

**SyR-e** (**Sy**nchronous **R**eluctance **e**volution)

Version 1.6 draft

User’s Manual

September, 17 2020

*This document is a “work in progress” written to give some basic guidelines to start using SyR-e. As the software evolves, we try to keep the manual up-to-date. We apologize for any error or omission and encourage the reader to notify us. Thank you for dedicating your time to our work.*

*Francesco Cupertino (*[*francesco.cupertino@poliba.it*](mailto:francesco.cupertino@poliba.it)*)*

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

*Simone Ferrari (*[*simone.ferrari@polito.it*](mailto:simone.ferrari@polito.it)*)*

# Introduction

## What is SyR-e

SyR-e stands for Synchronous Reluctance – evolution and is an open-source (\*) code developed in Matlab/Octave. SyR-e can design synchronous reluctance machines automatically by means of finite element analysis and multi-objective optimization algorithms.

SyR-e is available for download at <http://sourceforge.net/projects/syr-e/> and, from September 2020, on GitHub <https://github.com/SyR-e>.

It requires Matlab or Octave and FEMM software installed. Among the Octave distributions, the one which was tested with SyR-e is Octave UPM (Politechnic University of Madrid). Octave UPM is a customized version of GNU Octave compiled with GUI. The recommended FEMM version is 4.2, updated on April, 21 2019 (<http://femm.info>). The Matlab version used at the time of this report is R2019b. With Matlab versions older than 2016b, it could be some compatibility problems with the graphical interface.

The principle of operation of SyR-e is represented in Fig. 1. A Matlab script realizes a parameterised drawing of a synchronous reluctance machine as a *.fem* file that is quickly analysed by FEMM. The main results move back to Matlab for performance evaluation. This basic data flow can be used for automatic design purposes, with hundreds of potential machines tested by the multi-objective optimization algorithm, or for the analysis of existing machines, either the just optimized ones or other that are manually designed by the user. GNU Octave can replace Matlab for all mentioned purposes.

(\*) LICENSE: The C++ programs based on the original FEMM source code are licensed under the Aladdin Free Public License, as the original FEMM source is also provided under this license. The Matlab/Octave code is provided under the Apache Version 2.0 license. Further details and the texts of these licenses are provided with the source.

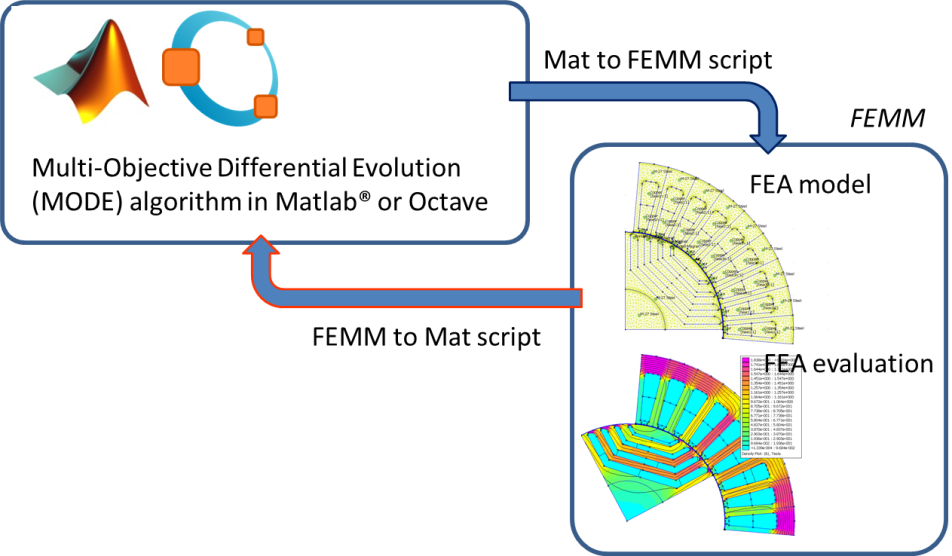


Figure 1 - Data flow to evaluate machine performances

## Background

SyR-e is not a commercial software and therefore no technical support is guaranteed. This User Guide gives to the reader the basic information so to allow a first use of SyR-e but it is not intended as a designer manual. SyR-e has been developed over the last few years and used to realize several designs and prototypes. The main technical details about the chosen design methodology can be found in [1] and [2]. The reader is encouraged to refer to related literature for more technical details on the design of synchronous reluctance machines.

The origin of SyR-e, dated back in 2009, was motivated by a twofold vision: 1) to investigate SyR rotor geometries with no prejudices from the existing literature and, 2) to provide an automatic design tool to non-expert designers. These two aspects are still the foundation of the current release, although the work in between has demonstrated that the SyR-e designed geometries are quite consistent with the previous literature.

To summarize the work done so far, different multi-objective optimization algorithms (MOOA) have been compared in [1]. It was shown that multi objective differential evolution (MODE) can guarantee superior performances in terms of speed of convergence and quality of the final result when compared with other state-of-the-art algorithms. The current SyR-e distribution embeds an open source version of the MODE algorithm, but this can be easily substituted with any other algorithm provided that it runs in Matlab/Octave and it is configured to manage the same input/output variables.

Originally, the first version of SyR-e can design only two types of rotor barriers: the circular ones and the segmented ones (all shown in [2]). In the latest versions, the fluid geometry of the flux barriers (explained in [4]) and the SPM rotor are added, as well as the IPM V-type motors.

