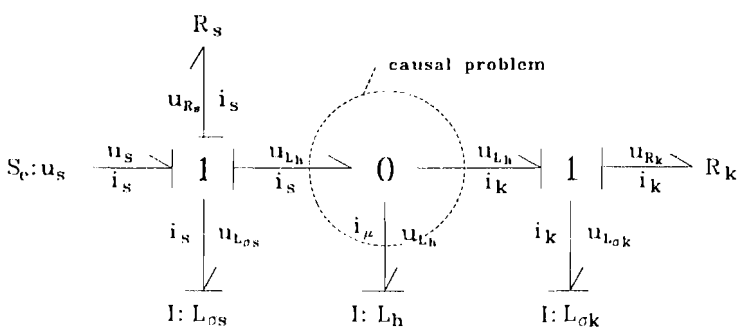


Figure 2.5: Bond-graph of equivalent circuit Figure 2.3.

This method shows that the circuit leads to a non-causal Bond-graph, because the voltage (effort) on the parallel connection can not be assigned. If no causality can be assigned to a Bond-graph, no simulation can be executed with simulation programs based on the solution of ordinary differential equations, according to [Bosch '94].

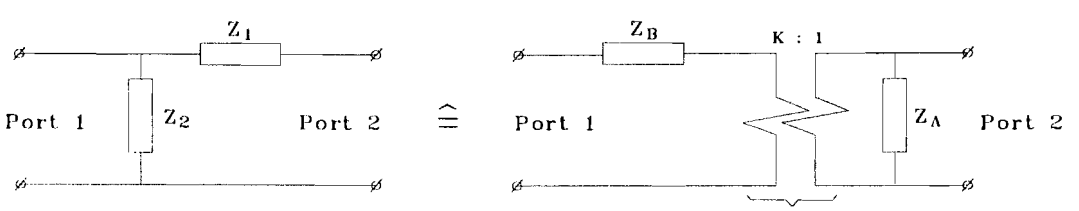


Figure 2.6: equivalent circuits with identical impedance's and transfer function

This causal problem can be solved by eliminating the redundancy in the circuit. When the current in two of the three inductors is given the third will be known through *Kirchhoff's* law. This means that the three inductors $L_{\sigma s}$, L_h and $L_{\sigma k}$ can be described by two inductors (model reduction). To reduce the model order one has different choices for taking an equivalent circuit for two of the three inductances. In Fig. 2.6 such an equivalent is given. When port 1 is open:

$$Z_{port2} = Z_1 + Z_2 = Z_A \qquad U_{port1} = U_{port2} \frac{Z_2}{Z_1 + Z_2} \tag{2-13}$$

If port 2 is shorted:

$$Z_{port1} = \frac{Z_1 \cdot Z_2}{Z_1 + Z_2} = Z_B \tag{2-14}$$

This leads to the following parameters for the equivalent circuit with Z_A and Z_B:

$$K = \frac{Z_2}{Z_1 + Z_2} \qquad Z_A = Z_1 + Z_2 \qquad Z_B = \frac{Z_1 \cdot Z_2}{Z_1 + Z_2} \tag{2-15}$$

Any circuit element connected to the secondary side of the ideal transformer can be transformed to the primary side. When applied on Fig. 2.4 with $L_{\sigma k} = Z_1$ and $L_h = Z_2$ the rotor leakage inductance $L_{\sigma k}$ can then be placed in series with the stator leakage inductance $L_{\sigma s}$. The stator and rotor leakage inductances are now replaced by one effective total leakage inductance L_σ and all leakage inductance is supposed to be concentrated in the stator. The rotor resistance seen at the primary side, is now represented by an effective rotor resistance R_k'. The resulting steady-state scheme is depicted in Fig. 2.7.

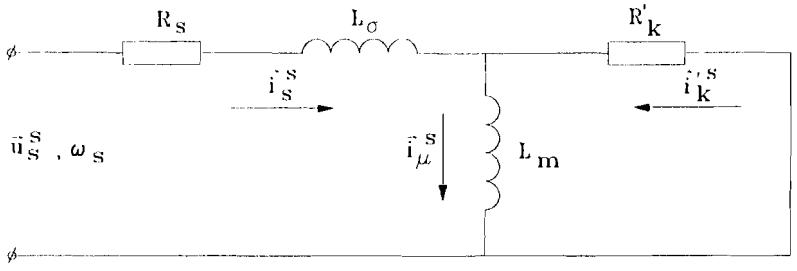


Figure 2.7: equivalent circuit with all leakage inductance concentrated in the stator.

The transformed parameters of Fig. 2.7 are related to the general locked-rotor steady-state model parameters in the following way, with the transformation factor k :

$$k = \frac{L_h}{L_h + L_{\sigma k}} = \frac{1}{1 + \sigma_k} \tag{2-16}$$

The total leakage inductance is then defined as:

$$L_\sigma = L_{\sigma s} + \frac{L_h \cdot L_{\sigma k}}{L_h + L_{\sigma k}} = L_{\sigma s} + \frac{L_{\sigma k}}{1 + \sigma_k} \tag{2-17}$$

The main inductance can be written as:

$$L_m = k^2(L_h + L_{\sigma k}) = \frac{L_h}{1+\sigma_k} \tag{2-18}$$

the rotor resistance

$$R_k' = k^2 R_k = \frac{R_k}{(1+\sigma_k)^2} \tag{2-19}$$

And the leakage factor is defined as:

$$\sigma_k = \frac{L_{\sigma k}}{L_h} \tag{2-20}$$

The stator resistance remains unchanged. The transformation results in a new set of rotor variables related to the original set by the constant *k*.

$$\vec{\Psi}_k' = k \cdot \vec{\Psi}_k \tag{2-21}$$

$$\vec{i}_k' = \frac{1}{k} \cdot \vec{i}_k \tag{2-22}$$

Substituting these new variables in Eq.(2-9, 2-10) gives:

$$\vec{\Psi}_s = (L_h + L_{\sigma s})\vec{i}_s + k L_h \vec{i}_k' \tag{2-23}$$

$$\vec{\Psi}_k' = k L_h \vec{i}_s + k^2(L_h + L_{\sigma k})\vec{i}_k' \tag{2-24}$$

Rewriting these Equations yields

$$\vec{\Psi}_s = \left(L_{\sigma s} + \frac{L_h L_{\sigma k}}{L_h + L_{\sigma k}} + \frac{L_h^2}{L_h + L_{\sigma k}} \right) \vec{i}_s + \frac{L_h^2}{L_h + L_{\sigma k}} \vec{i}_k' \tag{2-25}$$

$$\vec{\Psi}_k' = \frac{L_h^2}{L_h + L_{\sigma k}} \vec{i}_s + \frac{L_h^2}{L_h + L_{\sigma k}} \vec{i}_k' \tag{2-26}$$

Using the transformed parameters Eq.(2-16, 2-20) the following space vector notation for the single-cage locked-rotor machine is found:

$$\vec{\Psi}_s = L_\sigma \vec{i}_s + L_m(\vec{i}_s + \vec{i}_k') \tag{2-27}$$

$$\vec{\Psi}_k' = L_m(\vec{i}_s + \vec{i}_k') \tag{2-28}$$

The voltage space vector representation is then given by

$$\begin{aligned} k\vec{u}_k &= k^2 R_k \vec{i}_k' + \dot{\vec{\Psi}}_k' \\ \vec{u}_k' &= R_k' \vec{i}_k' + \dot{\vec{\Psi}}_k' = 0 \end{aligned} \tag{2-29}$$

$$\vec{u}_s = R_s \vec{i}_s + \dot{\vec{\Psi}}_s \tag{2-30}$$

2.2.2. DOUBLE-CAGE / DEEP-BAR INDUCTION MACHINE

In this chapter the influence of skin effect on the machine characteristics in addition to a single-cage machine is studied. Skin effects appears more or less in every electric machine. As it will be shown , it is possible to model skin effect by assuming that the machine has a double-cage rotor. The resulting equivalent circuit will be used later to create a dynamic double-cage machine model.
The double-cage induction machine was essentially made to upgrade the performance at start-up. The only difference with respect to the single-cage machine is its rotor construction. This consists of two rotor bars placed parallel, see Fig. 2.8a. The outer cage K_1 is a thin bar cage with high resistance and is placed immediately below the rotor surface, thus has little leakage inductance. As a result this cage produces a high pull-out torque and due to the low leakage inductance, this cage dominates at high rotor frequencies. The inner 'operational' rotor cage K_2 is thick and has a low resistance. Because it is embedded in the rotor iron the leakage inductance is high. This cage becomes mainly effective at low slip frequencies.

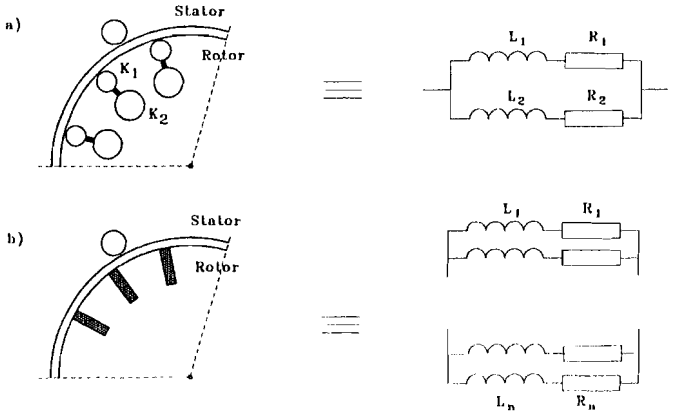


Figure 2.8: a)double-cage rotor b) Deep-bar rotor

The same effect can be achieved by the use of a deep-bar rotor instead of a double-cage, Fig. 2.8b. In case of a deep-bar rotor, the lower position of the bars is linked by more slot leakage flux than the upper part of the bars. Under transient conditions the inductance of the lower part of the bar is higher than that of the upper part, which will increase the effective resistance of the bar and therefore also the produced torque. This phenomenon is the same for both deep-bar and double-cage rotors.
A double- cage can be described by two parallel L-R branches, while a deep-bar is usually approximated by a large number of parallel L-R branches (Fig. 2.8b). However, as already said, both systems have one common characteristic: the effective rotor resistance increases and the effective rotor inductance decreases with frequency.

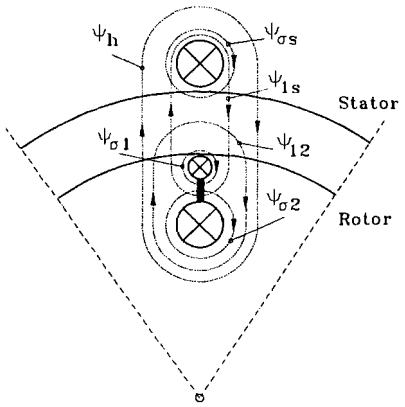


Figure 2.9: physical interpretation of the fluxes

In Fig.2.9 the physical interpretation of the fluxes in a double-cage induction machine is depicted. Using the space vector notation, the flux linkages can be expressed as function of the rotor and stator currents in the following way:

$$\bar{\Psi}_h^s = L_h\left(\bar{i}_s^s + \bar{i}_{k1}^s + \bar{i}_{k2}^s\right) \tag{2-31}$$

$$\bar{\Psi}_s^s = L_{\sigma s}\bar{i}_s^s + L_{1s}\left(\bar{i}_s^s + \bar{i}_{k1}^s\right) + \bar{\Psi}_h^s \tag{2-32}$$

$$\bar{\Psi}_{k1}^s = L_{1s}\left(\bar{i}_s^s + \bar{i}_{k1}^s\right) + L_{12}\left(\bar{i}_{k1}^s + \bar{i}_{k2}^s\right) + L_{\sigma 1}\bar{i}_{k1}^s + \bar{\Psi}_h^s \tag{2-33}$$

$$\bar{\Psi}_{k2}^s = L_{12}\left(\bar{i}_{k1}^s + \bar{i}_{k2}^s\right) + L_{\sigma 2}\bar{i}_{k2}^s + \bar{\Psi}_h^s \tag{2-34}$$

with:

L_h : main air gap inductance
L_{1s} : mutual inductance of upper cage and stator
L_{12} : mutual inductance of upper cage and lower cage
$L_{\sigma s}$: leakage inductance of the stator
$L_{\sigma 1}$: leakage inductance of the upper cage
$L_{\sigma 2}$: leakage inductance of the lower cage

The fluxes $\bar{\Psi}_{s1}$ and $\bar{\Psi}_{\sigma 1}$ have to cross more than once material with low permeability. As a result we can postulate that these fluxes are much smaller than the remaining leakage fluxes. Thus they are assumed to be approximately equal to zero. The flux Eq.(2.31 to 2-34) reduce then to:

$$\bar{\Psi}_h^s = L_h\left(\bar{i}_s^s + \bar{i}_{k1}^s + \bar{i}_{k2}^s\right) \tag{2-35}$$

$$\bar{\Psi}_s^s = L_{\sigma s}\bar{i}_s^s + \bar{\Psi}_h^s \tag{2-36}$$

$$\bar{\Psi}_{k1}^s = L_{12}\left(\bar{i}_{k1}^s + \bar{i}_{k2}^s\right) + \bar{\Psi}_h^s \tag{2-37}$$

$$\vec{\Psi}_{k2}^{s} = L_{12}\left(\vec{i}_{k1}^{s} + \vec{i}_{k2}^{s}\right) + L_{\sigma 2}\vec{i}_{k2}^{s} + \vec{\Psi}_{h}^{s} \tag{2-38}$$

The vector diagram for this simplified situation is given in Fig. 2.10. The voltage vector equations of stator and rotor are given by:

$$\vec{u}_{s}^{s} = R_{s}\vec{i}_{s}^{s} + \dot{\vec{\Psi}}_{s}^{s} \tag{2-39}$$

$$\vec{0} = R_{k1}\vec{i}_{k1}^{r} + \dot{\vec{\Psi}}_{k1}^{r} \tag{2-40}$$

$$\vec{0} = R_{k2}\vec{i}_{k2}^{r} + \dot{\vec{\Psi}}_{k2}^{r} \tag{2-41}$$

these equations results in the locked-rotor model of the double-cage machine, given in Fig. 2.11.

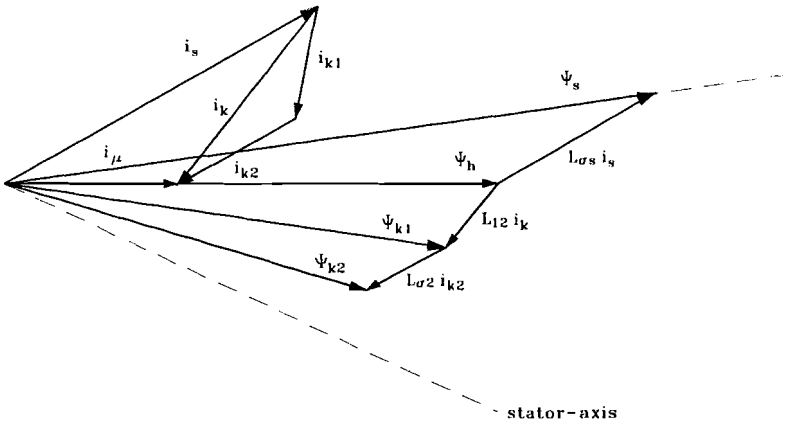


Figure 2.10: vector diagram of the double-cage machine

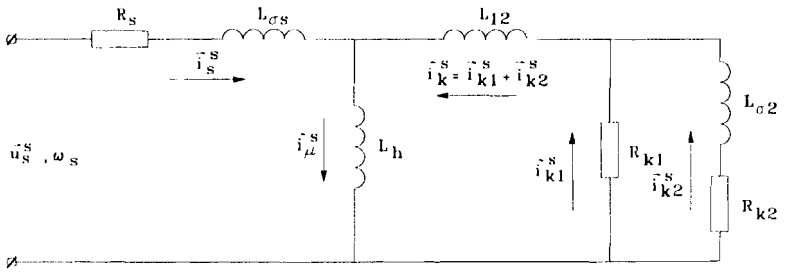


Figure 2.11: locked-rotor double-cage model.

As with the single-cage model, we also have a causality problem, which will be solved in a similar way. The three inductances of the T-network will be transformed to a two-inductance equivalent scheme (depicted in Fig. 2.12), with:

$$k = \frac{L_h}{L_h + L_{12}} \tag{2-42}$$

$$L_m = k^2\left(L_h + L_{12}\right) = \frac{L_h^2}{L_h + L_{12}} = \frac{L_h}{1 + \sigma_{12}} \tag{2-43}$$

$$L_\sigma = L_{\sigma s} + k\,L_{12} = L_{\sigma s} + \frac{L_{12}}{1+\sigma_{12}} \tag{2-44}$$

$$L'_{\sigma 2} = k^2 L_{\sigma 2} \tag{2-45}$$

$$R'_{k1} = k^2 R_{k1} \tag{2-46}$$

$$R'_{k2} = k^2 R_{k2} \tag{2-47}$$

The causal bond graph is given in Fig. 2.13, showing that the equivalent circuit can be used to derive an executable simulation model.

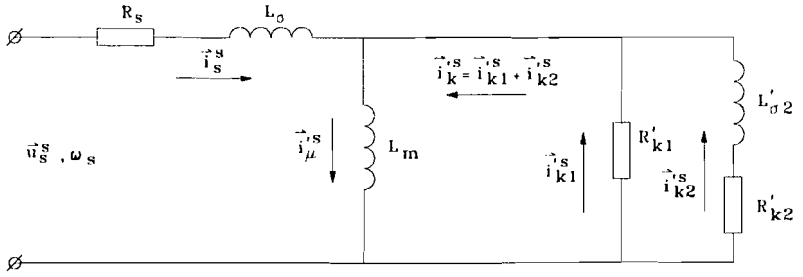


Figure 2.12: causal equivalent steady-state double-cage model

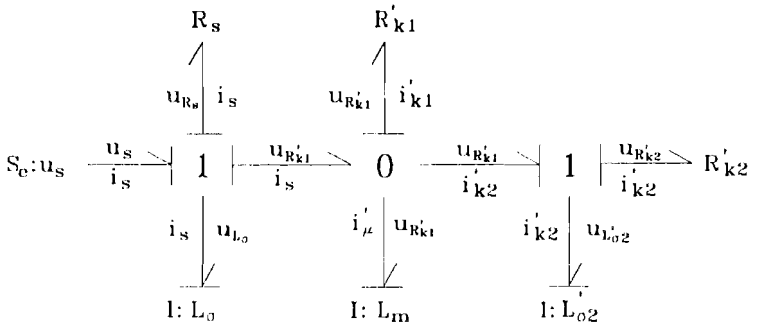


Figure 2.13: causal Bond-graph double-cage locked-rotor model.

Like in the single-cage situation a new set of rotor variables is introduced:

$$\bar{\Psi}'_{k1} = k \cdot \bar{\Psi}_{k1} \tag{2-48}$$

$$\bar{\Psi}'_{k2} = k \cdot \bar{\Psi}_{k2} \tag{2-49}$$

$$\bar{i}'_{k1} = \tfrac{1}{k} \cdot \bar{i}_{k1} \;,\quad \bar{i}'_{k2} = \tfrac{1}{k} \cdot \bar{i}_{k2} \quad \text{and} \quad \bar{i}'_{k} = \bar{i}'_{k1} + \bar{i}'_{k2} \tag{2-50}$$

Substituting the transformed parameters and variables into Eq's.(2-35 to 2-41) results in the flux space-vector notation:

$$\vec{\Psi}_s = L_m\left(\vec{i}_s + \vec{i}_k'\right) + L_\sigma \vec{i}_s \tag{2-51}$$

$$\vec{\Psi}_{k1}' = L_m\left(\vec{i}_s + \vec{i}_k'\right) \tag{2-52}$$

$$\vec{\Psi}_{k2}' = L_m\left(\vec{i}_s + \vec{i}_k'\right) + L_{\sigma 2}' \vec{i}_{k2}' \tag{2-53}$$

$$\vec{\Psi}_{\sigma 2}' = L_{\sigma 2}' \vec{i}_{k2}' \tag{2-54}$$

and the voltage space-vector notation:

$$\vec{u}_s = R_s \vec{i}_s + \dot{\vec{\Psi}}_s \tag{2-55}$$

$$\vec{0} = R_{k1}' \vec{i}_{k1}' + \dot{\vec{\Psi}}_{k1}' \tag{2-56}$$

$$\vec{0} = R_{k2}' \vec{i}_{k2}' + \dot{\vec{\Psi}}_{k2}' \tag{2-57}$$

The resulting vector diagram of a double-cage machine is depicted in Fig. 2.14.

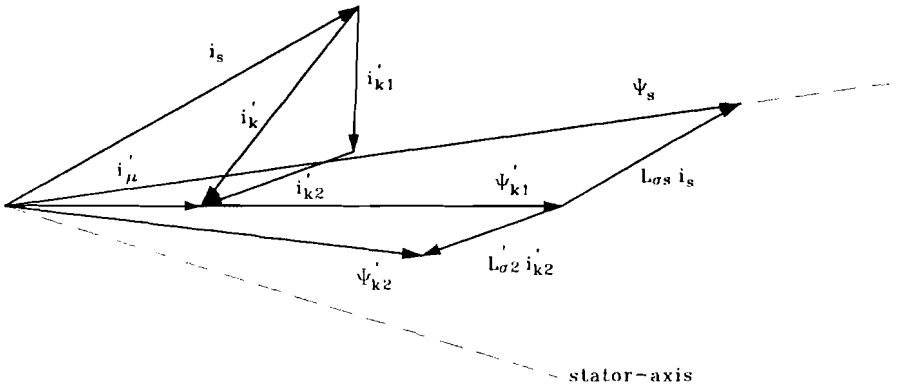


Figure 2.14: resulting double-cage vector diagram.

2.2.3. DEEP-BAR CAGE

When modelling a higher-order cage rotor, like in the case of a deep-bar rotor it will be convenient [Velt '94] to rewrite the rotor branch as a parallel L-R pair connected in series with some remaining resistance R_{ser} and inductance L_{ser}. In Fig. 2.15 this transformation is shown. Both schemes show identical admittance. For a double-cage machine the values can be rewritten as follows:

$$R_{ser} = \frac{L_1 R_2 + L_2 R_1 - L_1 L_2 (R_1 + R_2)}{L_1 + L_2} - \frac{R_1 R_2}{R_1 + R_2} \tag{2-58}$$

$$L_{ser} = \frac{L_1 L_2}{L_1 + L_2} \tag{2-59}$$

$$L_p = (L_1 + L_2)\frac{R_1 R_2}{(R_1 + R_2)^2} \tag{2-60}$$

$$R_p = \frac{R_1 R_2}{R_1 + R_2} \tag{2-61}$$

This series approximation is leads to digital simulation equations with higher time constants, which allows larger simulation steps. However, this series representation leads to a non-physical rotor model, where the stator flux remains unchanged.

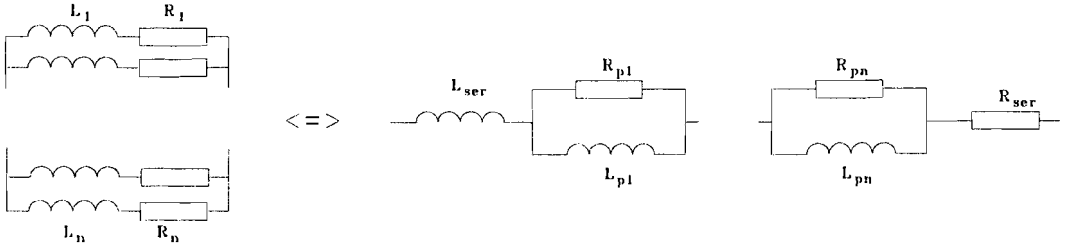


Figure 2.15: parallel to series transformation

2.3. DYNAMIC MODEL

In this paragraph a dynamic model for an asynchronous machine is derived, starting with the double-cage flux and voltage equations (Eq. 2-51 to 57). The physical machine consists of a stationary part (stator) and a rotating part (rotor), both with there own reference frame. The angle of displacement between these reference frames is the mechanical angle ρ^s. If the number of pole pairs of the machine equals one, the mechanical rotation speed is the derivative of this angle:

$$\omega_m = \dot{\rho}^s \quad \text{for } p = 1 \tag{2-62}$$

The mechanical rotation speed (shaft speed) is calculated using:

$$\dot{\rho}^s = \frac{1}{J}\int (M_{el} - M_{load})\,dt \qquad \rho^s = \int \dot{\rho}^s dt \tag{2-63}$$

with J the momentum of inertia of the machine plus load. The electrical torque, produced in the machine is given by:

$$M_{el} = \left[\boldsymbol{R}\left(\frac{\pi}{2}\right)\bar{\Psi}_s^s\right]^T \cdot \bar{i}_s^s = -i_s^{s1} \cdot \Psi_s^{s2} + i_s^{s2} \cdot \Psi_s^{s1} \tag{2-64}$$

To create a dynamic machine model these mechanical equations also have to be used. Therefore the space vectors of flux and voltages are rewritten in a way that the stator part is denoted in stator coordinates and the rotor part is given in rotor coordinates. The stator flux in stator coordinates will then be given in space vector notation by:

$$\bar{\Psi}_s^s = L_m\left(\bar{i}_s^s + \bar{i}_k^{'s}\right) + L_\sigma \bar{i}_s^s = \bar{\Psi}_{k1}^{'s} + L_\sigma \bar{i}_s^s \tag{2-65}$$

The rotor flux is denoted in rotor coordinates as follows:

$$\bar{\Psi}_{k1}^{'r} = L_m\left(\bar{i}_s^r + \bar{i}_k^{'r}\right) \tag{2-66}$$

$$\bar{\Psi}_{k2}^{'r} = L_m\left(\bar{i}_s^r + \bar{i}_k^{'r}\right) + L_{\sigma 2}^{'} \bar{i}_{k2}^{'r} \tag{2-67}$$

The voltage space-vector equations are split into stator coordinates for the stator voltage and rotor coordinates for the rotor voltage:

$$\bar{u}_s^s = R_s \bar{i}_s^s + \dot{\bar{\Psi}}_s^s \tag{2-68}$$

$$\bar{0} = R_{k1}^{'} \bar{i}_{k1}^{'r} + \dot{\bar{\Psi}}_{k1}^{'r} \tag{2-69}$$

$$\bar{0} = R_{k2}^{'} \bar{i}_{k2}^{'r} + \dot{\bar{\Psi}}_{k2}^{'r} \tag{2-70}$$

Looking at the voltage and flux equations shows that in this situation the stator current and the flux $\bar{\Psi}_{k1}^{'}$ are used both in rotor and stator coordinates. Therefore, this current and flux must be transformed from rotor- to stator coordinates or vice-versa. This is done with the use of the standard vector rotation matrix $\boldsymbol{R}(\alpha)$ defined as:

$$\mathbf{R}(\alpha) = \begin{bmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{bmatrix} \tag{2-71}$$

The stator current is transformed to rotor coordinates as follows:

$$\begin{bmatrix} i_s^{r1} \\ i_s^{r2} \end{bmatrix} = \mathbf{R}\left(-\rho^s\right) \cdot \begin{bmatrix} i_s^{s1} \\ i_s^{s2} \end{bmatrix} \tag{2-72}$$

Another example of the transformation between stator and rotor is:

$$\begin{bmatrix} \Psi_{k1}^{'s1} \\ \Psi_{k1}^{'s2} \end{bmatrix} = \mathbf{R}\left(\rho^s\right) \cdot \begin{bmatrix} \Psi_{k1}^{'r1} \\ \Psi_{k1}^{'r2} \end{bmatrix} \tag{2-73}$$

Rewriting Eq.(2-65 to 2-70) results in the following equations in integral notation. The stator voltage and flux are presented in stator coordinates. The rotor fluxes and currents are given in rotor coordinates, using Eq.(2.68)

$$\begin{cases} \Psi_s^{s1} = \int u_s^{s1} - R_s i_s^{s1}\, dt \\ \Psi_s^{s2} = \int u_s^{s2} - R_s i_s^{s2}\, dt \end{cases} \tag{2-74}$$

rewriting Eq.(2.65) results in the following equation for the stator current:

$$\begin{cases} i_s^{s1} = \dfrac{\Psi_s^{s1} - \Psi_{k1}^{'\,s1}}{L_\sigma} \\ i_s^{s2} = \dfrac{\Psi_s^{s2} - \Psi_{k1}^{'\,s2}}{L_\sigma} \end{cases} \tag{2-75}$$

With Eq.(2.69) in the integral form results in:

$$\begin{cases} \Psi_{k1}^{'\,r1} = -\int R_{k1}^{'} i_{k1}^{'\,r1}\, dt \\ \Psi_{k1}^{'\,r2} = -\int R_{k1}^{'} i_{k1}^{'\,r2}\, dt \end{cases} \tag{2-76}$$

The leakage flux of the inner-cage can be derived using Eq.(2.66 and 2.67).

$$\begin{cases} \Psi_{\sigma 2}^{'\,r1} = \int\left(-R_{k2}^{'} i_{k2}^{'\,r1}\right)dt - \Psi_{k1}^{'\,r1} = \int\left(R_{k1}^{'} i_{k1}^{'\,r1} - R_{k2}^{'} i_{k2}^{'\,r1}\right)dt \\ \Psi_{\sigma 2}^{'\,r2} = \int\left(-R_{k2}^{'} i_{k2}^{'\,r2}\right)dt - \Psi_{k1}^{'\,r2} = \int\left(R_{k1}^{'} i_{k1}^{'\,r2} - R_{k2}^{'} i_{k2}^{'\,r1}\right)dt \end{cases} \tag{2-77}$$

The currents flowing in the rotor circuit can now be calculated as follows:

$$\vec{i}_k^{\,'\,r} = \frac{\vec{\Psi}_{k1}^{'\,r}}{L_m} - \vec{i}_s^{\,r} \tag{2-78}$$

$$\vec{i}_{k1}^{\,'\,r} = \vec{i}_k^{\,'\,r} - \vec{i}_{k2}^{\,'\,r} \tag{2-79}$$

$$\vec{i}_{k2}^{\,'\,r} = \frac{\vec{\Psi}_{\sigma 2}^{'\,r}}{L_{\sigma 2}^{'}} \tag{2-80}$$

These equations are used to describe a block diagram of the resulting dynamic simulation model, which is given in Fig. 2.16.

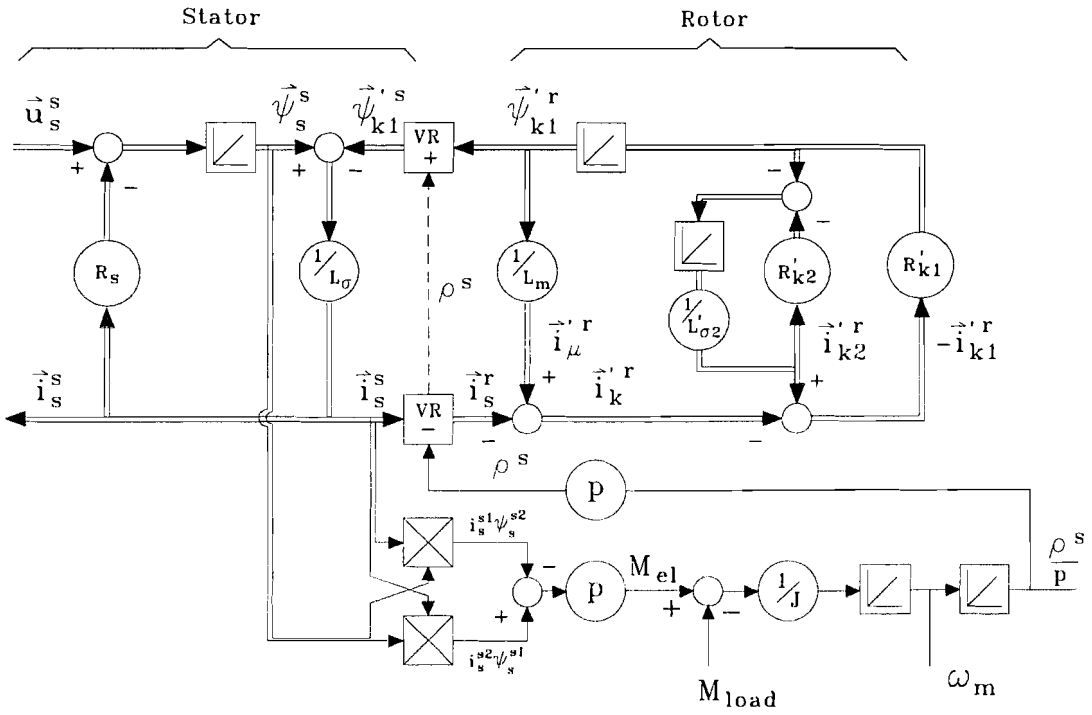


Figure 2.16: block diagram of the double-cage simulation model

This simulation model is used for the simulations made in this report. The block diagram can easily be rewritten to a single-cage model by removing the inner loop$\left(R'_{k2}, L'_{\sigma 2}\right)$. For higher order models only the rotor side has to be rewritten, by adding other inner loops.

2.4. DYNAMIC MODELLING USING THE IDEAL ROTATING TRANSFORMER

Another way to approach the dynamic machine modelling lies in expanding the locked-rotor model with an *Ideal Rotating Transformer* (IRTF) [Velt '94], see Fig.2.17. The IRTF is a theoretical device that connects two orthogonal per-phase circuits with a mechanical system. Therefore it has two electrical multi-ports, one for the stator and one for rotor, and one mechanical port. The IRTF has the following properties:

- no energy dissipation or storage
- no leakage flux, all flux is mutual
- no magnetising current (L=∞)
- the angle between rotor and stator reference frame ρ^s is equal to the shaft angle if the machine has two pole pairs (p=1)

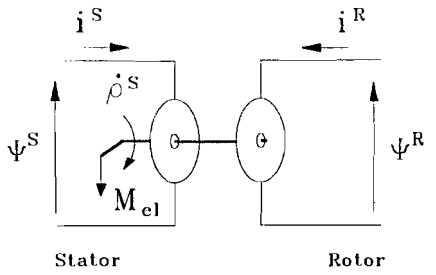


Figure 2.17: symbol for the Ideal Rotating TransFormer

An ideal transformer transforms current and voltage by means of the mutual main flux. Because the IRTF does not need a magnetising current, the current and flux on both sides of the IRTF have the same length. The difference lies in the reference frames of stator and rotor (Fig. 2.18).

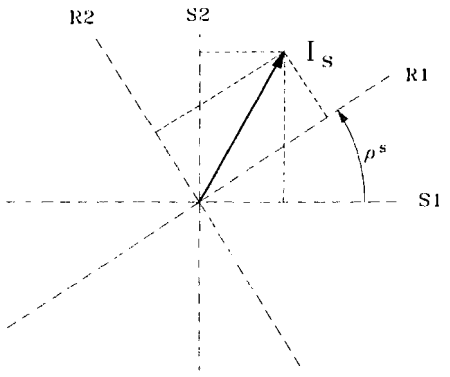


Figure 2.18: stator and rotor reference frame

The IRTF just transforms currents and fluxes from rotor to stator coordinates and vice versa, with the use of the standard vector rotation matrix **R**(α) (Eq. 2-71). One can look at the IRTF as an induction motor with $L_m = \infty$, $L_{\sigma s} = L_{\sigma k} = 0$, $R_s = 0$ *and* $R_k = \infty$. The current and flux vector can be represented in rotor coordinates (r1,r2) or in stator coordinates (s1,s2). The angle between the rotor and stator coordinate system ρ^s is the output of the mechanical part of the machine Eq. 2-63:

$$\dot{\rho}^s = \frac{1}{J}\int (M_{el} - M_{load})\,dt \quad \text{and} \quad \rho^s = \int \dot{\rho}^s\,dt \tag{2-81}$$

For the calculation of the electric torque M_{el} in the machine Eq. 2-64 is used:

$$M_{el} = \left[\boldsymbol{R}\left(\tfrac{\pi}{2}\right)\bar{\Psi}^s\right]^T \cdot \bar{i}^s = \begin{bmatrix} -\Psi^{s2} \\ \Psi^{s1} \end{bmatrix}^T \cdot \begin{bmatrix} i^{s1} \\ i^{s2} \end{bmatrix} \tag{2-82}$$

Only the reference frames of the flux and current on both sides of the IRTF are different. Therefore the electric torque can also be defined using flux and current in rotor coordinates.

$$M_{el} = -i^{s1}\cdot\Psi^{s2} + i^{s2}\cdot\Psi^{s1} = -i^{r1}\cdot\Psi^{r2} + i^{r2}\cdot\Psi^{r1} \tag{2-83}$$

IMPEDANCES AT AN IRTF

The elements of the two dimensional space vectors with real components $\bar{u}_s^s$ and $\bar{i}_s^s$, may be represented as complex valued numbers defined as:

$$I_s^S = i_s^{s1} + j\,i_s^{s2} \tag{2-84}$$

$$U_s^S = u_s^{s1} + j u_s^{s2} \tag{2-85}$$

The rotor circuit contains a resistor $R_k^{'}$ connected in series with an inductance L_K which , under steady-state conditions and sinusoidal voltages and currents, can be described as:

$$Z_k^R = R_k^{'} + j(\omega_s - \omega_m)L_k = \frac{U_k^R}{I_k^R} \tag{2-86}$$

As the rotor flux equals the time integral of the rotor impedance times the rotor current

$$\Psi_k^R = \frac{R_k^{'} + j(\omega_s - \omega_m)L_k}{j(\omega_s - \omega_m)} \cdot |I_k| \cdot e^{j(\omega_s - \omega_m)t} \tag{2-87}$$

with: $$I_k^R = |I_k| \cdot e^{j(\omega_s - \omega_m)t} \rightarrow I_k^S = |I_k| \cdot e^{j(\omega_s)t} \tag{2-88}$$

The impedance that appears at the stator side Z_k^S depends on the mechanical shaft speed ω_m:

$$U_k^S = \frac{d}{dt}(\Psi_k^S) = \frac{d}{dt}\left(\frac{R_k^{'} + j(\omega_s - \omega_m)L_k}{j(\omega_s - \omega_m)} \cdot |I_k| \cdot e^{j\omega_s t}\right) \tag{2-89}$$

$$U_k^S = \left(\frac{j\omega_s R_k^{'}}{j(\omega_s - \omega_m)} + j\omega_s L_k\right) \cdot I_k^S \tag{2-90}$$

A rotor impedance Z_k^R will thus be seen as a virtual impedance Z_k^R on the stator side:

$$Z_k^R = R_k' + j(\omega_s - \omega_m)L_k \quad \rightarrow \quad Z_k^S = \frac{\omega_s R_k'}{(\omega_s - \omega_m)} + j\omega_s L_k \tag{2-91}$$

An inductance on the rotor side of the IRTF can be moved across the IRTF to the stator side without changing it's value. A resistor however will appear at the stator side as a resistor with a value depending on the slip (ω_S-ω_m), which is equal to the, previously discussed, steady state circuit.
One can therefore conclude that inductances are free to move across the IRTF . The IRTF can thus be placed at 3 positions in the equivalent locked-rotor circuit: between R_s and L_σ, between L_σ and $L_m$or between L_m and the rotor circuit. After placing the IRTF the rotor resistance will have a constant value, independent of the slip frequency. In the block diagram of Fig. 2.16 the *vector rotating blocks (VR), which have the same function as the IRTF, can therefore also be moved without changing the model properties*. This option will be used for identification purposes in section 3.2. In Fig.2.19 the IRTF is placed between L_σ and L_m this is also the case in the block diagram of Fig.2.16.

