Quasi-Stationary Modeling and Simulation of Electrical Circuits using Complex Phasors

Anton Haumer Christian Kral Johannes V. Gragger Hansjörg Kapeller
arsenal research
Giefinggasse 2, 1210 Vienna, Austria
anton.haumer@arsenal.ac.at

Abstract

This paper presents how complex phasors are used for quasi-stationary analysis of electrical circuits, i.e. with sinusoidal excitation neglecting dynamic transients. The theoretical background of complex phasors is elaborated and a Modelica implementation – the AC Library – is presented. Additional examples demonstrate the possibilities of the application of complex phasors.

Keywords: electrical circuit, sinusoidal excitation, quasi-stationary analysis, complex phasors

1 Introduction

In the simulation of physical systems described by a system of algebraic and ordinary differential equations we distinguish different types of simulation analysis:

- The *transient analysis* is the most general analysis, showing both the dynamic transients as well as steady-state solutions (if steady-state is reached).
- A *stationary analysis* (sometimes also called DC analysis) eliminates the derivatives with respect to time, determining steady-state solutions.
- The so-called *small signal AC analysis* linearizes a non-linear model in a certain point of operation (which is found by a stationary analysis), only applying excitations with small amplitudes.

Mainly in the field of electrical engineering – due to the nature of electrical power plants that provide nearly perfectly sinusoidal voltages with fixed frequency and amplitude – one more type of analysis is of great importance:

- *Quasi-stationary analysis* applies sinusoidal excitations with known frequency, amplitude and phase shift. In a circuit with isolated sub-circuits, each sub circuit may be operated at different frequencies, however. Each frequency with respect to a sub-circuit is known due to the respective excitation. Fast dynamic transients are not considered. In a quasi-stationary analysis the unknown voltages and currents, with respect to their phase shift and amplitude, have to be determined. Regarding the consideration of exactly one frequency for each sub-circuit it has to be assumed that the only linear circuits are investigated.

This paper will demonstrate how complex phasors simplify the quasi-stationary analysis, and how complex phasors could be modeled using Modelica. Considering some limitations in the current Modelica version will lead to suggestions for improvement.

2 Complex Phasors

2.1 Representation of Sinusoidal Voltages and Currents

Any sinusoidal oscillation can be expressed by computing the real part of a complex time-dependent phasor according to Fig. 1:

$$a(t) = \hat{A}\cos(\omega t + \varphi) = \sqrt{2}A\,\mathrm{Re}\left(e^{j\omega t}e^{j\varphi}\right) \tag{1}$$

$$\underline{A} = A \cdot e^{j\varphi} \Rightarrow a(t) = \sqrt{2}\,\mathrm{Re}\left(\underline{A} \cdot e^{j\omega t}\right) \tag{2}$$

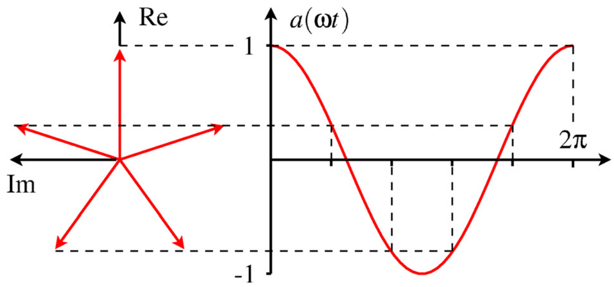

Fig. 1 The real part of a rotating phasor equals a sinusoidal oscillation; depicted phasor with $\varphi = 0$; left: complex phasor $\sqrt{2}\underline{A}e^{j\omega t}$; right: time domain signal $a(\omega t)$

The magnitude A of the complex phasor $\underline{A}$ is the root mean square (RMS) value of the cosine wave. The phase shift φ is the phase shift of the cosine with respect to its maximum at $t = 0$. Time dependence is considered by the phasor $e^{j\omega t}$ and $\sqrt{2}$ is the ratio between the amplitude and the RMS value of the sinus waveform.