## Static Magnetic Solver

Since FEMM is able to solve magneto-static problems, the transient behavior of the machines is usually approximated with a sequence of static simulations in which the rotor position and current phase angles are modified so to emulate their actual behavior. This procedure will be hereinafter referred as static time stepping and needs the number position (*geo.nsim*) and the corresponding rotor angular excursion (*geo.delta\_sim*) to be determined. To be more fast, SyR-e use the sliding gap boundary in FEMM. This means that the rotor is not physically rotated during the static time stepping simulation, but the rotation is obtained with the boundary condition. Further details can be found on FEMM website.

# Getting started

## FEMM installation

FEMM must be installed before running SyR-e, including OCTAVEFEMM support. OCTAVEFEMM is installed automatically with FEMM, typically in the directory *c:\Program Files\femm42\mfiles*. This path needs to be added to Matlab/Octave search path by typing the following lines at Matlab/Octave command prompt:

*addpath(’c:\\progra˜1\\femm42\\mfiles’);  
savepath;*

## SyR-e files

The SyR-e files must be copied in a single directory, for example *c:\SyR-e*. SyR-e must be launched from its root folder, e.g. the user must set the Matlab/Octave current folder to the SyR-e installation directory by typing the following line

*cd C:\SyR-e*

## SetupPath

Run the script “setupPath.m”. This will include in the Matlab/Octave path all the SyR-e subdirectories needed for machine analysis on a permanent basis. After the first time, you do not need to launch setupPath.m anymore. However, if the user move SyR-e to a different home directory, the previous paths needs to be manually removed from Matlab/Octave search path and “setupPath.m” needs to be re-executed from the new directory.

## Matlabpool: cluster on a single computer

Those Matlab users having the “Parallel Computing Toolbox” installed can execute the command *parpool* to enable the parallel execution of “for” loops (the key command is parfor). For more information about this feature, please refer to Matlab documentation. What happens on a single computer is that multiple FEMM instances are run in parallel. Their number depends on the number of microprocessor cores (workers) of the specific computer. If “Parallel Computing Toolbox” is not installed, SyR-e will work exactly in the same way, but the computational time will be longer for some simulations.

Matlab users who also have the MATLAB Distributed Computing Server installed can distribute the FEA simulations on a cluster of computers connected in a network. For example, a network of 3 computers having quad-core processors can use 12 FEA simulations in parallel, with substantiation shortening of the overall computational time.

## Back compatibility

If you have one motor designed with an older version of SyR-e, is possible to open it with the last version of SyR-e. All the missing data will be set to standard values and some messages will appear on the command window. To use the motor (for further optimization or post-processing) is highly recommended to save the machine with the new version of SyR-e.

# SyR-e operation with GUI

The easiest way to use SyR-e (only for Matlab users) is through the Graphical User Interface (GUI), launched by *GUI\_Syre*. In the GUI, there are some windows, with all the key parameters of one motor. The first five windows are used for the design, the sixth windows is for the optimization and the seventh window is for the FEA simulations and the eighth is for the export to Motor-CAD.

For the Octave users or for the advanced users, is possible to use SyR-e without GUI, with the same features described in this paragraph.

## Manual design

This section describes the use of all the parameters editable via the GUI (or using manual\_dataSet for the advanced users). The input data procedure is organized in five sections:

* Main data (e.g., number of pole pairs, number of slots, …)
* Stator & Rotor Geometry (e.g., tooth length, slot opening, ….)
* Other options (e.g. permitted joule losses, overspeed, …)
* Windings (e.g. filling factor, turns in series, coil span, …)
* Materials (e.g., stator and rotor materials)

For each parameter of the GUI there is an input field accompanied by the parameter’s name, its dimension in square brackets and the name of the corresponding Matlab variable in round brackets. When the GUI is launched, the default values of all parameters are taken from the *mot\_01.mat*.

### Main data window

The main data window represented in Fig. 3 is the default one after the launch of *GUI\_Syre.m.* This window allows to select the number of pole pairs (*geo.p*), the number of stator slots per pole and per phase (*geo.q*), the airgap thickness (*geo.g*), the stator outer radius (*geo.R*) and the airgap radius (*geo.r*), the shaft radius (*geo.Ar*), the stack length (*geo.l*) and the type of rotor geometry (*Circular*, *Seg*, *ISeg*, *Fluid*  and *SPM*). For more details about rotor geometry and its parameterization, please refer to [2], [4].