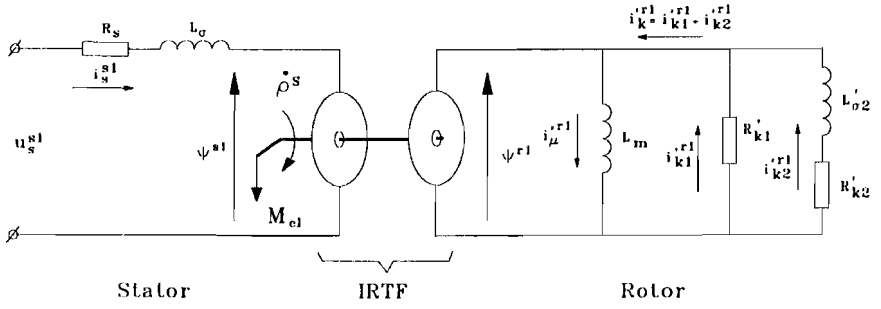


Figure 2.19: dynamic per-phase equivalent circuit with IRTF

Using the IRTF as an expansion of the locked-rotor equivalent circuits, results in a dynamic per-phase circuit as depicted in Fig. 2.19. This dynamic equivalent circuit can be translated to the dynamic block diagram (Fig. 2.16) of an induction machine as derived in Section.2.4.

2.5. CONVERTER MODELLING

Nowadays most electrical drive systems use power electronic converters as power supply. The generated voltage will always contain harmonics, which can be used for identification purposes. For simulations a PWM-inverter is studied, which is a voltage source converter. Voltage source converters consist of three main parts: a rectifier, a filter and an inverter. In Fig. 2.20 the general structure of a 3-phase voltage source converter fed motor drive is given. The 3-phase rectifier converts the 3-phase voltage into a rectified DC-voltage with some ripple. By means of the LC the rectified input current is smoothened and thereby reducing the voltage ripple on the DC-link. Finally, a 3-phase inverter transforms the DC-link voltage into a 3-phase AC-voltage. The fundamental frequency component can be adjusted both in frequency and in amplitude.

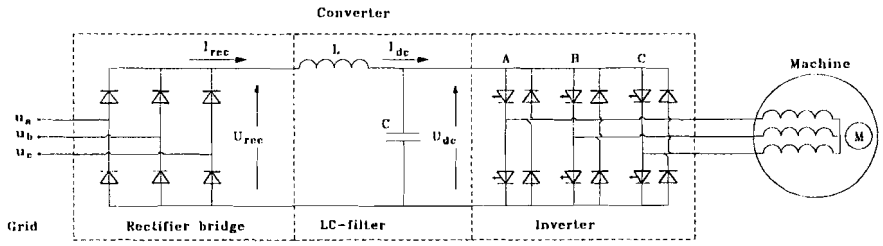


Figure 2.20: Voltage source converter fed motor drive.

The three phase rectifier is connected to the symmetrical three pase grid (380/220 V , 50Hz):

$$u_a = \hat{u} \cdot \sin(\omega_s t + \varphi) \tag{2-92}$$

$$u_b = \hat{u} \cdot \sin\left(\omega_s t - \tfrac{2}{3}\pi + \varphi\right) \tag{2-93}$$

$$u_c = \hat{u} \cdot \sin\left(\omega_s t - \tfrac{4}{3}\pi + \varphi\right) \tag{2-94}$$

We assume that the diodes in the rectifier are ideal, without voltage drop. The grid impedance is neglected.

$$U_{rec} = max[u_a\ u_b\ u_c] - min[u_a\ u_b\ u_c] \tag{2-95}$$

The LC-filter is used to reduce the voltage ripple on the U_{dc}, to reduce the harmonic content of the rectified current and to enable negative values of the DC-current for short periodes of time. This filter is also assumed to be ideal. Therefore, the DC-voltage is thought to be constant for simulation purposes.

$$U_{dc} = max(u_{line}) = \sqrt{6}\, u_{per\ fase} = 539\,V \tag{2-96}$$

The voltage source inverter consists of three phase legs; each leg consists of a pair of power semiconductor switching devices. Ideally each phase leg can be considered as a 2-level switch that can switch between two voltage levels: the positive and the negative DC-link terminal.

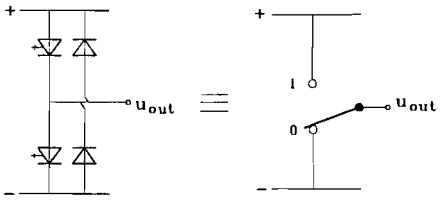


Figure 2.21: equivalent 2-level switches.

Using different switching patterns of the 2-level switches, it is possible to vary both the amplitude and frequency of the output voltages. Because of the inductive character of the asynchronous machine and the possibility to deliver reactive power of the converter (freewheel diodes), the voltage source converter is a suitable (voltage) supply. Therefore, the three terminals of the 3-phase induction machine can be directly connected with the inverter.

PWM-SWITCHING PATTERN

The conversion of a constant dc-link voltage to three-phase voltages whose amplitude and frequency are controllable can be done with a *pulse-width modulation* technique (PWM). The goal of pulse-width modulation is to achieve sinusoidal stator currents in the induction machine.
In case of sinusoidal PWM, the control signals of the 2-level switches are obtained by comparing a sinusoidal reference signal (added with 12% of third harmonic component), u_{ref}, with a triangular carrier signal u_{tri}, as depicted in Fig. 2.22.

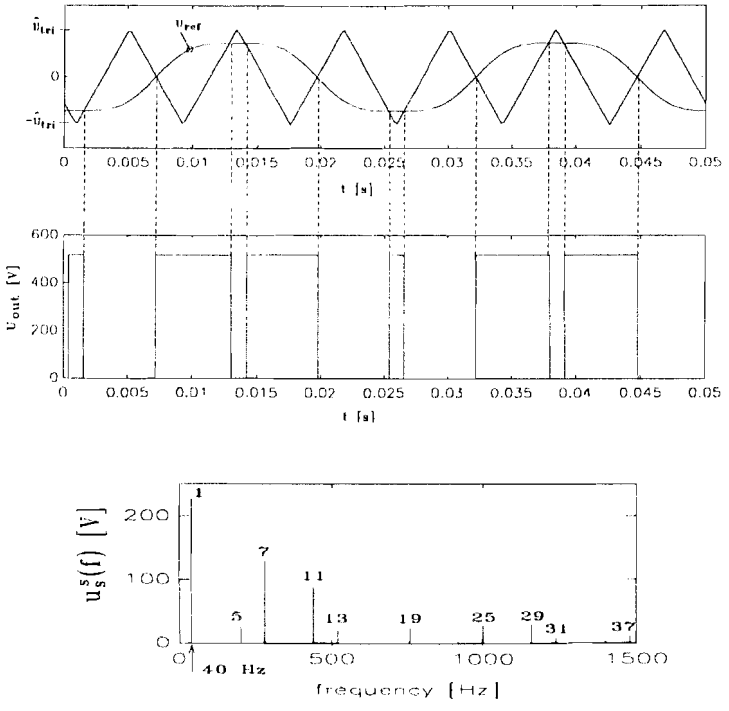


Figure 2.22: PWM switching pattern and spectrum of a inverter phase voltage

The inverter switching frequency is determined by the frequency ω_{tri}. The frequency and amplitude $\hat{u}_{tri}$ of the triangular carrier wave are usually hold constant. The amplitude and the frequency of the reference signal are the desired values of the fundamental output voltage component. The switching moments are determined by the points of intersection between the reference signal and the carrier. When $u_{ref} > u_{tri}$ the output voltage of the particular phase is connected to the positive terminal of the DC-voltage. When the reference voltage is larger than the triangle voltage $u_{ref} < u_{tri}$ then the output is connected to the negative terminal of the DC-voltage. Each phase is compared with a the same triangle carrier. The number of the carrier frequency periods that fit in one period of the reference signal frequency is called *pulse number*, defined as:

$$n_p = \frac{\omega_{tri}}{\omega_{ref}} \tag{2-97}$$

Thus there are n_p pulses in each half period of the output voltages. The relation between the amplitude of the reference signal and the carrier is called the *modulation index*, defined as:

$$m_i = \frac{\hat{u}_{ref}}{\hat{u}_{tri}} \tag{2-98}$$

In Fig 2.22. the modulation index and pulse number are set to a constant value: $m_i = 0.8$, $n_p = 3$. Three types of modulation are defined:

- sinus modulation, if $0 \le m_i \le 1$
- over modulation, if $m_i > 1$
- block modulation, if $m_i \to \infty$.

Only in case of sinus modulation the fundamental component of the output voltage is proportional with the modulation index:

$$u_{sa(1)} = m_i \frac{U_{dc}}{2} \cdot sin(\omega_s \cdot t) \tag{2-99}$$

For simulations in this report only sinus modulation is used, In that case the maximum value of the fundamental output voltage is half the DC-voltage. Thus if a constant DC-voltage, $U_{dc} = 539V$, is used with $m_i = 0.8$ the maximum per-phase fundamental voltage is 215.5V. Therefore the motor can not be used at rated power.
The output voltage of the inverter not only contains a fundamental component with frequency, $\omega_{ref} = \omega_s$, but harmonic components of the carrier frequency, ω_{tri}, as well. Increasing the value of the pulse-number n_p results in higher inverter switch losses, but reduces the machine harmonic losses. The choice of pulse-number is therefore a balance between inverter loss and machine harmonic losses.
Usually a third harmonic is added to the sine reference wave (12% of the fundamental component) to make the wave form more flat-topped, which allows a larger modulation index. If the connected machine windings are connected to a common star-point this third harmonic will not appear in the phase voltages.

A completely different approach for generating PWM results from representing the three-phase inverter output voltages in a space vector system. This approach is quite convenient to be implemented by microprocessors, being widely used nowadays. However, the space vector method generates virtually the same pulse pattern as the sinusoidal PWM with increased voltage utilisation (third harmonic addition). For this reason, only the sinusoidal PWM technique will be considered in this report.

INVERTER SIMULATION

For simulations of inverter fed asynchronous machines, the machine models according to Sec. 2.3 will be used. The input for these models are the two-phase voltages, which are the transformed three-phase voltages according to Appendix B. Therefore the three-phase stator voltages have to be provided by the inverter model. Because the stator voltages have the star-point *0* as reference point, the per-phase stator voltages have to be calculated with respect to this reference point, see Fig 2.23.

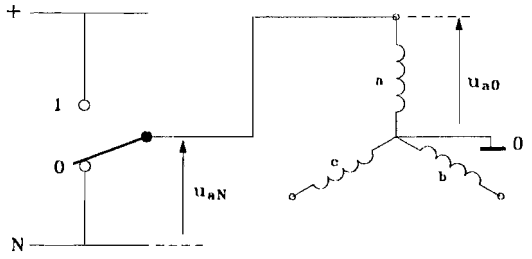


Figure 2.23: voltage references of motor and inverter.

The inverter output voltages with respect to the negative terminal N, are rewritten with the star-point of the machine as reference, using the following transformation:

$$u_{0N} = \tfrac{1}{3}\left(u_{aN} + u_{bN} + u_{cN}\right) \tag{2-100}$$

$$\begin{bmatrix} u_{a0} \\ u_{b0} \\ u_{c0} \end{bmatrix} = \begin{bmatrix} u_{aN} \\ u_{bN} \\ u_{cN} \end{bmatrix} - \begin{bmatrix} u_{0N} \\ u_{0N} \\ u_{0N} \end{bmatrix} = \frac{1}{3}\begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} u_{aN} \\ u_{bN} \\ u_{cN} \end{bmatrix} \tag{2-101}$$

These stator voltages with respect to the star-point can now be transformed to the two-phase representation, which can be fed into the machine simulation model.

2.6. COMPARING DOUBLE- AND SINGLE-CAGE EFFECTS

In this section the behaviour of a double-cage and a single-cage machine model will be compared, using the previously discussed machine and inverter models. The parameters of the double-cage machine are taken from previous work of [Doncl '86]. The used single-cage machine is an approximation of the double-cage machine steady-state speed/torque characteristic in the normal operating range.
To compare the effects of a double-cage with respect to a single-cage motor model, the following simulations are made:

- steady-state speed / torque curves;
- no load situation, motor fed with a PWM-inverter;
- dynamic response, motor fed with a PWM-inverter .

The simulation models are implemented in the simulation program "PSI/c", while this simulation program has the ability to simulate an ideal inverter model without errors in switching events. The used program is given in Appendix E2.

2.6.1. STEADY-STATE SPEED/TORQUE CURVES

In Fig. 2.24 the speed/torque curves of a double- and single-cage machine model are depicted. The difference lies at the points where the slip is high, thus at start-up. Here the torque produced by the double-cage machine model is significantly higher, as expected. The rated speeds are almost equal for both machines.

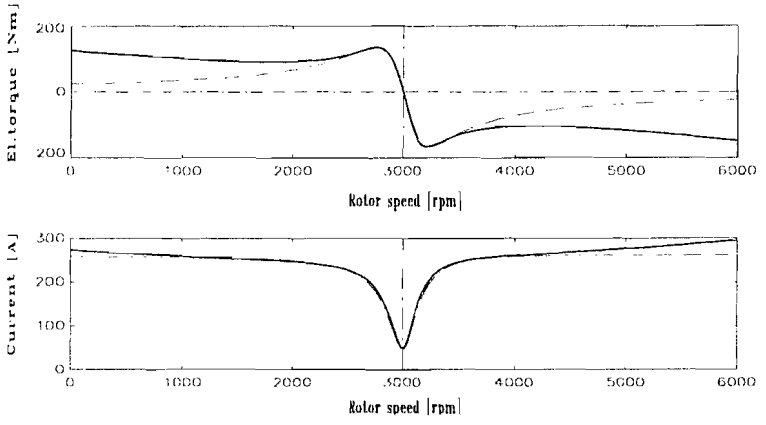


Figure 2.24: steady-state torque and current characteristics ——double-cage model, --- single-cage model.

Machines used for frequency controlled drives are using the most effective operation range of the curve, which is around synchronous speed. Inverters with variable frequency and voltage enable the use of only this area. Therefore, the advantage of the double-cage (higher torque at high slip frequencies) is no longer useful. However, as a result of the use of inverters, higher harmonics are generated. These components will excite the outer rotor-cage, with high resistance and low leakage inductance. Compared

to a single cage, the higher rotor resistance causes higher i^2R power loss for the harmonic currents and the lower leakage inductance will result in a higher torque ripple.

2.6.2. NO LOAD SITUATION, MOTOR FED WITH A PWM-INVERTER

The influence of higher harmonics produced by an inverter on electric torque and current of a double- and single-cage machine is studied using the PWM-inverter simulation model at fundamental frequency of 40 Hz (0,8 pu) and a dc-voltage of 537 V (1.73 pu). In Fig. 2.25 the torque and currents of the double-cage and single-cage model are shown.

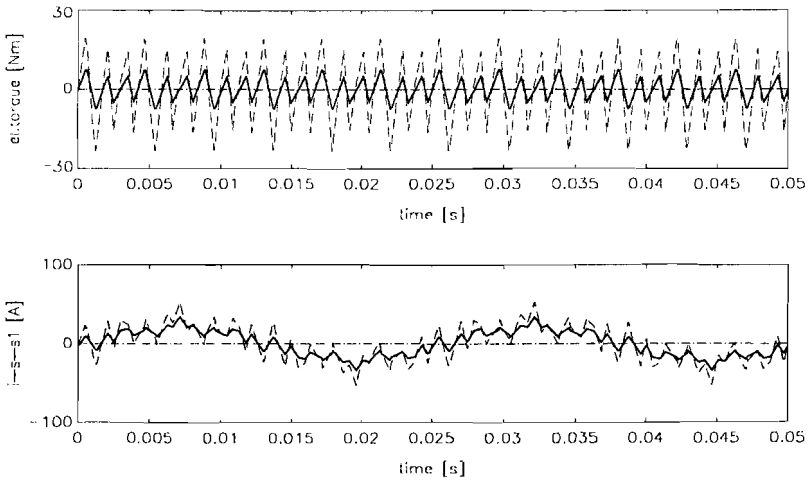


Figure 2.25: PWM inverter fed machine, pulse number=9
- - - double-cage model,—— single-cage model

The double-cage machine has a significant larger current and torque ripple compared to the single-cage machine. The stator fluxes of both machine models are almost equal, which is important in field-oriented control of a machine with skin effect. The torque and current ripple will decrease when the pulse number is increased (less harmonic content). However the percentile difference between double- and single-cage, current and torque ripple, will remain the same.

2.6.3. DYNAMIC RESPONSE, MOTOR FED WITH A PWM-INVERTER

To study the behaviour of the machine models, we will look at the response of the two machines models at a sudden load change $M_{load}=0 \rightarrow 40\%$ of the rated torque, at rated speed. In Fig. 2.26 the dynamic response, for a double- and single-cage machine, is given.

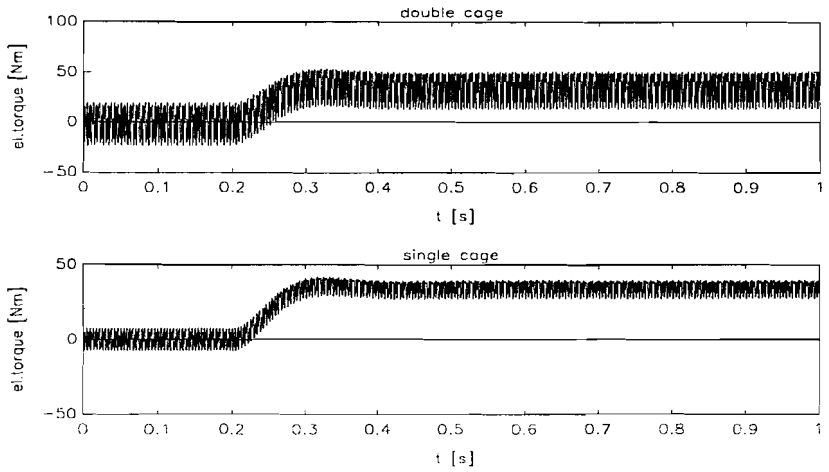


Figure 2.26: electric torque response at load change of a double- and single-cage machine model

The stator currents i_s^{s1} of the single- and double-cage machine model at the load change is depicted in Fig. 2.27.

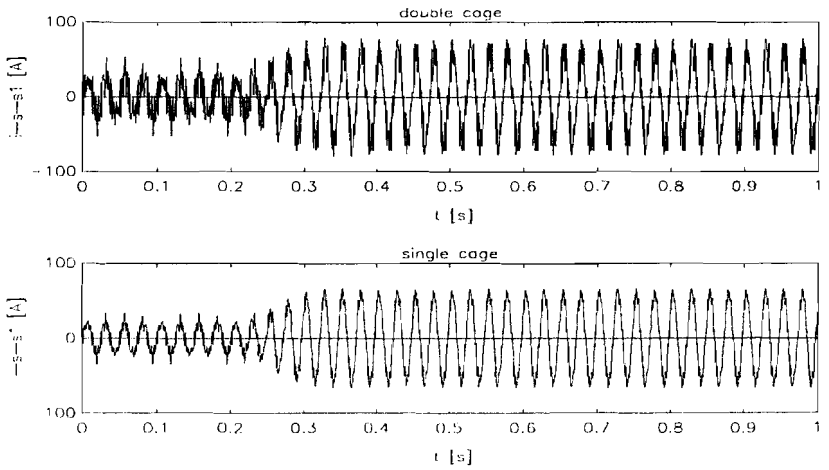


Figure 2.27: stator currents of double- and single-cage model at load change.

As a result of the performed simulations one can conclude that, if the steady-state operating condition is in the low slip region, the dominant time constant of both single- and double-cage models is the same. This means that a machine with skin effect (double-cage) can be approximated by an equivalent single-cage model when only the dominant time constant of the machine has to be known. This can be the case in a speed control system.

3. MACHINE PARAMETER IDENTIFICATION

In this chapter two identification methods, for machine parameter estimation, are discussed. Machine parameters are needed for control design or machine analysis.
The first method used to estimate the machine parameters is a method proposed by [Bla'94]. The method uses two flux-estimation models. Identification is done by adjusting the model parameters in different operation conditions, resulting in an estimation of single-cage machine parameters. Another method proposed by [Gort '94], using the Output Error Method, will also be studied. This method allows higher order machine parameters estimation (e.g. the double-cage parameters) and no special operation conditions are needed.

3.1. TWO-FLUX-MODEL METHOD

The estimation method proposed by Blaschke uses two auxiliary models, an U/I-model and an I/φ-model, to estimate the single-cage parameters L_σ, L_m, R_k' and R_s. In each model the cage-flux Ψ_k' is calculated, the U/I-model uses the stator resistance and the leakage inductance while the I/φ-model uses the main inductance and the rotor resistance. By adjusting these model parameters the fluxes can be made equal, leading to the desired machine parameters. In Fig. 3.1 a scheme of the two models is given. For the use of the method the following assumptions are made:

- R_s can easily be measured and is therefore known;
- constant saturation level (only small variations around a desired flux level are admitted).

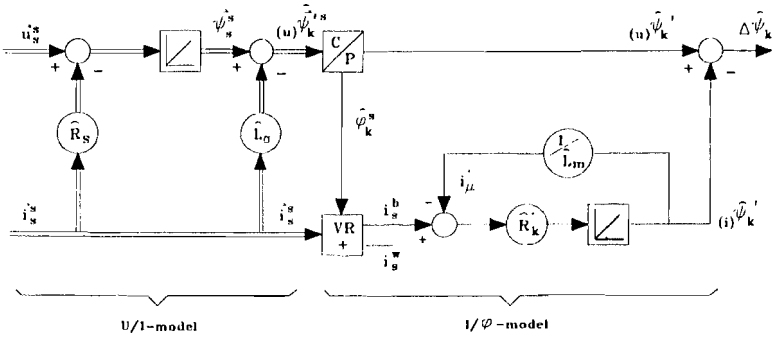


Figure 3.1: identification model, with U/I-model and I/φ-model

The U/I-model uses measured stator voltages and currents to estimate the stator flux.

$$\vec{u}_s^s = \hat{R}_s \vec{i}_s^s + \dot{\vec{\Psi}}_s^s \tag{3-1}$$

$$\hat{\vec{\Psi}}_s^s = \int \left(\vec{u}_s^s - \hat{R}_s \vec{i}_s^s \right) dt \tag{3-2}$$

The cage flux ${}_{(u)}\hat{\vec{\Psi}}_k^{'s}$ of the U/I-model can be derived by subtracting the leakage flux from the stator flux.

$$ {}_{(u)}\hat{\vec{\Psi}}_k^{'s} = \hat{\vec{\Psi}}_s^{s} - \hat{L}_\sigma \vec{i}_s^{s} \tag{3-3} $$

This "cage" flux in stator coordinates is transformed to polar notation. The output of U/I-model is the flux in polar form; the length of the flux vector ${}_{(u)}\hat{\Psi}_k^{'}$ and the angle $\hat{\varphi}_k^s$ between the cage flux and stator coordinates.

This angle $\hat{\varphi}_k^s$ is used as input for the I/φ-model. Rotating the stator current $\vec{i}_s^{\,s}$ over the angle $\hat{\varphi}_k^s$, results in a projection of the stator current, in stator coordinates, upon the flux coordinates. This is called field orientation, see Fig. 3.2. In flux coordinates the current component parallel to the flux is called the *blind*-component i_s^b and the perpendicular component is called the *work*-component i_s^w

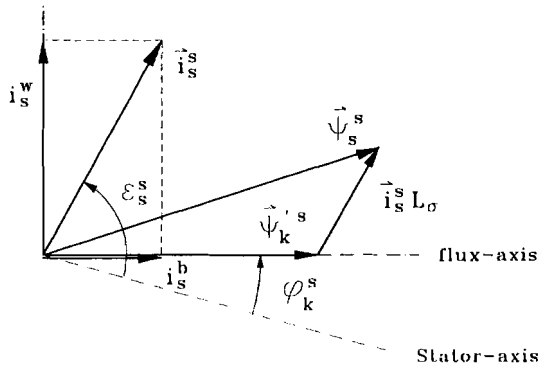


Figure 3.2: vector diagram of flux and current

In the single-cage flux-reference frame the following equations hold [Van '92]:

$$ i_\mu^{'} = i_s^b + i_k^{'b} \tag{3-4} $$

$$ o = i_s^w + i_k^{'w} \tag{3-5} $$

$$ -R_k^{'} i_k^{'b} = \dot{\Psi}_k^{'} \tag{3-6} $$

$$ -R_k^{'} i_k^{'w} = \Psi_k^{'} \dot{\varphi}^r \tag{3-7} $$

$$ \Psi_k^{'} = L_m \cdot i_\mu^{'} \tag{3-8} $$

$$ M_{el} = \Psi_k^{'} \cdot i_s^w \tag{3-9} $$

Thus with the stator current in flux-coordinates the flux ${}_{(i)}\hat{\Psi}_k^{'}$ of the I/φ-model can be calculated with the estimated parameters $\hat{L}_m$ and $\hat{R}_k^{'}$ using Eq. 3-4, 3-6 and 3-8:

$$ {}_{(i)}\hat{\bar{\Psi}}_k' = -\int \left(\frac{{}_{(i)}\hat{\bar{\Psi}}_k'}{\hat{L}_m} - \vec{i}_s^{\,b} \right) R_k' \, dt \tag{3-10} $$

This leads to two flux models with independent machine parameters, the U/I-model uses $\hat{R}_s$ and $\hat{L}_\sigma$ and the I/φ-model uses $\hat{L}_m$ and $\hat{R}_k'$. With the assumption that $\hat{R}_s$ is correctly measurable, only 3 parameters have to be estimated for the single-cage situation.
Using the flux difference $\Delta\hat{\Psi}_k = {}_{(u)}\hat{\Psi}_k' - {}_{(i)}\hat{\Psi}_k'$, the parameters $\hat{L}_\sigma$ and $\hat{L}_m$ can be estimated using only stationary conditions. The value of $\hat{R}_k'$ has to be identified under dynamic operation because of the fact that in a stationary situation $i_k^b = 0$.

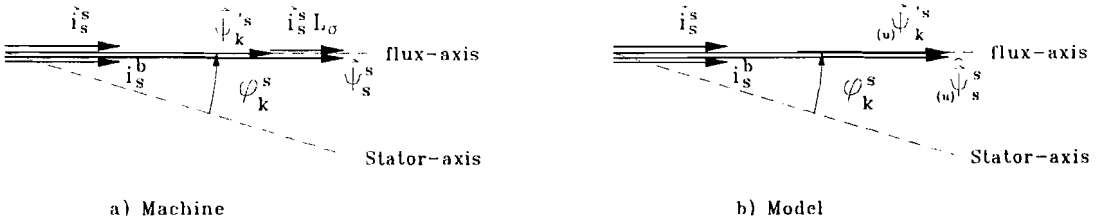
3.1.1. IDENTIFICATION STRATEGY

In the following the tuning procedure for $\hat{L}_\sigma$, $\hat{L}_m$ and $\hat{R}_k'$ will be discussed, using the two flux models. In fact, in order to avoid the problems related to the open integrator in Fig. 3.1 with $\bar{\Psi}_s^s$ as output, a stabilised U/I-model should be applied [Werf '94]. The stator resistance can be measured accurately at stand-still, thus $\hat{R}_s = R_s$. The estimation procedure starts at no-load situation where:

$$ M_{el} = 0 \quad \rightarrow \quad i_s^w = 0 \tag{3-11} $$

The vector diagram of currents and fluxes in the machine and in the auxiliary models at no-load are given in figure 3.3a. In this case the vectors $\vec{i}_s^{\,s}$, $\bar{\Psi}_s^s$ and $\bar{\Psi}_k^{'s}$ are all in one direction. At first the model parameters $\hat{L}_\sigma$ and $\hat{L}_m$ are made equal to zero, which results in the flux estimations as shown in Fig. 3.3b, where:

$$ {}_{(i)}\hat{\Psi}_k' = 0 \quad \text{and} \quad {}_{(u)}\hat{\bar{\Psi}}_k' = \bar{\Psi}_s \tag{3-12} $$



a) Machine b) Model

Figure 3.3: vector diagram at no load, a) machine b) model

The flux difference $\Delta\hat{\Psi}_k$ is therefore positive. Enlarging the value $\hat{L}_m$ until $\Delta\hat{\Psi}_k = 0$ leads to the first estimation of the main inductance:

$$ \hat{L}_m = L_m + L_\sigma \tag{3-13} $$

The second step is to operate the machine at half load. The steady-state vector diagram of the machine and the models, is depicted in Fig. 3.4. Because $\hat{L}_\sigma = 0$ the leakage flux between $\hat{\Psi}_s^s$ and $\hat{\Psi}_k^{'s}$ will be equal to zero. This results in a vector ${}_{(i)}\hat{\Psi}_k'$ which will be to long, with an angle $\hat{\varphi}_k^s$ larger then φ_k^s (of the machine). The angle between the estimated flux and the stator axis, which is too large, will lead to a vector component $\hat{i}_s^b$ which is also larger than the real value. Due to the large value of $\hat{i}_s^b$ and the large first estimation of $\hat{L}_m$, the estimated flux value of the I/φ-model will be larger than the flux estimated by the U/I-model:

$$ {}_{(i)}\hat{\Psi}_k' > {}_{(u)}\hat{\Psi}_k' \quad \rightarrow \quad \Delta\hat{\Psi}_k < 0 \tag{3-14}$$

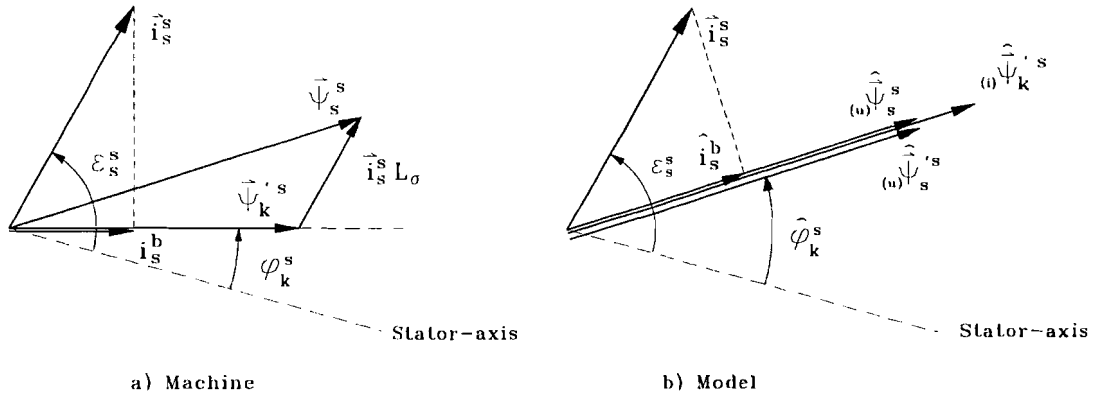


Figure 3.4: vector diagram at load, a) machine b) model

Enlarging $\hat{L}_\sigma$ will lead to an angle reduction and a larger estimated flux ${}_{(u)}\hat{\Psi}_k'$. When the flux difference is reduced to zero, the estimated leakage inductance will be larger than the actual value $\left(\hat{L}_\sigma > L_\sigma\right)$.

After this first iteration step both estimated values $\hat{L}_\sigma$ and $\hat{L}_m$ will be larger than the real values. The procedure can be repeated starting again at no-load. In this situation $\Delta\hat{\Psi}_k < 0$ as a result of the large $\hat{L}_m$. Decreasing the value of $\hat{L}_m$ until $\Delta\hat{\Psi}_k = 0$, leads to a better estimation where $\hat{L}_m = \hat{L}_{m(1)} - \hat{L}_{\sigma(1)}$. Due to the fact that $\hat{L}_{\sigma(1)}$ is too large, $\hat{L}_{m(2)} < L_m$. Under load condition $\hat{L}_\sigma$ has to be adjusted again, which eventually results in a better parameter estimation. The estimation will improve with the number of iterations performed.

3.1.2. OPTIMISATION OF THE METHOD

Altering the above discussed method slightly will make it possible to estimate the "correct" values for $\hat{L}_\sigma$ and $\hat{L}_m$ after just one iteration. This can be done by defining:

$$\hat{L}_{m(load)} = \hat{L}_{m(no-load)} - \hat{L}_\sigma \tag{3-15}$$

Starting at no-load with both $\hat{L}_\sigma$ and $\hat{L}_m$ equal to zero, leads to an estimated value

$$\hat{L}_{m(no-load)} = L_m + L_\sigma \tag{3-16}$$

Under load condition the estimated value $\hat{L}_\sigma$ must be enlarged to eliminate the flux difference. The value $\hat{L}_{m(load)}$ should then automatically be changed to:

$$\hat{L}_{m(load)} = L_m + L_\sigma - \hat{L}_\sigma \tag{3-17}$$

When in this case the model reaches the same vector geometry as in the machine, the estimated parameters will have the following values:

$$\hat{L}_\sigma = L_\sigma \tag{3-18}$$

$$\hat{L}_{m(load)} = L_m + L_\sigma - L_\sigma = L_m$$

It is obvious that after just one iteration, no-load/load, the right parameter values have been found. This simplification can be made by changing the scheme of Fig.3.1 as depicted in Fig.3.5.

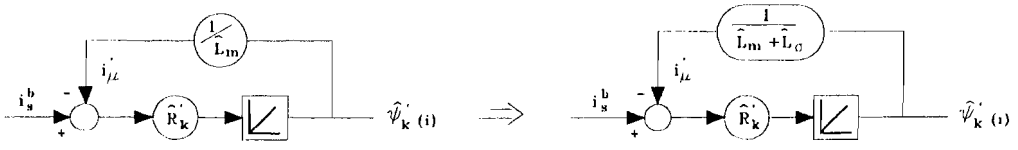


Figure 3.5 : optimisation of the scheme

3.1.3. ESTIMATION OF THE ROTOR RESISTANCE

As discussed in the previous section, only steady-state stator currents and voltages are needed for estimating the parameter values L_σ and L_m. In order to estimate the rotor resistance the rotor circuit has to be excited, creating a dynamic response.
After the values for $\hat{R}_s, \hat{L}_m$ and $\hat{L}_\sigma$ have been assigned in the two flux models it is obvious that the U/I-model is also valid under transient operation. The I/φ-model however only provides reasonable results during transient operation if the estimated rotor resistance equals the real value $\left(\hat{R}_k' = R_k'\right)$. In that case $\Delta\hat{\Psi}_k = 0$ will hold for both

steady-state and dynamic operation, presuming that $\hat{L}_\sigma$ and $\hat{L}_m$ are correctly estimated. To estimate the rotor resistance the stator voltage is changed stepwise. If $\hat{R}_k' < R_k'$ the time constant $\tau = \frac{\hat{L}_m}{\hat{R}_k'}$ will be too large and as a result the estimated cage flux ${}_{(i)}\hat{\Psi}_k'$ changes to slow. If however $\hat{R}_k' > R_k'$ the time constant is too small and the estimated cage flux ${}_{(i)}\hat{\Psi}_k'$ changes to rapid. This is illustrated in Fig. 3.6.

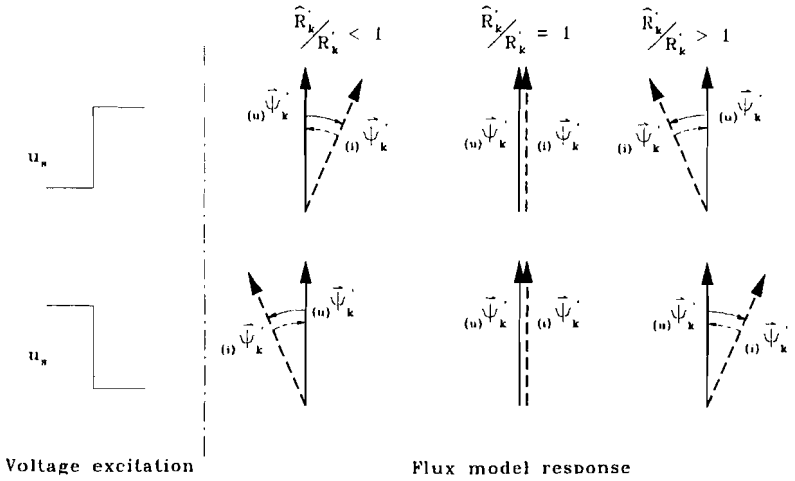


Figure 3.6: effect of deviation in the estimated rotor resistance on the estimated flux difference.

The dynamic responses will serve as a command for changing the value of $\hat{R}_k'$ in the right direction.

3.2. OUTPUT ERROR METHOD

Most estimation methods, both on-line and off-line, use the measured fundamental components of the phase voltages, currents and mechanical speed. In that case information concerning only two parameters can be obtained from each set of experimental data. When using single steady-state experiments, identification of all machine parameters requires varying operating situations (*two-flux-model method*).
A method proposed by [Gort '94], based on the output error method, will be discussed. This method takes into account the available switching harmonics produced by a normal converter and does not require additional external test signals for the purpose of identifying all machine parameters. The results will be compared later with the *two-flux-model method* [Bla '94].

3.2.1. IDENTIFICATION PROCEDURE

The identification approach used to identify the machine parameters is a so-called parametric identification (finite number of parameters). This method is based on the minimisation of a criterion function, which in turn is based on the prediction error method: the mismatch between real output and model output. A model structure has to be chosen and the parameters in this structure will be derived by minimising a cost function.
However, there will always be a purpose in trying to derive an optimal machine model. There are two reasons: machine analysis and control design. In machine analysis we want to predict the machine behaviour in different operation conditions. Usually the analysis is done by simulations. In control design the goal is to develop a controller, which makes the machine behave in a desired way.
In both cases the model must predict the behaviour of the machine, therefore a very logical quality criterion is a measure for it's prediction capacity. This capacity of prediction can be quantified by comparing the actual output with the calculated future output. If the error between real and predicted output is small the model will be good. Methods that are based on this principle are called "prediction error methods".