This background of complex phasors can also be applied to sinusoidal voltages and currents using complex voltage and current phasors:

$$v(t) = \hat{V}\cos(\omega t + \varphi_v) = \sqrt{2}\,\mathrm{Re}\left(\underline{V} \cdot e^{j\omega t}\right) \quad (3)$$

$$i(t) = \hat{I}\cos(\omega t + \varphi_I) = \sqrt{2}\,\mathrm{Re}\left(\underline{I} \cdot e^{j\omega t}\right) \quad (4)$$

Assuming sinusoidal excitation of an electric circuit, all voltages and currents are of sinusoidal waveform with the same angular frequency $\omega = 2\pi f$. Therefore the complex voltage phasor

$$\underline{V} = V \cdot e^{j\varphi_v} \quad (5)$$

and the complex current phasor

$$\underline{I} = I \cdot e^{j\varphi_I} \quad (6)$$

are sufficient to describe quasi-stationary voltages and currents.

The derivative of a complex phasor $\underline{A}$ with respect to time leads to:

$$\frac{da(t)}{dt} = \sqrt{2}\,\mathrm{Re}\left(\underline{A} \cdot j\omega e^{j\omega t}\right) \quad (7)$$

The time derivative of a sinusoidal waveform is thus considered in the complex domain by multiplying the original phasor with $j\omega$. This relationship also implies the result of the integration with respect to the time domain. Since the constant of integration is zero for quasi-stationary analysis, the complex representation of a time domain integration is determined by the division of the original phasor by $j\omega$.

2.2 Modeling a Linear Resistor

A linear resistor can be described by the algebraic equation:

$$v = R \cdot i \quad (8)$$

Using complex phasors of voltage and current, we can replace the algebraic equation by a complex algebraic equation:

$$\underline{V} = R \cdot \underline{I} \quad (9)$$

2.3 Modeling a Linear Conductor

A linear conductor can be described by the algebraic equation:

$$i = G \cdot v \quad (10)$$

Using complex phasors of voltage and current, we can replace the algebraic equation by a complex algebraic equation:

$$\underline{I} = G \cdot \underline{V} \quad (11)$$

2.4 Modeling a Linear Inductor

A linear inductor can be described by the differential equation:

$$v = L\frac{di}{dt} \quad (12)$$

Exploiting the sinusoidal waveform of the current (4), we can replace the differential equation by a complex algebraic equation:

$$\underline{V} = j\omega L \cdot \underline{I} = \underline{X}_L \cdot \underline{I} \quad (13)$$

We find a complex version of the equation describing a resistor, using the complex reactance $\underline{X}_L = j\omega L$.

2.5 Modeling a Linear Capacitor

A linear capacitor can be described by the differential equation:

$$i = C\frac{dv}{dt} \quad (14)$$

Exploiting the sinusoidal waveform of the voltage (3), we can replace the differential equation by a complex algebraic equation:

$$\underline{I} = j\omega C \cdot \underline{V} = \underline{Y}_C \cdot \underline{V} \quad (15)$$

We find a complex version of the equation describing a conductor, using the complex admittance $\underline{Y}_C = j\omega C$.

2.6 Kirchhoff's Laws

For complex voltages and currents, respectively, Kirchhoff's Laws can be applied equivalently:

$$\sum \underline{I}_i = 0 \quad (16)$$

The sum of all complex current phasors flowing to a node is zero.

$$\sum \underline{V}_i = 0 \quad (17)$$

The sum of all complex voltage phasors in a closed loop is zero; this also implies that directly connected nodes have the same complex potential. Both laws are inherently considered in Modelica connections.

2.7 Power

Multiplying a time dependent voltage (3) and the corresponding current (4), we obtain the instantaneous electrical power:

$$p(t) = v(t) \cdot i(t) \tag{18}$$

Substituting (3) and (4) in the electric power equation we obtain:

$$\begin{aligned} &p(t) = \\ &\sqrt{2}V\cos(\omega t + \varphi_V) \cdot \sqrt{2}I\cos(\omega t + \varphi_I) = \\ &V \cdot I \cdot [\cos(\varphi_V - \varphi_I) + \cos(2\omega t + \varphi_V + \varphi_I)] \end{aligned} \tag{19}$$

The instantaneous power oscillates with double the frequency of voltage and current, respectively. The average value of instantaneous power is dependent on the phase shift between voltage and current; this term is the active power:

$$P = V \cdot I \cdot \cos(\varphi_V - \varphi_I) = S \cdot \cos(\varphi_V - \varphi_I) \tag{20}$$