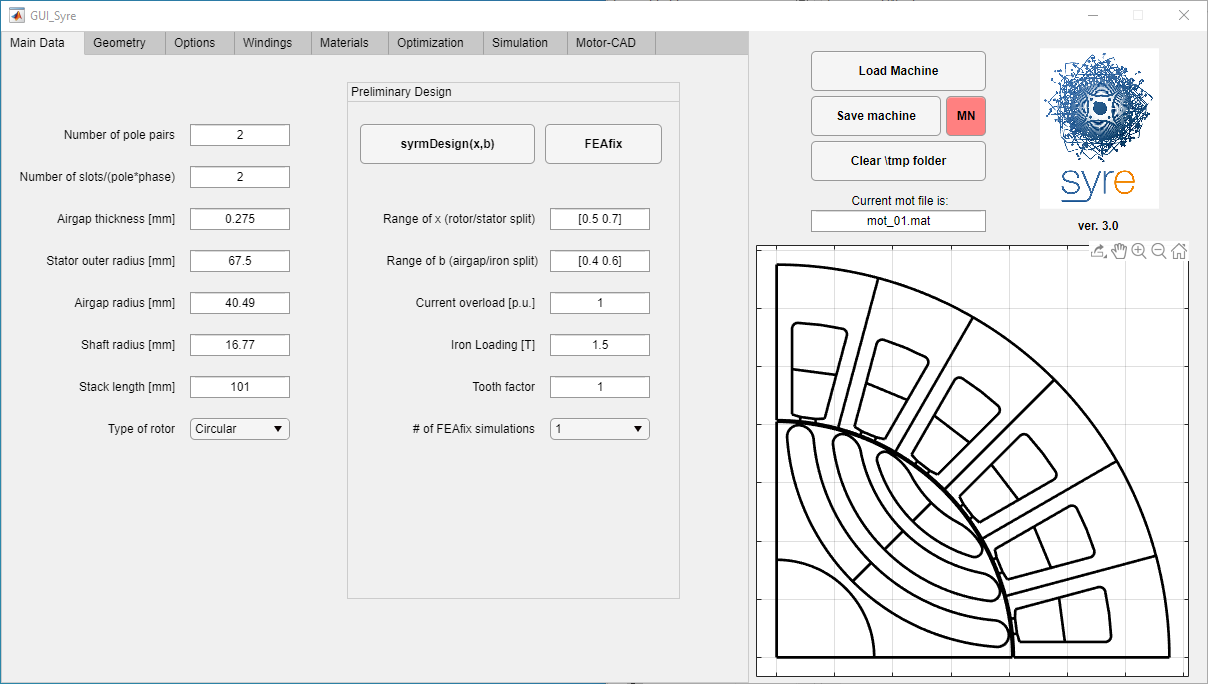


Figure 2 - Main data window

### Stator & Rotor Geometry window

#### Synchronous reluctance machine

The window is split in two parts related to stator and rotor parameters respectively. The left part (stator geometry) allows to select the tooth length (*geo.lt*) and width (*geo.wt*), the slot opening in p.u. of the slot pitch (*geo.acs*) and some details of the slot geometry. In particular, it is possible to modify thickness of the tooth shoe (*geo.ttd*), the angle of the connection between the tooth shoe and the tooth shank (*geo.tta*), and the fillet radius at the back corners of the slot (*geo.SFR*). It is present also a checkbox to design the stator with parallel slots instead of parallel teeth.

The right part of the window is for the parameters of the rotor geometry. If the rotor is a SyR type (*Circular*, *ISeg*, *Seg* and *Fluid*) you can select the number of rotor barriers (*geo.nlay*), the angles describing the barrier positions at the airgap (alpha) and their thicknesses (hc), the translation of the barrier along the q axis (geo.dx) and the dimension of the radial ribs (geo.pont). The last four parameters are vectors of geo.nlay elements. Figure 4 clarifies the definition of *i* (*i*=1,2,…*nlay*) angles and *hci* (*i*=1,2,… *nlay*) thicknesses. The angle of the first barriers is defined starting from the middle point of the pole while the other angles are defined as the angular displacement between two consecutive barriers. This choice is adapted to the use of a per unit representation of the *i* (*i*=1,2,… *nlay*) angles being the angle subtending half pole pitch the base value. Also the thicknesses *hci* (*i*=1,2,… *nlay*) can be represented conveniently in per unit values. If they are all 1 p.u. then the barriers are thick the same and occupy as much radial space as they can. A minimum thickness of the flux guides is fixed to guarantee rotor mechanical feasibility and avoid overlapping barriers. When all the p.u. heights are of a different value, e.g. 0.2, then the barriers are again thick the same with a value that is the 20% of the previous example. All other situations are combination of the previous ones. Angles and thicknesses are input in per unit but the actual values in degrees and millimetres are immediately calculated and displayed.

The translation of the barrier along the q-axis is expressed in p.u. with reference of the hc. This parameter is not available for the *ISeg* geometry. Each barrier can have a different dx, and figure 5 show three rotor lamination with different dx (all the barrier in this example have the same dx, but is possible to have different dx for each barrier). To note that the shape of the end-barrier arcs change with dx, because the thinner points of the radial ribs are defined by the *alpha* parameters, and for the three example, it is the same. (see “2016 08 30 - added dx in circ geometry.pptx” in ReadMe folder).

The radial ribs with usually are evaluated using the “Overspeed” parameter (in other option tab), but in some case can be useful to set manually the dimension of the radial ribs. To enable this feature, the tick near this field must be set. For Seg geometry only, it is possible to have split radial ribs, placed at the corner of the barriers [17]. Regarding the tangential ribs, they are out of the mechanical estimation now, but their dimension can be changed with the proper input.

For Circular and Seg geometries, the FBS angle is available to make asymmetric rotors, useful for torque ripple reduction. Further details could be found in [15] and [17].