Parameter identification deals with the problem of finding numerical values for the parameters resulting in the "best" description of a dynamic system, in this case an asynchronous machine. The dynamic properties of a system must be extracted from the observed input and output only. These in- and output signals are sampled with a sample time T_s. The Nyquist frequency is then equal to:

$$\omega_{nyq} = 2\pi\left(\tfrac{1}{2} f_s\right) = \frac{\pi}{T_s} \tag{3-19}$$

The measured system output at time $k \cdot T_s$ will be noted as $y(k)$, the input been denoted as $u(k)$. The output of the system is disturbed by an additive noise term v at the output, according to Fig. 3.7.

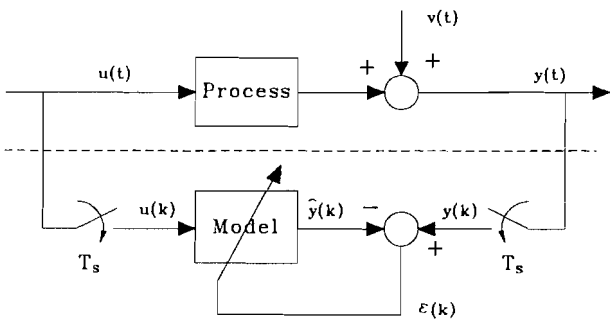


Figure 3.7: system representation

For strictly proper processes, the model output is denoted as $\hat{y}(k|k-1)$ which is called the one-step ahead prediction of $y(t)$, based on measurements up to $t=k-1$. If the process is not strictly proper, which is the case for the used machine models, the model output is no longer called a prediction but an estimation of the process output using input $u(t)$ measurements up to $t=k$ and output measurements $y(t)$ up to $t=k-1$.
Comparing the model output $\hat{y}(k|k)$with the real process output, leads to the so called output error:

$$\varepsilon(k|k) = y(k) - \hat{y}(k|k) \tag{3-20}$$

The model will be good if it has a small output error, but for one model the output error can be small at one instant and large for another. Therefore it's better to look at a sequence of prediction errors. To prevent positive and negative prediction errors from cancelling out, the square has to be taken. This results in the following loss function:

$$J_N = \frac{1}{N}\sum_{k=1}^{N}\varepsilon(k|k)^2 \tag{3-21}$$

The "best" model minimises this loss function. The model parameters are collected in the parameter vector θ. Within a selected model structure, the optimal model will have a parameter vector that minimises the loss function:

$$\hat{\theta}_N = \arg\min_{\theta \in D} J_N(\theta) \tag{3-22}$$

with

$$J_N(\theta) = \frac{1}{N}\sum_{k=1}^{N}\varepsilon^2(k|k,\theta) \tag{3-23}$$

The set D is the set of possible parameter vectors. This could be, for example, the set of parameter vectors that result in a stable model. It is clear that, to compute the prediction error, an estimation of the process output is needed.

3.2.2. ESTIMATION OF THE PROCESS OUTPUT

Prediction error methods are based on the estimation of the process output, using a process model. We will use the standard model representation as depicted in Fig. 3.8.

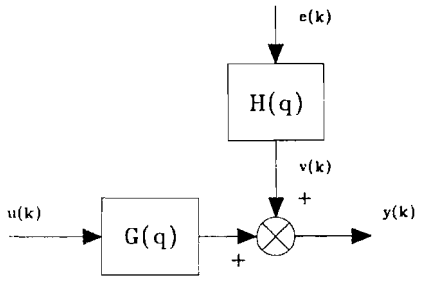


Figure 3.8: standard model representation

The process output can be written as follows:

$$y(k) = G(q)u(k) + H(q)e(k) = G(q)u(k) + v(k) \tag{3-24}$$

$$y(k) = \left[\sum_{n=0}^{\infty} g_n q^{-n}\right] u(k) + \left[\sum_{n=0}^{\infty} h_n q^{-n}\right] e(k) \tag{3-25}$$

Where $G(q)$ is the process model and $H(q)$is the noise model of the output signal. If the system is not strictly proper, the process model $G(q)$ is written as:

$$G(q) = g_0 + g_1 q^{-1} + g_2 q^{-2} \ldots. \tag{3-26}$$

The noise is defined by:

$$H(q) = 1 + h_1 q^{-1} + h_2 q^{-2} \ldots\ldots\ldots \tag{3-27}$$

with $h_0 = 1$. The signal $e(k)$ is a Zero-Mean White-Noise (ZMWN) sequence and q represents a shift operator, defined as:

$$q^n[u(k)] = u(k+n) \text{ and } q^{-n}[u(k)] = u(k-n) \tag{3-28}$$

From measurements, the set $\{y(l-1), u(l)\}$ for $l = k, k-1, \ldots\ldots$ is known. If $G(q)$ and $H(q)$ are also known, then $v(l)$ for l= k-1,k-2,... is known, since:

$$v(l) = y(l) - G(q)u(l) \tag{3-29}$$

The model output estimation at time t=k is then given by:

$$\hat{y}(k|k) = G(q)u(k) + \hat{v}(k|k-1) \tag{3-30}$$

If we assume that $G(q)$ is known and not strictly proper, $G(q)u(k)$ will contain values up to k and is therefore also known at time instant $t=k$. We have to obtain $\hat{v}(k|k-1)$ to calculate the model output. To do this, the noise filter $H(q)$ must satisfy three conditions; $H(q)$ must be

- stable,
- inversely stable
- monic ($H(q) = 1 + h_1 q^{-1} + h_2 q^{-2} + \ldots\ldots$)

The monic condition implies that the noise process can be written as an impulse response representation

$$v(k) = e(k) + \sum_{l=1}^{\infty} h_l \, e(k-l) \tag{3-31}$$

The best value of the coloured noise term $v(k)$ will be the expectation of Eq. 3-31.

$$\hat{v}(k|k) = E\{v(k)\} = E\{e(k)\} + E\left\{\sum_{l=1}^{\infty} h_l \, e(k-l)\right\} \tag{3-32}$$

Since $e(k)$ is a zero mean white noise sequence the first part is expected to be zero. The second part is deterministic because $v(l)$ and $e(l)$ are assumed to be known for $l \leq k-1$. This results in:

$$\hat{v}(k|k-1) = \sum_{l=1}^{\infty} h(l) \cdot e(k-l) = [H(q) - 1]e(k) \tag{3-33}$$

Since $H(q)$ is inversely stable, $H^{\perp}(q) = 1 + h_1' q^{-1} + h_2' q^{-2} \ldots\ldots$ exists such that:

$$H^{\perp}(q)H(q) = H(q)H^{\perp}(q) = 1 \tag{3-34}$$

From Eq. 3-24 and 3-34 follows that:

$$e(k) = H^{\perp}(q)v(k) \tag{3-35}$$

which, together with Eq. 3-33, yields:

$$\hat{v}(k|k-1) = [H(q) - 1]\, H^{\perp}(q)v(k) = [1 - H^{\perp}(q)]v(k) \tag{3-36}$$

Thus for calculation of the prediction of the noise term $v(k)$ only values of $v(l)$ up to $l \leq k-1$ are needed. The model output $G(q)u(l)$ uses input values up to $l \leq k$, which results in the following system output estimation:

$$\hat{y}(k|k) = G(q)u(k) + [1 - H^{\perp}(q)]v(k-1) \tag{3-37}$$

Since $v(l) = y(l) - G(q)u(l)$ for $l \leq k-1$, the output estimation can be written as:

$$\begin{aligned}\hat{y}(k|k) &= G(q)u(k) + \left[1 - H^{\perp}(q)\right]\left[y(k-1) - G(q)u(k-1)\right] \\ &= H^{\perp}(q)G(q)u(k) + \left[1 - H^{\perp}(q)\right]y(k-1)\end{aligned} \tag{3-38}$$

The error between estimated system output and the measured system output can now be rewritten as follows

$$\begin{aligned}\varepsilon(k|k) &= y(k) - \hat{y}(k|k) = \\ &= -H^{\perp}(q)G(q)u(k) + H^{\perp}(q)y(k-1)\end{aligned} \tag{3-39}$$

This expression is valid for any model structure that can be written as

$$y(k) = G(q)\cdot u(k) + H(q)\cdot e(k) \tag{3-40}$$

where $G(q)$ is not restricted with respect to the strictly proper properties. This general expression for the prediction of *y(k)* is valid for all model structures. However, different model structures can be used to describe a process.

3.2.3. MODEL SET STRUCTURE

The method used for machine identification is a so-called parametric identification. For parametric identification a process model structure has to be chosen a priori. Therefore different parametrisations for the process model G(q) and noise model H(q) in Eq. 3-40 exist [Bosch '94], such as:

- Finite Impulse Response model (FIR);
- Auto Regressive with eXogenous input model (ARX);
- Auto Regressive Moving Avarage with eXogenous input model (ARMAX);
- Output Error model (OE);
- Box-Jenkins model.

These models are special cases of the general process model:

$$A(q)y(k) = F^{\perp}(q)B(q)u(k) + D^{\perp}(q)C(q)e(k) \tag{3-41}$$

For estimating machine parameters an Output Error model is preferable because the noise model is not estimated separately. In machine models used for control development and analysis no noise model is used either, so this will be a good choice.

OE model

The Output Error model is given by:

$$\begin{aligned}F(q)x(k) &= B(q)u(k) \\ y(k) &= x(k) + e(k)\end{aligned} \tag{3-42}$$

In this case the process model and noise model are defined as:

$$G(q) = F^{\perp}(q)B(q) \qquad H(q) = 1 \tag{3-43}$$

Using Eq. 3-38 the OE estimation model can then be written as:

$$\hat{y}(k|k) = F^{\perp}(q)B(q)\cdot u(k) \tag{3-44}$$

with $B(q)$ and $F(q)$ representing sequences of the shift operator q defined as:

$$F(q) = 1 + f_1 q^{-1} + f_2 q^{-2} .... f_{N_f} q^{-N_f} \tag{3-45}$$

$$F^{\perp}(q) = 1 + f_1' q^{-1} + f_2' q^{-2} .... \tag{3-46}$$

$$B(q) = b_1 q^{-1} + b_2 q^{-2} .... b_{N_b} q^{-N_b} \tag{3-47}$$

The coefficients of the sequences *B(q)* and *F(q)* are collected in the parameter vector θ defined by:

$$\theta = \left[f_1 ..... f_{N_f}\ b_1 ..... b_{N_b} \right]^T \tag{3-48}$$

The estimated output signal is not a linear function of the parameter vector θ and therefore the OE estimation model is called a pseudo-linear regression model. This model can be written as a pseudo-linear regression model:

$$\hat{y}(k,\theta(i)) = \phi(k,\theta(i))\cdot\theta(i) \tag{3-49}$$

with $\theta(i)$denoting the parameter vector after i iterations. The regression vector is defined by:

$$\phi(k,\theta(i)) = \left[-\hat{y}(k-1,\theta(i)), ... , -\hat{y}(k-n_f,\theta(i)),\ u(k),\ ....., u(k-n_b) \right] \tag{3-50}$$

The regression vector ϕ contains the measured input $u(l)$ up to $l=k$ and the estimated system output $\hat{y}(l|\theta)$ for $l \le k-1$.

A criterion function which is often used is a quadratic function of the filtered prediction error:

$$J(\theta(i)) = \frac{1}{2k} \sum_{m=1}^{k} \left\{ \sum_{l=0}^{\infty} L_l\ \varepsilon(m-l) \right\}^2 \tag{3-51}$$

The predicted error is filtered with a stable linear filter $L(q)$, where the choice of the filter can influence the criterion function and therefore the estimated parameters. This is made clear when the criterion function is written in the frequency domain [Lju '87]:

$$J(\theta(i)) = \frac{1}{2\pi} \int_0^{\omega_{nyq}} \left[\left\| G_0(\omega) - \frac{B(\omega)}{F(\omega)} \right\|^2 \Phi_u(\omega) + \Phi_v(\omega) \right] \cdot \left\| L(\omega) \right\|^2 d\omega \tag{3-52}$$

where $G_0(\omega)$represents the process we want to identify. The model G will, in practice, never be able to describe the system G_0 exactly. The model G will always be an approximation of the real process.

The coefficients of the parameter vector $\hat{\theta}$ are the values that make G the best approximation of G_0 with a frequency weighting $\left\| L(\omega) \right\| \Phi_u(\omega)$. Using this frequency

weighting function, the errors in the different frequency ranges can be weighted. If the term $\|L(\omega)\|\Phi_u(\omega)$ is large, then the part

$$\left\|G_0(\omega)-\frac{B(\omega)}{F(\omega)}\right\| = \left\|G_0(\omega)-G(\omega,\theta(i))\right\| \tag{3-53}$$

in the criterion function is relative important and as a result the estimated parameters are tuned in a way that a small error is obtained at those frequencies.

NOISE INFLUENCE

There is no correlation between the noise power spectrum Φ_v and the parameter vector θ. Thus the contribution of the noise term will be the same for all parameter vectors. Consequently, the noise does not affect the estimated parameter vector $\hat{\theta}$. Hence, the estimation is consistent if the optimal process model $G_{opt}(q,\theta)$ is in the model set even when the noise model does not correspond to the real noise process. The estimation of the deterministic contribution $G(q,\theta)$ is separated from the noise contribution $v(k)$ when using an OE-model.
Because the OE-models are pseudo-linear in the parameters, the optimal parameter vector θ can only be found by means of a non-linear optimisation technique.

3.2.4. IDENTIFICATION OF THE MACHINE PARAMETERS

The problem with most identification methods is the fact that the rotor impedance is estimated at the stator side, using only stator voltage and currents. However, according to Eq. 2-91 the rotor resistance seen at the stator side is scaled by the slip:

$$slip = \frac{\omega_s}{\omega_s - \omega_r} \tag{3-54}$$

For sinusoidal signals the locked-rotor equivalent circuit is therefore changed to a non-linear slip dependent equivalent circuit, as depicted in Fig. 3.9.

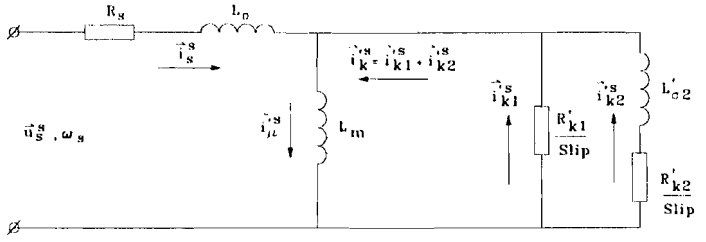


Figure 3.9: slip dependent equivalent circuit seen at the stator side.

This non-linear behaviour of the motor admittance cannot be described by a linear model. To solve this identification problem the stator resistance and the rotor parameters

are separated and the rotor parameters are identified in rotor coordinates, since the relation between the flux $\vec{\Psi}_s^r$ and the current $\vec{i}_s^r$ in rotor coordinates is described during steady-state and in transient situations by a linear input-output function. If the stator resistance R_s and rotor angle ρ^s are known, the stator flux and current can be translated to rotor coordinates from measured stator voltage and current in the following way:

$$\vec{i}_s^r = \boldsymbol{R}\left(-\rho^s\right)\cdot\vec{i}_s^s \tag{3-55}$$

$$\vec{\Psi}_s^r = \boldsymbol{R}\left(-\rho^s\right)\int_0^t\left\{\vec{u}_s^s(\tau)-R_s\vec{i}_s^s(\tau)\right\}d\tau \tag{3-56}$$

The rotor of the induction machine is assumed to be electrically symmetrical, thus the rotor start position can be chosen arbitrarily. For the single-cage rotor model (according to Fig. 2.7) the following rotor transfer function is found:

$$\Psi_s^{ra}(\mathrm{p}) = \frac{L_\sigma L_m\,\mathrm{p} + R_k'(L_\sigma + L_m)}{L_m\,\mathrm{p} + R_k'}\cdot i_s^{ra}(\mathrm{p}) = G_r(\mathrm{p})i_s^{ra}(\mathrm{p}) \quad a=\{1,2\} \tag{3-57}$$

where "p" represents the Laplace operator. However, the output error identification is applicable only to a corresponding discrete model for the rotor transfer function (in z-domain) of the single-cage machine.

$$G(z) = \frac{B(z)}{F(z)} = \frac{b_0 + b_1 z^{-1}}{1 + f_1 z^{-1}} \tag{3-58}$$

The discrete transfer function can be derived from the continuous time transfer function Eq. 3-56. The parameters of the discrete model can be written as a function of the parameters of the continuous model and the sampling time T_s using the Z-transformation, which assumes the use of an ideal Zero-Order Hold circuit. This results in the following relation:

$$\Psi_s^{ra}(z) = G(z)\cdot i_s^{ra}(z) \tag{3-59}$$

with:

$$\begin{aligned} f_1 &= -e^{-T_s\frac{R_k'}{L_m}} \\ b_0 &= L_\sigma \\ b_1 &= L_m + f_1(L_m + L_\sigma) \end{aligned} \tag{3-60}$$

These coefficients can be identified with the prediction error method. The parameter vector θ and the regression vector ϕ are then defined by:

$$\begin{aligned} \theta &= \begin{bmatrix} 1 & f_1 & b_0 & b_1 \end{bmatrix}^T \\ \phi &= \left[-\Psi_s^R(k-1,\theta), -\Psi_s^R(k-2,\theta), I_s^R(k), I_s^R(k-1)\right] \end{aligned} \tag{3-61}$$

where:

$$\Psi_s^R(k) = \Psi_s^{r1}(k) + j \cdot \Psi_s^{r2}(k) \tag{3-62}$$

$$I_s^R(k) = i_s^{r1}(k) + j \cdot i_s^{r2}(k) \tag{3-63}$$

As it will be explained later, it is convenient to combine the components of the space vectors in a complex representation.
The physical machine parameters of the continuous model can then be determined using the unique inverse transformation on the estimated discrete parameters (Eq. 3-60) in the following way:

$$\begin{aligned} L_\sigma &= b_0 \\ L_m &= \frac{b_1 + b_0 f_1}{1 + f_1} \\ R_k' &= -log(-f_1)\frac{L_m}{T_s} \end{aligned} \tag{3-64}$$

For a double-cage rotor as discussed in Chap. 2 the continuous-time flux current relation in rotor coordinates is found to be:

$$\Psi_s^{ra}(\mathrm{p}) = \left[\frac{\frac{L_{\sigma 2} L_m}{R_{k2}'}\mathrm{p} + L_m}{\mathrm{p}^2 \frac{L_m L_{\sigma 2}}{R_{k1}' R_{k2}'} + \mathrm{p}\left(\frac{L_m}{R_{k1}'} + \frac{L_m}{R_{k2}'} + \frac{L_{\sigma 2}}{R_{k2}'}\right) + 1} + L_\sigma \right] \cdot i_s^{ra}(\mathrm{p}) \tag{3-65}$$

For model orders ≥ 2, there are several ways to derive an approximative discrete transfer function from the continuous-time model description, instead of developing exact (but cumbersome) Z-transformations. If the sampling period T_s is short enough compared to the system time constants, Tustin's-transformation defined by

$$\mathrm{p} = \frac{2}{T_s}\frac{1 - z^{-1}}{1 + z^{-1}} \quad \longleftrightarrow \quad z = \frac{1 + \mathrm{p}\left(T_s/2\right)}{1 - \mathrm{p}\left(T_s/2\right)} \tag{3-66}$$

leads to satisfying results. By substituting then Eq. 3-65 into Eq. 3-64 one gets

$$\Psi_s^{ra}(z) = G(z,\theta) \cdot I_s^{ra}(z) \tag{3-67}$$

where $G(z,\theta)$ has the general form for the rotor circuit is given by:

$$\hat{G}(z,\theta) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + f_1 z^{-1} + f_2 z^{-2}} \tag{3-68}$$

The coefficients of the transfer function Eq. 3-68 can be identified with the OE-method. After identification, the estimated discrete time model is transformed again to a continuous transfer function, using the Tustin's-transformation, which results in:

$$\hat{G}(z,\theta)=\frac{b_0+b_1 z^{-1}+b_2 z^{-2}}{1+f_1 z^{-1}+f_2 z^{-2}} \xleftrightarrow{Tustin} \hat{G}(\mathrm{p})=\frac{w_0+w_1 \mathrm{p}+w_2 \mathrm{p}^2}{1+v_1 \mathrm{p}+v_2 \mathrm{p}^2} \tag{3-69}$$

However, it should be clear that the transformation from discrete to continuous time will always be an approximation, where the quality of the parameters is dependent of the used transformation. With the estimated continuous transfer function the continuous double-cage machine parameters, Eq. 3-65, can be derived using the following unique transformation:

$$\hat{L}_\sigma = w_2 / v_2 \tag{3-70a}$$

$$\hat{L}_m = w_0 - L_\sigma \tag{3-70b}$$

$$\hat{R}'_{k1} = \hat{L}_m \frac{w_1 - \hat{L}_\sigma v_1}{w_1 \hat{L}_m} \tag{3-70c}$$

$$\tau_1 = \frac{\hat{L}_m}{\hat{R}'_{k1}} \quad , \quad \tau_2 = \frac{w_1 - \hat{L}_\sigma v_1}{\hat{L}_m} \tag{3-70d}$$

$$\hat{R}'_{k2} = \frac{\hat{L}_m}{v_1 - \tau_1 - \tau_2} \tag{3-70e}$$

$$\hat{L}_{\sigma 2} = \hat{R}'_{k2} \tau_2 \tag{3-70f}$$

These continuous machine parameters can be used in the double-cage machine simulation model discussed in Chap. 2.

3.2.5. OFF-LINE DATA PRE-PROCESSING

When using an inverter, such as PWM-VSI, cyclo-converter, matrix-converter, the stator voltage and current will contain one dominant frequency, i.e. the fundamental supply voltage frequency ω_0 and a large number of switching harmonics. As a result of the transformation to rotor coordinates the frequency spectrum is shifted by the mechanical velocity $\omega_m = \dot{\rho}^s$ as follows:

$$\Phi_{i_s^R}(\omega) = \Phi_{i_s^S}(\omega - \omega_m) \qquad \Phi_{\psi^R}(\omega) = \Phi_{\psi^S}(\omega - \omega_m) \tag{3-71}$$

The most dominant frequency in rotor coordinates is the slip frequency $\omega_{slip} = \omega_0 - \omega_m$. The slip frequency is dependent of the load torque M_{load}. In steady-state the load torque is equal to the machine torque. Under no-load condition the slip frequency is

normally very low and because the data set must contain at least one period of the rotor frequency the data set must be very long. For identification it is therefore preferable that the machine operates under load condition with higher slip frequencies (1-2 Hz).

The measured stator voltages and currents can not be directly used in the identification algorithm. The signals are usually disturbed by:

- drift and offset, thus low frequency disturbances;
- spikes, due to cross talk between cables ;
- high frequency noise, due to sensors and AD conversion.

The pre-processing of the collected data set involves the following processing steps, to make the data suitable for identification purposes:

- trend determination and correction;
- offset correction;
- peak shaving (clipping);
- noise reduction, low pass filtering.

As mentioned before, the proposed identification method [Gort '94] is based on the reconstruction of the flux $\vec{\Psi}_s^r$ and current $\vec{i}_s^r$ in rotor coordinates. An important data pre-processing task is thus the reconstruction of these signals from measured stator voltage $\vec{u}_s^s$, current $\vec{i}_s^s$ and rotor position ρ^s. The stator current and flux are derived, using the following equations:

$$I_s^R = e^{\left(-j\rho^s\right)} \cdot I_s^S \tag{3-72}$$

$$\Psi_s^R = e^{\left(-j\rho^s\right)}\left[\Psi_s^S(0) + \int_0^t \left\{U_s^S(\tau) - R_s I_s^S(\tau)\right\} d\tau\right] \tag{3-73}$$

In the equations above, the components of the two dimensional space vectors

$$\vec{u}_s^s = \begin{bmatrix} u_s^{s1} \\ u_s^{s2} \end{bmatrix} \quad \text{and} \quad \vec{i}_s^s = \begin{bmatrix} i_s^{s1} \\ i_s^{s2} \end{bmatrix} \tag{3-74}$$

have been represented as complex numbers defined as:

$$I_s^S = i_s^{s1} + j\, i_s^{s2} \tag{3-75}$$

$$U_s^S = u_s^{s1} + j\, u_s^{s2} \tag{3-76}$$

The procedure of deriving the flux in rotor coordinates is sensitive to offset errors, because offset causes drift in the integration. Therefore offset must correctly be compensated and integration has to be accurate.

Due to the fact that the machine is driven by an inverter, the measured voltage and current will have many different frequency components. To avoid aliasing the continuous signals must be filtered with a continuous anti-aliasing filter before the

signals are converted to digital signals. The N measured data points are presented in the data set Z_N. The data set contains the digitised measured voltage $U_s^S(kT_s)$, the measured current $I_s^S(kT_s)$ and the measured rotor position $\rho^s(kT_s)$ or velocity $\dot{\rho}^s(kT_s)$. First the data is clipped to reduce the effect of outliers (spikes). If the mechanical velocity is measured the position can be derived by discrete integration. The start position can be chosen arbitrarily if the machine is symmetrical, which is assumed throughout this work.

In steady-state the electrical signals are periodic and the data pre-processing can be performed in the frequency domain, if the data set contains sufficient periods of the fundamental stator frequency f_0. The fundamental component is calculated by counting the number of samples between two zero crossings of both stator voltages after low-pas filtering u_s^{s1} and u_s^{s2}. The number of full periods N_p of the fundamental frequency component in the data set is determined as follows:

$$N_p = N\frac{f_0}{f_s} = N T_s f_0 \qquad N_p \in \mathrm{N} \tag{3-77}$$

The data set is now reduced to N_p whole periods with a length of m samples, where m is found to be:

$$m = \frac{N_p}{f_0 T_s} \tag{3-78}$$

The new data set is denoted as Z_m. The distance Δf between two observable frequencies is related to the total time duration of the data set, and is given by:

$$\Delta f = \frac{1}{mT_s} = \frac{f_0}{N_p} \tag{3-79}$$

Next, the derivative of the stator flux Ψ_s^S is calculated as follows:

$$\dot{\Psi}_s^S(kT_s) = U_s^S(kT_s) - R_s I_s^S(kT_s) \tag{3-80}$$

In this expression the stator resistance is used. The stator resistance is however easy measurable and therefore presumed to be known. The discrete Fourier transformation will be performed on this stator flux derivative $\dot{\Psi}_s^S$ and stator current I_s^S. The highest observable frequency component in the data set is the Nyquist frequency f_{nyq} :

$$f_{nyq} = \frac{1}{2T_s} \tag{3-81}$$

The spectra of the flux derivative and current in rotor coordinates is calculated in the following way:

$$\Phi_{I_s^S}(n\Delta f) = \sum_{k=1}^{m} I_s^S(kT_s) e^{-j2\pi n\Delta f kT_s} \tag{3-82}$$

$$\Phi_{\dot{\Psi}_s^S}(n\Delta f)=\sum_{k=1}^{m}\dot{\Psi}_s^S(kT_s)e^{-j2\pi n\Delta f kT_s} \tag{3-83}$$

with: $-\frac{m}{2}+1\le n\le\frac{m}{2}$

The following *data pre-processing* steps are performed on these flux and current spectra.

- offset compensation;
- low pass filtering;
- noise reduction;
- stator flux reconstruction.

Offset compensation is done by removing the low frequency part of the spectrum. All frequencies below f_{offset} are removed by setting the spectrum to zero.

$$\Phi(i\Delta f)=0 \quad \text{for} \quad i\in\left\{\frac{-f_{offset}}{\Delta f},\frac{f_{offset}}{\Delta f}\,|\,i\in Z\right\} \tag{3-84}$$

Low pass filtering to remove the high frequency components is done by setting all frequencies above f_{band} to zero.

$$\Phi(i\Delta f)=0 \quad \text{for} \quad i\in\left\{\frac{-m}{2}+1,\frac{-f_{band}}{\Delta f}\wedge\frac{f_{band}}{\Delta f},\frac{m}{2}\,|\,i\in Z\right\} \tag{3-85}$$

Noise reduction is done by removing the frequency components with a small amplitude relative to the amplitude of the fundamental component. The following criteria is used:

$$20\ log\frac{\|\Phi(i\Delta f)\|}{\|\Phi(f_0)\|}\le N_{noise}\rightarrow\begin{cases}\Phi_{I_s^S}(i\Delta f)=0\\ \Phi_{\Psi_s^S}(i\Delta f)=0\end{cases} \tag{3-86}$$

The *stator flux reconstruction* can be accurately performed in the frequency domain. Integration in the frequency domain is equal to multiplying the spectrum with $\frac{1}{j\omega}=\frac{1}{j2\pi n\Delta f}$, resulting in :

$$\Phi_{\Psi_s^S}(i\Delta f)=\frac{1}{j2\pi i\Delta f}\Phi_{\dot{\Psi}_s^S}(i\Delta f) \quad \text{for} \quad i\in\left\{\frac{-m}{2}+1,\frac{m}{2}\,|\,i\in Z \quad i\neq 0\right\} \tag{3-87}$$

Another advantage of integrating in the frequency domain is that no initial condition has to be defined. It is however important to perform all operations both on the current spectrum and the spectrum of the flux derivative, to avoid phase shift between the two signals. The signals I_s^S and Ψ_s^S in time domain are then derived using the inverse Fourier transformation.

$$I_s^S(kT_s) = m \sum_{n=-\frac{m}{2}+1}^{\frac{m}{2}} \Phi_{I_s^S}(n\Delta f) e^{j2\pi\Delta f T_s} \tag{3-88}$$

$$\Psi_s^S(kT_s) = m \sum_{n=-\frac{m}{2}+1}^{\frac{m}{2}} \Phi_{\Psi_s^S}(n\Delta f) e^{j2\pi\Delta f T_s} \tag{3-89}$$

with $1 \le k \le m$.

The input signals for identification of the rotor circuit, the stator flux and stator current in rotor coordinates are eventually found by rotating the complex values in the complex plane.

$$\begin{aligned} I_s^R(kT_s) &= e^{-j\rho^S(kT_s)} \cdot I_s^S(kT_s) \\ \Psi_s^R(kT_s) &= e^{-j\rho^S(kT_s)} \cdot \Psi_s^S(kT_s) \end{aligned} \tag{3-90}$$

These pre-processed values are collected in the pre-processed data set Z_N defined as:

$$Z_N = \left[I_s^R(T_s), \ldots\ldots, I_s^R(NT_s), \Psi_s^R(T_s), \ldots\ldots, \Psi_s^R(NT_s) \right] \tag{3-91}$$

In order to perform the identification only the real part (with coordinate R1) or the imaginary part (with coordinate R2) of the data set is enough, because there will be no difference in the identification results if the machine is symmetrical.

3.2.6. OFF-LINE IDENTIFICATION ALGORITHM

The optimal estimation of the parameter vector $\hat{\theta}$ can only be found analytically if the predicted output is linear in θ. This is however not the case for the output error models. Thus the optimal solution must be obtained with an iterative search (numerical optimisation). An optimisation method used in this study is the *Newton-Raphson* algorithm, given by:

$$\hat{\theta}(i+1) = \hat{\theta}(i) - \alpha \left[J''\left(\hat{\theta}(i)\right) \right]^{-1} J'\left(\hat{\theta}(i)\right) \tag{3-92}$$

The estimated parameter vector $\hat{\theta}(i)$, after i iterations, is updated every iteration using the first and second derivative J' and J'' of the *loss-function* J (Eq. 3-51) with respect to the parameter vector θ. Because an off-line identification method is used the information of the whole data-set is used for every iteration.

The term $\left[J''(\hat{\theta}(i))\right]^{-1} J'(\hat{\theta}(i))$ gives the direction of the parameter change. With the variable α the step size will be controlled. Using the loss-function Eq. 3-51 the derivatives are written as:

$$J'(\theta) = \frac{d}{d\theta} J(\theta) = -\frac{1}{2N} \sum_{k=1}^{N} \chi(k,\theta)\varepsilon(k,\theta) \tag{3-93}$$

with:

$$\varepsilon(k|k) = \Psi_s^{Rl}(k) - \hat{\Psi}_s^{Rl}(k|k) \tag{3-94}$$

$$\chi(k,\theta) = \frac{d}{d\theta}\varepsilon(k,\theta) = \frac{d}{d\theta}\Psi_s^{Rl}(k,\theta) \tag{3-95}$$

For the second derivative of the loss-function this yields:

$$J''(\theta) = \frac{d}{d\theta} J'(\theta) = \frac{1}{2N} \sum_{k=1}^{N} \left[\chi(k,\theta)\chi^T(k,\theta) - \frac{d}{d\theta}\left(\chi^T(k,\theta)\right)\varepsilon(k,\theta)\right] \tag{3-96}$$

Close to the optimal parameter vector value, the prediction errors will mainly determine a noise term. The prediction errors are therefore independent and the last term of Eq. 3-95 will become zero. This approximation is used in the Gauss-Newton method. With the rewritten search direction the algorithm becomes:

$$\hat{\theta}(i+1) = \hat{\theta}(i) + \alpha\left[\sum_{k=1}^{N} \chi\left(k,\hat{\theta}(i)\right)\chi^T\left(k,\hat{\theta}(i)\right)\right]^{-1} \sum_{k=1}^{N}\left[\chi\left(k,\hat{\theta}(i)\right)\varepsilon\left(k,\hat{\theta}(i)\right)\right] \tag{3-97}$$

Thus, for the iterative search, the gradient of the prediction error $\chi(k,\theta)$ has to be known. For the output error approach the gradient $\chi(k,\theta)$ can be written as follows:

$$\chi(k,\theta) = \frac{d}{d\theta_i}\hat{\Psi}_s^{Rl}(k,\theta) = F^{\perp}(q)\phi(k,\theta) \tag{3-98}$$

with the regression vector $\phi(k,\theta)$ defined as:

$$\phi(k,\theta) = \left[-\hat{\Psi}_s^{Rl}(k-1,\theta),\ldots,-\hat{\Psi}_s^{Rl}(k-N_f,\theta), i_s^{Rl}(k),\ldots,i_s^{Rl}(k-N_b)\right]^T \tag{3-99}$$

The realisation of the pre-processing and identification algorithm is done with the mathematical program "MATLAB". Using the MATLAB *System Identification Toolbox* the output error model is estimated with the M-file "OE.m", which returns a discrete parameter vector which minimises the loss-function (Eq. 3-51). This discrete parameter vector is then transformed in a continuous parameter vector, using the *Tustin's*-transformation. The estimated continuous parameter vector is used to derive the continuous machine parameters according to Eq. 3-70.

The inputs of the M-file are the pre-processed data set Z_N and the desired model order of numerator and denominator. The M-files used for pre-processing and identification are given in Appendix G.

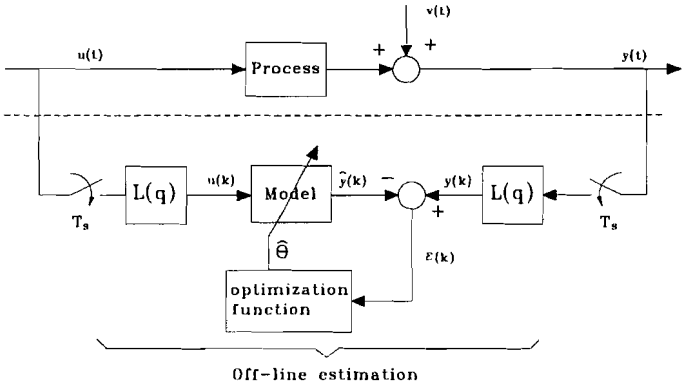


Figure 3.37: Output Error identification scheme

For identification purposes, the filtered output error is used. For linear SISO (single input single output) systems this is equivalent to filtering the input and output with this filter, as depicted in Fig. 3.10.

4. SIMULATION RESULTS

To validate the identification methods discussed in Chap. 3, the two methods are tested on simulation models. Using simulation models is the only way to validate the methods, because only in this case the parameters are known. Both single- and double-cage models are used for the validation of the methods.

4.1. MACHINE MODELS

In Chap. 2 a dynamic machine model (double-cage) is presented. The simulation model is implemented in the simulation program "PSI/c". To test the identification methods the parameters of a single-cage simulation model (SCaSM) and a double-cage simulation model (DCaSM) are estimated. In table 4.1 the parameters of the used single-cage machine model are given. The double-cage machine model parameters are given in Table 4.2.

Table 4.1: parameters of the SCaSM

parameter	value	
	[si]	[pu]
L_m	33.75 mH	3
L_σ	6.52 mH	0.58
R_k'	0.141 Ω	0.04
R_s	0.212 Ω	0.06

Table 4.2: parameters of the DCaSM

parameter	value	
	[si]	[pu]
L_m	33.09 mH	2.941
L_σ	2.012 mH	0.179
R_{k1}'	2.718 Ω	0.769
R_{k2}'	0.136 Ω	0.039
$L_{\sigma 2}'$	5.406 mH	0.481
R_s	0.212 Ω	0.06

For identification with the *two-flux-model method* (TFM-method) the machines are fed by sinusoidal voltage supply. In case of the *output error method* (OE-method) it is necessary to use a power supply, with sufficient harmonic content. The method is therefore performed on a data-set derived form an inverter fed machine model. The inverter used for the simulations is described in Sec. 2.5. The per unit reference values used in the simulations are denoted in Table 4.3.

Table 4.3: per unit reference values

Reference	Value
Voltage	311 V
Current	88 A
Frequency	50 Hz
Resistance	3.53 Ω
Inductance	11.25 mH

4.3. TWO-FLUX-MODEL METHOD

To test the TFM-method (Sec. 3.1) on the simulation models. The identification method and the machine models are implemented in the simulation program "PSI/c". The program is given in Appendix E1.
The method is performed applying the following test conditions:

- simulation time step $100\mu s$;
- stator frequency = 40 Hz;
- stator voltage amplitude = 311 V;
- load condition = 50% of rated torque;
- voltage step = 100% ↔ 80% of rated voltage.

4.3.1. SINGLE-CAGE ESTIMATION

Performing the optimised TFM-method on the SCaSM resulted after some iterations in the parameter estimations, given in Table 4.4. The stator resistance R_s is assumed to be known and no saturation effects are included in the models.