Apparent power S is defined as the product of the RMS values of the voltage and the current:

$$S = V \cdot I \tag{21}$$

Reactive power is defined as quadratic complement:

$$Q = \sqrt{S^2 - P^2} = S \cdot \sin(\varphi_V - \varphi_I) \tag{22}$$

Using complex phasors, we obtain:

$$\underline{S} = \underline{V} \cdot \underline{I} = P + jQ \tag{23}$$

In this equation, $\underline{S}$ is the complex apparent power; the amplitude of this complex quantity is the apparent power (21).

3 Design of an AC Modelica Library

3.1 Implementation of Complex Arithmetics

Unfortunately complex numbers are not an intrinsic data type in the Modelica language. As a workaround, a record Complex containing both the real and the imaginary part of the complex number can be defined:

```
record Complex
  Real re "Real part";
  Real im "Imaginary part";
end Complex;
```

In some cases, the polar representation of a complex phasor, consisting of length and phase angle, is advantageous, however:

$$\underline{A} = A_{Re} + j \cdot A_{Im} = \hat{A} \cdot e^{j\varphi} \tag{24}$$

```
record Polar
  Real len "Length of the phasor";
  Modelica.SIunits.Angle phi "Phase angle";
end Polar;
```

Of course we have to provide functions for complex arithmetic + - * /, like

```
function '+' "Complex add"
  input Complex c1;
  input Complex c2;
  output Complex c3 "= c1 + c2";
algorithm
  c3 := Complex(c1.re + c2.re,
                c1.im + c2.im);
end '+';
```

as well as complex functions like

- `abs` length of the phasor
- `arg` phase angle
- `conj` conjugate complex
- `sqrt` square root
- `exp` natural exponentiation
- `log` natural logarithm
- `sin` sine
- `cos` cosine

which can be implemented according to a mathematical textbook.

It is not very elegant to use these functions:

```
v = Complex.'*'(Complex(0, w*L), i);
```

Therefore an intrinsic implementation (or at least operator overloading) would allow reading, type and understanding code easier.

Additionally, we need conversion functions between rectangular and polar representation:

```
function fromPolar
  input Polar polar;
  output Complex result;
algorithm
  result.re :=polar.len*cos(polar.phi);
  result.im :=polar.len*sin(polar.phi);
end fromPolar;
function toPolar
  input Complex c;
  output Polar polar;
algorithm
  polar.len := Complex.'abs'(c);
  polar.phi := Complex.arg(c);
end toPolar;
```

The tricky part of the conversion from rectangular to polar representation is obtaining an angle that may be smoothly differentiated to obtain the angular velocity of the corresponding phasor. With the presented implementation the wrapping of the phase angle at 2π cannot be avoided. Instead, a continuous growth of the phase angle for non-zero frequency is desired.

A rather difficult exception of a smooth angle is the following example: Imagine a phasor with constant angle, but length varying with time. The length shrinks within a certain time to zero, growing again in the opposite direction afterwards. This would lead to a discontinuity by π when the phasors crosses the origin.

Additionally, we have to define complex phasors with physical units, like:

```
record ComplexVoltage = Complex (
    redeclare Modelica.SIunits.Voltage re,
    redeclare Modelica.SIunits.Voltage im);
record ComplexCurrent = Complex (
    redeclare Modelica.SIunits.Current re,
    redeclare Modelica.SIunits.Current im);
```

to take advantage of a tool's type checking capabilities. Furthermore we have to define the polar representations, too:

```
record PolarVoltage = Polar (
  redeclare Modelica.SIunits.Voltage len);
record PolarCurrent = Polar (
  redeclare Modelica.SIunits.Current len);
```

3.2 Propagation of the Common Frequency

Since different sub-circuits of an electrical circuit could have different frequencies – e.g. stator and rotor of an asynchronous induction motor– it would be advantageous to provide the local frequency of a component via the connector. Introducing an additional variable (reference angle, frequency or angular velocity) in the connector leads to over-determined connection equations. Fortunately, Modelica [4] provides methods to deal with this problem. This connector variable has to be defined as a type or record with an additional function definition:

```
record Reference
  Modelica.SIunits.Angle phi;
  function equalityConstraint
    input Reference ref1;
    input Reference ref2;
    output Real residue[0];
  algorithm
    residue := ...;
  end equalityConstraint;
end Reference;
```