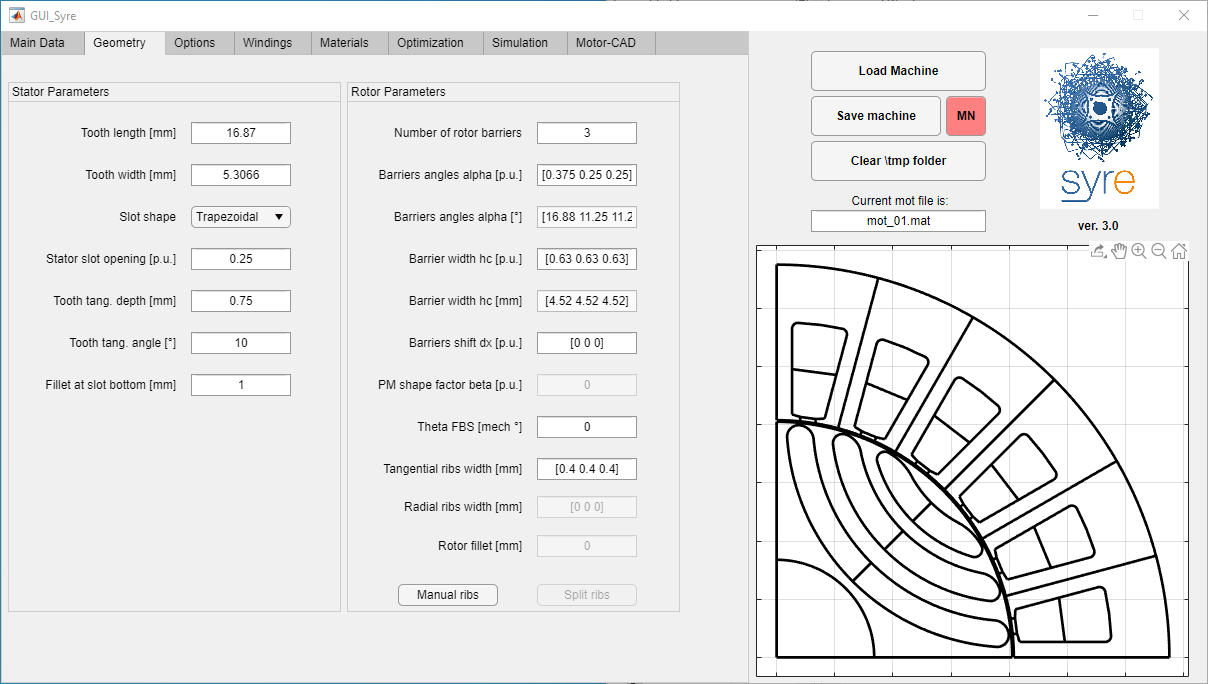


Figure 3 - Stator geometry window

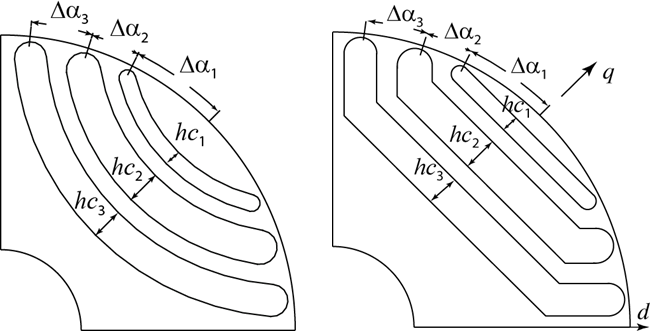


Figure 4 - Definition of i angles and hci thicknesses for circular (left) and segmented (right) rotor geometries in case of three rotor barriers

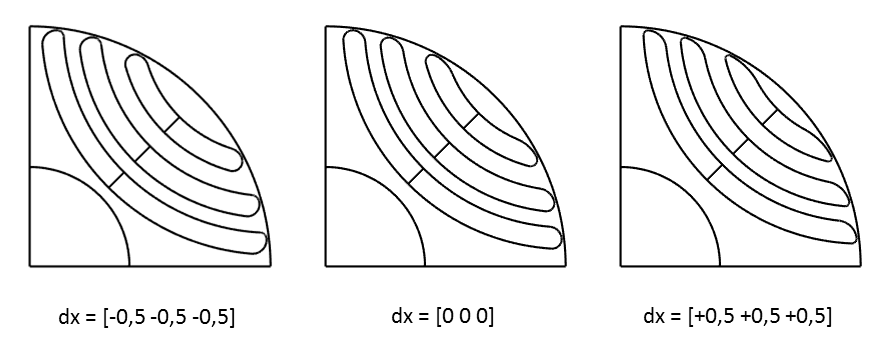


Figure 5 - Difference of dx setting in the barrier design: -0.5 (left), 0 (center) and +0.5 (right).

#### SPM machine

If the rotor type is SPM, the input are the PM angle span (expressed in electrical degree), the magnet width (hc [mm]) and the number of segment of the magnet (for this last input, use the dx field). If dx = 1, the magnet is parallel magnetized. While radial magnetization is needed, define dx as 20 or more.

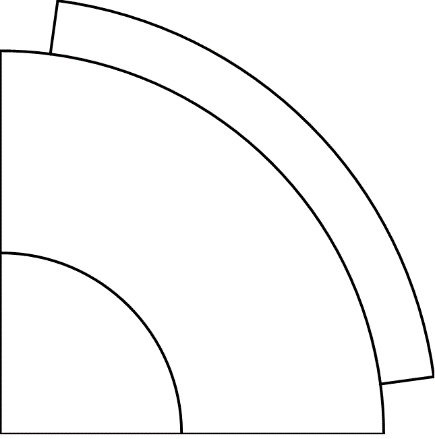
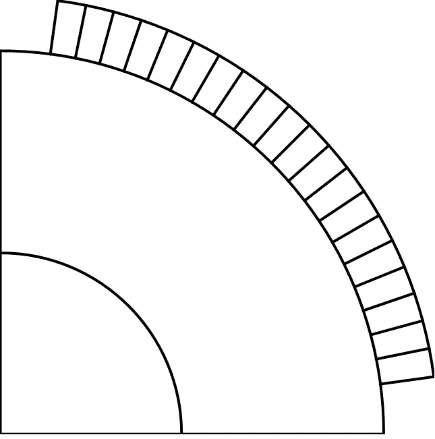
 

Figure 6 - Parallel and radial magnetization

The beta parameter is used as the shaping factor of magnet. When it is unity, the magnet surface is uniform. If it is less than unity, magnet shaping technique is applied on its outer surface. The detailed illustration is presented in [14].