Table 4.4: TFM-method, single-cage estimation of the SCaSM ($R_s = 0.212\,\Omega$)

parameter	SCaSM		estimation		error
	[si]	[pu]	[si]	[pu]	[%]
$\hat{L}_m$	33.75 mH	3	33.66 mH	2.99	3
$\hat{L}_\sigma$	6.52 mH	0.58	6.461 mH	0.574	0.9
$\hat{R}_k'$	0.141 Ω	0.04	0.142 Ω	0.0402	0.7

Using the optimised estimation procedure still a few iteration steps are needed to get a good estimation of main inductance L_m and the leakage inductance L_σ. Estimation of the rotor resistance R_k', using the stator voltage step procedure, is more time consuming. In the simulation the flux-deviation between the two estimated flux-values can be minimised very accurately, however more iteration steps will then be necessary.

4.3.2. SKIN EFFECT

In some situations it can be found useful to approximate a machine with significant skin-effect by a single-cage model. To test the usefulness of such an approximation, the TFM-method was tested on the double-cage simulation model (Table 4.2). Thus a single-cage model is matched to a higher order system (under modelling). To validate the estimated parameters the behaviour of the estimated and simulation model in their normal operation range is studied. The parameters of the estimated single-cage model are found using the procedure discussed in Sec. 3.1. The resulting parameters are given in Table 4.5.

Table 4.5: TFM-method, single-cage estimation of the sinusoidal fed DCaSM ($R_s = 0.212\,\Omega$)

parameter	value	
	[si]	[pu]
$\hat{L}_m$	29.08 mH	2.615
$\hat{L}_\sigma$	6.355 mH	0.565
$\hat{R}_k'$	0.105 Ω	0.030

This parameter estimation is performed with sinusoidal voltage supply. However the method can also be applied to a inverter-fed machine. In Sec. 2.5. the inverter model has been discussed. For the simulations the following inverter properties are applied:

- fundamental frequency $f_0 = 40Hz$;
- DC-voltage $U_{dc} = 538V$;
- Pulse number $pn = 3$.

The three phase inverter voltages are transformed to the two-phase representation and applied to the machine model and TFM-method. In Fig 4. the applied stator voltages, in stator coordinates s1 and s2, are depicted.

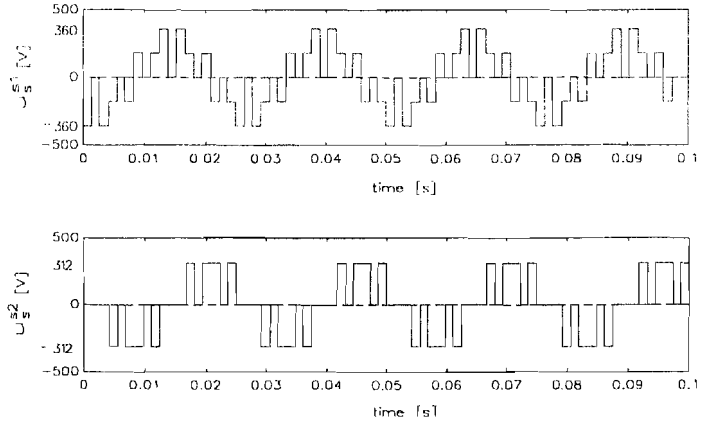


Figure 4.1: applied two-phase stator voltages to test the TFM-method at PWM-inverter supply.

To estimate the cage resistance R_k' a voltage step has to be applied. This is done by changing the DC-voltage between 100-80 % of the rated voltage.

As a result of the switching harmonics the flux difference $\Delta\Psi_k'$, necessary for tuning the model parameters, has a high harmonic content. For accurate estimation the DC-component of this flux must be tuned to zero, therefore the estimated flux-deviation is filtered with a 4th order Butterworth filter, with cut-off-frequency $f_c = 50Hz$. The resulting estimated single-cage parameters are given in Table 4.6.

Table 4.6: TFM-method, single-cage estimation of the inverter fed DCaSM ($R_s = 0.212\Omega$)

parameter	value	
	[si]	[pu]
$\hat{L}_m$	29.41 mH	2.613
$\hat{L}_\sigma$	5.737 mH	0.511
$\hat{R}_k'$	0.106 Ω	0.030

It is clear that the estimated leakage inductance $\hat{L}_\sigma$ is considerable different in both estimations. This is due to the effect of the higher harmonics on the double-cage.

The steady-state and dynamic behaviour of both estimated single-cage model results are compared with the double-cage reference model to study the usefulness of this single-cage approximation. In Fig 4.2 the steady-state speed/torque characteristic of the DCaSM and the estimated single-cage models are given, at sinusoidal voltage supply. It shows that only in the normal operation range of the machine, at small slip frequencies, the characteristics are almost equal.

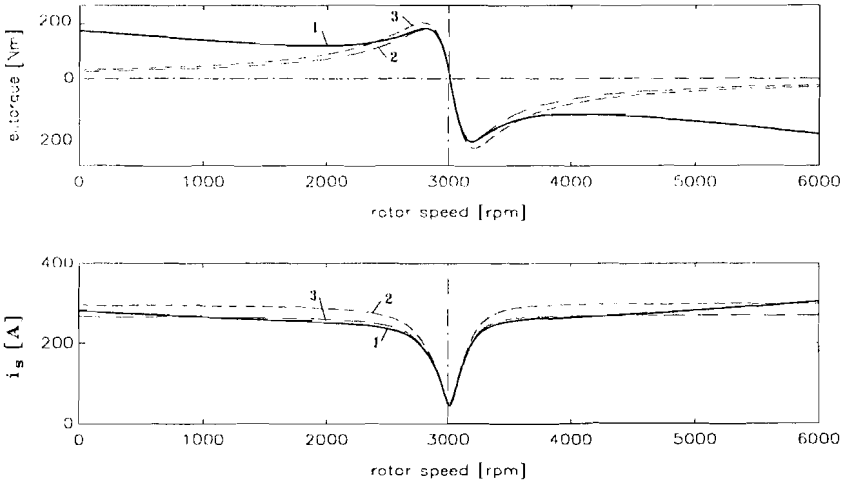


Figure 4.2: steady-state speed/torque characteristic of the double-cage machine (1); the estimated single-cage machine using sinusoidal voltages (2) and estimated single-cage using PWM-supply (3).

In Fig 4.3 the dynamic response, as a result of a load change, is given. In this case the machine models where fed with a sinusoidal voltage. The responses of both estimated

single-cage models almost match the response of the double-cage machine. The small difference in the responses of the estimated models is due to the difference in the leakage inductance of both estimated models.

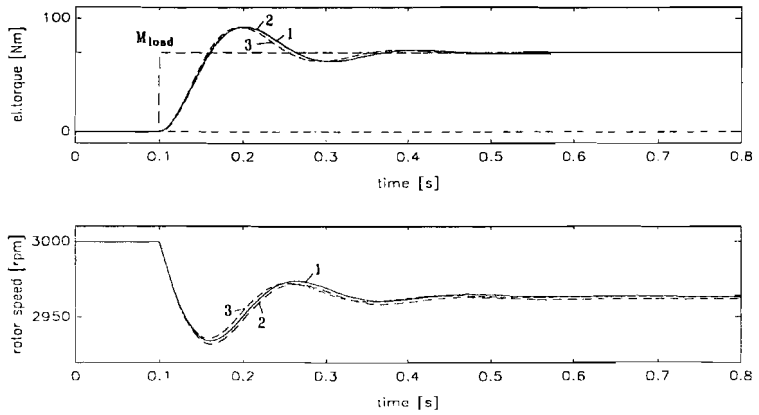


Figure 4.3: dynamic response at sinusoidal supply of the double-cage reference machine (1); estimated single-cage machine fed with sinusoidal voltage (2) and estimated single-cage machine fed with PWM-voltage (3).

A more significant difference between estimated models and the double-cage machine model will be found in the machine losses and torque ripples, in case the machine is fed by an voltage or current source inverter, see Sec. 2.6.
To study the influence of an inverter supply, the dynamic responses of the DCaSM and the estimated single-cage models are compared. Therefore, the machine models are fed by an PWM-inverter (pulse-number *pn*=3). In Fig. 4.4. the responses at a load change of 40% of the rated torque are given.

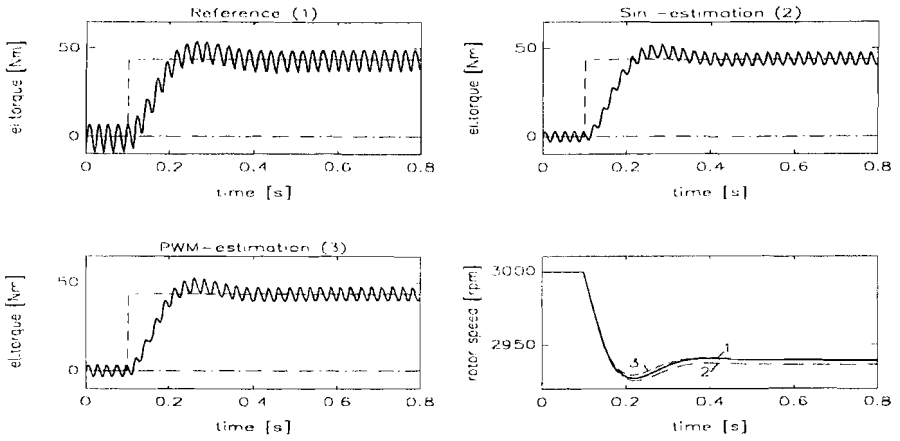


Figure 4.4: dynamic response at inverter supply (pn=3), of the double-cage reference machine (1); estimated single-cage machine fed with sinusoidal voltage (2) and estimated single-cage machine fed with PWM-voltage (3).

Figure 4.4 shows that the responses, except for the torque and current ripple, are almost equal. If we compare Fig. 4.3 and 4.4 it shows that the sinusoidal estimated single-cage

model has a slightly better response when fed with sinusoidal-voltage, the PWM estimated single-cage model has a better fit when using the PWM-inverter voltage.
Another way to validate the estimated parameters is to compare the Bode-plot of the simulated-rotor transfer function and estimated-rotor transfer function. In Fig. 4.5 the Bode-plot of the double-cage rotor transfer function is compared with those of the two estimated single-cage transfer functions.

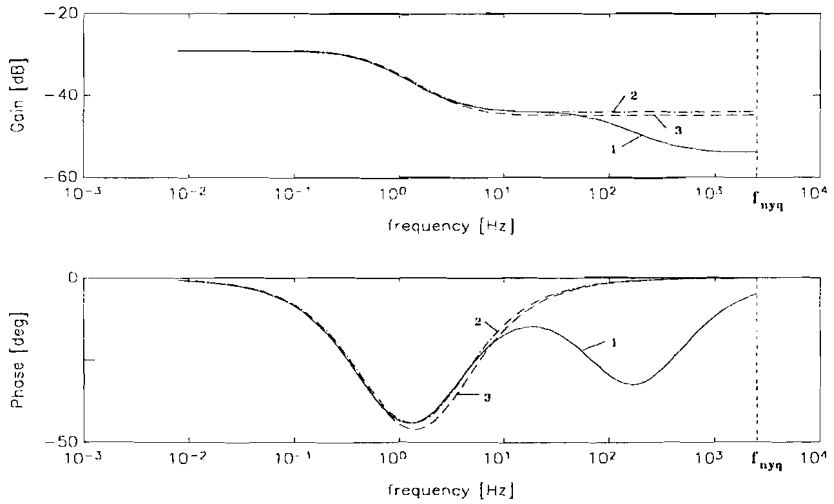


Figure 4.5: Bode-diagram of the double-cage machine (1); the estimated single-cage machine using sinusoidal voltages (2) and estimated single- cage using PWM-supply (3).

The Bode-plots shows a good low-frequency fit, but as expected the single-cage model is not able to describe the frequency behaviour above 10 Hz in case of the sinusoidal supply estimation. The estimation performed with a PWM-inverter supply shows to be less accurate.
However, it shows that using the two-flux-model method a machine with significant skin-effect can be approximated by a single-cage model. This estimated model can be used for describing the dominant time constant of the machine.

4.4. OUTPUT ERROR METHOD

To test the OE-method on the single- and double-cage machine models, the inverter model as described in Sec. 2.5 is applied to create the stator voltage supply for the machine models. The used inverter properties are given in Table 4.7.

Table 4.7: inverter properties.

fundamental frequency f_0	40 Hz
DC-link voltage U_{dc}	538 V
Pulse number pn	3

Sample frequency f_s	5 kHz
total measurement time	2.5 s
number of stator periods	99

For identification purposes the data-sets containing stator voltage, stator current and mechanical rotor angle are derived using a machine model at a simulation time step of $T = 100\mu s$. For simulations a Runga-Kutta 4 integration method is applied. The data is stored with a sample frequency of $f_s = 5\,kHz$. If however no precautions are taken, *aliasing* is likely to occur. Therefore the stator voltage and currents are filtered with a 4th-order Butterworth filter with a cut-off frequency set to $f_c = 1.5\,kHz$ and the data-set is stored after down sampling from 10 *kHz* to 5 *kHz*.

From the simulated stator voltage U_s^S, stator current I_s^S and mechanical rotor angle ρ^s, the stator flux Ψ_s^R and stator current I_s^R in rotor coordinates are reconstructed. The mechanical load M_{load} is hold constant at about 50% of the rated torque.

In the simulation models the rotor circuits in the rotor-axis R1 and R2 are equal. Therefore, estimation results will be independent of the choice of data, related to the R1- or R2-axis. The off-line OE-method is applied to data related to the R1-rotor axis, which results in an estimated discrete rotor transfer function $\hat{G}(q,\theta)$.

4.4.1. SINGLE-CAGE ESTIMATION

The main interest is to estimate the machine parameters which describe the machine at the fundamental frequency. Therefore the low frequency fit should be emphasised by means of the pre-filter $L(q)$. To estimate all machine parameters the input signal (current I_s^R) must be sufficiently exciting. If the input signal is a single sinusoidal signal only two parameters can be estimated from amplitude and phase information. A signal is called persistently exciting (PE) of order n if the spectrum is different from zero in at least n points in the interval $-f_{nyq} < f < f_{nyq}$. Thus, besides the fundamental component at least one higher harmonic should be present, for estimation of the single-cage parameters (PE = 2).

For identification the single-cage machine model is fed with a PWM-voltage supply. In Fig. 4.6. the harmonic content of the applied stator voltage and resulting stator current is given. The used data-set with a total measurement time of 2.5 seconds contains 12500 samples (N=12500).

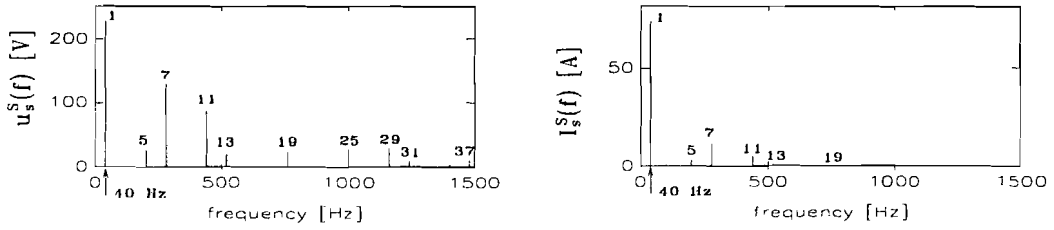


Figure 4.6: harmonic content of the applied stator voltage U_s^S (PWM with n_p=3) and resulting stator current I_s^S, using the SCaSM.

The stator flux Ψ_s^R and stator current I_s^R in rotor coordinates are reconstructed using the off-line method discussed in Sec. 3.2.5. using the Matlab m-file "PRE_PROC.m", Appendix G1. In Fig. 4.7 the flux and current in rotor coordinate R1, necessary for identification, are depicted.

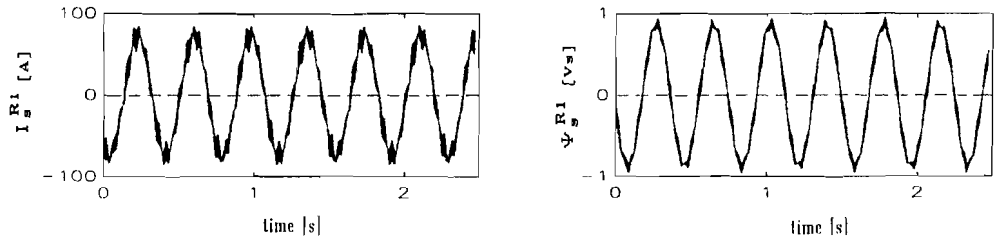


Figure 4.7: stator current I_s^{R1} and flux Ψ_s^{R1} in rotor coordinate r1.

For the single-cage identification three parameters have to be identified: L_σ, L_m and R_k', the stator resistance is assumed to be known. Using only one single sinusoidal voltage component, enables the estimation of two parameters from amplitude and phase information. Therefore, two-signal frequencies are needed to estimate the desired single-cage parameters. The filter $L(q)$ is used to emphasise the fundamental component plus one harmonic. Looking at the harmonic content of the current and flux in rotor coordinates gives an indication for the choice of the filter $L(q)$. The spectrum of reconstructed stator current and flux in rotor coordinates is given in Fig. 4.8. The spectrum contains peaks at the slip frequency f_{slip} and at the switching harmonics in rotor coordinates at $240 \pm f_{slip}$ and $480 \pm f_{slip}$ (40 Hz fundamental frequency).

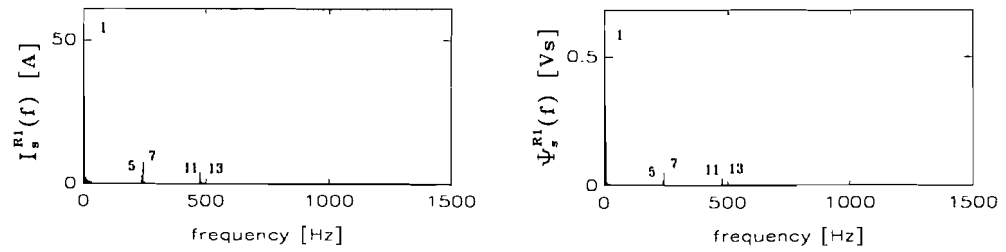


Figure 4.8: harmonic content of stator current I_s^{R1} and flux Ψ_s^{R1} in rotor coordinate r1.

The used pre-filter is a fourth-order Butterworth filter. The cut-off frequency of the applied low-pass pre-filter is chosen to be $f_c = 300Hz$. Filtering will cause phase shift. Therefore, to avoid errors due to this phase shift both input (current I_s^{R1}) and output (flux Ψ_s^{R1}) are filtered with the same pre-filter. The filtered data set is used as input for the OE-identification routine "PAR_EST.m" in the program "MATLAB" (Appendix G2). With the off-line method an OE-model is used to estimate the discrete transfer function between the reconstructed flux and current. From the resulting discrete parameters the physical parameters are derived, using the inverse Z-transformation according to Eq. 3-64. The resulting continuous time single-cage parameters are given in Table 4.8.

Table 4.8: OE-method, single-cage estimation of the SCaSM. pre-filter at 300 Hz ($R_s = 0.212\Omega$)

parameter	simulation		estimation		error
	[si]	[pu]	[si]	[pu]	[%]
L_m	33.75 mH	3	33.53 mH	2.981	0.65
L_σ	6.52 mH	0.58	6.527 mH	0.580	0.03
R_k'	0.141 Ω	0.04	0.1414 Ω	0.040	0.02

In Fig. 4.9 the output-error between the reconstructed stator flux

$$\Psi_s^{R1} = \int \left(U_s^{R1} - I_s^{R1} R_s \right) dt \tag{4-1}$$

and the estimated flux

$$\hat{\Psi}_s^{R1}(k) = \hat{G}(q) I_s^{R1}(k) \tag{4-2}$$

is shown. The reconstructed flux is derived by integration in the frequency domain using the discussed off-line method. The estimated stator flux is calculated using the estimated discrete transfer function $\hat{G}(q,\theta)$ with the unfiltered measured stator current as input signal.

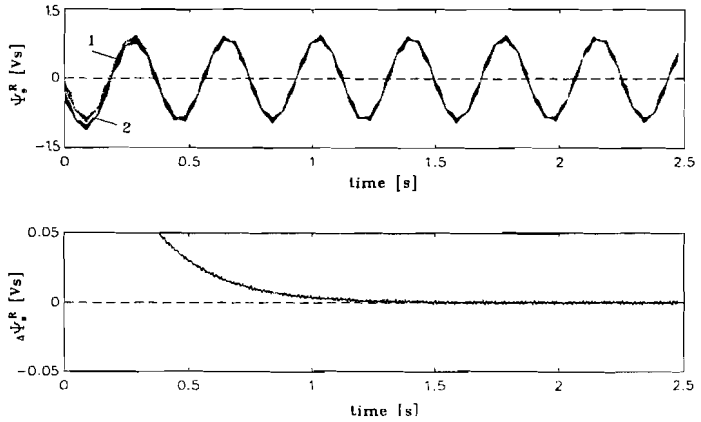


Figure 4.9: resulting output-error between 1) reconstructed and 2) estimated stator flux. low pass pre-filter at 300 Hz.

Because, the initial value of the estimated stator flux is not correct , the output of the transfer function needs some time to converge. Used on the SCaSM the OE-estimation method results in a good estimation of the simulated machine parameters (error < 1%). As long as the cut-off frequency of the pre-filter is chosen < 500Hz, the filter has little influence on the estimation results. If the cut-off frequency is chosen higher, the high frequency components are emphasised too much, which results in a less accurate estimation at the fundamental frequency component.

4.4.2. DOUBLE-CAGE ESTIMATION

To be able to compare the identification methods, the OE-method is first applied to fit a single-cage estimation on the double-cage simulation model. In this case the low-frequency fit should be emphasised with the use of the pre-filter $L(q)$.
The double-cage machine model is fed with a PWM-voltage, the properties of the inverter are defined in Table 4.7. In Fig. 4.10 the harmonic content of the stator current of the double-cage machine is given. Comparing the harmonic content of the stator current resulting from the DCaSM with the stator current I_s^S of the SCaSM (Fig. 4.5) shows significant higher amplitudes of the switching components in case of the double-cage model.

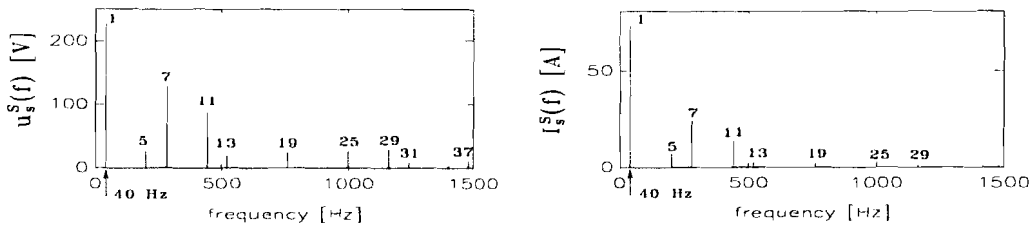


Figure 4.10: harmonic content of PWM-supply voltage U_s^S and resulting stator current I_s^S, using the DCaSM.

After pre-processing the current and voltage the resulting current I_s^R and flux Ψ_s^R (Fig. 4.11) in rotor coordinates are applied in the OE-identification method.

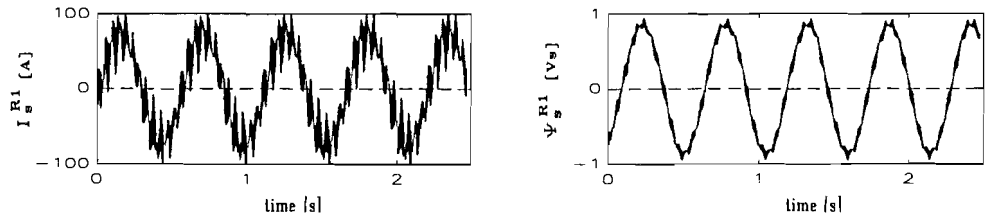


Figure 4.11: stator current I_s^{R1} and flux Ψ_s^{R1} in rotor coordinate r1.

In Fig. 4.12 the harmonic content of both current and flux in rotor coordinates is given. An optimal estimation can be found by looking at the resulting output-error between the reconstructed and estimated flux Ψ_s^{R1}.

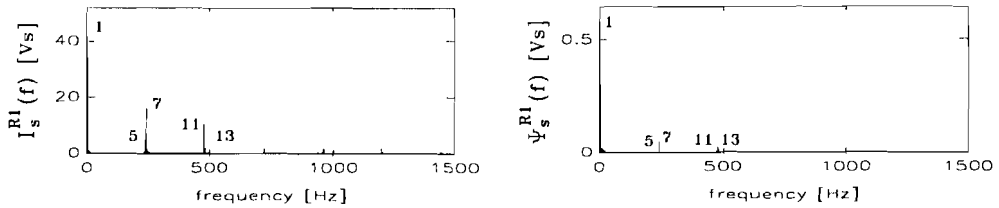


Figure 4.12: harmonic content of reconstructed stator current I_s^{R1} and flux Ψ_s^{R1} in rotor coordinate r1.

In practice the continuous-time system to be identified is never completely described by the chosen model. Usually the estimated model is of lower order than the real system. Trying to describe a system by means of a lower-order model is called under modelling. In that case it is not possible to identify the true parameters of the process. A clear effect of under modelling is that the estimated parameters depend on the weighted input spectrum. If the input signal is sufficiently exciting, the pre-filter can be used to shape the model-fit in a relevant frequency region.

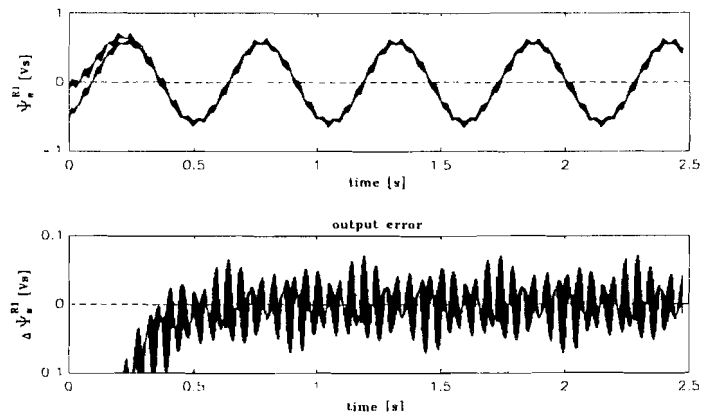


Figure 4.13: resulting prediction error, when applying the unfiltered current to the estimated single-cage rotor transfer function.

Filtering the data-set with the cut-off frequency of the pre-filter set equal to $f_c = 100Hz$, in order to emphasis the low-frequency fit, results in an optimal output-error. In Fig. 4.13 the resulting output-error is depicted.

It is clear that the estimated transfer function is not able to predict the high frequency behaviour correctly, which is probably a result of under modelling. The estimated discrete rotor transfer function is now transformed to a continuous time approximation. The physical parameters are derived using Eq. 3-64, the result is given in Table 4.9.

Table 4.9: OE-method, estimated single-cage parameters of the DCaSM (low-pas filter at 100 Hz)

parameter	value	
	[si]	[pu]
$\hat{L}_m$	17.2 mH	1.53
$\hat{L}_\sigma$	3.4 mH	0.30
$\hat{R}'_k$	0.131 Ω	0.0371

It shows that the estimated main inductance is very low compared to the main inductance of the DCaSM (Table 4.2). To validate these estimated single-cage machine parameters, the Bode-plots of the simulated and estimated continuous rotor transfer function are compared, see Fig. 4.14.

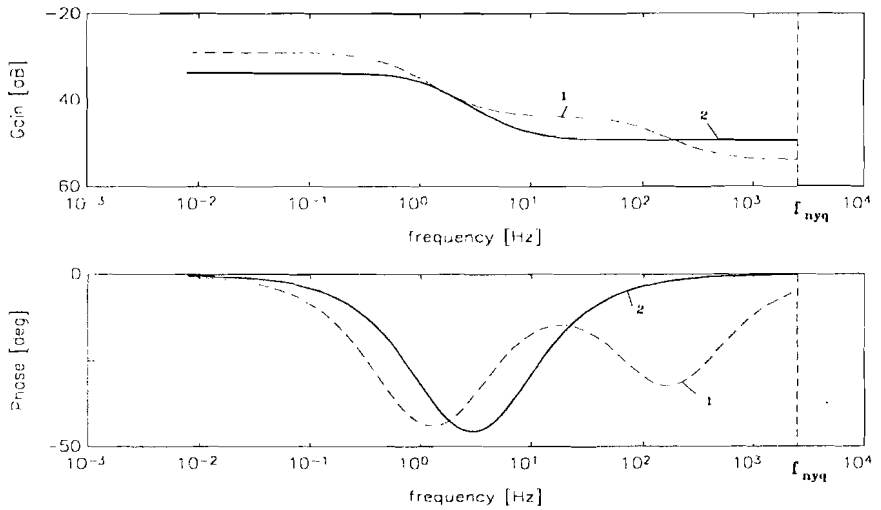


Figure 4.14: Bode-diagram of the double-cage simulation model rotor transfer function (1) and the estimated single-cage rotor transfer function (2).

Due to a combination of under modelling and an extreme double-cage effect, the OE-method is not able to emphasise the low-frequency fit, like in case of the TFM-method. However, in contrast to the TFM-method, the OE-method should be able to estimate higher order machine models. The desired model order can be found by increasing the order of the estimated model, thereby looking at the resulting prediction-error. If, as a

result of a higher order estimation the prediction-error is not decreased anymore, an optimal model order is selected. Following this procedure one must keep in mind that it takes one harmonic signal to identify two parameters.

The ability to estimate higher order systems is tested on the double-cage reference machine (Table 4.2). A double-cage model has five rotor parameters which must be identified. Therefore, at least three different exciting signals are needed. The stator current I_s^R and flux Ψ_s^R, in rotor coordinates were already given in Fig. 4.12. The cut-off frequency of the pre-filter $L(\omega)$ is set to $f_c = 600Hz$. Applying the filtered data-set to a second-order model estimation with the OE-identification method, results in the output error given in Fig. 4.15.

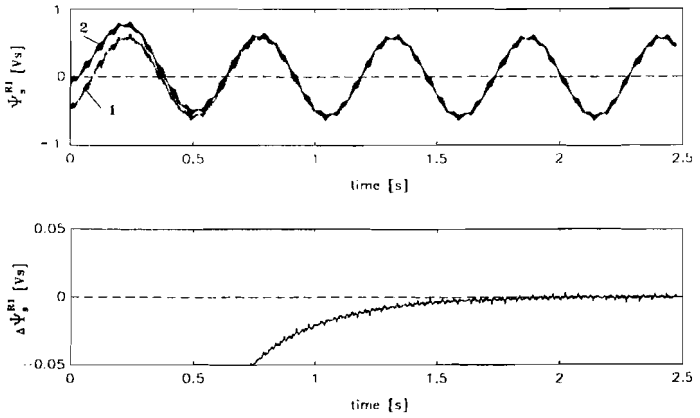


Figure 4.15: resulting output-error between reconstructed (1) and estimated flux (2), when applying the unfiltered current as input to the estimated double-cage rotor transfer function.

The estimated discrete transfer function resulting from the OE-routine, is then transformed to continuous time transfer function, using Tustin's-transformation Eq. 3-70. Tustin's-transformation is an approximation of the estimated discrete transfer function, which in many cases leads to satisfying results. Using Eq. 3-70 the continuous time machine parameters are derived. The estimated continuous-time parameters are given in Table 4.10.

Table 4.10: OE-method, double-cage estimation of the DCaSM (R_s = 0.212Ω)

parameter	DCaSM		estimation		error [%]
	[si]	[pu]	[si]	[pu]	
L_m	33.09 mH	2.941	33.33 mH	2.962	0.73
L_σ	2.012 mH	0.179	2.021 mH	0.180	0.45
R'_{k1}	2.718 Ω	0.769	2.7189 Ω	0.770	0.515
R'_{k2}	0.136 Ω	0.039	0.1355 Ω	0.0384	0.34
$L'_{\sigma 2}$	5.406 mH	0.481	5.427 mH	0.482	0.39

The error introduced using Tustin's-transformation is studied by comparing the Bode-plots of (1): the simulated continuous-time rotor transfer function, (2) the estimated

discrete transfer function and (3) the transformed estimated continuous-time transfer function, as depicted in Fig. 4.16.
Due to the chosen structure, the estimated discrete transfer function will always have zero-phase on the Nyquist frequency. This estimated discrete transfer function is therefore not able to describe the real continuous rotor transfer function accurately for frequencies just below the Nyquist frequency, because the continuous transfer function has no zero-phase at the Nyquist frequency.

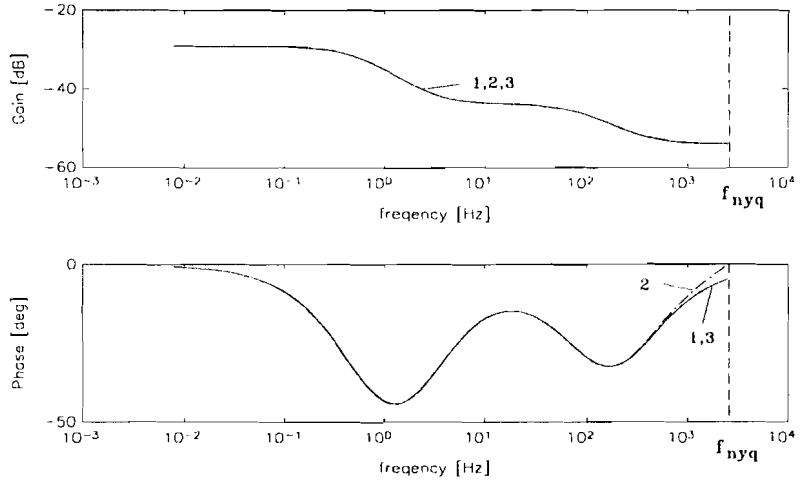


Figure 4.16: Bode-plot of the rotor transfer function: simulation (1), discrete estimation (2) and transformed (Tustin) continuous-time estimation (3).

The leakage inductance L_σ dominates the rotor transfer function in the high frequency region K_{high}. The low frequency gain K_{DC} is defined mainly by the main inductance L_m. From Eq. 3-64 it follows that:

$$K_{DC} = \lim_{\omega \to 0} G(\omega) = L_\sigma + L_m \approx L_m \tag{4-3}$$

$$K_{high} = \lim_{\omega \to \infty} G(\omega) = L_\sigma \tag{4-4}$$

Thus, increasing the sample frequency results in a more accurate estimation of the leakage inductance L_σ.
The results show that the OE-method is able to estimate higher order models, like a double-cage, and that using Tustin's-transformation leads to a accurate discrete to continuous transformation of the estimated double-cage model.

5. ESTIMATION OF REAL MACHINE PARAMETERS

In this chapter the parameters of a Heemaf 11kW SKA-machine (Appendix C) are estimated. According to the speed/torque curve this is an asynchronous machine with significant skin-effect. Using the TFM-method the machine is identified as being a single-cage machine. With the OE-method the effect of a second order model identification is studied.

5.1. MEASUREMENTS

For both identification methods, the three-phase voltages and currents have to be measured. The measured three phase voltages and currents are fed into a DSP-system. The DSP-system was already available and is build up around a TM320c30 processor. The system is elaborately discussed in the thesis of [Jans '93].
In this DSP-system the measured values are scaled and transformed to a two-phase representation, according to Appendix B. The two-phase representation of voltage and current serves as input to the TFM-method (Sec. 3.1) or they are stored to create the necessary data-set that an be to applied in the OE-method (Sec. 3.2).

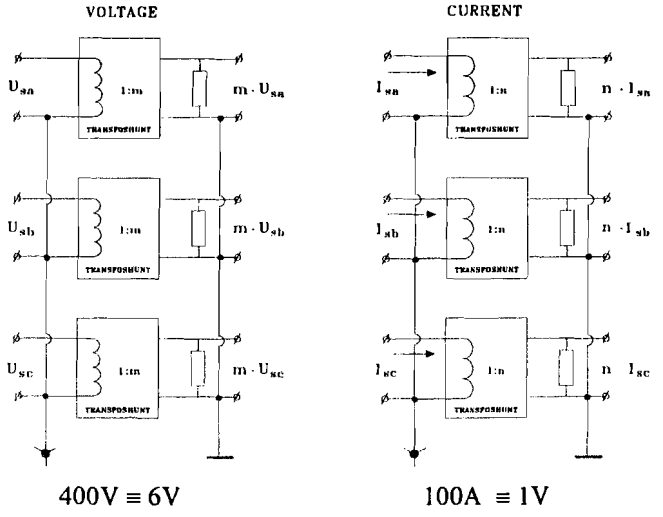


Figure 5.1: measurement of the three-phase voltages and currents

The stator terminal voltages and currents are measured using LEM's (transfoshunts) as depicted in Fig. 5.1. The voltages are observed with respect to an artificial star node. For the realisation of this measurement an existing system is used (BRAMAMATIC, nr. EM 1764). The system has three three-phase voltage and current measurements and was, for this purpose, extended with a direct output of the desired voltage and current measurements.

The measured analogue values are transformed to a digital value, using a 12 bits A/D-converter. The input range of the A/D-converter is -10 to 10V. After a scaling correction this equals an integer value between -2048 and 2048. To be able to use the values in the DSP-system, scaling will be necessary. The input/output routines for the analogue board are defined in the DSP-program "P06.asm". For the digital board the I/O routines are defined in the program "P09.asm".

5.2. TWO-FLUX-MODEL METHOD

To apply the TFM-method on the machine, the two flux-models are implemented in a DSP-system. The measured voltages and currents are transformed to digital values and scaled to be used in the DSP-system. For the TFM-method all measured values and system parameters are translated to a Per Unit system. Using the references given in Appendix C1, this lead to the scaling factors given in Table 5.1.

Table 5.1: import scaling factors used for the TFM-method.

scimp	value
U	$^{2.14}/_{2048}$ pu
I	$^{1.94}/_{2048}$ pu

With the use of four DC-voltage sources the estimated machine parameters R_s, $\hat{R}_k'$, $\hat{L}_\sigma$ and $\hat{L}_m$ are fed into the DSP-system via an A/D-converter. This way the parameters can be tuned on-line. In Fig. 5.2 the realised configuration is depicted. The two flux-models have been implemented in a DSP-program given in Appendix F1. However, for stability reasons the U/I-model is realised in an other way as depicted in Fig. 3.1. Due to measurement noise the output of an integrator without feedback will drift away. To avoid this integration drift an extended U/I-model with feed-back is used, as depicted in Fig. 5.2. A thorough description of this U/I-model is given in [Jans '93]. In the I/φ-model no drift problems will occur.