Additionally, the following functions are used to allow a tool to break algebraic loops:

- `Connect`
  defines a breakable branch
- `Connections.branch`
  defines a non-breakable branch
- `Connections.root`
  defines a root node in a virtual connection graph
- `Connections.potentialRoot`
  defines a potential root node in a virtual connection graph

3.3 Single Phase Components

The connector definition

```
connector Pin
  Types.ComplexVoltage v;
  flow Types.ComplexCurrent i;
  Types.Reference ref;
end Pin;
```

not only contains complex potential and complex current, but also the record providing the local frequency respectively phase angle of the reference frame as explained in 3.2.

Additionally, basic components as ground, resistor, conductor, capacitor and inductor are defined. Furthermore, we need sensors and voltage sources as well as current sources. Fig. 2 gives an overview of the implemented components.

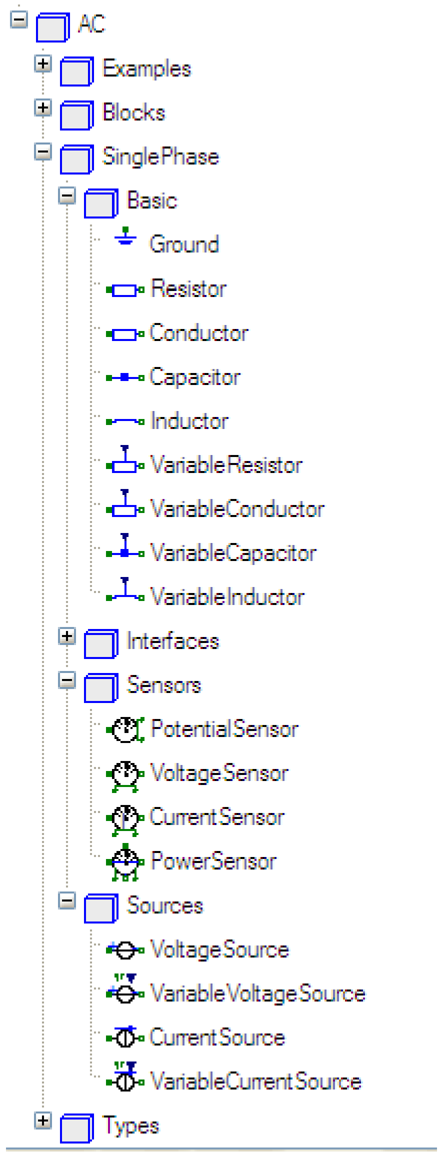

Fig. 2 Structure of the AC library

As an example, the implementation of the inductor as well as the partial models that inductor extends from are shown:

```
partial model TwoNode
  Types.ComplexVoltage v =
    Complex.'-'(p.v, n.v);
  Types.ComplexCurrent i = p.i;
  Modelica.SIunits.AngularVelocity w =
    der(p.ref.phi);
  AC.SinglePhase.Interfaces.PositivePin p;
  AC.SinglePhase.Interfaces.NegativePin n;
equation
  Connections.branch(p.ref, n.ref);
  p.ref.phi = n.ref.phi;
end TwoNode;
```

`TwoNode` defines the complex voltage drop along the component as well as the angular velocity by differentiating the reference phase angle.

```
partial model OnePort
  extends TwoNode;
equation
  Complex.'+'(p.i, n.i) = Complex.'0'();
end OnePort;
```

`OnePort` additionally defines that the sum of currents flowing into the component is zero.

```
model Inductor
  extends Interfaces.OnePort;
  parameter
    Modelica.SIunits.Inductance L=1;
equation
  v = Complex.'*'(Complex(0, w*L), i);
end Inductor;
```

Using these partial models `Inductor` is a simple implementation of (13).

3.4 Auxiliary Blocks

Additionally to the basic components, blocks with complex inputs / outputs are needed. Therefore a complex signal is defined, as well as a polar signal:

```
connector ComplexSignal = AC.Types.Complex;
connector PolarSignal = AC.Types.Polar;
```

These connectors are used to define `ComplexInput`, `ComplexOutput`, `PolarInput` and `PolarOutput`. Instances of these output signal connectors are needed for sensors, as well as input signal connectors for variable sources.