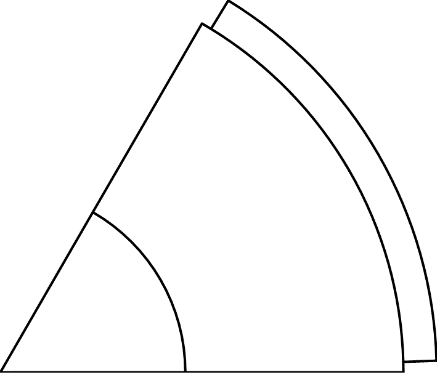
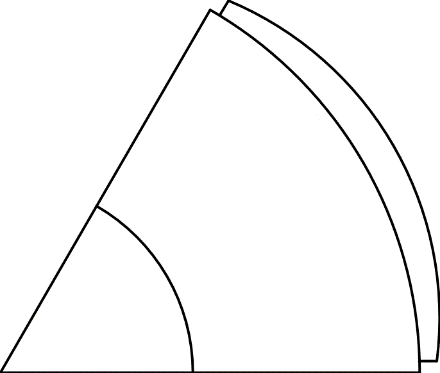
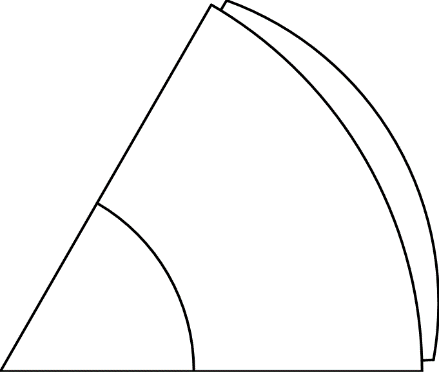
  

Figure 8 – Manget shaping,

#### V-type IPM machine

If the rotor type is set to Vtype, the IPM machine is designed. It follows the same rules for parameters, while the barrier thickness is expressed in per unit of the airgap length. The angle between the d axis and the magnet is defined in p.u., as . Imposing will create a straight magnet (I-type geometry), while will set the PM parallels to the pole boundaries. The parameter dx control the tangential ribs shape: with dx=0, the end barriers are round, otherwise the ribs are with constant width. Fore more details see “2019 07 22 - IPM Vtype geometry parameters”. It is possible to set more than 1 barrier, and the parameters become vectors.

### Other Options window

The thermal loading factor (and so, the continuous stall loss (*per.Loss*)) is the input data that defines the rated loading of the machine. The motor rated current, called *i0*, is calculated accordingly (*per.loss* = 3Ri02). The evaluation of *i0* from the loss input requires that the phase resistance be estimated, on the basis of stator geometry and winding layout, and including the active parts and end-connections length.

The target copper temperature (*per.tempcu*) is the one used to estimate the coils resistance and determine *i0* from *per.loss*. The Housing temperature is used to evaluate the estimated copper temperature with an easy thermal model [10].

*Overspeed* is the maximum expected rotational speed and used to size the radial ribs. The calculation is made considering the centrifugal force at steady state and disregarding the magnetic pull and the structural effect of the tangential ribs at the airgap [3]. If the thickness of the calculated radial ribs falls below the minimum mechanical tolerance defined by *geo.pont0* (that is the last parameter defined in the current window), then the radial ribs are disregarded (as it is the case in Fig. 5).

*Mesh* and *Mesh MOOA* are two variables that control the density of the mesh in the finite element problem (used by file *mfiles\dimMesh.m*). There are two different mesh resolution to set: a general mesh resolution and the airgap mesh resolution. Usually, during the optimization, the mesh is coarser than the post-processing. In Table 1, the equations to obtain the resolutions are reported for both cases.

Table 1 - Mesh resolution definitions

|  |  |  |
| --- | --- | --- |
|  | **During optimization** | **During post-processing** |
| **General mesh resolution** |  |  |
| **Airgap mesh resolution** |  |  |

