

# Electrical machine analysis using free software

David Meeker, QinetiQ North America, USA

Nicola Bianchi, University of Padova, Italy

Johan Gyselinck, Université Libre de Bruxelles

Ruth V. Sabariego, KU Leuven, Belgium

Luigi Alberti, University of Padova, Italy

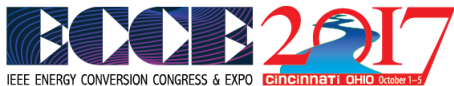
Gianmario Pellegrino, Politecnico di Torino, Italy

Francesco Cupertino, Politecnico di Bari, Italy

July 31, 2017

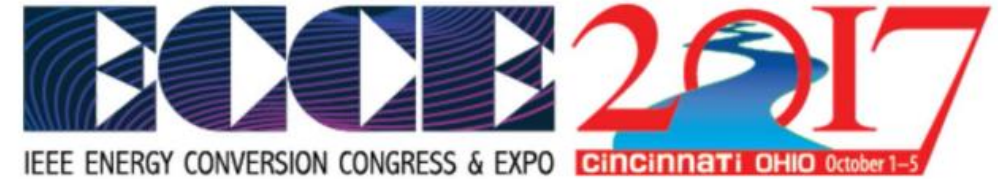
# Electrical machine analysis using free software

- David Meeker, **QinetiQ North America, USA**
- Nicola Bianchi, **University of Padova, Italy**
- Johan Gyselinck, **Université Libre de Bruxelles**
- Ruth V. Sabariego, **KU Leuven, Belgium**
- Luigi Alberti, **University of Padova, Italy**
- Gianmario Pellegrino, **Politecnico di Torino, Italy**
- Francesco Cupertino, **Politecnico di Bari, Italy**



# List of presentations

- **A five minutes Introduction**  
Nicola Bianchi,
- **FEMM Overview**  
David Meeker,
- **PM motor design with FEMM and LUA scripting**  
Nicola Bianchi,
- **Finite-element modelling of electrical machines using ONELAB (Gmsh & GetDP)**  
Johan Gyselinck, Ruth V. Sabariego,
- **KOIL - a tool to design the windings**  
Luigi Alberti,
- **SyR-e : Synchronous Reluctance Evolution**  
Gianmario Pellegrino, Francesco Cupertino



# *Electrical Machine Analysis using Free Software*

## ***FEMM (Finite Element Method Magnetics) Overview***

David Meeker, Ph.D.

QinetiQ North America



# What is FEMM?

- 2D/Axisymmetric Finite Element Program
  - Magnetics
    - DC
    - Eddy Currents
  - Electrostatics
  - Heat Conduction
  - Current Flow / Quasielectrostatic
- Graphical Pre-/Post- Processing
- Interfaces with many popular analysis tools

# Philosophy

- Focus is on solving practical problems
  - Users shouldn't *have* to write code to solve problems
  - Users shouldn't even have to be PDE experts to build and solve problems
  - It should be easy for users to get derived results from the solution (e.g. loss, power, energy, etc.), not just field plots.
- Meant to be used with commonly available desktop/laptop hardware
- Algorithms/Methods are usually “fastest algorithms that are easy to implement”

# Availability

- Distributed under Aladdin Free Public License
  - Source code available
  - No restrictions on using the code, e.g. to solve problems for commercial applications
  - Prevents some third party from using some or all of the program for their commercial product without a special license.
- Primarily available at <http://www.femm.info>
- Yahoo Group: <http://tech.groups.yahoo.com/group/femm/>

# Information on Website

- FAQ
- Step-by-Step Tutorials
- Manuals
  - Translated into French, Bulgarian, Romanian, Vietnamese
  - Examples in Spanish
  - Third-party website hosts German documentation
  - Book on FEMM available in Russian
- Many documented example problems
- User Contributions area

# Platforms

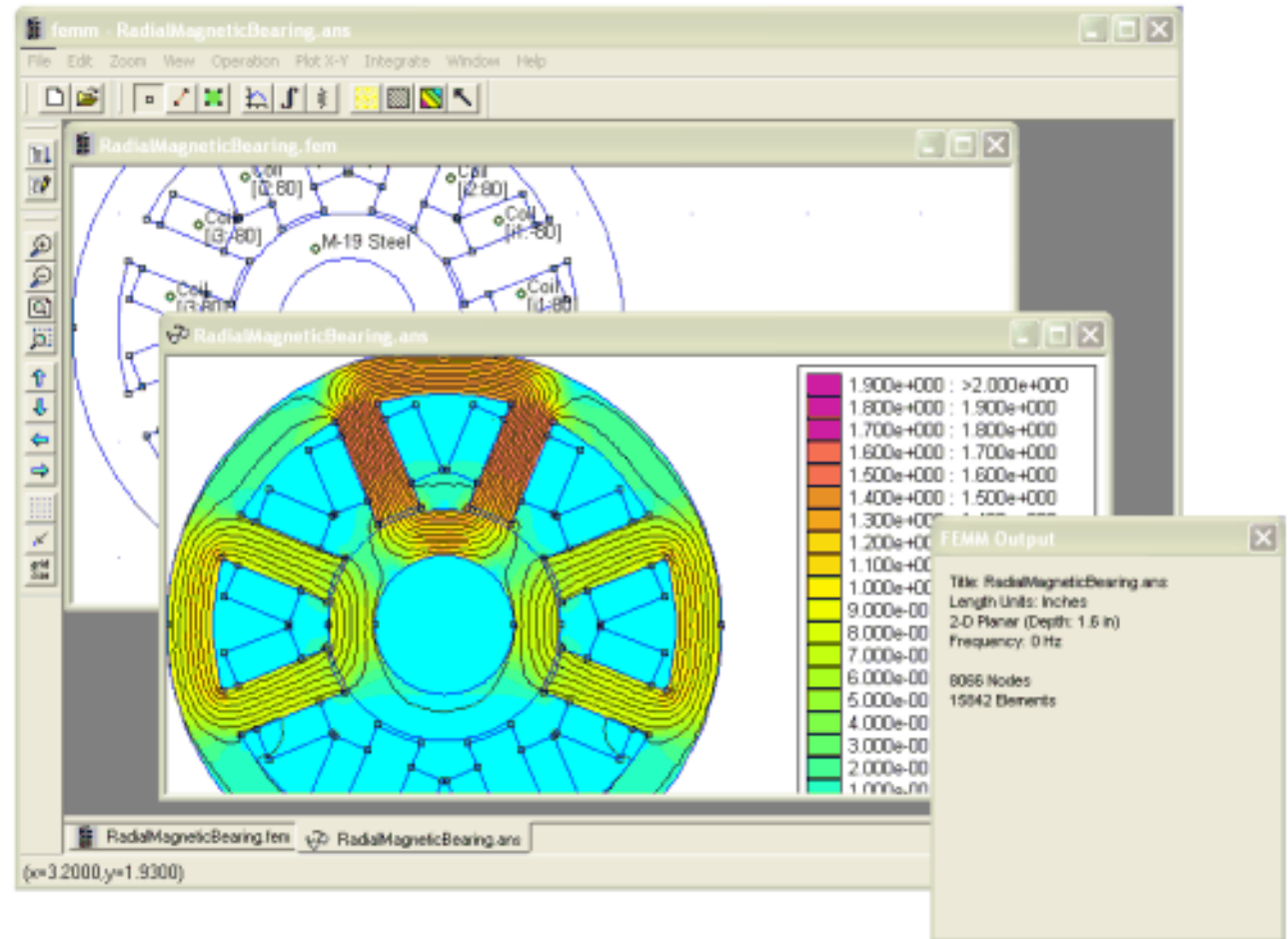
- Primarily intended to run on Windows machines
  - Compatible with all versions from WinXP-Windows 10
  - Executable distributions currently compiled with Visual C++ 2008
    - Possible to compile with later versions
    - Old compiler used because it works well in Wine
    - 32- and 64-bit versions available
- Runs on Linux, Mac machines under Wine
  - Program was modified to fix parts that didn't work right under Wine
  - Performance on Wine is comparable to Windows.

# Other Linux Option: XFEMM

- Fork of FEMM by Richard Crozier (University of Edinburgh)
- Removes all MFC constructs, replaces with standard templates.
- Primary intent is faster interactions with Matlab and to run on Linux without needing WINE

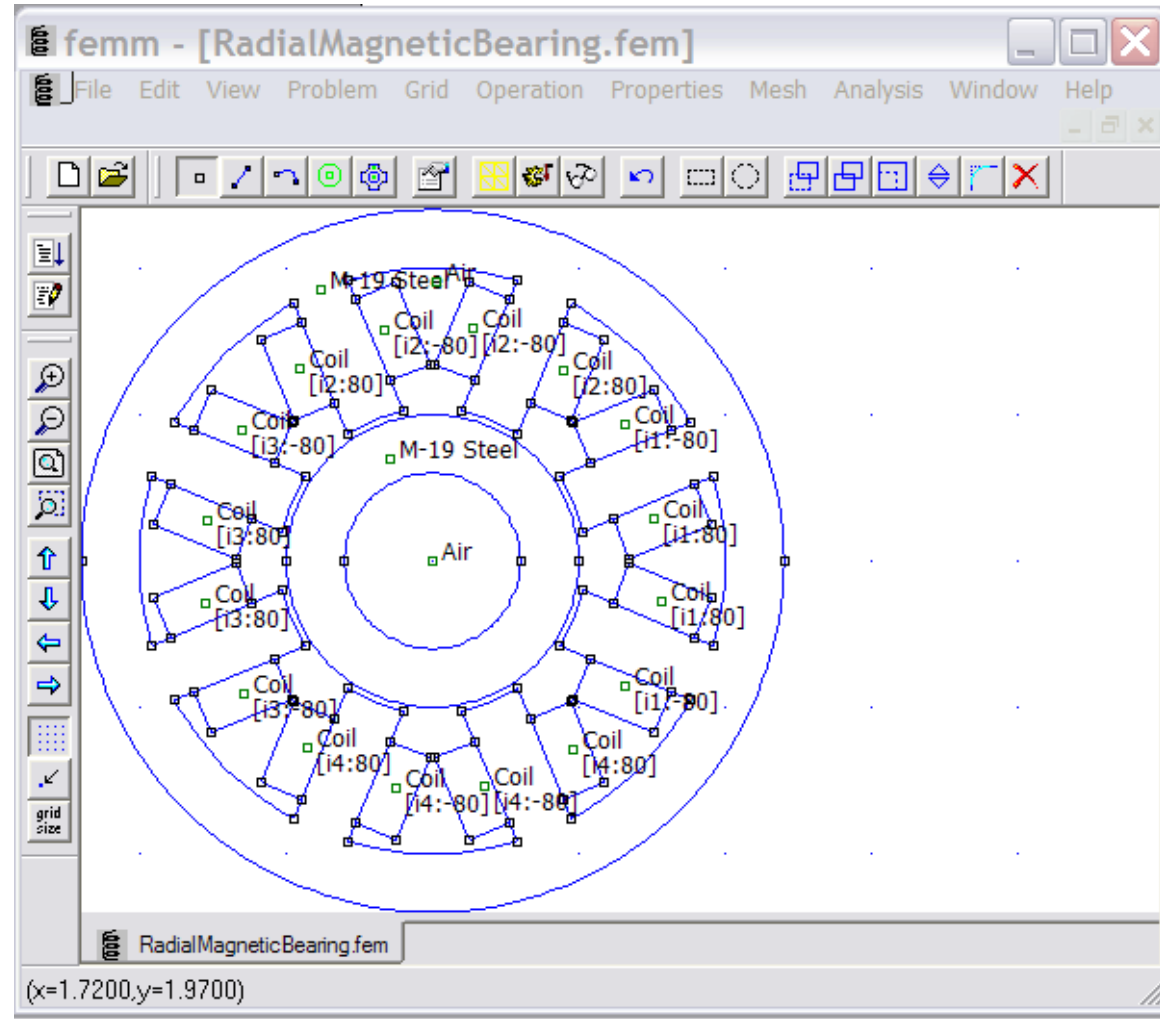
# Basic Structure

- MDI application
- Input Document
- Solution Document
- Output Window
- Taskbars / buttons for most functions



# Input Document

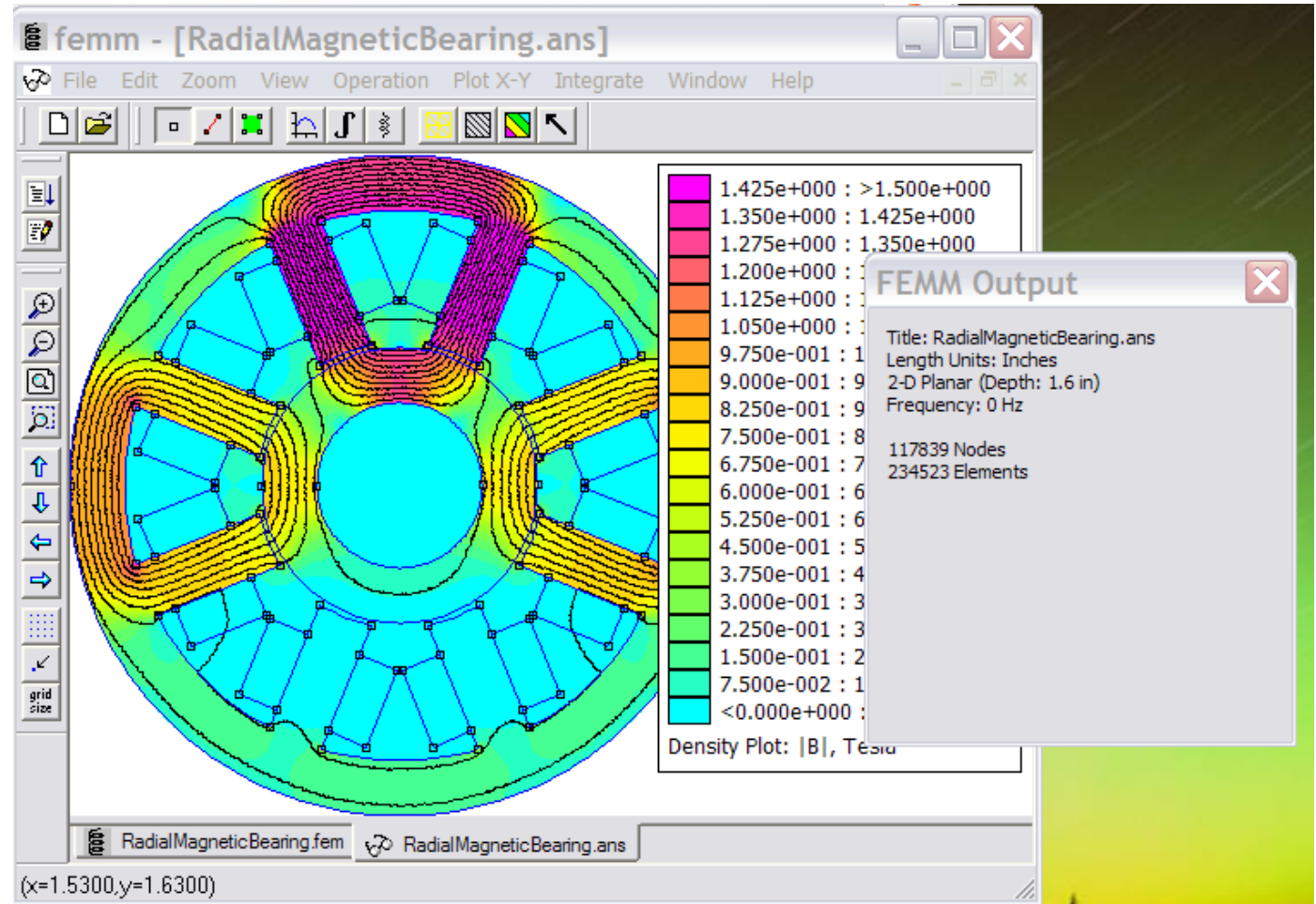
- Point, Line, Arc, Block Label, and Group Modes
- DXF Import/Export
- Define Materials, Boundary conditions, “Circuit” Properties
- Copy, Move, Rotate, Mirror
- Initiate meshing, Analysis, Solution Viewing





# Solution Document

- Point, Contour, Block modes
- Perform Integrals
- “Circuit” Properties
- Line Plots
- Vector Plots
- Density Plots
- Contour Plots



# Mesh Generation

- FEMM uses Triangle by Jonathan Shewchuk
  - Shewchuk is a CS professor at Berkeley
  - Done in conjunction with his PhD work at CMU
  - A widely used mesh generator for 2D problems because of its speed, robustness, and availability
- Triangle 1.3 compiled in the .exe distribution
  - Source for compiling with 1.5, 1.6 included
- In FEMM, Triangle wrapped into a Dialog-type application.

# Triangle Geometry Definition

- Triangle expects a PSLG (Planar-Straight-Line Graph) as input.
  - Set of points that define endpoints of lines
  - List of segments connecting defined lines
- No rigid *a priori* notion of regions
  - Block Labels, like pins on a map, define different regions of the problem
  - Determination of the shape of the region associated with the label is a by-product of the meshing process

# FEMM as Triangle GUI

- Original FEMM design conceived as a way of building geometries that Triangle could parse
- Geometry Construction Steps:
  - Define Points at ends of line segments in the geometry
  - Define Line Segments that join up the points
  - Define Block Labels that break the problem into regions
  - Label Line Segments with BCs, as appropriate.

# Finite Element Formulation

- All solvers use 1<sup>st</sup> Order triangular elements
  - Driven by ease of coding.
  - Not adaptive meshing, but heuristics make fine mesh in the places where it's needed (*i.e.* at corners)
- Vector potential formulation for magnetics
- Selection of “right” Axisymmetric Magnetic formulation was nontrivial
  - Some approaches yield poor accuracy near  $r=0$
  - Henrotte *et al*, *Trans. Magn.* 29(2):1352-1355 Mar 1993.
- “Superconvergent Patch Recovery” to smooth the piece-wise constant flux density

# Nonlinear Time Harmonic Magnetics

- Goal is to correctly compute the amplitude of the fundamental for constant frequency AC problems
- Computation effort is much lower than performing a time-stepping analysis
- “Effective” B-H curves for nonlinear materials
- Other “effective” materials that model eddy currents & hysteresis in laminated materials, skin effect and proximity effect in windings.



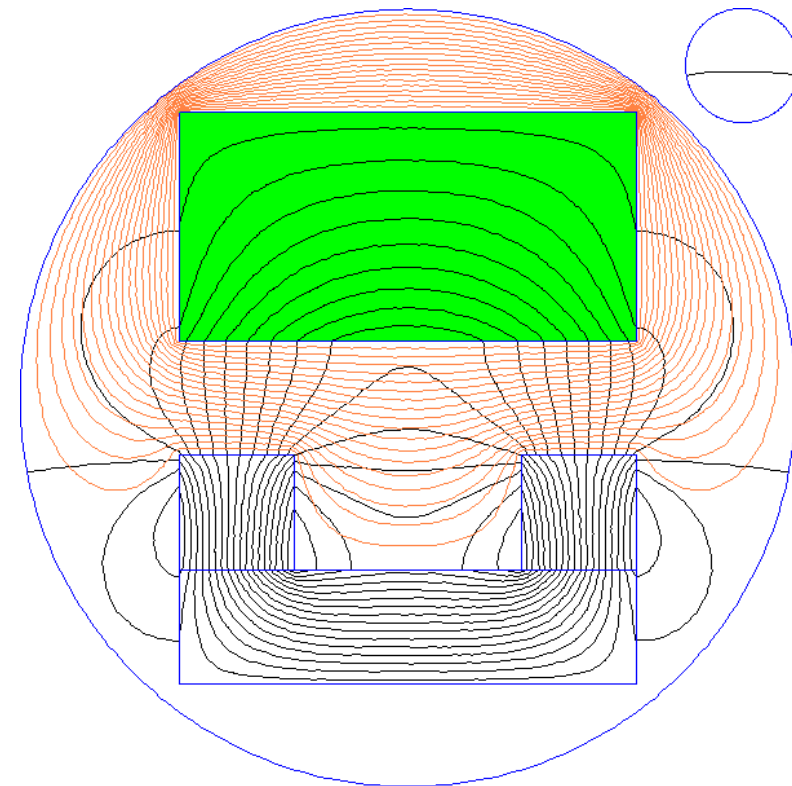
# Numerical Methods—Matrix Solver

“Homegrown” linear solvers used in FEMM

- Solver for Real Symmetric problems:
- Reverse Cuthill-McKee algorithm used to renumber nodes, elements
- Preconditioned Conjugate Gradient
- Uses SSOR (“Evans Method”) Preconditioner
  - Trivial to build, no space to store, net speed similar to Incomplete Cholesky
- Sparse storage via a tree of linked lists
- Solver for Complex Symmetric problems (AC Magn.)
- BiCG with SSOR Preconditioner

# Force Calculation: “Weighted Stress Tensor”

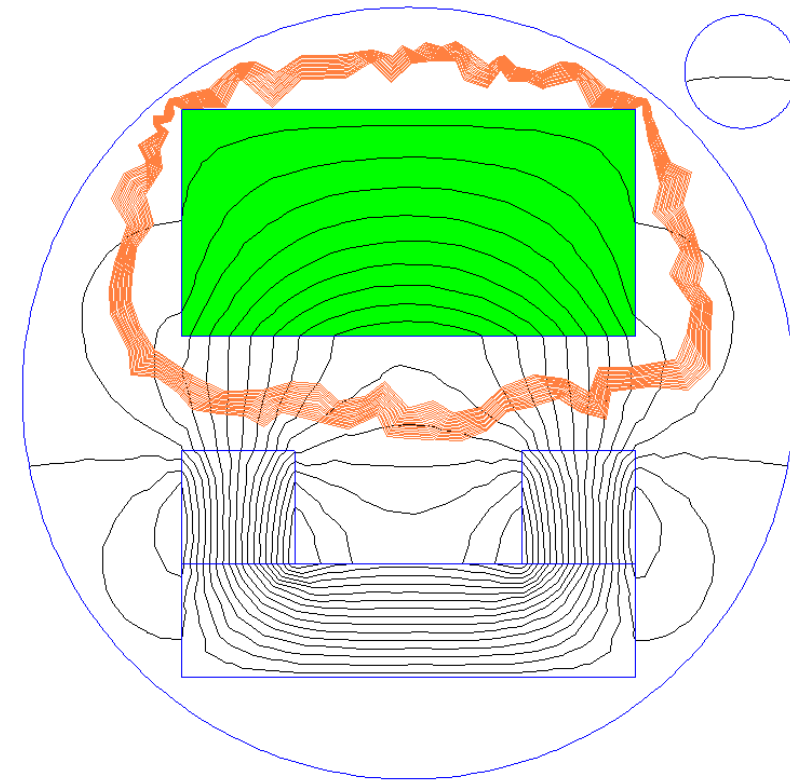
- Maxwell Stress Tensor line integral not very accurate
- Convert to a volume integral to “average over many possible paths.
- Auxiliary  $\nabla^2 u = 0$  problem solved
- Stress tensor contours follow  $\text{curl}(u)$





# Weighting Scheme

- Several different weighting schemes investigated
- Most quickly converging turns out to be one obtained by rounding the results of the field solution
- Rounded weighting scheme is the default; others possible if code is recompiled.



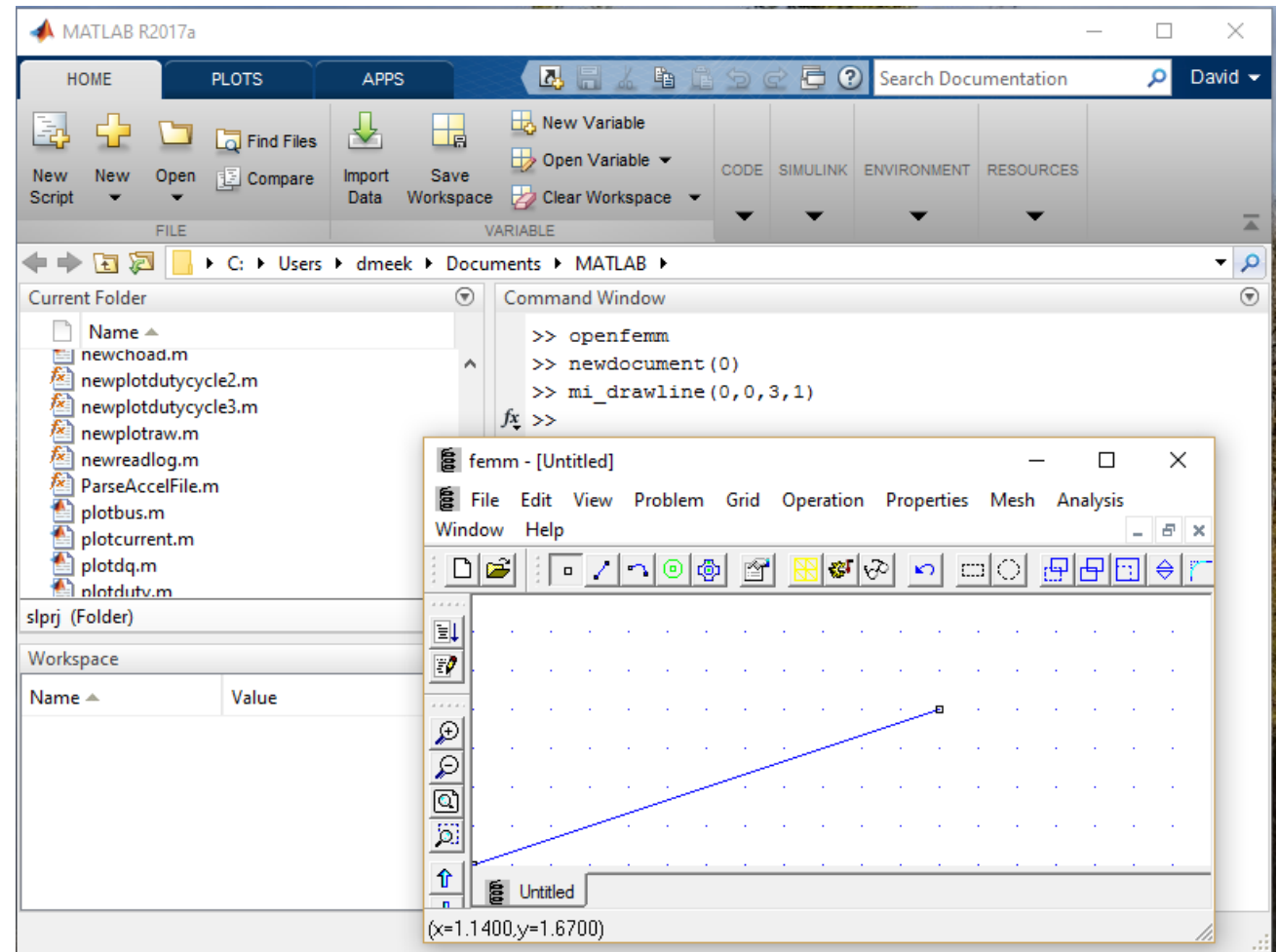
# Scripting Engine

- Lua is used by FEMM for scripting
- “Lua is a powerful, fast, lightweight, embeddable scripting language.”
- Version Lua used with FEMM is 4.0
  - Hacked Lua to make complex numbers the base number type, make trig functions use radians
  - Newer Lua (5.3.4) available, but motivation to update the Lua version used by FEMM is weak.



# Matlab/Octave Interface

- Uses Matlab/Octave Actxserver functions to instantiate and communicate with FEMM
- Back-up temporary file communication if activex not available or not installed.
  - Native install of Octave able to talk to FEMM running in Wine on Linux
- Every LUA function mirrored as a Matlab function



# Scilab Interface

- Most recently added interface
- Uses Scilab support for functions in DLLs to use ActiveX interface to FEMM
- Every Lua function implemented as an analogous Scilab function
- Ported to Scicoslab, a fork of Scilab



# Mathematica Interface

- Uses .NET/Link, an ActiveX interface
- Currently tested with versions 7-10
- Again, all Lua functionality mirrored as “native” Mathematica functions
- Set of detailed Mathematica examples included in the .exe distribution.



# Other Interfaces

- Java interface “JFEMM”
- Talk to MS Excel via ActiveX interface
- Connect to FEMM using any Visual Studio language

# Recent Directions

- **Incremental Permeability Formulation**

- Primarily used to analyze devices like speakers where the problem of interest is an AC perturbation about a DC set point with saturation.

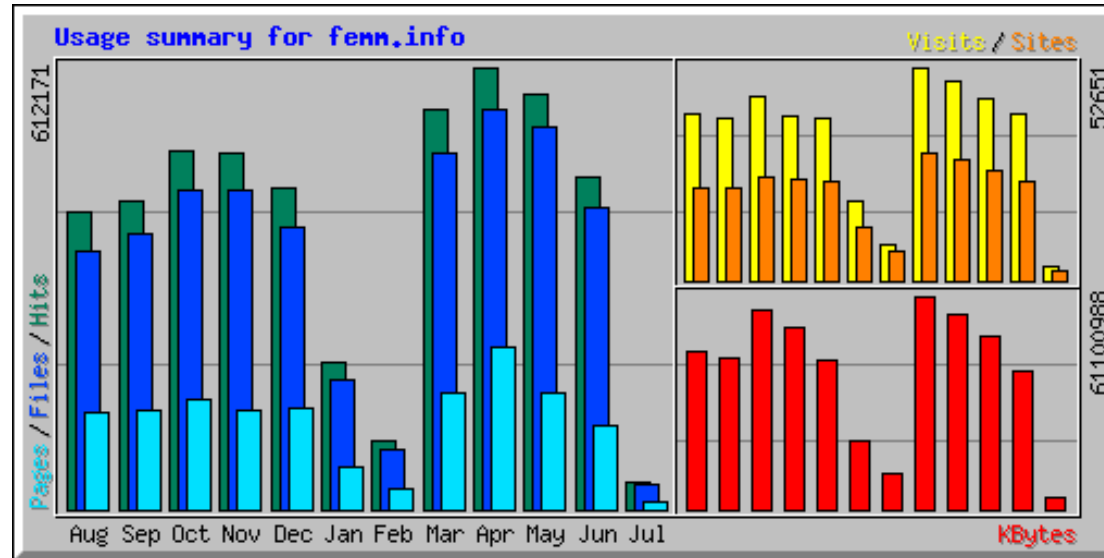
- **Frozen Permeability Formulation**

- Widely used to model PM machines

- **Optimization of Non-PM Motors for Electric Vehicles**

- Uses FEMM & Mathematica for optimal design of a class of flux switching machines.

# Level of Usage



- On average, about 9000 downloads of the .exe per month
- About 1000 source code downloads per month, on average
- Over 19 yrs, ~1M downloads of exe
- Although many download are probably old users upgrading or installing on new machines, seems possible that the user base could be on the order of 100,000.



# Use in Academic Publications

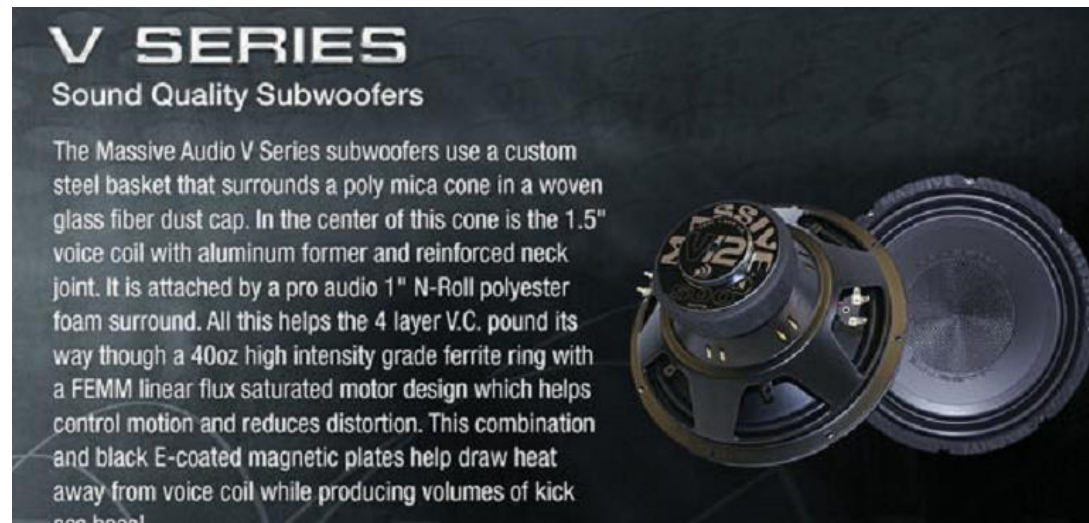
- As of this 04Jul2017, search of IEEExplore for “FEMM” yields 862 documents
- 20 papers in COMPEL
- Hundreds of dissertations and theses
- Many more documents where results, figures are used but program is not attributed

# Example Uses: 2017 IEEE Trans. Magn.

**87 so far this year; most recent ten are:**

- Optimal Design of Electromagnetic Energy Harvester Using Analytic Equations
- Computation of magnetic liquid free surface shape in a quasihomogeneous magnetic field with Differential Evolution
- Computation Time Analysis of the Magnetic Gear Analytical Model
- Efficient Simulation of Magnetic Components Using the MagPEEC-Method
- Nonlinear Analytical Prediction of Magnetic Field and Electromagnetic Performances in Switched Reluctance Machines
- Magnetoimpedance in Samples with Patterned Surfaces for the Detection of Magnetic Particles and Ferrofluids
- Magnetic Modeling of Saliency Effect for Saturated Electrical Machines With a New Calculation Method
- Response Surface Models for the Uncertainty Quantification of Eccentric Permanent Magnet Synchronous Machines
- Coated-Strand Litz Wire for Multi-Megahertz Frequency Applications
- Unbalanced Magnetic Forces Due to Rotor Eccentricity in a Toroidally Wound BLDC Motor

# Typical Use: Speaker Design



- Has been used to design many production speakers
- Used in conjunction with speaker design codes from RedRock Acoustics, LoudSoft

# Electromagnetics Education

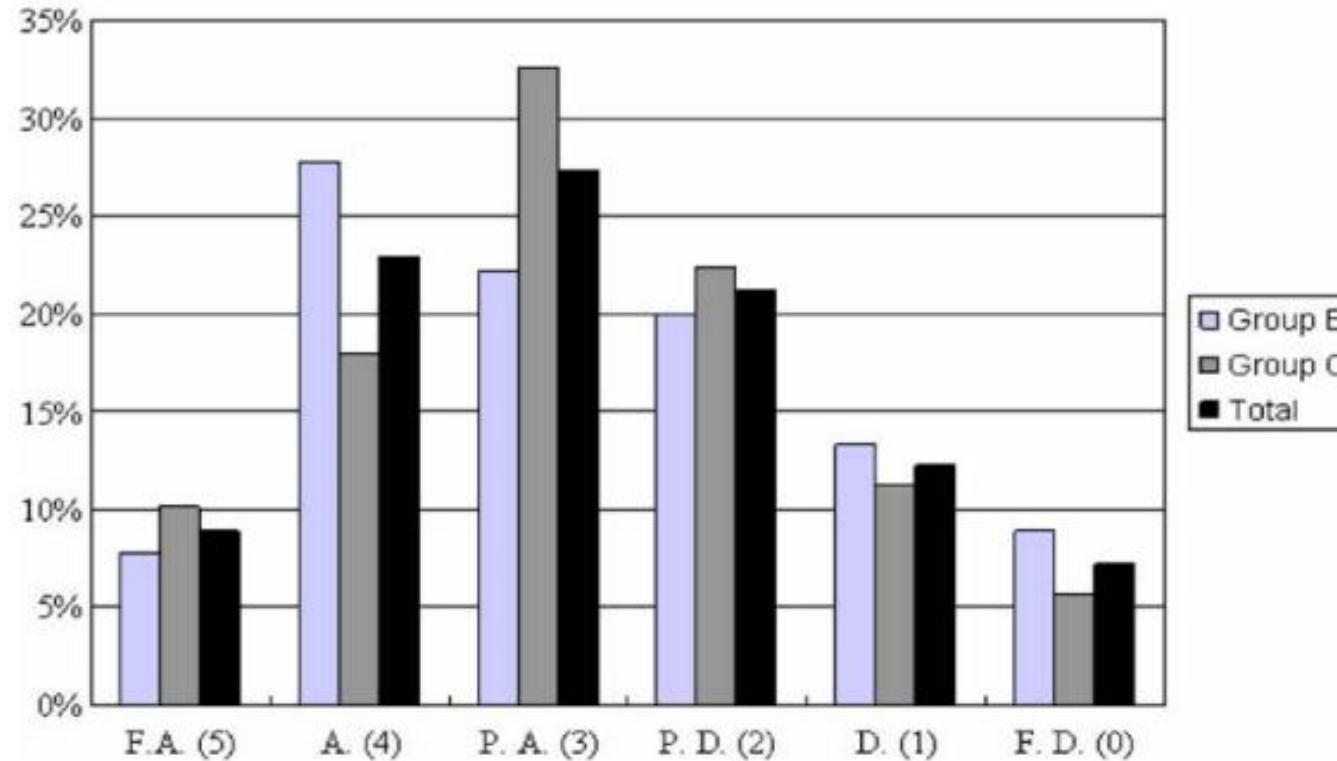


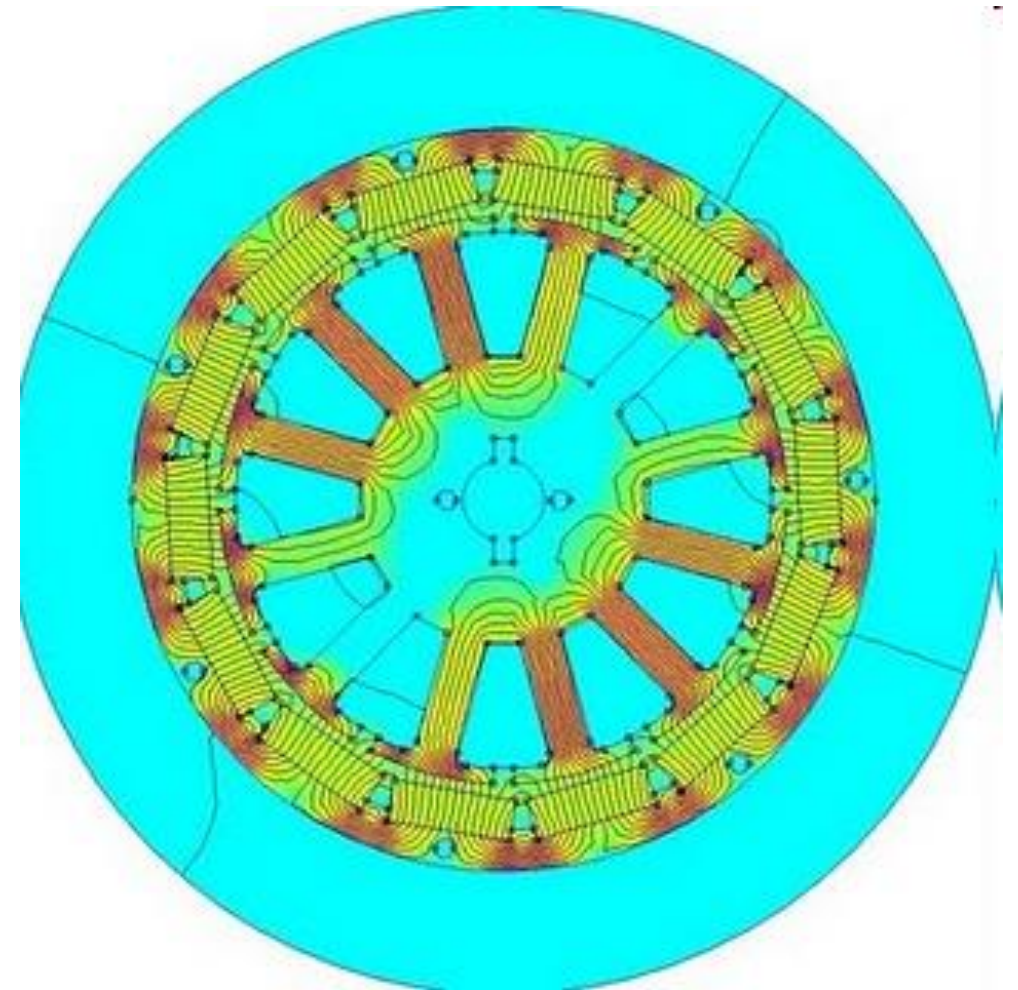
Fig. 13 [Q2.3], Software interest

- K. Baltzis, "The finite element method magnetics (FEMM) freeware package: May it serve as an educational tool in teaching electromagnetics?" Education and Information Technologies 15(1):19-36, 2010.

<http://www.springerlink.com/content/q1p6284553665717/fulltext.pdf>

# Use by Hobbyists

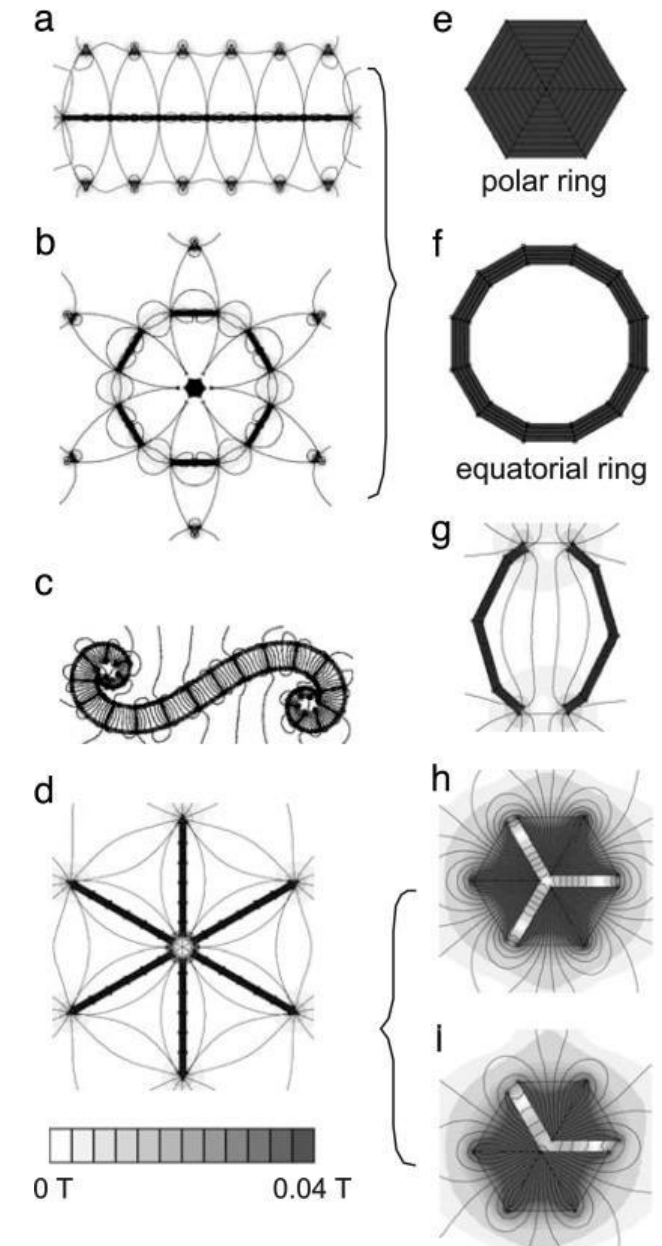
- LRK motors for model aircraft
- Motors for electric vehicles: cars, scooters, bikes
- Low-Power wind turbines
- DIY Speakers





# Unusual Applications

- M. Boncheva *et al.*, “Magnetic self-assembly of three-dimensional surfaces from planar sheets,” Proc Natl Acad Sci U S A. 2005 March 15; 102(11): 3924–3929.
- <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC554830/>
- G. Kletetschka *et al.*, Magnetic zones of Mars: Deformation-controlled origin of magnetic anomalies. Meteoritics & Planetary Science, 44: 131–140, 2009



# Future Directions

- Time-transient magnetic analysis / Motion
- Connection to external circuits, perhaps some Spice-like program
- Better coupling between problem domains
- Use multiple cores or GPU to speed up solution time
- 3D?

# PM motor design with FEMM and Lua scripting

Nicola Bianchi



Electric Drive Laboratory  
Department of Electrical Engineering  
University of Padova  
Padova, Italy

2017, October 1<sup>st</sup>

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others



- 1 A brief introduction of FEM analysis**
- 2 Introduction to Permanent Magnet Motors**
- 3 Magnetic Analysis of PM Motors**
- 4 The Electromechanical Torque**
- 5 Other predictions**

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

# A brief introduction of FEM analysis

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## FEMM (Finite Element Method Magnetics)

- it is very easy to be used,
- it is organized so as to easily understand how a finite element method works,
- it is freeware, so that anyone can download it and use it.

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

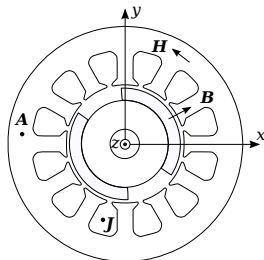
### Torque

Torque ripple reduction

### Others

## In a 2D field problem

- current density  $\mathbf{J} = (0, 0, J_z)$
- vector magnetic potential  $\mathbf{A} = (0, 0, A_z)$
- magnetic field strength  $\mathbf{H} = (H_x, H_y, 0)$
- flux density  $\mathbf{B} = (B_x, B_y, 0)$



Motor section in the plane  $(x, y)$ .

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

The first step is to define the problem:

- **Type:** select a planar symmetry, i.e. the motor section is studied in the plane  $(x, y)$ ;
- **Units:** select the measure unity for drawing, e.g. millimeter;
- **Frequency:** select the frequency, e.g. equal to zero for magnetostatic analysis;
- **Depth:** select the length of the structure, along the  $z$  axis, normal to the plane  $(x, y)$ , e.g.  $L_z=100$  mm.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

The steps of a finite element problem are

- **pre-processing**: preparing the field problem (using the editor `femme.exe`);
- **mesh**: subdivision of the domain in small elements (using `triangle.exe`);
- **solution**: solution of the field problem (using `fkn.exe`);
- **post-processing**: analysis of the field solution and computation of integral quantities (using the editor `femmview.exe`).

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

- Drawing
  - points
  - segments
  - arcs
- Mark
  - labels
  - groups
- Properties
  - materials
  - boundary conditions
  - circuits

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

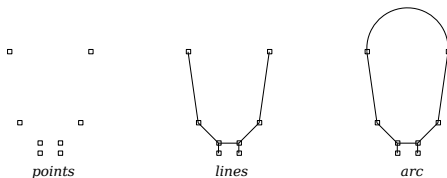
No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

- Points
- Segments
- Arcs



Example of drawing a stator slot.

## Motor drawing

Stator drawing

Air gap drawing

Rotor drawing

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

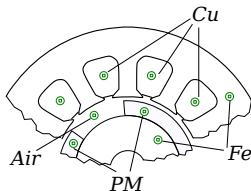
Torque ripple reduction

## Others



## Labels

- to give a name to the block;
- to assign the material properties, e.g. the electrical conductivity, and the relative magnetic permeability;
- to assign a current density to the block.



Label and material used.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Stator Label

object	label
lamination stack	Fe
wedges	Air
slot no. 1	Cu1
slot no. 2	Cu2
slot no. 3	Cu3
...	...

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Stator Groups

object	label	group
lamination stack	Fe	1000
wedges	Air	1000
slot no. 1	Cu1	1001
slot no. 2	Cu2	1002
slot no. 3	Cu3	1003
...	...	...

### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

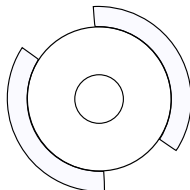
### Torque

Torque ripple reduction

### Others

## Other Groups

object	label	group
air gap	Air	0
rotor	...	10



*group = 10*

Sketch of the rotor, as group 10.

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## Four main boundary conditions

- **Dirichlet:** a given value of  $A_z$  along a line.
- **Neumann:** flux density lines normal to a line.
- **Periodic:**  $A_{z,segment1} = A_{z,segment2}$ .
- **Anti-periodic:**  $A_{z,segment1} = -A_{z,segment2}$ .

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMS  
References

### Magnetic Analysis

No load operation  
Operation under load

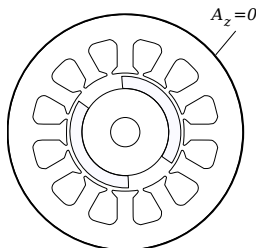
### Torque

Torque ripple reduction

### Others

## How to do

- 1 A boundary condition  $A_z = 0$  is defined in the problem.
- 2 The external lines are selected.
- 3 The boundary condition is assigned.



Boundary condition assigned along  
the outer periphery of the stator.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

The material of the objects can be defined or selected from a given `Material Library`.

## How to do

The appropriate material is assigned to the object:

- 1 Fe is assigned to stator and rotor laminations,
- 2 Cu is assigned to the stator slots,
- 3 NdFeB is assigned to the rotor PMs,
- 4 Air is assigned to the air spaces.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Current

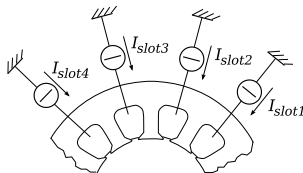
Stator currents are defined as "circuit"  
ideal current source connected to one or more blocks.

$I_{slot1}$ : current source connected to first slot,

$I_{slot2}$ : current source connected to second slot,

$I_{slot3}$ : current source connected to third slot,

... and so on.



External circuits connected to the coil sides

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others



## Current density

An alternative is to assign a current density to the various materials, e.g. to materials of the stator slots.

The peak value of current density in the slot is defined as

$$\hat{J}_{slot} = k_{fill} \sqrt{2} J_C$$

With  $k_{fill} = 0.4$ ,  
With  $J_C = 6 \text{ Arms/mm}^2$   
It is  $\hat{J}_{slot} = 3.4 \text{ A/mm}^2$ .

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

## PM motor intro

- PMS
- References

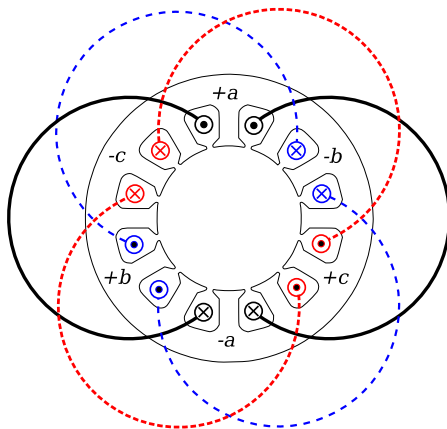
## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others



Classical representation of the stator winding

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## Slot matrix

- Matrix dimension is  $m \times Q$
- The generic element  $k_{jq}$  indicates the how much the  $q$ -th slot is filled by conductors of the  $j$ -th phase.
- $k_{jq} = 1$  means that the  $q$ -th slot is completely filled by conductors of the  $j$ -th phase;
- $k_{jq} = 0.5$  means that only 50% of the  $q$ -th slot is filled by conductors of the  $j$ -th phase;
- $k_{jq} = 0$  means that no conductor of the  $j$ -th phase is within the  $q$ -th slot.
- sign indicates the direction.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMS  
References

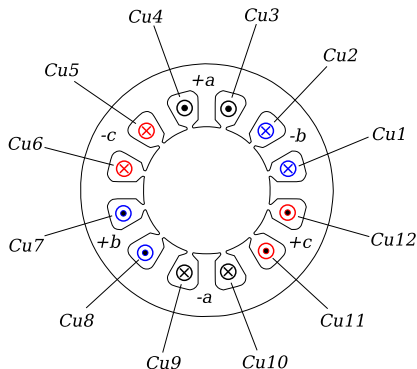
## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others



Definition of the labels of the stator slots.