Additionally some useful blocks are defined:

- `ToComplex` generates a complex phasor from real inputs, either real and imaginary part or amplitude and phase angle.
- `FromComplex` generates real outputs – real and imaginary part as well as amplitude and phase angle – either from a complex input or a polar input.
- `ToPolar` takes a complex input and generates a polar representation of the phasor as the output.
- `FromPolar` takes a polar representation of a complex phasor on the input and generates a complex phasor as the output.
- `FromPolar` calculates the complex sum of an array of complex input phasors.

4 Simulation Examples

For calculating quasi-stationary characteristic curves of an electrical circuit varying a parameter the usage of the AC library is advantageous. This will be demonstrated on four examples:

- Current of a series resonance circuit, varying the supply frequency
- Voltage of a parallel resonance circuit, varying the supply frequency
- Torque and current of an asynchronous induction machine, varying slip
- Terminal voltage of a synchronous induction machine, varying load impedance (resistive and inductive).

4.1 Series Resonance Circuit

As a first example, we model a series resonance circuit (Fig. 3).

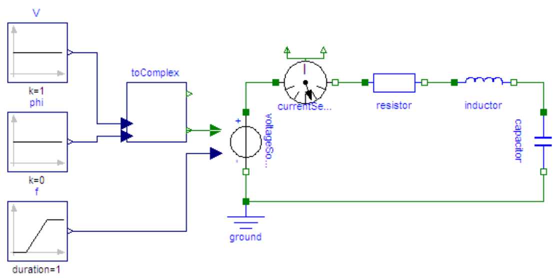

Fig. 3 Model of a series resonance circuit

We apply sinusoidal voltage with constant amplitude and phase to a series connection of a resistor, an inductor and a capacitor. Frequency varies according to a ramp.

Analytically the resonant frequency of this simple experiment can be determined:

$$\omega_{res} = \frac{1}{\sqrt{LC}} \qquad (25)$$

With $L = \frac{1}{2\pi}$H and $C = \frac{1}{2\pi}$F we derive a resonance frequency at f = 1 Hz. From the amplitude (Fig. 4) as well as the phase shift (Fig. 5) of the current, the resonance frequency is evident.

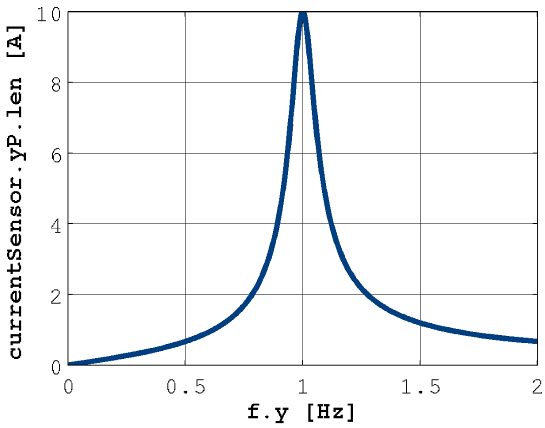

Fig. 4 Amplitude of current versus excitation frequency

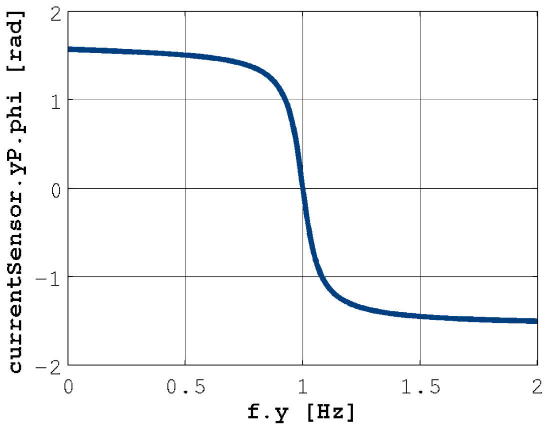

Fig. 5 Phase shift of current versus excitation frequency

4.2 Parallel Resonance Circuit

Furthermore, we investigate a parallel resonance circuit (Fig. 6).