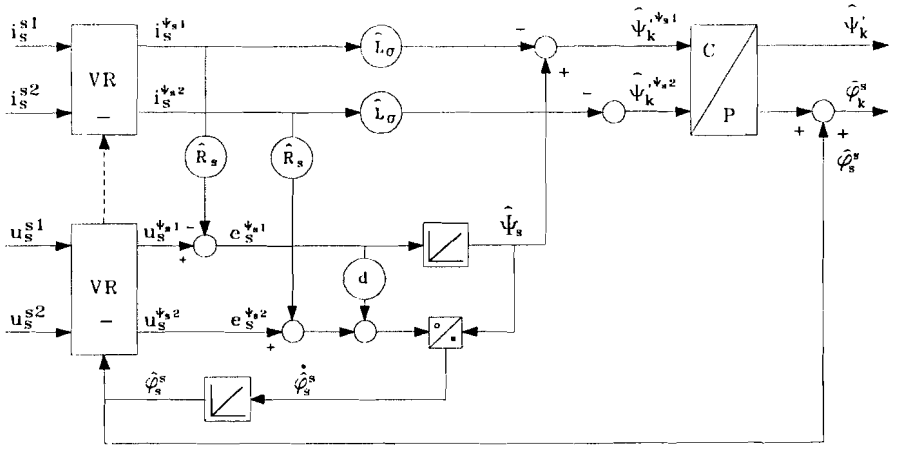


Figure 5.2: applied U/I-model in the TFM-method.

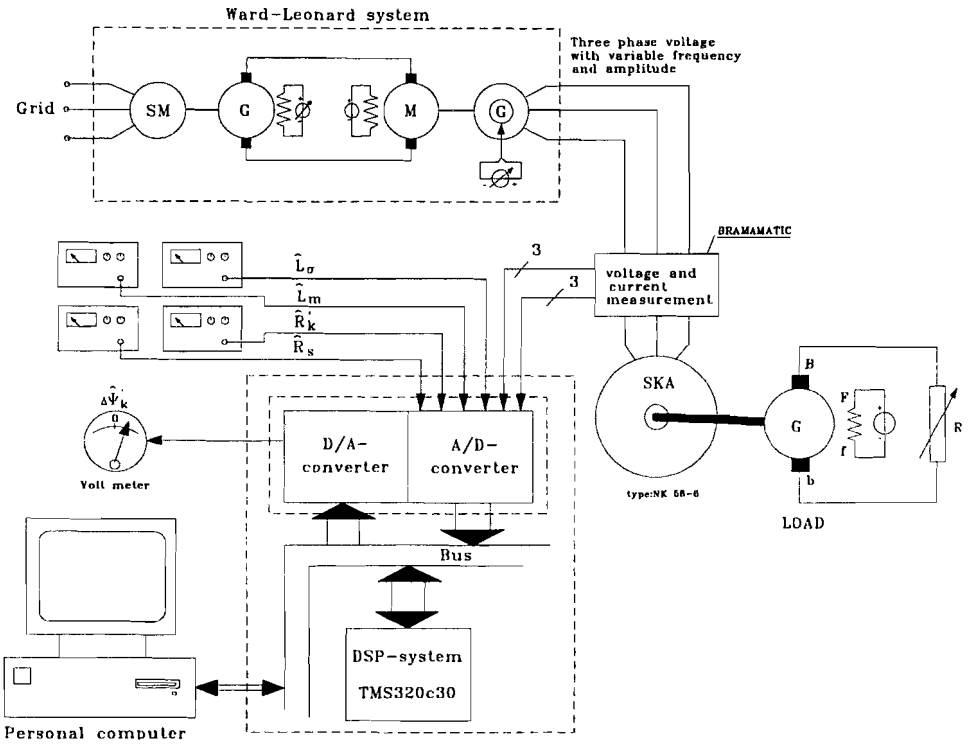


Figure 5.3: configuration used for identification with sinusoidal voltages and currents

The per phase stator voltages and currents are measured, using the BRAMAMATIC. Via an A/D-converter the measured values are fed into the DSP-system. In the DSP-system the 3-phase voltages and currents are transformed to a 2-phase representation which are the inputs of the two flux-models. Output of the TFM-method is the flux difference $\Delta\hat{\Psi}_k$. Via the A/D-board this analogue signal can be measured with a voltmeter. This flux deviation can be made equal to zero by tuning the model parameters under different operation conditions.

The identification method is performed using a sinusoidal voltage supply. Because a voltage with variable frequency and amplitude is needed a Ward-Leonard configuration had to be used. In Fig. 5.3. the applied system configuration is given.
The system is operated in the following way. First a synchronous machine is used to drive a DC-generator. The excitation of the DC-generator is variable, resulting in a variable DC-voltage. This voltage is used to feed a DC-motor, which drives a synchronous generator. Changing the excitation of the DC-generator and synchronous generator, results in a three phase voltage with variable amplitude and frequency. The machine to be identified is then fed with this variable voltage. The load of the machine is created with a 15kW DC-machine, used as generator with constant field excitation.
For correct use of the TFM identification method, using the above configuration, it is important not to drive the machine far into saturation. This is important, because otherwise the parameter estimation will not converge. This can be explained as follows: using this configuration the flux level can not be hold at a constant value when changing form no-load to load condition. A change in flux level results in changing machine parameters as a result of saturation. However, this will not be the case if no saturation

occurs or if the flux can be hold at a constant value (e.g. applying the TFM-method on a v/f-controlled machine).
Thus to be able to use the method correctly one needs to know where saturation starts. A first indication of the saturation region can be found by measuring stator voltage and current at constant frequency. In this case the stator voltage is proportional with the machine flux. At no-load the rotor current will be small and the stator current will be almost equal to the magnetising current. At small flux level (small stator voltage amplitude) no saturation occurs and the relation between voltage and current will be proportional.
Identification of the machine parameters is performed with a stator frequency $f_s = 40Hz$ and a per phase voltage $U_s \cong 100\,V$rms. First the stator resistance is determined, which can easily be measured with an ohm-meter. The estimation procedure, of the TFM-method is described in Sec. 3.1. The measured stator resistance, the estimated main inductance $\hat{L}_m$, the concentrated leakage inductance $\hat{L}_\sigma$ and the estimated rotor resistance $\hat{R}_k'$ in cold conditions are given in Table 5.2.

Table 5.2: TFM-method, estimated single-cage parameters (cold) of the Heemaf SKA-machine at sinusoidal supply. ($U_s \cong 100V$rms., $f_s = 40Hz$)

parameter	Value	
	[pu]	[si]
$\hat{R}_s$	0.039	0.238 Ω
$\hat{L}_\sigma$	0.22	4.4 mH
$\hat{L}_m$	2.14	43 mH
$\hat{R}_k'$	0.026	0.159 Ω

The estimation of the rotor resistance, as described before, is not very easy in the applied configuration because it is not possible to make a good stepwise voltage change by means of changing the rotor excitation of the synchronous generator. Another way to estimate $\hat{R}_k'$ is to use (Eq. 5-7):

$$-R_k' \cdot i_k'^{w} = \Psi_k' \cdot \dot{\varphi}^r \tag{5-1}$$

Where $\dot{\varphi}^r$ equals the slip frequency of the machine. Using the two flux models, both cage flux Ψ_k' and current $i_k'^{w}$ are estimated. If the slip frequency is known the rotor resistance can be derived. The mechanical speed of the machine is measured with a tacho-meter. The estimation of the rotor resistance was performed at no-load (slip 0.12%) and at load (slip 0.48%), the results being given in Table 5.3.

Table 5.3: rotor resistance estimation (cold), of the Heemaf 11kW SKA-machine

f_s [Hz]	Slip [%]	$i_k'^{w}$ [pu]	$\lvert\hat{\Psi}_k'\rvert$ [pu]	$\hat{R}_k'$	
40	0.12	0.032	0.428	0.013 pu	0.08 Ω
40	0.48	0.047	0.352	0.029 pu	0.17 Ω

It's clear that the rotor resistance is frequency dependent, which is typical for skin effect. This effect will be discussed further on in this report.
At a slip of 0.48% the rotor resistance is about equal to the value estimated in Table 5.2. Consequently, if the mechanical speed can be measured the last discussed method is easier to perform. Nevertheless, both methods show however similar values for the rotor resistance estimation.
The single-cage machine parameters can also be calculated using the measured voltages and currents in load and no-load tests, see Appendix C2. The estimated and calculated single-cage parameters are given in Table 5.4.

Table 5.4: estimated and calculated single-cage parameters (R_s = 0.238Ω)

parameter	calculation		TFM-estimation	
	[si]	[pu]	[si]	[pu]
L_m	29.6 mH	1.48	43 mH	2.15
L_σ	5.8 mH	0.29	4.4 mH	0.22
R_k'	0.1637 Ω	0.027	0.17 Ω	0.028

The results show a large deviation in the estimated main inductance, which is caused by the saturation effect in the machine. In this situation whereby the machine flux is not controlled, the TFM-method must be performed on the machine working in the unsaturated region. The calculated machine parameters according to Appendix C2 are derived using rated voltage. Because almost all machines are designed to make optimal use of the iron at rated voltage the machine will be working in saturation. In the following section the saturation effects will be further studied.

SATURATION EFFECT

A physical asynchronous machine is built out of iron, copper, aluminium and insulation. To optimise the machine's power-to-weight ratio, any magnetic saturation at rated flux density is usually the case. The magnetising inductance L_h is therefore in general considerably non-linear around the operating point. The saturation effect will be estimated using the TFM-method.
According to Eq.'s(2-16 to 2-19) the single-cage parameters, except the stator resistance, of Fig. 2.7 are all dependent of the transformation factor k. Which implies that the parameters are dependent of the magnetising inductance and some rotor leakage inductance. However, it is usually assumed that most of the leakage flux travels through air and is therefore not affected by magnetic saturation.
In the model, depicted in Fig. 2.16 leakage flux has influence on the total leakage inductance L_σ and partly in the main inductance L_m. To be able to estimate the saturation effect in the machine, the parameters R_s, R_k' and L_σ are assumed to be constant and the saturation effect of the magnetising inductance L_h is thought to be concentrated in the main inductance L_m. Due to this assumption a small error will be

made because R_k' and L_σ are, to some extend dependent of the changes in the magnetising inductance L_h according to Eq.'s (2-42 to 2-47).
The saturation effect is observed with the use of the two flux models. When the stator voltage is increased at a constant frequency, the flux in the machine is changed proportionally. If saturation is reached, and the main inductance of the machine L_m decreases, the estimated value $\hat{L}_m$ in the I/φ-model will be incorrect. This results in an estimated flux ${}_{(i)}\hat{\Psi}_k' = \hat{L}_m \cdot i_\mu'$ which will be larger than the "real" flux value ${}_{(u)}\hat{\Psi}_k'$ and therefore $\Delta\hat{\Psi}_k < 0$. Because $\hat{R}_k'$ and $\hat{L}_\sigma$ are thought to be independent of saturation, the deviation in $\Delta\hat{\Psi}_k$ must be adjusted by decreasing the value of $\hat{L}_m$. The measured estimated saturation curve is given in Fig. 5.4, this curve being obtained through measurements. In the machine model described in Chap. 2, the relation between the "cage" flux Ψ_k' and the "magnetising" current i_μ' is given by L_m. Thus for modelling saturation effects the constant L_m has to be replaced by the estimated current-flux relation.

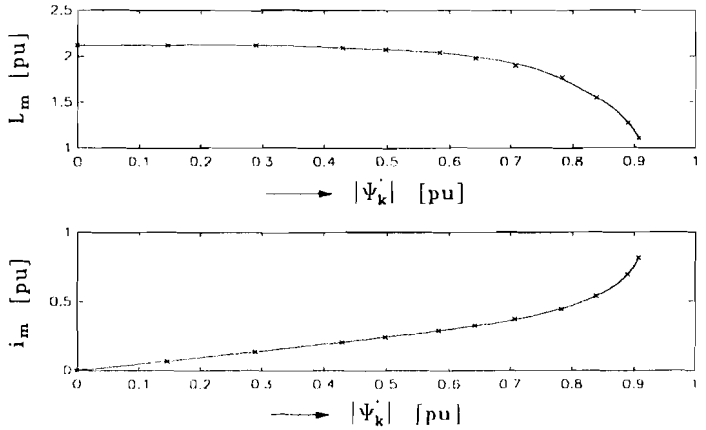


Figure 5.4: estimated saturation effect on the main inductance

The flux-current relation depicted in Fig. 5.4 has to be modelled by a function, which can be used in the machine model already presented in Chap. 2.

REMARKS

Advantages of the TFM-method:
- only terminal voltage and current measurements
- no rotor position sensor needed
- good fit for low frequency components

Disadvantages:
- The two flux-model method can only be used for estimation of single-cage machine parameters.
- The use of sinusoidal voltage sources (although not necessery) brings additional problems for this identification method because variable amplitude and frequency are required. Looking at Fig. 5.1. reveals that four machines are needed to create the desired stator voltage.

5.3. OUTPUT ERROR METHOD

To perform the OE-method the SKA-machine is fed by an inverter-voltage. The used inverter is a standard Holec-inverter (type PMO, nr. EM 1800) with a voltage/frequency control unit. The fundamental output voltage component is controlled with this voltage/frequency control unit. To be able to estimate all machine parameters the applied terminal voltage must have sufficient harmonic content. The inverter is set to a fundamental frequency of 50.62 Hz; the pulse number of the inverter then equals 3. The DC-voltage is about 539V.
To apply the Output-Error method a data-set of measured two-phase voltages and currents and the rotor angular position are needed. Therefore the measured three-phase voltages and currents are transformed into a two-phase representation in the DSP-system. The rotor position is measured using a 1000 pulse encoder and a digital module. This digital module is a simple counter, that counts the flanks of the encoder pulses. Depending on the rotating direction, it counts up or down from 40 to 4040. In the DSP-system this integer value is scaled to an angle between 0 and 2π (Fig. 5.5).

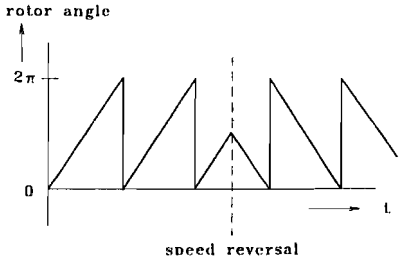


Figure 5.5: rotor angle representation

The measured values of voltage and current must be scaled to the real voltage and current values, before storing in the data-set. In Table 5.5. the used scaling factors are given.

Table 5.5: import scaling used for creation of OE-method data-set

scimp	value
U	$666.67/2048$ V
I	$100/2048$ A

In the DSP-program these scaling factors are placed in the file "P06.ASM". The rotor angle transformation is given in "P09.ASM", see Appendix F2.

5.3.1. DATA ACQUISITION

The terminal voltage U_s^S, current I_s^S and rotor position were measured during steady-state operation at a constat load, using a sample frequency of $f_s = 10\,kHz$. To avoid aliasing the voltages and currents are pre-filtered with an analogue 2nd order low-pass Butterworth filter with cut-off frequency $f_c = 1.5\ kHz$.

To be able to store a data-set with sufficient rotor periods in the DSP-system, the measured data is down sampled to 5 *kHz* and the RAM-memory was changed from 16k to 64k. This enables a total measuring time of 2.4 seconds at a sample frequency of 5 *kHz*. The operation conditions of the SKA-machine are given in Table 5.6.

Table 5.6: operation conditions of the Heemaf SKA-machine and Holec PWM-inverter.

fundamental frequency f_0	50.62 Hz
DC-link voltage U_{dc}	539 V
Pulse number pn	3
Stator current $I_{sa,eff}$	16.8 A
Rotor speed	986.3 rpm.
Slip	2.58 %
Slip frequency	1.31 Hz
Applied load	≈50 Nm

The scheme of the system configuration, used to create the necessary data-set, is depicted in Fig. 5.6.

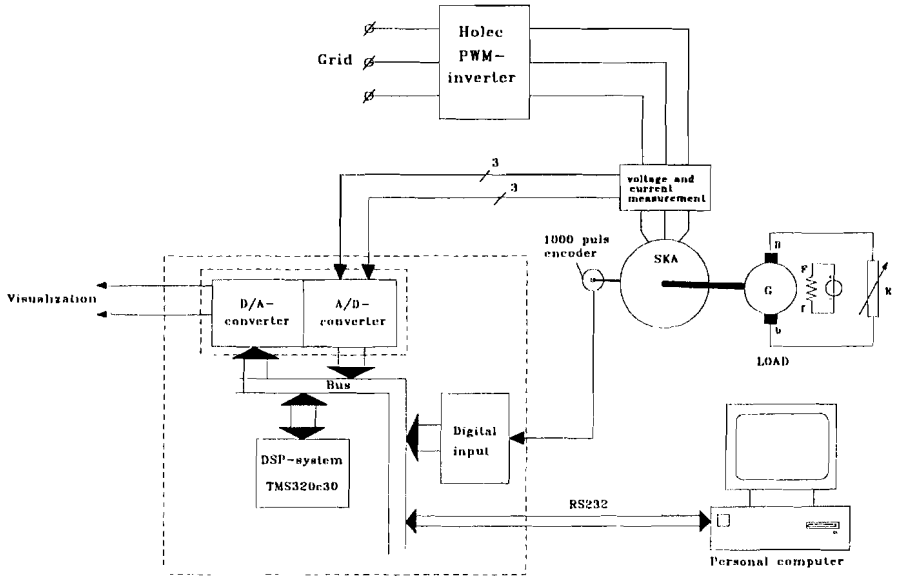


Figure 5.6: configuration used for creation of a data-set.

To store the measured data on in the external memory of the DSP-system, two macro's are used [S. Bosga]: "MATINIT" defines and initialises the necessary memory space; "NAARMAT" stores the data in the defined section and is called each time data needs to be stored. The data stored in the external memory of the DSP-system can then be stored in a data-file on disk using the "C30-DEBUGGER". This data-file is converted to a MATLAB file using the program "STRIPPER.exe". The procedure to store the measured data on disk is more thoroughly discussed in Appendix F3.

The spectrum of the measured stator voltage and current is given in Fig. 5.7. It shows that significant harmonics are present in the signals (5^{th} at 253.1 Hz, 7^{th} at 354.34 Hz and 11^{th} at 556.8 Hz). The stator flux and current in rotor coordinates (Sec. 3.2.5.), given in Fig. 5.8, are reconstructed according to the discussed off-line method, using the Matlab M-file "PRE_PROC.m" (Appendix G1).

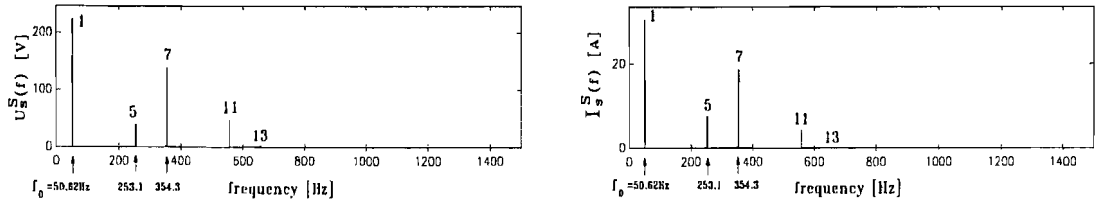


Figure 5.7: harmonic content of terminal voltage U_s^S and current I_s^S.

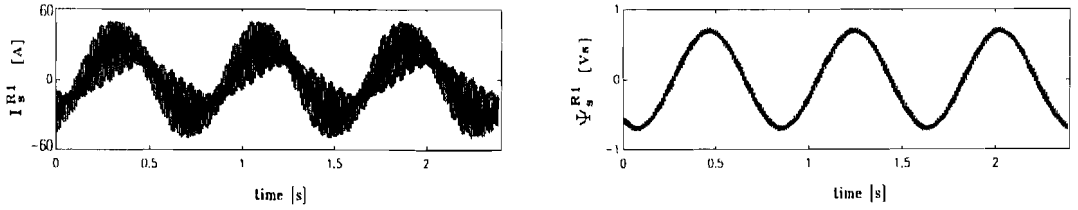


Figure 5.8: reconstructed flux Ψ_s^{R1} and current I_s^{R1} in rotor coordinate r1.

The harmonic content of current and flux is depicted in Fig. 5.9. After transformation to rotor coordinates the spectra contain significant peaks at the slip frequency $f_{slip} = f_0 - f_m = 1.31\,Hz$ and at the switching harmonics on $303.7 \pm f_{slip}$ and $607.4 \pm f_{slip}$.

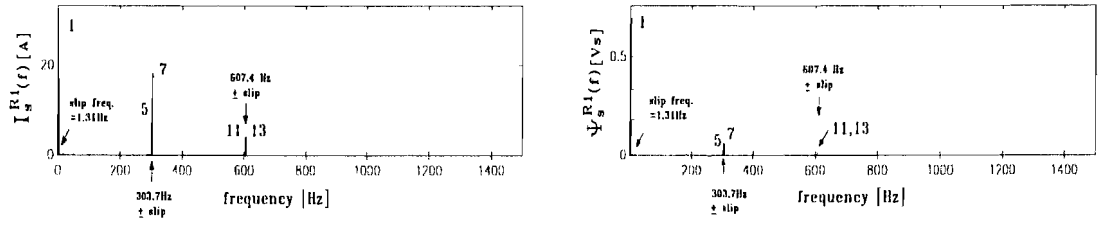


Figure 5.9: Harmonic content of the reconstructed flux Ψ_s^{R1} and current I_s^{R1} in rotor coordinate r1.

The OE-identification method is applied to the reconstructed flux and current signal. The data-set exists of 12000 samples. The amplitude of the fundamental component of the stator flux is found to be $|\Psi_{s,0}| = 0.69\,Vs$.

5.3.2. SINGLE-CAGE ESTIMATION

A first step in using the OE-method is to derive a proper model order. This is, like in the simulations, done by starting with a single-cage parameter estimation. If the estimated result changes significantly when changing the pre-filter $L(q)$, the chosen model-order is not correct. By increasing the model order the resulting prediction error should become smaller until the best approximation of the model order is selected.
To estimate the single-cage parameters the filter is set to $f_c = 300Hz$, thereby including two harmonic signals, according to Fig. 5.9. The resulting output-error is given in Fig. 5.10. It shows that the higher frequency components are weighted to much, resulting in a good prediction of the high frequency component but a large low frequency miss fit. For a better estimation of the low frequency behaviour, the low frequencies in the output error have to be more emphasised by means of the pre-filter.

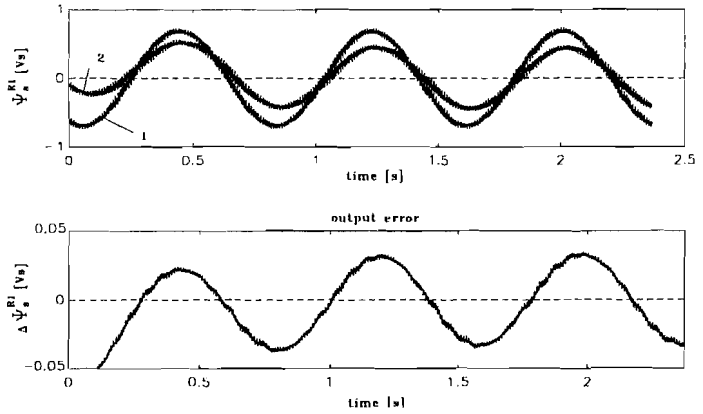


Figure 5.10: resulting output-error between reconstructed (1) and estimated flux (2), using a single-cage rotor model estimation. Pre-filter at 300 Hz.

Using a pre-filter cut-off frequency of 200 Hz results in a better low frequency fit, but still the fundamental component is present in the output-error signal, see Fig. 5.11. Therefore, one can conclude that not all low frequency information is used in the minimisation of the output-error.

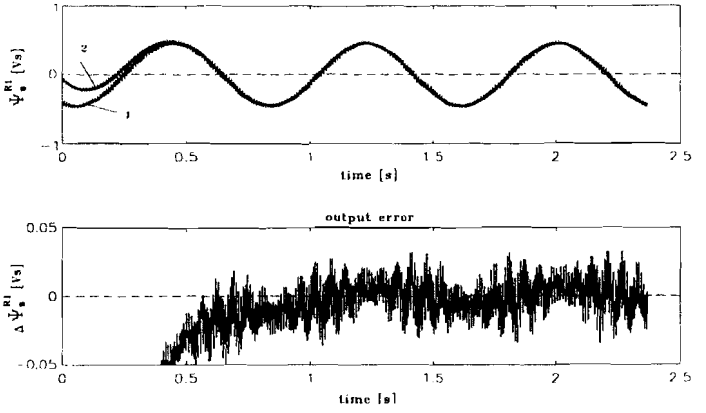


Figure 5.11: resulting output-error between reconstructed (1) and estimated flux (2), using a single-cage rotor model estimation. Pre-filter at 200 Hz.

Lowering the cut-off frequency of the pre-filter to $f_c = 50\,Hz$, results in the output-error given in Fig. 5.12. It shows that the output-error is dominated even more by the high-frequency components and the fundamental frequency component is further minimised. The estimated discrete rotor transfer function is transformed to a continuous time single-cage rotor transfer function, which should give a good description of the rotor circuit for low frequencies. The estimated continuous time single-cage parameters are then derive using Eq. 3-64. In Table 5.7. the resulting estimated single-cage parameters of both TFM-method and OE-method are compared.

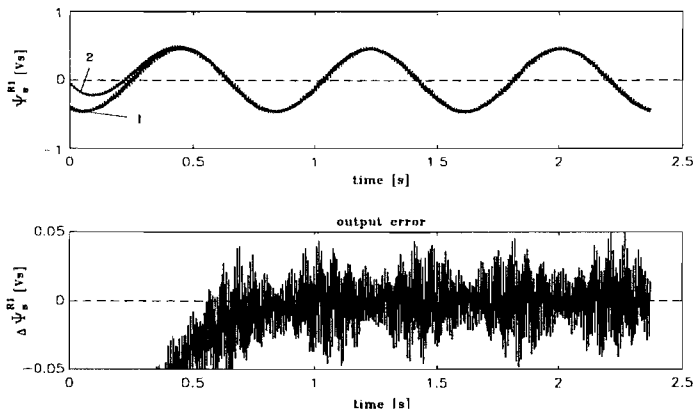


Figure 5.12: resulting output-error between reconstructed (1) and estimated flux (2), using a single-cage rotor model estimation. Pre-filter at 50 Hz.

The OE-method is applied on measured voltages and currents at a fundamental flux level of $|\Psi_{s,0}| = 0.69\,Vs$. The amplitude of the cage-flux is derived using Eq.'s(2-27 to 2-28) and equals $|\Psi'_{k,0}| = 0.65\,Vs$. Therefore, to be able to compare the results of the estimation methods, the saturation effect must be taken into account. This is done by changing the estimated main inductance found with the TFM-method, using the estimated saturation curve, depicted in Fig. 5.4.

Table 5.7: estimated single-cage parameters for the SKA-machine. at a flux level of $|\Psi'_{k,0}| = 0.65\,Vs$ *and measured stator resistance* ($R_s = 0.238\Omega$)

parameter	OE-method		TFM-estimation	
	[si]	[pu]	[si]	[pu]
$\hat{L}_m$	38.6 mH	1.93	39 mH	1.95
$\hat{L}_\sigma$	3.2 mH	0.16	4.4 mH	0.22
$\hat{R}'_k$	0.198 Ω	0.032	0.17 Ω	0.028

Taking the saturation effect into account the estimation of the main inductance $\hat{L}_m$ is almost equal for both estimation-methods. However there is a significant difference in the estimated leakage inductance $\hat{L}_\sigma$. Using the OE-method the leakage inductance is estimated by using an exciting signal at about 300 Hz. This estimated value is not equal

to the value estimated with the TFM-method whereby the machine is only excited at low slip frequencies using different operation conditions. The difference in the estimated value of the leakage inductance due to different exciting procedures indicates under modelling. In the next section the OE-method is applied to derive double-cage parameters using the same data-set.

5.3.3. DOUBLE-CAGE ESTIMATION

In the previous section it is shown that using the OE-method to estimate a single-cage model will not result in a frequency independent model. A better approximation is found using a double-cage estimation model. To be able to estimate all double-cage parameters, the input signal must contain at least three harmonic signals. Looking at the harmonic content of the current I_s^{RI} shows sufficient harmonic components to be able to estimate all parameters. The cut-off frequency of the pre-filter $L(q)$ is chosen at $f_c = 500\ Hz$. In Fig. 5.13 the output-error between the estimated and reconstructed stator flux in rotor coordinates is depicted. It shows that applying a double-cage model the output-error method results in a better high-frequency fit without losing low-frequency information. Changing the cut-off frequency of the pre-filter from 300 to 800 Hz does not result in significant changes in the estimated parameters, which indicates that the chosen model order is a satisfying approximation of the real system.

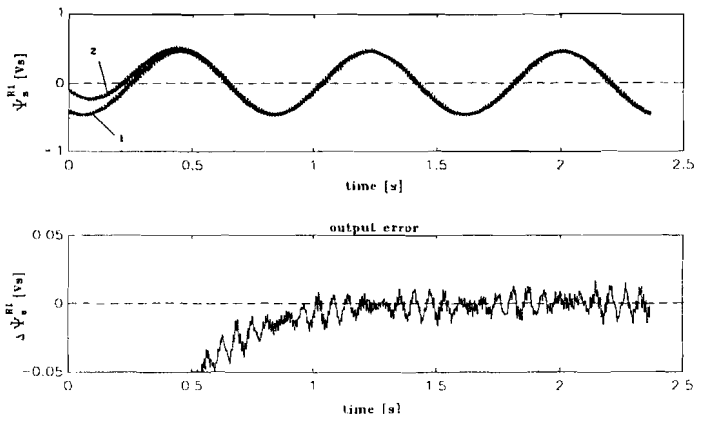


Figure 5.13: resulting output-error between reconstructed (1) and estimated flux (2), using a double-cage rotor model estimation. Pre-filter at 500 Hz.

Using higher order models did not result in a smaller output-error, which is probably due to insufficient higher harmonic frequency components in the used data-set. For a third order estimation the input signal must at least contain four significant harmonic components, to be able to identify 7 model parameters. The estimated discrete second order transfer function is then transformed to a continuous time approximation using Tustin's-transformation. The continuous time rotor transfer function is rewritten according to Eq. 3-70 resulting in the estimated continuous time double-cage parameters, as given in Table 5.8.

Table 5.8: OE-method, estimated double-cage parameters for the SKA-machine. Low-pass pre-filter at 500 Hz. Measured stator resistance $R_s = 0.238\Omega$

parameter	OE-estimation	
	[si]	[pu]
$\hat{L}_m$	41.87 mH	2.094
$\hat{L}_\sigma$	2.46 mH	0.123
$\hat{R}'_{k1}$	3.166 Ω	0.519
$\hat{R}'_{k2}$	0.219 Ω	0.036
$\hat{L}'_{\sigma 2}$	1.997 mH	0.10

In Fig. 5.14 the Bode-plots of the estimated single- and double-cage continuous rotor transfer functions as a result of both tested estimation methods are given.

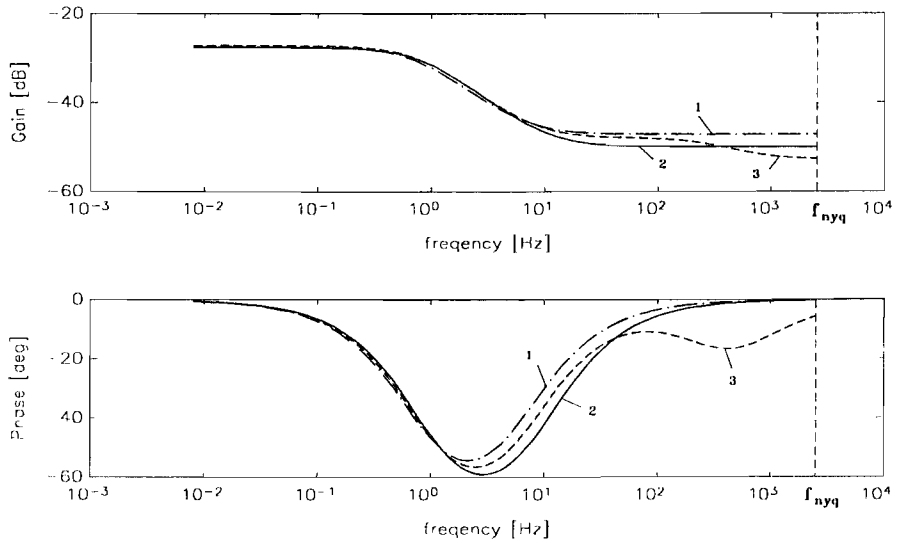


Figure 5.14: Bode-plots of the estimated rotor transfer functions
1) Two-flux-model method single-cage estimation;
2) OE-method single-cage estimation;
3) OE-method double-cage estimation.

The Bode-plots show that the low frequency fit is about the same for all estimated models. The high frequency fit, dominated by the leakage inductance L_σ, is significant different for both single-cage models. Using the OE-method for the estimation of the single-cage model, the leakage inductance L_σ is determined at a harmonic frequency component of about 300 *Hz*. Looking at the Bode-plots it shows that the gains of the OE-method estimated double-cage and single-cage transfer functions are matched at about 300 *Hz*. In the TFM-method all single-cage machine parameters are determined at small slip frequencies. This different approach of identification will cause the difference in estimated leakage inductance value for both methods.

Applying the OE-method, the estimation results are evaluated by comparing the resulting error between reconstructed and estimated stator flux Ψ_s^{RI} and by comparing the Bode-plots. In the following section the estimated machine models are compared with the real machine properties, using the single- and double-cage estimated simulation models.

REMARKS

Advantages of the OE-method:

- identification is performed without additional test signals or special load conditions;
- easy to perform off-line;
- online estimation is posible;
- able to estimate higher order rotor circuits, leading to a better current/flux reconstruction.

Disadvantages:

- accurate measurement of the rotor position or speed is necessery.

5.4. VERIFICATION OF THE ESTIMATION RESULTS

As seen in the previous sections the Heemaf 11kW SKA-machine is identified, as being a single-cage machine when applying the TFM-method. Applying the OE-method the machine is identified as single- or double-cage machine. The goal of the described machine modelling and the studied identification methods is to derive a machine model which fits the real machine properties as good as possible.
In this section the estimated machine parameters derived in Chap. 5 are applied to the simulation models which have been discussed in Chap. 2. In the previous sections the following three machine models have been identified:

- TFM-method, single-cage model;
- OE-method, single-cage model;
- OE-method, double-cage model.

For the validation of these estimated machine models, two tests are performed whereby the properties of the simulation model will be compared with the real SKA-machine properties. For validation the following tests are performed:

1. comparing the measured and simulated steady-state speed/torque curves;
2. using the measured stator voltage and rotor speed as input for the estimated machine models and comparing the resulting stator currents with the measured stator currents.

5.4.1. STEADY-STATE CURVES

A first validation is to compare the steady-state curves of the real machine and the estimated models. Therefore, the steady-state speed/torque curve of the SKA-machine is measured. The curve can not be measured at rated voltage, because of the fact that too large currents will flow through the machines windings at large slip frequencies (about 5-8 times the rated current). To be able to measure the full speed/torque curve form stand still to synchronous speed, the stator current must be limited. Therefore, the applied stator voltage must be kept small. For the measurement of the steady-state speed/torque curve of the SKA-machine a constant stator voltage of $U_{ab,eff} = 100V$ is applied. The simulation models have been tested using the same stator voltage. In Fig. 5.15 the measured and simulated steady-state speed/torque curves are depicted.
The measured curve (1) shows a significant double-cage effect, high pull-out torque. Naturally the both estimated single-cage models (2) and (3) will not match this effect. The estimated double-cage (4) was assumed to be able to match the measured curve better. However, it appears that this estimated double-cage model results in better fit (higher pull-out torque) but is not able to predict the real steady-state torque behaviour.

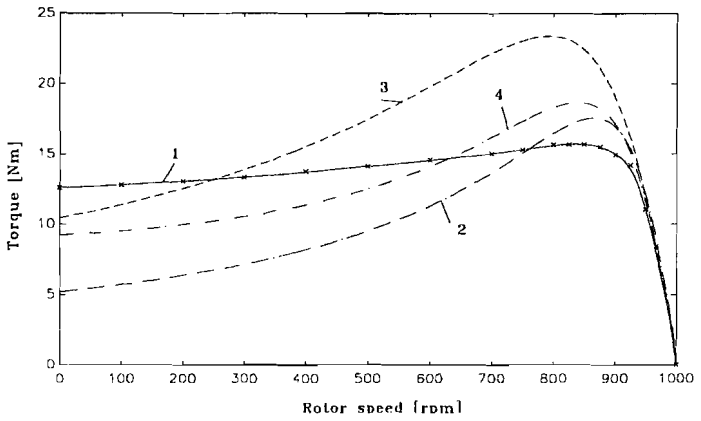


Figure 5.15: the steady-state speed/torque curves at a stator voltage of $U_{ab,eff} = 100V$.
1. Heemaf SKA-machine
2. TFM-method estimated single-cage model,
3. OE-method estimated single-cage model,
4. OE-method estimated double-cage model.

All estimation methods are performed in the normal operation range of the machine. Therefore, it is not fair to compare the hole speed/torque curves. It will be more appropriate to compare the steady-state curves in the normal operation area. In Fig. 5.16 the

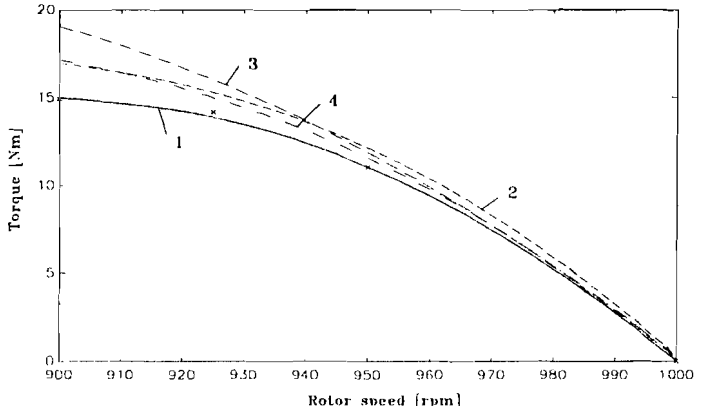


Figure 5.16: the steady-state speed/torque curves at a stator voltage of $U_{ab,eff} = 100V$.
1. Heemaf SKA-machine
2. TFM-method estimated single-cage model,
3. OE-method estimated single-cage model,
4. OE-method estimated double-cage model.