## Slots Matrix

$q$	1	2	3	4	5	6	7	8	9	10	11	12
$a$	0	0	1	1	0	0	0	0	-1	-1	0	0
$b$	-1	-1	0	0	0	0	1	1	0	0	0	0
$c$	0	0	0	0	-1	-1	0	0	0	0	1	1

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Current assignment

The current in the  $q$ -th slot can be expressed as

$$I_{slot,q} = n_c (k_{a,q} I_a + k_{b,q} I_b + k_{c,q} I_c)$$

where

$q$	1	2	3	4	5	6	7	8	9	10	11	12
$k_a =$	0	0	1	1	0	0	0	0	-1	-1	0	0
$k_b =$	-1	-1	0	0	0	0	1	1	0	0	0	0
$k_c =$	0	0	0	0	-1	-1	0	0	0	0	1	1

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMS  
References

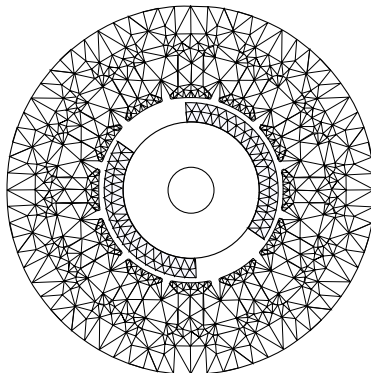
## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others



Mesh of the structure

## FEM introduction

### Meshing and solving

- Post-processing
- Use of the LUA script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

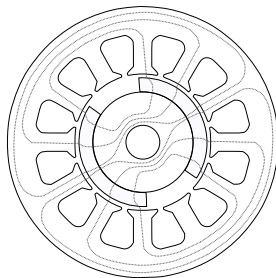
No load operation

Operation under load

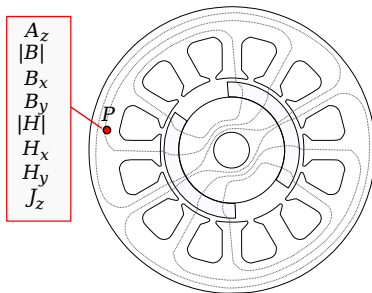
## Torque

Torque ripple reduction

## Others



Flux lines of the solved structure



Electric and magnetic quantities detected in the point P of the solved structure.

## FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation

Operation under load

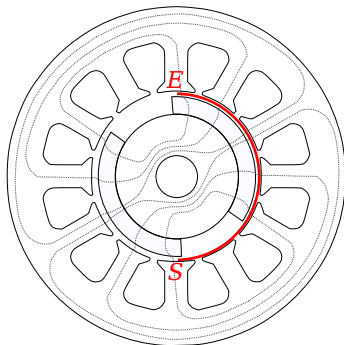
## Torque

Torque ripple reduction

## Others



$$\Phi = L_z \int_{line} B_n dl$$



## FEM introduction

Meshing and solving

Post-processing

Use of the LUA script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation

Operation under load

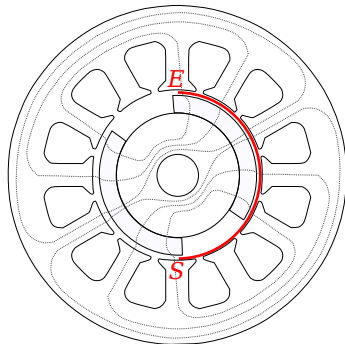
## Torque

Torque ripple reduction

## Others

Magnetic flux computation from a line integration.

$$\Phi = L_z \cdot (A_{z,S} - A_{z,E})$$



Magnetic flux from magnetic vector potential

$$\int_S \mathbf{B} \cdot \mathbf{n} dS = \int_S \text{curl} \mathbf{A} \cdot \mathbf{n} dS = \oint_{l_S} \mathbf{A} \cdot \mathbf{t} dl$$

## FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation

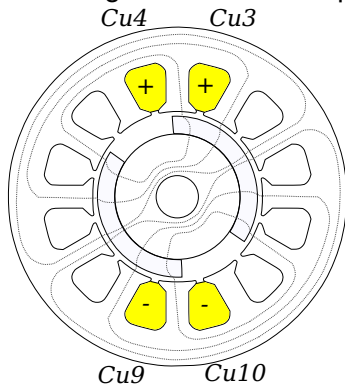
Operation under load

## Torque

Torque ripple reduction

## Others

Slots containing the coil sides of phase *a*



$$A_z \longrightarrow \frac{1}{S_{slot}} \int_{S_{slot}} A_z dS$$

FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

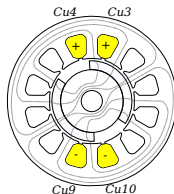
No load operation

Operation under load

Torque

Torque ripple reduction

Others



$$\Lambda_a = n_c L_z \frac{1}{S_{slot}} \left( \begin{aligned} &+ \int_{S_{slot,3}} A_z dS \\ &+ \int_{S_{slot,4}} A_z dS \\ &- \int_{S_{slot,9}} A_z dS \\ &- \int_{S_{slot,10}} A_z dS \end{aligned} \right)$$

## FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation

Operation under load

## Torque

Torque ripple reduction

## Others

Using the slot matrix

$q$	1	2	3	4	5	6	7	8	9	10	11	12
$k_a =$	0	0	1	1	0	0	0	0	-1	-1	0	0
$k_b =$	-1	-1	0	0	0	0	1	1	0	0	0	0
$k_c =$	0	0	0	0	-1	-1	0	0	0	0	1	1

$$\Lambda_a = n_c L_z \frac{1}{S_{slot}} \sum_{q=1}^Q \int_{S_{slot,q}} k_{a,q} A_z dS$$

FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

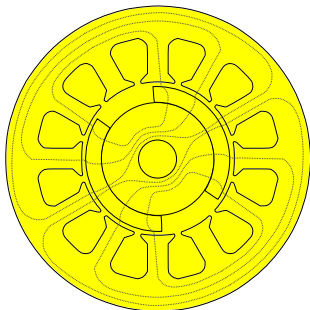
No load operation

Operation under load

Torque

Torque ripple reduction

Others



The whole structure is selected

The magnetic energy

$$W_m = L_z \int_{S_{all}} \left( \int H dB \right) dS$$

## FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation

Operation under load

## Torque

Torque ripple reduction

## Others

The magnetic energy

$$W_m = L_z \int_{S_{all}} \left( \int H dB \right) dS$$

The magnetic coenergy

$$W'_m = L_z \int_{S_{all}} \left( \int B dH \right) dS$$

The integral of  $\mathbf{A} \cdot \mathbf{J}$

$$W_{AJ} = L_z \int_{S_{all}} A_z J_z dS$$

FEM introduction

Meshing and solving

Post-processing

Use of the LUA script

PM motor intro

PMs

References

Magnetic Analysis

No load operation

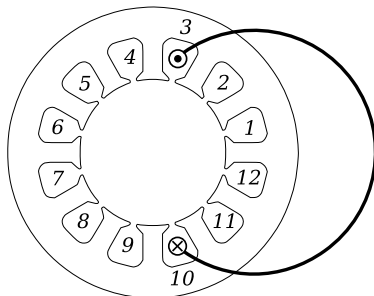
Operation under load

Torque

Torque ripple reduction

Others

The aim is to modify the current flowing in the coil, whose sides are in the slots 3 and 10.



Coil sides in slot 3 and slot 10.

FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

No load operation

Operation under load

Torque

Torque ripple reduction

Others



## The geometrical data

```
S_slot = 36.8 -- (mm2)
k_fill = 0.4
J_c = 6 -- (A/mm2)
J_slot = k_fill * sqrt2 * J_c
I_max = S_slot * J_slot
```

### FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

### PM motor intro

PMs

References

### Magnetic Analysis

No load operation

Operation under load

### Torque

Torque ripple reduction

### Others

## Current value

$$I_{\text{slot}} = I_{\text{max}} * (n / N_{\text{sim}})$$

## Current assignment to slot 3 and slot 10

```
modifycircprop("Islot", 3, +I_slot)
modifycircprop("Islot", 10, -I_slot)
analyse()
```

FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

No load operation

Operation under load

Torque

Torque ripple reduction

Others

For given current

$I_a = +4$  -- current of the a-phase

$I_b = -2$  -- current of the b-phase

$I_c = -2$  -- current of the c-phase

Using the slot matrix, the computation of the current in the  $q$ -th slot becomes

$$I_{\text{slot}[q]} = k_a[q] * I_a + k_b[q] * I_b + k_c[q] * I_c$$

FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

No load operation

Operation under load

Torque

Torque ripple reduction

Others

Starting from  $d$ - and  $q$ -axis values

$I_d = -3$  --  $I_d$  is the  $d$ -axis current

$I_q = +4$  --  $I_q$  is the  $q$ -axis current

the  $dq \rightarrow abc$  transformation is used

```
Ia = Id*cos(thetame) - Iq*sin(thetame)
Ib = (-Id/2 - Iq*sqrt(3)/2) * cos(thetame)
    + (-Id*sqrt(3)/2 + Iq/2) * sin(thetame)
Ic = (-Id/2 + Iq*sqrt(3)/2) * cos(thetame)
    + (+Id*sqrt(3)/2 + Iq/2) * sin(thetame)
```

FEM introduction

Meshing and solving

Post-processing

Use of the LUA script

PM motor intro

PMs

References

Magnetic Analysis

No load operation

Operation under load

Torque

Torque ripple reduction

Others

This is an alternative to assign the current.

After computing the current density

```
Islot = ka[q]*Ia + kb[q]*Ib + kc[q]*Ic  
Jslot = Islot / S_slot
```

it is assigned to the slot block

```
modifymaterial("Cu" .. q, 4, Jslot)
```

FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

No load operation

Operation under load

Torque

Torque ripple reduction

Others

Slot cross area (slot 3):

```
groupselectblock(1003)
Sup = blockintegral(5)
clearblock()
```

Integral of  $A_z$  (slot 3):

```
groupselectblock(1003)
intA3_r, intgA3_i = blockintegral(1)
clearblock()
```

FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

No load operation

Operation under load

Torque

Torque ripple reduction

Others

After selecting the whole domain

```
groupselectblock()
```

magnetic energy, coenergy, integral of  $\mathbf{A} \cdot \mathbf{J}$  are computed:

```
Energy = blockintegral(2)  
Coenergy = blockintegral(17)  
AJ_intgr = blockintegral(0)
```

## FEM introduction

Meshing and solving

Post-processing

Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation

Operation under load

## Torque

Torque ripple reduction

## Others

# Introduction to Permanent Magnet Motors

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others



## The rotor configurations

- SPM rotor,
- inset rotor,
- IPM rotor.

Four-pole 24-slot motors.

### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

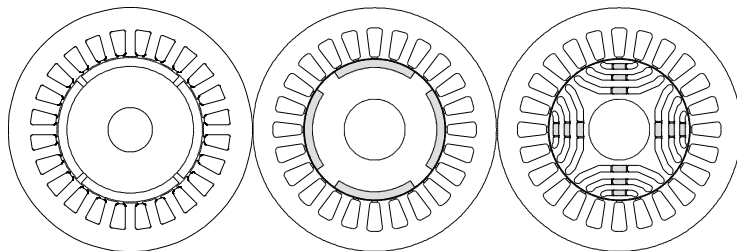
Torque ripple reduction

### Others

# The rotor configurations

PM motor design  
with FEMM and  
LUA scripting

**N. Bianchi**  
Univ. Padova  
Italy



## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

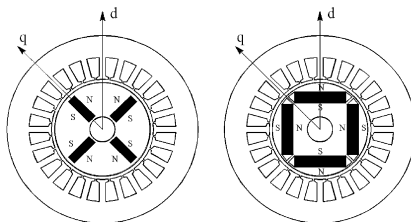
## Torque

Torque ripple reduction

## Others

## Direction of the magnetization of the PMs

- tangentially magnetized PMs,
- radially magnetized PMs.



... with two or more flux-barriers per pole.

### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

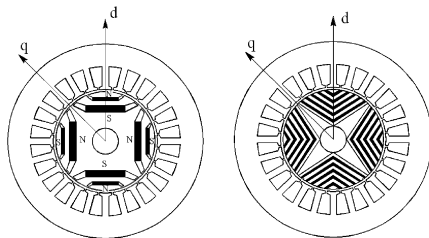
### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others



IPM motor with  
(a) two flux-barriers per pole, and  
(b) axially laminated rotor.

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

## PM motor intro

- PMs
- References

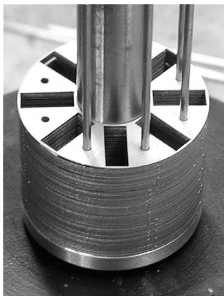
## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others



(a) IPM rotor with tangentially magnetized PMs,  
(b) an IPM motor and laminations with radially  
magnetized PMs.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

# Two main types of magnetic materials

PM motor design  
with FEMM and  
Lua scripting

N. Bianchi  
Univ. Padova  
Italy

## The soft magnetic materials

are easily magnetized and demagnetized, and they are used to carry magnetic flux.

## The hard magnetic materials

are hardly magnetized and demagnetized, and they are generally referred to as permanent magnets.

FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

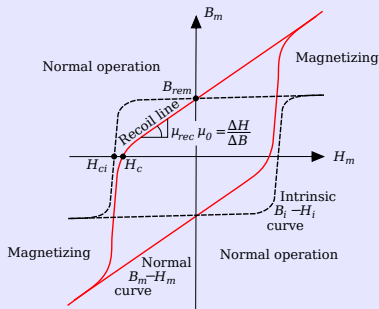
- No load operation
- Operation under load

Torque

- Torque ripple reduction

Others

## Hard magnetic material hysteresis loop and characteristic parameters.



The PMs are magnetized in quadrant I (or III) and operate in quadrant II (or IV). Properties and performance characteristics are described on quadrant II.

### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Two important quantities associated with the demagnetization curve

- residual flux density (or remanence)  $B_{rem}$ ,
- the coercive force  $H_C$ .

PMs are designed to operate on the linear part of the demagnetization curve, between  $B_{rem}$  and  $H_C$ .

## The differential relative magnetic permeability

of the recoil line is labelled  $\mu_{rec}$  and is slightly higher than unity.

## RecoilLine

$$B_m = B_{rem} + \mu_{rec}\mu_0 H_m$$

FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

PM motor intro

PMs

References

Magnetic Analysis

No load operation  
Operation under load

Torque

Torque ripple reduction

Others



## Demagnetization

When the flux density becomes lower than  $B_{knee}$  the PM is demagnetised and it exhibits lower flux density versus field strength.

This means that the PM is “irreversibly” demagnetized (in the sense that it requires a new process of magnetization).

### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

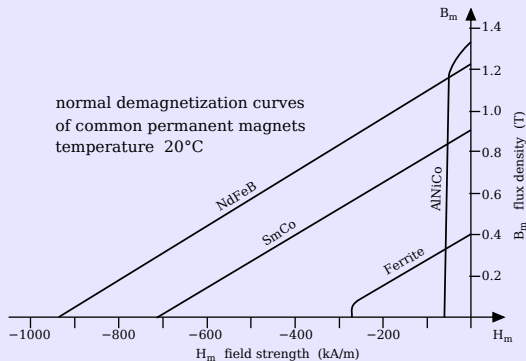
- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Demagnetization curves of common PM materials.



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

# Main properties of hard magnetic material

PM motor design  
with FEMM and  
LUA scripting

N. Bianchi  
Univ. Padova  
Italy

## Main properties of PMs

	$B_{rem}$ (T)	$H_c$ (kA/m)	$T_{max}$ (°C)	$\{B_m H_m\}_{max}$ (kJ/m <sup>3</sup> )	$\Delta B_{rem} / \Delta T$ (%/°C)
Ferrite	0.38	250	300	30	-0.20
AlNiCo	1.20	50	540	45	-0.02
SmCo	0.85	570	250	140	-0.04
NdFeB	1.15	880	180	260	-0.12

### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

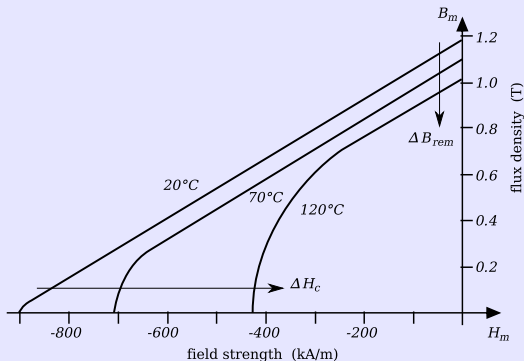
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Effect of the temperature on PM demagnetization curve.



## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

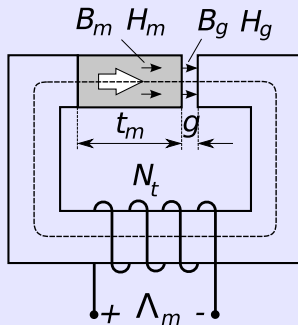
- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## Magnetic device



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

#### PMs

- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Ampere law

$$H_m t_m + H_g g = 0$$

## Gauss law

$$\Phi = B_m A_m = B_g A_g$$

together the two constitutive equations of air and PM.

### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

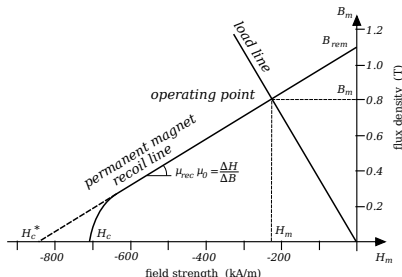
- Torque ripple reduction

### Others

## The load line

$$B_m = - \left( \mu_0 \frac{t_m}{g} \frac{A_g}{A_m} \right) H_m$$

The operating point of the PM is determined as the intersection of the PM demagnetization curve and the load line.



## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

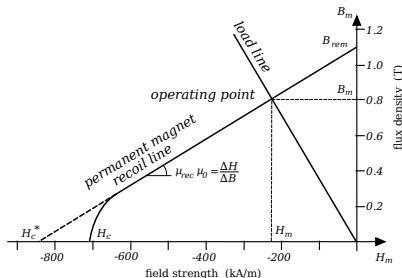
## Torque

- Torque ripple reduction

## Others

## Flux density under load

$$B_m = B_{rem} \frac{1}{1 + \frac{\mu_{rec} g}{t_m} \frac{A_m}{A_g}}$$



## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others



## Load line

$$B_m = - \left( \mu_0 \frac{t_m}{g} \frac{A_g}{A_m} \right) H_m + \mu_0 \frac{A_g}{A_m} \frac{N_t I}{g}$$

The load line remains a straight line, but it is translated along the  $H_m$  axis of a quantity  $\Delta H_i$  proportional to the current.

It should be verified that the minimum flux density is not below the knee of the PM demagnetizing curve, where an irreversible demagnetization of the PM occurs.

### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

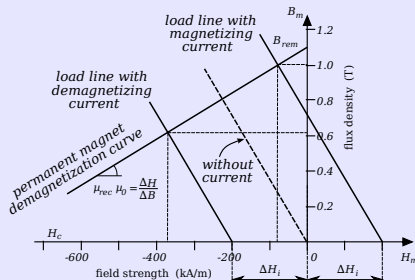
- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## PM operating point with current.



## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

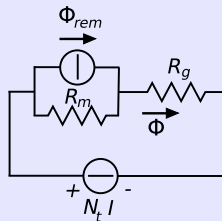
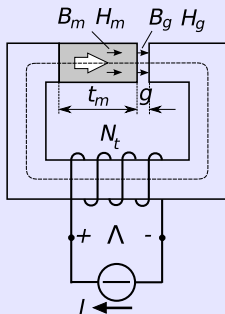
No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Magnetic circuit and equivalent magnetic network.



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

#### PMs

- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

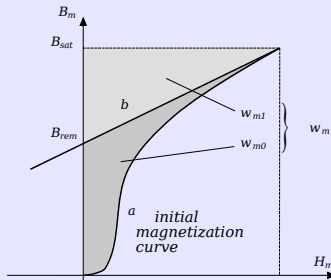
- Torque ripple reduction

### Others

## Magnetic energy density in PM

$$w_m = \int_0^{B_m} H_m dB_m$$

## BH characteristic and energy density of a hard magnetic material



### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

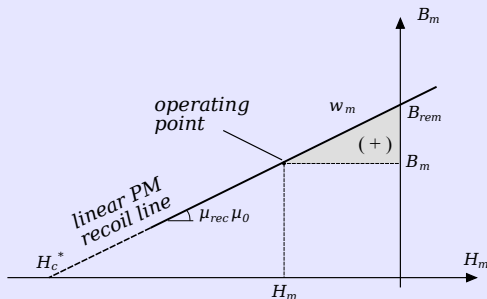
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Operating point and corresponding energy density



### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs

References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Magnetic Energy Computation for Energy Conversion Aim

$$w_m = \frac{1}{2\mu_{rec}\mu_0} (B_{rem} - B_m)^2 = \frac{1}{2}\mu_{rec}\mu_0 H_m^2$$

## Magnetic coenergy

$$w'_m = \int_0^{H_m} B_m dH_m$$

that is a negative coenergy.

## Magnetic coenergy

$$w'_m = \int_{H_C^*}^{H_m} B_m dH_m$$

that is a positive coenergy.

### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

#### PMs

- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## The difference between the coenergy density

$$\int_{H_c^*}^{H_m} B_m dH_m - \int_0^{H_m} B_m dH_m = \frac{1}{2} H_c^* B_{rem}$$

is a constant quantity.

The change of the magnetic coenergy is the same with the two calculation procedures.

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

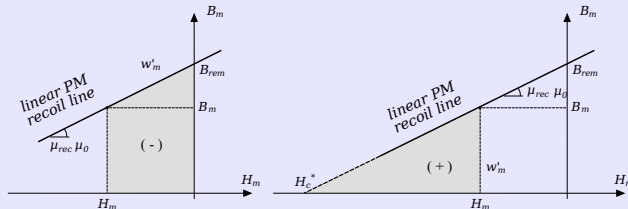
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Operating point and magnetic coenergy density



## If the linear recoil line is adopted

$$w'_m = \frac{1}{2} \mu_{rec} \mu_0 (H_m - H_c^*)^2 = \frac{1}{2 \mu_{rec} \mu_0} B_m^2$$

### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

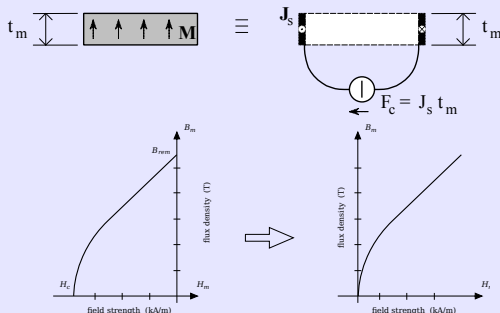


# PM approximated by a current-source coils

PM motor design  
with FEMM and  
LUA scripting

N. Bianchi  
Univ. Padova  
Italy

A PM can be represented by an equivalent distribution of current density



FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

PM motor intro

PMs  
References

Magnetic Analysis

No load operation  
Operation under load

Torque

Torque ripple reduction

Others

FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

PM motor intro

PMS

References

Magnetic Analysis

No load operation  
Operation under load

Torque

Torque ripple reduction

Others

# References

## References

- The stator reference frame is characterized by the  $a$ -,  $b$ -, and  $c$ -axis (or the equivalent  $\alpha$ - $\beta$  reference frame).
- The rotor reference frame is characterized by the  $d$ - and  $q$ -axis.
- The relative position between the rotor and the stator reference frames is individuated by the electrical angle  $\vartheta_m^e$ .

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMS

## References

## Magnetic Analysis

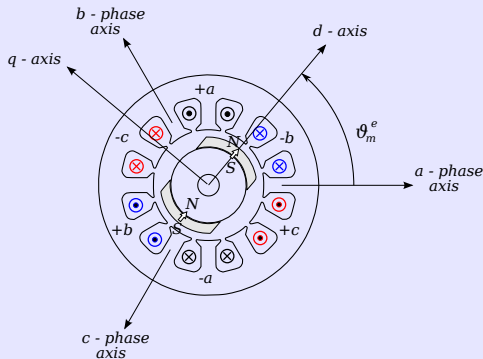
No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Reference frames



## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs

References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Space vector in stator reference frame

$$\vec{i}_s^s = i_a + i_b e^{j2\pi/3} + i_c e^{j4\pi/3}$$

or

$$\text{space vector } \vec{i}_s^s = i_{s,\alpha}^s + j i_{s,\beta}^s$$

$$i_{s,\alpha}^s = i_a - \frac{1}{2}i_b - \frac{1}{2}i_c$$

$$i_{s,\beta}^s = \frac{\sqrt{3}}{2}(-i_b + i_c)$$

FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

PM motor intro

PMS

References

Magnetic Analysis

No load operation  
Operation under load

Torque

Torque ripple reduction

Others

## Space vector in rotor reference frame

$$\vec{i}_s^r = \vec{i}_s^s e^{-j\vartheta_m^e}$$

or

## Space vector $\vec{i}_s^r = i_{sd} + j i_{sq}$

$$i_{sd} = i_{s,\alpha}^s \cos(\vartheta_m^e) + i_{s,\beta}^s \sin(\vartheta_m^e)$$

$$i_{sq} = -i_{s,\alpha}^s \sin(\vartheta_m^e) + i_{s,\beta}^s \cos(\vartheta_m^e)$$

FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

PM motor intro

PMs

References

Magnetic Analysis

No load operation  
Operation under load

Torque

Torque ripple reduction

Others

# Magnetic Analysis of PM Motors

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

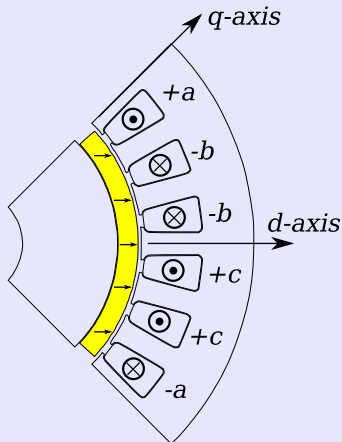
- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## Geometry of one pole of the SPM motor



### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

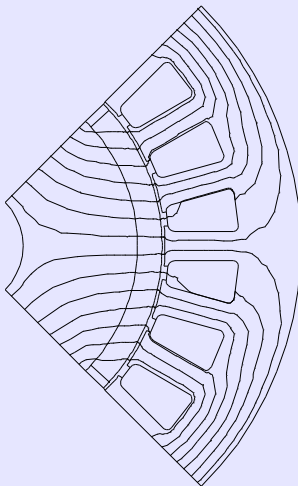
### Torque

Torque ripple reduction

### Others



## Flux lines



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation

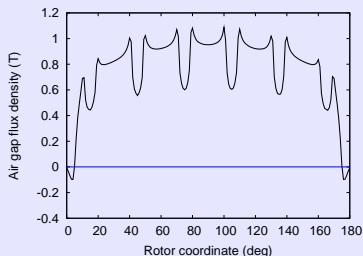
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Flux density distribution



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Flux linkages

$$\Lambda_a = n_c L_{stk} \sum_{q=1}^{Q_s} k_{a,q} \frac{1}{S_{slot}} \int_{S_{slot}} A_z dS$$

$$\Lambda_b = n_c L_{stk} \sum_{q=1}^{Q_s} k_{b,q} \frac{1}{S_{slot}} \int_{S_{slot}} A_z dS$$

$$\Lambda_c = n_c L_{stk} \sum_{q=1}^{Q_s} k_{c,q} \frac{1}{S_{slot}} \int_{S_{slot}} A_z dS$$

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

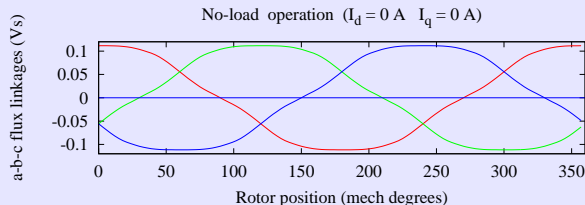
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Flux linkages versus rotor position



### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

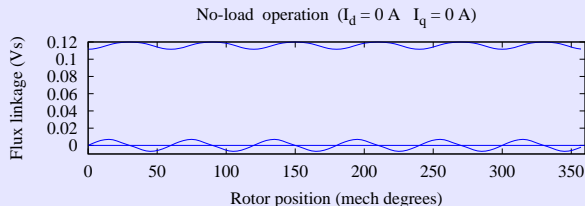
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## The $d$ - and $q$ -axis flux linkages versus rotor position.



### PM flux linkage

At no-load there is only  $d$ -axis flux linkage, while the  $q$ -axis flux linkage is equal to zero.

Therefore, at no load, the PM flux linkage is  $\Lambda_m = \Lambda_d$ .

### Analytical estimation

$$\Phi = B_g \frac{\pi D L_{stk}}{2p}$$

#### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

#### PM motor intro

- PMs
- References

#### Magnetic Analysis

- No load operation
- Operation under load

#### Torque

- Torque ripple reduction

#### Others

## Computation of the voltage under load

Assuming a constant mechanical speed  $\omega_m$  (i.e. electrical speed  $\omega = p\omega_m$ ):

$$e(t) = \frac{d\lambda(t)}{dt} = \frac{d\lambda(\vartheta_m^e)}{d\vartheta_m^e} \omega$$

It is more convenient:

- to express the flux linkage waveform by means of its Fourier series expansion.
- Then, each harmonic of the series is derived, and then all harmonics of EMF are summed together.
- The final EMF waveform is achieved.

FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

PM motor intro

PMS  
References

Magnetic Analysis

No load operation

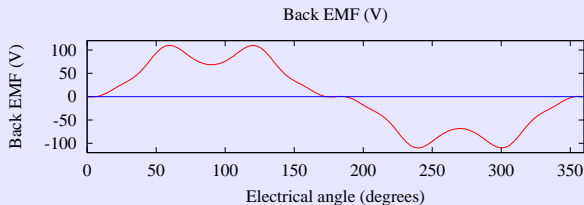
Operation under load

Torque

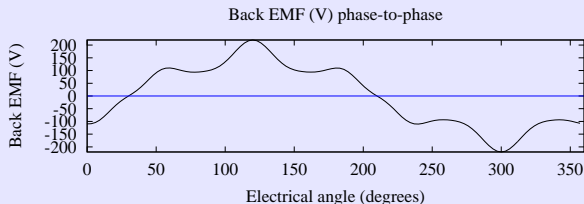
Torque ripple reduction

Others

## No load phase EMF



## No load phase-to-phase EMF



### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

A simple code to achieve the Fourier serie expansion is as follows

```
for n = 1 : Nharmo,
A(n) = 0;
B(n) = 0;
for tt = 1 : Ntheta,
A(n) = A(n) + 2/Ntheta * Flink(tt) *
cos(n*thetae(tt)*pi/180);
B(n) = B(n) + 2/Ntheta * Flink(tt) *
sin(n*thetae(tt)*pi/180);
C(n) = sqrt(A(n)^2 + B(n)^2);
F(n) = atan2(B(n), A(n));
end;
end;
```

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others



The EMF waveform is achieved as

```
w = 300;  
for tt = 1: Ntheta;  
EMF(tt) = 0;  
for n = 1 : Nharmo,  
Ce(n) = n * C(n);  
EMF(tt) = EMF(tt) + w * Ce(n) *  
sin(n*thetae(tt)*pi/180 + F(n));  
end;  
end;
```

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation

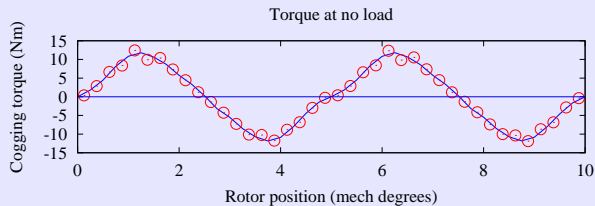
- Operation under load

## Torque

- Torque ripple reduction

## Others

## Cogging torque versus rotor position



### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Before starting the analysis under load

it is mandatory to adopt the correct reference.

This means to know the correct position of the rotor with respect the stator reference frame.

## How to position the rotor in phase with the stator

The reference position between the rotor and the stator is when the  $d$ -axis (i.e., the rotor reference axis) is parallel with the  $a$ -phase axis (i.e., the stator reference axis).

When the rotor is in such a position,  $\vartheta_m^e = 0$ .

FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

PM motor intro

PMs  
References

Magnetic Analysis

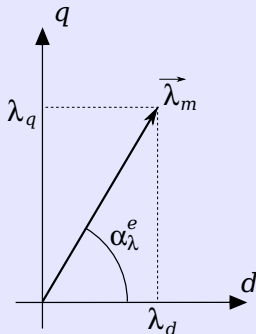
No load operation  
Operation under load

Torque

Torque ripple reduction

Others

## Flux linkage space vector and its components



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

If  $\alpha_{\lambda}^e=0$ , then the rotor is in phase with the stator, otherwise the rotor has to be rotated of a mechanical angle corresponding to the electrical angle of the flux linkage vector, that is:

$$\vartheta_m = -\frac{\alpha_{\lambda}^e}{p}$$

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

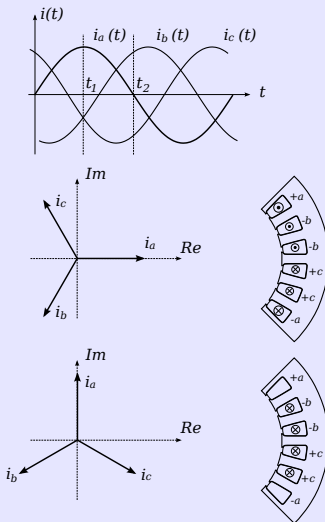
No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Three phase current



## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Time instant $t_1$

$$\begin{aligned}i_a(t_1) &= \hat{I} \\i_b(t_1) &= -\hat{I}/2 \\i_c(t_1) &= -\hat{I}/2\end{aligned}$$

## Time instant $t_2$

$$\begin{aligned}i_a(t_2) &= 0 \\i_b(t_2) &= \sqrt{3}\hat{I}/2 \\i_c(t_2) &= -\sqrt{3}\hat{I}/2\end{aligned}$$

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## $d$ -axis stator current

- A positive  $d$ -axis current is magnetizing, increasing the flux produced by the PM.
- Conversely a negative  $d$ -axis current is demagnetizing, since it weakens the PM flux.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation

Operation under load

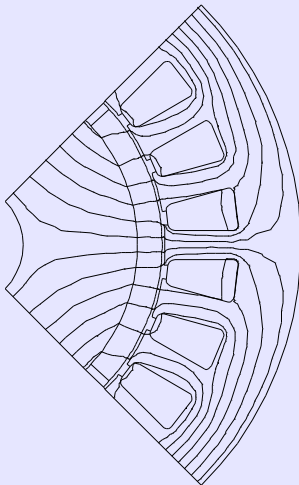
## Torque

Torque ripple reduction

## Others



## Flux lines



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation

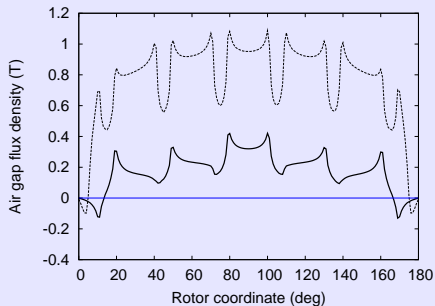
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Flux density distribution



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation

- Operation under load

### Torque

- Torque ripple reduction

### Others

## $q$ -axis stator current

- The  $q$ -axis current induces a flux in quadrature to the flux due to the PM.
- Positive and negative  $q$ -axis current produces the same effect.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation

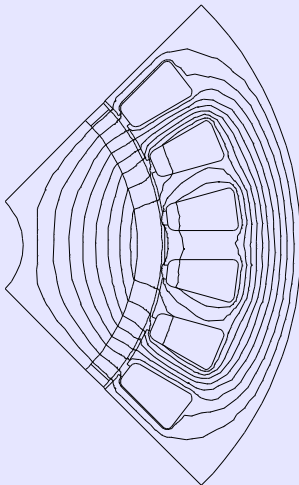
Operation under load

## Torque

Torque ripple reduction

## Others

## Flux lines



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation

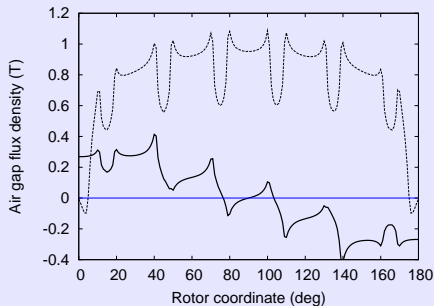
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Flux density distribution



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation

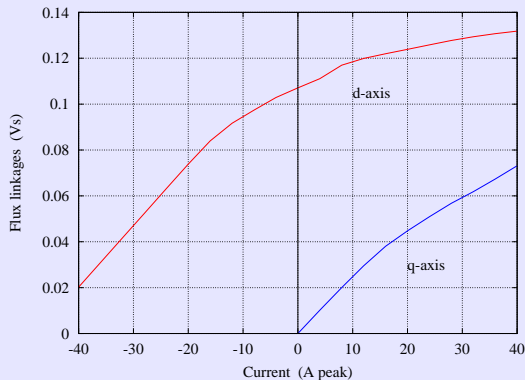
- Operation under load

### Torque

- Torque ripple reduction

### Others

## $d$ - and $q$ -axis flux linkages versus current.



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## $d$ -axis inductance

$$L_d = \frac{\Lambda_d - \Lambda_m}{I_d}$$

## $q$ -axis inductance

$$L_q = \frac{\Lambda_q}{I_q}$$

## saliency ratio

$$\xi = \frac{L_q}{L_d}$$

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMS  
References

### Magnetic Analysis

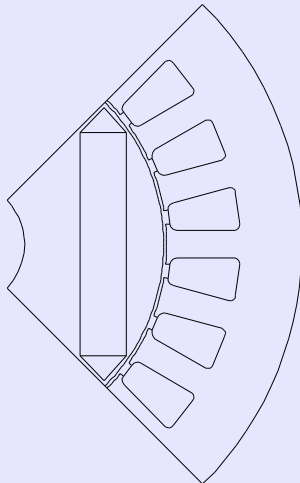
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Geometry



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation

- Operation under load

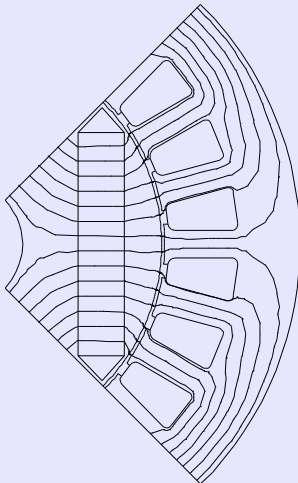
### Torque

- Torque ripple reduction

### Others



## Only PM



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation

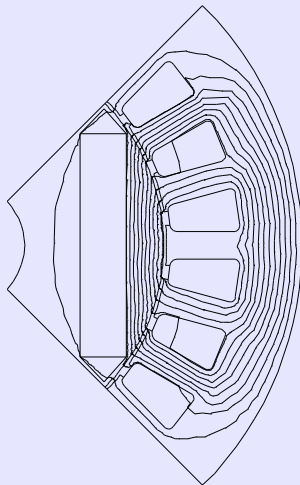
- Operation under load

### Torque

- Torque ripple reduction

### Others

only  $I_q$



## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

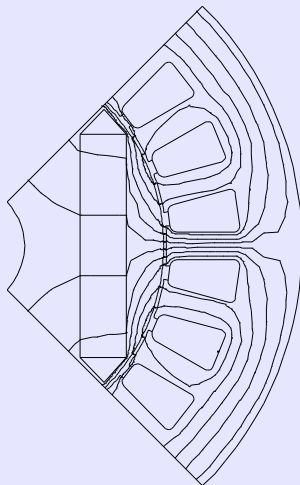
- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

only  $I_d$



## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

In the synchronous  $d - q$  reference frame (which is rotating at the electrical angular speed  $\omega$ ), the  $d$ - and  $q$ -axis flux linkage components are given by

$$\begin{aligned}\lambda_d &= \Lambda_m + L_d i_d \\ \lambda_q &= L_q i_q\end{aligned}$$

## The $d$ - and $q$ -axis voltage components

$$\begin{aligned}v_d &= R i_d + \frac{d\lambda_d}{dt} - \omega \lambda_q \\ v_q &= R i_q + \frac{d\lambda_q}{dt} + \omega \lambda_d\end{aligned}$$

### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMS  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

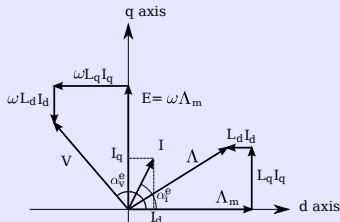
### Others

# Magnetic model of the PM synchronous motor

PM motor design  
with FEMM and  
Lua scripting

N. Bianchi  
Univ. Padova  
Italy

## Steady-state vector diagram in $d - q$ reference frame



## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

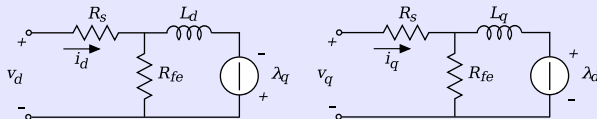
## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Equivalent circuit of PM synchronous motor



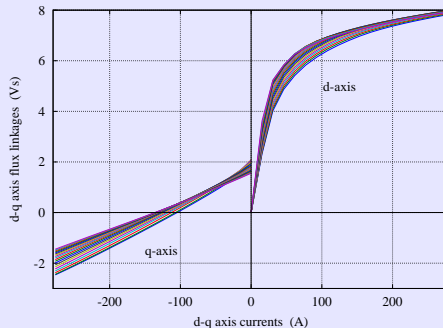
in  $d - q$  reference frame including iron losses resistance

## Torque

Torque ripple reduction

## Others

## Cross-saturation effect



$$\begin{aligned}\lambda_d &= \lambda_d(i_d, i_q) \\ \lambda_q &= \lambda_q(i_d, i_q)\end{aligned}$$

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

# The Electromechanical Torque

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Maxwell stress tensor

$$T_{Mxw} = \frac{D}{2} \int_{line} B_r \cdot H_t dl$$

directly from the field solution.

## In the synchronous reference frame

$$T = \frac{3}{2}p(\lambda_d i_q - \lambda_q i_d) + \frac{\partial W'_m}{\partial \vartheta_m}$$

where  $W'_m$  is the magnetic coenergy, which must be considered as a state function of the state variables  $\vartheta_m$ ,  $i_d$  and  $i_q$ , i.e.,  $W'_m = W'_m(\vartheta_m, i_d, i_q)$

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others



## The first term

$$T_{dq} = \frac{3}{2}p(\lambda_d i_q - \lambda_q i_d).$$

or

$$T_{dq} = \frac{3}{2}p(\vec{\lambda} \times \vec{i}).$$

where  $\vec{\lambda} = \lambda_d + j\lambda_q$  and  $\vec{i} = i_d + ji_q$   
and  $\times$  means the cross vector product.

It is independent of the particular reference frame.

It is not limited to the synchronous reference frame.

Therefore, it can be used in stationary and any other reference frame.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

The torque ripple is mainly described in the torque term expressed by  $\frac{\partial W'_m}{\partial \vartheta_m}$ .

The torque term  $T_{dq}$  is slightly affected by the harmonics of the flux linkages and it results to be suitable for the computation of the average torque.

**In an ideal system,**

the  $d$ - and  $q$ -axis flux linkages are constant, as well as the magnetic energy.

Therefore, the motor torque is constant and equal to  $T_{dq}$ .

FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

PM motor intro

- PMs
- References

Magnetic Analysis

- No load operation
- Operation under load

Torque

- Torque ripple reduction

Others

When the stator currents are zero, it is  $T_{dq}=0$ .

## Cogging torque

$$T_{cog} = \frac{\partial W'_m}{\partial \vartheta_m} = - \frac{\partial W_m}{\partial \vartheta_m}$$

Cogging torque is the ripple torque due to the interaction between the PM flux and the stator teeth.

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

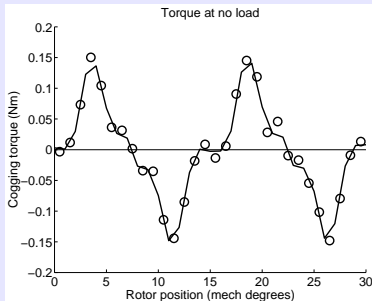
### Others

# Torque behavior at no load (cogging torque)

PM motor design  
with FEMM and  
Lua scripting

N. Bianchi  
Univ. Padova  
Italy

## SPM motor



## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

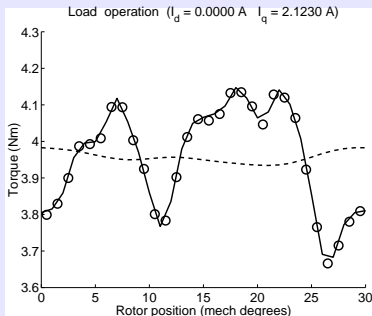
## Torque

Torque ripple reduction

## Others

Solid line: computation using Maxwell stress tensor,  
Dashed line: using  $T_{dq}$  only, Circles: using  $T_{dq} + \frac{\partial W'_m}{\partial \vartheta_m}$ .

## SPM motor



Solid line: computation using Maxwell stress tensor,

Dashed line: using  $T_{dq}$  only, Circles: using  $T_{dq} + \frac{\partial W'_m}{\partial \theta_m}$ .

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

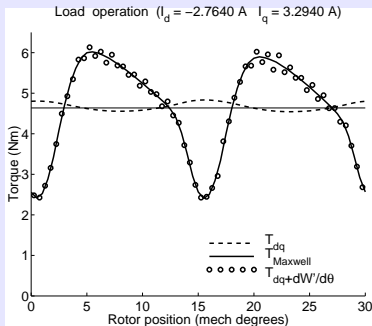
- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## IPM motor



Torque versus rotor position.

Torque behavior at  $\hat{I} = 4.3$  A and  $\alpha_i^e = 130$  deg.

## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

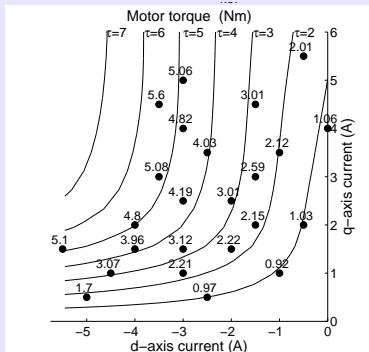
No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## IPM motor



Torque maps.

Predicted  $T_{dq}$  and measured (dots) torque in  $(i_d, i_q)$  plane.

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

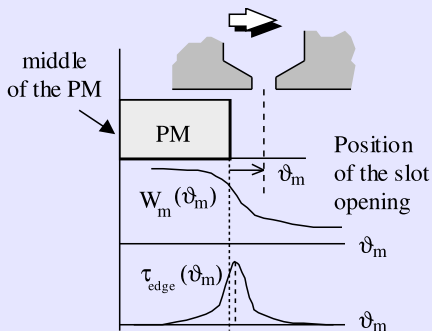
- Torque ripple reduction

## Others

# Cogging torque phenomenon

PM motor design  
with FEMM and  
Lua scripting

N. Bianchi  
Univ. Padova  
Italy



Simple model of the cogging torque mechanism, based on the superposition of PM edge torques  $\tau_{edge}$ ,

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

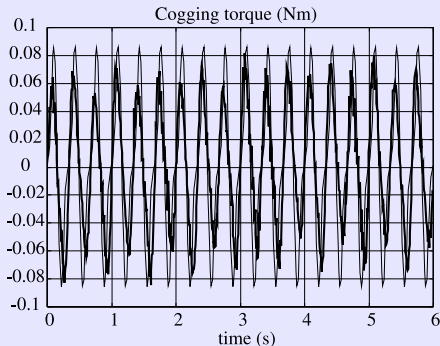
- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others





Comparison between predicted (thin line) and measured (bold line) cogging torque (motor rated torque is 3 Nm).

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## Number of $T_{cog}$ periods in a slot pitch rotation

For a rotor with identical PM poles, equally spaced around the rotor, the number of  $T_{cog}$  periods during a slot pitch rotation is given by

$$N_p = \frac{2p}{GCD\{Q, 2p\}}$$

where  $GCD$  means Greater Common Divisor.

The value of  $N_p$  is an index that shows if the elementary cogging torque waveforms are in-phase or not.

FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

PM motor intro

- PMs
- References

Magnetic Analysis

- No load operation
- Operation under load

Torque

- Torque ripple reduction

Others

## Number of $T_{cog}$ periods in a slot pitch rotation

$2p$	2	2	2	2	4	4	4	8	8	8
$Q$	3	6	9	12	6	9	12	6	9	15
$GCD$	1	2	1	2	2	1	4	2	1	1
$N_p$	2	1	2	1	2	4	1	4	8	8

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

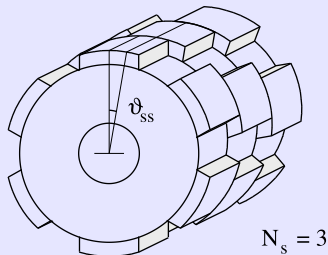
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Stepped Skewing



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## rotor with PMs of different arc width



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

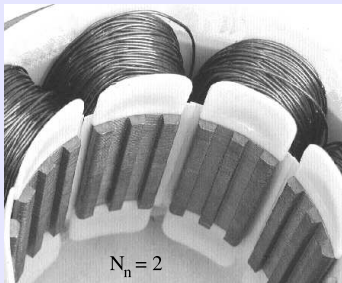
## Notches in the tooth (dummy slots)



$$N_n = 1$$



$$N_n = 2$$



A proper number of notches  $N_n$  is obtained when  
 $GCD\{(N_n + 1), N_p\} = 1$

### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

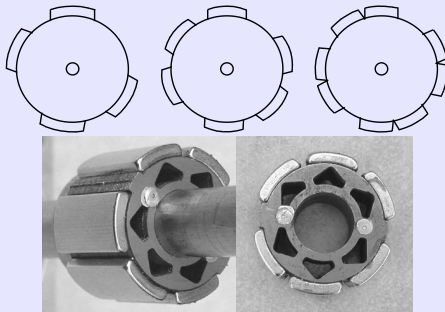
- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Shifting of the PMs



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

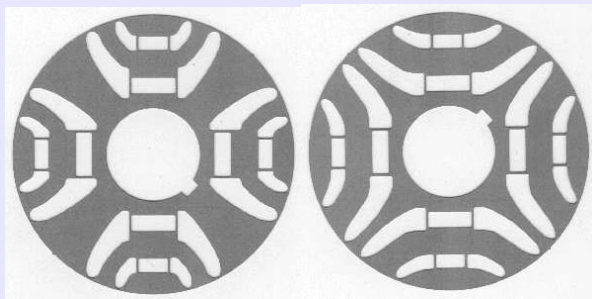
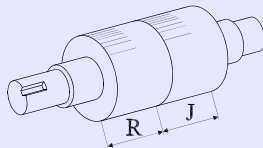
- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Two-part rotor



Photos of the "Romeo and Juliet" laminations

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

- No load operation
- Operation under load

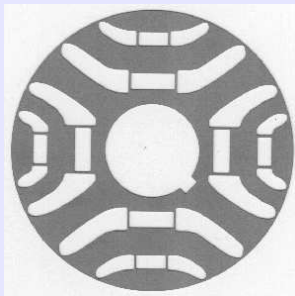
## Torque

- Torque ripple reduction

## Others



## "Machaon" rotor



It is formed by laminations with flux-barriers of different geometry, large and small alternatively under the adjacent poles.