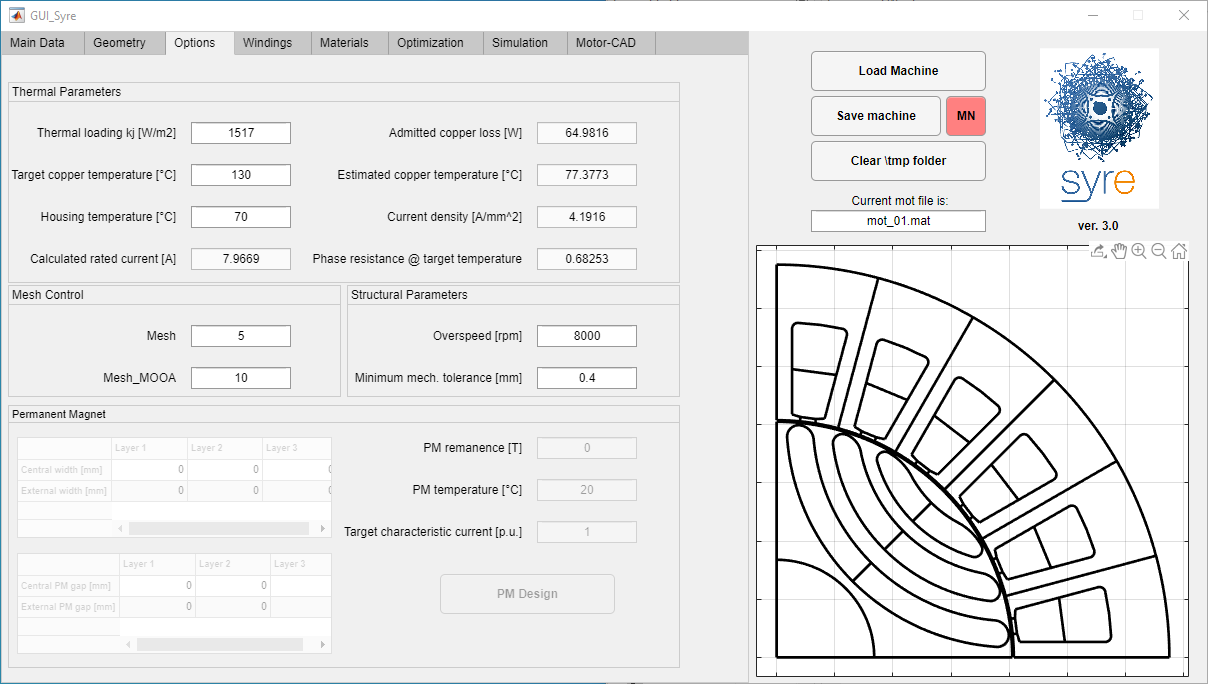


Figure 7 - Other options window

### Permanent Magnet sub-windows

If a permanent magnet material is selected for Circular, Seg, ISeg and Vtype geometry, the Permanent Magnet Tab is enabled. It is possible to set the PM width manually, or using the “PM Design” button, to reach the characteristic current set as input. Further details on the PM design procedure could be found in [17]. Additional table allows to define the gap between the PMs and the iron, if PMs thinner than the barrier are selected.

### Windings window

The copper filling factor (*geo.kcu*), the number of turn in series per phase (*geo.Ns*) and the shortening factor (*geo.Kracc*) are input in this window. If *nsh* is the number of slots by which the winding is shortened and *nfp* is the number of slots covered by the full pitch winding, the shortening factor is defined as (*nfp*-*nsh*)/*nfp*.

There is a table for visualization of the winding layout. The table has two rows, corresponding to the inner and outer layers of the windings, respectively. Single and double layer windings are feasible in this release of SyR-e. The number of columns is equal to the number of slots represented in the machine sketch to the right end of the GUI. The first column (Slot n° 1) corresponds to the slot on the horizontal axis and so on, in counterclockwise direction. The numbers 1, 2, 3, -1, -2, -3 in the table refer to the position of the conductors of the three phases in the slots. Their positive and negative signs determine the direction of the phase current into the slot. Single layer windings are a subcase of double layer ones: when the two numbers corresponding to each single slot are equal, a single layer winding will be represented.

Every time *geo.q* or *geo.Kracc* are changed, the winding table is recalculated using a version of Koil software (http://koil.sourceforge.net/ by Luigi Alberti) purposely rebuilt to work with SyR-e under Windows operating systems and distributed with SyR-e package. It is always possible input the numbers in the table manually and customize the winding layout. It is recommended to use the “SAVE CONFIGURATION” button after each change of windings parameter, to make the manual modifications effective.

It is possible also design a stator with multiple three-phase sets, by setting properly the related input. About the winding, the numbers of the phases follow the original one, e.g.: [4,5,6] for the second set, [7 8 9] for the third set and so on. The winding configuration must be saved with “Save Configuration” button.