It shows that in the normal operation range (950 to 1000 rpm.) the estimated and measured curves do not differ much. Therefore, one can conclude that in the normal operation range the steady-state behaviour can be described by the TFM-method estimated single-cage model (2) or OE-method estimated double-cage model (4). In this steady-state situation the OE-method estimated single-cage model (3) results in a too high torque compared to the measured torque-curve. While the electric torque of the machine is dependent of the leakage inductance L_σ, the estimated leakage inductance

must be to small. This can also be seen in Fig. 5.14, here the OE-method single-cage is estimated using two frequency components, the slip frequency and the switching harmonic at about 300 Hz.

5.4.2. VALIDATION BASED ON MEASURED DATA

Another, and perhaps better way to validate the estimated machine models is to compare the measured signals with the simulated signals. The measured stator voltage $\vec{u}_s^s$ and mechanical rotor speed serve as input for the estimated machine simulation models. The output of the simulation models will be estimated stator current $\hat{\vec{i}}_s^s$ in stator coordinates. If the estimated machine model is accurate the predicted stator current should correspond with the measured stator current. To be able to compare these currents for validation, the amplitude spectra of the measured and estimated stator current have been compared.

TFM-METHOD SINGLE-CAGE

In Fig. 5.17 the harmonic content of the measured and simulated stator is depicted. The difference between measured and simulated stator current is given in Fig. 5.18.

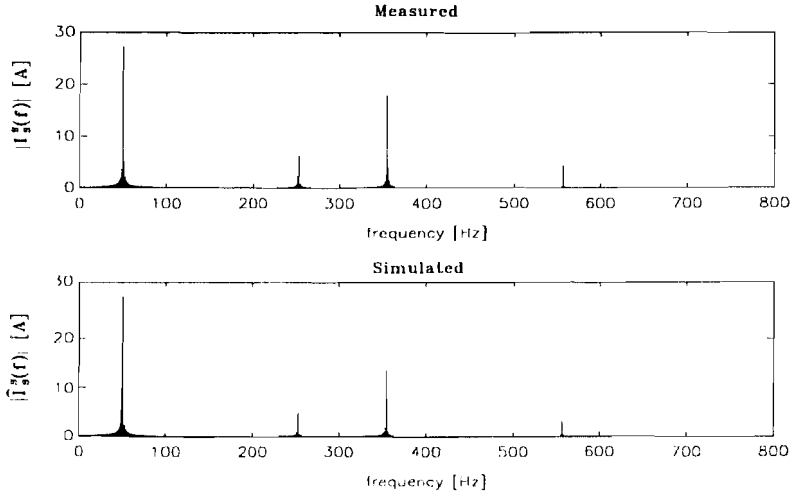


Figure 5.17: amplitude spectrum of measured and simulated stator current. The TFM-method SCaSM is fed with the measured stator voltage.

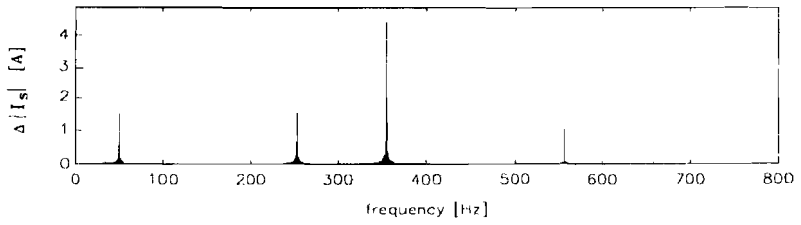


Figure 5.18: deviation between measured and simulated stator current . Using the TFM-method SCaSM fed with the measured stator voltage.

It is clear that the model is not able to predict the amplitude of the higher harmonic components. The estimation error on the fundamental frequency component is about 0.5%. Less accurate is the estimation of the 5th harmonic component with an error of about 25%. The error in the 7th harmonic is also about 25%. Therefore, the TFM-method single-cage estimation is only valid to estimate the fundamental frequency component.

OE-METHOD, SINGLE-CAGE

Applying the measured voltages and rotor speed to the OE-method estimated SCaSM results in the amplitude spectra, depicted in Fig. 5.19. The difference between measured and simulated current is given in Fig. 5.20.

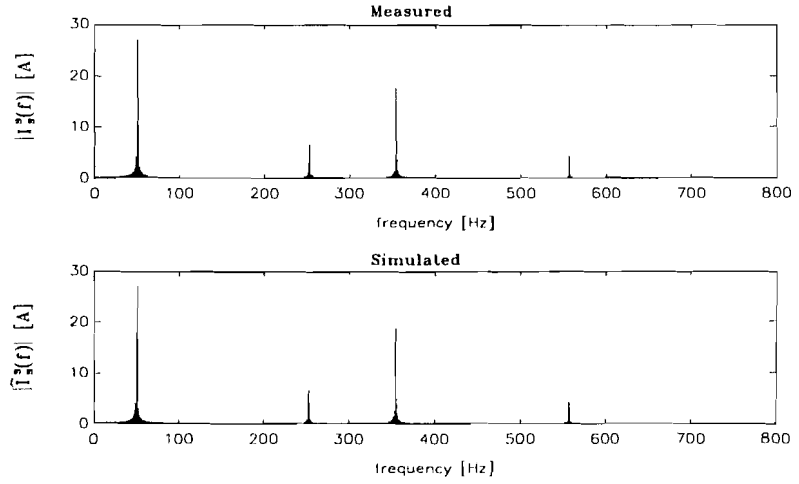


Figure 5.19: amplitude spectrum of measured and simulated stator current. The OE-method SCaSM is fed with the measured stator voltage.

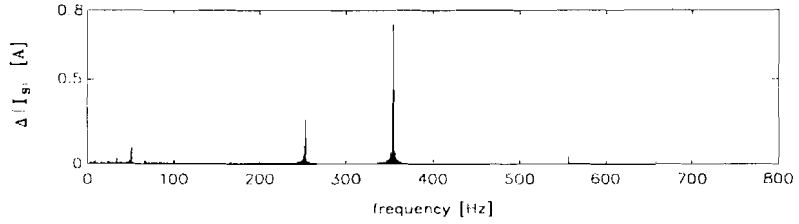


Figure 5.20: deviation between measured and simulated stator current . Using the OE-method SCaSM fed with the measured stator voltage.

Comparing these results with the estimated stator current of the TFM-SCaSM (Fig. 5.19) shows that the OE-method SCaSM gives a better estimation of the real stator current. The fundamental frequency component is estimated with an error of about 0.5%. The 5th harmonic component is estimated less accurate with an error of about 5%. The 7th harmonic has an estimation error of about 5%. The OE-method estimated single-cage model gives a better estimation of machines stator current.

OE-METHOD, DOUBLE-CAGE:

The best estimation of the stator current is found by using the estimated DCaSM. In Fig. 5.22 the difference between the harmonic content of measured and simulated stator current is shown.

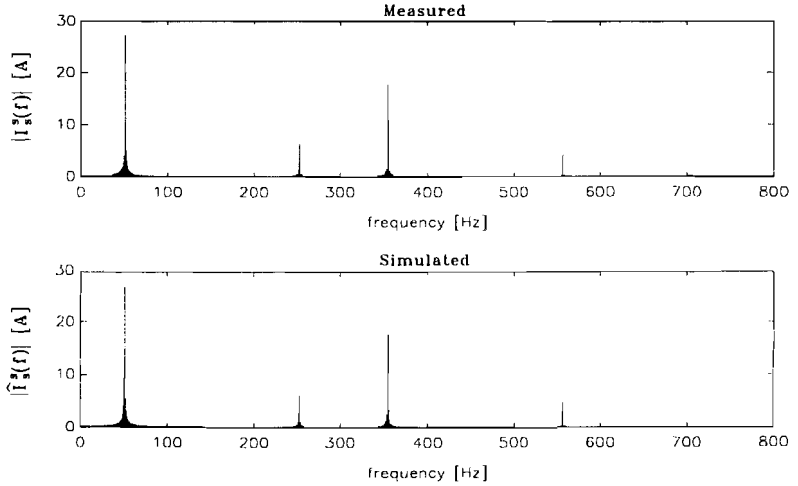


Figure 5.21: amplitude spectrum of measured and simulated stator current. The OE-method DCaSM is fed with the measured stator voltage.

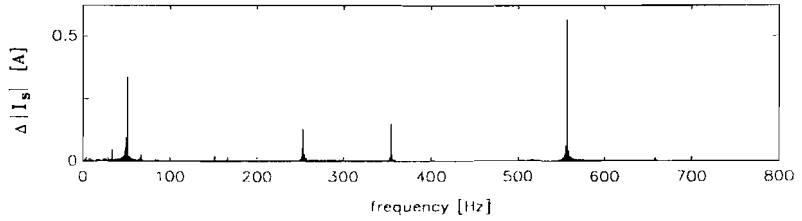


Figure 5.22: deviation between measured and simulated stator current . Using the OE-method DCaSM fed with the measured stator voltage.

Using the DCaSM results in an accurate estimation of the fundamental component (1%), the 5^{th} harmonic (error 2%), the 7^{th} harmonic (error 1%) and the 11^{th} harmonic (error 15%). These results show that the estimated double-cage model leads to a significantly better prediction of the machines stator current, compared to both estimated single-cage models.

6. CONCLUSIONS AND RECOMMENDATIONS

6.1. CONCLUSIONS

In Chap. 2 modelling of single and double-cage induction machines is discussed. Keeping the stator part and rotor part of the machine separated [Velt '94], results in a simple machine model. In this case the rotor circuit is a linear system which can easily be expanded to a higher order rotor circuit.

In Chap. 3 two parameter identification methods are discussed. The methods are tested on simulation models in Chap. 4. The TFM-method is based on the conventional steady state no-load and load tests and is only able to estimate single-cage parameters. The method performs well on single-cage simulation models. Applying the method to a double-cage machine model results in a good estimation of the dominant time constant. This leads to a single-cage machine model which enables the prediction of the dynamic responses at sinusoidal supply. The difference lies in the effects of the higher harmonics on the machines behaviour. Higher harmonics are caused by inverter supply. In modern electric drive systems an inverter supply is always applied. These switching harmonics result in significantly higher current and torque ripples when using a double-cage machine.
Secondly, an output-error method based on linear input-output models is considered. The model parameters are found by minimising the squared difference between the predicted outputs based on the measured inputs and outputs. To be able to estimate all machine parameters, sufficient frequency components must be applied to the machine. Therefore, the switching harmonics due to the inverter supply are used as excitation sources. The OE-method is based on a linear in-output model and is therefore not directly applicable to the measured data-set because there is no linear relation between the stator currents and voltages. By transforming the measured stator signals to rotor coordinates this non-linearity is cancelled out. Transformation to rotor coordinates requires knowledge of the rotor position and stator resistance. The transformed signals correspond with the current and flux which are induced in the rotor. For the transformation off-line routines are used.
With the use of a pre-filter the low-frequency fit can be emphasised. When changing the pre-filter one should keep in mind that the estimated leakage inductances are mainly defined by the switching harmonics and that the estimated main inductance and rotor resistance are mainly defined by the slip frequency. A too low pre-filter cut-off frequency will therefore result in badly estimated leakage inductances.
Applying the OE-method to the single- and double-cage simulation models results in a very accurate estimation of the desired single- or double-cage machine parameters.
However, trying to estimate a single-cage machine model with data of the double-cage simulation model did not result in a good estimation of the dominant rotor time constant. This indicates that the method is sensitive for under modelling.

In chap. 5 both methods are applied to a Heemaf 11kW, SKA-machine. Both methods are first applied to estimate the single-cage machine parameters. As a result of the used configuration for the voltage supply, it's important not to drive the machine far into saturation if the TFM-method is used. Because the flux in the machine can not be hold at a constant value when changing the machine from no-load to load operation, the parameters can not be tuned to a stable value as a result of saturation. Decreasing the stator voltage to about 100 V no saturation occurs and the machine parameters can be tuned correctly. In the tested situation the TFM-results in the unsaturated single-cage parameters. However, usually machine parameters at rated voltage are desired. Therefore, the saturation effect is assumed to be concentrated in the main inductance. At constant values of the rotor resistance and the leakage inductance the stator voltage was increased, resulting in a flux difference between both estimated cage-fluxes. This deviation is made equal to zero by adjusting the main inductance, resulting in a saturation dependent main inductance.

The OE-method is applied to a measured data-set of stator voltage, stator current and rotor position. The OE-method is performed at only one flux level while the used PWM-inverter contained a voltage/frequency-control unit, keeping the flux in the machine at a constant level. Based on this data-set both single- and double-cage machine parameters were estimated. The main difference in the single-cage estimation, using the TFM-method or the OE-method, is the value of the leakage inductance. This is caused by the fact that the two methods are developed using a different approach. The TFM-method is based on the classic machine equations and minimising the error between two estimated fluxes, while the OE-method is based on minimising the error between the real system-output and the output of an estimated machine model. Validation of the estimated single- and double-cage models shows that the OE-method estimated double-cage model results in the best prediction of the measured stator current, hen the SKA-machine is supplied with a PWM-inverter.

The goal of the tested models and identification methods was to achieve an accurate description of the real machine behaviour. It is clear that it is not possible to give one model which describes the real machine completely. In this report no special purpose for the machine model was defined. It should have been better to define such a purpose. In that case the model estimation can be concentrated on a particular operation point.

6.2. RECOMMENDATIONS

- The problem using the TFM-method, in combination with the sinusoidal voltage supply configuration of Fig. 5.3, is that the flux level could not be hold at a constant value if the machine load changes. Therefore, it should be better to apply this method in combination with a control-system (e.g. field-oriented control).

- The output-error identification is based on discrete models. In this report Tustin's-transformation is used to transform the estimated discrete parameters to continuous time parameters. This method led to satisfying results. However, it is possible that other transformations perform even better.

- In [Gort '94] the output-error method is proposed to be used for on-line parameter estimation. Therefore, recursive algorithms of the OE-method are derived. This recursive method can be implemented in a DSP system. In combination with a field oriented controller the machine parameters can be adapted continuously. In that case an on-line estimation of the stator resistance must also be applied.

- The data pre-processing necessary for the OE-method is performed off-line, where integration is done in the frequency domain. The integrated signals will however always be an approximation, An optimal way for data pre-processing is the use of analogue integrators to reconstruct the stator flux. This can be done with the aid of op-amp circuits and special analogue divices for transformation of the three-phase voltages and currents to the desired two-phase representation. This also reduces the necessary calculation time in the DSP-system.

- To validate the estimated machine models it should be better to test these models in combination with a machine control unit.

BIBLIOGRAPHY

[Alg '70] **Alger P.L.**
"Induction machines, their behaviour and uses".
Second edition, Gordon end Breach science publishers, 1970.

[Bla '93] **Blaschke, F.**
"Voorläufiger bericht über verbesserungen bei der feldorientierten regelug von asynchronmaschinen"
Bad vöslau, Österreich, 1994.
Technische Universiteit Eindhoven.

[Bosch '93] **Bosch, P.P.J. van den,**
"Modelling and simulations PSI/c".
Delft University of Technology,
Second edition, September 1993
ISBN 90-71898-17-2

[Bosch '94] **Bosch P.P.J. van den; A.C. van der Klauw.**
Stochastische Systeem theorie
"Modelling, identification and simulation of dynamical systems".
Collegedictaat (5N060)
Faculteit der Elektrotechniek, Technische Universiteit Eindhoven
Measurement and control group, 1994.

[Boze '86] **Boze, B.K.**
"Power electronics and AC-drives".
Printice-Hall, Englewood Cliffs, New Jersey 07632, 1996
ISBN: 0-13-686882-7 025

[Doncl '86] **Doncker, R. de**
"Synthesis and digital implementation of adaptive field-oriented controllers for induction machines with air gap flux control and deepbar compensation".
Katholieke Universiteit Leuven, Departement Elektrotechniek, Vermogen en Industriele Toepassingen, Proefschrift, December 1986
UDC 621.314.5

[Donc '84] **Doncker, R. de; W. Geysen.**
"The double-cage induction motor model for the simulation of inverter fed machines".
IMACS, Electrical Machines and Converters-modelling and simulation, 1984.

[Donc2 '86] **Doncker, R. de; A. Vandenput; W. Geysen.**
"A digital oriented controller using the double-cage induction motor model".
IEEE Trans. on Industry Applications, p.502 -509, 1986.

[Donc '92] **Doncker, R. de,**
"Field-Oriented controller with rotor deep bar compensation circuits".
IEEE Trans. on Industry Applications, Vol. 28, No.5,
September/October 1992

[Gar '80] **Garces, L.J.**
"Parameter adaption for the speed-controlled static AC-drive with squirrel cage induction motor".
IEEE Trans. on Industry Applications, Vol.IA-16, no.2, p.173-178, 1980.

[Gort1 '94] **Gorter, R.J.A.; A. Veltman and P.P.J. van den Bosch.**
"Skin effect impact on induction motor parameter identification using an output-error identification method".
Proceedings PESC '94, Taipei 1994

[Gort2 '94] **Gorter, R.J.A.; A. Veltman and P.P.J. van den Bosch.**
"Parameter estimation for induction machines, using the output-error identification method".
Proc. EPE Symposium on electric drive Design and Applications,
Lausanne, 1994

[Gort '94] **Gorter, R.J.A.**
"A parameter estimator for induction machines".
Eindhoven: Instituut Vervolgopleidingen,
Eindhoven University of Technology, 1994
ISBN 90-5282-357-X geb.

[Jans '93] **Jansen, P.W.G.**
"realisatie van real-time simulaties op een modulair DSP-systeem"
Afstudeerverslag, Technische Universiteit Eindhoven.
Faculteit der Elektrotechniek, Vakgroep Elektromechanica en vermogenselektronica, 1993.

[Lag '92] **Lagonotte, P.; H.Al. Miah and J.C. Trigeassou.**
"Modelling and non-linear identification of static parameters of a double-cage induction machine".
Proc. Int. Conf. on electrical machines 1992, Vol. 2, p.592-96.

[Leo '85] **Leonhard, W.**
"Control of electrical drives".
Springer Verlag; Berlin, Heidelberg, New York, Tokyo, 1985

[Lju '87] **Ljung, L.**
"System identification", theory for the user
Prentice-Hall, 1987
ISBN: 0-13-881640-9 025

[Lju '91] **Ljung, L.**
"System Identification Toolbox- user's guide for use with Matlab"
1991.

[Moo '93] **Moon, S.I. and A. Keyhani**
"Estimation of induction machine parameters from standstill time domain data".
IEEE-IAS Annual meeting, October 3-8, 1993, Toronto, Canada, p.336-342.

[Mach '90] **Schot, J.A.**
"Elektrische machines 1"
College dictaat (5K050)
Technische Universiteit Eindhoven,
Faculteit der Elektrotechniek, Vakgroep Elektromechanica en vermogenselektronica, 1990.

[Slem '] **Slemon G.R.**
"Electric Machines and Drives"
Addison-Wesley publishing company
ISBN: 0-201-57885-9.

[Tex '93] **Texas Instruments,**
"TMS320c30, users guide"
Revision E, June 1991.

[Van '92] **Vandenput, Prof. Dr.ir. A.**
"Elektriche aandrijvingen".
Collegedictaat: Elektrische Aandrijftechniek (5P120))
Technische Universiteit Eindhoven, Fac. E, Vakgroep EMV, Maart 1992.

[Velt '94] **Veltman, A.**
"The fish method: interaction between AC-machines and switching power converters".
Technical University Delft, PhD-thesis 1994.
ISBN: 90-9006763-9

[Werf '94] **Werf, Johan van der**
Het toepassen van een spannings/stroom-model voor directe koppelregeling van een asynchrone machine
Afstudeerverslag EMV 94-13
Technische Universiteit Eindhoven, Faculteit der Elektrotechniek, Vakgroep Elektromechanica en vermogenselektronica, December 1994.

APPENDIX A: THE BOND GRAPH METHOD

A bond graph is composed of components that exchang energy or power through connections. These connections are called (power) bond. In contrast with block diagrams these connections represent an exchange of power instead of a one-way transport of information. Each connection (bond) transports power or energy and is represented by a line. This power is the product of two variables, namely *effort* ***e*** and the *flow* ***f***. The effort and flow are generalisations of similar physical phenomena, see table A.1.

table A.1

	Electrical	Mechanical	Hydraulic
effort ***e***	voltage ***u***	force ***F***	pressure ***p***
flow ***f***	current ***i***	velocity ***v***	flow ϕ

The same description and variables can be used in different physical systems. This is useful if a physical system is for example an electromotor that drives a pump. Components and variables of the different physical systems (electronic, electro-magnetic, mechanical and hydraulic) are used in one system description.
Besides effort and flow other variables can be used, namely the *displacement* ***q*** and the *momentum* ***p***.

$$q = \int f\,dt \qquad\qquad p = \int e\,dt$$

These variables ***p*** and ***q*** preserve their value if a coordinate transformation is performed and are therefore called *preserved variables.*

BONDS

Bonds can be defined more precisely by adding a direction and causality of the variables effort and flow. A bond is represented by a line. The symbol for effort is put above or right of the line. The symbol for the flow is placed below or left of the line as depicted in figure A.1a.

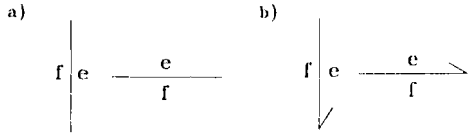


figure A.1: the bond

The direction of the positive power flow is indicated by a half arrow, as shown in figure A.1b.

CAUSALITY

Causality is introduced artificially to calculate the model described by a bond graph. Physical processes however do not have causality. They represent a parallel system with mutually influencing elements. Consequently, causality means assigning cause and effect or input and output to a physical phenomon as expressed by a mathematical expression.
Causality can be expressed in a bond graph by assigning a cross-stroke at one of the two ends of a bond. If the effort e is the cause and the flow f the effect ($f = G \cdot e$), then the cross-stroke is placed at the side of the component. Otherwise, if the flow f is the cause and the effort e is the effect ($e = R \cdot f$), then the cross-stroke is placed opposite to the component side, as illustrated in figure A.2.

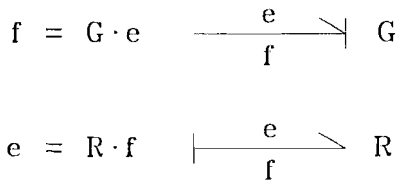


Figure A.2: causality of R and G element

Remark: The causality stroke is introduced purely for determining the calculation sequence. It has no relation to the half arrow of the bond, which indicates the positive power flow.

BOND GRAPH ELEMENTS

A bond is a connection between two elements. The basic elements are given in table A.2

symbol	*Name*	*Function*	*Electric example*
Source:			
$S_e \ \frac{e}{f}$	e-source	e	u-source
$S_f \ \frac{e}{f}$	f-source	f	i-source
One-ports:			
$\frac{e}{f} \ R$	Resistor	$e = R \cdot f$	Resistor
$\frac{e}{f} \ G$	Conductivity	$f = G \cdot e$	Resistor
$\frac{e}{f} \ C$	Capacity	$e = \frac{1}{C}\int f \, dt$	Capacitor
$\frac{e}{f} \ I$	Inertia	$f = \frac{1}{I}\int e \, dt$	Coil
Multi-ports:			
$\frac{e_1}{f_1}\ 0\ \frac{e_2}{f_1}$	0-junction	$e_1 = e_2 = e_3$ $f_1 + f_2 + f_3 = 0$	Parallel
$\frac{e_1}{f_1}\ 1\ \frac{e_1}{f_2}$	1-junction	$f_1 = f_2 = f_3$ $e_1 + e_2 + e_3 = 0$	Serial

SOURCE CAUSALITY

Sources always prescribe causality. An effort source S_e will impose an effort on the system. Consequently the causal stroke has to be drawn at the end of the bond where the effort points to. A flow-source S_f imposes a flow on the system. Hence the cross stroke must be located at the component side of S_f, the flow leaves the source. This is illustrated in figure A.3.

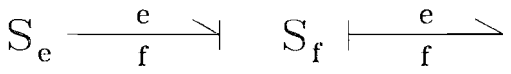


Figure A.3: Source causality

INTEGRAL CAUSALITY

A capacitor or an inertia can be described both by means of an integral equation and a differential equation. However, in simulation software an integrator is supported while a differentiator cannot be realised accurately. This means that the effort is always the input of an inertia and a flow is always the input of a capacity.

$$f = \frac{1}{L}\int e\,dt \qquad e = \frac{1}{C}\int f\,dt$$

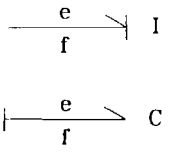


Figure A.4: Integral causality

CAUSALITY WITH JUNCTIONS

A 0-junction represents a parallel circuit with all efforts having the same value. Consequently, only one effort can be an "input". The flows at the other bonds of this 0-junction have to be "inputs". There is always one causality stroke at a 0-junction.
A series circuit, where all flows having the same value, is represented by a 1-junction. This means that only one flow can be an "input". The efforts in the other bonds at that junction will become "inputs". Only one bond, of all bonds will have a cross stroke at a 1-junction. In figure A.5. examples of this junction causality are shown.

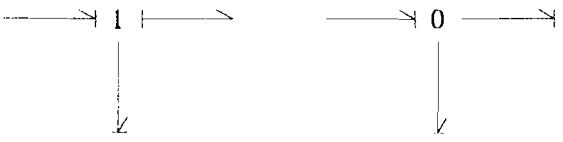


Figure A.5: Causality with junctions.

CAUSALITY OF RESISTORS AND CONDUCTIVITY'S

In practice either a R or a G element can be selected, depending upon the causality requirements. This is illustrated in figure A.6.

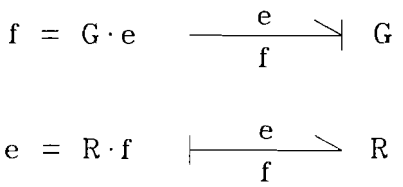


Figure A.6: Causality of R and G elements.

APPENDIX B: COORDINATE TRANSFORMATIONS

In this section the transformation form a symmertrical three-phase system to a two-phase system is described. The transformation is derived under the condition that the power in the three-phase system equals the power in the equivalent two-phase system [Van '92, Chap. 11].

Current from 3→2 is transformed in the following way:

$$\begin{bmatrix} i_s^{s1} \\ i_s^{s2} \end{bmatrix} = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_{sa} \\ i_{sb} \\ i_{sc} \end{bmatrix}$$

The transformation 3→2+1 for flux and voltage is defined by:

$$\begin{bmatrix} u_s^{s1} \\ u_s^{s2} \\ u_{s0} \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} u_{sa} \\ u_{sb} \\ u_{sc} \end{bmatrix}$$

As a result of this transformation with equal power in both three- and two-phase systems the parameters L_m, L_σ, R_k and R_s have to change. The parameters of the three-phase system are denoted with "(3)". In the two-phase equivalent system the parameters are denoted with "(2)". The stator voltage equation for the two-phase can be written as:

$$\begin{bmatrix} u_s^{s1} \\ u_s^{s2} \end{bmatrix} = \frac{2}{3} R_s(3) \begin{bmatrix} i_s^{s1} \\ i_s^{s2} \end{bmatrix} + \frac{2}{3} L_\sigma(3) \begin{bmatrix} \dot{i}_s^{s1} \\ \dot{i}_s^{s2} \end{bmatrix} + \begin{bmatrix} \dot{\Psi}_k^{'s1} \\ \dot{\Psi}_k^{'s2} \end{bmatrix}$$

Thus the three-phase parameter values are transformed to the equivalent two-phase parameters by:

$$R_s(2) = \frac{2}{3} R_s(3)$$

$$L_\sigma(2) = \frac{2}{3} L_\sigma(3)$$

$$L_m(2) = \frac{2}{3} L_m(3)$$

$$R_k'(2) = \frac{2}{3} R_k'(3)$$

Where the inductance $L_m(3)$ can be calculated from measured data, found by performing a no-load and load test according to Appendix C. This method results in the 3 three-phase parameter values.

APPENDIX C: MACHINE PARAMETERS

C1: MACHINE CHARACTERISTICS AND PU REFERENCES

Heemaf EM 3008; SKA-machine nr. 552047; Type: NK 58-6

ΔY 220/380 V	41.5/24 A
11kW / 15pk	cos φ = 0.83
rated speed	950 rpm
rated slip	5 %
rated frequency	50 Hz
number of pole pairs	3

Per Unit references	
$U_{ref} = \sqrt{2} \cdot U_{s,eff}$	311 V
$I_{ref} = \frac{3}{2}\sqrt{2} \cdot i_{s,eff}$	51 A
$\omega_{ref} = 2 \cdot \pi \cdot f_{s,rated}$	100π rad/s
t_{ref}	3.18 s
$R_{ref} = \frac{U_{ref}}{I_{ref}}$	6.1 Ω
$L_{ref} = \frac{U_{ref}}{I_{ref} \cdot \omega_{ref}}$	0.02 H
$M_{ref} = p \cdot \left(\frac{U_{ref} \cdot I_{ref}}{\omega_{ref}} \right)$	151,5 Nm
$\Psi_{ref} = \frac{U_{ref}}{\omega_{ref}}$	0.99 Vs
$J_{ref} = \frac{U_{ref} \cdot I_{ref}}{\omega_{ref}^3}$	$5.12 \cdot 10^{-4}$ kgm^2

C2: CALCULATION OF SINGLE-CAGE MACHINE PARAMETERS

The single-cage machine parameters of the test-machine can be calculated, with data of three measurements:

- a dc-measurement of the stator resistance
- an ac-measurement during no-load operation
- an ac-measurement during load-operation.

In no-load situation one can assume that the machine rotates almost at synchronous speed (slip ≈ 0). The equivalent per-phase circuit for an induction machine is then given in Fig. C1.

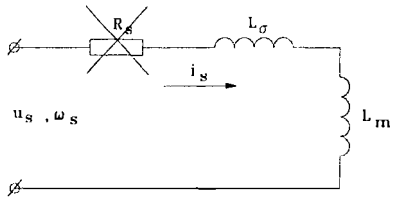


Figure C1: No-load equivalent per-phase circuit

For most machines the stator resistance can be neglected, resulting in the following equation:

$$\frac{\left|u^s_{s,no-load}\right|}{\left|i^s_{s,no-load}\right|} = \omega_s(L_m + L_\sigma) = X_{L_\sigma} + X_{L_m} \qquad \text{(C-1)}$$

where the stator voltage magnitude during no-load, $\left|u_{s,no-load}\right|$, and the stator current magnitude during no-load, $\left|i_{s,no-load}\right|$ are known from a measurement in a no-load situation. Under load conditions the equivalent per-phase circuit is given in Fig. C.2.

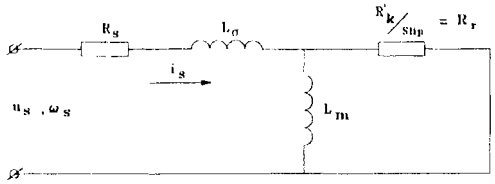


Figure C2: Equivalent per-phase circuit under load condition

The input impedance is defined as:

$$Z_{in,load} = \frac{\left|u^s_{s,load}\right|}{\left|i^s_{s,load}\right|}\left\{cos(\varphi_{s,load}) + j \cdot sin(\varphi_{s,load})\right\} \qquad \text{(C-2)}$$

where the stator voltage magnitude, $\left|u_{s,load}\right|$, the stator current magnitude, $\left|i_{s,load}\right|$ and the load angle φ_s between stator voltage vector and stator current vector are known from

measurements under load condition. The per phase stator input impedance of the equivalent circuit in Fig. C2 can also be written as:

$$Z_{in,load} = \left(R_s + \frac{(\omega_s L_m)^2 R_r}{(\omega_s L_m)^2 + R_r^2} \right) + j \cdot \left(\omega_s L_\sigma + \frac{\omega_s L_m R_r^2}{(\omega_s L_m)^2 + R_r^2} \right) \tag{C-3}$$

The single-cage parameters can then be derived using Eq. (C-1, C-2 and C-3) in the following way:

$$\frac{X_{L_m}^2 R_r}{X_{L_m}^2 + R_r^2} = \frac{\left| u_{s,load}^s \right|}{\left| i_{s,load}^s \right|} cos(\varphi_{s,load}) - Rs = A \tag{C-4}$$

$$X_{L_\sigma} + \frac{X_{L_m}^2 R_r^2}{X_{L_m}^2 + R_r^2} = \frac{\left| u_{s,load}^s \right|}{\left| i_{s,load}^s \right|} sin(\varphi_{s,load}) = B \tag{C-5}$$

$$X_{L_\sigma} + X_{L_m} = \frac{\left| u_{s,no-load}^s \right|}{\left| i_{s,no-load}^s \right|} = C \tag{C-6}$$

Because the constant values, A, B, C, are known from measurements, the single-cage induction machine parameters can be derived, by solving the equation system, (C-4), (C-5) and (C-6):

$$L_m(3) = \frac{X_{L_m}}{\omega_s} = \frac{1}{\omega_s} \left((C - B) + \frac{A^2}{(C - B)} \right) \tag{C-7}$$

$$R_k'(3) = S_{load} \cdot R_r = S_{load} \left(A + \frac{A^3}{(C - B)^2} \right) \tag{C-8}$$

$$L_\sigma(3) = \frac{X_{L_\sigma}}{\omega_s} = \frac{1}{\omega_s} \left(B - \frac{A^2}{(C - B)} \right) \tag{C-9}$$

The calculated parameters are parameters of the equivalent three-phase circuit (this is indicated with (3) in Eq.(C-7, C-8, C-9). To transfer the parameters to a two-phase representation, all parameters must be multiplied with a factor $\frac{2}{3}$, see Appendix B.

$$L_m(2) = \tfrac{2}{3} L_m(3)$$

$$R_k'(2) = \tfrac{2}{3} R_k'(3)$$

$$L_\sigma(2) = \tfrac{2}{3} L_\sigma(3)$$

SINGLE-CAGE PARAMETER CALCULATION OF THE 11 KW SKA-MACHINE

To calculated the single-cage parameters of the Heemaf 11 kW SKA-machine the measured values of the following no-load and load condition are used.

No-load test	
$u_{s,eff}$ (per phase)	220 V
$i_{s,eff}$ (per phase)	13.2 A
load angle φ_s	≈90°
speed ω_{mech}	≈1000 rpm

Load test	
$u_{s,eff}$ (per phase)	220 V
$i_{s,eff}$ (per phase)	16.5 A
load angle φ_s	57.6°
speed ω_{mech}	986 rmp

The measurements result in the following single-cage machine parameters:

parameter	value	
	[si]	[pu]
L_m	29.6 mH	1.48
L_σ	5.8 mH	0.29
R_k'	0.1637 Ω	0.0268
R_s	0.238 Ω	0.0385

APPENDIX D: MOMENTUM OF INERTIA

The total inertia of the 11 kW machine and load machine is determined with the use of an oscillation test. To do this, a mass attached to the axis of machine, see Fig. D.1. Where ℓ is the length from the rotor axis to the mass m. J_R is the total inertia of both AC-machine and DC-machine. The inertia of the system with the pendulum mass is then.

$$J_{tot} = J_R + \ell^2 m \tag{D-1}$$

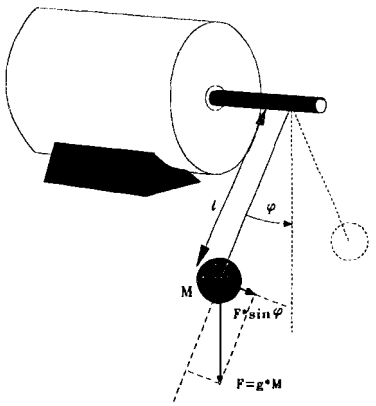


Figure D.1: estimation of inertia

The torque equals the total system inertia J_{tot} times the acceleration of the angle $\ddot{\varphi}$:

$$M = J_{tot}\ddot{\varphi} \tag{D-2}$$

The torque M also equals the force times the length to the machine axis:

$$\ell mg \cdot sin\varphi = \ddot{\varphi}\left(J_R + \ell^2 m\right) \tag{D-3}$$

If the angle φ is small $sin\varphi \cong \varphi$ this yields:

$$\ell mg = s^2\left(J_R + \ell^2 m\right) \tag{D-4}$$

$$s^2 = \frac{\ell mg}{\left(J_R + \ell^2 m\right)} \tag{D-5}$$

This resuilts in the system resonance frequency ω_0 :

$$\omega_0^2 = \left(\frac{2\pi}{T_{pendulate}}\right) = \frac{\ell mg}{\left(J_R + \ell^2 m\right)} \tag{D-6}$$

results in the following expression for the momentum of inertia of both machines

$$J_R = \frac{\ell mg}{\left(\frac{2\pi}{T_{pendulate}}\right)} - \ell^2 m \qquad \text{(D-2)}$$

With the use of an angle sensor the period time of the system is measured. The results are given in table D.1.

Table D.1: Measured inertia of the SKA-machine and DC-load machine .