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

## PM motor intro

- PMS
- References

## Magnetic Analysis

- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

# Torque ripple reduction (IPM motor)

PM motor design  
with FEMM and  
Lua scripting

N. Bianchi  
Univ. Padova  
Italy

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

$I$ (A)	classic IPM		R&J		Machaoon	
	$T_{avg}$ (Nm)	$\Delta T / T_{avg}$ (%)	$T_{avg}$ (Nm)	$\Delta T / T_{avg}$ (%)	$T_{avg}$ (Nm)	$\Delta T / T_{avg}$ (%)
2.64	2.14	13.1	1.96	4.84	2.182	4.757
2.84	2.39	12.2	2.18	4.91	2.430	4.720
5.30	5.02	11.6	4.63	4.92	5.240	5.784

# Other predictions

## FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

## PM motor intro

- PMs
- References

## Magnetic Analysis

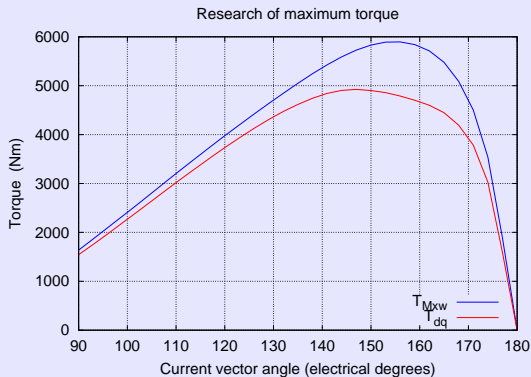
- No load operation
- Operation under load

## Torque

- Torque ripple reduction

## Others

## Torque versus current vector angle



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

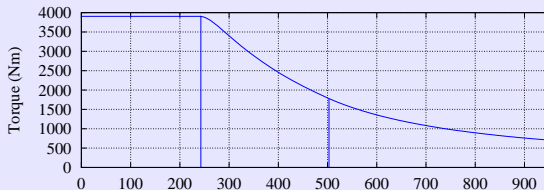
### Others

# Torque versus speed curve

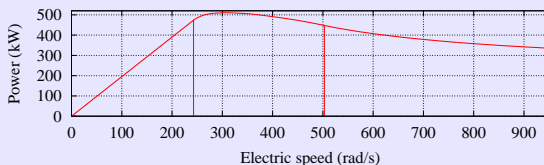
PM motor design  
with FEMM and  
LUA scripting

N. Bianchi  
Univ. Padova  
Italy

## Torque and power versus speed



Torque and power



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

- No load operation
- Operation under load

### Torque

- Torque ripple reduction

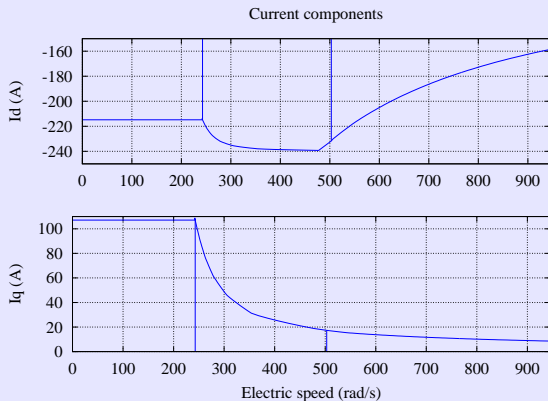
### Others

# Torque versus speed curve

PM motor design  
with FEMM and  
Lua scripting

N. Bianchi  
Univ. Padova  
Italy

## $d - q$ axis current versus speed



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

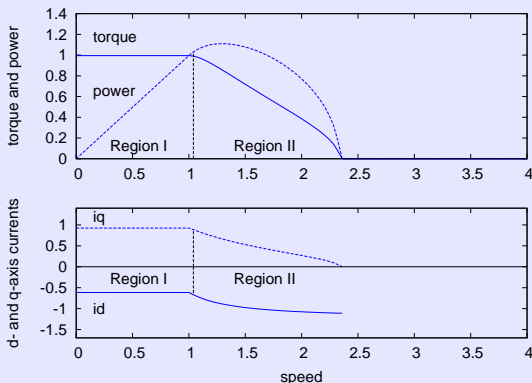
- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Maximum torque control with voltage and current limits, in case of $\Lambda_m - L_d I_N > 0$



Torque, power,  $d$ - and  $q$ -axis currents versus speed.

### FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

### PM motor intro

PMs  
References

### Magnetic Analysis

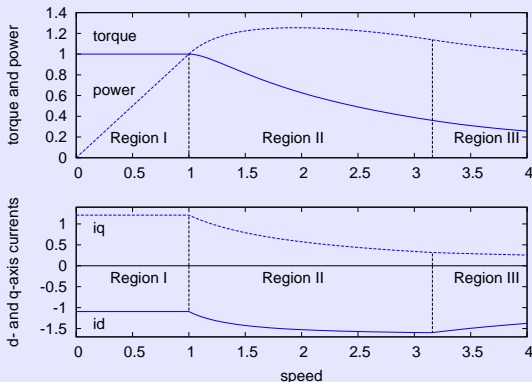
No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others

## Maximum torque control with voltage and current limits, in case of $\Lambda_m - L_d I_N < 0$



Torque, power,  $d$ - and  $q$ -axis currents versus speed.

### FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

### PM motor intro

PMs  
References

### Magnetic Analysis

No load operation  
Operation under load

### Torque

Torque ripple reduction

### Others



## The short-circuit fault

In the event of a three-phase short-circuit,  $v_d = v_q = 0$ .  
Thus,

$$I_{d,shc} = -\frac{\omega^2 L_q \Lambda_m}{R^2 + \omega^2 L_d L_q}$$

and

$$I_{q,shc} = -\frac{\omega \Lambda_m R}{R^2 + \omega^2 L_d L_q}$$

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

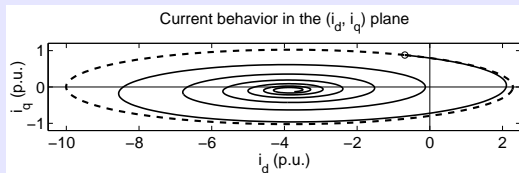
No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

## Short-circuit current trajectory in the $(i_d, i_q)$ plane



## The steady-state braking torque

$$T_{brk} = -\frac{3}{2}pR\Lambda_m^2\omega \frac{R^2 + \omega^2 L_q^2}{(R^2 + \omega^2 L_d L_q)^2}$$

## The maximum braking torque

$$T_{brk}^* = \frac{3}{2}p\frac{\Lambda_m^2}{L_q}f(\xi)$$

It occurs at the speed

$$\omega^* = \frac{R}{L_q}\sqrt{\chi}$$

with  $f(\xi) \approx \xi - 1$ .

FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

PM motor intro

PMs  
References

Magnetic Analysis

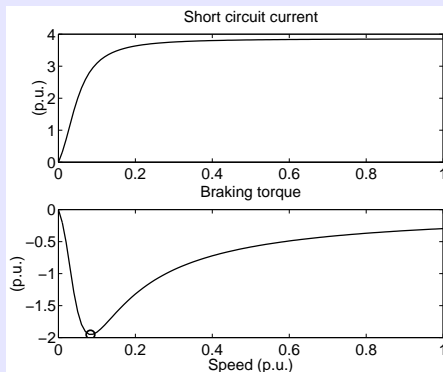
No load operation  
Operation under load

Torque

Torque ripple reduction

Others

## Short-circuit current and braking torque versus speed



### FEM introduction

- Meshing and solving
- Post-processing
- Use of the LUA script

### PM motor intro

- PMs
- References

### Magnetic Analysis

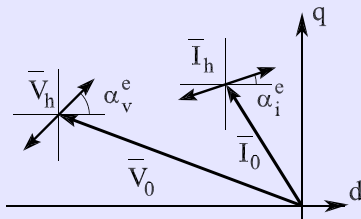
- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Phasor diagram with steady-state and high frequency components



Such a study is repeated, varying the voltage vector angle  $\alpha_v^e$ , so as to estimate the rotor position error signal  $\varepsilon$ .

## The rotor position estimation error signal

$$\varepsilon(\alpha_v^e) = k_{st} \frac{I_{max} - I_{min}}{2} \sin 2(\alpha_v^e - \alpha_{I_{max}}^e).$$

### FEM introduction

- Meshing and solving
- Post-processing
- Use of the Lua script

### PM motor intro

- PMs
- References

### Magnetic Analysis

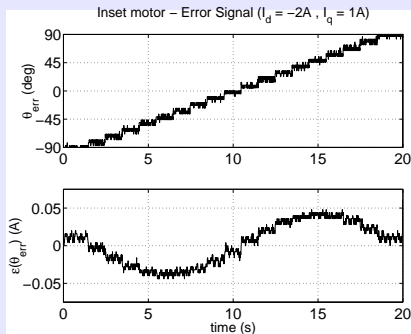
- No load operation
- Operation under load

### Torque

- Torque ripple reduction

### Others

## Experimental



## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

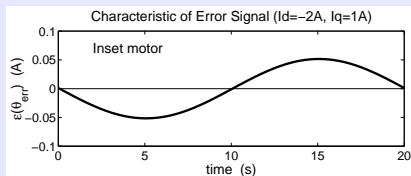
No load operation  
Operation under load

## Torque

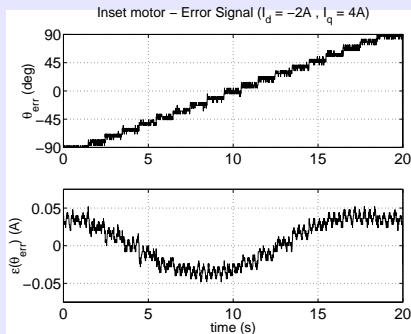
Torque ripple reduction

## Others

## Prediction



## Experimental



## FEM introduction

Meshing and solving  
Post-processing  
Use of the LUA script

## PM motor intro

PMs  
References

## Magnetic Analysis

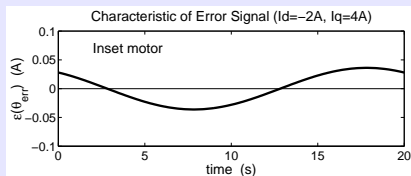
No load operation  
Operation under load

## Torque

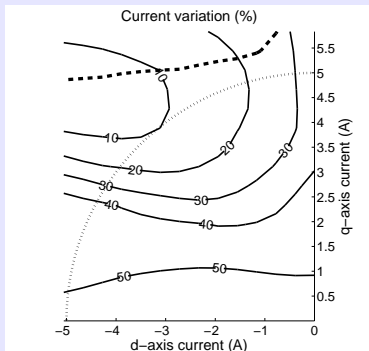
Torque ripple reduction

## Others

## Prediction



Contour map of the angle error in the current signal (inset motor).



$$\Delta I_{\%} = 100 \frac{I_{max} - I_{min}}{I_{max} + I_{min}}$$

## FEM introduction

Meshing and solving  
Post-processing  
Use of the Lua script

## PM motor intro

PMs  
References

## Magnetic Analysis

No load operation  
Operation under load

## Torque

Torque ripple reduction

## Others

*Thank you!*

**FEM introduction**

Meshing and solving  
Post-processing  
Use of the Lua script

**PM motor intro**

PMs  
References

**Magnetic Analysis**

No load operation  
Operation under load

**Torque**

Torque ripple reduction

**Others**



## Tutorial

# Electrical machine analysis using free software: **ONELAB – Gmsh – GetDP**

Johan Gyselinck<sup>1</sup> and Ruth V. Sabariego<sup>2</sup>

<sup>1</sup> BEAMS dpt., Université Libre de Bruxelles, Belgium, [Johan.Gyselinck@ulb.ac.be](mailto:Johan.Gyselinck@ulb.ac.be)

<sup>2</sup> Dpt. Electrical Engineering ESAT/Electa, EnergyVille, KU Leuven, Belgium



BRUSSELS  
SCHOOL  
OF **ENGINEERING**



# Content

- who are we?
- Gmsh, GetDP, ONELAB: a quick introduction
- 2D magnetics models of rotating machines  
demo: 3kW squirrel-cage induction motor
- 3D magnetics
- multi-physics examples: heat transfer, elasticity, etc.
- some advanced magnetics features:
  - homogenisation of lamination stacks and windings
  - other eddy-current methods
  - Harmonic Balance
  - hysteresis

**Johan J. C. Gyselinck** obtained his M.Sc. and PhD degree in electromechanical engineering at the Ghent University (Belgium) in 1991 and 2000 respectively. From 2000 till 2004 he was postdoctoral researcher and lecturer at the University of Liège (Belgium). Since 2004 he is professor at the Université Libre de Bruxelles (ULB, Belgium). His main teaching and research activities are situated in the domain of low-frequency numerical magnetics, and electrical machines and drives. The magnetics branch comprises advanced numerical methods for finite-element modeling (material modeling, homogenization of lamination stacks and windings considering eddy-current effects, harmonic balance, etc), using the open-source programs Gmsh and GetDP, and their practical application to electrical machines and other devices. The research on electrical drives (of various types: PMSMs, SRMs, IMs) is focused on control, fault detection, and vibrations and acoustics, and is carried out both via simulation (e.g. with Simulink) and experimentally.

**Ruth V. Sabariego** is associate professor with the Department of Electrical Engineering at the KU Leuven, Belgium, since October 2013. She graduated in Telecommunication Engineering in 1998 at the University of Vigo, Spain. In October 2000, she joined the Department of Electrical Engineering and Computer Science, University of Liège, Belgium, where she received the PhD degree in Applied Sciences in 2004 and stayed as a post-doctoral researcher till September 2013. Her primary area of expertise involves applied mathematics and computational electromagnetics with a current focus on multi-physics and multiscale modelling.

# Quick intro to ONELAB – Gmsh – GetDP

**Gmsh** - <http://gmsh.info/>

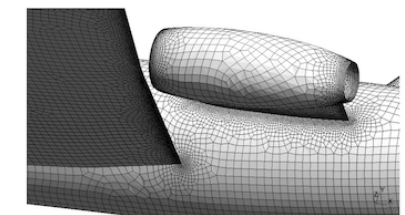
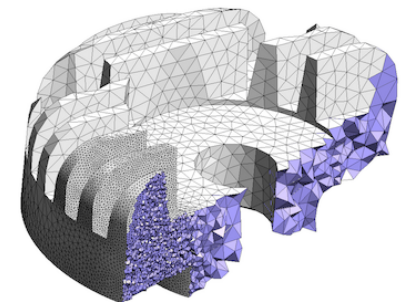
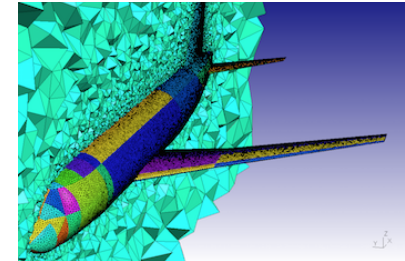
A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities

Christophe Geuzaine and Jean-François Remacle

[Download](#) | [Documentation](#) | [Licensing](#) | [Screenshots](#) | [Links](#) | [References](#) | [!\[\]\(23d9fc146e83b5c3013cfa32c784f8d5\_img.jpg\)](#) | [!\[\]\(f5c463b8c1554ac5049d611bd8e33a51\_img.jpg\)](#)

Gmsh is a free 3D finite element mesh generator with a built-in CAD engine and post-processor. Its design goal is to provide a fast, light and user-friendly meshing tool with parametric input and advanced visualization capabilities. Gmsh is built around four modules: geometry, mesh, solver and post-processing. The specification of any input to these modules is done either interactively using the graphical user interface or in ASCII text files using Gmsh's own scripting language.

See the screencasts for a [quick tour of Gmsh's graphical user interface](#), or the reference manual for a more thorough [overview of Gmsh's capabilities](#) and some [frequently asked questions](#).



# Quick intro to ONELAB – Gmsh – GetDP

**Gmsh** – [gmsh.info/](http://gmsh.info/)

## Download

---

Gmsh is distributed under the terms of the [GNU General Public License \(GPL\)](#):

- **Current stable release (version 3.0.3, June 23 2017):** [Windows \(32 bit\)](#), [Linux](#), [MacOS](#) and [source code](#)

A **tutorial** introducing all key features and concepts is included in all the versions in the *tutorial* directory.

*Make sure to read these examples before sending questions or bug reports!*

- Development version:
  - Automated nightly snapshots ([dashboard](#)): [Windows \(32 bit\)](#), [Linux](#), [MacOS](#) and [source code](#)
  - Git access: `git clone http://gitlab.onelab.info/gmsh/gmsh.git`
- All versions: [binaries](#) and [sources](#)

## Documentation

---

- [Reference manual](#) (also available in [PDF](#) and in [plain text](#))
- [Screencasts](#) showing how to use the graphical user interface
- [Gitlab development site](#) with a [time line](#) of changes and the [bug tracking](#) database
- [Changelog](#)
- Mailing lists:
  - [gmsh](#) (archived [here](#)) is the public mailing list for Gmsh discussions, and is the best place to ask questions (and get answers!)

# Quick intro to ONELAB – Gmsh – GetDP

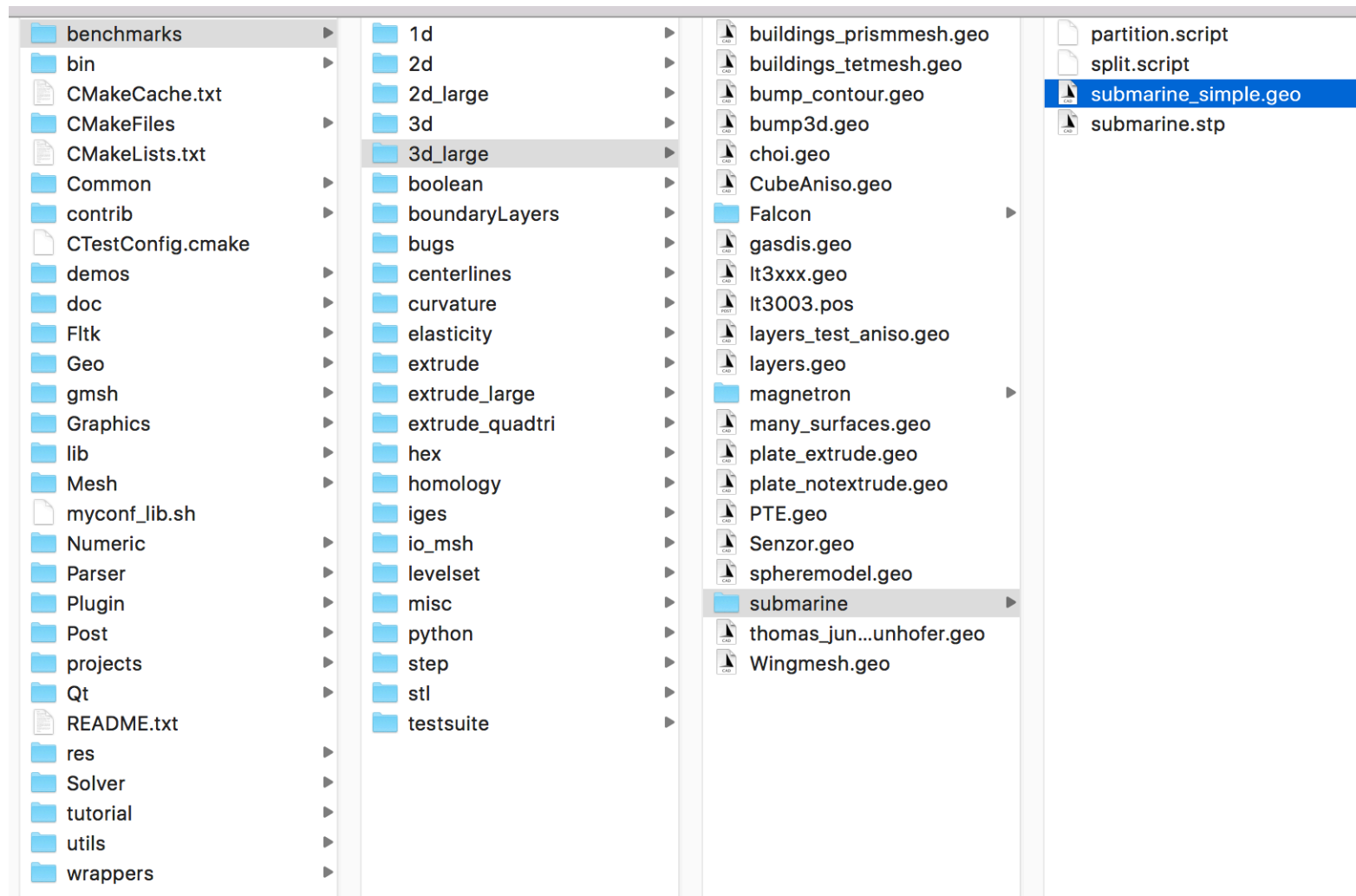
## Gmsh – [.../gmsh.html](#) - What-Gmsh-is-pretty-good-at

Here is a tentative list of what Gmsh does best:

- quickly describe simple and/or “repetitive” geometries, thanks to user-defined macros, loops, conditionals and includes (see [User-defined macros](#), [Loops and conditionals](#), and [General commands](#));
- parametrize these geometries. Gmsh’s scripting language enables all commands and command arguments to depend on previous calculations (see [Expressions](#), and [Geometry commands](#)). Using the OpenCASCADE geometry kernel, Gmsh gives access to all usual constructive solid geometry operations;
- generate 1D, 2D and 3D simplicial (i.e., using line segments, triangles and tetrahedra) finite element meshes for CAD models in their native format (without translations) when linked with the appropriate CAD kernel (see [Mesh module](#));
- specify target element sizes accurately. Gmsh provides several mechanisms to control the size of the elements in the final mesh: through interpolation from sizes specified at geometry points or using flexible mesh size fields (see [Mesh commands](#));
- create simple extruded geometries and meshes (see [Geometry commands](#), and [Mesh commands](#));
- interact with external solvers through a simple client-server architecture (see [Solver module](#));
- visualize and export computational results in a great variety of ways. Gmsh can display scalar, vector and tensor datasets, perform various operations on the resulting post-processing views (see [Post-processing module](#)), can export plots in many different formats (see [General options list](#)), and can generate complex animations (see [General tools](#), and [t8.geo](#));
- run on low end machines and/or machines with no graphical interface. Gmsh can be compiled with or without the GUI (see [Compiling the source code](#)), and all versions can be used either interactively or directly from the command line (see [Running Gmsh on your system](#));
- configure your preferred options. Gmsh has a large number of configuration options that can be set interactively using the GUI, scattered inside command files, changed on the fly in scripts, set in per-user configuration files, or specified on the command-line (see [Running Gmsh on your system](#) and [Options](#));
- and do all the above on various platforms (Windows, Mac and Unix), for free (see [Copying conditions](#)), using simple script files and/or a small but powerful GUI.

# Quick intro to ONELAB – Gmsh – GetDP

## Gmsh – source code, benchmarks, ...





# Quick intro to ONELAB – Gmsh – GetDP

## Gmsh – scripting language, a few typical lines

```

2 Include "npmsm_data.geo";
3
4 Sina = Sin(Pi/Nss) ;
5
6 // parallel side slot opening
7 Ps_6 = newp ; Point(newp) = {ws1, Sqrt(rRext*rRext-ws1*ws1), 0., pslo}; // inner radius
8
9 // lines and arcs
0 Ls_16 = newl ; Line(newl) = {Ps_2,Ps_17};
1
2 Line Loop(newl1) = {Ls_11, Ls_1, -Ls_7, Ls_8, -Ls_4, -Ls_9, Ls_10, -Ls_12, Ls_13, -Ls_2};
3 StatorIron_[] = news; Plane Surface(news) = {newl1-1};
4
5 // mirroring half stator slot contours (before regions!)
6 OuterStator_[] += Symmetry {1,0,0,0,0} { Duplicata{Line{OuterStator_[]}} } ;
7
8 Printf("ggg %g",Stator_MBB_point_[2]);
9
0 If (SymFac > 1)
1   Physical Line("Stator radial boundary (master)",STATOR_RADIAL) = {StatorRadialMaster_[]};
2 EndIf
3
4 Color Gold { Surface{ cond_6[]}; } // B+
5
6 MeshAlgorithm Surface { StatorAirgapLayer_[] } = 1 ; // 2D mesh algorithm (1=MeshAdapt, 2=Automatic, 5=I
7
8 // interesting points on stator side (for some post-processing)
9 If (Nsip)
0   For k In {1:Nsip}
1     PPP = newp ; Point(newp) = {x_sip~{k}, y_sip~{k}, 0., pMB} ;
2     P_sip_[] += PPP ;
3   EndFor
4   Physical Point("Stator-side points",4321) = {P_sip_[]};
5 EndIf

```



# Quick intro to ONELAB – Gmsh – GetDP

**GetDP** – [getdp.info](http://getdp.info)

## A General Environment for the Treatment of Discrete Problems

Patrick Dular and Christophe Geuzaine

[Download](#) | [Documentation](#) | [Licensing](#) | [Links](#) |  | 

GetDP is a free finite element solver using mixed elements to discretize [de Rham-type complexes](#) in one, two and three dimensions. The main feature of GetDP is the closeness between the input data defining discrete problems (written by the user in ASCII data files) and the symbolic mathematical expressions of these problems.

### Licensing

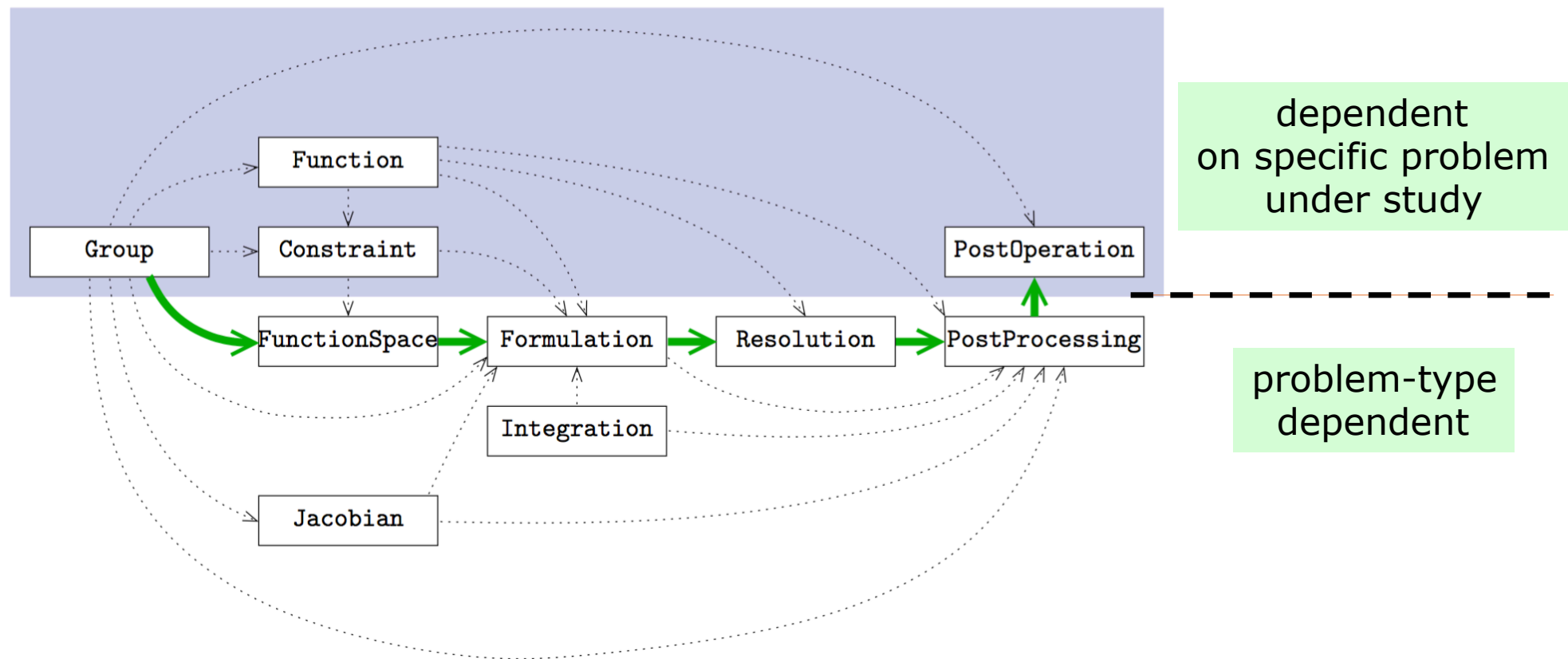
---

GetDP is copyright (C) 1997-2017 by [P. Dular](#) and [C. Geuzaine](#), [University of Liège](#) (see the [CREDITS](#) file for more information), and is distributed under the terms of the [GNU General Public License \(GPL\)](#) (version 2 or later).

# Quick intro to ONELAB – Gmsh – GetDP

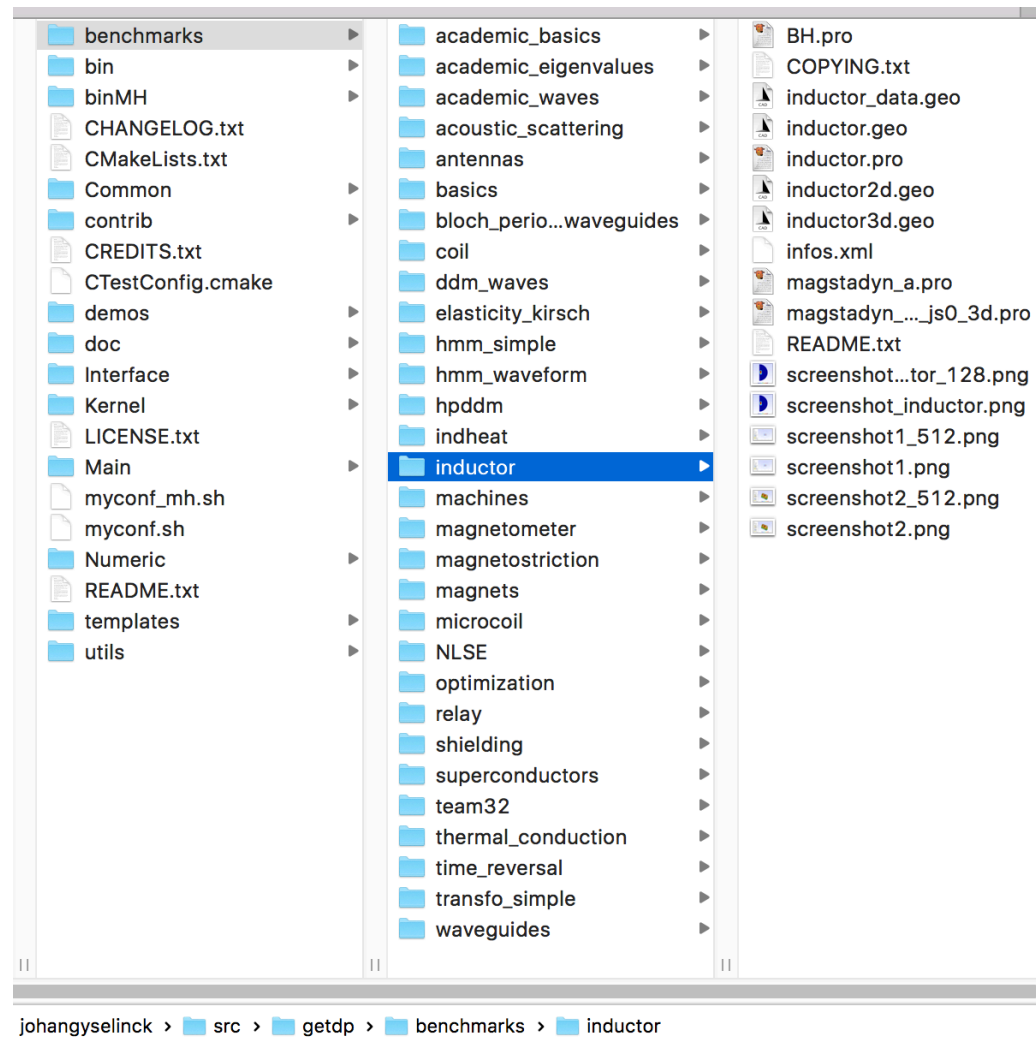
## GetDP – Definition of Discrete Problems

10 objects defined in *text data files* (“*.pro* files”)



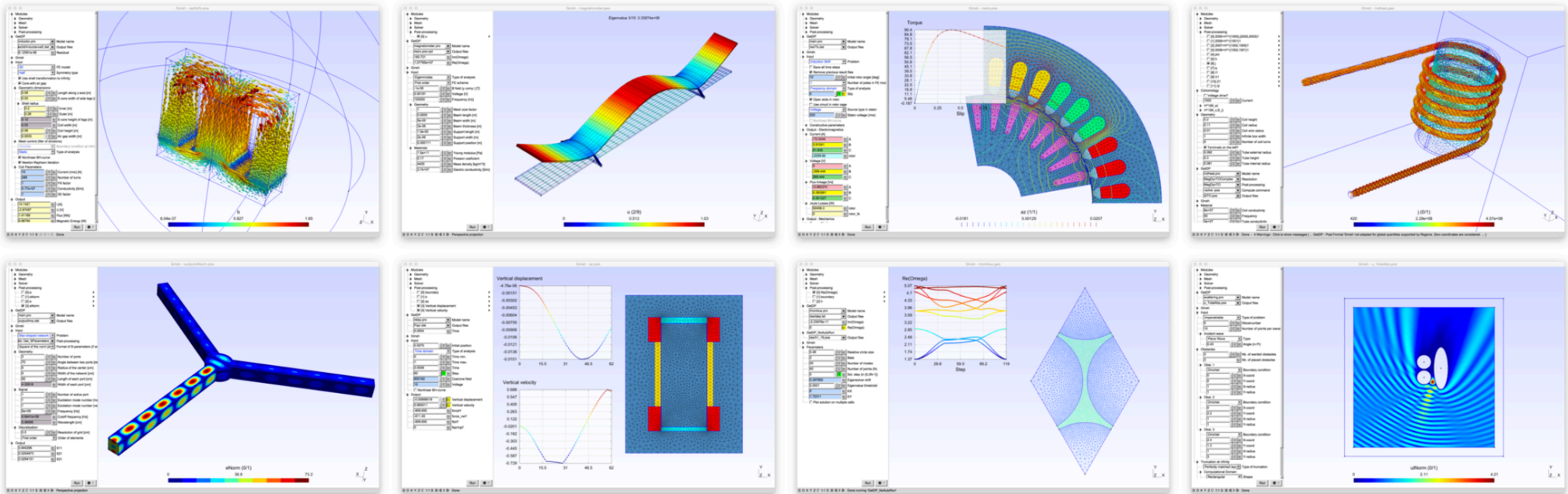
# Quick intro to ONELAB – Gmsh – GetDP

## GetDP – source code



# Quick intro to ONELAB – Gmsh – GetDP

**ONELAB** – [onelab.info](http://onelab.info)



ONELAB (*Open Numerical Engineering LABoratory*) is an open-source, lightweight interface to finite element software. It is completely free: the default ONELAB software bundle contains the mesh generator [Gmsh](#) and the finite element solver [GetDP](#). Many other codes (free or not) can be easily interfaced as well.

# Quick intro to ONELAB – Gmsh – GetDP

**ONELAB** – [onelab.info](http://onelab.info)

## Use existing clients

---

*Native* clients directly embed the ONELAB library:

- **GetDP**: a finite element solver for electromagnetism, heat transfer, acoustics and generic PDEs
- **Gmsh**: a mesh generator and post-processor (Gmsh also plays the role of ONELAB server)
- **Onelab/Mobile**: GetDP and Gmsh on iPhone, iPad and Android devices

Other clients (*non-native*) are interfaced with ONELAB by a system of input file pre-processing. The different steps of a simulation (meshing, solving, post-processing) are controlled by a python script. See worked-out examples with:

- **Elmer**: a finite element solver for multi-physics problems developed by CSC
- **OpenFOAM**: an open source CFD software package developed by OpenCFD

Any software driven by input data files (e.g. Code\_Aster, Abaqus, CalculiX, FreeFem, Gnuplot, ...) can be readily interfaced in the same way.

Here are some [useful hints to efficiently use the ONELAB graphical user interface](#).

# ICEM 2016 tutorial, 4 September 2016

<http://w1.icem.cc/2016/index.php/tutorials>

## Finite-Element and Lumped-Parameter Modelling and Simulation of Permanent-Magnet Synchronous Machines

### From Academic State-of-the-Art to Design-Office Practice

**Johan Gyselinck <sup>1</sup> and Adrian-C. Pop <sup>2</sup>**

<sup>1</sup> BEAMS dpt, Université Libre de Bruxelles,  
Brussels, Belgium, [Johan.Gyselinck@ulb.ac.be](mailto:Johan.Gyselinck@ulb.ac.be)

<sup>2</sup> Advanced Development Drives Group, Brose,  
Würzburg, Germany, [AdrianCornel.Pop@brose.com](mailto:AdrianCornel.Pop@brose.com)

ICEM 2016, Lausanne, Switzerland, September 4-7



2D FE modelling of PMSMs  
with ONELAB – Gmsh – GetDP

2D & 3D FE modelling  
with Ansys/Maxwell

lumped-parameter modelling  
with MATLAB/Simulink

# ICEM 2016 tutorial

[w1.icem.cc/2016/images/doc/Gyselinck\\_Pop.zip](http://w1.icem.cc/2016/images/doc/Gyselinck_Pop.zip)

- the presenters  
Johan Gyselinck, Gmsh & GetDP & ONELAB  
Adrian-C. Pop, Brose
- introduction to PMSMs
- specifications and topology selection
- 2D magnetostatic FE modelling

11:15 – 11:25

- optimisation and utility of 3D FEA
- 2D FE modelling with eddy currents (along z)
- lumped-parameter modelling and control
- lamination stacks and windings (proximity effect)
- conclusions

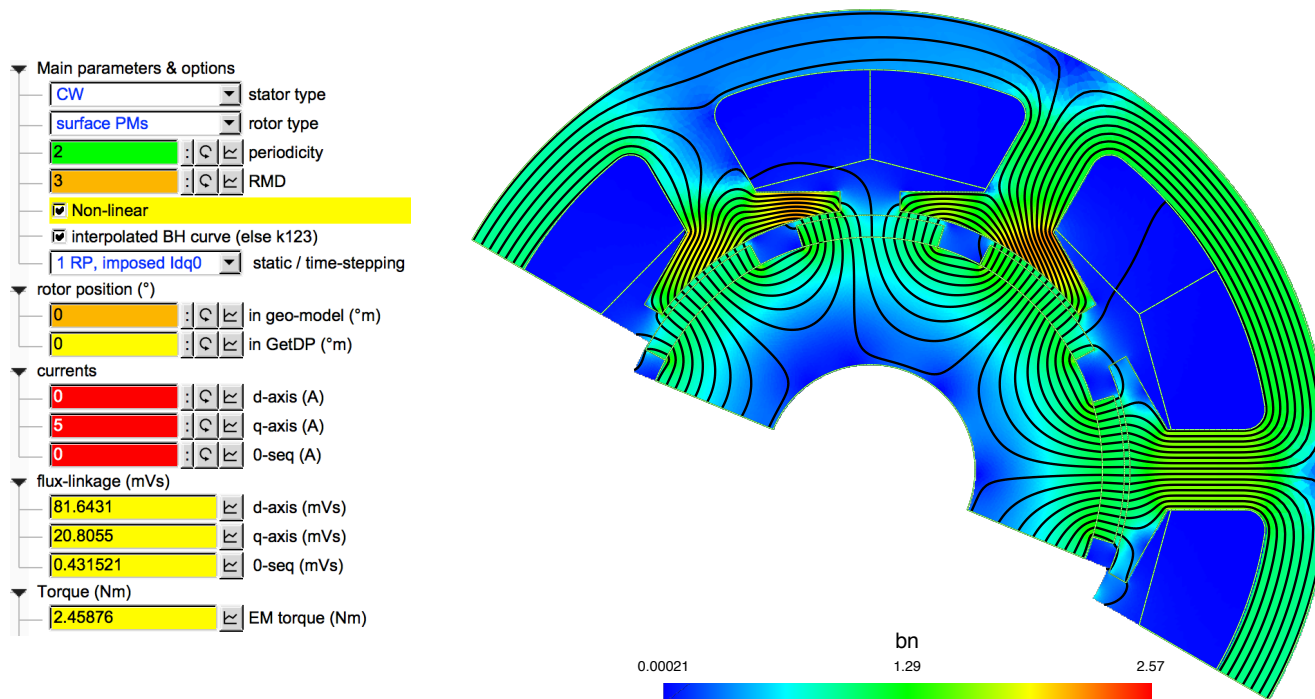
standard and  
non-standard features

# ICEM 2016 tutorial

**Just 1 slide**

## 2D static FE modelling – Our CW machine (2/5)

**Flux lines and flux density, load**



J. Gyselinck & A.-C. Pop, ICEM, Lausanne, Switzerland, 4-7 September 2016  
Tutorial: Finite-element and lumped-parameter modelling and simulation of PMSMs

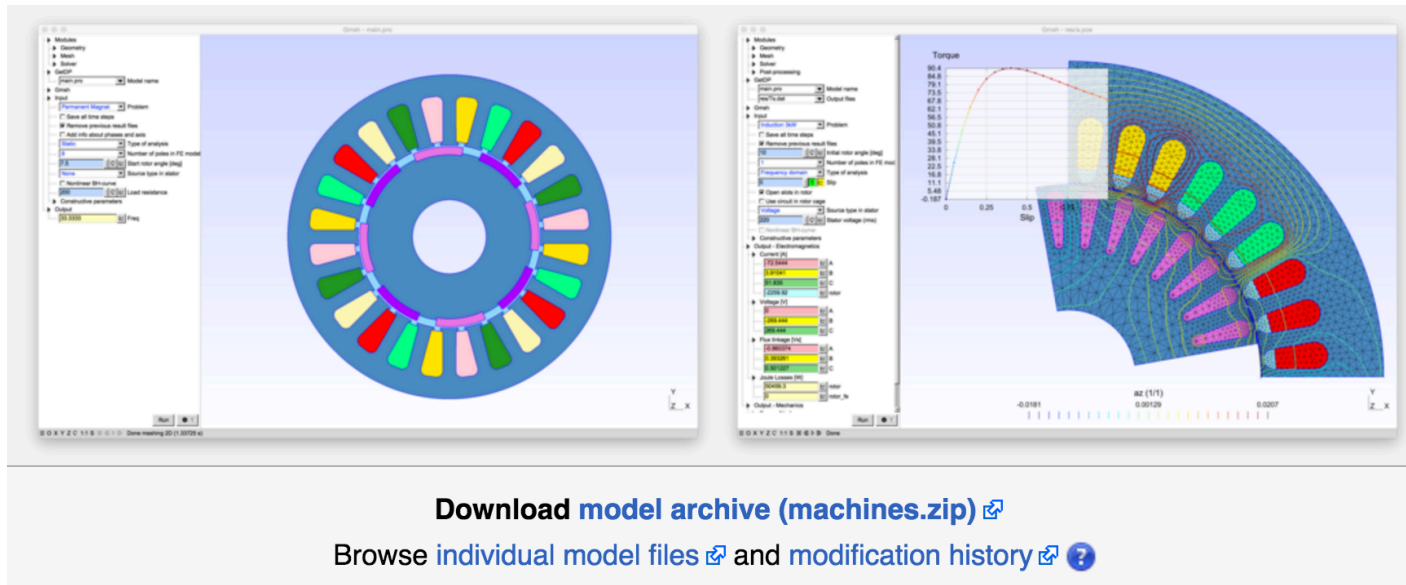
38

**ICEM**  
International Conference on Electrical Machines



# 2D magnetics modelling of electric machines

[http://onelab.info/wiki/Electric\\_machines](http://onelab.info/wiki/Electric_machines)



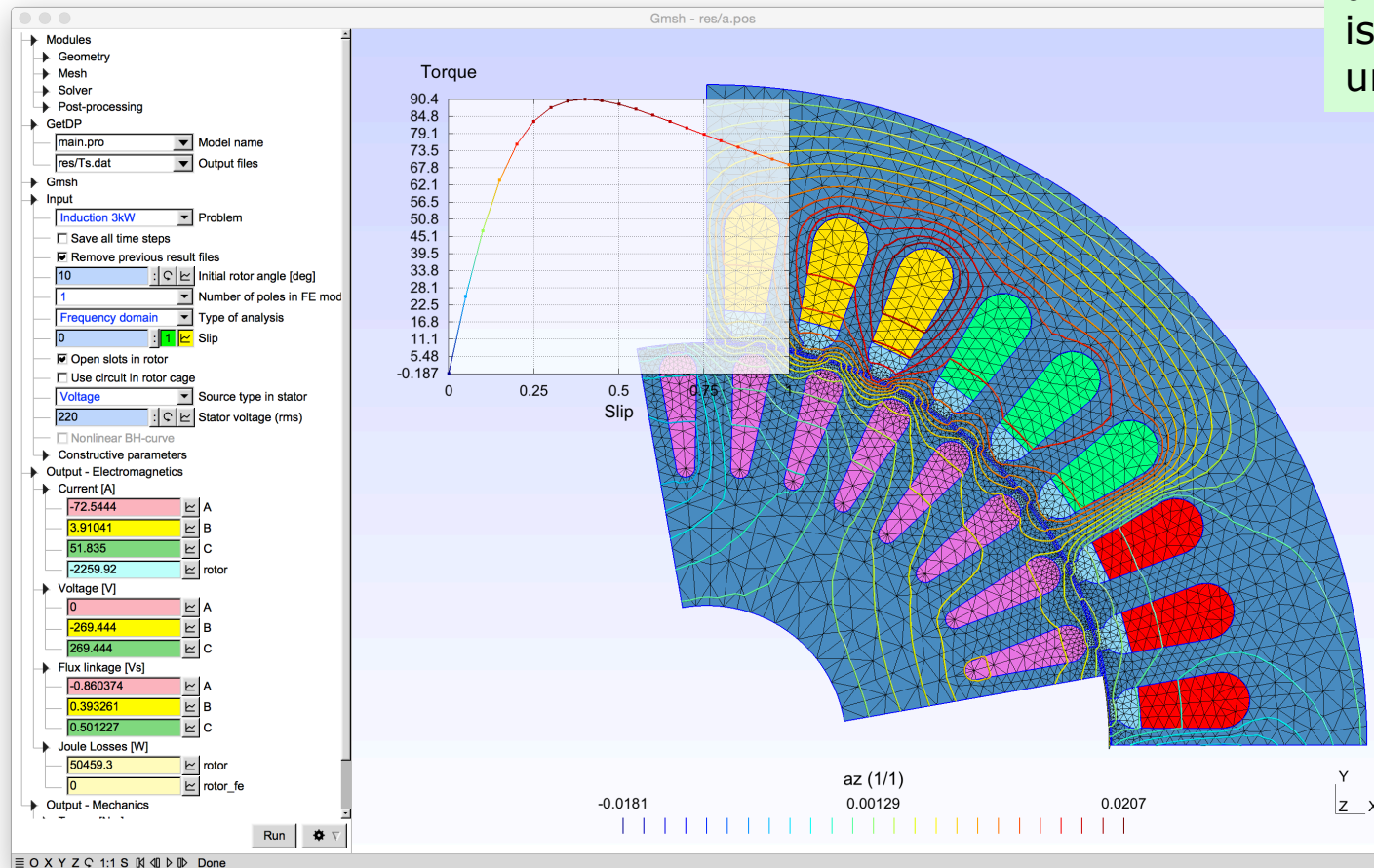
common features

- 2D magnetic vector potential formulation
- rotation with moving band
- iron core modelling: isotropic, nonlinear
- stranded and massive conductors
- electric circuit coupling
- (anti-) periodicity conditions for modelling 1 pole or pole pair
- static, time stepping, and frequency domain (no saturation)

# 2D magnetics modelling of electric machines

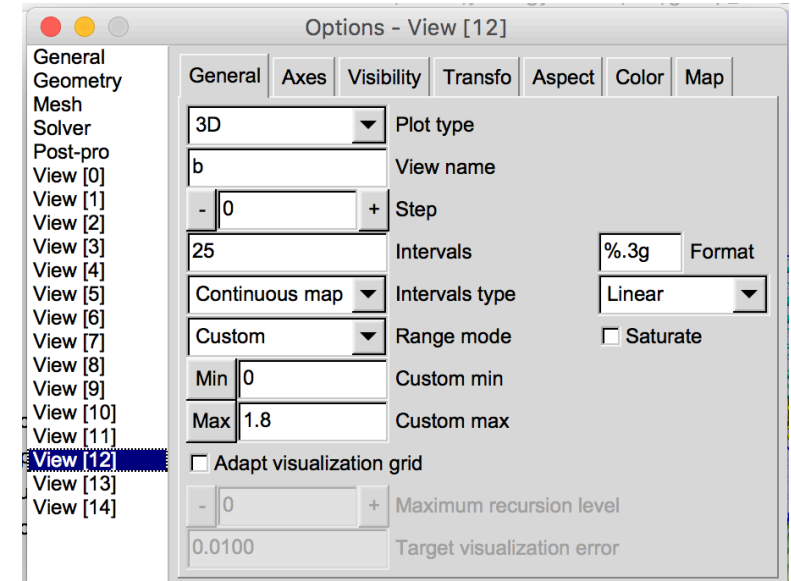
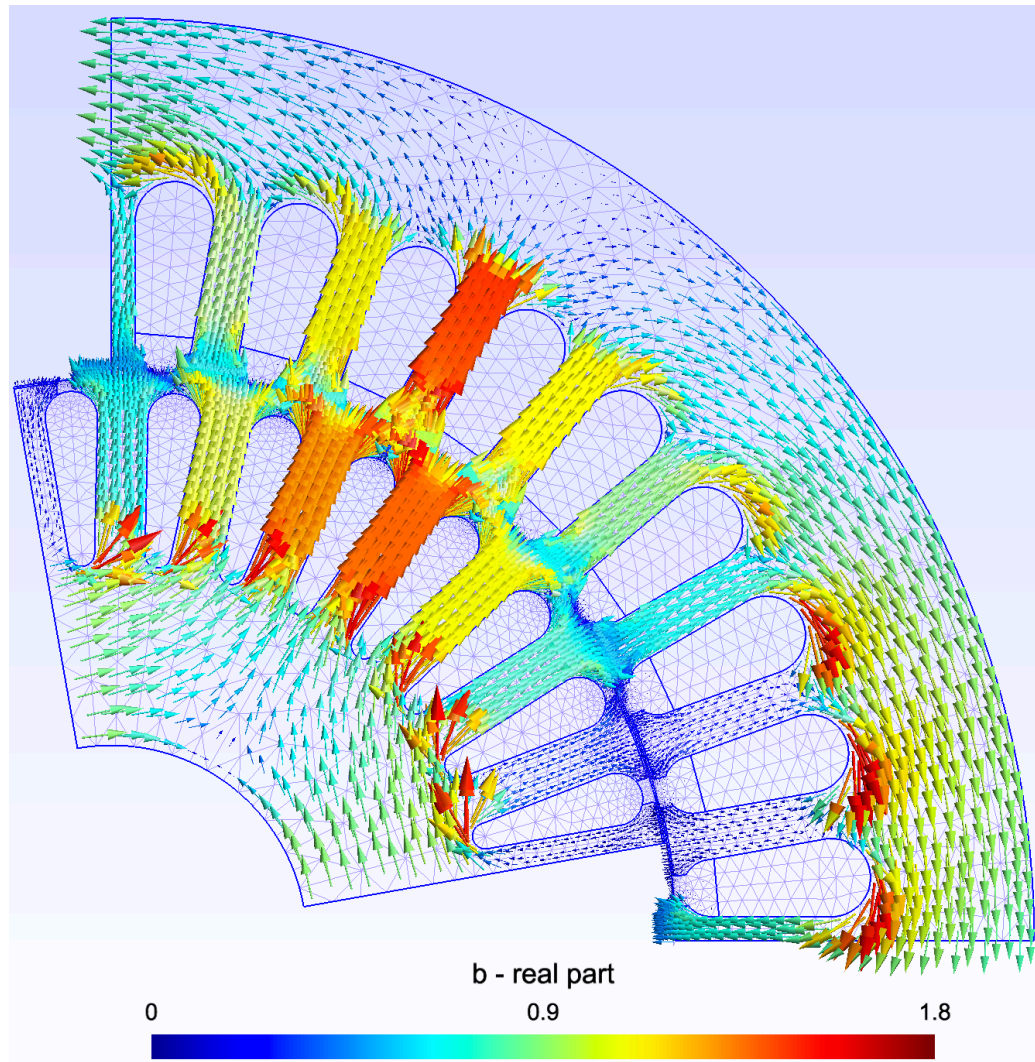
## Demo: 3kW squirrel-cage induction motor

an extended version  
is presently  
under construction



# 2D magnetics modelling of electric machines

## Demo: 3kW squirrel-cage induction motor



# 3D magnetics model of machines

## Particular issues – movement

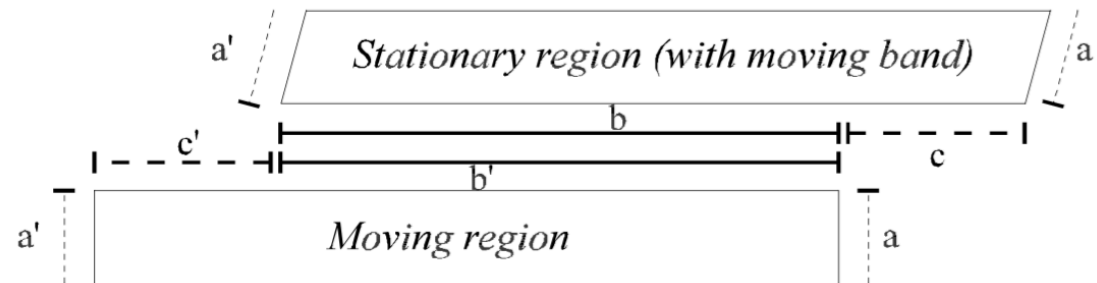
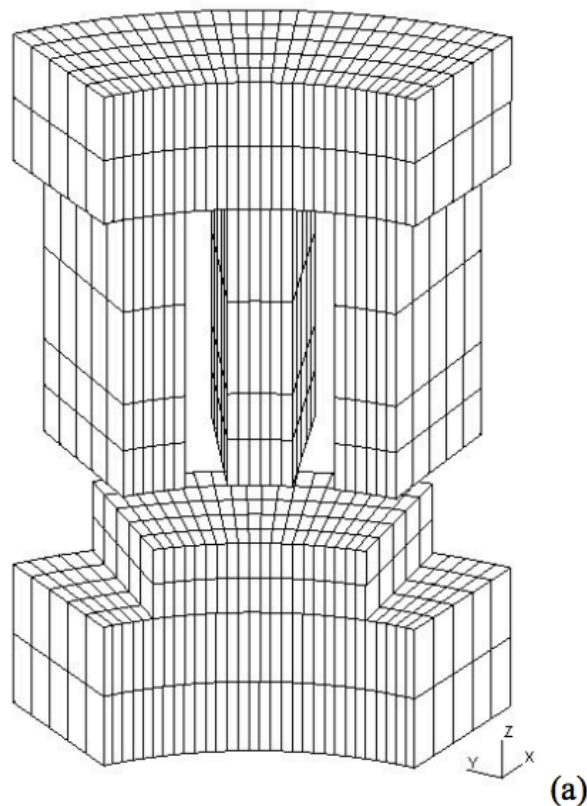
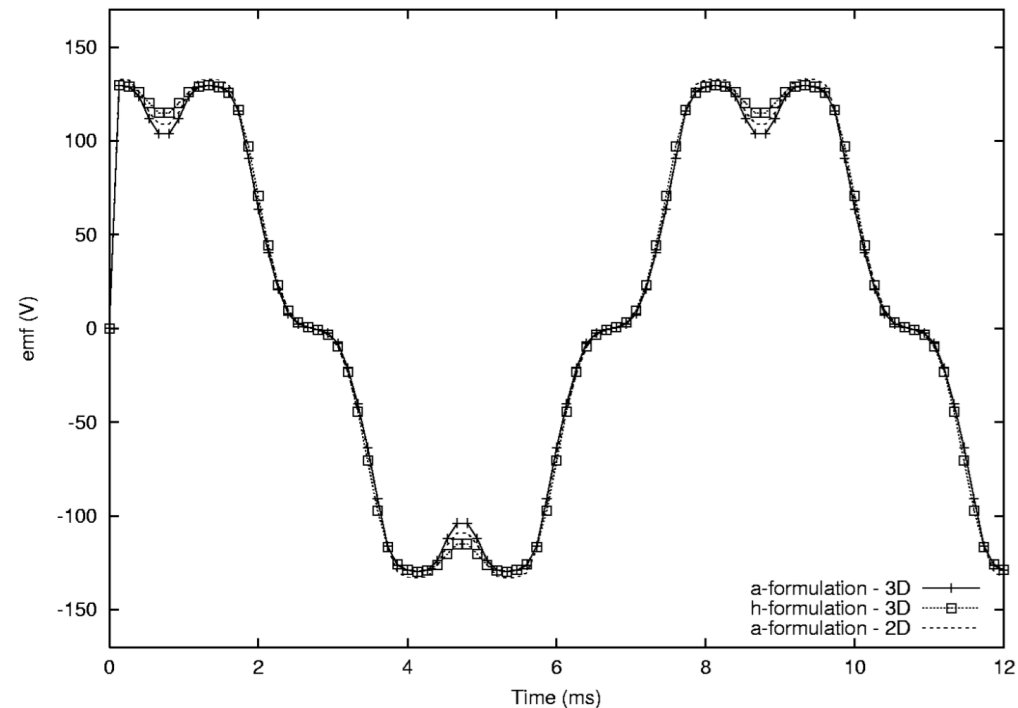
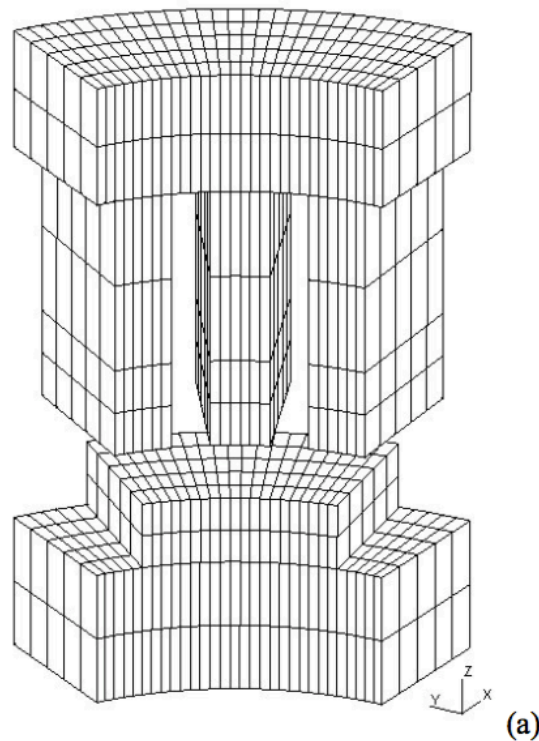


Fig. 1. Connection boundary conditions with the moving band.