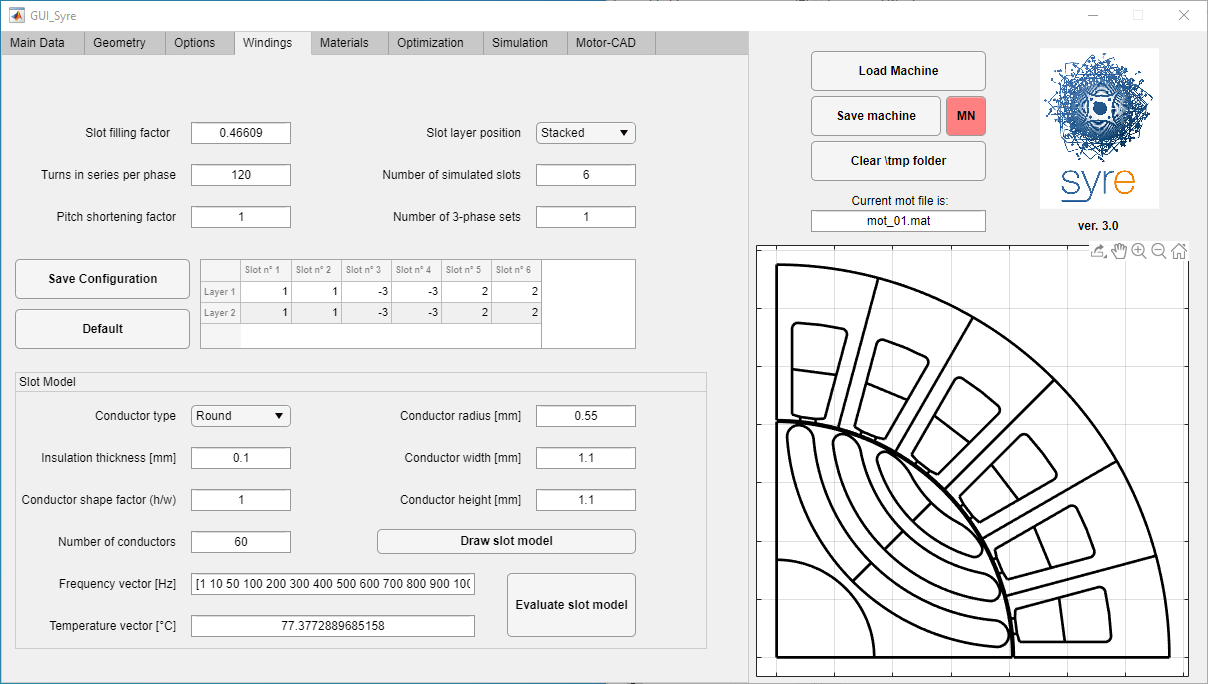


Figure 9 - Windings window

For example, two slots per pole per phase (*geo.q* = 2) and no shortening (geo.Kracc = 1). The table says:

1 1 -3 -3 2 2

1 1 -3 -3 2 2

A chorded, double layer version is (Kracc=5/6):

1 1 -3 -3 2 2

-2 1 1 -3 -3 2, or alternatively1 1 -3 -3 2 2

1 -3 -3 2 2 -1

Phase permutations have no effect because the offset between the three phase windings and the rotor in position zero is calculated automatically.

The parameter *Qs* is the number of slots to use in the simulation. It can be varied for the non-conventional winding machines, with an increase of the computational time of FEMM. The default value is the minimum allowed.

If the designed machine is a fractional slot machine, the “slot layer position” tick can be selected to design layer in the coil tooth style.

The subwindow “slot Model” allows to define the slot model of the motor and place the conductor in the slot and compute the skin effect model.

There are two ways to create the slot model:

* Imposing the slot filling factor, shape factor and number of conductors, then, pushing “Draw slot model”, the conductor sizes will be computed
* Imposing the number and sized of conductor, setting the slot filling factor equal to NaN, then slot filling factor will be automatically computed after pushing “Draw slot model”.

Then, slot model can be evaluated for different frequencies and temperatures, by clicking “Evaluate slot model”.

### Materials window

In the material tab you can set the material for slots, stator core, rotor core, flux barrier and shaft, with a drop-down window. The mass of each section of the motor and the rotor inertia are automatically computed according to the geometry and the material properties. To see the material library and add/remove items, the buttons at the bottom of the window can be used. The blue button can be used to see the material properties from the library. The material database is stored in three .m files (one for each type of material).

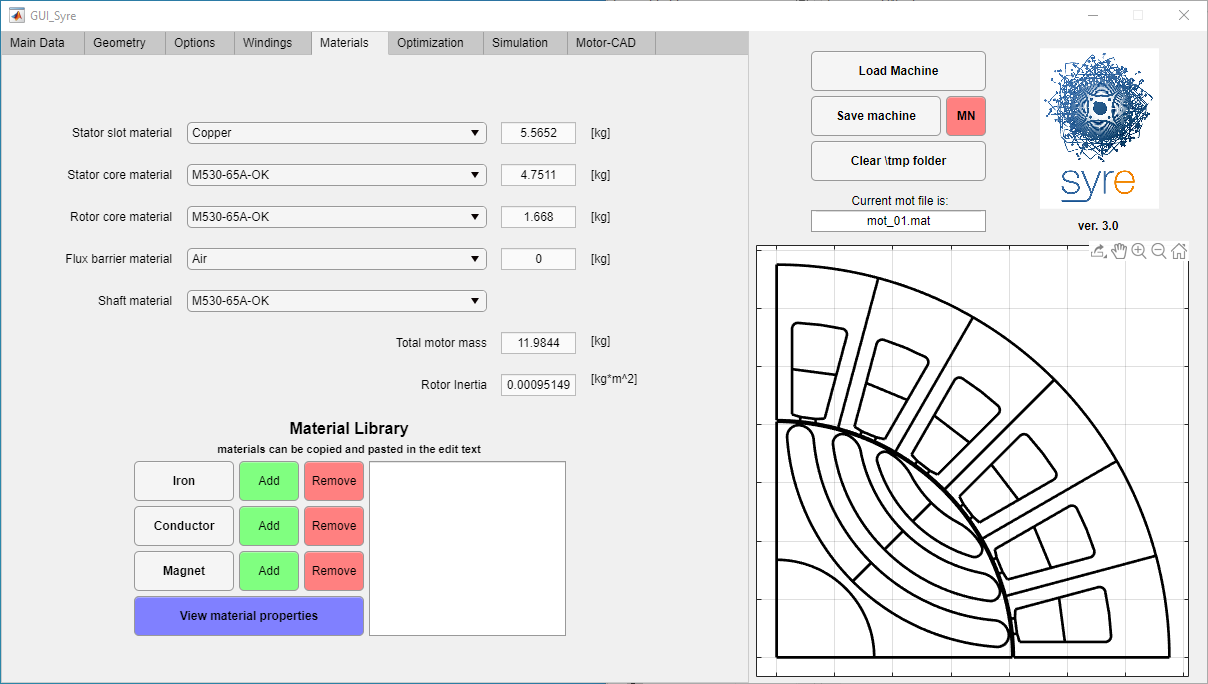


Figure 10 - Materials window

### Preview and save a single machine

Every time a parameter is modified a preview of the machine appears in the right side of SyR-e window. The “Save Machine” button allows to save the current machine configuration. Two files are created, a .mat file containing the current data set and a .fem file that is the machine model for finite element analysis with FEMM. The .mat file can be reloaded at any time using the open button at the top left corner of SyR-e window. The FEMM model can be run manually or with the aid of the SyR-e “Post Processing” window.