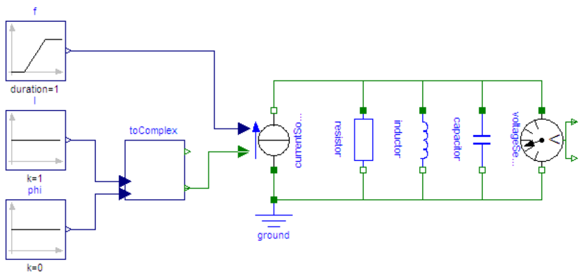

Fig. 6 Model of a parallel resonance circuit

We inject a sinusoidal current with constant amplitude and phase to a parallel connection of a resistor, an inductor and a capacitor. Frequency varies according to a ramp. The resonant frequency of the parallel resonant circuit is:

$$\omega_{res} = \frac{1}{\sqrt{LC}} \tag{26}$$

With $L = \frac{1}{2\pi}$ H and $C = \frac{1}{2\pi}$ F we derive a resonance frequency at f = 1 Hz. From the amplitude (Fig. 7) as well as the phase shift (Fig. 8) of the voltage, the resonance frequency is evident.

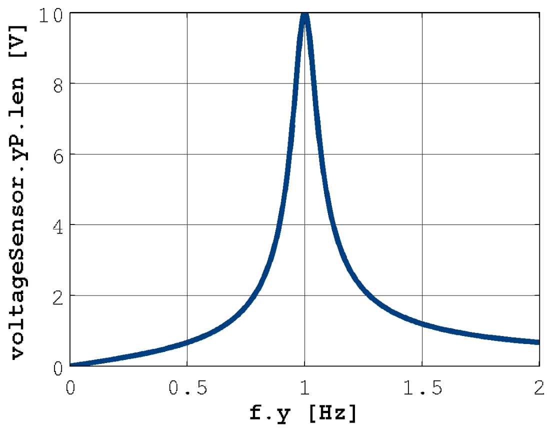

Fig. 7 Amplitude of voltage versus excitation frequency

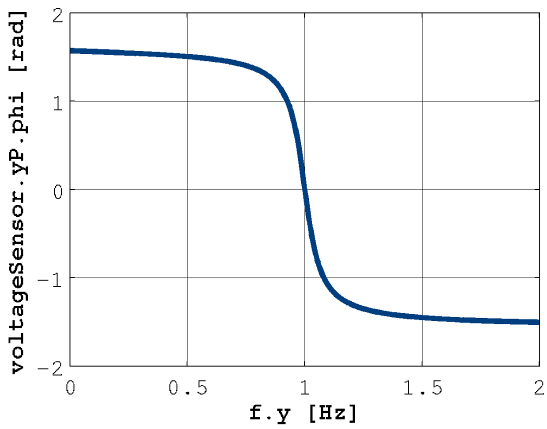

Fig. 8 Phase shift of voltage versus excitation frequency

4.3 Asynchronous Induction Machine

Quasi-stationary operation of a three-phase asynchronous induction machine with squirrel cage (AIMC) may be described by an equivalent circuit as depicted in Fig. 9. This equivalent circuit represents one phase of a symmetrical three phase asynchronous induction machine, however.

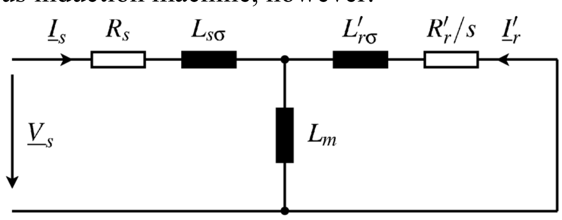

Fig. 9 Single phase equivalent circuit of an AIMC

In this equivalent circuit R_s is the stator resistance, $L_{s\sigma}$ is the stator leakage inductance, and L_m is the main field inductance. In the rotor circuit $L_{r\sigma}'$ is the rotor leakage inductance and R_r' is the rotor resistance. Both these rotor components refer to an equivalent stator winding and are thus indicated by '. An implementation of this equivalent circuit in Modelica is shown in Fig. 10.