Ferreira da Luz, M. V., Dular, P., Sadowski, N., Gyselinck, J., & Bastos, J. P. A. (2002). 3D finite element analysis of axial flux permanent magnet motor. In Proceedings of the 10th International IGTE Symposium on Numerical Field Calculation in Electrical Engineering.

# 3D magnetics model of machines

## Particular issues – formulations



Ferreira da Luz, M. V., Dular, P., Sadowski, N., Gyselinck, J., & Bastos, J. P. A. (2002). 3D finite element analysis of axial flux permanent magnet motor. In Proceedings of the 10th International IGTE Symposium on Numerical Field Calculation in Electrical Engineering.



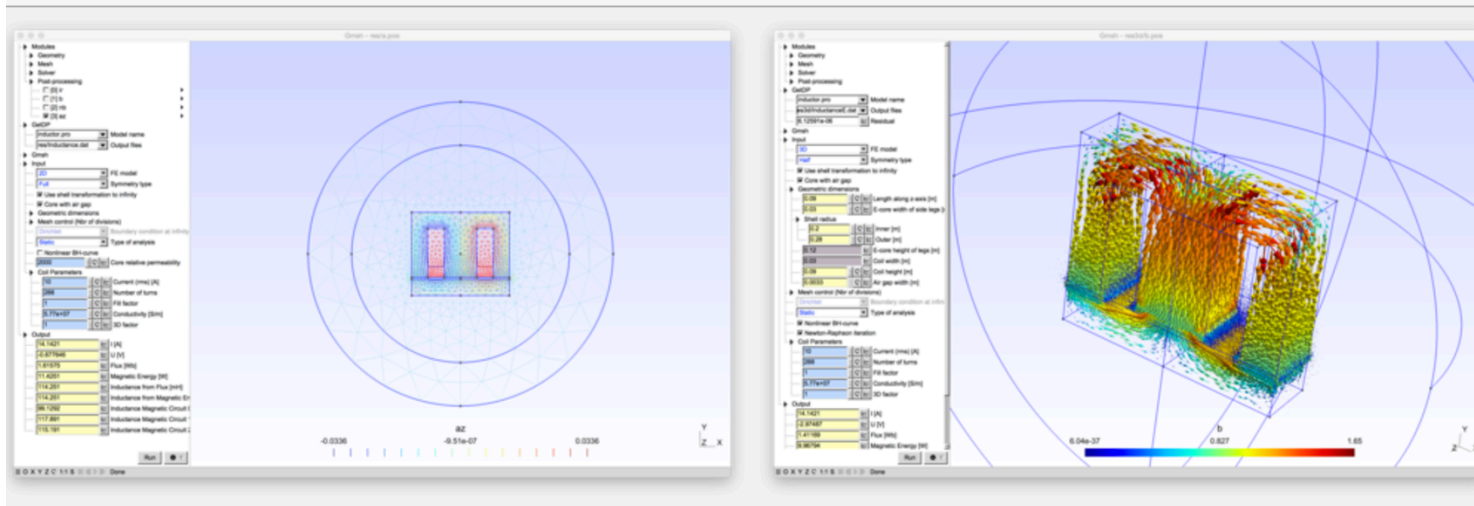
# 3D magnetics model of machines

## Particular issues – a-formulation and 3D gauging

<http://onelab.info/wiki/Inductor>

- tree-cotree gauging
- Coulomb gauge

2D and 3D model of an inductor/core system.



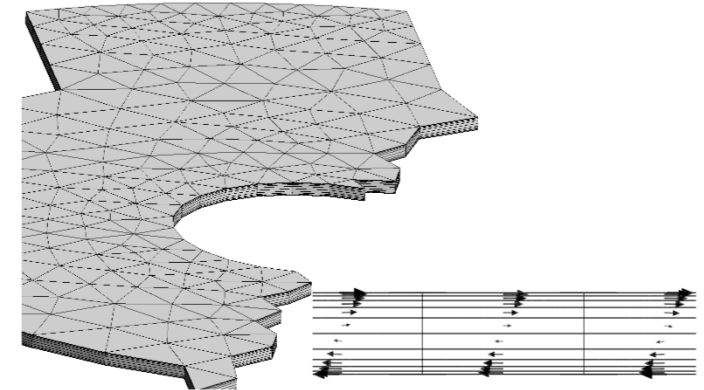
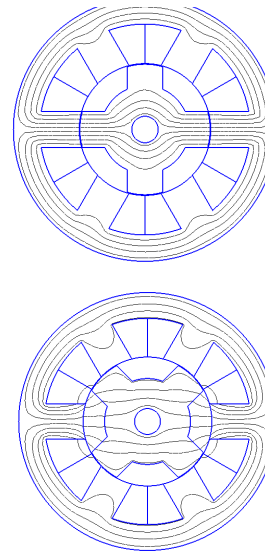
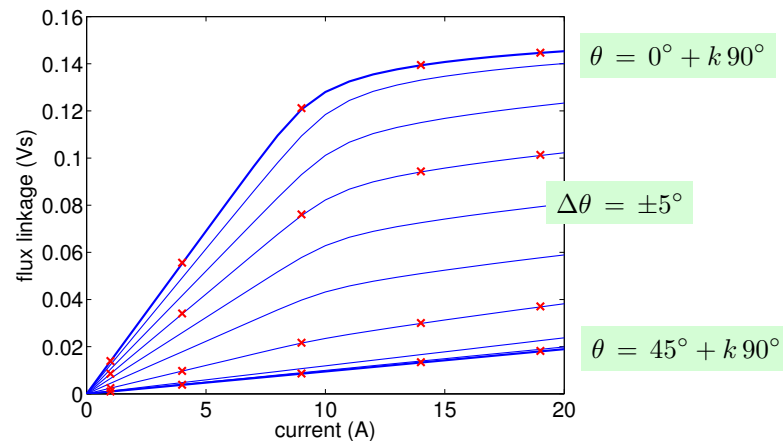
Download [model archive \(inductor.zip\)](#)

Browse [individual model files](#) and [modification history](#)

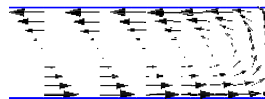
# 3D magnetics model of machines

## 6/4 SRM: 1 stator and 1 rotor lamination

static flux-current-position characteristics



**X** : 3D FE model by extruding 2D model



- 2, 4, 6 or 8 layers of prismatic elements for considering eddy currents
- MVP formulation, circuit coupling with imposed voltage or current
- 66882, 123008, 169134 or 205788 real-valued or complex-valued unknowns

consideration of eddy currents (and skin effect)

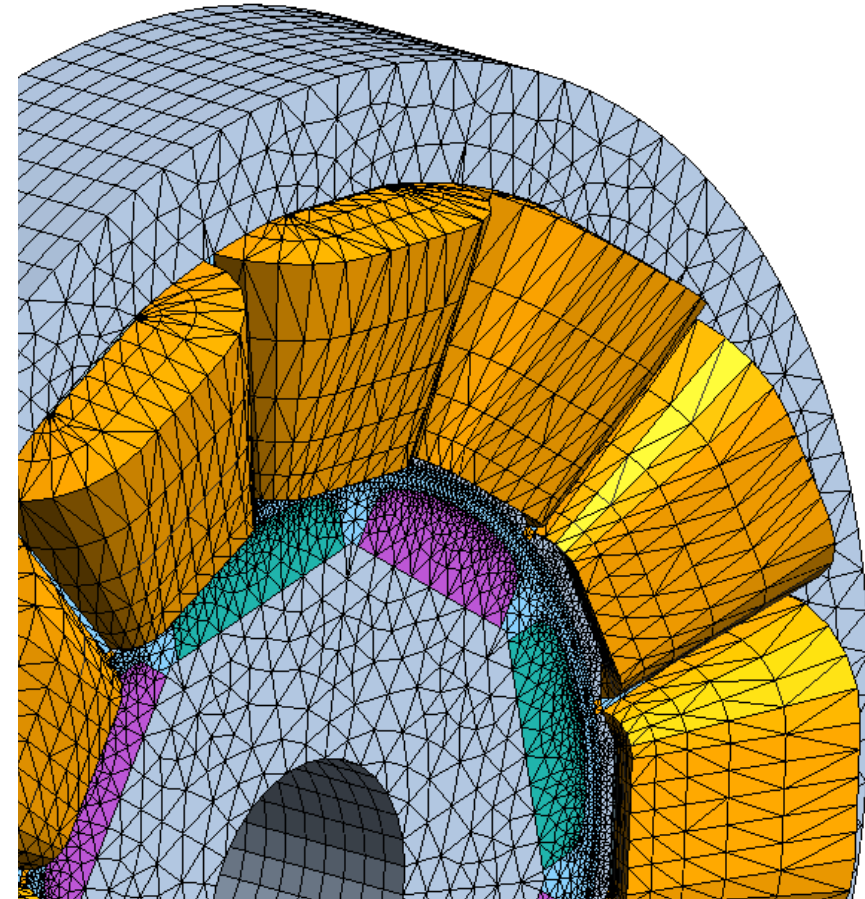
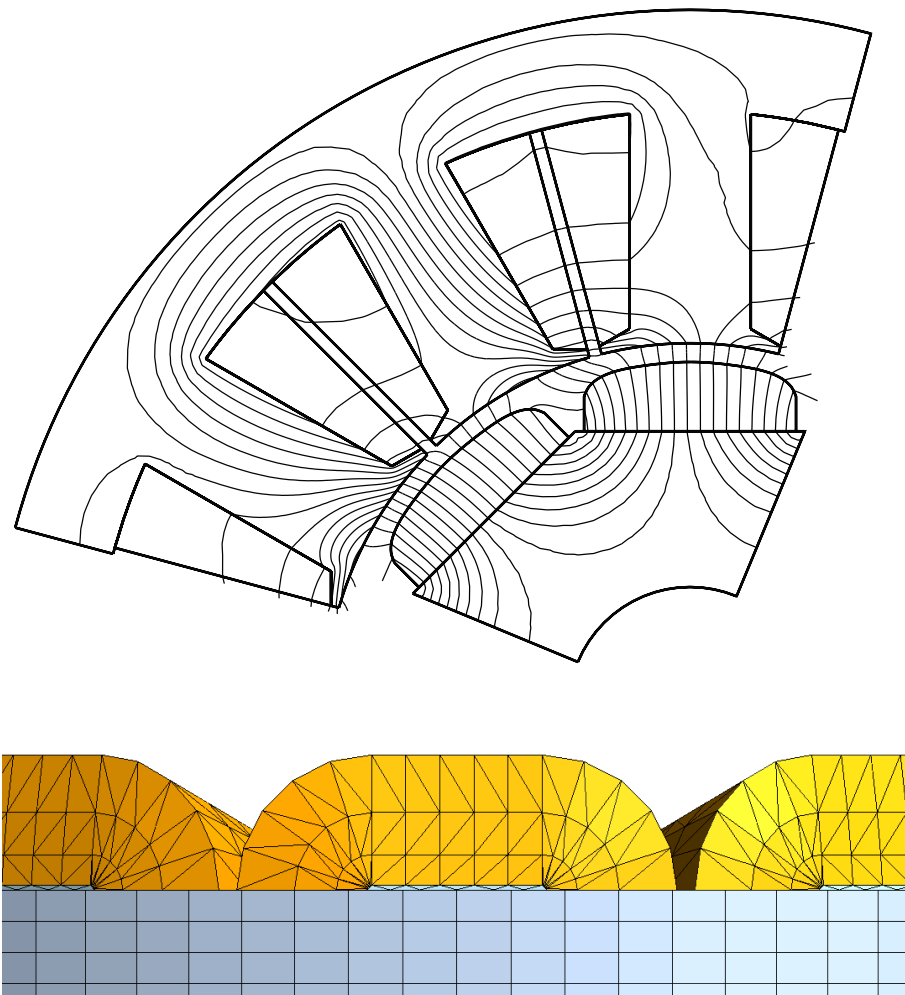
- 3D model
- 2D model and homogenisation technique

Gyselinck, J., Geuzaine, C., & Sabariego, R, Homogenisation of windings and laminations in time-domain finite-element modeling of electrical machines, CEFC 2012.

[CEFC2012 SRM JG FINAL 1p.pdf](#)

# 3D magnetics model of machines

## 12/8 Surface PMSM with concentrated winding



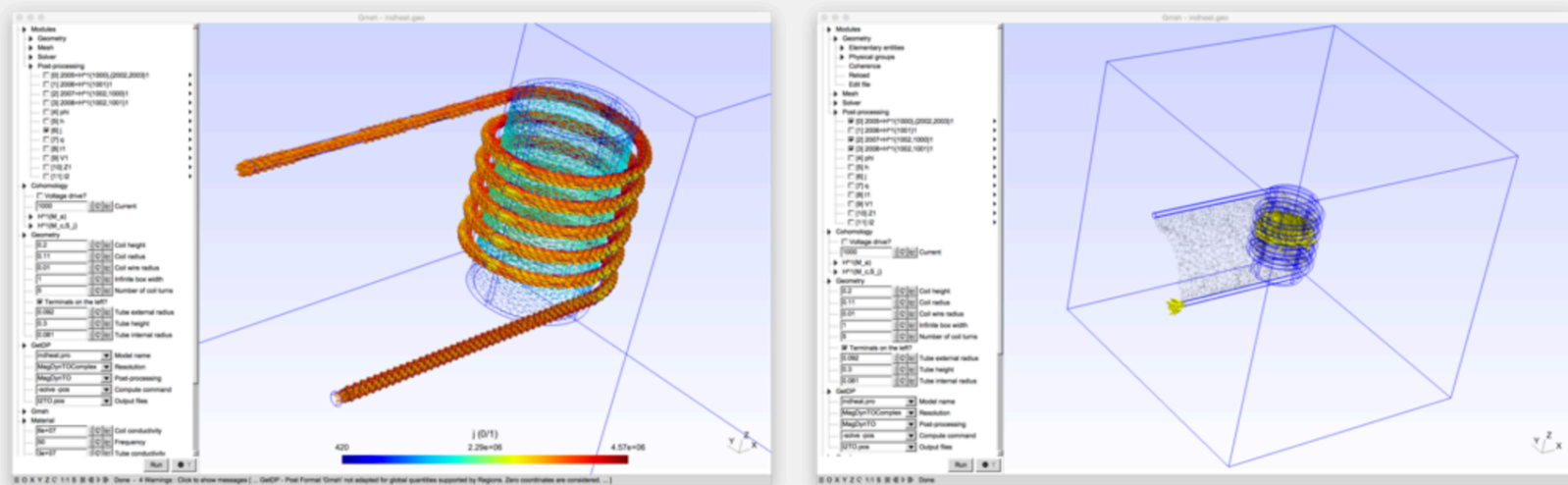
courtesy of Diogo Pinto (Brose/FHWS/ULB)



# 3D magnetics & multi-physics

[onelab.info/wiki/Magnetodynamics\\_with\\_cohomology\\_conditions](http://onelab.info/wiki/Magnetodynamics_with_cohomology_conditions) (1/1)

Magneto-thermal model of an induction heating device.



Download [model archive \(indheat.zip\)](#)

Browse [individual model files](#) and [modification history](#)

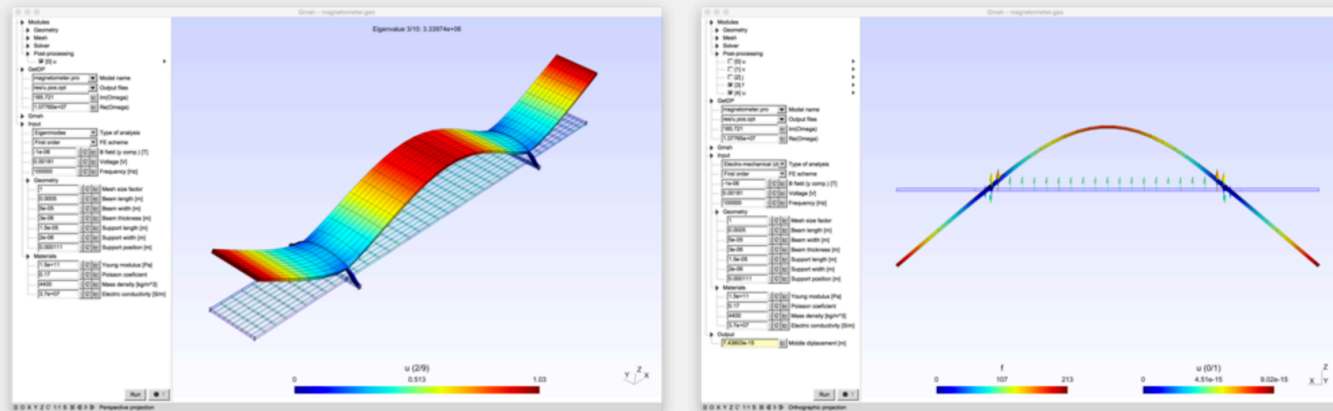
$T - \Omega$  potential formulation

$$\mathbf{h} = \sum_{i \in N(M_a)} \phi_i \mathbf{grad} \, n_i + \sum_{i \in E(M_c \setminus (\partial M_a \cap \partial M_c))} t_i \mathbf{e}_i + I_1 \mathbf{E}_1 + I_2 \mathbf{E}_2$$

# Multi-physics: some examples

<http://onelab.info/wiki/Magnetometer> (1/4)

Xylophone bar magnetometer.



Download [model archive \(magnetometer.zip\)](#)

Browse [individual model files](#) and [modification history](#)

## Additional information

This is a 3D, multiphysics model of a MEMS magnetometer, coupling electromagnetic (electrokinetic), mechanical (elastic) and thermal models. To run the model, open **magnetometer.pro** with Gmsh.

When an alternating current flows through it, the magnetometer starts to vibrate if it is placed in a magnetic field. The amplitude of the vibration can be used to measure the magnetic field. To increase sensitivity, the frequency of the current is tuned to excite a resonant mode of the xylophone-type structure. Several analyses are predefined: uncoupled modal analysis of the structure (**Eigenmodes**) and calculation of the current flow (**Electrokinetics**), static and dynamic electro-mechanical analysis (**Electro-mechanical**), and coupled electro-thermal analysis (**Electro-thermal**).

# Multi-physics: some examples

## Magnetometer – electrokinetics and elasticity (2/4)

```

34 Include "Electrokinetics.pro"
35 Include "Elasticity.pro"
36 Include "Thermal.pro"
37
38 Resolution {
39   { Name Analysis ;
40     System {
41 ...
42     If(Flag_AnalysisType == 3)
43       { Name Sys_EleKin; NameOfFormulation Electrokinetics_v; }
44       { Name Sys_Mec;   NameOfFormulation Elasticity3D_u_coupled_transient; Type Complex; Frequency Freq;}
45     EndIf
46 ...
47   }
48   Operation {
49     CreateDir["res/"];
50     If(DEGRE2)
51       SetGlobalSolverOptions["-petsc_prealloc 400"];
52     EndIf
53 ...
54     If(Flag_AnalysisType == 2 || Flag_AnalysisType == 3)
55       Generate[Sys_EleKin]; Solve[Sys_EleKin]; SaveSolution[Sys_EleKin];
56       Generate[Sys_Mec];   Solve[Sys_Mec];   SaveSolution[Sys_Mec];
57       PostOperation[EleKin];
58       PostOperation[Mec];
59     EndIf
60 ...
61   }
62 }
63 }
64

```

# Multi-physics: some examples

## Magnetometer – heat transfer eqn (3/4)

```

22 { Name TheDyn; Type FemEquation;
23   Quantity {
24     { Name t; Type Local; NameOfSpace TSpace; } // temperature field, unknown
25     { Name h; Type Local; NameOfSpace HSpace; } // magnetic field, known
26   }
27   Equation {
28     // conduction and heat capacity term
29     Galerkin { [ k * Dof{Grad t} , {d t} ] ; In Omega_c2; Integration Int; Jacobian Vol; }
30     Galerkin { DtDof [ rho * c * Dof{t} , {t} ] ; In Omega_c2; Integration Int; Jacobian Vol; }
31
32     // convection BC
33     Galerkin { [ h * Dof{t} , {t} ] ; In BdOmega_c2; Jacobian Sur ; Integration Int ; }
34     Galerkin { [ -h * AmbT , {t} ] ; In BdOmega_c2; Jacobian Sur ; Integration Int ; }
35
36     // radiation BC
37     Galerkin { [ sigma_sb*ep * ({t})^4 , {t} ] ; In BdOmega_c2; Jacobian Sur ; Integration Int ; }
38     Galerkin { [ -sigma_sb*ep * (AmbT)^4 , {t} ] ; In BdOmega_c2; Jacobian Sur ; Integration Int ; }
39
40     // source heat due to eddy currents
41     Galerkin { [ -1./sigma * ( Re[{Curl h}]*Re[{Curl h}] + Im[{Curl h}]*Im[{Curl h}]), {t} ] ;
42       In Omega_c2; Integration Int; Jacobian Vol; }
43   }
44 }

```

# Multi-physics: some examples

## Magnetometer: electrokinetics and heat transfer (4/4)

```

70 Resolution {¶
71   { Name TheDyn;¶
72     System {¶
73       { Name B; NameOfFormulation TheDyn; }¶
74       { Name A; NameOfFormulation MagDynT0; Type ComplexValue; Frequency Freq;}¶
75     }¶
76     Operation {¶
77       // magnetics eddy-current problem (linear, frequency domain)¶
78       InitSolution[A]; Generate[A]; Solve[A]; SaveSolution[A];¶
79       ¶
80       // heat transfer (nonlinear, time-stepping)¶
81       InitSolution[B];¶
82       TimeLoopTheta[time0t, time1t, dtimet[], theta] {¶
83         IterativeLoop[NL_NbrMax, NL_Eps, NL_Relax] {¶
84           GenerateJac[B]; SolveJac[B];¶
85         }¶
86         SaveSolution[B];¶
87       }¶
88     }¶
89   }¶
90 }¶

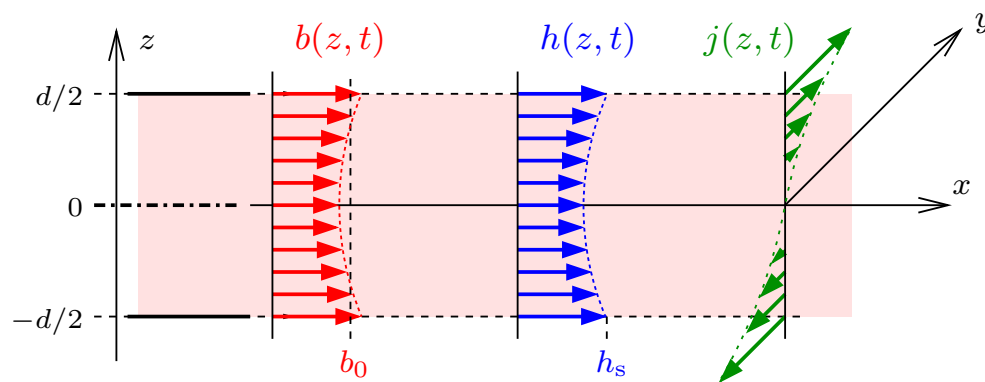
```

# Some advanced magnetics features

## Homogenisation of lamination stacks

see also the  
ICEM2016 tutorial

Mostly: 1D eddy-current (EC) problem



variation with  $z$  only

- ignoring the U-turns of the ECs
- assuming width  $\gg$  thickness
- zero net current

$$\frac{\partial^2 h}{\partial z^2} = \sigma \frac{\partial b}{\partial t}, \quad h = h(b)$$

$$-d/2 \leq z \leq d/2$$

### different cases:

- |                 |   |  |
|-----------------|---|--|
| rather trivial  | { | 1. no saturation, sinusoidal regime at any frequency |
|                 |   | 2. low frequency (no skin effect) but saturation     |
| skin-effect BFs | ← | 3. saturation and skin effect                        |
| [IEEE2016]      | ← | 4. insulation faults and net current                 |
|                 |   | 5. perpendicular flux density and net current        |

# Some advanced magnetics features

## Homogenisation of lamination stacks

### 1. sinusoidal regime, no saturation

complex permeability,

link between

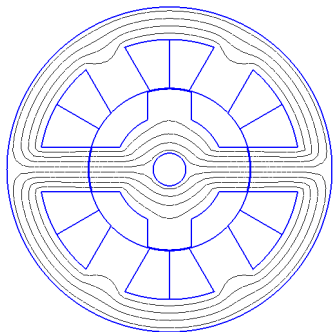
- average flux-density
- surface magnetic field

$$\bar{\mu} = \frac{\bar{b}_0}{\bar{h}_s} = \mu_{dc} \tanh\left(\frac{1-i}{2} \frac{d}{\delta}\right)$$

$\mu_{dc}$  : in absence of ECs

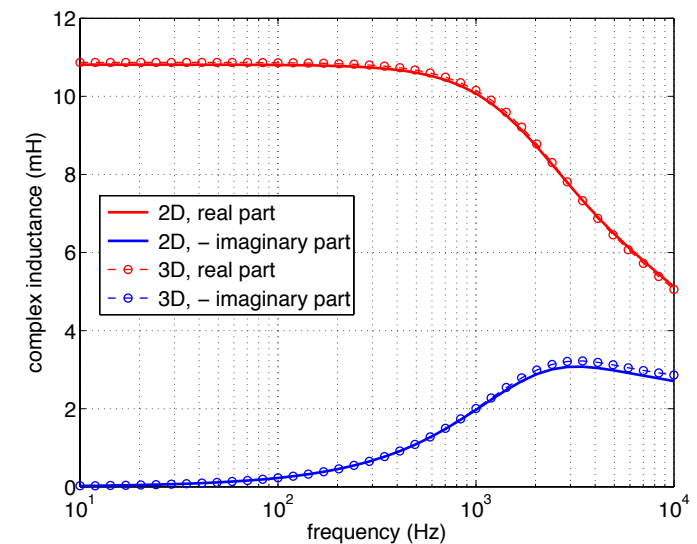
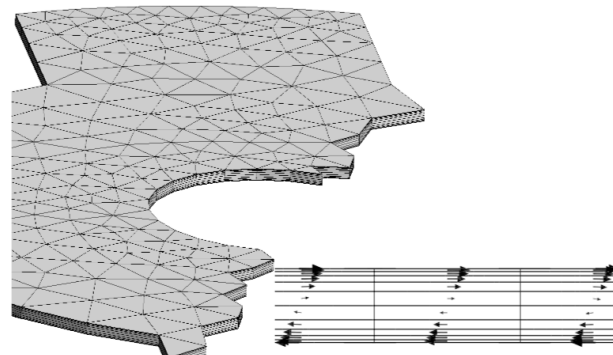
$$\delta = \sqrt{\frac{2}{\omega \sigma \mu_r \mu_0}}$$

6/4 SRM  
[CEFC2012]



consideration of ECs and skin effect

- 3D model (1 lamination)
- 2D model and complex  $\mu$



# Some advanced magnetics features

## Homogenisation of lamination stacks

2. low frequency (no skin effect), but saturation

classical EC losses:

$$p_{cl} = \frac{\sigma d^2}{12} \left( \frac{db}{dt} \right)^2 \quad (\text{W/m}^3)$$

- instantaneous
- averaged over lam. thickness

$$h_s(t) = \underbrace{h_{dc}(b(t))}_{\text{without ECs}} + \frac{\sigma d^2}{12} \frac{db}{dt}$$

- without ECs
- linear or nonlinear, possibly hysteretic and anisotropic

very simple implementation, e.g. in GetDP:

```

If (Flag_Lam)
  Galerkin { DtDof [ Fac_Lam[] * Dof{d a} , {d a} ] ; In Domain_Lam ; Jacobian Vol ; Integration I1 ; }
EndIf
  
```

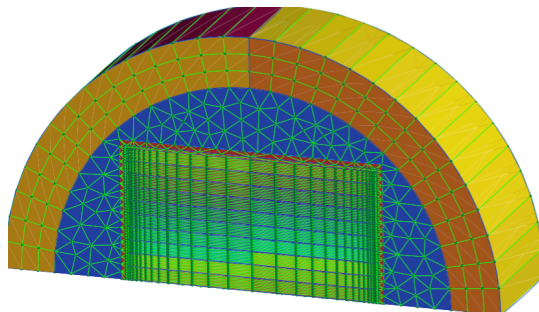


# Some advanced magnetics features

## Homogenisation of lamination stacks

### 3. saturation and skin effect (skin-effect BFs)

skin-effect BFs for approximating  
significant variation of  $\mathbf{b}$  throughout lam. thickness  
more BFs: higher accuracy and higher CPU time

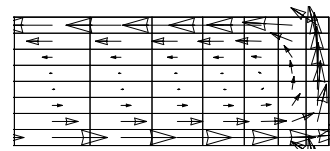


brute-force model

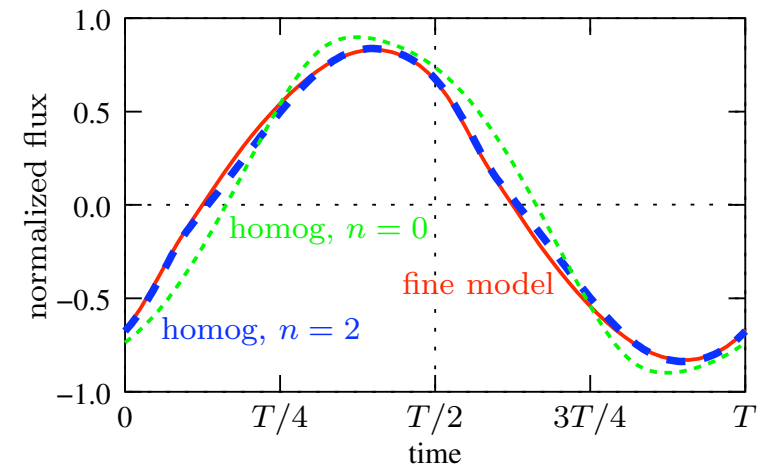
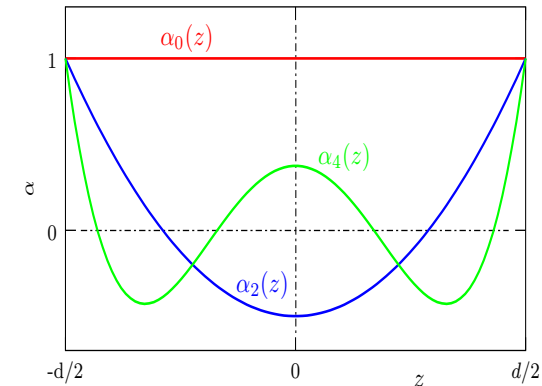
1/128<sup>th</sup> part  
of ring core

2 x 10 lam.

8 layers per  
lamination



500 Hz current

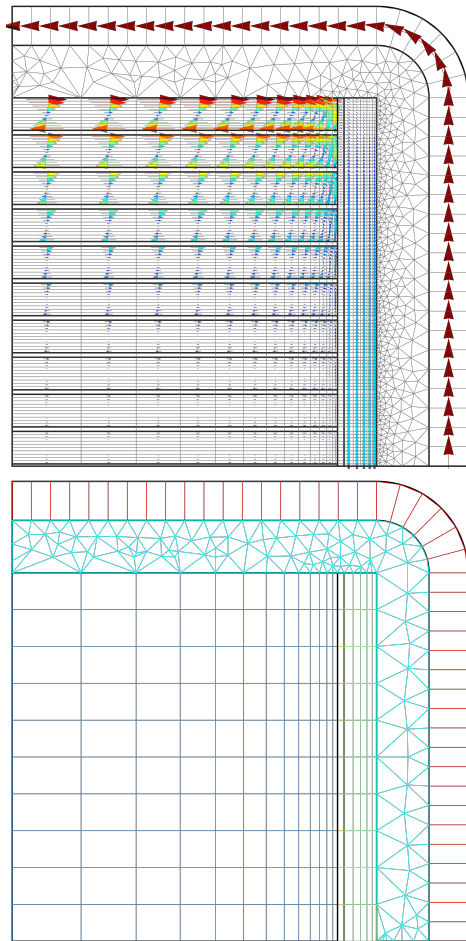


Gyselinck, Johan, Ruth Sabariego, and Patrick Dular. "A nonlinear time-domain homogenization technique for laminated iron cores in three-dimensional finite-element models." IEEE transactions on magnetics 42.4 (2006): 763-766.

# Some advanced magnetics features

## Homogenisation of lamination stacks

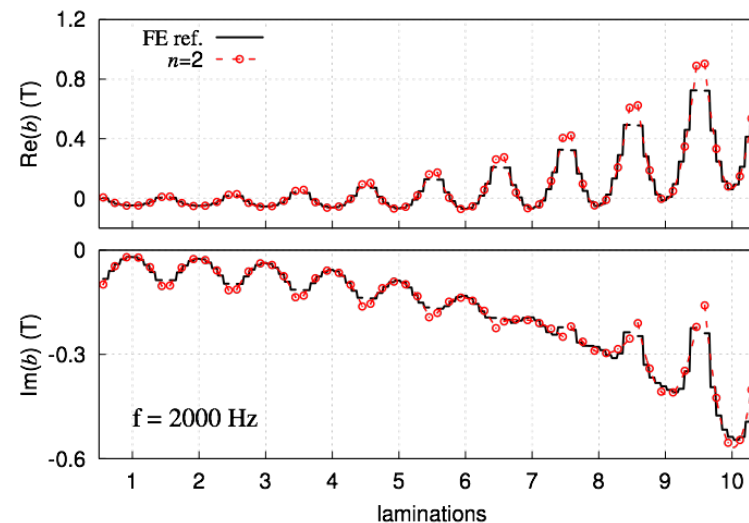
### 4. insulation faults and net current



2D test case with **in-plane current density** and **out-of-plane flux density**

- skin effect in each lamination
- overall skin effect in stack due to insulation imperfection

results  
at 2kHz

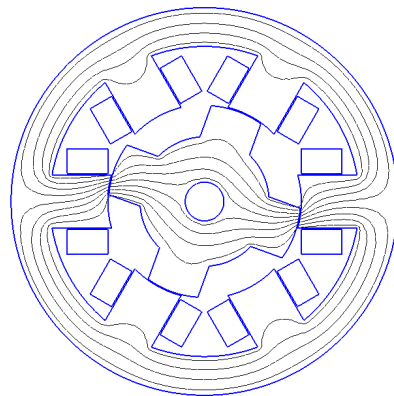


Gyselinck, J., Dular, P., Krähenbühl, L., & Sabariego, R. V. (2016). Finite-Element Homogenization of Laminated Iron Cores With Inclusion of Net Circulating Currents Due to Imperfect Insulation. IEEE Transactions on Magnetics, 52(3), 1-4.

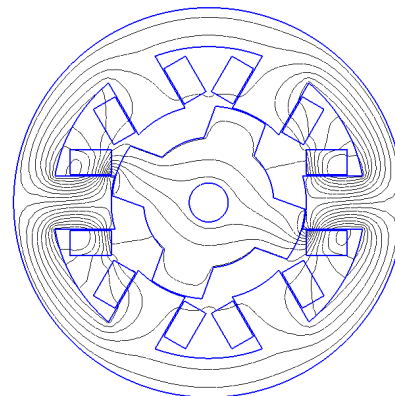
# Some advanced magnetics features

## Skin and proximity effect in windings

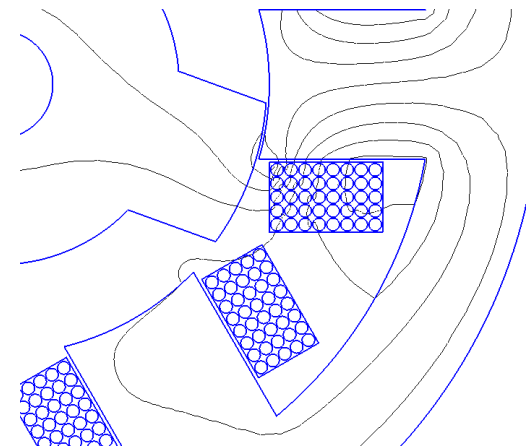
e.g. 6/4 SRM [CEFC2012]



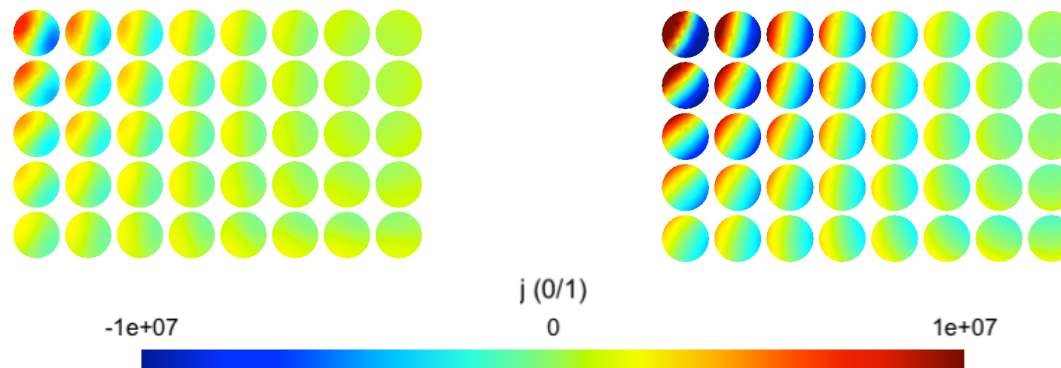
in phase



in quadrature



**current density** (A/m<sup>2</sup>), in phase and in quadrature with imposed current @ 10 kHz



# Some advanced magnetics features

## Homogenisation of windings: LF proximity effect

round wires, low frequency (negligible reaction field)

- analytical solution
- very easy implementation (e.g. in GetDP)

$$P_{\text{prox}} = \frac{\lambda \sigma r^2}{2} \pi^2 f^2 \hat{b}^2 \quad (\text{W/m}^3)$$

$$\mathbf{h}_{\text{dyn}}(t) = \nu_0 \mathbf{b}(t) + \frac{\lambda \sigma r^2}{4} \frac{d\mathbf{b}}{dt} \quad \lambda : \text{fill factor}$$

```

If (Flag_Prox)¶
  Galerkin { DtDof [ Fac_Prox[] * Dof{d a} , {d a} ] ; In Domain_Prox ; Jacobian Vol ; Integration I1 ; }¶
EndIf¶

```

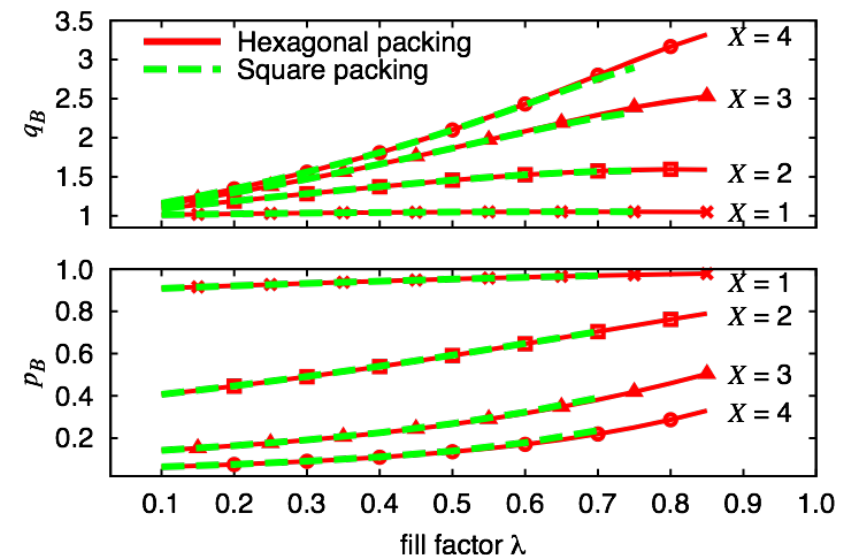
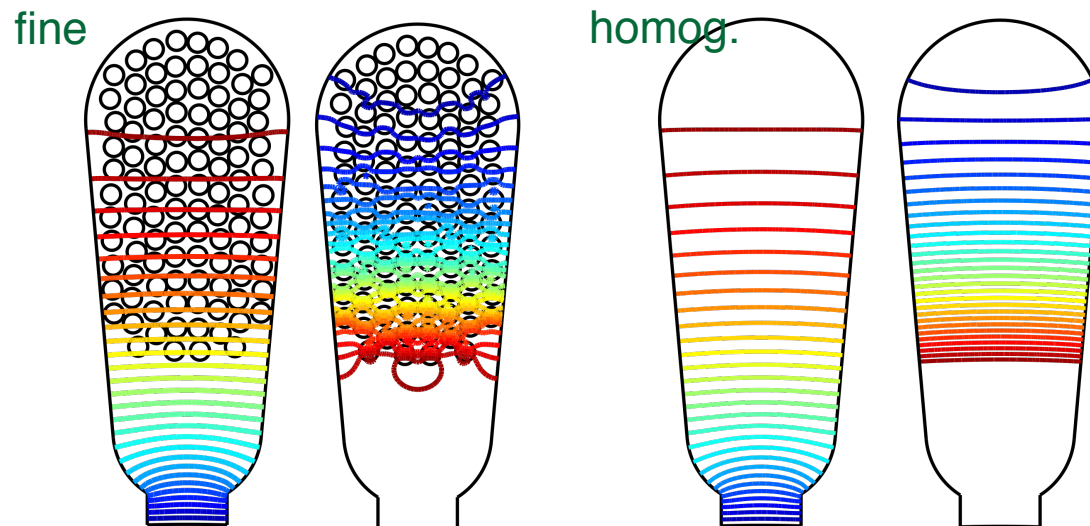
**skin effect:** implementation via electrical circuit

# Some advanced magnetics features

## Homogenisation of windings

frequency-domain and time-domain homogenisation

- skin-effect and proximity effect coefficients (linked to active and reactive power)
- dependent on conductor shape and fill factor
- tunable order of TD approximation (accuracy vs CPU time)



Gyselinck, Johan, Ruth Sabariego, and Patrick Dular. "Time-domain homogenization of windings in 2-D finite element models." IEEE transactions on magnetics 43.4 (2007): 1297-1300.

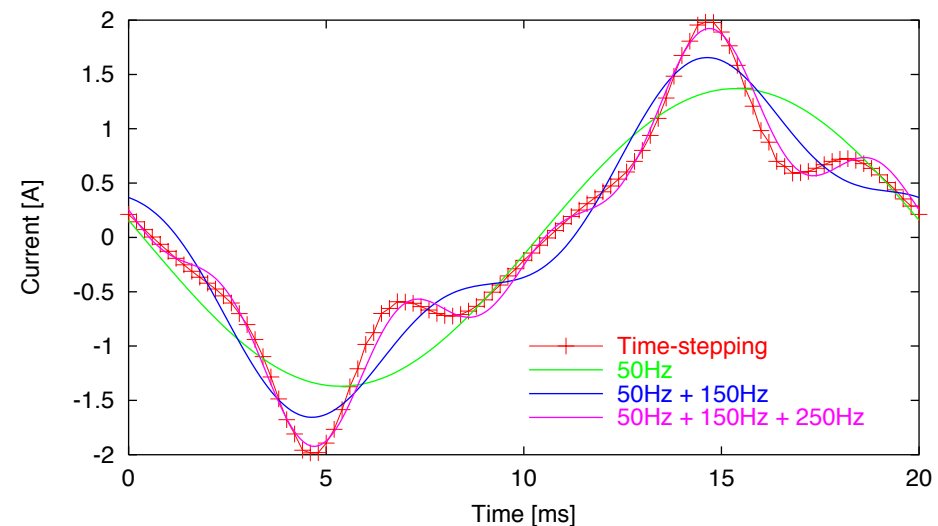
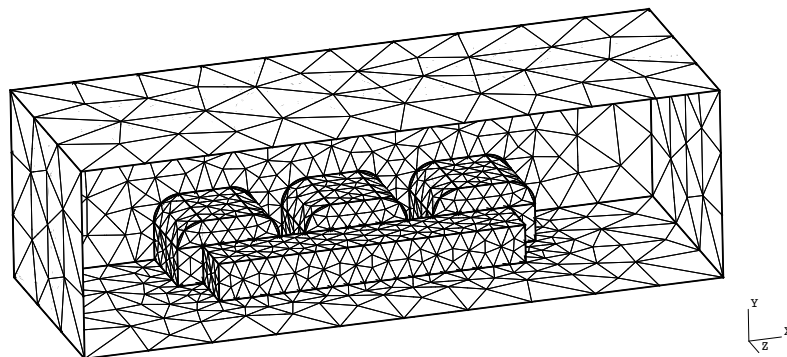
# Some advanced magnetics features

## Harmonic balance (models will be online sometime soon)

**periodic steady-state solution** is obtained by solving 1 big system of equations

- truncated Fourier series, with preset list of frequencies
- source of harmonics:
  - supply, magnetic saturation, lumped components in circuit, movement
- in some cases quicker than plain time-stepping

3-phase inductor,  
50 Hz sinusoidal voltage supply

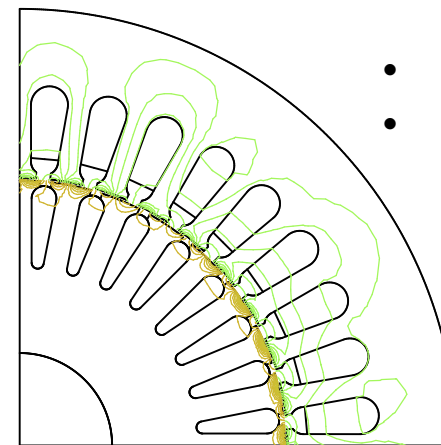
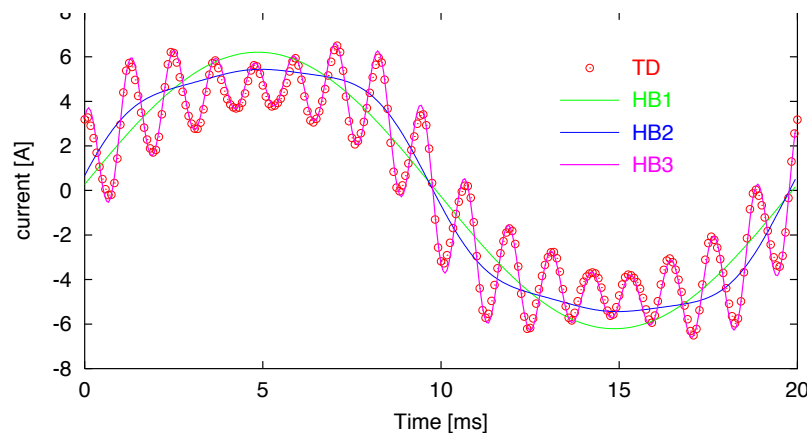
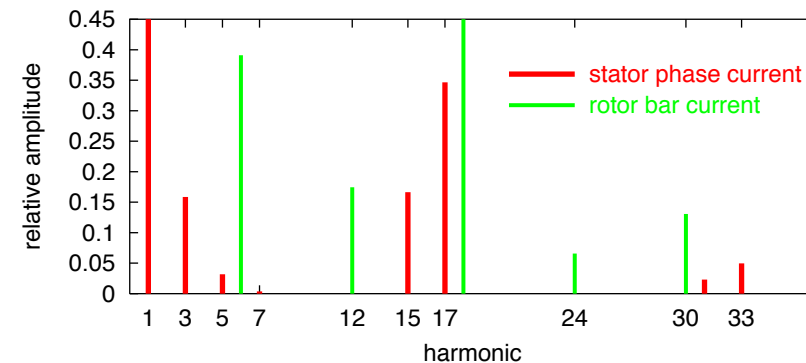
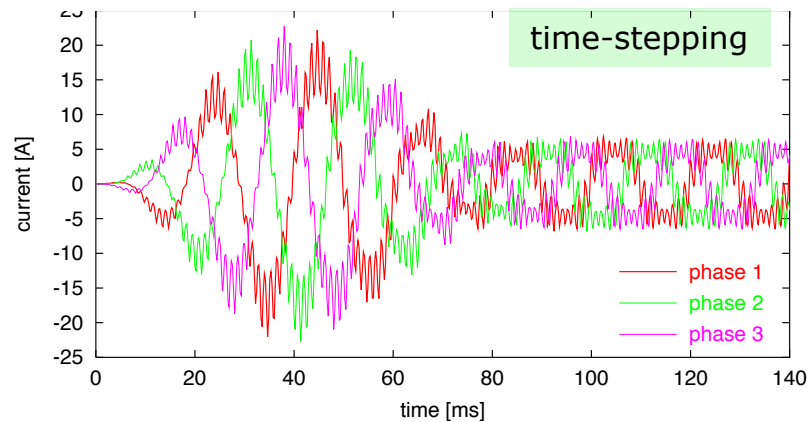


Gyselink, J., Dular, P., Geuzaine, C., & Legros, W. (2002). Harmonic-balance finite-element modeling of electromagnetic devices: a novel approach. IEEE Transactions on Magnetics, 38(2), 521-524.