T_{osc}	m	ℓ	J_R	
[sec]	[kg]	[m]	[kgm^2]	[pu]
1.75	5.125	0.371	0.742	1449

APPENDIX E: PSI/C SIMULATION PROGRAM'S

E1: TFM_ID.PSM:

```
%-------------------------------------------------------------------------
%   TFM_ID.psm
%
%   SINGLE-CAGE MACHINE IDENTIFICATION
%   ----------------------------------
%   Two-Flux-Model (TFM) method [Blaschke]
%   Performed on a double-cage machine model fed by
%   a sinusoidal voltage supply
%-------------------------------------------------------------------------
%  reference values for pu-system  (Chap. 3 Vandenput)

U_ref       = PAR(311);                      %[V] sqrt(2)*[<Usq (eff)>]n
I_ref       = PAR(88);                       %[A] 2/3*sqrt(2)*[<Isq (eff)>]n
omega_ref   = PAR(100*pi);                   %[rad/s]
R_ref       = PAR(U_ref/I_ref);              %[Ω]
L_ref       = PAR(U_ref/(I_ref*omega_ref));  %[H]
flux_ref    = PAR(U_ref/omega_ref);          %[Vs]
m_ref       = PAR(U_ref*I_ref/omega_ref);    %[Nm]
J_ref       = PAR(U_ref*I_ref/(omega_ref)^3);%[kgm^2]

%-------------------------------------------------------------------------
% Sinusoidal voltage supply and load condition
%-------------------------------------------------------------------------

M_load        = PAR(0.3*m_ref);      % Mechanical load

u_s_des       = PAR(0.8*311);        % Voltage amplitude
alfaP_s_s_des= PAR(100*pi);          % Supply voltage frequency

alfa_s_s_des =   INTMOD(alfaP_s_s_des PAR: .83144, pi);
u_s_s1       =   u_s_des * cos(alfa_s_s_des); % Two-phase stator voltage in
u_s_s2       =   u_s_des * sin(alfa_s_s_des); % stator coordinates

%-------------------------------------------------------------------------
%   DUBBEL-CAGE INDUCTION MOTOR MODEL (Chap. 2)
%
%   The used machine parameters are defined in [Donc '86]
%-------------------------------------------------------------------------
% Machine parameters according to the Jordan model
% Given in PU and translated to SI-values
%-------------------------------------------------------------------------

Lh           = PAR(3   * L_ref);
Lsig_s       = PAR(0.12* L_ref);
L_12         = PAR(0.06* L_ref);
Lsig_2       = PAR(0.5 * L_ref);
r_s          = PAR(0.06* R_ref);
R_k1         = PAR(0.8 * R_ref);
R_k2         = PAR(0.04* R_ref);

p            = PAR(1);              % Number of pole pairs.
J_mach       = PAR(1449*J_ref);     % Mechanical inertia.
```

```
%-------------------------------------------------------------------------
%       TRANSFORMATION of the MACHINE parameters to MODEL parameters
%-------------------------------------------------------------------------
%       transformation to equivalent model with 2 inductances

k             = PAR(Lh/(Lh+Lsig_12));
Lm            = PAR(k*k*(Lh+Lsig_12));
R_k1_acc      = PAR(k*k*R_k1);
Lsig          = PAR(Lsig_s + Lh*Lsig_12/(Lh+Lsig_12));
Lsig_2_acc    = PAR(k*k*Lsig_2);
R_k2_acc      = PAR(k*k*R_k2);

%-------------------------------------------------------------------------
%       ROTOR PART in rotor coordinates
%-------------------------------------------------------------------------

i_k_acc_r1    = flux_k_acc_r1/Lm-i_s_r1;     ; total cage current
i_k_acc_r2    = flux_k_acc_r2/Lm-i_s_r2;

i_k1_acc_r1   = i_k_acc_r1 - i_k2_acc_r1;    ; current through outer cage
i_k1_acc_r2   = i_k_acc_r2 - i_k2_acc_r2;

i_k2_acc_r1   = flux_sig2_r1 / Lsig_2_acc;
i_k2_acc_r2   = flux_sig2_r2 / Lsig_2_acc;

E_sig2_r1     = -R_k1_acc*i_k1_acc_r1 + R_k2_acc*i_k2_acc_r1;
E_sig2_r2     = -R_k1_acc*i_k1_acc_r2 + R_k2_acc*i_k2_acc_r2;

flux_sig2_r1 = INT(E_sig2_r1 PAR: .184425);
flux_sig2_r2 = INT(E_sig2_r2 PAR: 7.9778E-2);

flux_k_acc_r1= INT(-i_k1_acc_r1*Rk1_acc PAR: .470608);
flux_K_acc_r2= INT(-i_k1_acc_r2*Rk1_acc PAR: -0.529156);

%-------------------------------------------------------------------------
%       ROTATION OF FLUX AND CURRENT
%-------------------------------------------------------------------------

flux_k_acc_s1= flux_k_acc_r1*cos(rho_s) - flux_k_acc_r2*sin(rho_s);
flux_k_acc_s2= flux_k_acc_r1*sin(rho_s) + flux_k_acc_r2*cos(rho_s);

i_s_r1        =  i_s_s1*cos(rho_s) + i_s_s2*sin(rho_s);
i_s_r2        = -i_s_s1*sin(rho_s) + i_s_s2*cos(rho_s);

%-------------------------------------------------------------------------
%       STATOR PART in stator coordinates
%-------------------------------------------------------------------------

flux_s_s1     = INT(u_s_s1-i_s_s1*r_s PAR: .582941);
flux_s_s2     = INT(u_s_s2-i_s_s2*r_s PAR: -0.500511);

i_s_s1        = (flux_s_s1-flux_k_acc_s1)/lsig;
i_s_s2        = (flux_s_s2-flux_k_acc_s2)/lsig;

%-------------------------------------------------------------------------
%       MECHANICAL PART
%-------------------------------------------------------------------------

M_el          = p*(-i_s_s1 * flux_s_s2 + i_s_s2 * flux_s_s1);
rhoP_s        = p*INT(1/J_mach*(M_el-M_last) PAR: 306.726);
rho_s         = INTMOD(rhoP_s PAR: 3.71331E-2, pi);
```

```
%-------------------------------------------------------------------------
%        UI-MODEL (fig 5.2)
%-------------------------------------------------------------------------
S1           =  0;   % damping constants
S2           =  1;

%-------------------------------------------------------------------------
%        Estimated machine parameters
%-------------------------------------------------------------------------
r_s_est      = 0.06*R_ref;
r_k_acc_est  = 0.0297*R_ref;
Lsig_est     = 0.53*L_ref;
Lm_est       = 3.2*L_ref;

%--- ROTATION OF THE STATOR VOLTAGE AND CURRENT TO STATOR FLUX REFERENCE FRAME

u_s_p1_obs   =  u_s_s1*cos(phi_s_s_obs) + u_s_s2*sin(phi_s_s_obs);
u_s_p2_obs   = -u_s_s1*sin(phi_s_s_obs) + u_s_s2*cos(phi_s_s_obs);

i_s_p1_obs   =  i_s_s1*cos(phi_s_s_obs) + i_s_s2*sin(phi_s_s_obs);
i_s_p2_obs   = -i_s_s1*sin(phi_s_s_obs) + i_s_s2*cos(phi_s_s_obs);

%--- S1 AND S2 ARE USED IN THE STABILISING LOOP ---

fluxP_s_p1_obs =  (u_s_p1_obs- i_s_p1_obs*r_s_est)/(1-S1);
fluxP_s_p2_obs =  u_s_p2_obs - i_s_p2_obs*r_s_est - S2*fluxP_s_p1_obs;

%--- flux_s_p1_obs IS THE FLUX AMPLITUDE IN THE STATOR FLUX REFERENCE FRAME

flux_s_obs   =  INT(fluxP_s_p1_obs PAR: .76824);

phiP_s_s_obs =  fluxP_s_p2_obs / flux_s_obs;
phi_s_s_obs  =  INTMOD(phiP_s_s_obs PAR: -0.708966, pi);

%--- determine the MODEL FLUX flux_k_p_obs

flux_k_p1_obs=  flux_s_obs - i_s_p1_obs*Lsig_est;
flux_k_p2_obs= -i_s_p2_obs*Lsig_est;

%--- CARTHESISCH to POLAR transformation

flux_k_obs  =  sqrt(flux_k_p1_obs*flux_k_p1_obs+flux_k_p2_obs*flux_k_p2_obs);
phi_k_p_obs =  ATAN2(flux_k_p2_obs,flux_k_p1_obs);

phi_k_s_obs =  phi_k_p_obs + phi_s_s_obs;

%-------------------------------------------------------------------------
%        I/phi-model  (Optimized method)
%-------------------------------------------------------------------------
i_s_b      =  i_s_s1*cos(phi_k_s_obs) + i_s_s2*sin(phi_k_s_obs);
i_s_w      = -i_s_s1*sin(phi_k_s_obs) + i_s_s2*cos(phi_k_s_obs);

flux_k_est = INT(fluxP_k_est PAR: .611175);

Lm_real    = Lm_est-Lsig_est;
fluxP_k_est= (i_s_b - flux_k_est/(Lm_real))*R_k_acc_est;

%--------------- COMPARE THE TWO ESTIMATED FLUXES ----------------------

deltaflux  = flux_k_est - flux_k_obs;

%-------------------------------------------------------------------------
```

E2: OE_DATA.PSM:

```
%-------------------------------------------------------------------------
%  OE_DATA.psm
%
%  PWM INVERTER FED DOUBLE-CAGE INDUCTION MACHINE MODEL
%
%  To apply the OE-method a data-set of stator voltages,current and rotor
%  position is created.
%
%  Inverter properties: pulse number pn=3
%                       stator frequency fs= 40 Hz
%                       DC-voltage = 539 V
%                       Sine modulation (+ 12% third harmonic)
%-------------------------------------------------------------------------
%  reference values for pu-system  (Chap. 3, Van '92)

U_ref      = PAR(311);                        % [V] sqrt(2)*[<Usq(eff)>]n
I_ref      = PAR(88);                         % [A] 2/3*sqrt(2)*[<Isq(eff)>]n
omega_ref  = PAR(100*pi);                     % [rad/s]
R_ref      = PAR(U_ref/I_ref);                % [Ω]
L_ref      = PAR(U_ref/(I_ref*omega_ref));    % [H]
flux_ref   = PAR(U_ref/omega_ref);            % [Vs]
M_ref      = PAR(U_ref*I_ref/omega_ref);      % [Nm]
J_ref      = PAR(U_ref*I_ref/(omega_ref)^3);% [kgm^2]

%-------------------------------------------------------------------------
%  CONTROL PARAMETERS
%-------------------------------------------------------------------------
u_s_des      = PAR(1.73*U_ref);     % amplitude stator-voltage
alfaP_s_s_des= PAR(0.8*omega_ref);  % angle-frequency stator-voltage
M_load       = PAR(0.5*M_ref);      % load-torque in PU * reference

%-------------------------------------------------------------------------
%  PWM-INVERTER MODEL
%-------------------------------------------------------------------------
% ---- INVERTER-MODEL PARAMETERS ----

u_dc         = PAR(517);          % dc-link voltage
Mi           = PAR(0.8);          % modulationindex
pg           = PAR(3);            % pulsnumber
harm3        = PAR(0.12);         % amplitude of 3-th harmonic in ref. signal

% ---- TRIANGLE CARRIER WAVE GENERATION ----

A_tri        = u_s_des/Mi;                      % amplitude triangle wave
alfaP_tri    = (pg*alfaP_s_s_des);
alfa_tri     = INTMOD(alfaP_tri PAR:-1.57153, pi);
u_tri        = A_tri*(abs(alfa_tri)*2/pi - 1);  % triangle carrier wave
u_harm3      = harm3*u_s_des;                   % amplitude of third harmonic

% ---- 3-PHASE REFERENCE VOLTAGES WITH 3th HARMONIC COMPONENT ----

alfa_s_s_des = INTMOD(alfaP_s_s_des PAR: 2.61799, pi); % start value 0
u_s_a        = u_s_des*cos(alfa_s_s_des)       -u_harm3*cos(3*alfa_s_s_des);
u_s_b        = u_s_des*cos(alfa_s_s_des-2*pi/3)-u_harm3*cos(3*alfa_s_s_des);
u_s_c        = u_s_des*cos(alfa_s_s_des+2*pi/3)-u_harm3*cos(3*alfa_s_s_des);
```

```
%-------------------------------------------------------------------------
u_dc          = PAR(539);   % DC-link, mean value of the rectified voltage
Mi            = PAR(0.8);   % Modulationindex
pg            = PAR(3);     % Pulsnumber
harm3         = PAR(0.12);  % Amplitude for 3-de harmonic in ref. signal

%-------------------------------------------------------------------------
% SWITCH EVENTS AT INTERSECTION OF REFERENCE VOLTAGE AND TRIANGLE WAVE
% USING STATE EVENTS (PSI-manual)

s_a_in        =   STE(u_tri-u_s_a PAR:1e-6,-1);
s_a_uit       =   STE(u_tri-u_s_a PAR:1e-6, 1);
s_b_in        =   STE(u_tri-u_s_b PAR:1e-6,-1);
s_b_uit       =   STE(u_tri-u_s_b PAR:1e-6, 1);
s_c_in        =   STE(u_tri-u_s_c PAR:1e-6,-1);
s_c_uit       =   STE(u_tri-u_s_c PAR:1e-6, 1);

timer         =   TIM(1 PAR:1);
state         =   s_a_in|s_a_uit|s_b_in|s_b_uit|s_c_in|s_c_uit|timer;

% ---- SWITCH ACTIONS OF THE INVERTER ----

u_s_a_10      =   FFL(0, s_a_uit, s_a_in PAR: 0,0);
u_s_b_10      =   FFL(0, s_b_uit, s_b_in PAR: 1,0);
u_s_c_10      =   FFL(0, s_c_uit, s_c_in PAR: 0,0);

u_s_aN        =   u_s_a_10*u_dc;    % voltage between positive side and the
u_s_bN        =   u_s_b_10*u_dc;    % negative side (N) of the inverter
u_s_cN        =   u_s_c_10*u_dc;    % output

% ---- VOLTAGE BETWEEN ONE PHASE AND THE STAR NODE OF THE MACHINE ----

u_s_a0        = 1/3*(2*u_s_aN -   u_s_bN   -   u_s_cN);
u_s_b0        = 1/3*( -u_s_aN + 2*u_s_bN   -   u_s_cN);
u_s_c0        = 1/3*( -u_s_aN -   u_s_bN   + 2*u_s_cN);

% -- VOLTAGE REFERENCE CORRECTION AND 3-PHASE --> 2-PHASE TRANSFORMATION ---

u_s_s1        =   1/3*(2*u_s_aN - u_s_bN - u_s_cN);
u_s_s2        =   1/3*(sqrt(3)*u_s_bN - sqrt(3)*u_s_cN);

%*************************************************************************
%-------------------------------------------------------------------------
% DUBBEL-CAGE INDUCTION MOTOR MODEL (see Chap. 2)
%-------------------------------------------------------------------------
% Machine parameters according to the Jordan model
% Given in PU and translated to SI-values
%-------------------------------------------------------------------------
Lh            = PAR(3    * L_ref);
Lsig_s        = PAR(0.12* L_ref);
L_12          = PAR(0.06* L_ref);
Lsig_2        = PAR(0.5 * L_ref);
R_s           = PAR(0.06* R_ref);
R_k1          = PAR(0.8 * R_ref);
R_k2          = PAR(0.04* R_ref);

J_mach        = PAR(1449*J_ref);      % mechanical inertia
```

```
%-------------------------------------------------------------------------
% TRANSFORMATION the MACHINE parameters to MODEL parameters
%-------------------------------------------------------------------------
% transformation to equivalent model with 2 inductances

k             = PAR(Lh/(Lh+L_12));
Lm            = PAR(k*k*(Lh+L_12));
Rk1_acc       = PAR(k*k*R_k1);
Lsig          = PAR(Lsig_s + Lh*L_12/(Lh+L_12));
Lsig_2_acc    = PAR(k*k*Lsig_2);
Rk2_acc       = PAR(k*k*R_k2);

%-------------------------------------------------------------------------
%   ROTOR PART in rotor coordinates
%-------------------------------------------------------------------------

i_k_acc_r1    = -i_s_r1+flux_k_acc_r1/Lm;
i_k_acc_r2    = -i_s_r2+flux_k_acc_r2/Lm;

i_k1_acc_r1   = i_k_acc_r1-i_k2_acc_r1;
i_k1_acc_r2   = i_k_acc_r2-i_k2_acc_r2;

i_k2_acc_r1   = flux_sig2_r1 / Lsig_2_acc;
i_k2_acc_r2   = flux_sig2_r2 / Lsig_2_acc;

E_sig2_r1     = -Rk1_acc*i_k1_acc_r1 + Rk2_acc*i_k2_acc_r1;
E_sig2_r2     = -Rk1_acc*i_k1_acc_r2 + Rk2_acc*i_k2_acc_r2;

flux_sig2_r1  = INT(E_sig2_r1 PAR: -9.485E-2);
flux_sig2_r2  = INT(E_sig2_r2 PAR: -0.188543);

flux_k_acc_r1= INT(-i_k1_acc_r1*Rk1_acc PAR: -1.22539);
flux_K_acc_r2= INT(-i_k1_acc_r2*Rk1_acc PAR: .140882);

%-------------------------------------------------------------------------
% ROTATION OF FLUX AND CURRENT
%-------------------------------------------------------------------------

flux_k_acc_s1= flux_k_acc_r1*cos(rho_s) - flux_k_acc_r2*sin(rho_s);
flux_k_acc_s2= flux_k_acc_r1*sin(rho_s) + flux_k_acc_r2*cos(rho_s);

i_s_r1        =  i_s_s1*cos(rho_s) + i_s_s2*sin(rho_s);
i_s_r2        = -i_s_s1*sin(rho_s) + i_s_s2*cos(rho_s);

%-------------------------------------------------------------------------
% STATOR PART in stator coordinates
%-------------------------------------------------------------------------

flux_s_s1     = INT(u_s_s1-i_s_s1*r_s PAR: .671947);
flux_s_s2     = INT(u_s_s2-i_s_s2*r_s PAR: 1.23295);

i_s_s1        = (flux_s_s1-flux_k_acc_s1)/lsig;
i_s_s2        = (flux_s_s2-flux_k_acc_s2)/lsig;

%-------------------------------------------------------------------------
% MECHANICAL PART
%-------------------------------------------------------------------------

mel           = -i_s_s1 * flux_s_s2 + i_s_s2 * flux_s_s1;
rhoP_s        = INT(1/theta* (mel-mlast) PAR: 247.279);
rho_s         = INTMOD(rhoP_s PAR: -2.0011, pi);
```

```
%*************************************************************************

% -------- FILTERING OUTPUT VALUES FOR DATA-SET WITHOUT ALIASING !!! -------

% --- Using fourth order Butterworth filter's ---

wb         = PAR (1500*2*pi);            % Filter cut-off frequency 1500 Hz
tau2       = par 1/(2*0.38*wb);          % Astrom and Wittenmark
tau1       = par 1/(2*0.92*wb);          % Adaptive control pag. 416.

% ---------------------- FILTER i_s_s1 ----------------------------------
    filt3  = inf (filt2-is1 PAR: 1.3405E-4,tau2,tau2);
    filt2  = int (wb*wb*filt1 par: 13.7941);
    filt1  = inf (i_s_s1-filt2 par: 1.39877E-4,tau1,tau1);
    is1    = int (wb*wb*filt3 par: 12.7686);

% ---------------------- FILTER i_s_s2 ----------------------------------
    filt31 = inf (filt21-is2 PAR: 2.64929E-4,tau2,tau2);
    filt21 = int (wb*wb*filt11 par: 83.7418);
    filt11 = inf (i_s_s2-filt21 par: 2.70726E-4,tau1,tau1);
    is2    = int (wb*wb*filt31 par: 81.7792);

% ---------------------- FILTER u_s_s1 ----------------------------------
    filt32 = inf (filt22-us1 PAR: -3.5402E-9,tau2,tau2);
    filt22 = int (wb*wb*filt12 par: -172.333);
    filt12 = inf (u_s_s1-filt22 par: -6.867E-22,tau1,tau1);
    us1    = int (wb*wb*filt32 par: -172.333);

% ---------------------- FILTER u_s_s2 ----------------------------------
    filt33 = inf (filt23-us2 PAR: 2.7727E-20,tau2,tau2);
    filt23 = int (wb*wb*filt13 par: 298.49);
    filt13 = inf (u_s_s2-filt23 par: 0,tau1,tau1);
    us2    = int (wb*wb*filt33 par: 298.49);

%-------------------------------------------------------------------------
```

APPENDIX F: DSP-PROGRAM'S, TMS320C30 CODE

In this appendix the following DSP-program's are given:

- TFM_ident: two-flux-model method identification
- OE_data: program for storing a data-set used to apply the output-error method;
- Matlab.asm: In this file the macro's for creating a Matlab-file are defined.

F1: TFM_IDENT

To apply the TFM-method the following files are necessery:

- TFM_ID.asm: Main program of the implemented two-flux-models
- BLOCKS.asm: Initialization and definition of variables and macros used in TFM_ID.asm

```
; ************************************************************************
; *  TFM_ID.ASM
; *  ----------
; *          Two-Flux-Model method parameter identification [Bla '94]
; *          for asynchronous machines
; *          On-line tuning of the single-cage machine parameters:
; *          Lm, Lsig, Rk_acc and Rs
; *
; *  INPUT:  3-phase stator voltages and currents and four parameter values
; *  OUTPUT: flux difference between U/I-model flux and I/phi-model flux
; *
; *  TMS320C30 code
; *
; *  author : J. v.d. Werf
; *  updates: S.M. Schilperoort
; *  date   : 15-7-'94
; ************************************************************************

; ************************************************************************
; *  Main code automaticaly called after initialisation
; ************************************************************************

; ------------------------------------------------------------------------
;  Application independent
; ------------------------------------------------------------------------
; --- initialisation : reset and interrupt vectors (application independent)
        .include "../c30/p05cinit.asm"

; --- macro definitions for global names(application independent)
        .include "../c30/globdef.asm"

; --- include mathlib for testing purposes (application independent)
        .include "../c30/mathlib.asm"

; --- input-output data migration through ADCs (application independent)
        .include "../c30/p06.asm"
```

```
; ------------------------------------------------------------------------
;  Application dependent
; ------------------------------------------------------------------------
; --- auxiliary (global) variables for program flow (application dependent)
        .data
        definei sample,          402    ; timer period, 402 == 100us
        define  sample_pu, 3.1415927e-2 ; (timer period)*100pi
        define  _sample_pu, 31.830989   ; 1pu time == 1/(2pi*50) sec

; --- block, variable definitions and used macros
        .include blocks.asm

; --- interrupt services (application dependent)
        .include "../c30/isr1.asm"

; ************************************************************************
; -------------------- beginning of code segment --------------------------
; --------- all program code should be inserted after this point ----------
;-------------------------------------------------------------------------
        .text                       ;start of program code
PROGRAM:  LDP   PROGRAM,DP          ;label PROGRAM defined at p05boot.asm

;--- set sample time ---
        LDI     @T0PER,AR0          ;timer 0 period register
        LDI     @sample,R0          ;desired period
        STI     R0,*AR0             ;AD and DA conv start at +flanc
        LDF     @sample_pu,R0       ;@sample_pu == integration step
        STF     R0,@dt              ;
        LDF     @_sample_pu,R0      ;
        STF     R0,@_dt             ;

;--- loop from conv to conv ---
lus:    LDI     EINT1,IE            ;enable ext int 1 only (comes from ADC)
        XOR     IF,IF               ;reset int request occurence bits
        OR      GIE,ST              ;enable execution interrupt
        IDLE                        ;wait interrupt 1, then execute ISR1
                                    ;(almost) nothing happens inside ISR1
                                    ;at return IE() == 00H & GIE-bit == 0
        XOR IF,IF                   ;reset int bits, always do that!

;-------------------------------------------------------------------------
; ------------------------- I/O instruction -------------------------------

; --- Measured voltage and current ---
        point_to p06_b1
        import @i_sa_mes,ch1,scimp_1$96      ; Scaling factors
        import @i_sb_mes,ch2,scimp_1$96      ; scimp_@ are defined
        import @u_sa,ch3,scimp_2$14          ; in "p06.asm".
        import @u_sb,ch4,scimp_2$14
        import @u_sc,ch5,scimp_2$14

; --- Machine parameters ---
        point_to p06_b2
        import @inv_Lm_est,ch1,scimp_0$60
        import @Lsig_est,ch2,scimp_1$00
        import @R_s_est,ch3,scimp_0$10
        import @R_k_acc_est,ch4,scimp_1$00
```

```
;*************************************************************************
; Two-Flux-Model method for parameter identification
;-------------------------------------------------------------------------

;-------------------------------------------------------------------------
; Transformation of voltage and current to a 2-dimensional representation
; (see BLOCKS.asm)
;-------------------------------------------------------------------------
        u_from_3_to_2
        IUVW_to_s1s2 @i_sa_mes,@i_sb_mes,@i_s_s1,@i_s_s2

;-------------------------------------------------------------------------
; UI-model with feedback, (see Fig. 5.2).
;-------------------------------------------------------------------------
;--- Check for statorflux. It has to be positive, > 0.01 and < 3.0 ---
        LDF  @Sflux_s_obs,R7
        BP   posflux
        NEGF R7                 ; make flux positive
        STF  R7,@Sflux_s_obs
        LDF  @Sphi_s_s_obs,R6   ; plus or minus pi for statorflux angle
        CMPF @PI,R6
        BP   minPI
        ADDF @PI,R6             ; plus pi
        STF  R6,@Sphi_s_s_obs
        B    posflux
minPI   SUBF @PI,R6             ; minus pi
        STF  R6,@Sphi_s_s_obs
posflux CMPF @THREE,R7          ; compare flux with 3.0
        BN   smalthr            ; Smaller then Three
        LDF  @THREE,R5
        STF  R5,@Sflux_s_obs
        B    vectrot            ; Flux is correct
smalthr CMPF 0.01,R7            ; compare flux with 0.01
        BGE  vectrot
        LDF  0.01,R7
        STF  R7,@Sflux_s_obs

;-------------------------------------------------------------------------
; Rotation to stator flux-coordinates
;-------------------------------------------------------------------------
vectrot LDF @Sphi_s_s_obs,R7
        sin R7                  ; output of sin function to R0
        NEGF R0
        STF R0,@sin_neg
        cos R7                  ; output of cos function to R0
        STF R0,@cos_neg

        vr @cos_neg,@sin_neg,@u_s_s1,@u_s_s2,u_s_ps1_obs,u_s_ps2_obs
                        ; rotate stator voltage to flux coordinates
        vr @cos_neg,@sin_neg,@i_s_s1,@i_s_s2,i_s_ps1_obs,i_s_ps2_obs
                        ; rotate stator current to flux coordinates
;-------------------------------------------------------------------------
; EM-voltages in U/I-model
;-------------------------------------------------------------------------
        LDF @u_s_ps1_obs,R4
        LDF @u_s_ps2_obs,R5
        LDF @i_s_ps1_obs,R6
        LDF @i_s_ps2_obs,R7
        MPYF @R_s_est,R6  ; R6=R_s_est*i_s_ps1_obs
        MPYF @R_s_est,R7  ; R7=R_s_est*i_s_ps2_obs
        SUBF R6,R4        ; R4=e_s_ps1_obs=u_s_ps1_obs-R_s_est*i_s_ps1_obs
        SUBF R7,R5        ; R5=e_s_ps2_obs=u_s_ps2_obs-R_s_est*i_s_ps2_obs
```

```
;-------------------------------------------------------------------------
; Stator-flux with feedback
;-------------------------------------------------------------------------
        MPYF @S1p,R4      ; R4 = fluxP_s_obs
        LDF  R4,R6        ; R6 for later use
        integrate Sflux_s_obs,R4,ONE
;-------------------------------------------------------------------------
; Stator-flux angle in statorcoordinates with feedback
;-------------------------------------------------------------------------
        MPYF @S2,R6
        ADDF R6,R5
        div  R5,@Sflux_s_obs         ; output to R0
        LDF  R0,R4
        integrate Sphi_s_s_obs,R4,ONE
;-------------------------------------------------------------------------
; Cage-flux of rotor-cage
;-------------------------------------------------------------------------
        LDF  @i_s_ps1_obs,R6       ;
        LDF  @i_s_ps2_obs,R7       ; flux_k_ps1=i_s_ps1*Lsig_est+flux_s
        MPYF @Lsig_est,R6          ; flux_k_ps2=i_s_ps2*Lsig_est
        MPYF @Lsig_est,R7          ;
        SUBF @Sflux_s_obs,R6       ;
        NEGF R6                    ;
        NEGF R7
        capo R6,R7,flux_k_acc_obs,phi_k_ps_obs
        LDF  @phi_k_ps_obs,R5
        ADDF @Sphi_s_s_obs,R5
        STF  R5,@phi_k_s_obs

;-------------------------------------------------------------------------
; Estimation of the cage-flux using the I/phi-model (Fig.3.1)
;-------------------------------------------------------------------------
; Rotation of the stator current to shortcircuit rotor-flux-coordinates
;-------------------------------------------------------------------------
        LDF  @phi_k_s_obs,R7
        sin  R7              ; output of sin function to R0
        NEGF R0
        STF  R0,@sin_neg
        cos  R7              ; output of cos function to R0
        STF  R0,@cos_neg
        VR   @cos_neg,@sin_neg,@i_s_s1,@i_s_s2,i_s_b_nr,i_s_w_nr

;-------------------------------------------------------------------------
; Integration and feedback loop = I/phi-model
;-------------------------------------------------------------------------
        LDF  @i_s_b_nr,R4
        LDF  @Sflux_k_acc_nr,R1
        MPYF @inv_Lm_est,R1
        SUBF R1,R4
        MPYF @R_k_acc_est,R4
        integrate Sflux_k_acc_nr,R4,ONE

;-------------------------------------------------------------------------
; Comparing the two estimated fluxes "flux_k_acc_est" and "flux_k_acc_nr"
;-------------------------------------------------------------------------
        LDF  @flux_k_acc_obs,R0
        SUBF @Sflux_k_acc_nr,R0
        STF  R0,@delta_flux_k     ; flux difference
```

```
;-------------------------------------------------------------------------
; Determine stator flux in stator coordinates for visualiation
;-------------------------------------------------------------------------
        LDF  @Sphi_s_s_obs,R7
        sin  R7                  ; output of sin function to R0
        STF  R0,@sin_pos
        cos  R7                  ; output of cos function to R0
        STF  R0,@cos_pos
        poca @Sflux_s_obs,@cos_pos,@sin_pos,flux_s_s1,flux_s_s2

; -------------------------------------------------------------------------
; Write to analog outputs A/D-board p06
; -------------------------------------------------------------------------
        point_to p06_b1
        export  @flux_k_acc_obs,ch1,scexp_4$00   ; Scaling factors
        export  @Sflux_k_acc_nr,ch2,scexp_4$00   ; scexp_@ are defined
        export  @delta_flux_k,ch3,scexp_0$20     ; in "p06.asm".
        export  @u_s_s1,ch4,scexp_4$00
        export  @u_s_s2,ch5,scexp_4$00
        export  @i_s_s1,ch6,scexp_4$00
        export  @i_s_s2,ch7,scexp_4$00
        export  @m_el_obs,ch8,scexp_1$00

; -------------------------------------------------------------------------
; --- the cycle is now finished ...
; -------------------------------------------------------------------------
        update_states                   ;update states to new values
        BR   lus                        ;jump back

; -------------------------------------------------------------------------
        .end                           ;END OF CODE
; -------------------------------------------------------------------------
```

```
; ************************************************************************
; *  blocks.asm:
; *
; *  Initialisation of variables and MACROS used in TFM_ID.asm
; *
; *  update : S.M. Schilperoort
; *  date   : 15-07-94
; ************************************************************************
; --------- math-macros and  precompiled C-routines
;         .include mathlib.asm  (already included at sinusmod.asm)
; ************************************************************************
; * declare all used variables
; * reserve space and assign to initial value in data section
; ************************************************************************
         .data
;*************** simulation step ******************

         define dt,              3.1415927e-2 ; == 100 us, defaut
         define _dt,             31.830989    ; don't forget to reassign  dt

;*************** machine parameters *****************

;--- Initial values of the estimated machine parameters [PU] ---
         define r_s_est,         0.39   ;  stator resistance
         define r_k_accacc_est,  0.03   ;  rotor resitance
         define lsig_acc_est,    0.3    ;  leakage inductance
         define inv_lm_est,      0.38   ;  1/lm_est

; ------------------------------------------------------------------------
;                 Input/Output vaiables
; ------------------------------------------------------------------------
;--- measured signals ---
         define  i_sa_mes,        0.0    ; current in fase a
         define  i_sb_mes,        0.0    ; current in fase b
         define  i_sc_mes,        0.0    ;
         define  u_sa,            0.0    ;
         define  u_sb,            0.0    ;
         define  u_sc,            0.0    ;
         define  rho_s,           0.0
         define  sinrho,          0.0
         define  cosrho,          1.0
         define  rho_max,         0.0

         define  d_m_el_nr,       0.0   ; error in torque
         define  delta_flux_k,    0.0   ; flux difference between the
         define  flux_s_s1,       0.0   ; flux in stator in statorcoordinates
         define  flux_s_s2,       0.0   ; used for verification
         define  flux_k_ps1,      0.0
         define  flux_k_ps2,      0.0

;------------------------------------------------------------------------
;                 Internal variables
;------------------------------------------------------------------------
;----------------------------- non-states -------------------------------

; --- voltages and currents in s1/s2 coordinates ---
         define  u_s_s1,          0.0    ;
         define  u_s_s2,          0.0    ;
         define  i_s_s1,          0.0    ;
         define  i_s_s2,          0.0    ;
```

```
; --- Variables for the UI_model---

; --- signals ---
        define  sin_neg,          0.0 ;
        define  cos_neg,          1.0 ;
        define  u_s_ps1_obs,      0.0 ;
        define  u_s_ps2_obs,      0.0 ;
        define  i_s_ps1_obs,      0.0 ;
        define  i_s_ps2_obs,      0.0 ;

;--- feedback variables U/I-model---
        define  S1p,              1.0 ;
        define  S2,              -1.0 ;

;--- fluxes ---
        define flux_k_acc_obs,    1.0 ;
        define phi_k_s_obs,       0.0 ;
        define phi_k_ps_obs,      0.0 ;
        define sin_pos,           0.0 ;
        define cos_pos,           1.0 ;
        define eps_k_ps,          0.0 ;
        define i_k_est,           0.0 ;
        define i_k_ps1,           0.0 ;

;--- currents in flux coordinates used in the I/phi-model ---
        define i_s_b_nr,          0.0 ;
        define i_s_w_nr,          0.0 ;

;-------------------------------------------------------------------------
;---------------------------- state-variables ----------------------------
; don't forget to assign good initial values to get a stable start

;----  scalar states  --------

Astates           .word states
Anew_states       .word new_states

states  define    Sflux_s_obs,    1.0
        define    Sflux_k_acc_nr, 1.0

;---- angle-states (states are always between -2 PI and 2 PI) -----
anglst
        define    Salpha,         0.0
        define    Sphi_s_s_obs,   0.0
new_states
        .asg      (new_states-states),NUMBER_STATES   ;Used in update_states
        .asg      (new_states-anglst),NUMBER_SCALAR   ;Used in update_states

;---- reserve space for new calculated states   ------
        .space    NUMBER_STATES

;- define space for previous value of state-input (used in integrate macro)-

bak     define    Sflux_s_obs_Bak,    0.0
        define    Sflux_k_acc_nr_Bak,0.0
        define    Salpha_Bak,         1.0
        define    Sphi_s_s_obs_Bak,   0.0
```

```
; ************************************************************************
; ************************** MACROS ***************************************

; ************************************************************************
; *  macro parameter name conventions:
; *      R_<name>       : register type parameter : R0-R7
; *      RD_<name>      : register type parameter : R0-R7
; *                       or direct type parameter : @....
; *      A_<name>       : pointer type parameter, adress of a variable
; *
; ************************************************************************

; ************************************************************************
; *  macro     : update_states
; *
; *  calling syntax:    update_states
; *  description:       Copy new calculated state-values to current
; *                     state var. Keeps angle type state between
; *                     -2PI and 2PI
; *  inputs:    -
; *  output:    -
; *  uses:      R0, AR0, AR1
; *
; *  cycles:    18
; ************************************************************************
update_states  .macro
               .asg 0,x
               LDI @Astates,AR1            ; copy to states
               LDI @Anew_states,AR0        ; copy from new_states
               LDF *AR0++,R0               ; load R0 with first state
               .loop
               .break (x==NUMBER_STATES-1) ; loop until all states are copied
               LDF *AR0++,R0               ; store st and load next st in R0
               STF R0,*AR1++
               .eval x+1,x
               .endloop
               STF R0,*AR1                 ; store last state
               .endm

; ************************************************************************
; *  macro     : IUVW_to_s1s2
; *
; *  calling syntax:  IUVW_to_s1s2
; *  description:      transformation of 3 to 2 phase representation
; *                    In stator coordinates.
; *                    Using 2 measured stator currents
; *
; *  inputs:    I_sa,I_ab
; *  outputs:   I_s_s1,I_s_s2
; *  uses:      R0,R1
; *  updates:
; *  cycles:    8
; ************************************************************************
IUVW_to_s1s2   .macro I_sa,I_sb,I_s_s1,I_s_s2

               LDF   I_sa,R0
;              NEGF  R0
               MPYF  0.5,R0          ;R0 = 1/2 i_sa
               LDF   I_sb,R1         ;R1 = i_sb
;              NEGF  R1
               ADDF  R0,R1           ;R1 = 1/2 i_sa + i_sb
```

```
                MPYF  @SQR3,R1        ;R1 = i_s_s2 = sqr(3) {1/2 i_sa + i_sb}
                MPYF  3.0,R0          ;R0 = i_s_s1 = 1 1/2 i_sa
                STF   R0,I_s_s1
                STF   R1,I_s_s2
                .endm


; ************************************************************************
; *  macro     : u_from_3_to_2   (Use with BRAMAMATIC EM 1567)
; *
; *  calling syntax:    u_from_3_to_2
; *  description:       voltage transformation
; *
; *  inputs:    U_sa,U_sb,U_sc
; *  outputs:   U_s_s1, U_s_s2
; *  uses:      R1,
; *  updates:
; *  cycles:
; ************************************************************************
u_from_3_to_2 .macro U_sa,U_sb,U_sc,U_s_s1,U_s_s2

                LDF  U_sa,R1              ; stator voltages
                NEGF R1                   ; sign correction due to measurement
                STF  R1,U_s_s1            ; u_s_s1 = u_sa
                LDF  U_sb,R1              ;
                SUBF U_sc,R1              ;
                NEGF R1                   ; sign correction
                MPYF @INVSQR3,R1          ; u_s_s2 = 1/sqr3 usb - 1/sqrt3 usc
                STF  R1,U_s_s2            ;
                .endm

; ************************************************************************
; *  macro     : i_from_3_to_2
; *
; *  calling syntax:    i_from_3_to_2
; *  description:       current transformation
; *
; *  inputs:    I_sa,I_sb,I_sc
; *  outputs:   I_s_s1,I_s_s2
; *  uses:      R1,
; *  updates:
; *  cycles:
; ************************************************************************
i_from_3_to_2 .macro I_sa,I_sb,I_sc,I_s_s1,I_s_s2

                LDF  @I_sb,R1          ;
                ADDF @I_sc,R1          ;
                MPYF @HALF,R1          ;
                NEGF R1                ; -1/2*(I_sb+I_sc)
                ADDF @I_sa,R1          ; I_sa-1/2*(I_sb+I_sc)=I_s_s1
                STF  R1,@I_s_s1        ;
                LDF  @I_sb,R1          ;
                SUBF @I_sc,R1          ;
                MPYF @HALFSQR3,R1      ; 1/2*sqrt(3)*(I_sb-I_sc)=I_s_s2
                STF  R1,@I_s_s2
                .endm
; ************************************************************************
```