For an induction machine slip

$$s = 1 - \frac{\omega}{\omega_s} \qquad (27)$$

is the relative deviation of the mechanical angular velocity ω from the synchronous angular velocity:

$$\omega_s = \frac{2\pi f}{p} \qquad (28)$$

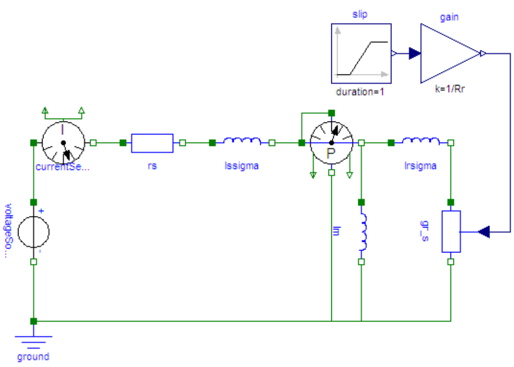

Fig. 10 Model of an AIMC

In the presented example slip is modeled as a ramp from slip 100% (i.e. stand-still) to slip 0% (i.e. no-load). Dividing the rotor resistance by the slip is equivalent to multiplying the rotor conductance by slip. The slip dependent rotor conductance is thus modeled by a variable conductance `gr_s`. Using the conductance avoids division of zero slip at no-load:

$$R'_{r,actual} = \frac{R'_r}{s} \qquad (29)$$

The motor parameters used for this example are the same as those of the dynamic model `Electrical.Machines.BasicMachines.AsynchronousInductionMachines.AIM_SquirrelCage`.

This leads to the quasi-stationary motor characteristics depicted in Fig. 11 and Fig. 12. The horizontal axis of these plots shows the relative (per unit) speed which is equal to (1–slip).

Fig. 12 shows only 1/3 of the total air gap power of the machine since only one phase is modeled. The total airgap torque can thus be determined by:

$$T = \frac{3 \cdot P_{airgap}}{\omega_s} \qquad (30)$$

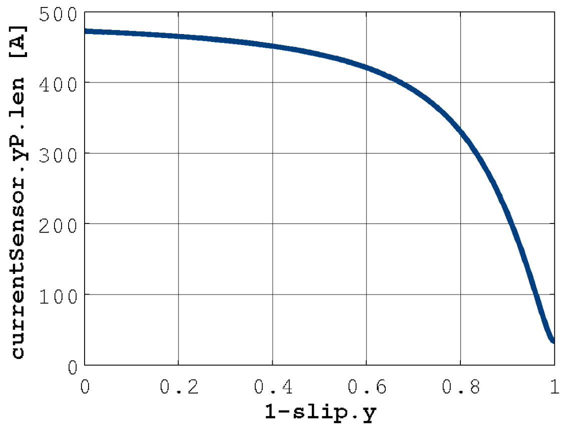

Fig. 11 Stator current versus (1-slip)

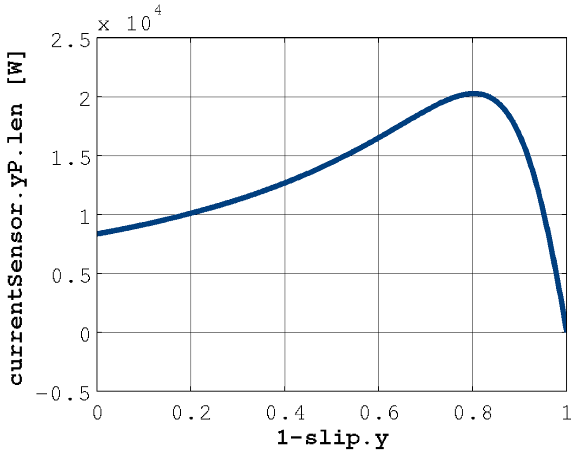

Fig. 12 Airgap power versus (1-slip)

4.4 Synchronous Induction Machine

A synchronous induction machine feeding an isolated system is presented in this example. Two cases are investigated: resistive load (Fig. 13) and inductive load (Fig. 14).

In both cases, constant excitation is assumed. Synchronous induced voltage is modeled by a voltage source with constant complex voltage phasor. `ld` represents the synchronous reactance and `rs` the resistance of one phase. Variable load is prescribed by a ramp with logarithmic scale.