# Some advanced magnetics features

## Harmonic balance – rotating machines

3kW IM, rated 50Hz sinusoidal voltage supply, synchronous speed



- 17f in stator
- 18f in rotor

Gyselinck, J., Vandevelde, L., Dular, P., Geuzaine, C., & Legros, W. (2003). A general method for the frequency domain FE modeling of rotating electromagnetic devices. IEEE transactions on magnetics, 39(3), 1147-1150.



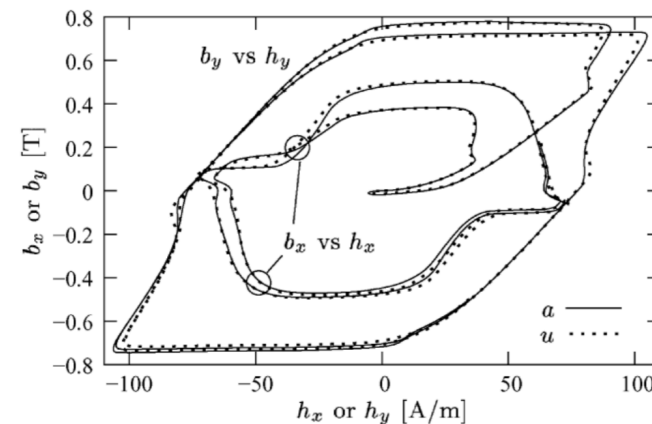
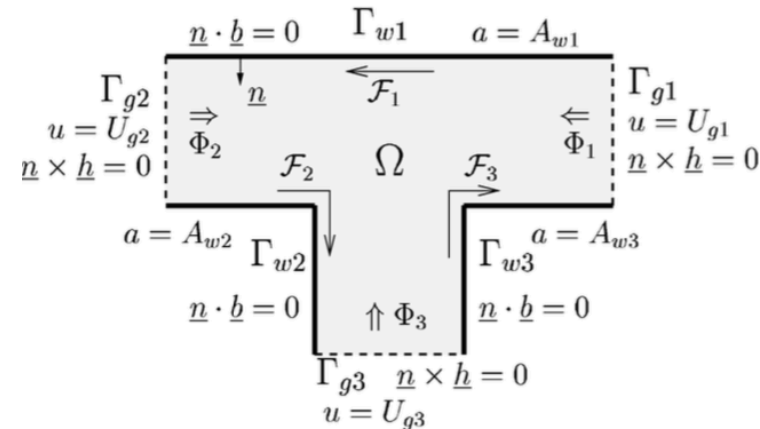
# Some advanced magnetics features

## Hysteresis modelling

vector Jiles-Atherton model

applied to a T-domain,  
2 different FE formulations

imposed MMFs with phase shift,  
-> rotational flux-density



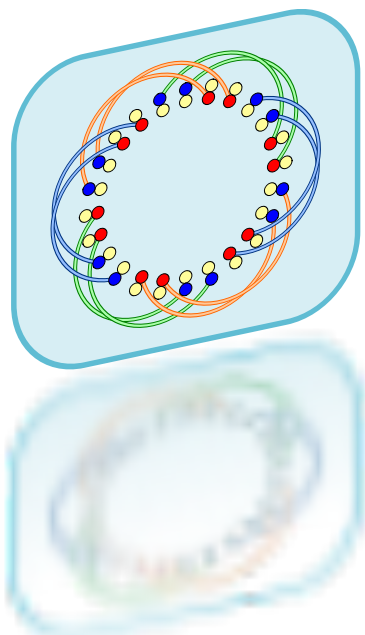
Gyselinck, J., Vandeveld, L., Melkebeek, J., & Dular, P. (2004). Complementary two-dimensional finite element formulations with inclusion of a vectorized Jiles-Atherton model. COMPEL-The international journal for computation and mathematics in electrical and electronic engineering, 23(4), 959-967.



# **koil**: a tool to design the winding of rotating electric machinery

Luigi Alberti

Department of Industrial Engineering  
University of Padova  
[luigi.alberti@unipd.it](mailto:luigi.alberti@unipd.it)



available at: <http://koil.sourceforge.net/>

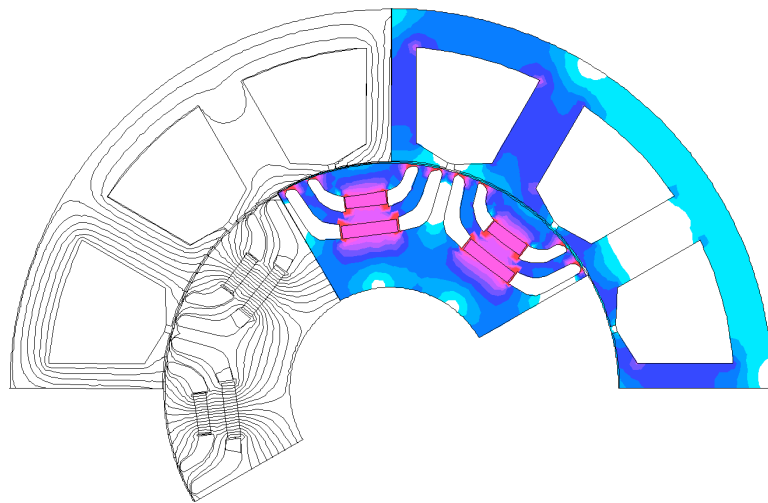
# Brief history



IEEE Industry Applications Society



## Theory and Design of Fractional-Slot PM Machines



Nicola Bianchi  
Michele Dai Pré  
Luigi Alberti  
Emanuele Fornasiero

Dept. of Electrical Engineering - University of Padova

Sponsored by the  
IEEE-IAS Electrical Machines Committee

TUTORIAL COURSE NOTES  
May 2<sup>nd</sup>, 2007

The development of *koil*  
*started in 2007 in conjunction*  
*with a tutorial sponsored by*  
*the IEEE-IAS Electrical*  
*Machines Committee*

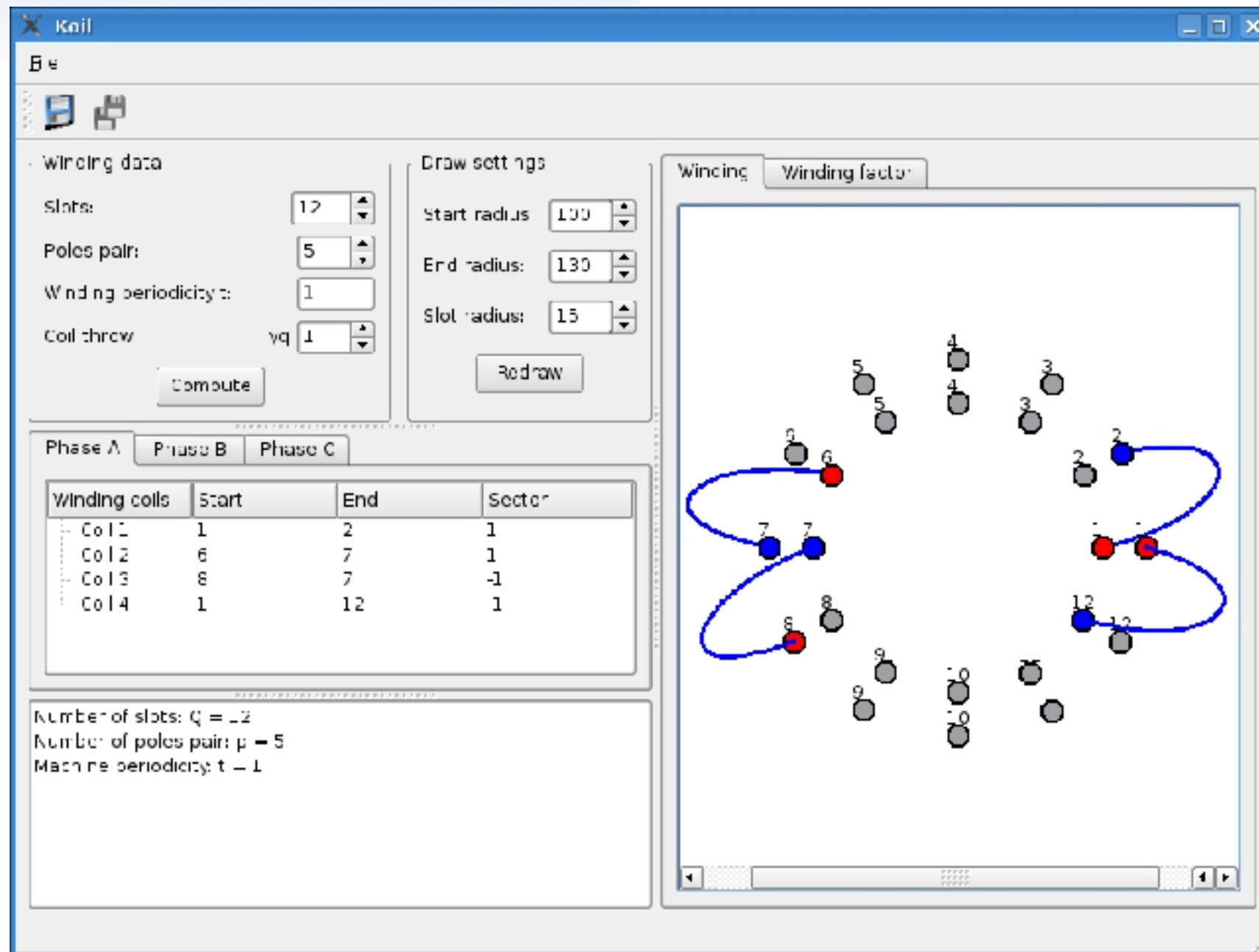
Bianchi, Nicola; Dai Pré, Michele; Alberti,  
Luigi and Fornasiero, Emanuele,

*“Theory and Design of Fractional-Slot PM  
Machines” CLEUP 2007, Padova, ISBN  
978-88-6129-122-5*

koil ver. 1.0  
KOIL AGL 1.0

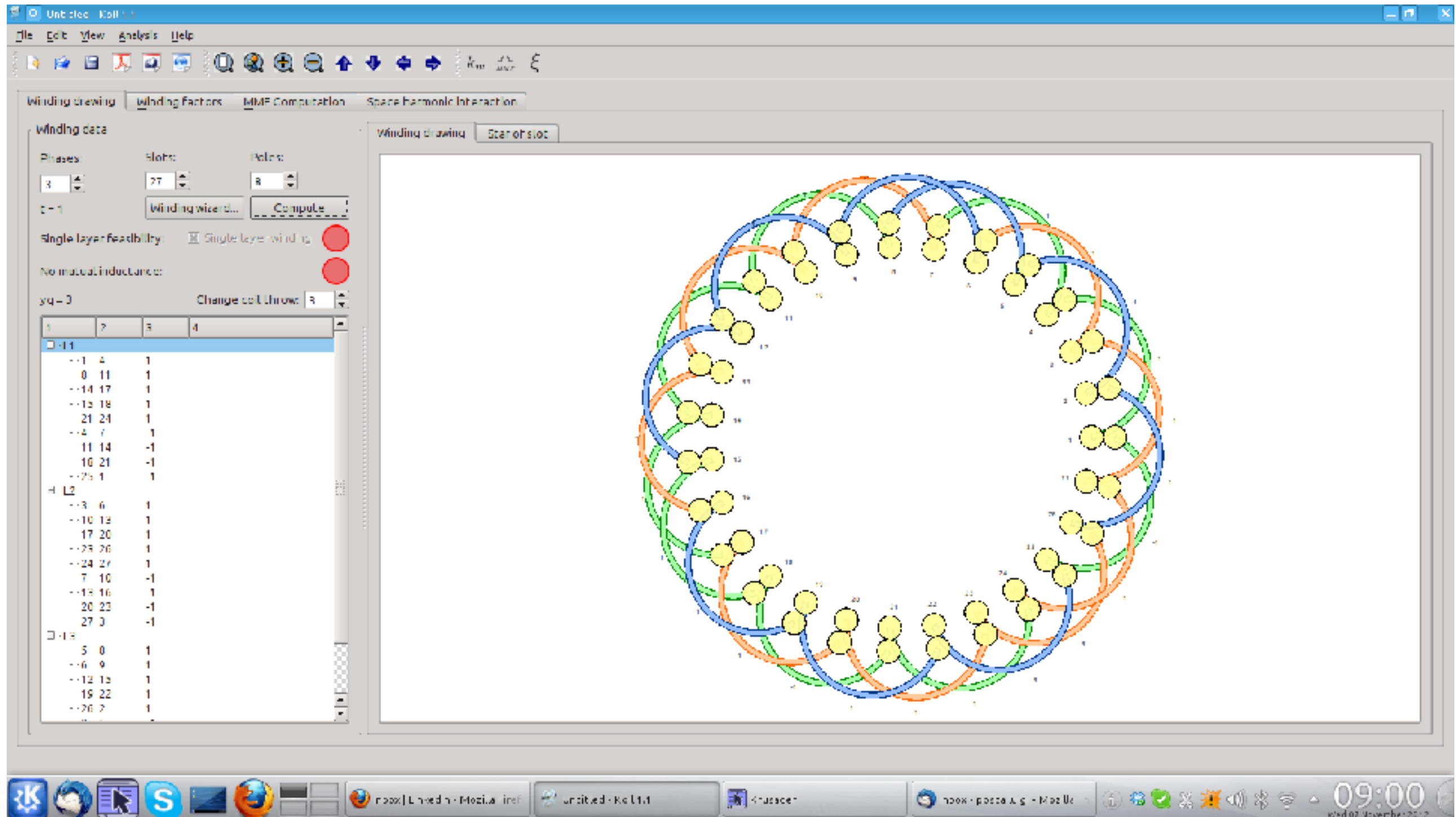
Manual for version 1.0.

Implementation of  
algorithm for layout the  
winding automatically

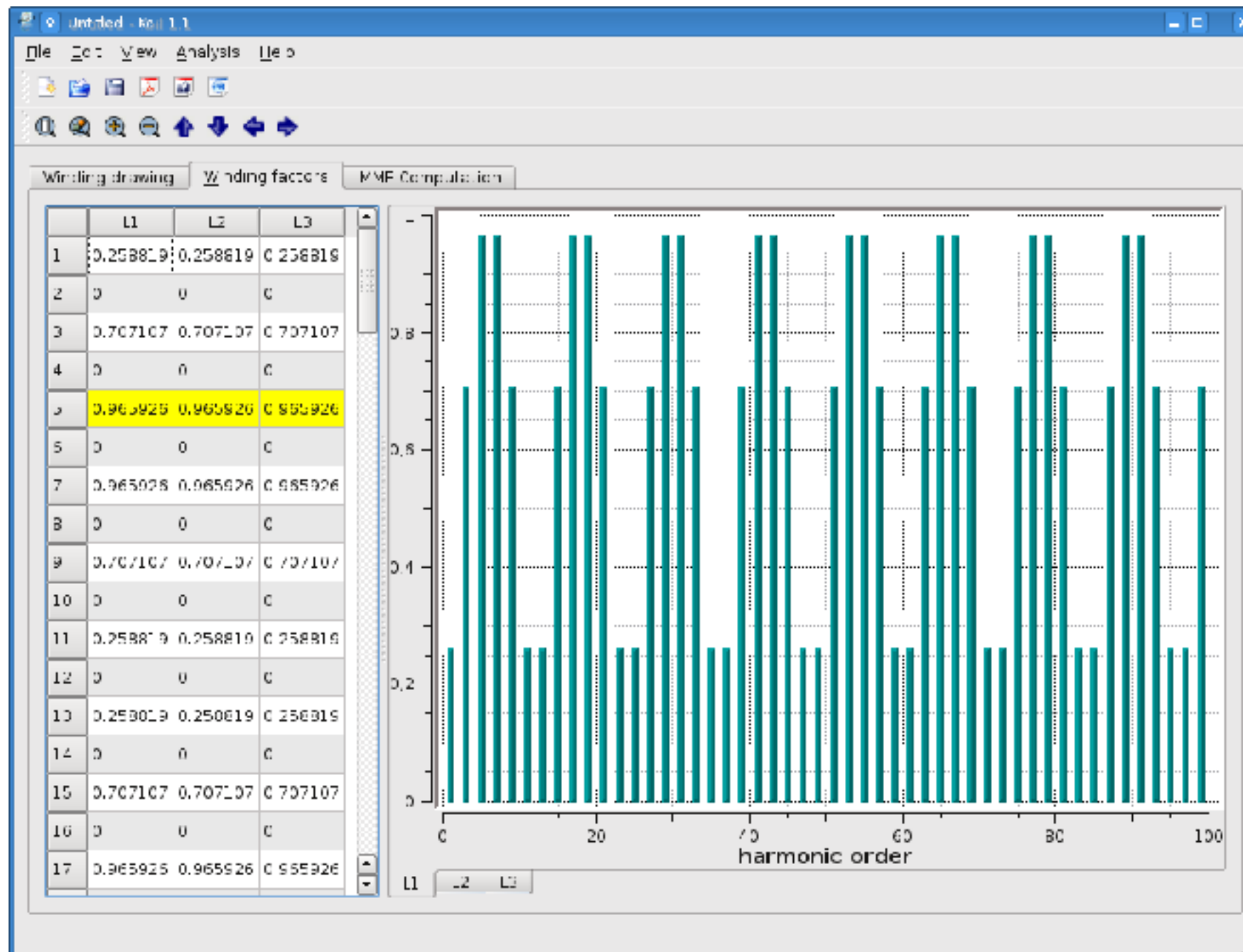


Example:  
12-slot  
10-pole  
winding

# In the following versions Improvements of UI....

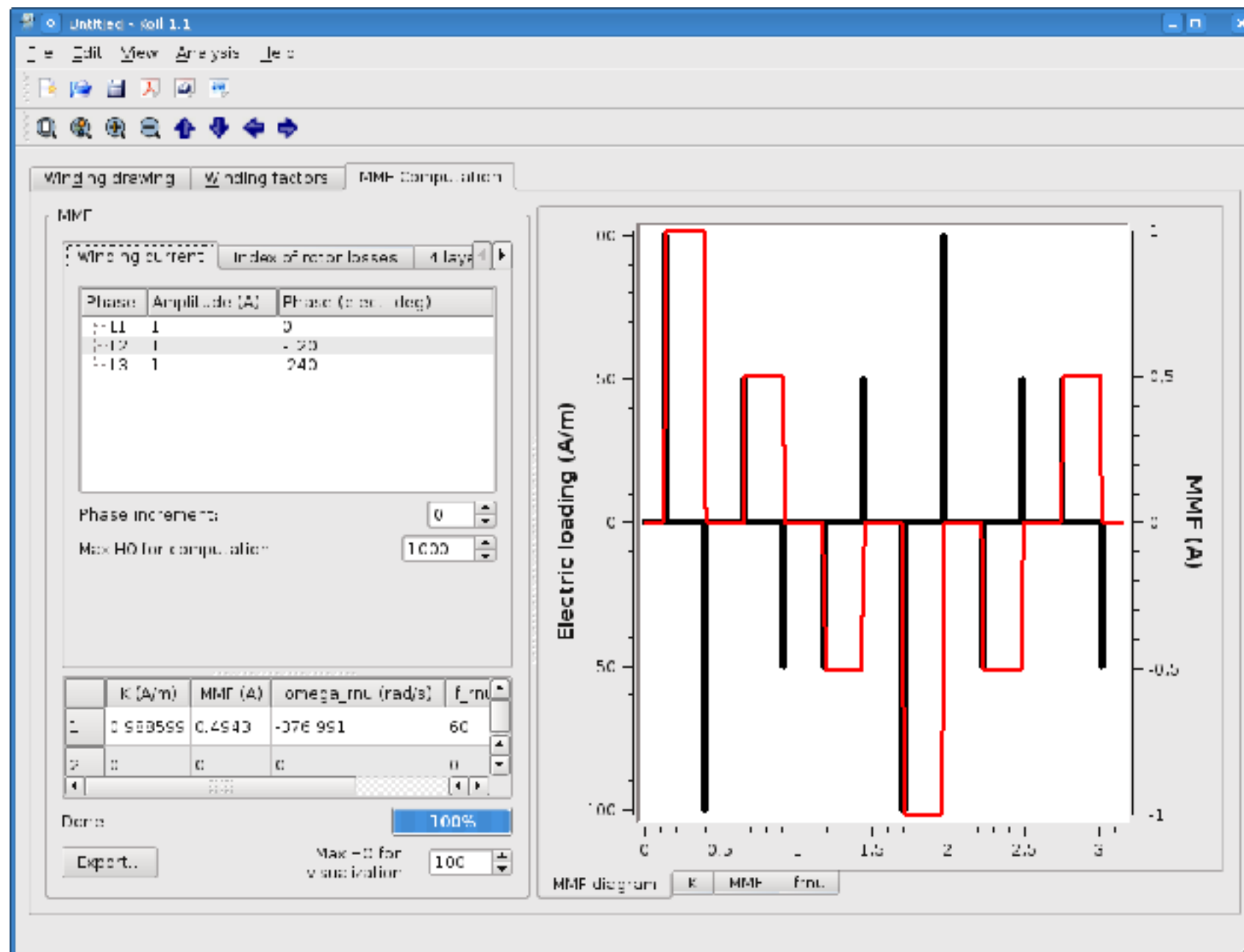


...and improvements of computational capabilities:



Winding factors

...and improvements of computational capabilities:



mmm harmonics  
and waveforms

...and improvements of computational capabilities:

Winding wizard

Slots Q

Select range from 0 to 24

Poles 2p

Select range from 2 to 14

Winding factor kw

Select range from 0.00 to 1.00

Mutual coupling between phases

☐ No mutual coupling

Coil throw yq

☐ Choose yq 1

Single layer

☐ Single layer possible

☒ Only double layer

Machine periodicity

☐ No sub-harmonics

☐ Periodicity multiple C

☐ Periodicity equal to C

Equal groups

☐ equal groups of coils CN

☐ Number of coils per group C

44 items Search winding

	Q	2p	kw	l	Nmag
1	0	2	1	2	0.5
2	12	4	1	4	0.5
3	18	6	1	6	0.5
4	24	8	1	8	0.5
5	12	2	0.965925	2	0.4
6	24	4	0.965925	4	0.4
7	18	2	0.959795	2	0.375
8	24	2	0.957602	2	0.363036
9	21	2	0.953148	2	0.368427
10	21	4	0.953148	2	0.411785
11	21	10	0.953148	2	0.636354

OK Cancel

winding wizard:  
to easily compare  
different windings

Fornasiero, E.; Alberti, L.; Bianchi, N. & Bolognani, S.

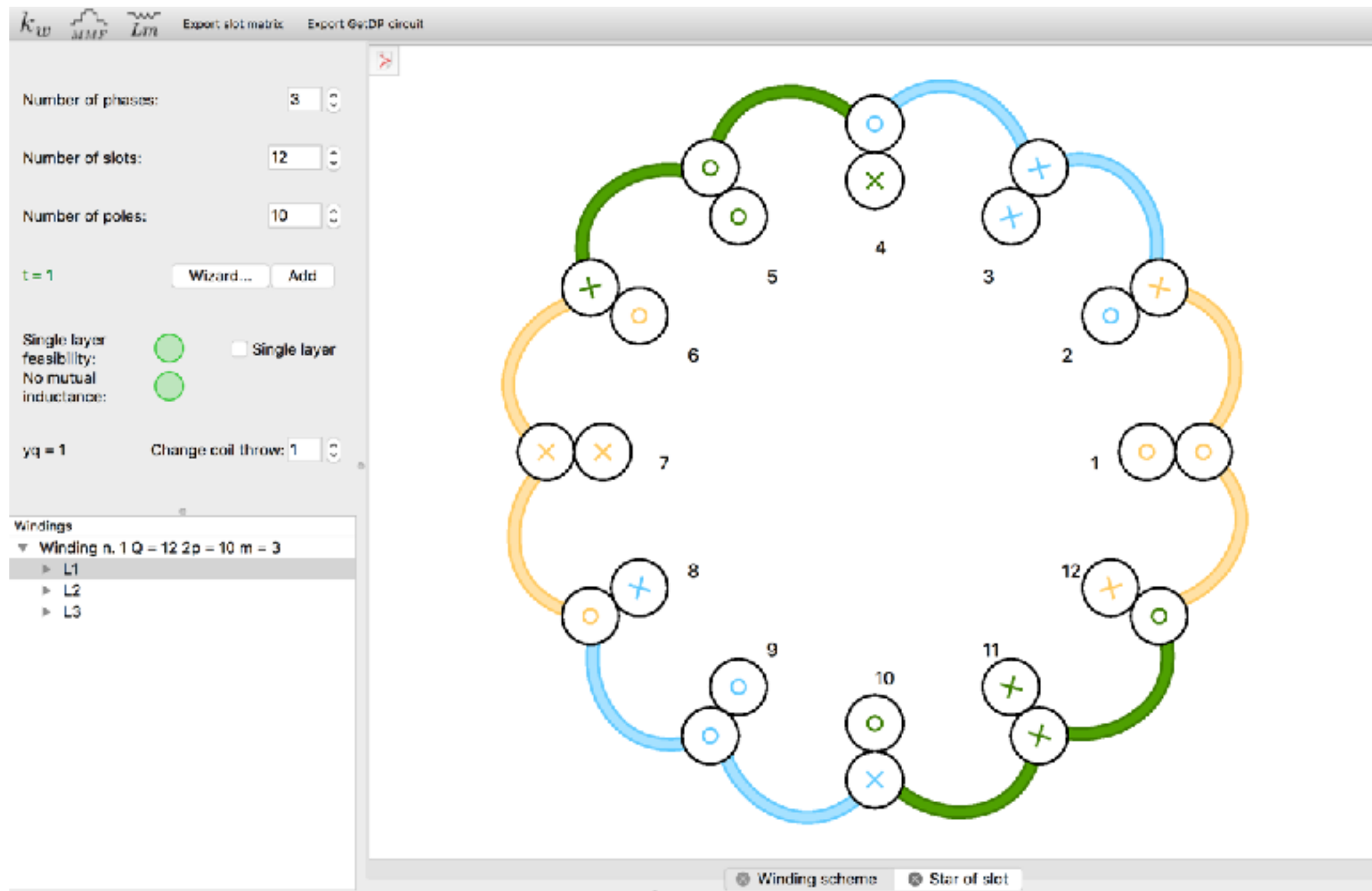
*“Considerations on Selecting Fractional-Slot Nonoverlapped Coil Windings”*

*IEEE Transactions on Industry Applications*, **2013**, 49, 1316-1324



# Version 2.0 (current release)

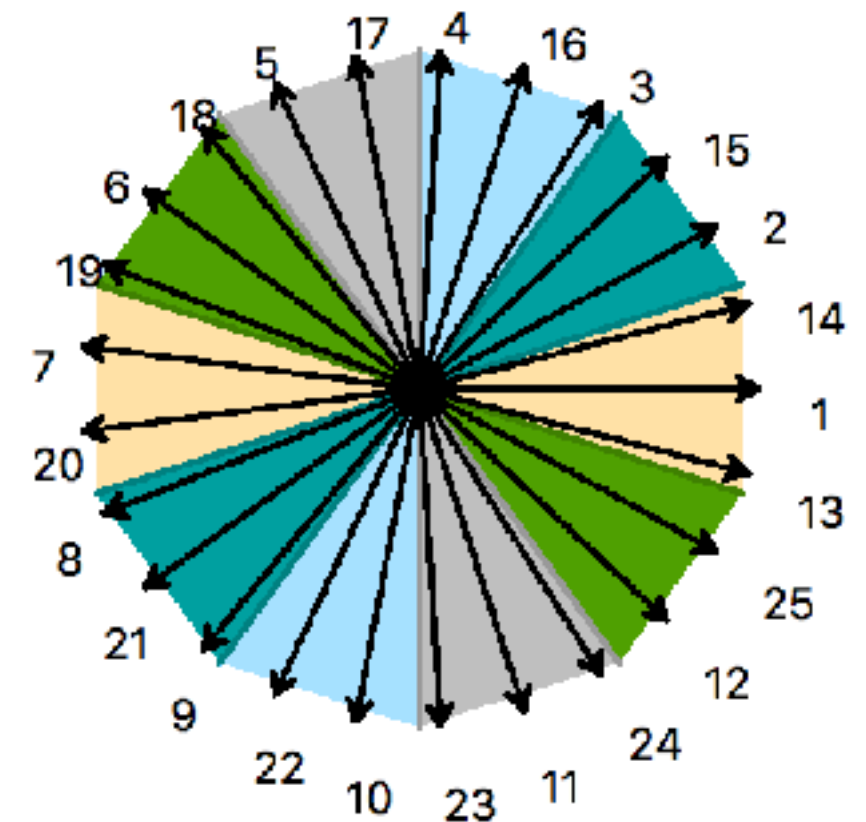
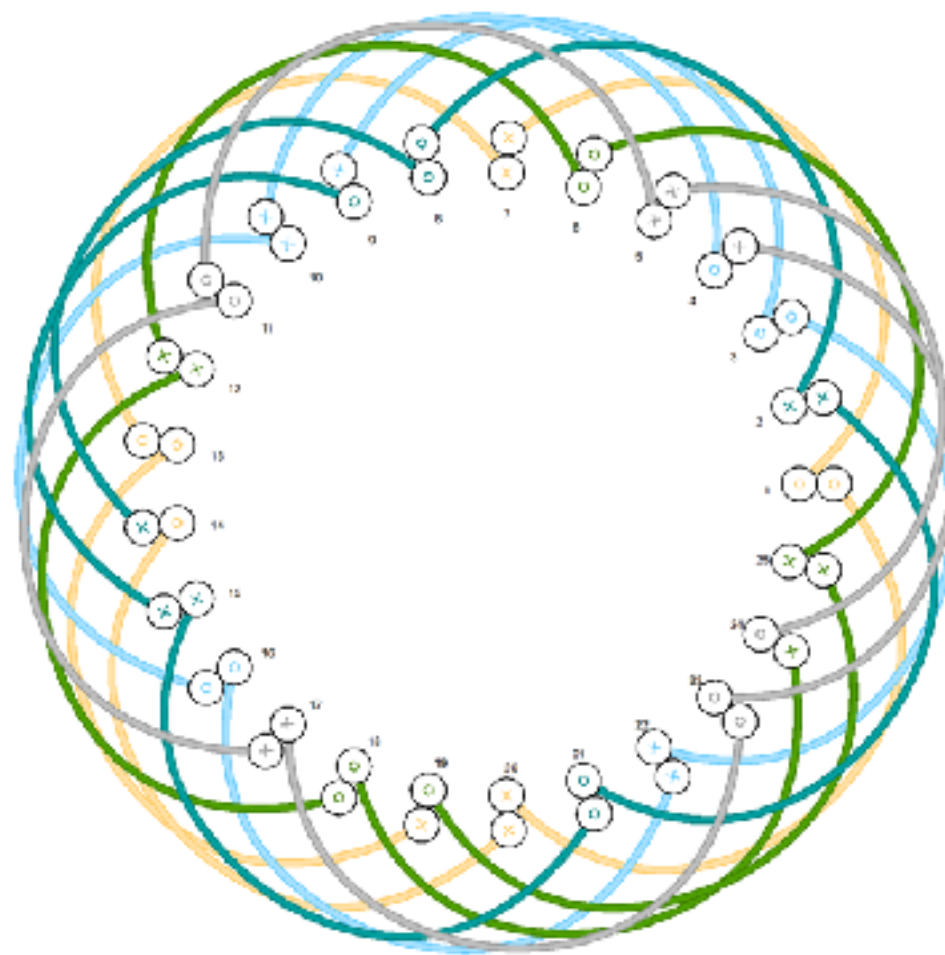
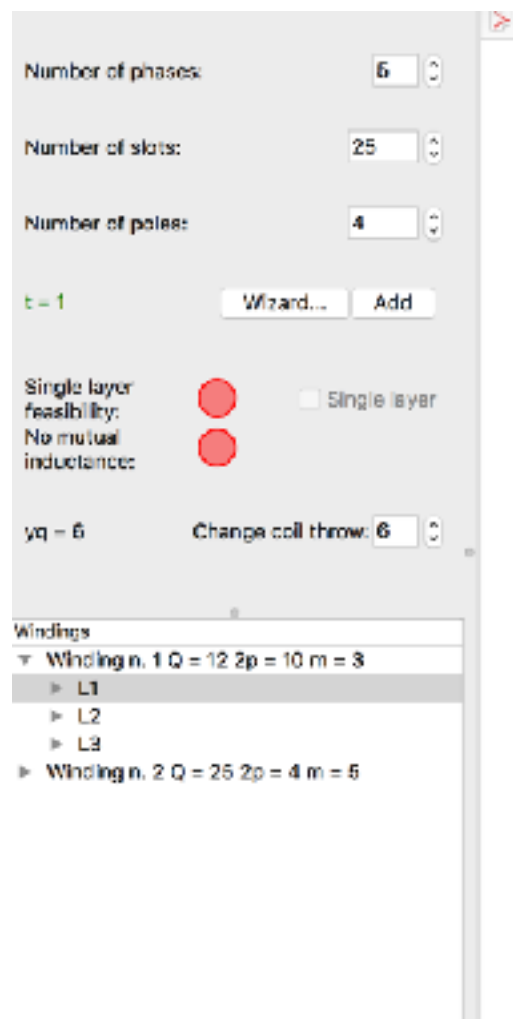
new UI with improved graphic





# Version 2.0 (current release)

any number of phases can be manage

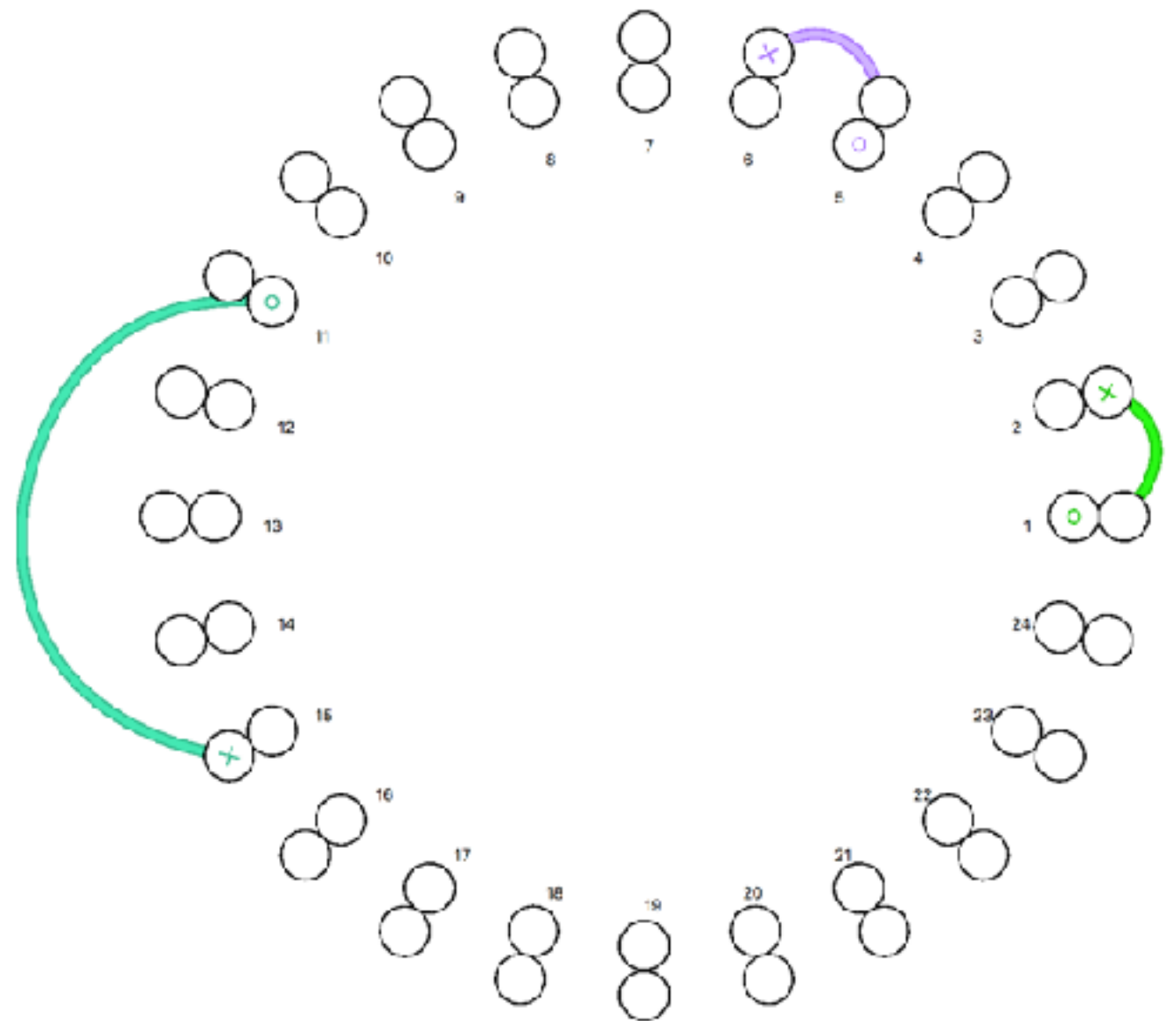


# Version 2.0 (current release)



added Python scripting support  
useful to analyse custom windings  
(unbalanced, not symmetrical, multi-layer...)

```
1 # define a new multiphase windings
2 test = koil.addmPhaseWinding('Test')
3
4 # add the slots
5 test.drawSlots(24)
6
7 # add the phases
8 a = test.addWinding('a')
9 b = test.addWinding('b')
10 #...
11
12 # add the coils to each phases
13 test.addCoil(1,2,'a')
14 #...
15 #...
16
17 kw = a.getWindingFactor()
18
```



# Development:



Code less.  
Create more.  
Deploy everywhere.

Written in C++ using Qt for UIs



cross platform development



easy to script and automatise  
with python scripting

More info, tutorial and examples at:

<http://koil.sourceforge.net/>

# Selected paper with example/uses case of koil

Alberti, Luigi

“Koil reference manual: the documentation for Koil, a program for the design of windings”

<http://koil.sourceforge.net/>

Bianchi, N.; Dai Pré, M.; Alberti, L. & Fornasiero, E.  
“Theory and Design of Fractional-Slot PM Machines”  
*CLEUP (ISBN 978-88-6129-122-5)*, **2007**, 196

Alberti, L. & Bianchi, N.

“Analysis of asynchronous machines for direct drive wind power generation” IEEE International Electric Machines and Drives Conference, 2009. IEMDC '09., **2009**, 1838-1843

Alberti, L. & Bianchi, N.

Experimental Tests of Dual Three-Phase Induction Motor Under Faulty Operating Condition  
*IEEE Transactions on Industrial Electronics*, **2012**, 59, 2041 -2048

Alberti, L. & Bianchi, N.

“Theory and Design of Fractional-Slot Multilayer Windings” IEEE Transactions on Industry Applications, **2013**, 49, 841-849

Fornasiero, E.; Alberti, L.; Bianchi, N. & Bolognani, S.  
“Considerations on Selecting Fractional-Slot Non-overlapped Coil Windings” IEEE Transactions on Industry Applications, **2013**, 49, 1316-1324

Popescu, M.; Dorrell, D.; Alberti, L.; Bianchi, N.; Staton, D. & Hawkins, D.

Thermal Analysis of Duplex Three-Phase Induction Motor Under Fault Operating Conditions  
*IEEE Transactions on Industry Applications*, **2013**, 49, 1523-1530

Alberti, L.; Barcaro, M. & Bianchi, N.

“Design of a Low-Torque-Ripple Fractional-Slot Interior Permanent-Magnet Motor” IEEE Transactions on Industry Applications, **2014**, 50, 1801-1808

# Electrical machine analysis using free software - Section 4

## SyR-e : Synchronous Reluctance Evolution

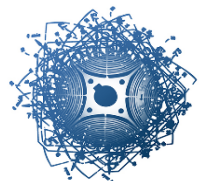
---

ECCE 2017 TUTORIAL

[gianmario.pellegrino@polito.it](mailto:gianmario.pellegrino@polito.it)

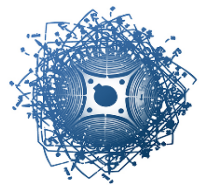


**POLITECNICO  
DI TORINO**



# Outline

- What is SyR-e
- What SyR-e can do
- Example of fast FEA modelling
- Example of preliminary design
- Example of design optimization
- Centrifugal stress modelling
- Conclusion





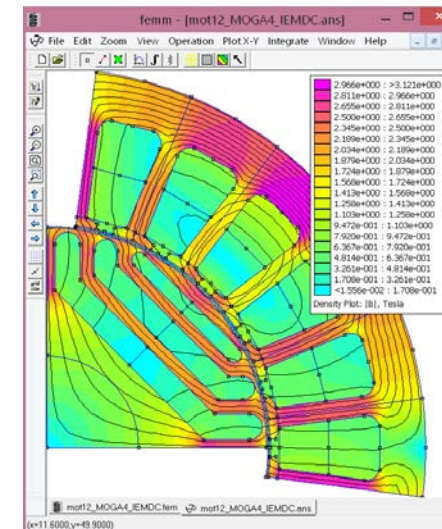
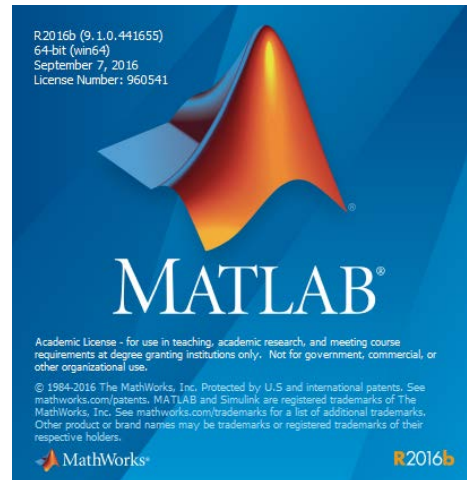
# What is SyR-e



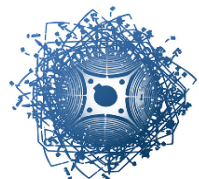
SyR-e: **S**ynchronous **R**eluctance – **e**volution is a design tool for synchronous e-machines

Launched on September 2014 in occasion of another ECCE Tutorial, it is the output of a collaboration between **Politecnico di Torino** and **Politecnico di Bari**, in Italy

It runs under Matlab, using FEMM as a client for magneto-static FEA



SyR-e is downloadable on SourceForge at <https://sourceforge.net/projects/syr-e/>, under the **APACHE License, Version 2.0**



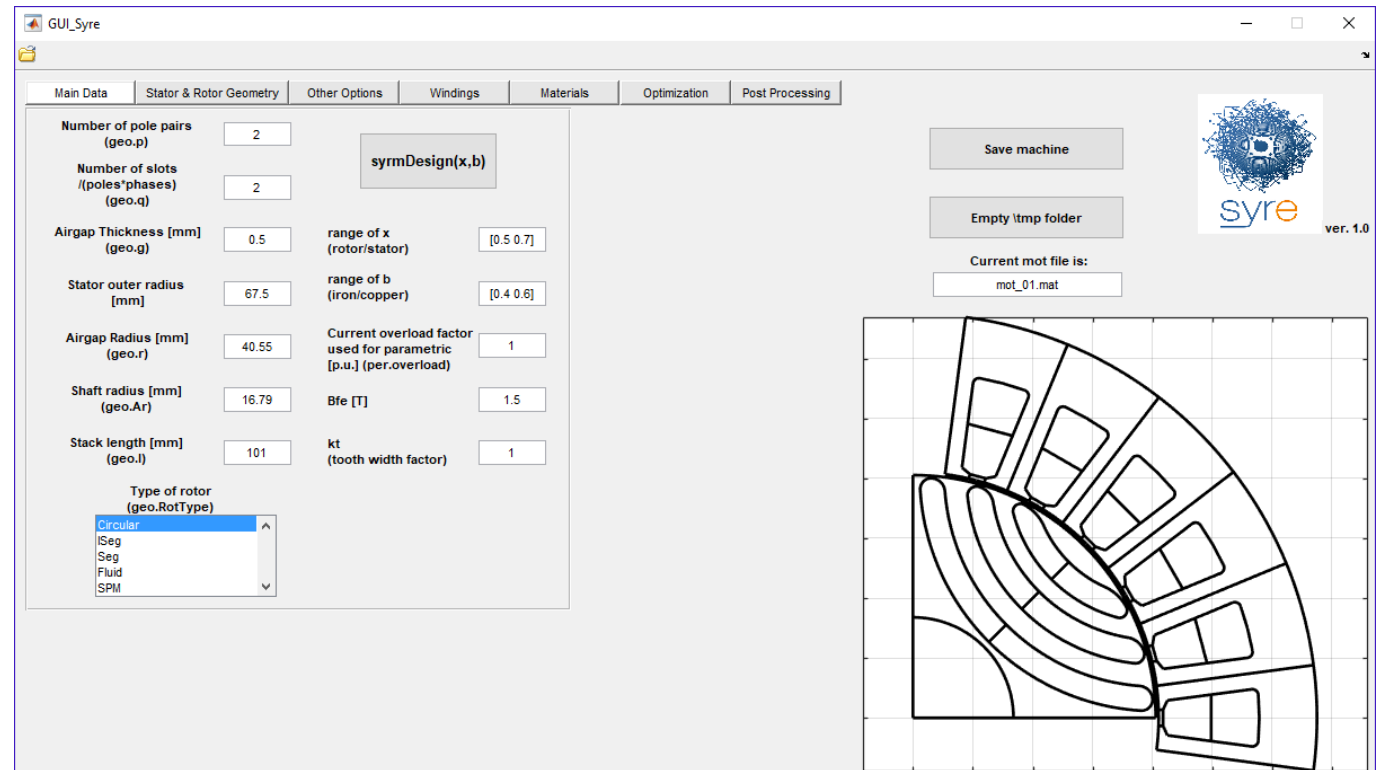


# SyR-e GUI

SyR-e has a Graphical User Interface (GUI) built in Matlab, for ease of data/command input

Design examples are provided:

- mot\_01.fem, mot\_01.mat (initial design)
- RAWP.fem, RAWP.mat (demo)



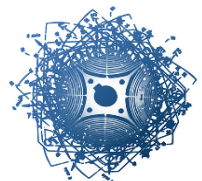
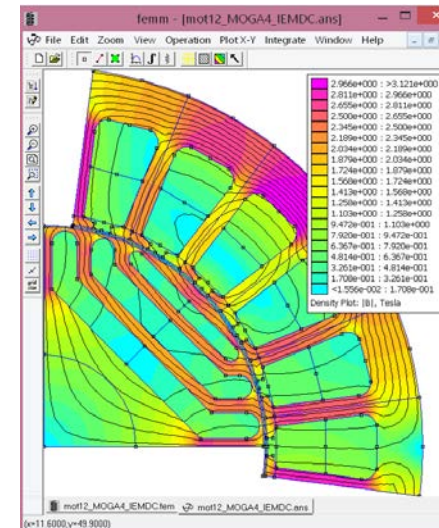
# Is SyR-e Open-Source?