F2: OE_DATA

The macros used in OE_DATA.asm are also defined in BLOCKS.asm given in Appendix F1.

```
; ************************************************************************
; ************************************************************************
; *  OE_DATA.asm:   Used for creating a data-set to apply to the
; *                 OE-identification
; *
; *  storing:       stator voltage and current in the two phase
; *                 stator coordinates
: *
; *  Using:         MATLAB.ASM (macro's MATINIT and NAARMAT) storing
; *                 data in the external memory of the C30 DSP-system,
; *                 in the debugger this data can be put on disk.
; *                 With the C-program "STRIPPER" the data can be
; *                 transformed to matlab format.
; *  TI320C30 code
; *
; *  author : S.M. Schilperoort
; *  date   : 06-08-'94
; ************************************************************************
; ************************************************************************
; *    Main code automatically called after initialisation
; ************************************************************************

; ************************************************************************
;      Application independent
; ************************************************************************

; --- initialisation : reset and interrupt vectors (application independent)
       .include "../c30/p05cinit.asm"
;
; --- macro definitions for global names(application independent)
       .include "../c30/globdef.asm"

; --- include mathlib for testing purposes (application independent)
;     mathematical library
       .include "../c30/mathlib.asm"

; --- include matlab to create a matlab file
       .include matlab1.asm

; --- input-output data migration through ADCs (application independent)
       .include "../c30/p06.asm"

; --- input-output data migration through fully digitised board
       .include "../c30/p09.asm"

; ************************************************************************
; *    Application dependent
; ************************************************************************

; --- block definitions and macros describing the machine
       .include blocks.asm

; --- interrupt services (application dependent)
       .include "../c30/isr1.asm"
```

```
; ---------------------------------------------------------------------
; --- auxiliary (global) variables for program flow (application dependent)
; --- define sample time and pu translation ---

       .data
       definei sample,          402      ; timer period, 402 == 100us
       define  sample_pu, 3.1415927e-2 ; (timer period)*100pi
       define  _sample_pu, 31.830989   ; 1pu time == 1/(2pi*50) sec

; *********************************************************************
;      Declare all used variables
;      reserve space and assign to initial value in data section
; *********************************************************************
; --- Simulation step ---
                                        ; 1pu == 1/100pi sec
       define dt,         3.1415927e-2    ; == 100 us, default
       define _dt,        31.830989       ; don't forget to reassign  dt

; --- measured phase voltages ---
       define  u_sa_mes,        0.0
       define  u_sb_mes,        0.0
       define  u_sc_mes,        0.0

; --- measured phase currents ---
       define  i_sa_mes,        0.0
       define  i_sb_mes,        0.0

; --- semi four phase voltages ---
       define  u_s_s1,          0.0
       define  u_s_s2,          0.0

; --- semi four phase currents ---
       define  i_s_s1,          0.0
       define  i_s_s2,          0.0

; --- angles ---
       define  rho_s,           0.0

; --- variables used for "naarmat" ---
       definei time,            0  ; counts the number of time steps
       definei counter,         0  ; used for matlab storage
       definei previous,        0  ; used for matlab storage
       define  start,           0  ; command to save data

; --- filtered values ---
       define  u_s_s1_f,        0
       define  u_s_s2_f,        0
       define  i_s_s1_f,        0
       define  i_s_s2_f,        0

; *********************************************************************
; *---------------------- Beginning of code segment ----------------------
; *--------- The program code should be inserted after this point ----------
; *********************************************************************
              .text                ;start of program code

PROGRAM:LDP    PROGRAM,DP           ;label PROGRAM defined at p05boot.asm

;-------------------------- set sample time -------------------------------
       LDI     @T0PER,AR0           ;timer 0 period register
       LDI     @sample,R0           ;desired period
       STI     R0,*AR0              ;AD and DA conv start at +flanc
```

```
      LDF     @sample_pu,R0      ;@sample_pu == integration step
      STF     R0,@dt             ;
      LDF     @_sample_pu,R0     ;
      STF     R0,@_dt            ;
      LDI     0, R0              ; initialisation of time and
      STI     R0, @time          ; used in naarmatlab

;-----------------------Initialisation matlab file -----------------------
;
;   sample frequency 10kHz , 12000 samples at 5kHz
;   total measuring time = 2.4 sec
;--------------------------------------------------------------------------

      matinit @dt,2,12000,"""u_s_s1_f""","""u_s_s2_f""","""i_s_s1_f""",
            """i_s_s2_f""","""rho_s"""

;--------------------- Program loop from conv to conv ---------------------
lus:   LDI    EINT1,IE           ;enable ext int 1 only (comes from ADC)
      XOR     IF,IF              ;reset int request occurence bits
      OR      GIE,ST             ;enable execution interrupt
      IDLE                       ;wait interrupt 1, then execute ISR1
                                 ;(almost) nothing happens inside ISR1
                                 ;at return IE() == 00H & GIE-bit == 0
      XOR IF,IF                  ;reset int bits, always do that!

;--------------------------inport values ----------------------------------
      point_to p06_b1
      import @i_sa_mes,ch1,scimp_100$0  ; measured currents result in [A]
      import @i_sb_mes,ch2,scimp_100$0  ;
      import @u_sa_mes,ch3,scimp_666$0  ; measured voltages result in [V]
      import @u_sb_mes,ch4,scimp_666$0  ;
      import @u_sc_mes,ch5,scimp_666$0  ;
      import @start,ch8,scimp_10$0      ; if value of switch is > 2V,
                                        ; then start storing data
      point_to p09
      get_rot_pos @rho_s                ; rotor position [rad]

;--------------------------------------------------------------------------
;     3->2 transformation to a 2-dimensional frame
;     result: u_s_s1,u_s_s2,i_s_s1 and i_s_s2
;--------------------------------------------------------------------------

      u_from_3_to_2 @u_sa_mes,@u_sb_mes,@u_sc_mes,@u_s_s1,@u_s_s2
      IUVW_to_s1s2 @i_sa_mes,@i_sb_mes,@i_s_s1,@i_s_s2

;--------------------------------------------------------------------------
;     write data to matlab file, if write=0
;
; the macro naarmat uses the actual time and the start command "counter" if
; counter is -1 no storage, if counter is 0 , the desired number of mesure-
; ments is stored in section matlab. The number of samples is
; initialised in macro "matinit"
;--------------------------------------------------------------------------
      LDF  @start,R0                    ; if input value start > 2V
      CMPF @TWO,R0
      BP   start0                       ; then data aquisition is started
      LDI  0,R1
      STI  R1,@counter
start0
      naarmat @time,2,@counter,@previous,@u_s_s1_f,@u_s_s2_f,
              @i_s_s1_f,i_s_s2_f,@rho_s
```

```
;-------------------------------------------------------------------------
;      Write variables to D/A-converter
;-------------------------------------------------------------------------
       point_to p06_b1
       export   @u_s_s1,ch1,scexp_666$0
       export   @u_s_s2,ch2,scexp_666$0
       export   @i_s_s1,ch3,scexp_100$0
       export   @i_s_s2,ch4,scexp_100$0
       export   @u_sa_mes,ch5,scexp_666$0
       export   @rho_s,ch8,scexp_8$00

;-------------------------------------------------------------------------
; the cycle is now finished ... increment the timer
;-------------------------------------------------------------------------
       LDI      @time, R1
       ADDI     1, R1
       STI      R1, @time

       BR       lus                ; jump back to start new
                                   ; calculation

;-------------------------------------------------------------------------
       .end                        ;END OF CODE
;-------------------------------------------------------------------------
```

```
*************************************************************************
*      P09.asm
*      ------
*      input-output signal flow through P09-boards
*
*************************************************************************
* by JDuarte
* updated by S.M. Schilperoort 25-8-94
*************************************************************************
*--- addressing the fully digital IO boards ---
        .data              ;warning: check possible conflict with p06
;p09    .word   00804000H   ;switch sw40  (3 2 1) must be: 111
;p09    .word   00804400H   ;                              110
p09     .word   00804800H   ;                              101
*
*-----------------------------------------------------------------
*--- offsets (from basic address) for the registers at each board ---
ax_pos          .set 3      ; 16 bits, 360 degrees == 900 points;
                            ;  reset to 0H (+sense) or FFFFH (-sense)
                            ;  by input /OH after each revolution;
                            ;  Pos dir : 0000H --> 0384H (= 900),
                            ;  Neg dir : FFFFH --> FC7BH
                            ;  (in this case add +1 >>> 2-cmps !)
*-----------------------------------------------------------------
*--- data handle and scaling ---
*-----------------------------------------------------------------
; ax position (angular position with a 1000 points encoder)
mask_16      .set   0FFFFH           ;allows 16 LSBs
mask_12      .set   0FFFH
mask_msb      .set   4000H              ;allows only bit 14
```

```
neg_prefix    .word  0FFFF8000H          ;sets the 17 MSBs of a 32 bits reg
sc_ax_pos     .float 6.283185307e-3      ; = 2pi/1000 points

sc_rot_pos    .float 1.570796327e-3      ; = 2pi/4000 points

*---------------------------------------------------------------------
*--- macro definitions for general purpose use ---
;---------------------------------------------------------------------
; macro for reading rotor position with an encoder
;---------------------------------------------------------------------
get_ax_pos  .macro  value
            LDI     *+AR7(ax_pos),R7    ;read value from channel
            AND     mask_12,R7          ;
            TSTB    mask_msb,R7         ;test rotation sense
            BZ      posdir?             ;
            OR      @neg_prefix,R7      ;extend to 32bits
            ADDI    1000,R7             ;
            B       fl?
posdir?     AND     07fffH,R7
fl?         FLOAT   R7                  ;
            MPYF    @sc_ax_pos,R7       ;
            STF     R7,value            ; store input value as float
            .endm

;---------------------------------------------------------------------
; macro for reading rotor position with a 1000 points encoder
; extended to use with a 12-bits 4000 up/down counter.
; Counter on digital board (EPLD) count up/down dependent on the rotation
; direction. As a result of the used construction the counter counts
; form 40 to 4040. (S.M. Schilperoort)
;---------------------------------------------------------------------
get_rot_pos .macro  value
            LDI     *+AR7(0),R7         ; Read value from channel
            AND     mask_12,R7          ; use only 12 bits
            FLOAT   R7                  ;
            SUBI    40,R7               ; "off-set" correction
            MPYF    @sc_rot_pos,R7      ; scale integer value, result 0 - 2pi
            STF     R7,value            ; store input value as float
            .endm
;---------------------------------------------------------------------
```

F3: STORE DATA IN MATLAB-FILE

By S. Bosga, Technical University Eindhoven, Department EMV.

The macros defined in "MATLAB.asm" are used to store data in the C30 DSP system and to make a MATLAB data-file. To do this two macros are used: *matinit* defines and initialises memory space and *naarmat* stores the desired data in the DSP memory. The stored data can then be written to disc, using the *Debugger*. Using the program *Stripper* the file is changed to matlab-format.

Macro: matinit

call: *matinit dT,step,length,P1,.....P8*

The macro *matinit* creates a section "*matsec*" in the internal DSP memory. The first variable stored is the sample time in [pu]. Variable *step* is the number time-steps between two data samples. *Length* determines the number of samples per variable. *P1..P8* are the names of the desired variables to store. The variable names must be placed between triple quotes. The macro matinit also defines the variable: *matlen* which is the length of the matlab-file in words. *Matinit* uses the macro *mateen* to reserve and initialize the memory-space per variable. mateen also defines a table *mattab* (data-section). In this table the starting adresses of the memory space are placed.

Macro: naarmat

Call: *naarmat time,step,counter,previous,P1,.....P8*

This macro is called each timestep. *Step* is the numer of time steps between two storage of data. The data is stored if *counter* equals zero. *Counter* is incremented evert time data is stored until the defined memory space is full. At that moment *counter* is set to -1.

Store on disc

This is done in the *Debugger* using the instruction "*ms matsec, *matlen,filename*". The debugger automaticly add's the extension ".obj". The resulting data-file is given in the COFF-format. Exept the desired data a header is added to the file.

Conversion to Matlab-format: Stripper.exe

The program *Stripper* removes the header of the stored data-file. Using *Stripper* results in a matlab-file (".mat") which can directly be used in MATLAB.

```
; ************************************************************************
; *     MATLAB.asm
; *
; *     Macro's for creating a Matlab file with the TMS320C30
; *
; *     Auteur:         Sjoerd Bosga(changes for use with C30
; *                     by Simon Schilperoort)
; *     Creation:       22 maart 1994
; *     updates:        11 augustus 1994
; ************************************************************************
; ########################################################################
; #     Macro:  matinit (initialisation of a Matlab file)
; #     In:     Dt, Tussen, Lengte (variable ore number)
; #             P1 t/m P8 (names of variables)
; #     Reg's:  R1, AR1
; #     SGB, 22 maart 94
; #     n.b. Names of variables must be placed between
; #          triple quotes (bv. ,"""u_s_s1""",...)
; #-----------------------------------------------------------------------
matinit  .macro  Dt,Tussen,Lengte,P1,P2,P3,P4,P5,P6,P7,P8
         .mlist
; -----------------------------------------------------------------------
matsec   .sect   "matlab"           ; start of matlab file
         .global matsec
; -----------------------------------------------------------------------
         .text
         LDF     Dt, R1
         MPYF    Tussen, R1         ; R1=Dt*Tussen=time between measurements
         TOIEEE  R1, R1
         mateen  """deltat""",1,R1  ; Matlab-variable deltat = R1
; -----------------------------------------------------------------------
mattab   .data                      ; table is a table with the start adresses
                                    ; of the storage space
; -----------------------------------------------------------------------
         .text                      ; Create storage space for P1 t/m P8
         LDF     0.0, R1            ; R1 = initial value storage space
         TOIEEE  R1, R1
         mateen  P1,Lengte,R1       ; Lengte=number of words in storagespace
         mateen  P2,Lengte,R1
         mateen  P3,Lengte,R1
         mateen  P4,Lengte,R1
         mateen  P5,Lengte,R1
         mateen  P6,Lengte,R1
         mateen  P7,Lengte,R1
         mateen  P8,Lengte,R1
; -----------------------------------------------------------------------
         .sect   "matlab"           ; Sign for end of matlab file
matend   .word   99999999h          ;
         .global matend
; -----------------------------------------------------------------------
         .data                      ; Adress of table with adresses
mattab   .word   mattab
matlen   .word   matend-matsec      ; Length of matlab file
opslen   .word   Lengte             ; Space for 1 variable
         .global amattab
         .global matlen
         .global opslen
; -----------------------------------------------------------------------
         .text                      ; Continue assemble in .text
         .endm
; -----------------------------------------------------------------------
```

```
; ######################################################################
; #     Macro:  mateen  (init. of one varable in a Matlab file)
; #     In:     Naam (string between triple quotes)
; #             Len (variable ore number)
; #             Initwaarde (register, IEEE format)
; #             P1 t/m P8 (names of variables)
; #     Reg's:  R1, AR1
; #     SGB, 22 maart 94
; #     n.b. Only used by macro "matinit"
; #---------------------------------------------------------------------
mateen   .macro   Naam,Len,Initwaarde
         .if      ($symlen(Naam)>0)
                                        ;Length of a variabele name plus 1
         .asg     $symlen(Naam)+1-2,Naml ;null comming up) (-2 for quotes)
         .asg     """""", Spaties
         .if      (Naml=5)
         .asg     0, Spaties
         .endif
         .if      (Naml=9)
         .asg     0, Spaties
         .endif
         .if      (Naml=13)
         .asg     0, Spaties
         .endif
; ---------------------------------------------------------------------
.sect    "matlab"                  ; Matlab file format: see Matlab User
                                   ; Guide p. 3-122
         .word    10               ; type(PC,column-wise,single-precision,no text)
         .word    1                ; mrows (1 row)
         .word    Len              ; ncols (Len column)
         .word    0                ; imagf (no imaginar part)
         .word    Naml             ; namlen (length of variable name)
         .string  Naam, Spaties    ; name of variable and zero's
mdata?   .space   Len              ; storage space
; ---------------------------------------------------------------------
         .data
amdata?  .word   mdata?            ; start adress of space reserved for this var.
; ---------------------------------------------------------------------
         .text
         LDI      @amdata?, AR1
         .if      (Len>1)          ; if more than one word: repeat
         LDI      (Len-1), RC
         RPTS
         .endif
         STI      Initwaarde, *AR1++ ; initialisation of storage space
                                     ; Caution  STI ipo STF for C30 use !!
; ---------------------------------------------------------------------
         .endif
         .endm
; ---------------------------------------------------------------------
```

```
; ##########################################################################
; #     Macro:  naarmat (write to Matlab file)
; #     In:     Detijd, Tussen, (variable ore number)
; #             Teller, Eerder, (variables)
; #             P1 t/m P8 (variables ore numbers)
; #     Reg's:  R0, R1
; #     SGB, 24 maart 94
; #-------------------------------------------------------------------------
naarmat  .macro   Detijd,Tussen,Teller,Eerder,P1,P2,P3,P4,P5,P6,P7,P8
         LDI      Teller, R1       ; If (Teller=-1) no storage
         CMPI     -1, R1
         BEQ      naaren?

         CMPI     0, R1            ; If (Teller=0) start storage
         BEQ      naart0?

         LDI      Detijd, R0       ; Store ones per (Tussen) time steps
         SUBI     Eerder, R0       ;
         CMPI     Tussen, R0
         BLT      naaren?

         LDI      -1, R0
         CMPI     @opslen, R1      ; If (Teller>=max) than storage space
         BGE      naarmv?          ; full and make Teller= -1

naart0?  LDI      Detijd, R1
         movel    R1, Eerder
         LDI      @amattab, AR0    ; AR0 := adress of mattab
         LDI      Teller, IR0      ; IR0 := Teller

         .if      ($symlen(P1)>0) ; Store P1 on adress:
         LDF      P1, R0           ;  *+(*+mattab(0))(Teller)
         TOIEEE   R0, R0
         LDI      *AR0++, AR1      ; AR1 := adress of P1 storage space
         STI      R0, *+AR1(IR0)   ; let op STI ipv STF voor C30!!
         .endif

         .if      ($symlen(P2)>0) ; Store P2 on adress:
         LDF      P2, R0           ;  *+(*+mattab(1))(Teller)
         TOIEEE   R0, R0
         LDI      *AR0++, AR1      ; AR1 := adress of P2 storage space
         STI      R0, *+AR1(IR0)
         .endif

         .if      ($symlen(P3)>0) ; Store P3 on adress:
         LDF      P3, R0           ;  *+(*+mattab(2))(Teller)
         TOIEEE   R0, R0
         LDI      *AR0++, AR1      ; AR1 := adress of P3 storage space
         STI      R0, *+AR1(IR0)
         .endif

         .if      ($symlen(P4)>0) ; Store P4 on adress:
         LDF      P4, R0           ;  *+(*+mattab(3))(Teller)
         TOIEEE   R0, R0
         LDI      *AR0++, AR1      ; AR1 := adress of P4 storage space
         STI      R0, *+AR1(IR0)
         .endif

         .if      ($symlen(P5)>0) ; Opslaan van P5 op adres:
         LDF      P5, R0           ;  *+(*+mattab(4))(Teller)
         TOIEEE   R0, R0
         LDI      *AR0++, AR1      ; AR1 := adress of P5 storage space
```

```
        STI     R0, *+AR1(IR0)
        .endif

        .if     ($symlen(P6)>0) ; Store P6 on adress:
        LDF     P6, R0          ;  *+(*+mattab(5))(Teller)
        TOIEEE  R0, R0
        LDI     *AR0++, AR1     ; AR1 := adress of P6 storage space
        STI     R0, *+AR1(IR0)
        .endif

        .if     ($symlen(P7)>0) ; Store P7 on adress:
        LDF     P7, R0          ;  *+(*+mattab(6))(Teller)
        TOIEEE  R0, R0
        LDI     *AR0++, AR1     ; AR1 := adress of P7 storage space
        STI     R0, *+AR1(IR0)
        .endif

        .if     ($symlen(P8)>0) ; Store P8 on adress:
        LDF     P8, R0          ; *+(*+mattab(7))(Teller)
        TOIEEE  R0, R0
        LDI     *AR0++, AR1     ; AR1 := adres of P8 storage space
        STI     R0, *+AR1(IR0)
        .endif

        LDI     Teller, R0
        ADDI    1, R0
naarmv? moveI   R0, Teller
naaren? .endm

; ###########################################################################
;  initialisation table nessesery for the TOIEEE macro
;  (page 11-45 user's guide)
; ---------------------------------------------------------------------------
        .data
ieeetab .word   0FF800000h
        .word   0FF000000h
        .word   07F000000h
        .word   080000000h
        .word   081000000h
aieeetab .word  ieeetab
        .global aieeetab
; ---------------------------------------------------------------------------
```

```
; ########################################################################
; #   macro: TOIEEE (FAST Version page 11-46, user's guide)
; #
; #   function: conversion between the 320c30 format and the IEEE
; #             floating point numbers. The number to be converted
; #             is in the upper 32 bits of R0. The result will be
; #             in the lower 32 bits of R0
; #
; #   uses:     R0, AR1
; #   result:   R0
; #   cycles:   14 (worst case)
; ------------------------------------------------------------------------

TOIEEE   .macro   num_in,num_out
         LDI      @aieeetab,AR1    ; AR1 points to table (page 11-45 user's guide)
         LDF      num_in,R0        ; determine the sign of the number
         LDFZ     *+AR1(4),R0      ; If zero, load appropriate number
         BND      NEG?             ; Branch to NEG if negative (delayed)
         ABSF     R0               ; Take the absulute value of the number
         LSH      1,R0             ; Eliminate the sign bit in R0
         PUSHF    R0               ; place R0 onto the stack (float)
         POP      R0               ; Place number in lower 32 bits of R0 (int)
         ADDI     *+AR1(2),R0      ; Add exponent bias (127)
         LSH      -1,R0            ; Add the positive sign
         BR       RES?
NEG?     POP      R0                ; Place number in lower 32 bits of R0 (INT)
         ADDI     *+AR1(2),R0      ; Add exponent bias (127)
         LSH      -1,R0            ; Make space for the sign
         ADDI     *+AR1(3),R0      ; Add the negative sign
RES?
         LDI      R0,num_out       ; place result in num_out
         .endm
;------------------------------------------------------------------------
```

APPENDIX G: MATLAB M-FILES

For the OE-method in Matlab the following m-files

- Pre_proc: reconstruction of stator current and stator flux in rotor coordinates.
- Par_est: estimation of the machine parameters using OE-method.
- Specint: Integration in frequency domain.

G1: PRE_PROC.M:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  PRE_PROC.m
%  ----------
%  This M-file pre-processes the measured data-set.
%  Measured voltage and current in the two-phase representation
%  Are used for the reconstruction of the stator flux. The recon-
%  structed stator flux and stator current are transformed to
%  rotor coordinates using the measured rotor position rho_s.
%
%  The reconstructed Flux and current will then be applied for
%  OE-identification method "PAR_EST.m".
%
%  INPUT:  u_s_s1_f,u_s_s2_f, i_s_s1_f, i_s_s2_f and rho_s
%  OUTPUT: IsR (stator current in rotor coordinats)
%          FsR (stator flux in rotor coordinats)
%
%   S.M. Schilperoort 8-1994
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dT=1/5000;                          % deltat= sample time;
UsS=(u_s_s1_f-j*u_s_s2_f);          % Define stator voltage and current
IsS=(i_s_s1_f+j*i_s_s2_f);          % as complex values (measured voltage has
Motpos=rho_s;                       % rotation direction opposit to current !!)
x=max(size(UsS));                   % Length of data set
t=0:1:x-1;
t=t*dT;                             % total time

%%%%%%%%%%%%%%%%%%%%%% SIGNAL INFORMATION %%%%%%%%%%%%%%%%%%%%%%%%%%

clc;
numsamp=len(UsS);                    % Number of samples
dT=t(2)-t(1);                        % dT = sample time
fs=1/dT;                             % fs = sample frequency
fnyq=fs/2;                           % fnyq= Nyquist frequency
maxtime=max(t);                      % total record time
disp(['Record time      = ',num2str(maxtime),' [s]']),
disp(['Recorded samples = ',int2str(numsamp),' [#]']),
disp(['Record Sample frequentie = ',int2str(fs),' [Hz]']),
disp([' ']);
disp(['Machine information']);
p=input('Number of pool pairs =...');

%%%%%%%%%%%%%%%%%%%% OPERATION CONDITIONS %%%%%%%%%%%%%%%%%%%%%%%%%

%--- Show rotor position ---
```

```
newplot(211)
plot(t,Motpos);
title('Rotor position')
xlabel('time [s]');
ylabel('rho_s [deg]');

%--- Show mechanical rotor speed ---

mp=exp(j*p*Motpos);              %
mp=anglecon(mp);                 % Make continuous angle
mv=diff([mp(1);mp])/(dT*2*pi); % Differentiat rotor postition
mv(1)=0;                         % start values mech.velocity (mv)
mv(2)=0;                         %
fbut=700/fnyq;                   % filter at 700 Hz
[B,A]=butter(4,fbut);            % 4th order butterworth filter
mv=filter(B,A,mv);               % filter motor velocity
mv=mv*60/p;                      % Speed in [rpm]
x=5000:1:len(t);

plot(t(x),mv(x));                % plot mech. velocity
xlabel('time [s]');
ylabel('omega_m [rpm]');

shaftspeed=mean(mv);             % Calculate mean mech.velocity
pause

%% Remove part of data-set if it has no correct start-value %%%%%

a=input('take data from ... sec ');     % Shorting the data set from t=a
b=a/dT+1;                               % not using data upto t=a
c=len(t);                               % c = max time
UsS=UsS(b:c);                           % correction data length
IsS=IsS(b:c);                           % data from t=b to t=c
Motpos=Motpos(b:c);                     %
t=[0:1:len(UsS)-1]*dT;                  % new max time

%%%%%%%%%%%%%%%%%%%%%%%% CALCULATE  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%     Fundamental frequency component
%     Number of stator periods in the data set
%     Rotor speed
%     Slip frequency

clc;
disp('determine fundamental frequency component of te stator voltage')
disp('Filter UsS, low-pass at 100[Hz]...');  % filtering to determine
                                             % fundamental frequency
mt=max(t);                       % Total time
F1=1/mt;                         %
Flp=100;                         % 100 Hz cut-off frequency
Nlp=Flp/fnyq;                    % Normalized filter frequency
[B,A]=butter(4,Nlp);             % 4th order lowpass filter
UsSf=filter(B,A,UsS);            % filtering stator voltage
AV=anglecon(UsSf);               % continue angle of UsS
n=len(AV);                       % new number of samples
W=[0.5*n:n];                     % Take last n/2 samples
DAV=detrend(AV(W));              % Determine trend in AV.
sAV=AV(W)-DAV;                   % Detrend correction.
dPdT=(sAV(2)-sAV(1))/dT;         % determine sloop of the signal
Ffund=dPdT/(2*pi);               %
Nper=floor(max(t)*abs(Ffund));   % Number of full stator periods.
c=floor(Nper/(dT*abs(Ffund)));   % rearange data, start data at t=0
```

```
if rem(c,2)==1,                 %
        c=c-1;
end;

%--- clear variables ---
clear t a fre mag q w Flp Nlp B A Av AV W DAV sAv dPdT UsSf
clear Set finput pha sAV test1 F1 mt n numsamp filen

%------------------------------------------------------------------------

t=[0:1:(c-1)];                  % Data-set has Nper full periods in
t=t*dT;                         % stator frequency.
t=row(t);                       % Make row-vector
Motpos=Motpos(1:c);             % Rotor position up to t=c
IsS=IsS(1:c);                   %
UsS=UsS(1:c);                   %
disp(['Fundamental frequency = ',num2str(Ffund),' [Hz]']),
disp(['Number of stator periods = ',num2str(Nper),' ']),
disp([' ']);
disp(['Rotor speed = ',num2str(shaftspeed),'[rpm]']),
disp(['Slip frequency =',num2str(abs(Ffund)-abs(shaftspeed*p/60)),'[Hz]']),
disp([' ']);
clear a b c ;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Flux reconstruction
% R.J. Gorter maart 94
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('Stator flux reconstruction ');

k=menu('Method ','Freq. domain','Detrend ');

Rs=input('two phase stator resistance [ohm]? ')

%%%%%%%%%%%%% INTEGRATION IN FREQUENCY DOMAIN %%%%%%%%%%%

if k==1
    FsS=specint(t,(UsS-Rs*IsS),0.8*Ffund);  % SPECINT= spectrum integration

%%%%%%%%%%%%% INTEGRATION IN TIME DOMAIN %%%%%%%%%%%%%%%%
%       Using a 4th order Addams Moulton routine

elseif k==2
    bd=[1 -1];
    ad=dT/24*[9 19 -5 1];               % Adams-Moulton integration
    FsS=filter(ad,bd,UsS-Rs*IsS).';     %  ,,      ,,         ,,
    RPnts=[1/6:1/6:fix(Nper)];          % Define number new time steps
    IPnts=len(t)/Nper*RPnts;            %
    n=len(t);                           %
    NP1=fix(Nper);                      %
    MPnts=zeros(NP1,6);                 %
    if NP1 >1,                          % Determine the drift per period
      for k=[1:6],                      %
      W=[k:6:6*NP1];
      MPnts(:,k)=FsS(IPnts(W))-mean(FsS(IPnts(W)));
      end;                              %
      Drift=mean(MPnts')';              %
      Tper=1/Ffund;                     %
      VDrift=mean(diff(Drift))/Tper;    %
    elseif NP1==1,                      %
      Drift=FsS(n/Nper+1)-FsS(1);       %
      VDrift=Drift/Tper;                %
```

```
    end;                                %
    FlxDrift=VDrift*row(t);             % Drift correction per period
    FsS=FsS-FlxDrift*sign(Ffund); clear FlxDrift;
    FsS=FsS-mean(FsS(1:NP1/Nper*n));    %

    clear RPnts IPnts MPnts Drift VDrift FlxDrift FlxS Tper n;

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Filtering and remove DC component
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    [bd ad]=c2dm([1 0],[1 10],dT,'tustin');
    FsS=filter(bd,ad,FsS);
    IsS=filter(bd,ad,IsS);
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Rotating stator signals to Rotor coordinats
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

IsR=exp(-j*p*Motpos).*IsS;
FsR=exp(-j*p*Motpos).*FsS.';

%--- Plot reconstructed flux and current ----
newplot(211);
plot(t,IsR);
title('Current and flux in rotor coordinates')
xlabel('time [s]');
ylabel('IsR [A]');
plot(t,FsR);
xlabel('time [s]');
ylabel('FsR [Vs]');

end;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

G2: PAR_EST.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  PAR_EST.m
%
%  OE-method; Identification of the asynchronous machine parameters
%
%  INPUT: IsR and FsR (stator current and flux in rotor coordinats)
%
%  OUTPUT: dependent on chosen model order
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x=1:10:len(t);
clc;
newplot(211)
plot(t(x),real(IsR(x)))           % show stator current in rotor coordinats
subplot(212)                      %
plot(t(x),real(FsR(x)))           % show stator flux in rotor coordinats
Z=[row(real(FsR)) row(real(IsR))];
pause;

%------------- Per-Filtering the desired frequency region ------------------

fnyq=.5/(t(2)-t(1));              % nyquist freq = 1/2*sample freq.

FMAX=input(['Pre-filter (L) cutoff [Hz] < fnyq=',num2str(fnyq),'[Hz]...:']);

if FMAX<fnyq,
   [Bf,Af]=butter(4,FMAX/fnyq);
   [In,zpI]=filter(Bf,Af,flipud(Z(:,2)));
   [Uit,zpU]=filter(Bf,Af,flipud(Z(:,1)));
   [In]=filter(Bf,Af,Z(:,2),zpI);
   [Uit]=filter(Bf,Af,Z(:,1),zpU);
   Z=[Uit In];
   clear Fmax In Uit zpI zpU Bf Af w fvss Mvss;
end;
clc;

%%%%%%%%%%%%%%%%%%%%%%%% OE model structure %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TH=oe(Z,[Ordnb Ordna 0]);

%%%%%%%%%%%%%%%%%%%%%%%% Estimation results %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
present(TH);
pause;

%%%%%%%%%%%%%%%%%%%%%%%% SHOW OUTPUT ERROR %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Z=[FsR.' IsR.'];                  % data-set vector
FsRhat=predict(Z,TH);             % estimated stator flux

newplot(211)
axis([0,2.5,-1.5,1.5]);
preerror=(Z(:,1)-FsRhat);
plot(t(x),[FsRhat(x) Z(x,1)]);
gridxy(0,-3);
gridxy(0,0);
title('stator flux in rotor coordinat R1')
xlabel(' time [s]')
ylabel(' flux [Vs]')
```

```
subplot(212)
axis([0,2.5,-.05,.05])
plot(t(x),preerror(x));
title('prediction error')
xlabel(' time [s]')
gridxy
pause

%-------------- SHOW BODE-DIAGRAM OF ESTIMATED TRANSFER FUNCTION ------------

TH(1,2)=dT;                                % Denote sample time
[a,Bd,c,d,Fd]=th2poly(TH);                 % discrete polynomials

fre=logspace(-5,pi,100);                   % Define freq. range
fre=fre/pi*fnyq;

[mag_d,phase_d]=dbode(Bd,Fd,dT,fre*2*pi); % Discrete bode plot

%-----------------------------------------------------------------------------
%      Chosen model order = 1 , calculate
%      Lsig, Lm, and Rr for continious time
%-----------------------------------------------------------------------------

if Ordna==1

   %***** INVERSE Z-transform => see Section 3.2 ****
   f0=Deni(2);                     % B=Numi=[b1 b0]
   b0=Numi(2);                     % F=Deni=[f1 f0]
   b1=Numi(1);
   disp('Lsig Lm Rk tr')
   Lsig=b1                         % inverse z-transform
   Lm=(b0-b1*f0)/(1+f0)            % calculate single-cage machine
   Rk=-log(-f0)*Lm/T1              % parameters
   tk=-log(-f0)/T1                 %

   B0=(Lsig*Lm)/Rk;                % calculate the continuous time
   B1=Lsig+Lm;                     % parameters
   F0=Lm/Rk;
   Bc=[B0 B1];                     % continuous time polynomial
   Fc=[F0 1];                      % parameters

%-----------------------------------------------------------------------------
%      Chosen model order=2: calculate the continuous
%      time physical double-cage parameters
%-----------------------------------------------------------------------------

elseif Ordna==2
   %******** TUSTIN TRANSFORMATION ***************
   b0=Bd(1);
   b1=Bd(2);                       % Parameters of the estimated
   b2=Bd(3);                       % discrete transfer function
   f1=Fd(2);
   f2=Fd(3);

   B0=((b0-b1+b2)*dT^2)/4;
   B1=(b0-b2)*dT;
   B2=(b0+b1+b2);
   F0=((1-f1+f2)*dT^2)/4;
   F1=(1-f2)*dT;
   F2=(1+f1+f2);
   Bc=[B0 B1 B2];                  % Parameters of the estimated
   Fc=[F0 F1 F2];                  % discrete transfer function
```

```
   %***** Calculation of the pysical continous parameters ****
   Bc=Bc/Fc(3);
   Fc=Fc/Fc(3);
   w2=Bc(1);              % Bc and Fc are the estimated
   w1=Bc(2);              % polynomials in continous
   w0=Bc(3);              % time. Bc=[b2 b1 b0]
   v2=Fc(1);              % Fc=[f2 f1 f0]
   v1=Fc(2);

   Lsig=w2/v2             % machine parameters:
   Lm=(w0-Lsig)           % total leakage inductance Lsig
   t2=(w1-Lsig*v1)/Lm;    % main inductance Lm
   t1=v2/t2;              % cage resistance Rk1
   Rk1=Lm/t1              % cage resistance Rk2
   Rk2=Lm/(v1-t1-t2)      % cage leakage inductance Lsig2
   Lsig2=Rk2*t2
end;

   [mag_d,phase_d]=dbode(Bd,Fd,dT,2*pi*fre); % Discrete bode plot
   [mag_c,phase_c]=bode(Bc,Fc,fre*2*pi);     % Continious bode plot

   semilogx(fre,20*log10(mag_c),'w',fre,20*log10(mag_d),'g');
   xlabel('frequency [Hz]');
   ylabel('Gain [dB]');
   semilogx(fre,phase_c,'w',fre,phase_d,'g');
   xlabel('frequency [Hz]');
   ylabel('Phase [deg]');
   pause

end;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

G3: SPECINT.M

```
%-------------------------------------------------------------------------
function [y]=specint(t,y,fband)
%-------------------------------------------------------------------------
% function [y]=specint(t,y,fband),
% Integration in freq domain;
% DC offset is removed, phase is correct;
%
% fband reduces all componetns between -fband fband
% R.J.A. Gorter 14-2-94
%-------------------------------------------------------------------------
filenaam='specint';
n=max(size(t));
if rem(n,2)==1,
   last=n-1;
else,
   last=n;
end;
if nargin==1,
    I=t(1:last);
    t=1/last*[0:last-1]';
else,
    t=row(t(1:last));
    I=row((y(1:last)));
    j=sqrt(-1);
```

```
end;
t=t-t(1);
S=fft(I)./last;

Magp=2*abs(S(2:last/2));          % DC component removed.
Anglep=angle(S(2:last/2));        % DC component removed.
Magn=2*abs(S(last/2+1:last));
Anglen=angle(S(last/2+1:last));

%-------------------------------------------------------------------------
% amplitude integration
% pos. freq.
%-------------------------------------------------------------------------
dT=t(2)-t(1);
dF=1/(dT*last);
freq=2*pi*dF*[0:last/2-1]';
freqdc=freq(2:len(freq));         % DC component removed.
Magintp=Magp./freqdc;
Magintp=[ 0 Magintp']';           % DC component = 0;
Anglep=Anglep-pi/2;
Anglep=[ 0 Anglep']';             %DC component = 0;
freqinv=2*pi*dF*row([(last/2):-1:1]);
Magintn=Magn./freqinv;

Anglen=Anglen+pi/2;
Ang=[Anglep' Anglen']';
Mag=[Magintp' Magintn']';
freq=[0 freqdc' freqinv'];

if nargin==3,
fband=abs(fband);
Nskip=fix((fband)/dF)-1;
Mag(1:Nskip,1)=zeros(Nskip,1);
x=len(Mag);
xmin=x-Nskip+1;
Mag(xmin:x,1)=zeros(Nskip,1);
end;

I=0.5*Mag.*(cos(Ang)+i*sin(Ang));

y=ifft(I)*last;                   % inverse fourier transformation
%-------------------------------------------------------------------------
```