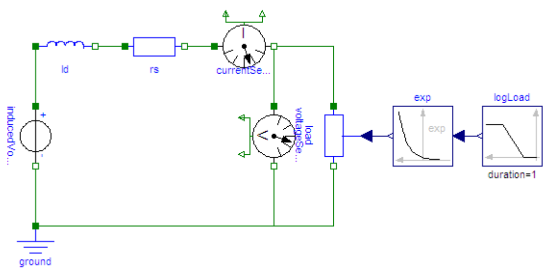

Fig. 13 Synchronous induction machine with R-load

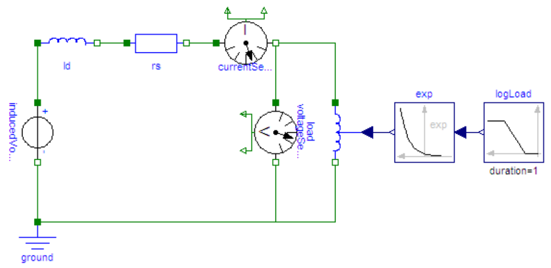

Fig. 14 Synchronous induction machine with L-load

Fig. 15 shows the characteristic voltage versus current from nearly no-load (high resistance and inductance, respectively) to nearly short circuit (low resistance and inductance, respectively).

The machine parameters used for this example are the same as those of the dynamic model `Electrical.Machines.BasicMachines.SynchronousInductionMachines.SM_ElectricalExcited`.

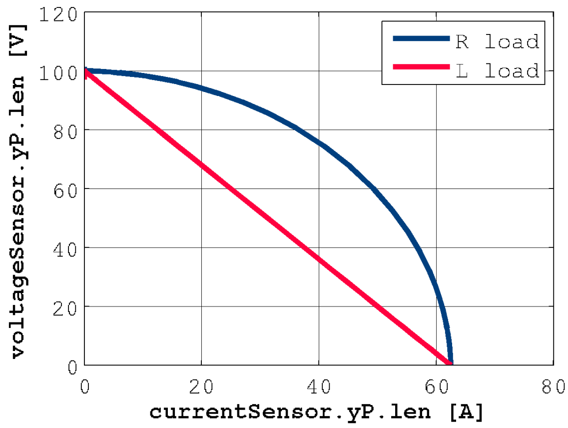

Fig. 15 Voltage versus current for resistive load and inductive load

5 Conclusions and Outlook

The design of a Modelica library for quasi-stationary analysis of electrical single-phase circuits has been presented. The application of complex algebraic equations instead of dynamic differential equations leads to high performance simulations.

With respect to the current Modelica version 3.0, the implementation of complex numbers is possible but not really satisfying. The authors would suggest the introduction of complex numbers as an intrinsic data type. This data type and complex arithmetics would improve the Modelica language, however.

Based on the presented draft of an AC library, the next steps will be extending the components for multi-phase circuits as well as modeling of asynchronous and synchronous induction machines for quasi-stationary analysis. These machine models are planned to be based on space phasors as described in [6]; the transformation of space phasors with respect to different reference frames has to be implemented.

For applications focused on the energy consumption of an electric drive over a longer period of time, the fast electrical transients can be neglected. Using complex quasi-stationary models would lead to faster simulations, however.

References

[1] M. L. Boas, Mathematical Methods in the Physical Sciences. J. Wiley & Sons 1966

[2] T. D. Burton, Introduction to Dynamic Systems Analysis. McGraw Hill 1994

[3] R. C. Dorf, The Electrical Engineering Handbook. VDE 1993

[4] Modelica Specification, version 3.0. http://www.modelica.org/documents/ModelicaSpec30.pdf

[5] P. Fritzson, Principles of Object-Oriented Modeling and Simulation with Modelica 2.1. Piscataway, NJ: IEEE Press, 2004.

[6] C. Kral, A. Haumer, Modelica libraries for dc machines, three phase and polyphase machines. 4th International Modelica Conference 2005, Hamburg, Germany

[7] O. Enge, C. Clauß, P. Schneider, P. Schwarz, M. Vetter, S. Schwunk, Quasi-stationary AC Analysis Using Phasor Description With Modelica. 5th International Modelica Conference 2006, Vienna, Austria