SyR-e syntax is compatible with GNU Octave

Not supported in Octave:

- parallel processing (parfor command)
- SyR-e GUI

Altogether, using Matlab is recommended

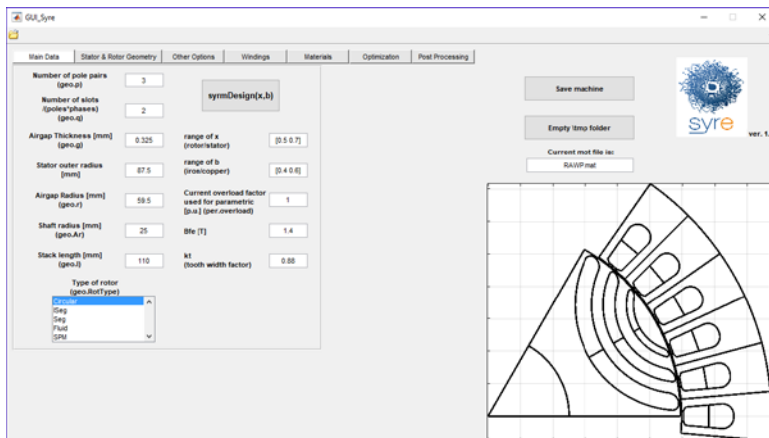


# What SyR-e can do

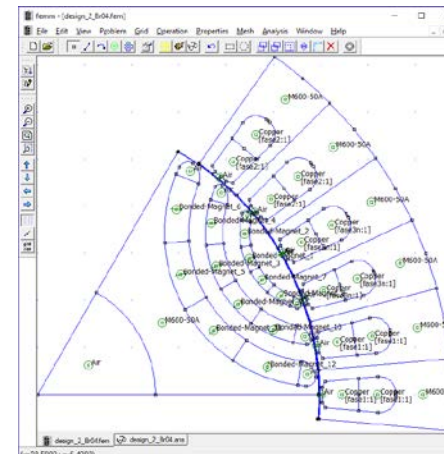
The key feature of SyR-e is the capability of **fast modelling and FEA simulation**

Moreover, SyR-e includes:

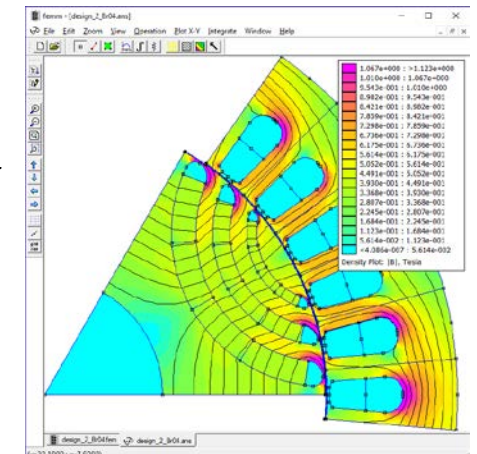
- **Equations** for preliminary design ( $x, b$  plane approach)
- **Optimization** of selected design goals using MODE
- Estimates of **copper temperature** and **centrifugal stress**
- Scripts for advanced manipulation of FEA output
- Capability of export towards other CAD environments (Autocad, Infolytica)



Save Machine



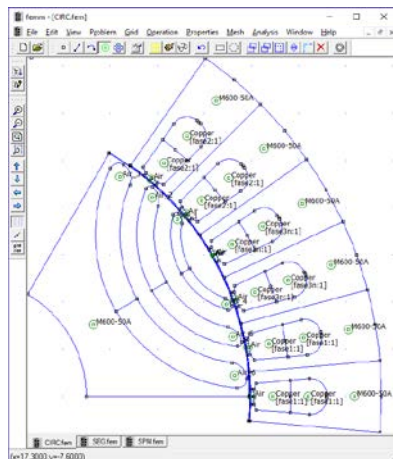
Post Processing\Start



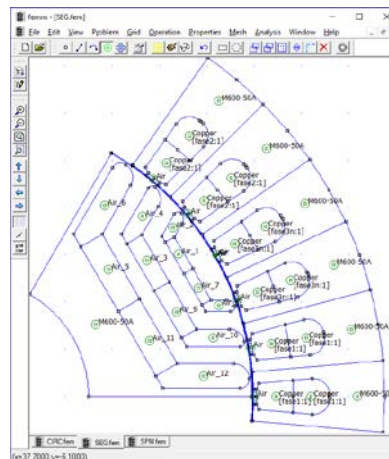
# Machine Types

- ✓ Synchronous Reluctance and PM-assisted Synchronous Reluctance machines:
  - circular barriers
  - angled barriers (SEG)
  - fluid barriers
- ✓ Surface-mounted PM machines
- ✓ Distributed and concentrated winding machines

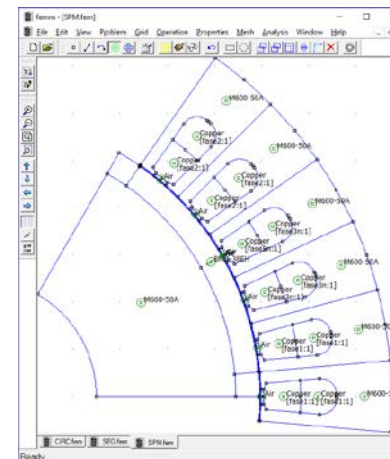
Circular barriers



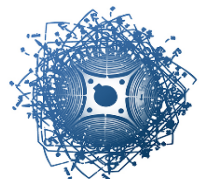
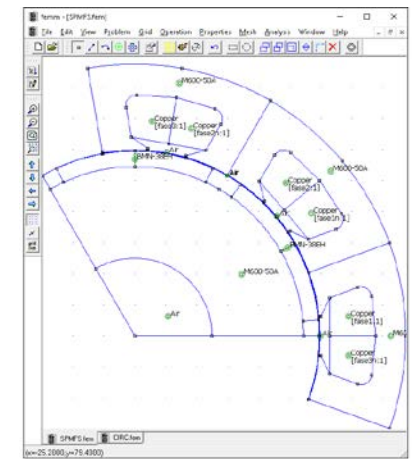
SEG barriers



Surface-mounted PMSM



Concentrated Windings



# KOIL for SyR-e

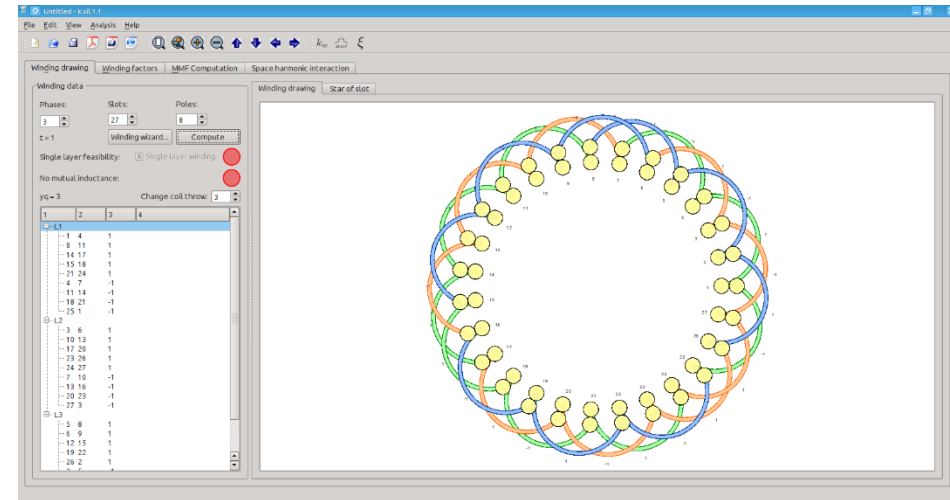
SyR-e uses **koil\_syre** for winding design

This is a Matlab executable version of Koil

Koil is a program to compute the windings of rotating electrical machinery, started in 2008 at the Electric Drives Laboratory (EDLab) of the University of Padova, in Italy

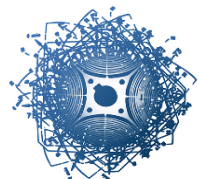
Koil is distributed under the terms of the [GNU General Public License \(GPL\)](http://www.gnu.org/licenses/gpl-3.0.html)

<http://koil.sourceforge.net/>



```
Command Window

*****
This is koil_syre, a koil forc to be interfaced with Syre
visit koil.sourceforge.net Copyright 2009-2014 Luigi Alberti
*****
fx
```



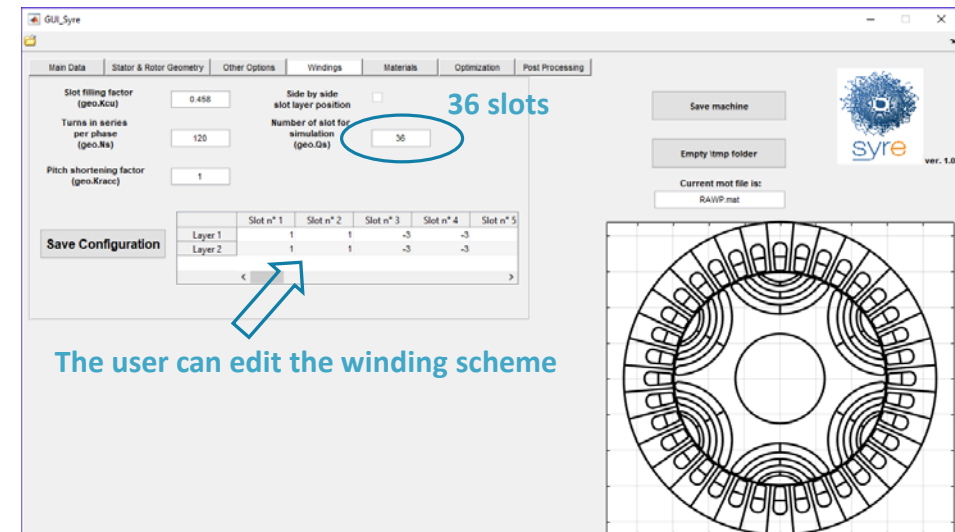
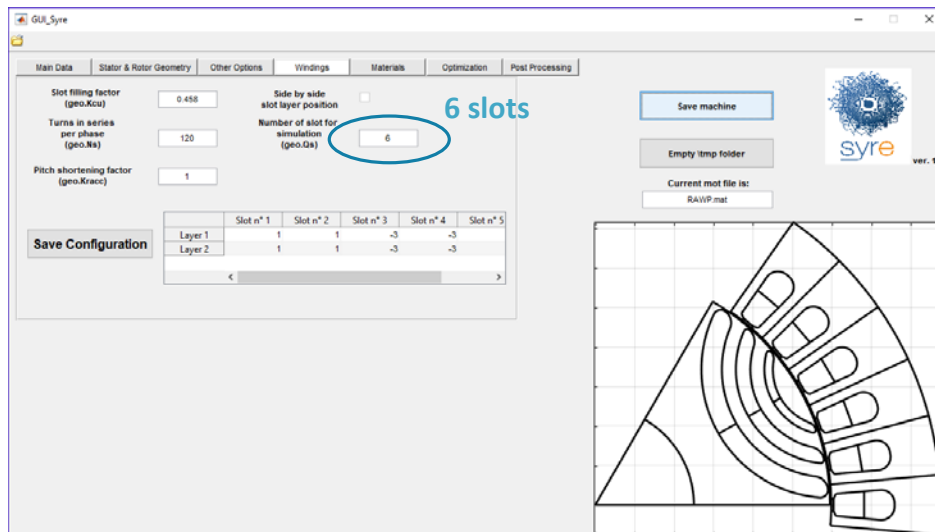
# Automatic or User-defined Winding

KOIL designs the winding distribution into slots automatically, given the slot/pole/phase number

Example:  $p = 3$  (pole pairs),  $q = 2$  (slots/pole/phase)

**Odd-periodic symmetry** = one rotor pole and **6 stator slots**

Optionally, bigger portions of the machine can be modelled, for example for modifying the winding set manually (e.g. **all 36 slots**)





# Demo: RAWP.fem

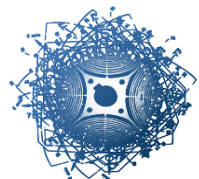
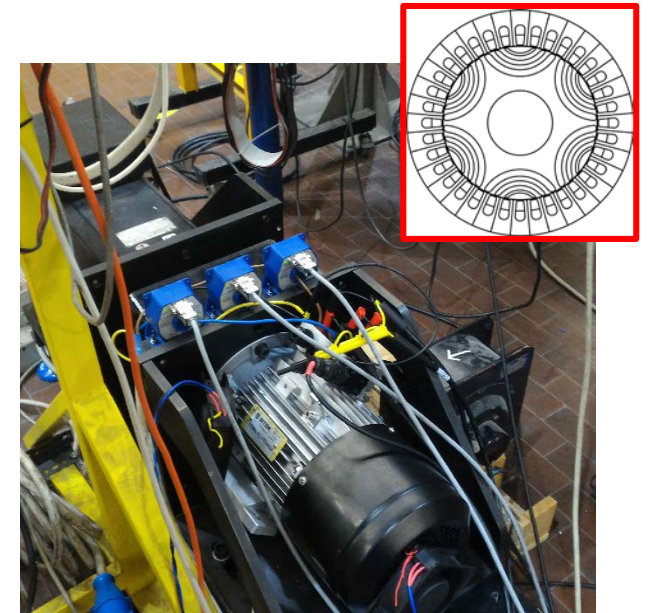
This SyR motor prototype uses the stator of a 2.2 kW, 3 pole-pair induction motor

Different rotors of the synchronous type were designed and tested in replacement of the original squirrel cage rotor [1], including:

- Surface-Mounted PM (SPM),
- Synchronous Reluctance
- PM-assisted Synchronous Reluctance

The RAWP demo (RAWP.fem, RAWP.mat) refers to the SyR machine geometry. This is available online.

[1] C. Bianchini, M. Davoli, G. Pellegrino, F. Immovilli and E. Lorenzani, "Low cost PM synchronous servo-applications employing asynchronous-motor frame," *2015 IEEE Energy Conversion Congress and Exposition (ECCE)*, Montreal, QC, 2015, pp. 6090-6095



# Reference Data

## Baseline Induction Motor

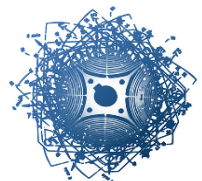
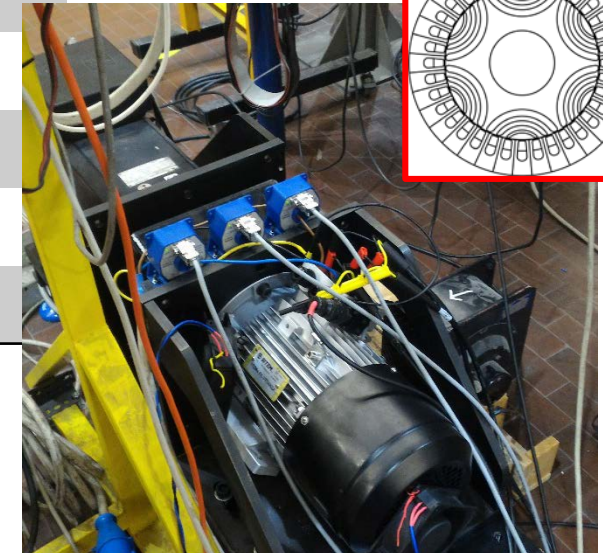
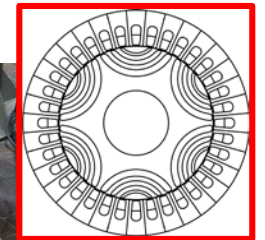
Power	kW	2.2
Torque	Nm	21.7
Speed	rpm	969
Voltage	V	400
Current	A	5.0
Frequency	Hz	50
Pole pairs		3
PF		0.75
Efficiency		86.0 % (IE3)

## RAWP demo – SyR rotor

Power	kW	2.2
Torque	Nm	21.0
Speed	rpm	1000
Voltage	V	157
Current	A	13.2
Frequency	Hz	50
Pole pairs		3
PF		0.70
Efficiency		85.0 %

### NOTES:

- The SyR machine was rewound for a lower voltage
- The reported results does not refer to the same control strategy for the IM and the SyR machine





# Initial Input Data

## Geometry

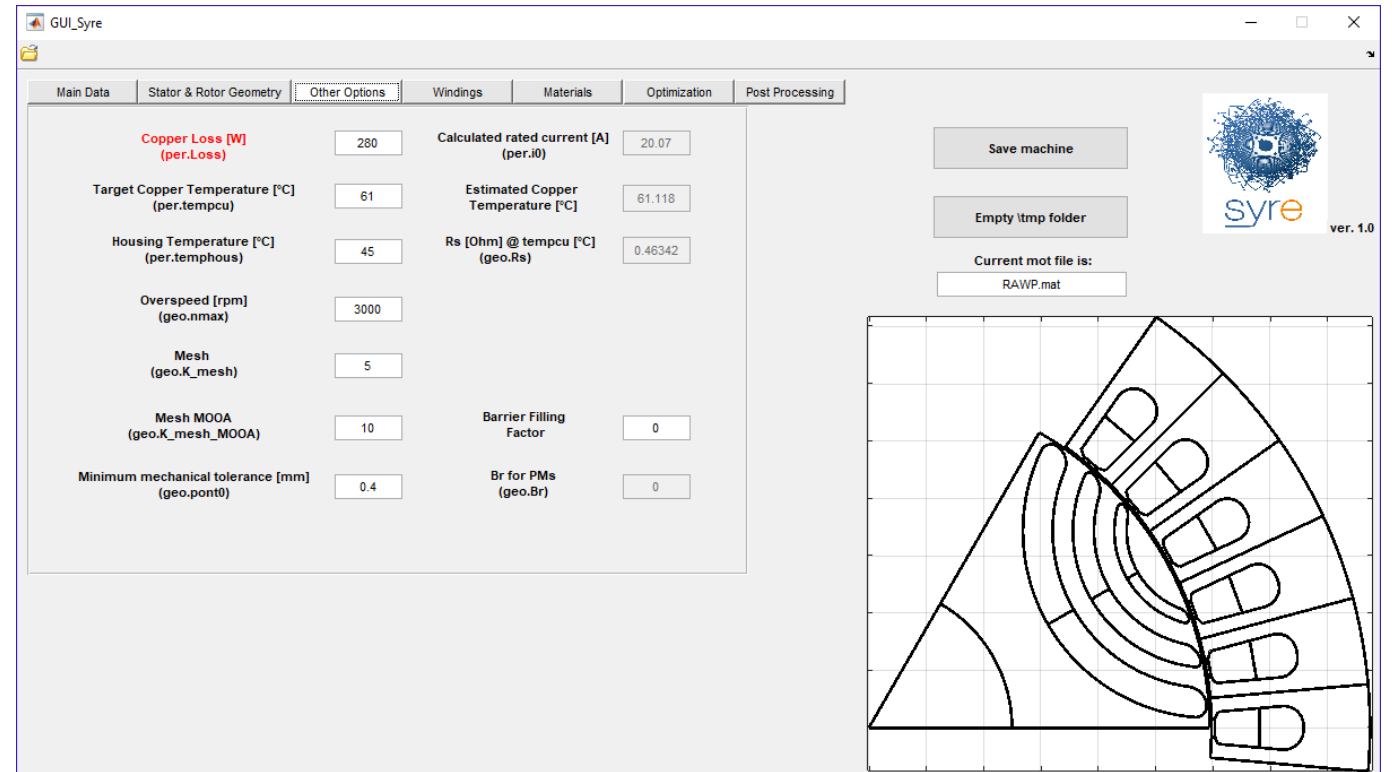
- Number of pole pairs
- Stack size (outer diameter, length)
- Airgap length
- Rotor number of layers
- Layers thickness and positions

## Winding

- Number of slots per pole per phase
- Slot filling factor

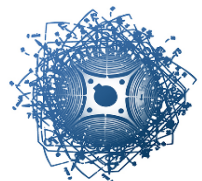
## Current loading

- Target copper temperature
- **Copper loss at stall**



The screenshot displays the GUI\_Syre software interface, which is used for motor design. The interface includes a menu bar with options: Main Data, Stator & Rotor Geometry, Other Options, Windings, Materials, Optimization, and Post Processing. The main area is divided into two columns of input fields. The left column contains fields for Copper Loss [W] (per.Loss), Target Copper Temperature [°C] (per.tempcu), Housing Temperature [°C] (per.temphous), Overspeed [rpm] (geo.nmax), Mesh (geo.K\_mesh), Mesh MOOA (geo.K\_mesh\_MOOA), and Minimum mechanical tolerance [mm] (geo.pont0). The right column contains fields for Calculated rated current [A] (per.i0), Estimated Copper Temperature [°C], Rs [Ohm] @ tempcu [°C] (geo.Rs), Barrier Filling Factor, and Br for PMs (geo.Br). On the right side of the interface, there are buttons for 'Save machine', 'Empty tmp folder', and a text box for 'Current mot file is: RAWP.mat'. A logo for 'syre ver. 1.0' is also present. At the bottom right, there is a diagram of a motor cross-section showing the stator and rotor geometry.

Parameter	Value
Copper Loss [W] (per.Loss)	280
Calculated rated current [A] (per.i0)	20.07
Target Copper Temperature [°C] (per.tempcu)	61
Estimated Copper Temperature [°C]	61.118
Housing Temperature [°C] (per.temphous)	45
R <sub>s</sub> [Ohm] @ tempcu [°C] (geo.Rs)	0.46342
Overspeed [rpm] (geo.nmax)	3000
Mesh (geo.K_mesh)	5
Barrier Filling Factor	0
Mesh MOOA (geo.K_mesh_MOOA)	10
Br for PMs (geo.Br)	0
Minimum mechanical tolerance [mm] (geo.pont0)	0.4



# Reference Copper Loss

Baseline Induction Motor		
Power	kW	2.2
Torque	Nm	21.9
Speed	rpm	969
Voltage	V	400
Current	A (rms)	<b>5.0</b>
Frequency	Hz	50
Pole pairs		3
PF		0.75
Efficiency		86.0 %
$R_s$	$\Omega$	<b>2.81 (50°C)</b>

Copper loss per stack outer surface  $k_j$  [W/m<sup>2</sup>] is another way of defining the electrical loading or current density of the machine, independently of the number of turns

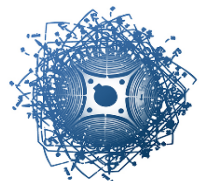
If the stator geometry is unchanged as in here, the copper loss indicator defines the current density in the conductors uniquely

For changes of geometry, this metrics showed to be more intimately related to the steady-state copper temperature rather than current density

Reference value here was:

$$P_{Cu} = 3 \cdot R_s \cdot I^2 = \mathbf{210\ W\ @\ 50\ ^\circ C}$$

Loss per stack outer surface is  $k_j = \mathbf{4.9\ kW/m^2}$



# Copper Temperature

The inputs are:

- Copper loss
- Target copper temperature

SyR-e evaluates:

- Estimated copper temperature
- Phase current  $i_0 = 1$  p.u. and resistance (given the number of turns)

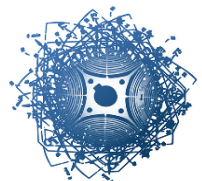
A **thermal network** estimates the copper temperature given the housing temperature

Target and estimated temperature **must be equal**

The screenshot shows the GUI\_Syre software interface with the 'Other Options' tab selected. A red box highlights the input parameters for copper temperature estimation, and a blue box highlights the resulting estimated temperature.

Parameter	Value
Copper Loss [W] (per.Loss)	210
Calculated rated current [A] (per.i0)	17.5586
Target Copper Temperature [°C] (per.tempcu)	50
Estimated Copper Temperature [°C]	50.1611
Housing Temperature [°C] (per.temphous)	38
Rs [Ohm] @ tempcu [°C] (geo.Rs)	0.4541
Overspeed [rpm] (geo.nmax)	3000
Mesh (geo.K_mesh)	5
Mesh MOOA (geo.K_mesh_MOOA)	10
Barrier Filling Factor	0
Minimum mechanical tolerance [mm] (geo.pont0)	0.4
Br for PMs (geo.Br)	0

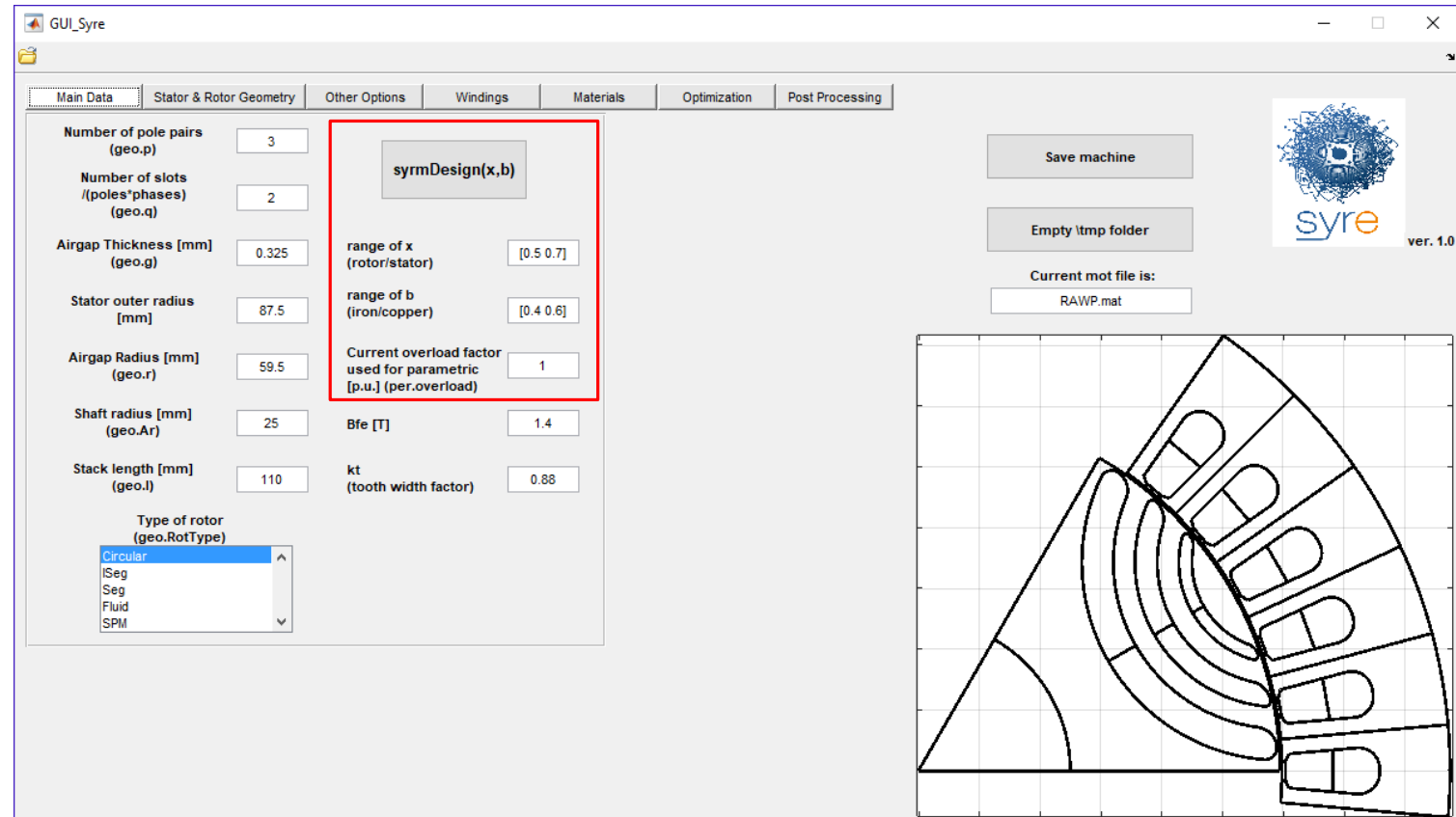
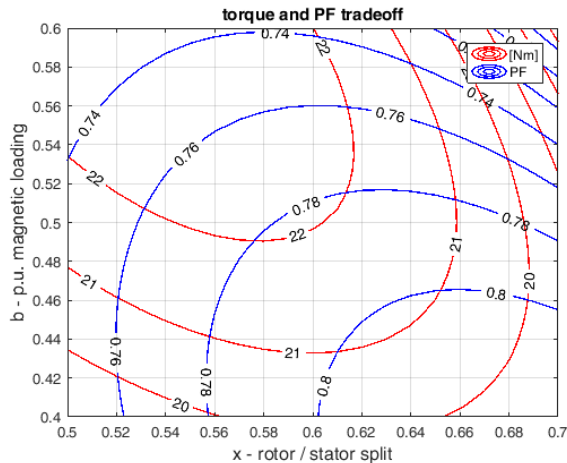
Buttons: Save machine, Empty tmp folder, Current mot file is: RAWP.mat



# syrmDesign(x,b): Design Equations

The preliminary design feature of SyR-e evaluates the **torque and power-factor contours** of the SyR machine as a function of the two key parameters:

- **$x$  (rotor/stator split)**
- **$b$  (iron/copper split)**



# Example Results

Torque and PF have different trends

Any point of the plane corresponds to one design

Left to right:

- rotor gets bigger, and slots get smaller

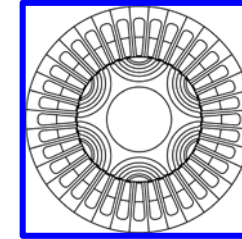
Up to down:

- Back iron, teeth and rotor iron get slimmer

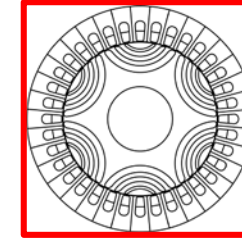
All the designs have the same peak flux density (stator back iron and rotor flux carriers)

Stator teeth size can be calibrated via the dedicated parameter  $k_t$

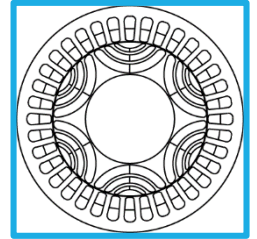
$x = 0.55, b = 0.55$



$x = 0.68, b = 0.55$



$x = 0.68, b = 0.42$



**syrmDesign(x,b)**

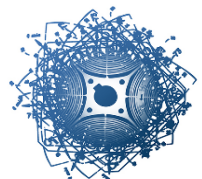
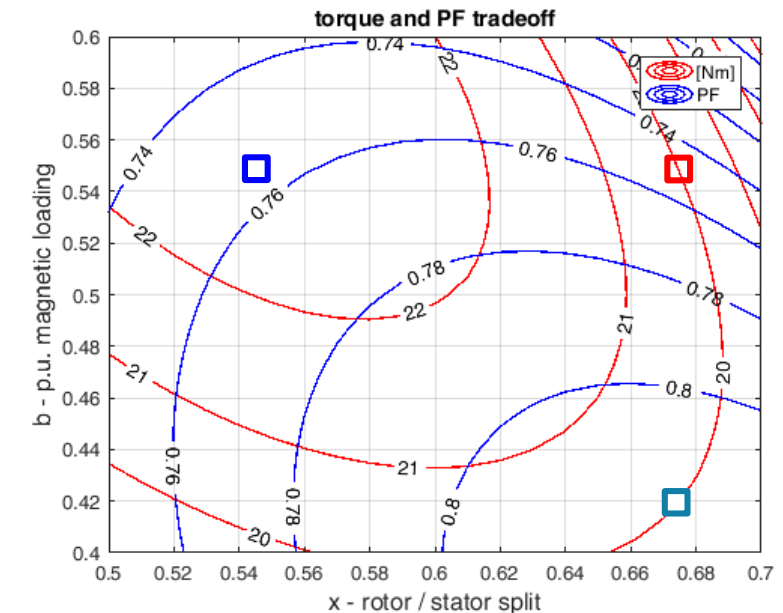
range of x (rotor/stator)

range of b (iron/copper)

Current overload factor used for parametric [p.u.] (per.overload)

**Bfe [T]**

**kt (tooth width factor)**

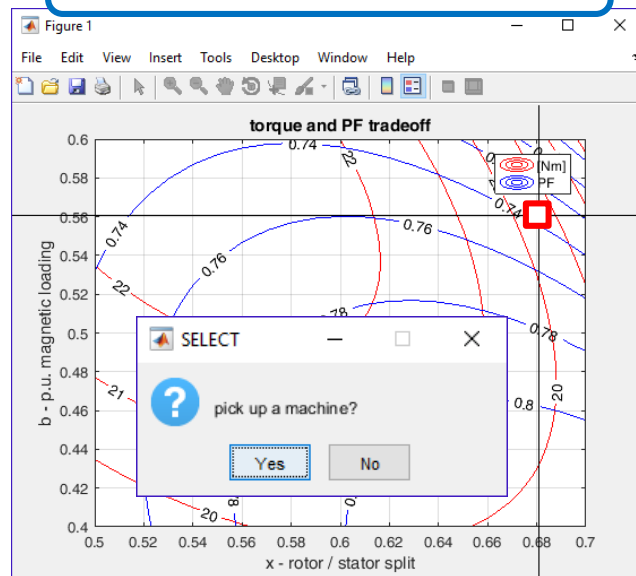


# Preliminary Design Method

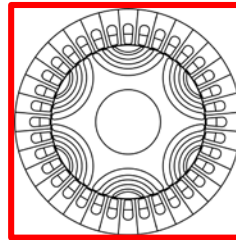
Stack size and airgap:  $D, L, g$   
Other design Specs:  $p, B_{Fe}, k_{Cu} \dots$   
Thermal loading:  $k_j [\text{W}/\text{m}^2]$



Torque( $x, b$ ) and PF( $x, b$ )



Selected design → FEA



$x, b = 0.68, 0.55$

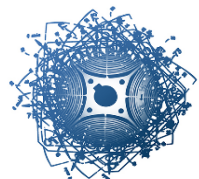
The core of design equations for SyR machines come from [2], with more recent refinement

The values  $x = 0.68, b = 0.55$  are compatible with the stator of the baseline IM

According to the model, the expected output is:

- Torque = 19.5 Nm
- Power Factor = 0.72

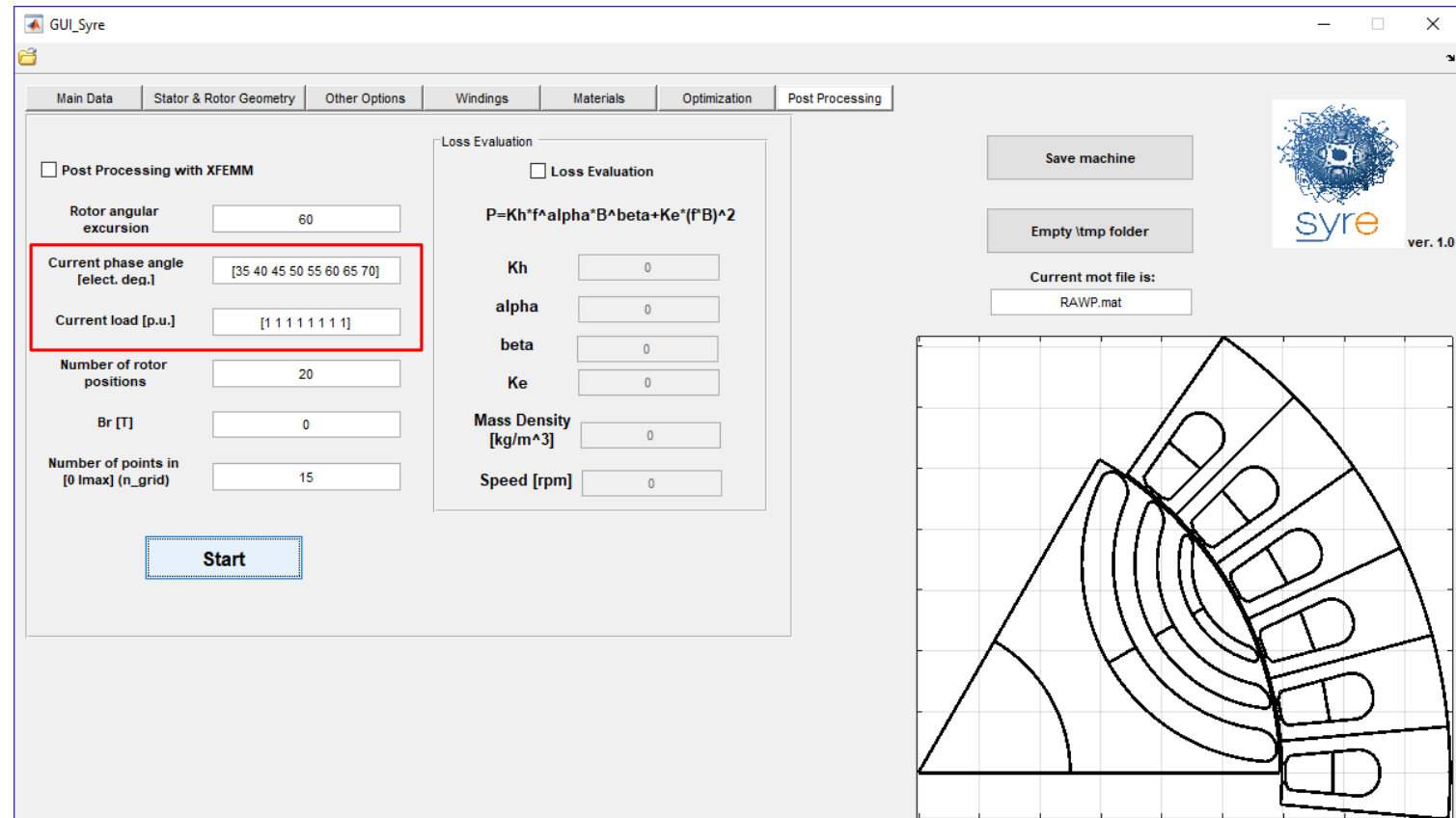
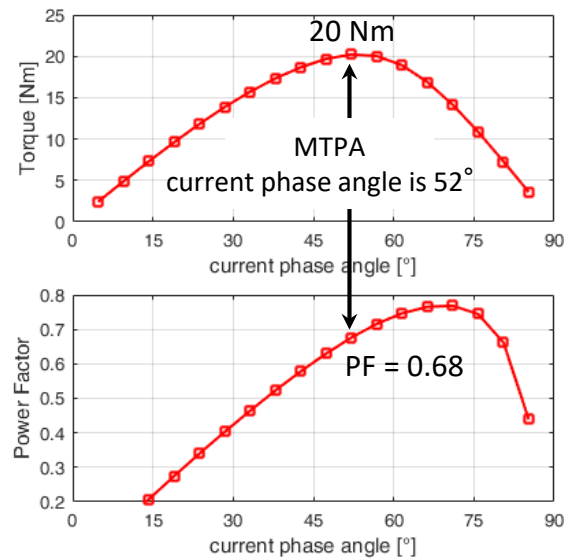
[2] Vagati, A.; Fratta, A.; Franceschini, G.; Rosso, P., "AC motors for high-performance drives: a design-based comparison," Industry Applications, IEEE Transactions on , vol.32, no.5, pp.1211,1219, Sep/Oct 1996



# Post Processing

`syrmDesign(x,b)` suggests a tentative current angle (here was  $56.6^\circ$ )

FEMM is launched for multiple current angles to evaluate the MTPA phase angle at  $i_0 = 1$  p.u.



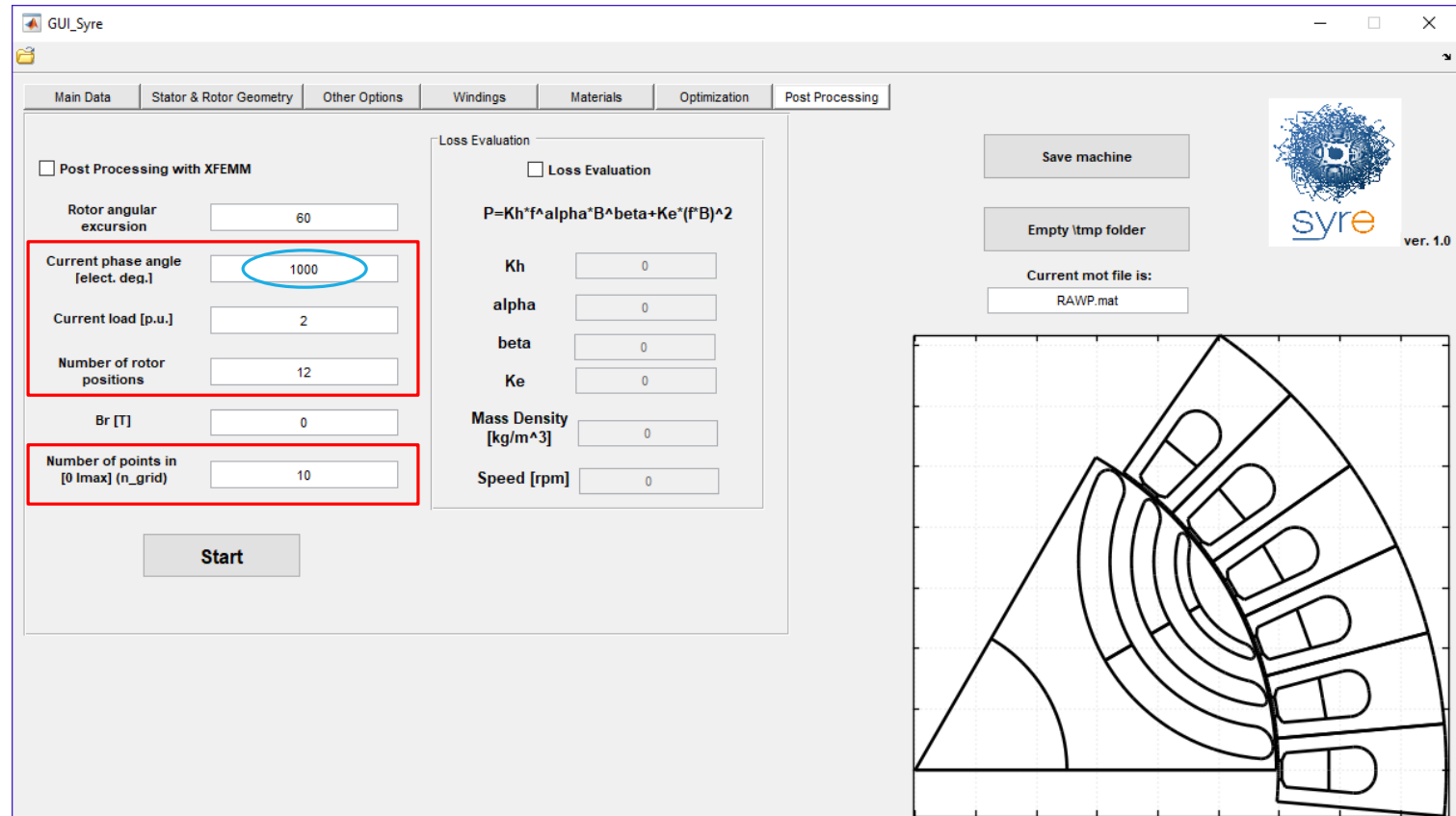
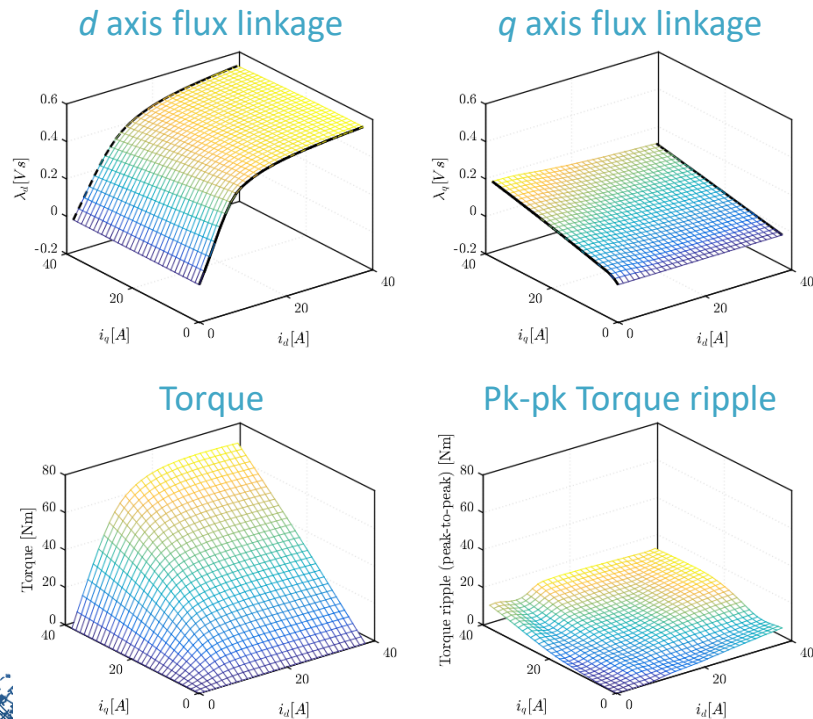
`syrmDesign` does not refer explicitly to the MTPA condition. For this design, the  $56.6^\circ$  phase angle suggested by `syrmDesign` is in between the max torque and max PF conditions



# Flux Maps

In a similar manner, output maps are produced:

- $dq$  flux maps
- Torque and torque ripple maps



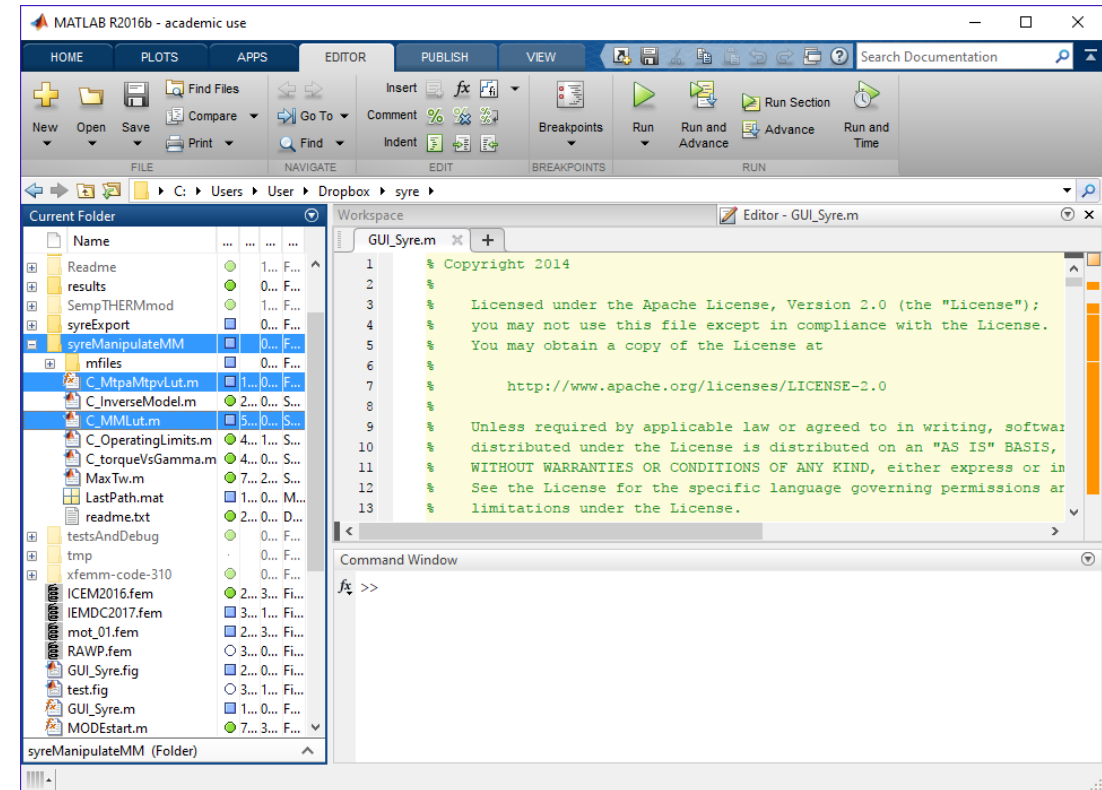
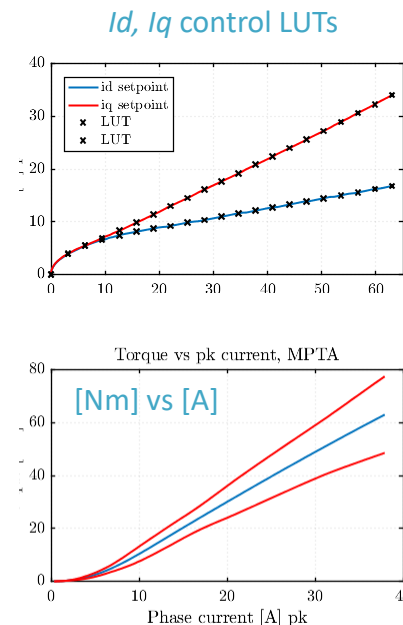
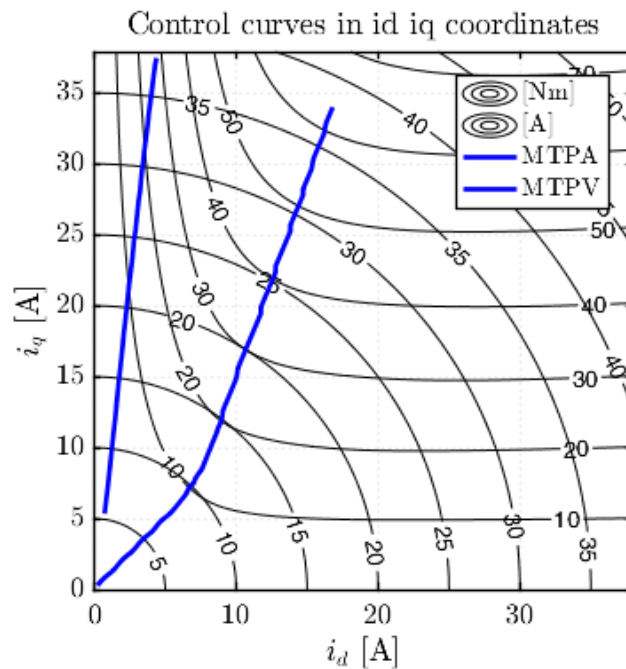
Phase angle = 1000 is an internal code to launch the flux map feature



# syreManipulateMM

Out of the GUI control, additional scripts are available for flux maps manipulation:

- Torque and torque factor [Nm/A] vs current
- MTPA and MTPV
- Torque and power vs speed profiles

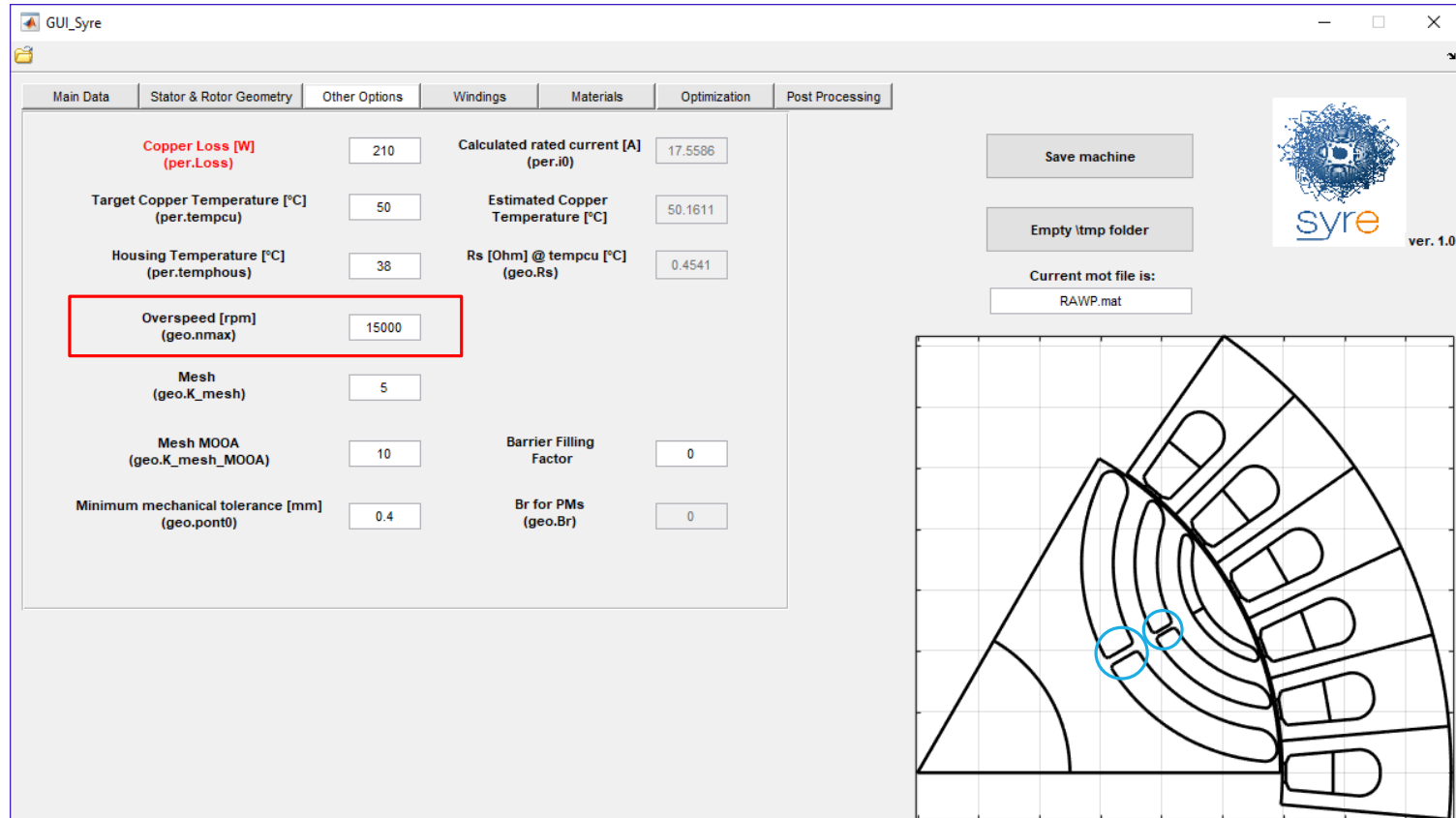


# Structural co-design

The over speed input field determines the size of the additional radial ribs included for rotor integrity

Those are evaluated assuming that each rib supports all the centrifugal load downstream of the respective layer, and imposing a stress limit of 285 Mpa (yield strength of the selected silicon steel M600-50A)

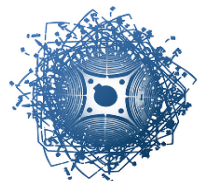
This feature is on-line during design optimization



The screenshot displays the GUI\_Syre software interface, which is used for structural co-design. The interface is divided into several tabs: Main Data, Stator & Rotor Geometry, Other Options, Windings, Materials, Optimization, and Post Processing. The Main Data tab is currently active, showing various input fields and calculated values.

Parameter	Value
Copper Loss [W] (per.Loss)	210
Calculated rated current [A] (per.i0)	17.5586
Target Copper Temperature [°C] (per.tempcu)	50
Estimated Copper Temperature [°C]	50.1611
Housing Temperature [°C] (per.temphous)	38
Rs [Ohm] @ tempcu [°C] (geo.Rs)	0.4541
<b>Overspeed [rpm] (geo.nmax)</b>	<b>15000</b>
Mesh (geo.K_mesh)	5
Mesh MOOA (geo.K_mesh_MOOA)	10
Barrier Filling Factor	0
Minimum mechanical tolerance [mm] (geo.pont0)	0.4
Br for PMs (geo.Br)	0

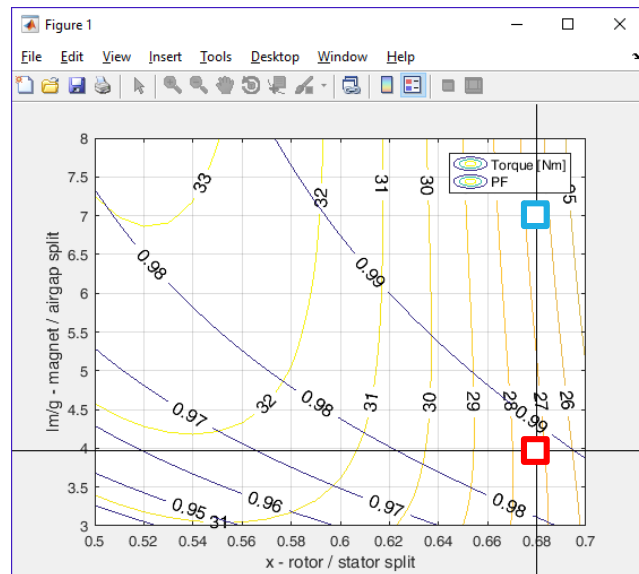
On the right side of the interface, there are buttons for "Save machine", "Empty tmp folder", and a field for "Current mot file is:" with the value "RAWP.mat". Below these buttons is a diagram of a rotor cross-section, showing the stator and rotor geometry. The diagram is a semi-circular cross-section of a motor, with the stator on the left and the rotor on the right. The rotor has several radial ribs, and the diagram shows the distribution of the magnetic field lines.



# Surface-Mounted PM Machines

The same preliminary design approach applies to SPM machines

In place of  $b$ , the parameter on the y-axis here is the magnet to airgap length ratio ( $l_m/g$ )



The GUI\_Syre software interface displays various design parameters for a surface-mounted permanent magnet machine. The parameters are organized into tabs: Main Data, Stator & Rotor Geometry, Other Options, Windings, Materials, Optimization, and Post Processing. The Main Data tab is active, showing parameters such as Number of pole pairs (geo.p), Number of slots / (poles\*phases) (geo.q), Airgap Thickness (mm) (geo.g), Stator outer radius (mm), Airgap Radius (mm) (geo.r), Shaft radius (mm) (geo.Ar), Stack length (mm), range of x (rotor/stator), range of b (iron/copper), Current overload factor used for parametric [p.u.] (per.overload), Bfe [T], and kt (tooth width factor). A red box highlights the 'syrmDesign(x,b)' button. Below the parameters, there are two circular diagrams showing the rotor and stator geometry. A red box highlights the rotor diagram, and a blue box highlights the stator diagram. On the right side of the interface, there are buttons for 'Save machine', 'Empty tmp folder', and 'Current mot file is: SPM.mat'. A 3D model of the machine is shown in the bottom right corner.

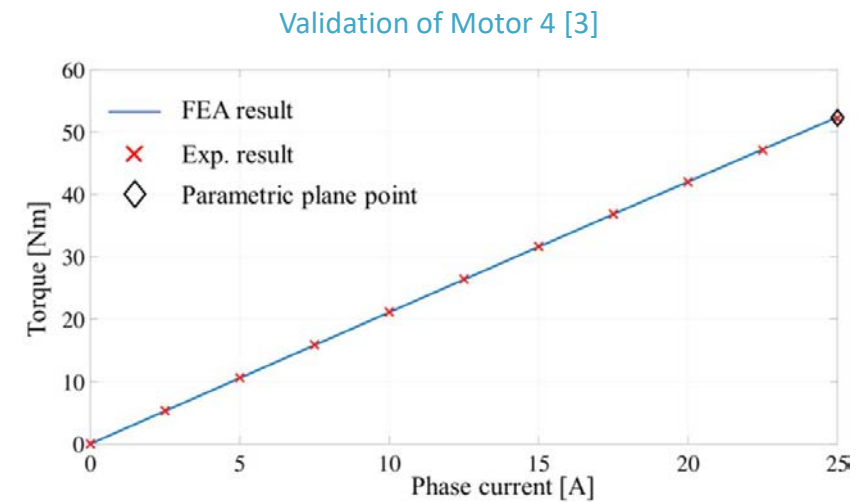
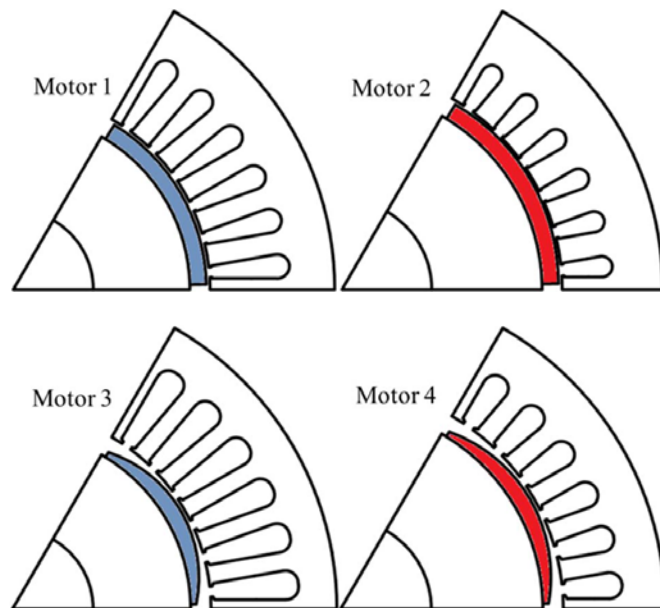
[3] Lu, C.; Ferrari, S.; Pellegrino, G.; Bianchini, C.; Davoli, M.; "Parametric Design Method for SPM Machines Including Rounded PM Shape," 2017 IEEE Energy Conversion Congress and Exposition (ECCE), Cincinnati, OH

# Example of Results

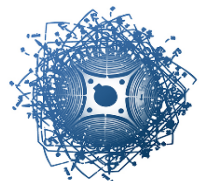
Example of application of SyR-e to the design of SPM machines is in [3].

This covers:

- distributed winding types
- Shaping of the magnet length (examples #3 and #4)



[3] Lu, C.; Ferrari, S.; Pellegrino, G.; Bianchini, C.; Davoli, M.; "Parametric Design Method for SPM Machines Including Rounded PM Shape," 2017 IEEE Energy Conversion Congress and Exposition (ECCE), Cincinnati, OH



# Design Optimization

The **Optimization** tab runs MODE optimization

Key input fields:

- Population size and # of generations
- Inputs are selectable
- Goals are selectable

Fast FEA evaluation:

- 5 rotor positions per candidate, random current phase angle during optimization
- Accurate re-evaluation of the Pareto-optimal designs (30 rotor positions)

GUI\_Syre

Main Data | Stator & Rotor Geometry | Other Options | Windings | Materials | Optimization | Post Processing

Optimization Parameters

Max number of generations: 60

Population size: 60

Time step settings during MODE

# of simulated positions (geo.nsim\_MOOA): 5

Rotor angular excursion [elect. deg.]: 30

Time step settings at Pareto front re-evaluation

Simulated positions (geo.nsim\_singt): 30

Rotor angular excursion [elect. deg.]: 60

XFEMM/FEMM

☐ XFEMM

☒ Remove TMP File (geo.RemoveTMPfile) it only works with

Bounds

tip angle barrier #1 [p.u.] (bounds\_dalpha\_1): [0.25 0.5] ☒

other barriers [p.u.] (bounds\_dalpha): [0.17 0.5] ☒

hc [p.u.] (bounds\_hc): [0.2 1] ☒

dx [p.u.] (bounds\_Dx): [-0.75 0.75] ☒

Airgap thickness [mm] (bounds\_g): [0.4 0.8] ☐

Br [T] (bounds\_Br): [0.2 0.8] ☐

Airgap radius [mm] (bounds\_xr): [52 78] ☐

Tooth width [mm] (bounds\_wt): [3.8 6.3] ☐

Tooth length [mm] (bounds\_lt): [15 22.5] ☐

Stator Slot Open [p.u.] (bounds\_acs): [0.2 0.3] ☐

Tooth Tang Depth [mm] (bounds\_ttd): [0.8 1.2] ☐

Current phase angle [elect. deg.] (bounds\_gamma): [40 75] ☐

Optimize

Current overload factor used for optimization [p.u.] (per.overload): 2

Optimization Objectives

Torque [Nm] (per.min\_exp\_torque) (NEGATIVE VALUE): ☒ -10

Peak-to-peak torque ripple [Nm] (per.min\_exp\_torque): ☒ 8

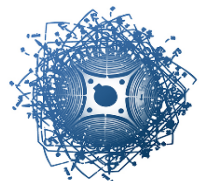
Max Cu mass [kg] (per.max\_Cu\_mass): ☐ 0

Save machine

Empty tmp folder

Current mot file is: RAWP.mat

syre ver. 1.0



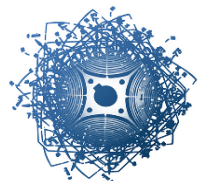
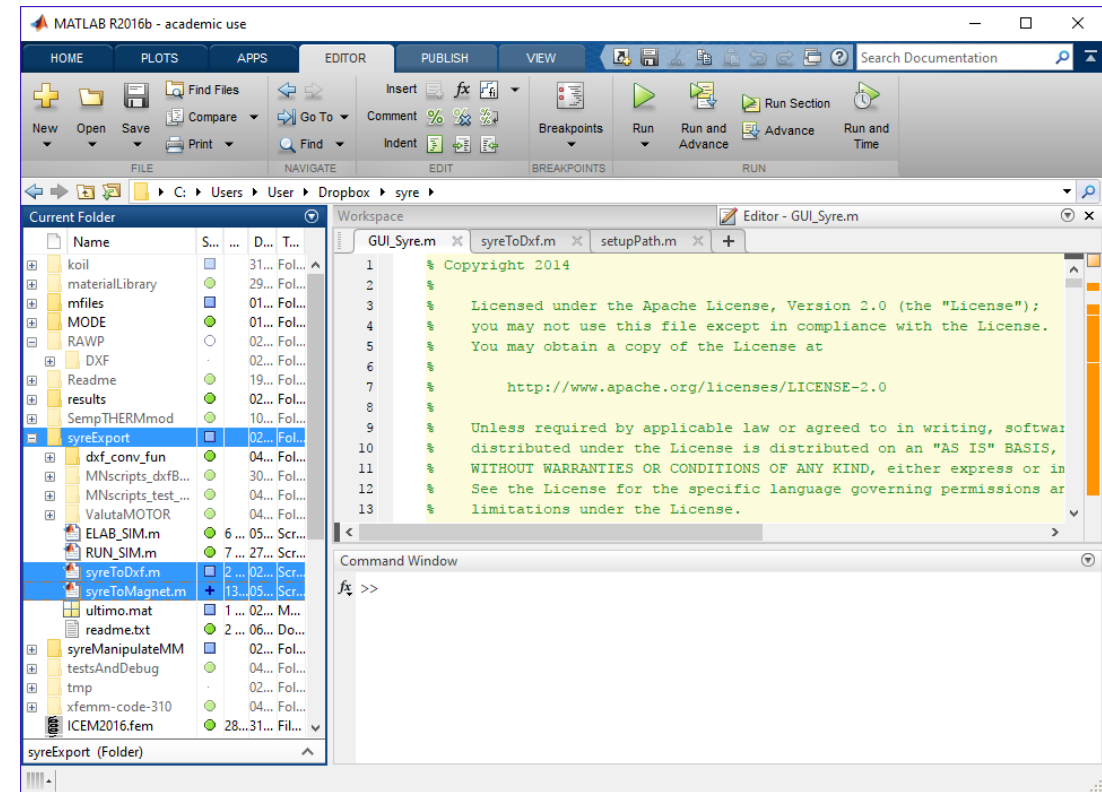
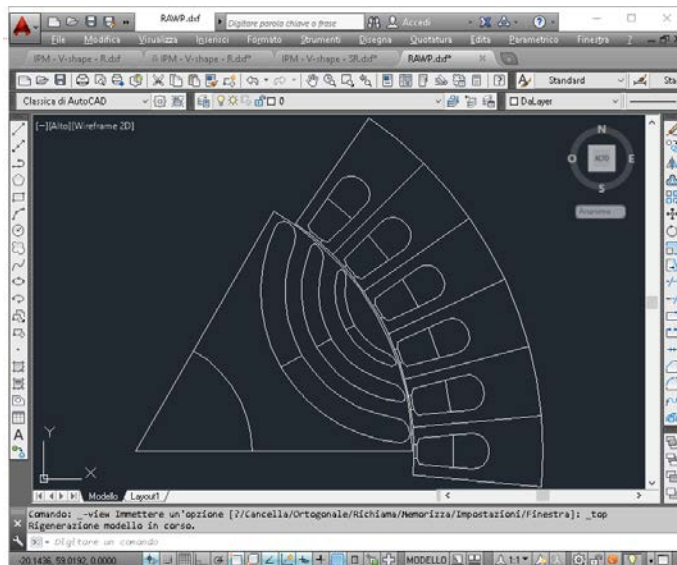
# syreExport

Syre models are exportable to:

- Autocad (dxf format)
- Magnet by Infolytica

Other CAD environments are easily reachable using the exported dxf model

Example of dxf model





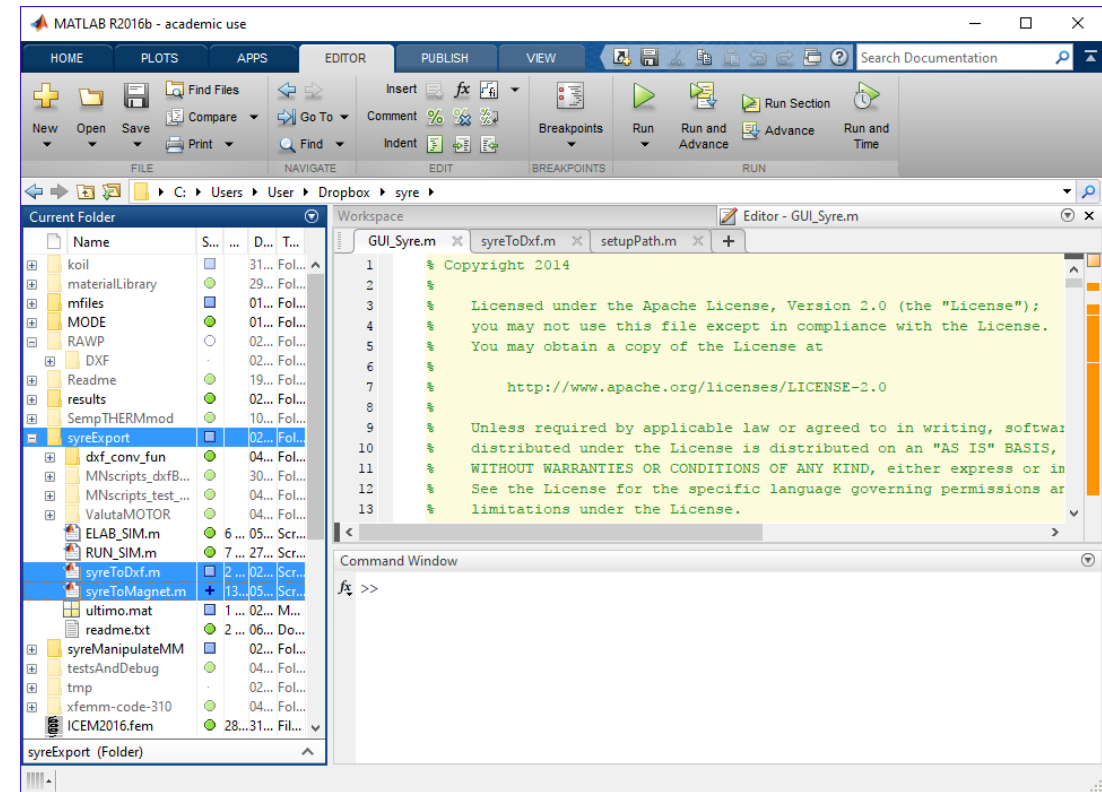
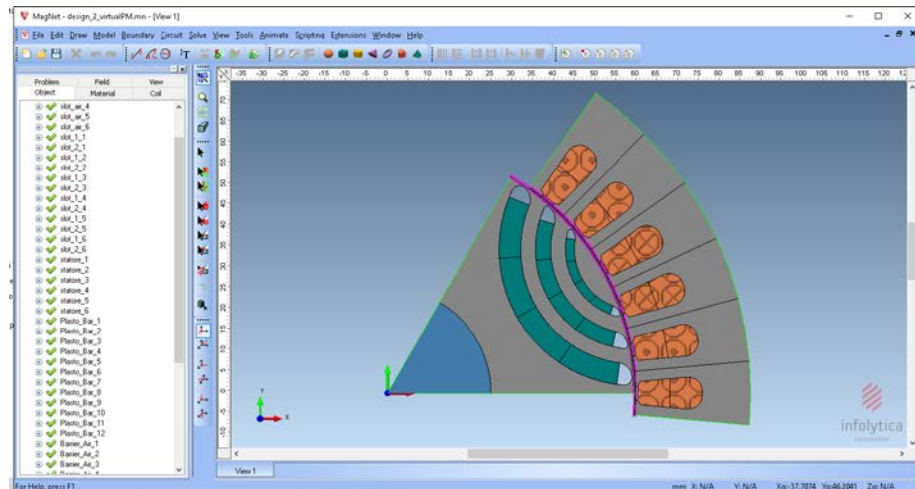
# Export to Infolytica\Magnet

Commercial FEA tools are needed for:

- Transient simulation (e.g. solid components such as PMs and solid conductors)
- Accurate evaluation of iron loss

Other platforms are easily reachable using the provided scripts as template

Example of Magnet model



# Future

Improve the structural co-design using:

- beam network approach for rotor of the SyR type, aiming at joint optimization of the magnetic and structural aspects of the rotor
- Sleeve sizing equations for SPM rotors

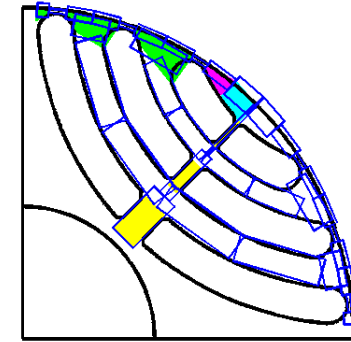
Extend `syrmDesign(x,b)` approach to fractional slot configurations

Validate iron loss estimate using FEMM (static FEA)

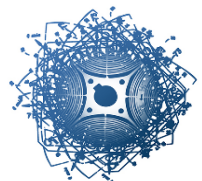
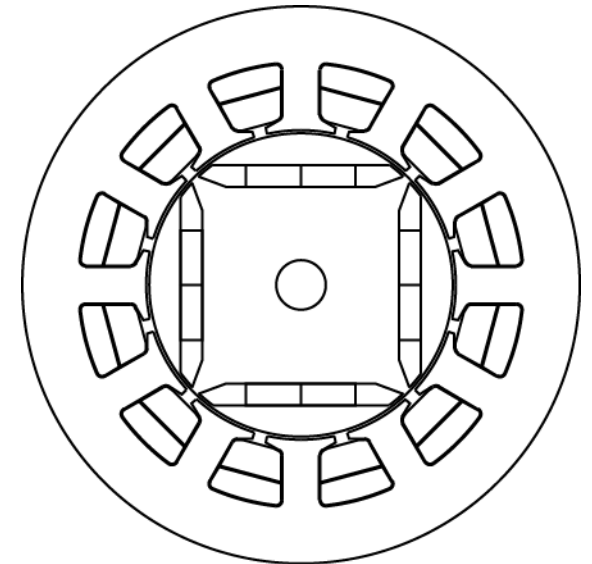
Include typical IPM rotor geometries (single layer V-shape or I-shape, triangular)

Improve documentation

Beam model for centrifugal stress analysis



Example of IPM rotor geometry





# Conclusion

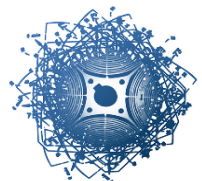
SyR-e is a set of Matlab scripts + GUI for design and FEA evaluation of synchronous reluctance, PM-assisted synchronous reluctance and surface-mounted PMSMs

The presented key features of SyR-e are:

- Fast FEA modeling, from text or GUI input
- Fast FEA evaluation of the selected machine design
- Design equations, for preliminary design
- Thermal and structural aspects includes, via approximated models
- Design optimization

Although the use of Matlab poses some limitation to SyR-e's general availability and reach, it is ambitiously intended as an open platform for comprehensive machine design, A to Z

**Work is ongoing (as usual ..)**



# Electrical machine analysis using free software - Section 4

## SyR-e : Synchronous Reluctance Evolution Automated Design of Surface Mounted PM Motors

ECCE 2017 TUTORIAL

Francesco.cupertino@poliba.it



Politecnico  
di Bari



syre

# Outline

- Optimization algorithms
- Optimization of permanent magnet motors using SyR-e
- Example of design optimization
- Conclusions and future work



# Search and optimization

## **Problem:**

Let  $\mathcal{R}$  be the domain of allowable values for a vector “ $\mathbf{x}$ ”.

Find the values of a vector “ $\mathbf{x}$ ” element of  $\mathcal{R}$  that minimizes a scalar valued loss function  $L(\mathbf{x})$ .

Such a problem can be encountered in all areas engineering, physical sciences, business or medicine

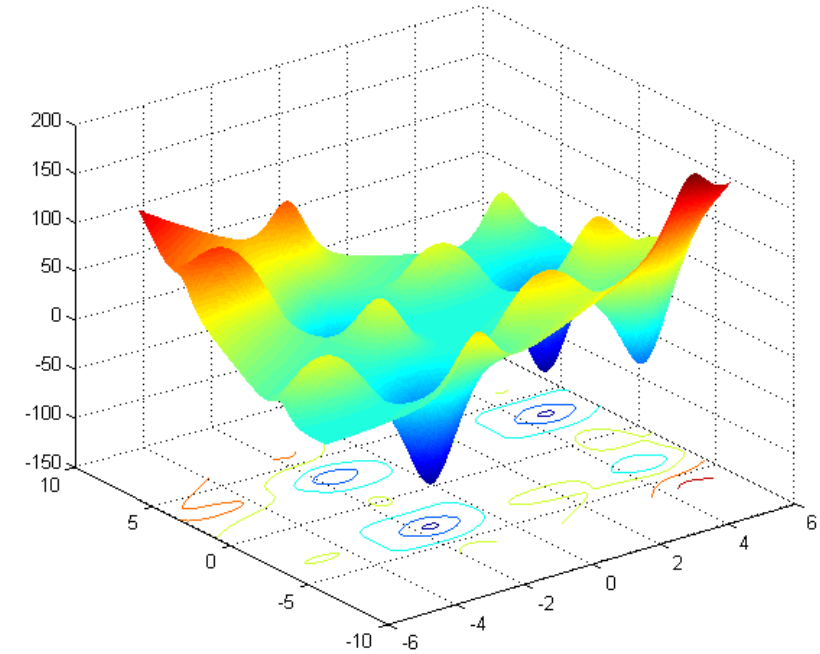
Doing the most with the least is the essence of the optimization objective

# Local vs Global optimization

Due to the limitation of almost all the optimization algorithm it is only possible to approach a local optimum.

The local minimum may be still a fully acceptable solution for the resources available (human time, money, computer time, ...) to be spent in the optimization.

Some algorithms are sometimes able to find global solutions (e.g. simulated annealing, genetic algorithms, and differential evolution among others)



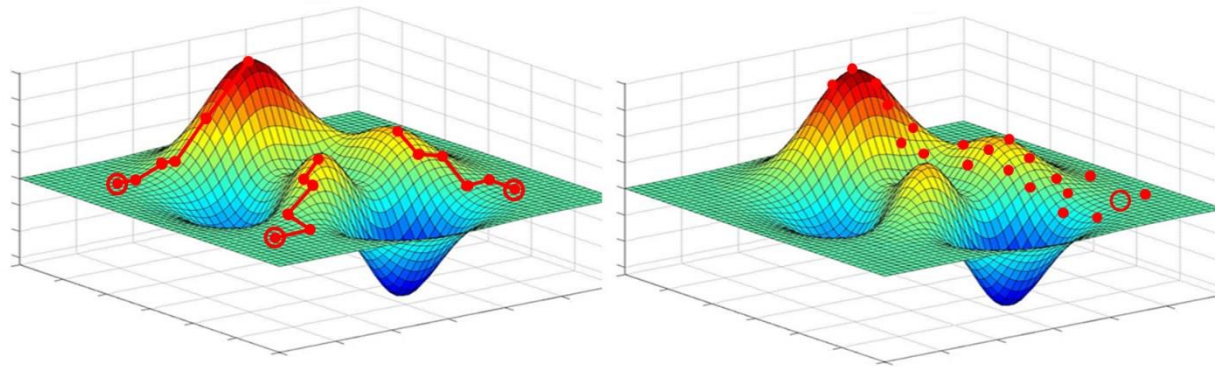
# Stochastic vs Deterministic optimization

There is stochastic optimization when:

1. There is noise in the loss function measurements.
2. There is a random choice to select the search direction.

These hypotheses contrast with standard deterministic optimization that assumes to have perfect information about the loss function (steepest descent) and its derivatives (Newton Raphson)

In most practical problems such an information is not available and deterministic algorithms are inappropriate



# No free lunch theorems

Wolpert and Macready (1997)

The no free lunch theorem for search and optimization applies to finite spaces and algorithms that do not resample points.

All algorithms that search for an extremum of a cost function perform exactly the same when averaged over all possible cost functions.

An algorithm that is effective in one class of problems is **guaranteed** to be ineffective on another class.

Just to get a feel for the theorem, consider the “needle in the haystack” problem: no search algorithm can beat blind random search.

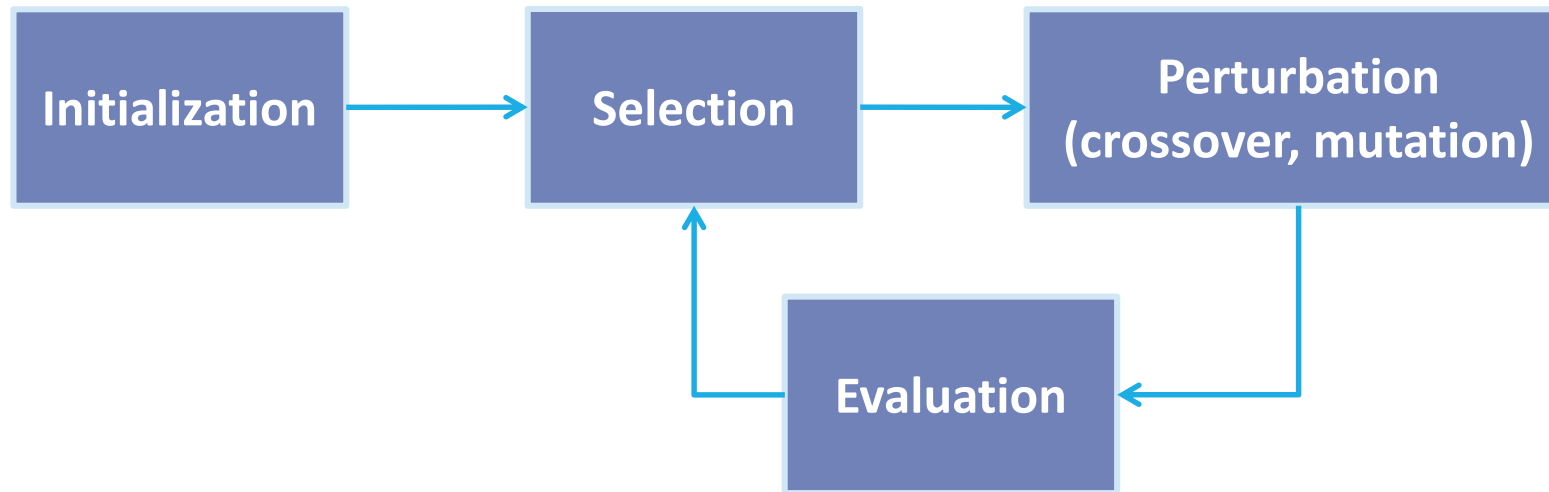


Pixel2013 @  
<https://pixabay.com/>





# General scheme of a population based GSA



A set of candidate solutions (population) is randomly selected and iteratively modified according to some probabilistic rules so to converge to the **global optimum** of the objective function with a certain degree of probability.

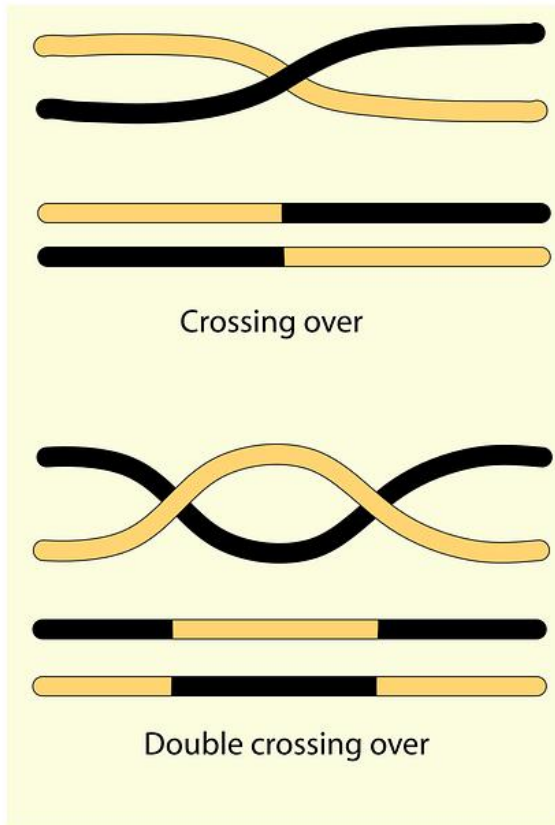
***Genetic Algorithms***, ***Differential Evolution*** and ***Simulated Annealing*** are popular algorithms adopting different **perturbation principles**



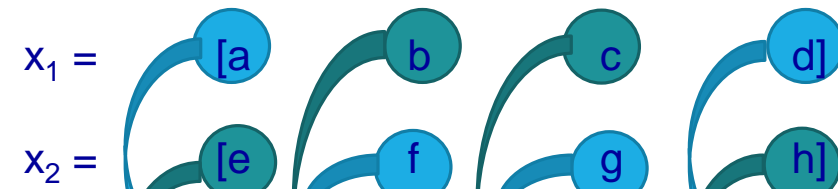


# Crossover

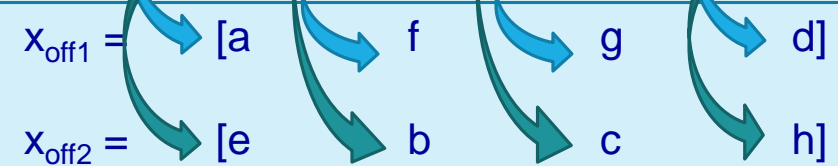
One or more parameters are exchanged between two of the selected solutions ( $x_1$  and  $x_2$ ), generating two new solutions ( $x_{\text{off}1}$  and  $x_{\text{off}2}$ )



**PARENTS**



**CHILDREN**



Crossover is a critical feature of optimization algorithms:

- It greatly accelerates search early in evolution of a population
- Crossover **exploits** the available information on the search problem (**local search**)

Clicker-Free-Vector-Images @  
<https://pixabay.com/>

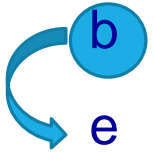


# Mutation

A random perturbation is applied to the potential solution:



Before mutation:  $x_1 = [a \quad b \quad c \quad d]$   
After mutation:  $x_{mut1} = [a \quad e \quad c \quad d]$



It allows to **explore new search areas** that would be not reachable with deterministic algorithms or using only crossover (**global search**).

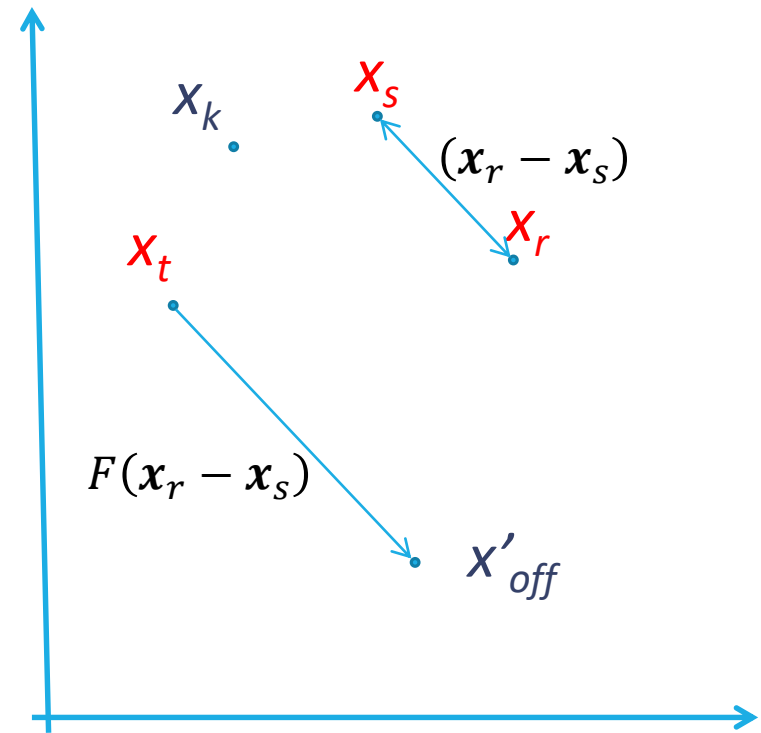
# Mutation in Differential Evolution algorithms

For each individual  $x_k$  of the parents population, other three parent individuals are randomly extracted:  $x_r$ ,  $x_s$ , and  $x_t$

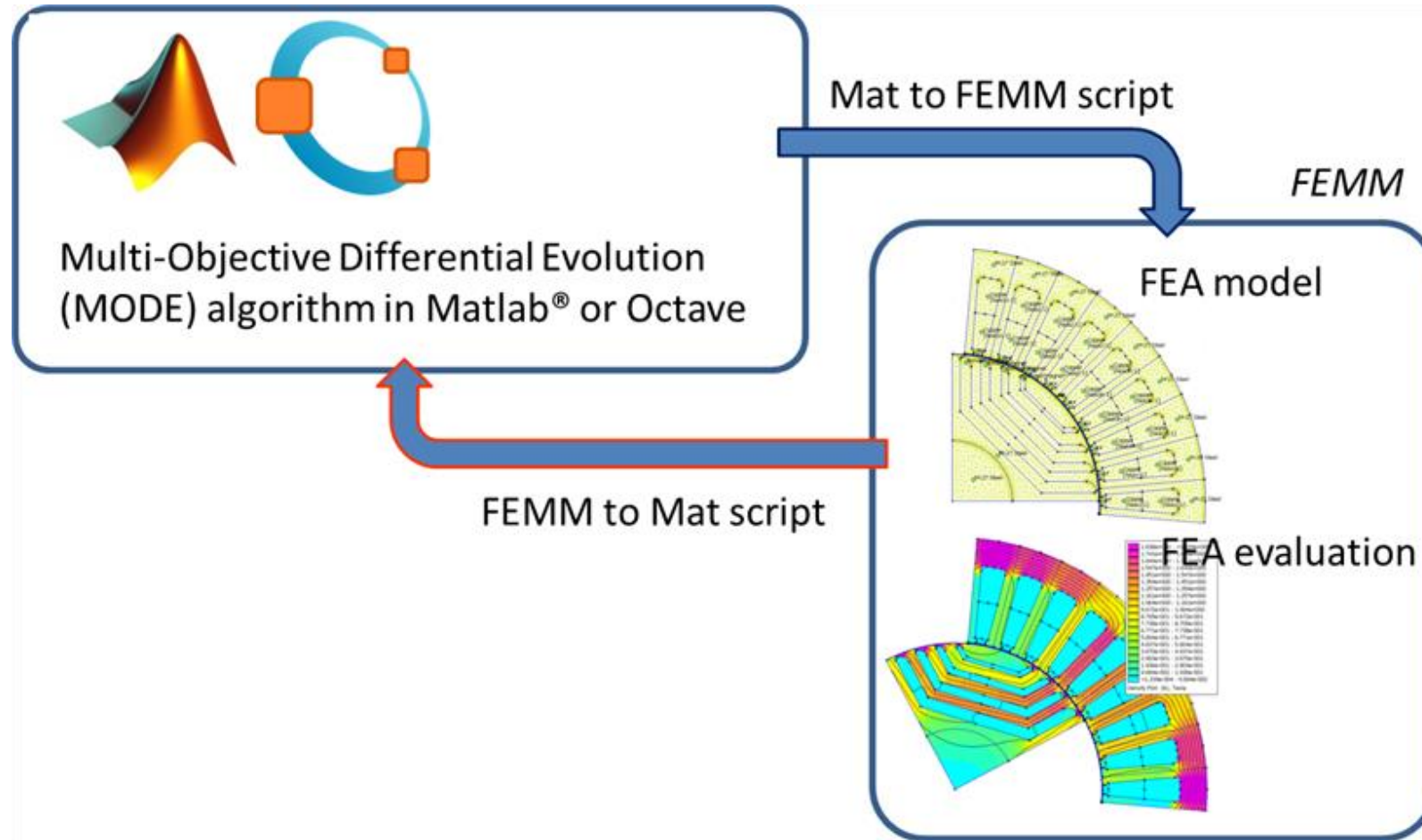
A provisional solution is generated by mutation as:

$$x'_{off} = x_t + F(x_r - x_s)$$

$F \in [0, 2]$  is a scale factor which controls the length of the exploration



# Data flow in the Evaluation stage



# Critical Issues

## GSA configuration:

- representation
  - population size, mutation rate, crossover rate
  - selection, deletion policies
  - crossover, mutation operators
- ... *practice and some “good luck”*

## Termination criteria

Solution is only as good as the fitness function (critical issue for most optimization algorithms)

Ref.

Cupertino F. et al. (2014) Design of Synchronous Reluctance Motors with Multiobjective Optimization Algorithms. IEEE Transactions on Industry Applications, vol 50: 3617-3627. doi: 10.1109/TIA.2014.2312540.

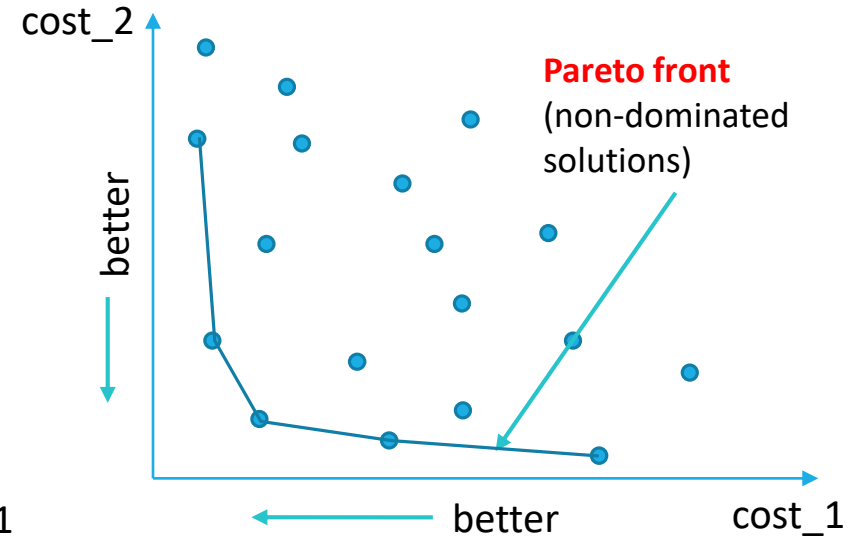
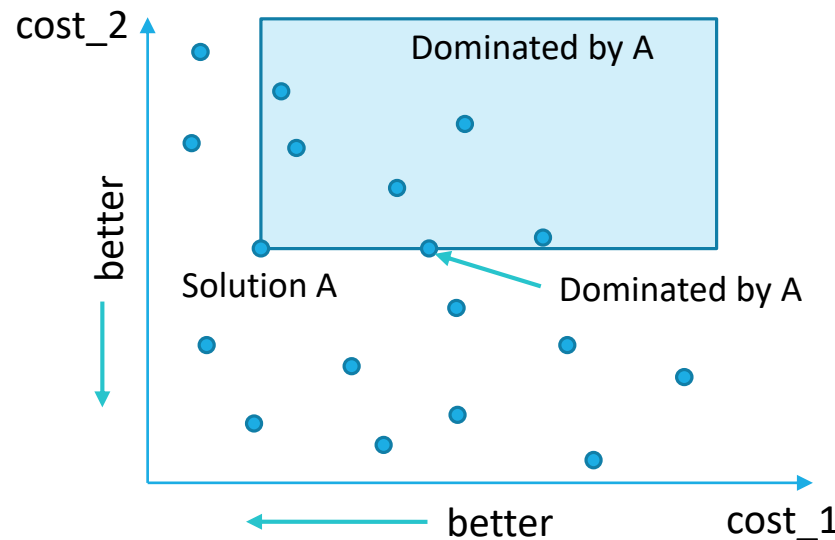
# From single to multi-objective: Pareto Dominance

How to define “better” in multi-objective design?

- Solution *A* **Pareto-dominates** solution *B*,
  - if *A* is at least as good as *B* in all objectives, and superior to *B* in at least one objective.

Pareto front (best options)

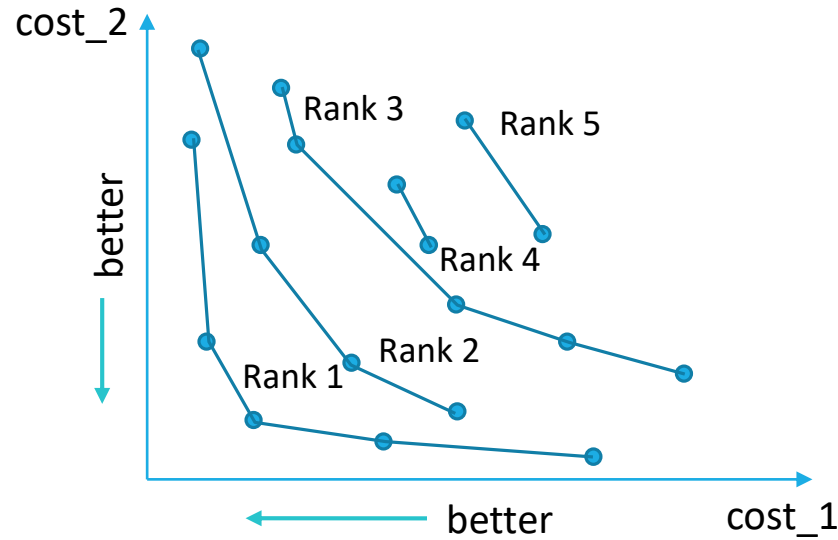
- Solutions not Pareto-dominated by others



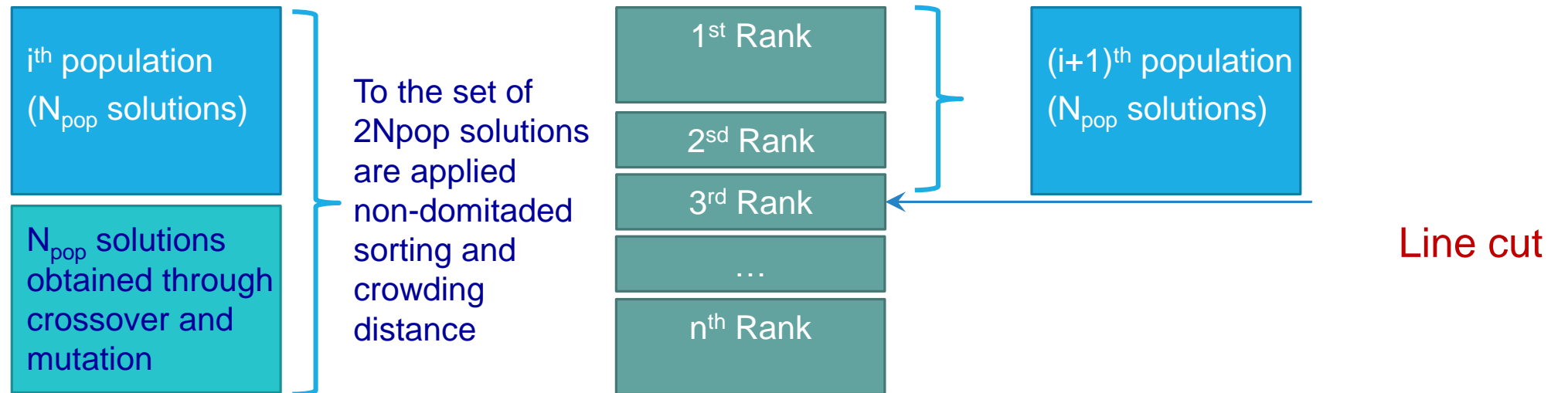
# Non-Dominated Sorting Genetic Algorithm

## *Pareto front rank*

- Rank 1: Pareto front of  $P$
- Rank 2: Pareto front of ( $P - \text{Rank 1}$ )
- Rank 3: Pareto front of ( $P - \text{Rank 1} - \text{Rank 2}$ )
- ...
- *Individual's Pareto front rank is used as fitness of the solutions*



# A single GSA iteration





# GSA implementations

## Matlab Global Optimization Toolbox uses NSGA-II

Ref. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II", IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 6, NO. 2, pp. 182-197, Apr. 2002.

A wide range of algorithms are available at MATLABCENTRAL website.

Among the others, special mention is given to GODLIKE - that includes NSGA-II, Simulated annealing and Differential Evolution

Ref. Rody Oldenhuis, "GODLIKE - A robust single-& multi-objective optimizer", 24 Jul 2009, [Online], available: <http://www.mathworks.it/matlabcentral/fileexchange/24838-godlike-a-robust-single-multi-objective-optimizer>

**SyR-e** includes a Multi-objective differential evolution algorithm called **DEMOWSA**  
(SA stands for self-adaptation)

Ref. A. Zamuda, J. Brest, B. Boskovič, and V. Žumer, "Differential Evolution for Multiobjective Optimization with Self Adaptation," in The 2007 IEEE Congress on Evolutionary Computation CEC 2007. IEEE Press, 2007, pp. 3617–3624.



# Optimization of permanent magnet motors using SyR-e



**SyR-e** was first developed for **Synchronous Reluctance** machines, but during its **e**volution, it gained several additional packages, including some features for PM machines



# Main Stator Geometric Parameters

## Slot/pole combinations

## Inner and outer stator diameters

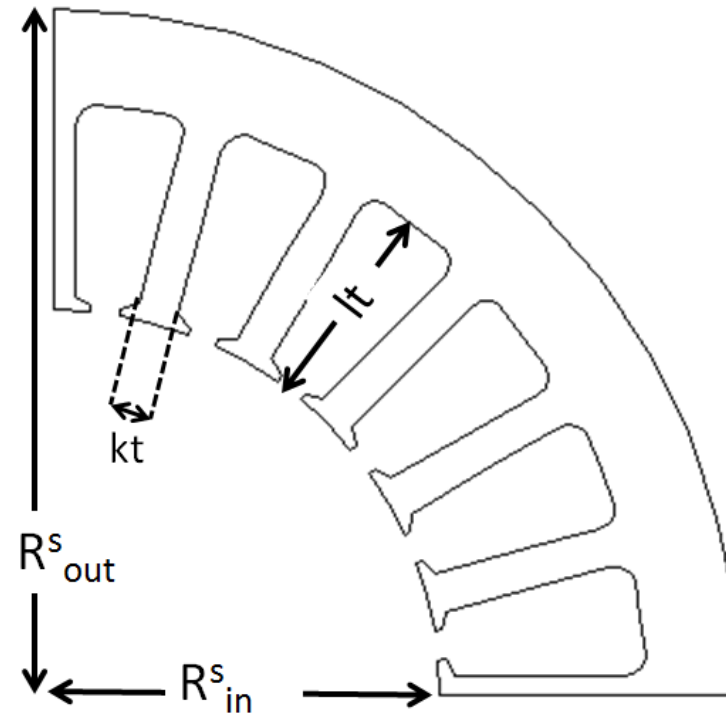
## Teeth length and thickness

$k_t$  and  $l_t$  depend on split ratio, lamination quality and desired saturation level

## Back-iron tickness

## Slots opening

Reducing slot opening generally improves airgap flux distribution



SyR-e where: GUI



syre

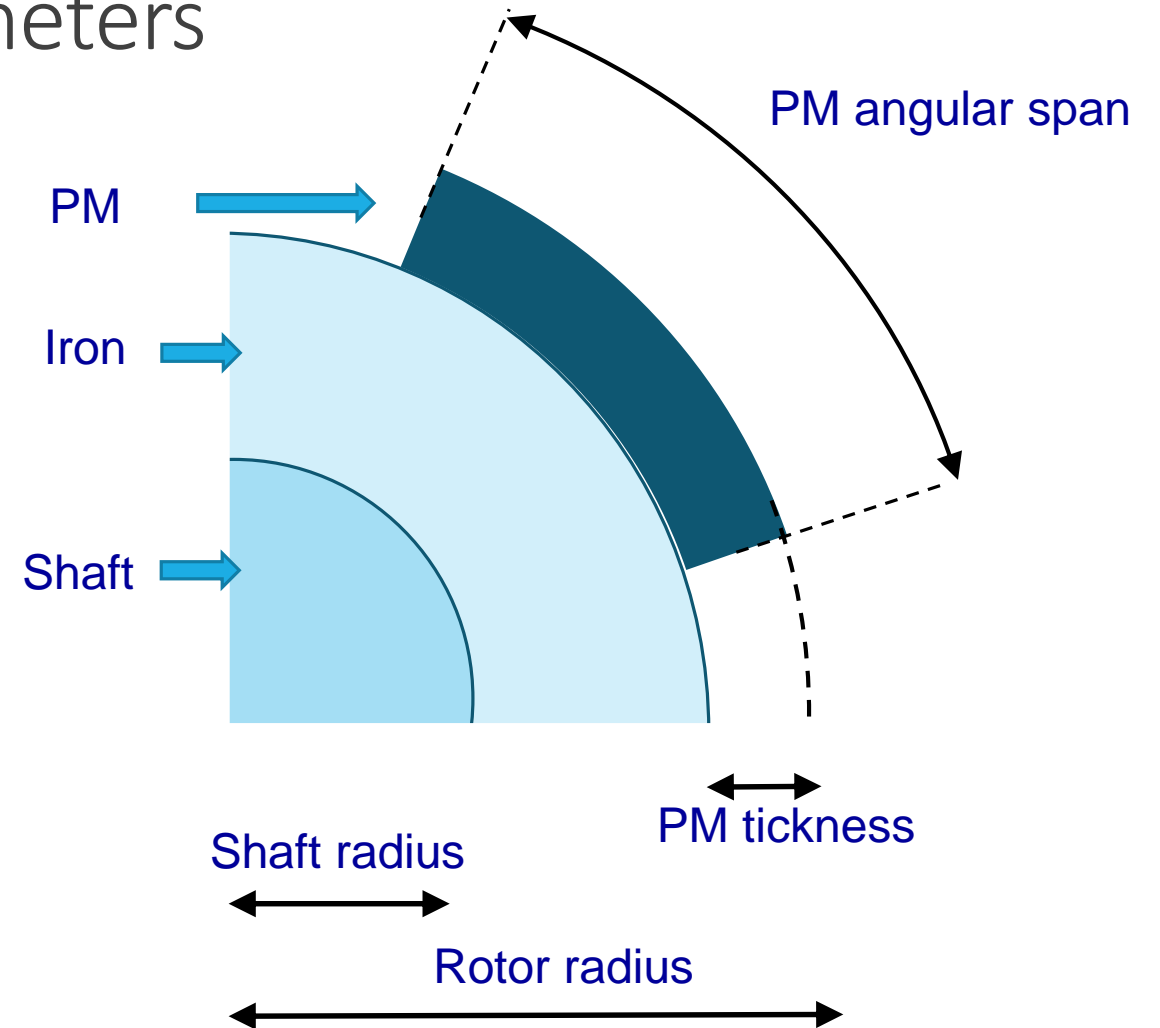
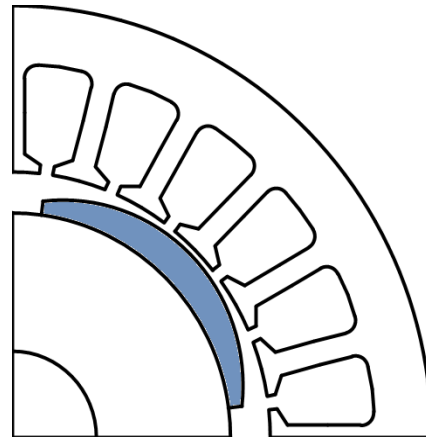
# Main Rotor Geometric Parameters

**Shaft radius**

**Rotor radius**

**Magnet tickness and angular span**

**Variable thickness is going to be integrated**



SyR-e where: GUI



syre

# Potential objectives

Potential objectives: **torque**, **torque ripple**, output power, efficiency, power factor, cost of the materials or weight of the active parts.

The torque maximization at given current and stator geometry inherently maximizes the torque per Joule loss ratio improving efficiency.

A comprehensive optimization of efficiency would require the evaluation of core losses at each function call of the MOOAs. We are working on this.

SyR-e where: the objectives (cost function) are defined in the m-file named Data0.m and then calculated in the m-file named FEMMfitness.m, after the execution of the FEA simulations.

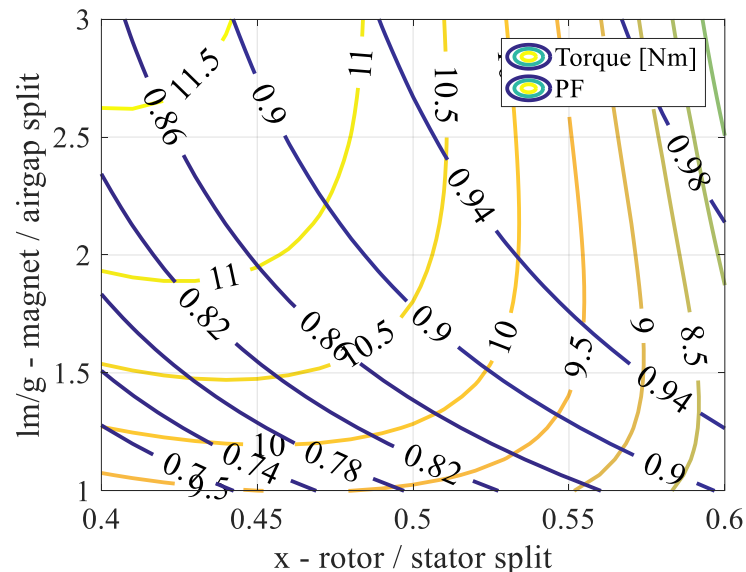
# Preliminary design using SyR-e: flow chart

START

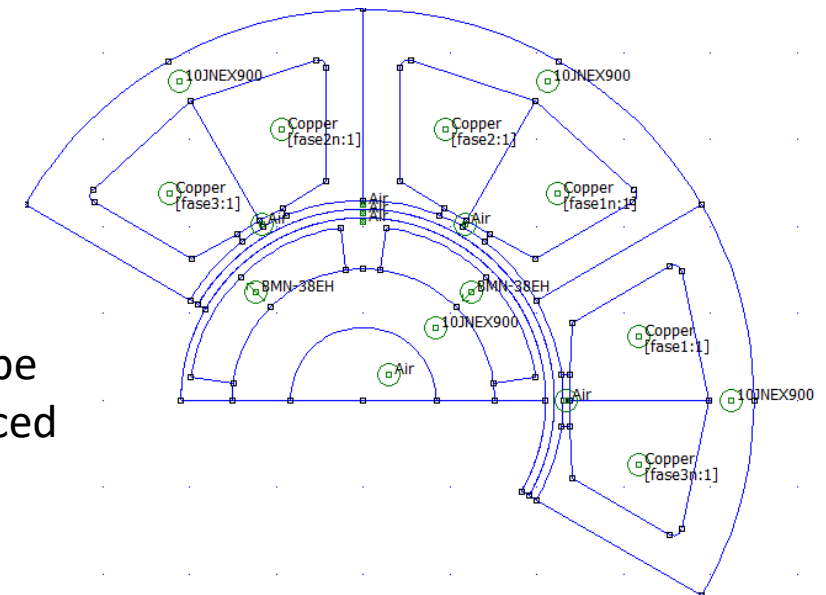
- ☐ Definition of Rotor/Stator split ratio;
- ☐ Definition of the ratio between magnet thickness and airgap ( $l_m/g$ );
- ☐ Calculation of the airgap magnetic flux density ( $B_g$ );
- ☐ Stator preliminary geometric sizing;
- ☐ Calculation of the d-axis stator flux ( $\lambda_{m,pole}$ );
- ☐ Calculation of the maximum current for a given stator joule losses ( $i_0$ );
- ☐ Calculation of stator inductances (magnetizing and leakage)
- ☐ Calculation of power factor and torque.

SyR-e where:  
syrmDesign(x,b) button in  
the GUI, then syrmDesign.m  
script.

END

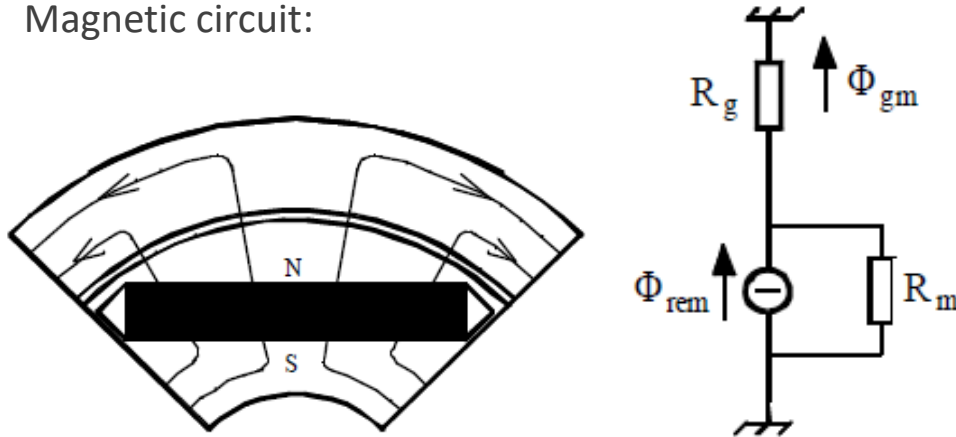


A FEMM model can be  
automatically produced  
for performance  
validation



# Magnetic flux density at the airgap

Magnetic circuit:



SyR-e where:  
syrmDesign(x,b) button in  
the GUI, then syrmDesign.m  
script.

**Procedure:**

$$R_m = \frac{l_m}{\mu_0 \mu_r a l}; R_g = \frac{k_c g}{\mu_0 a l}$$

$$B = \frac{R_m}{R_m + R_g} \cdot \phi_{rem} \cdot \frac{1}{a l} = \frac{l_m}{l_m + k_c \mu_r g} \cdot B_{rem} \cdot a l \cdot \frac{1}{a l} = \frac{B_{rem}}{1 + k_c \mu_r g / l_m}$$

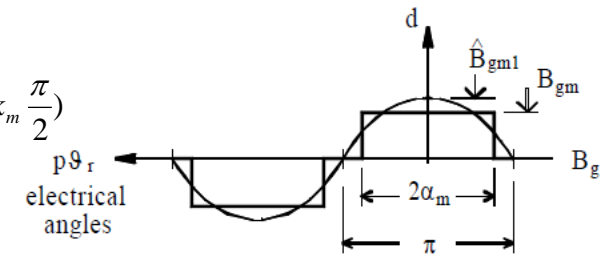
After B calculation, its fundamental component is derived using FFT analysis:

$$k_b = \frac{4}{\pi} \cdot B_{gm} \cdot \sin \alpha_m = \frac{4}{\pi} \cdot B_{gm} \cdot \sin(k_m \frac{\pi}{2})$$

$$k_m = \frac{Span_{magnet}}{a}$$

**Parameters:**

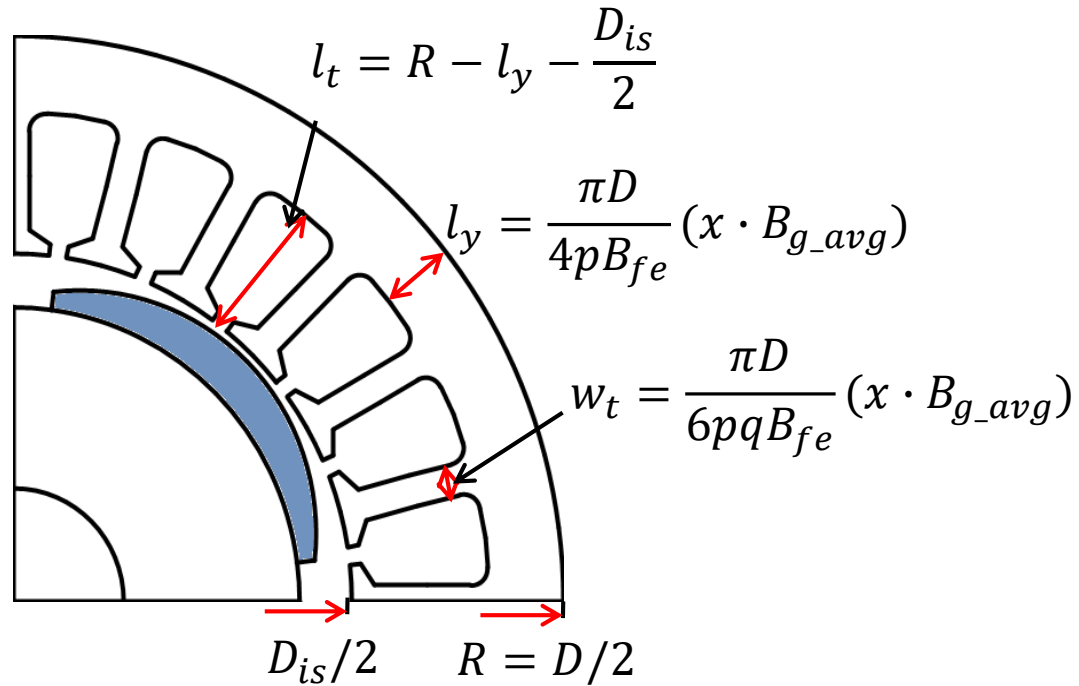
- $k_c$  = Carter coefficient
- $l_m$  = magnet tickness
- $l$  = stack length
- $g$  = airgap
- $a$  = polar pitch
- $B_{rem}$  = magnet remanence
- $\mu_r$  = magnet relative permeability
- $R_m$  = magnet reluctance
- $R_g$  = airgap reluctance



Ref: Vagati A et al. (2004) Design, analysis, and control of interior PM synchronous machines. Tutorial presented at IEEE IAS Annual Meeting, Seattle. (Chapter 3 – Bianchi)



# Stator preliminary sizing



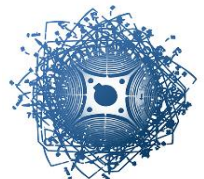
Defined the airgap flux density and the desired saturation level in the stator iron, the main stator geometric parameters can be determined.

Ref.:

Hanselman (2003) Brushless Permanent Magnet Motor Design, The Writers' Collective.

Pellegrino et. Al. (2016) The Rediscovery of Synchronous Reluctance and Ferrite Permanent Magnet Motors, Springer Brief.

SyR-e where:  
syrmDesign(x,b) button in  
the GUI, then syrmDesign.m  
script.





# Stator current from Joule losses- (kj)

## Procedure:

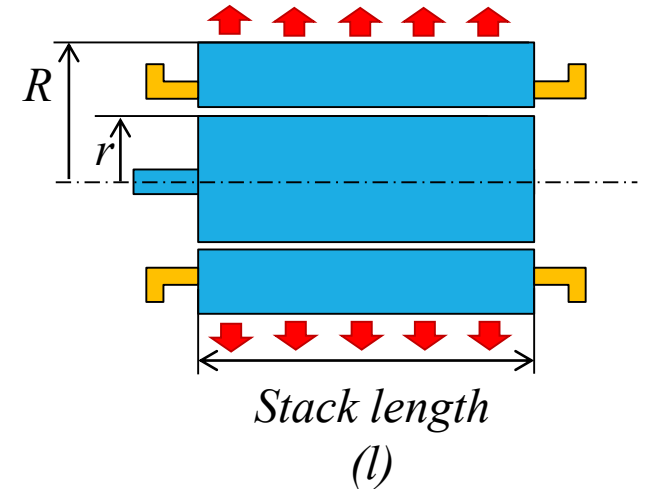
The electrical loading is set by the specific heat per square meter of outer stack surface.

$$k_j = \frac{P_{Cu}}{\text{stack cylinder surface}} = \frac{\frac{3}{2}R_s \cdot i_0^2}{2\pi \cdot Rl}$$

Stator resistance can be given as a function of stator geometric parameters, then the stator current can be derived as follows:

$$i_0 = \frac{\pi}{3N_s} R^{1.5} \sqrt{k_{cu} \frac{k_j}{\rho_{cu}} \frac{l}{l+l_{end}}} \cdot \sqrt{\frac{A_{slot}}{\pi R^2}}$$

SyR-e where:  
syrmDesign(x,b) button in  
the GUI, then syrmDesign.m  
script.



Parameters:

- $P_{Cu}$  Joule losses
- $R_s$  stator resistance
- $l$  = stack length
- $l_{end}$  = end turns length
- $A_{slot}$  = slot surface
- $k_{cu}$  = filling factor
- $N_s$  = turns in series
- $a$  = passo polare
- $\rho_{cu}$  = copper resistivity

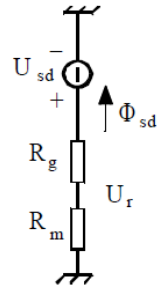
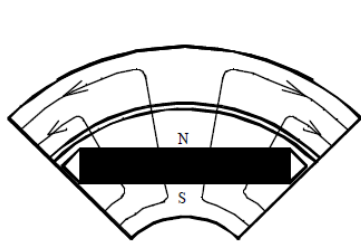
Ref:

Vagati et al. (1992) Design Criteria of High Performance Synchronous Reluctance Motors, IAS annual meeting.



# Inductance calculation

Magnetizing inductance:



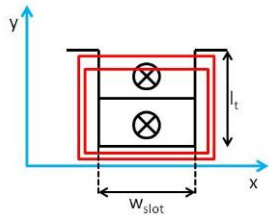
$$L_m = \frac{3}{2} \cdot \frac{8}{\pi} \cdot \left( \frac{k_w N_s}{p} \right)^2 \cdot \mu_0 \cdot L \cdot \frac{Dx/g}{l_m/g + k_c}$$

SyR-e where:  
syrmDesign(x,b) button in  
the GUI, then syrmDesign.m  
script.

Parameters:

- $l_m$  = magnet thickness
- $l$  = stack length
- $p$  = pole pairs
- $g$  = airgap
- $a$  = polar pitch
- $k_c$  = Carter coefficient
- $D$  = airgap diameter
- $k_w$  = winding factor
- $N$  = turns in series
- $k_2$  = factor that depends on winding configuration
- $Q$  = number of slots

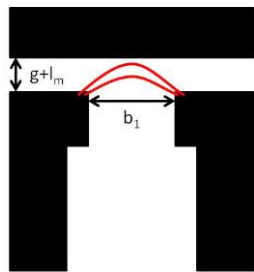
Slot leakage inductance:



$$L_{\text{slot,pole}} = \mu_0 \cdot l \cdot N^2 \cdot \frac{l_t}{a} \cdot \frac{1}{1 - \frac{B}{B_{fe}} k_t}$$

$$\frac{L_{\text{slot},n_l=2}}{L_{\text{slot},n_l=1}} = 1 - \frac{3}{4 \cdot Q_0}$$

Tooth tip inductance:



$$L_{\text{tip}} = \frac{12}{Q} \cdot \mu_0 \cdot l \cdot N_s^2 \cdot k_2 \cdot \frac{5 \left( \frac{g+l_m}{b_1} \right)}{5 + 4 \left( \frac{g+l_m}{b_1} \right)}$$

Ref:

Vagati A et al. (2004) Design, analysis, and control of interior PM synchronous machines. Tutorial presented at IEEE IAS Annual Meeting, Seattle. (Chapter 3 – Bianchi)  
Pyrhönen et al. (2009) Design of Rotating Electrical Machines, Wiley  
Cordovil and Chabu (2016) Analytical Calculation of Slot Leakage Inductance in Multiphase Electrical Machines, XXII International Conference on Electrical Machines (ICEM)



# Power factor and torque calculation

Internal power factor calculation:

$$\cos \varphi = \frac{\lambda_{m,pole}}{\sqrt{\lambda_{m,pole}^2 + (L_{pole} I_q)^2}}$$

where:

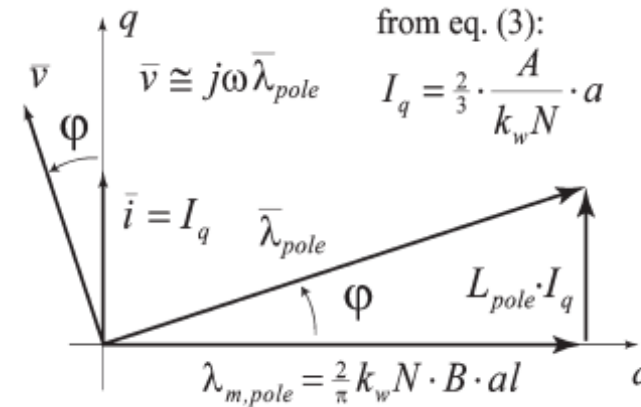
$$L_{pole} = L_{md} + L_{toothtip} + L_{slotleakage}$$

$$I_q = i_0 \cdot loadMax$$

Torque calculation:

$$T = \frac{3}{2} \cdot p \cdot \lambda_{m,pole} \cdot I_q$$

SyR-e where:  
syrmDesign(x,b) button in  
the GUI, then syrmDesign.m  
script.



Parameters:

- $L_{pole}$  = total phase inductance
- $A$  = electric loading
- $B$  = magnetic loading
- $k_w$  = winding factor
- $N$  = turns in series
- $a$  = polar pitch
- $\lambda_{m,pole}$  = stator flux due to the magnets
- $I_q$  = q-axis stator current
- $loadMax$  = overload factor



# Example: design specifications and initial settings

As a design example, an high speed actuator for aeronautic applications is considered.

	REQUIREMENT	UNIT	VALUE
MOTORING PHASE ONLY	Base Speed	[rpm]	50000
	Continuous Power	[kW]	50
	Overload Power	[kW]	75
	Continuous Torque	[Nm]	10
	Overload Torque	[Nm]	14
	Overload time	[min]	5
COOLING SYSTEM	Type	--	60/40 glycol/water
	Minimum coolant temperature	[°C]	-55 (-5 nominal ground)
	Maximum coolant temperature	[°C]	85 (40 nominal ground)
	Minimum ambient temperature	[°C]	-55
	Maximum ambient temperature	[°C]	85
	Liquid Flow Rate	[l/min/ kW]	0,45

Parameter	Initial setting
<b>Motor type</b>	Surface mounted PM with concentrated windings
<b>Stator radius (mm)</b>	45
<b>Axial length (mm)</b>	120
<b>Airgap + retaining sleeve [mm]</b>	3
<b>Allowed Joule losses [W]</b>	1000
<b>kj , joule losses / external stator surface [kW/m²]</b>	30
<b>Pole pairs</b>	2
<b>Stator slots</b>	6
<b>Iron material</b>	10JNEX900



# Preliminary design of PM machines design using SyR-e

1

Stack size and airgap:  $D, L, g$   
Other design Specs:  $p, B_{Fe}, k_{Cu} \dots$   
Thermal loading:  $k_j$

2

Torque( $x, lm/g$ ) and PF( $x, lm/g$ )

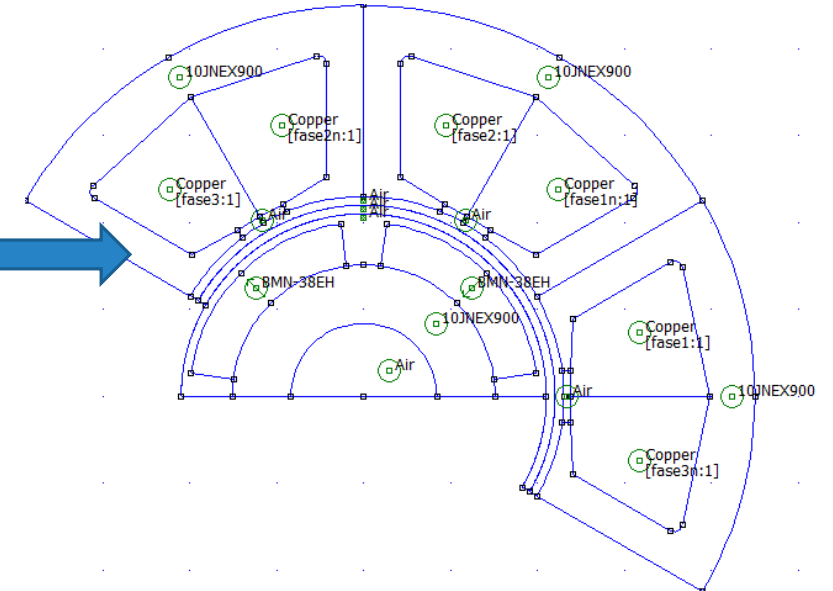
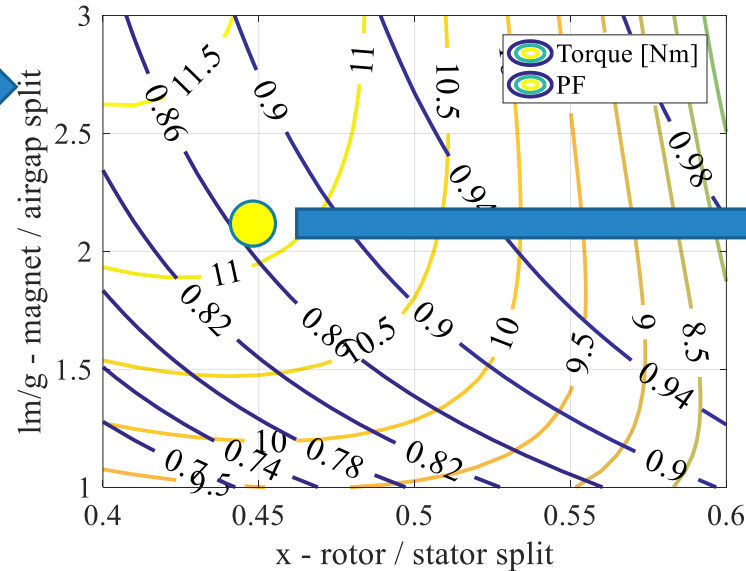
3

Automatically produced  
FEMM model

The screenshot shows the GUI\_Syre software interface with the following parameters:

- Number of pole pairs (geo.p): 2
- Number of slots / (poles\*phases) (geo.q): 0.5
- Airgap Thickness [mm] (geo.g): 3
- Stator outer radius [mm] (geo.R): 45
- Airgap Radius [mm] (geo.r): 20.5
- Shaft radius [mm] (geo.Ar): 8.4
- Stack length [mm] (geo.l): 120
- Type of rotor (geo.RotType): SPM
- range of x (rotor/stator): [0.4 0.7]
- range of b (iron/copper): [2 6]
- Current overload factor used for parametric [p.u.] (per.overload): 1
- Bfe [T]: 1.5
- kt (tooth width factor): 1

A button labeled **syrmDesign(x,b)** is visible, with an arrow pointing to the graph in step 2.



# SyR-e post processing tool

Preliminary design

Specs:

Rotor radius 20.3 mm

Magnet tickness 6.5 mm

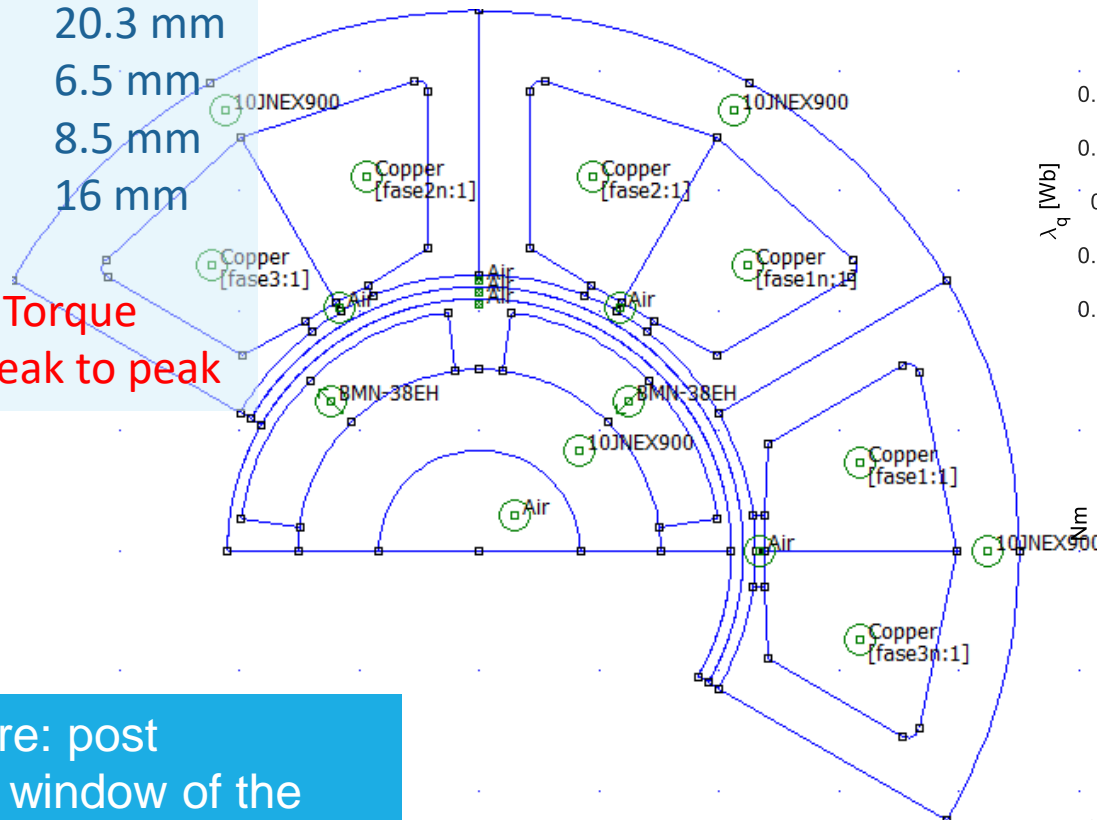
Tooth width 8.5 mm

Tooth length 16 mm

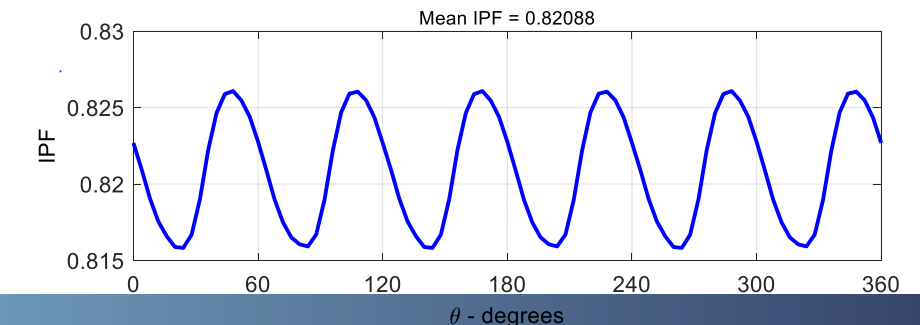
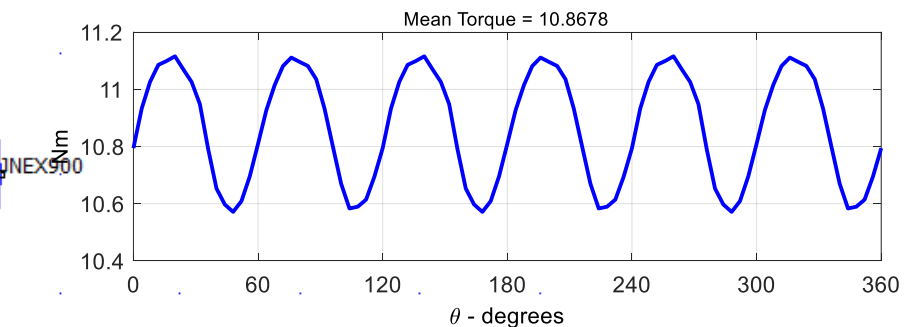
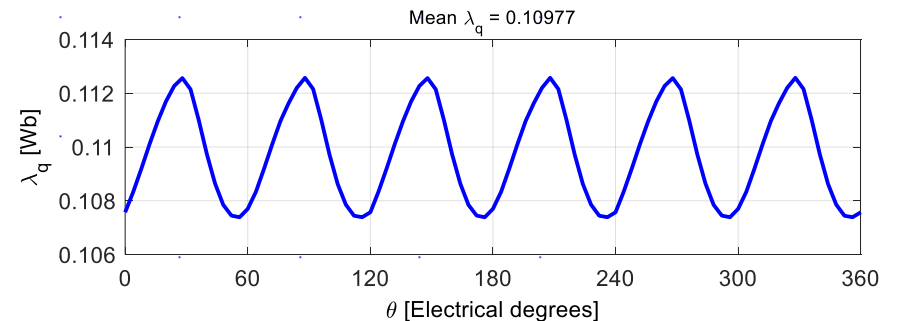
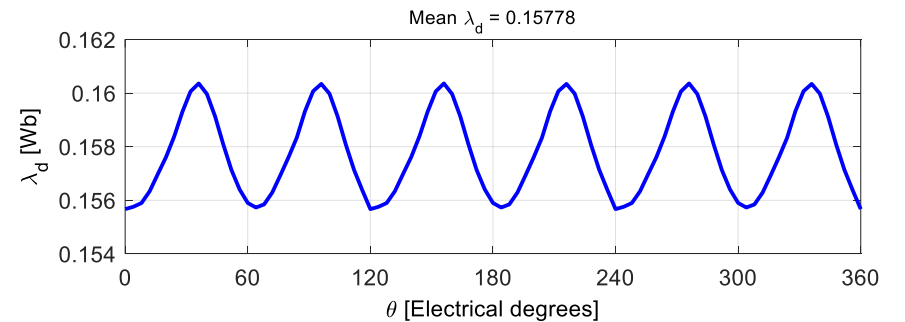
Performances

10.8Nm Average Torque

0.55 Nm ripple peak to peak



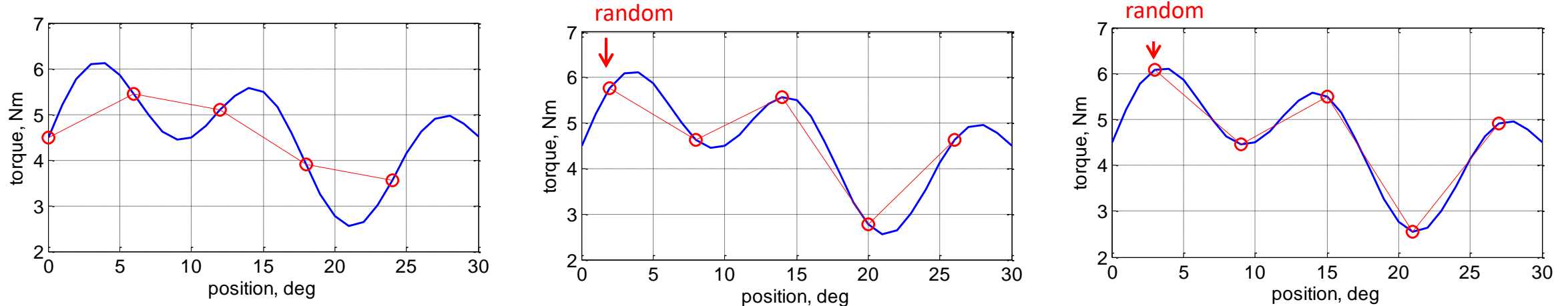
SyR-e where: post processing window of the GUI, 30 positions evaluated over a stator slot pitch



# Selection of the cost function

**Average torque** and **torque peak to peak oscillations** are evaluated.

In order to reduce computational burden it is possible to calculate torque and its oscillations over a reduced number of rotor positions (e.g. 5 regularly sampled position over a stator slot pitch) adding an initial random offset to the first position.



This introduces a noise in the cost function measurement.

During the optimization process, thanks to the high number of motor evaluations, this noise is filtered out.

SyR-e where: the objectives (cost function) are defined in the m-file named Data0.m and then calculated in the m-file named FEMMfitness.m. The number of positions is defined via the GUI.

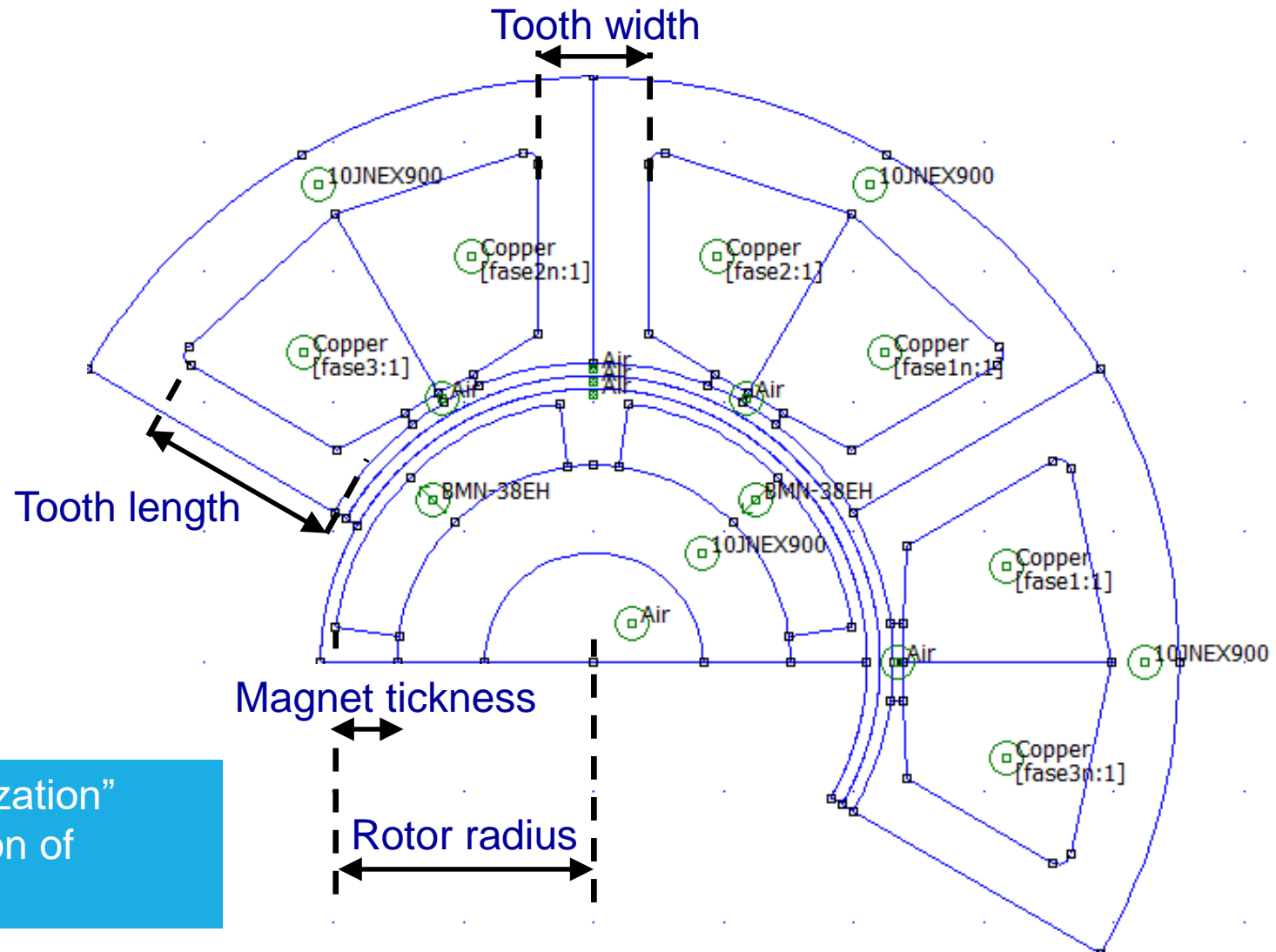




# Selection of the variables to be optimized

As an example, 4 variables have been selected for the optimization process:

1. Rotor radius
2. Magnet thickness
3. Tooth length
4. Tooth width



SyR-e where: GUI “optimization” window allows the selection of variables and their range





# Optimization settings

## Preliminary design

Rotor radius      20.3 mm  
Magnet tickness 6.5 mm  
Tooth tickness    8.5 mm  
Tooth length      16 mm

## Search space

Rotor radius      [15 25] mm  
Magnet tickness   [3 7]    mm  
Tooth tickness    [7 10] mm  
Tooth length      [13 23] mm

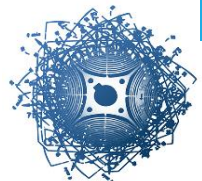
Population size = 50 ( $10 \times \text{number of parameters} + 10$ )  
Maximum number of iterations = 60 ( $\text{Pop\_size} \times 1.2$ )

FEMM calls for each evaluation  
Optimization stage:  
5 positions over 1 stator slot pitch

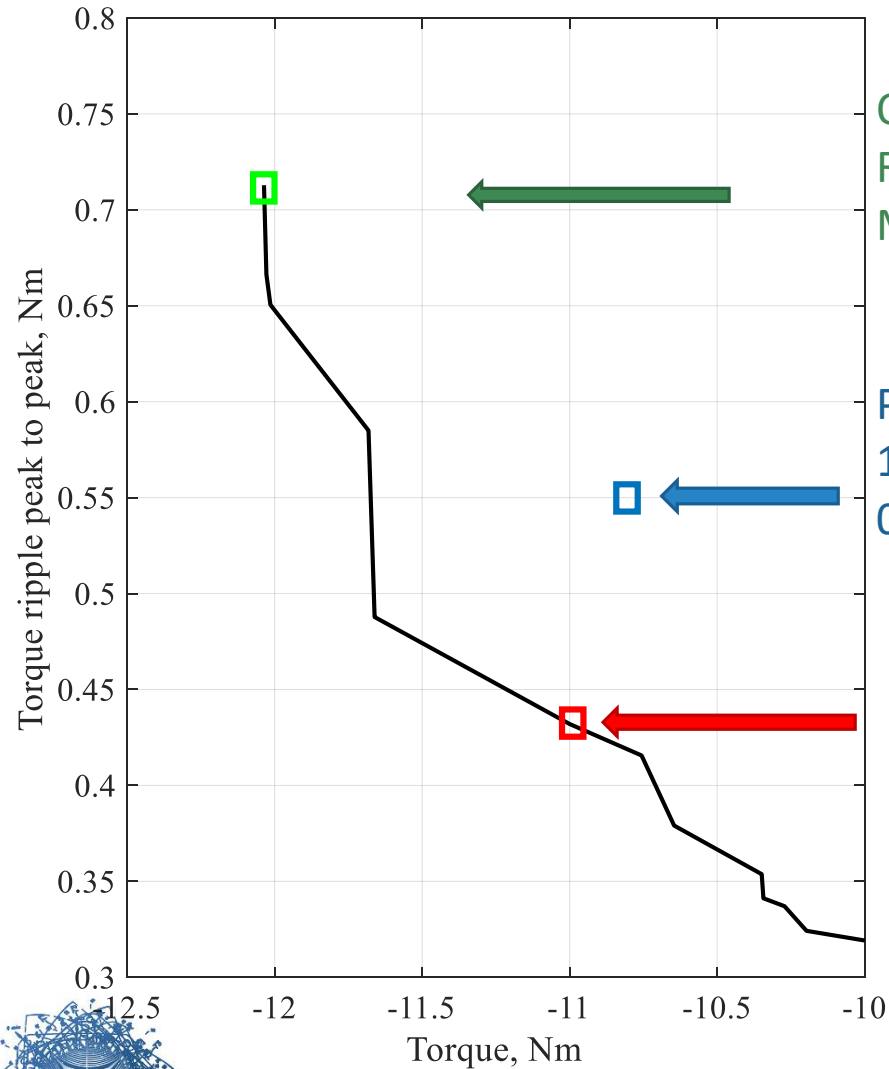
Final re-evaluation of the last pareto front  
30 positions over 1 stator slot pitch

SyR-e where: GUI  
“optimization” window,  
“OPTIMIZE” button

Total Execution time:  
3 hours on a quad-core xeon workstation



# Results of optimization

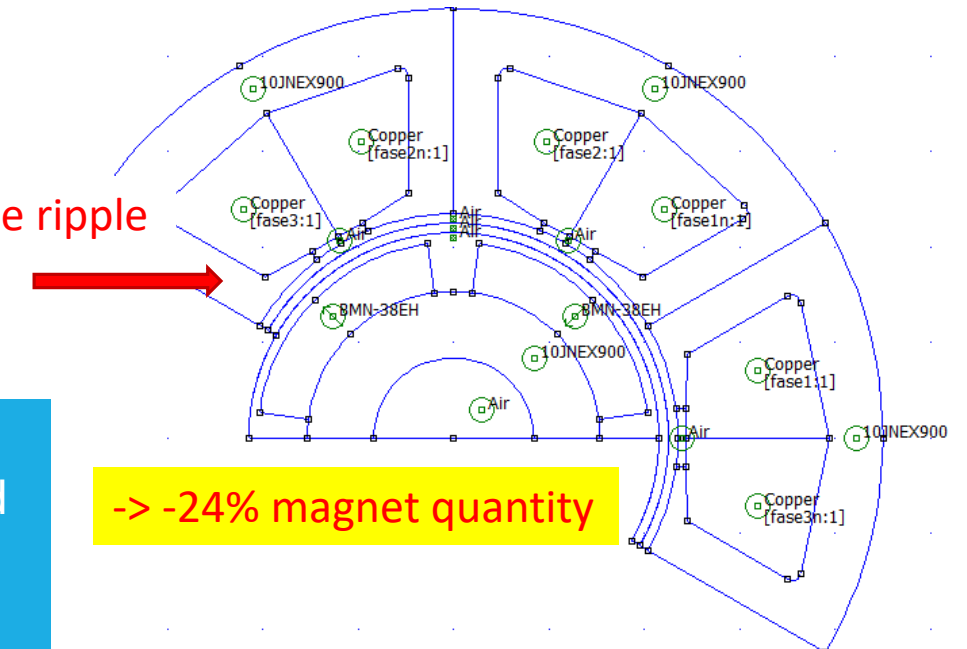
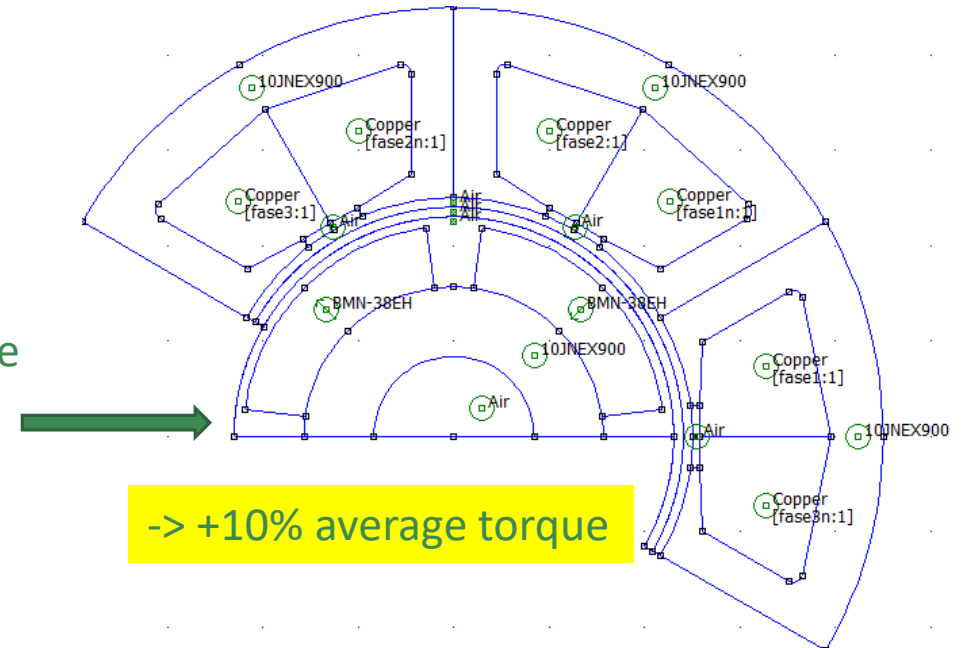


Optimized for maximum torque  
Rotor radius 22 mm  
Magnet thickness 6.4 mm

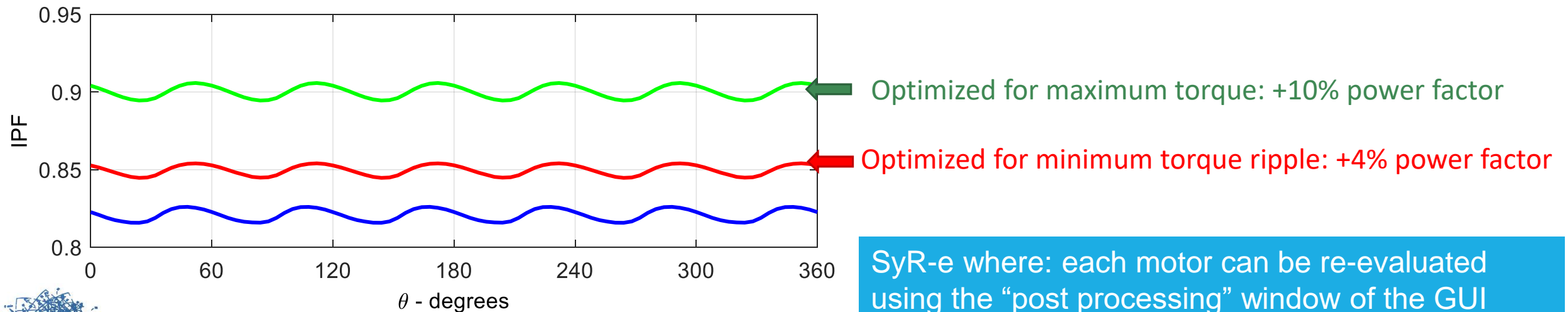
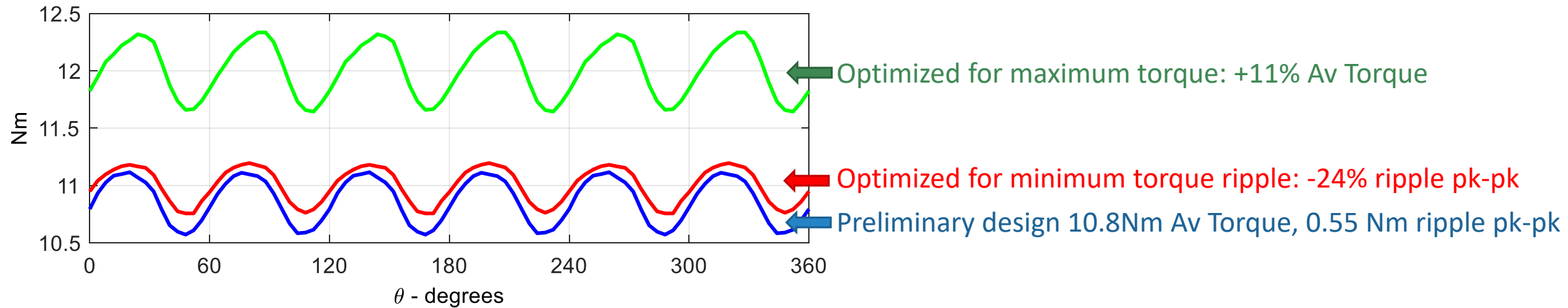
Preliminary design  
10.8Nm Average Torque  
0.55 Nm ripple pk-pk

Optimized for minimum torque ripple  
Rotor radius 20.5 mm  
Magnet thickness 5.1 mm

SyR-e where: all the motor FEMM models and parameters are saved in the results folder



# Improvements with optimization algorithm (post processing tool)



SyR-e where: each motor can be re-evaluated using the “post processing” window of the GUI



# How to collaborate to the SyR-e project

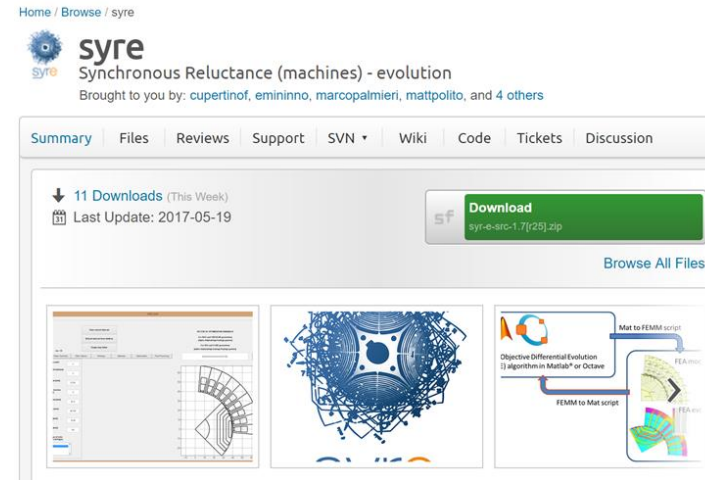
SyR-e public repository is  
<http://sourceforge.net/projects/syr-e/>



SyR-e repository for developers is  
<https://syre.svn.cloudforge.com/syre>



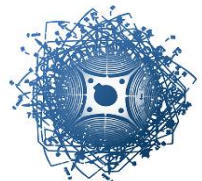
Then we use Tortoise SVN to update  
local copies of the software



Users can point-out bugs or develop new pieces of code  
and then send them via e-mail to:

Gianmario Pellegrino [gianmario.pellegrino@polito.it](mailto:gianmario.pellegrino@polito.it)  
Francesco Cupertino [francesco.cupertino@poliba.it](mailto:francesco.cupertino@poliba.it)

We are also willing to include new developers in the  
CloudForge project.



syre

# Conclusions and future development

This presentation reviewed the basics of multi-objective optimization algorithms and showed their use in SyR-e to design surface PM machines.

Next steps for PM machines in SyR-e:

- Preliminary design (some features for fractional slot windings needs to be improved)
- Sleeve sizing equations
- Include other type of PM rotors
- ... suggestions?

Thank you for the attention.