“Empty \tmp folder” deletes all the files in the subfolder *\tmp*. Many files are temporary stored is this folder during optimization, and it is the good practice to empty the folder before each optimization run.

## Automatic design with parametric analysis

In the main data window, there is also the “syrmDesign” button, which perform a parametric design of SyR motor according to [11] and [16], SPM motor according to [12] and single layer V-type motors according to [18]. The parameters of the analysis are the split ratio *x* defined as the ratio between rotor and outer stator radius and the induction ratio *b* defined as the ratio between the peak of the airgap flux density and the peak flux density in the stator iron. For the SPM and V-type motors, instead of this parameter, the ratio between the length of the PM over the airgap width.

Before to start the parametric analysis, the user must set the main data of the machine, plus the boundary of the parameters, the overload current during the analysis, the induction in the stator yoke iron (Bfe) and the tooth factor (kt) that define how the stator teeth are narrow.

When the user push the syrmDesign button, the contour plot of torque and power factor appear in the x-b plane (or x-lm/g plane) and a machine can be selected and saved pushing on the preferred point of the plot.

It is possible, for all the geometries, to use the FEAfix approach to increase the accuracy of the design plane. The number of FEA simulations for each plane could be set and, if a parallel pool is enabled, the simulations will run in parallel. If 1000 simulations is select, all the design plane will be FEA-simulated. Further details could be found in [16].

## Execution of the Optimization

During the optimization, some stator and rotor parameters are automatically modified, in order to reach the best in term of some objective. The optimization process consists of two parts: the evolution process and the re-evaluation of the Pareto front, with better accuracy than the evolution process. Before pushing the “Optimize” button and start the optimization, three type of input must be set:

* Optimization algorithm parameters
* Optimization variables
* Optimization objectives

Before to start the optimization, remember to delete the \tmp folder to avoid lack of memory on the computer.

If “Parallel Computing Toolbox” in installed on the computer, optimization process always uses parallel computing. Some information about the computational time and Pareto front are visualized during the optimization process.

If the process fails for technical reasons (e.g. black-out), it is possible to restart a previous optimization from the last generation, launching “restartOptimization.m”

### Optimization parameters

The multi-objective algorithm used for the optimization is a variant of the well-known Differential Evolution described in [9]. On the left of the Optimization tab there are the optimization setting, which are the number of generation of the evolution process and the number of the population for each generation. In addition, the angle span and the number of simulation for the evaluation of the performance of each candidate must be set (see [1] for more details). The evaluation parameters can be different for the evolution process and the final evaluation of the Pareto front and are all settable in the Optimization tab. The overload current to use in the optimization process can be set on the left of the optimization tab. Using a quite high overload current during the optimization (e.g. 2) can guarantees a lower torque ripple both at rated current and at overload current [1].

### Optimization variables

The optimization variables are the geometrical parameters that the optimization algorithm change during the evolution process. They are:

* Flux barrier angles (for SPM, is the angle span of the magnet)
* Flux barrier thickness (for SPM is the thickness of the magnet)
* Translations along the q-axis
* Airgap thickness
* Airgap radius
* Tooth width
* Tooth length
* Stator slot opening
* Tooth shoe thickness
* Beta parameter (PM shaping factor, for SPM and Vtype)
* PM dimensions (for Circular, Seg, ISeg and Vtype only, expressed in per unit here)
* FBS angle (for Circular and Seg only)

The last optimization variable is gamma (, which are the angle between the current vector and the d-axis (maximum permanence axis). Each motor is evaluated using a single current angle. At the end of the optimization is expected to be very close to the Maximum Torque per Ampere (MTPA) angle condition MTPA, which is the one maximizing the torque per Joule loss.

For each variable are present a 2-state button, to enable/disable the variable in the evolution process and a text-box, to set the boundary values.

In the definition of the search bounds and the number of variables to be included in the optimization process some preliminary test should be realized so to verify that the chosen search space does not contain unfeasible machines. For example, the sum of rotor radius and tooth length should be lower than the stator radius. This kind of checks are not executed automatically by SyR-e then a particular care must be used in the definition of bounds. A suggestion could be to start with bounds close to a machine with known performances so to gain experience and confidence with the software.

### Optimization objectives

The optimization objectives define which individual are good and which are not good. The objective actually available are:

* Maximum torque
* Minimum torque ripple, which is the peak-to-peak torque ripple. In the past was used the standard deviation of the torque divided by the medium torque.
* Minimum copper mass
* Minimum PM mass

Each objective has a 2-state button to be enabled or disabled and one textbox to set its penalization limit. During the evolution process, all the motors that have their performance higher than the maximum value set are penalized. This discourages the optimization algorithm to further search in not promising regions of the search space.

### Optimization results

At the end of the optimization, a .mat file and a folder are created in the \results folder named according to the scheme OUT\_*date*T*time* where *date* and *time* are taken when the optimization ends. The OUT\_*date*T*time*.mat file contains the data set used to launch the optimization and a data structure called OUT with the main optimization results (Pareto front solutions and their performances). In the \results\OUT\_*date*T*time* folder there are the .fem files relative to the final Pareto front machines, the .bmp images representing the same machines and some .fig files reporting the Pareto front and the values of the optimization variables of the optimized machines.

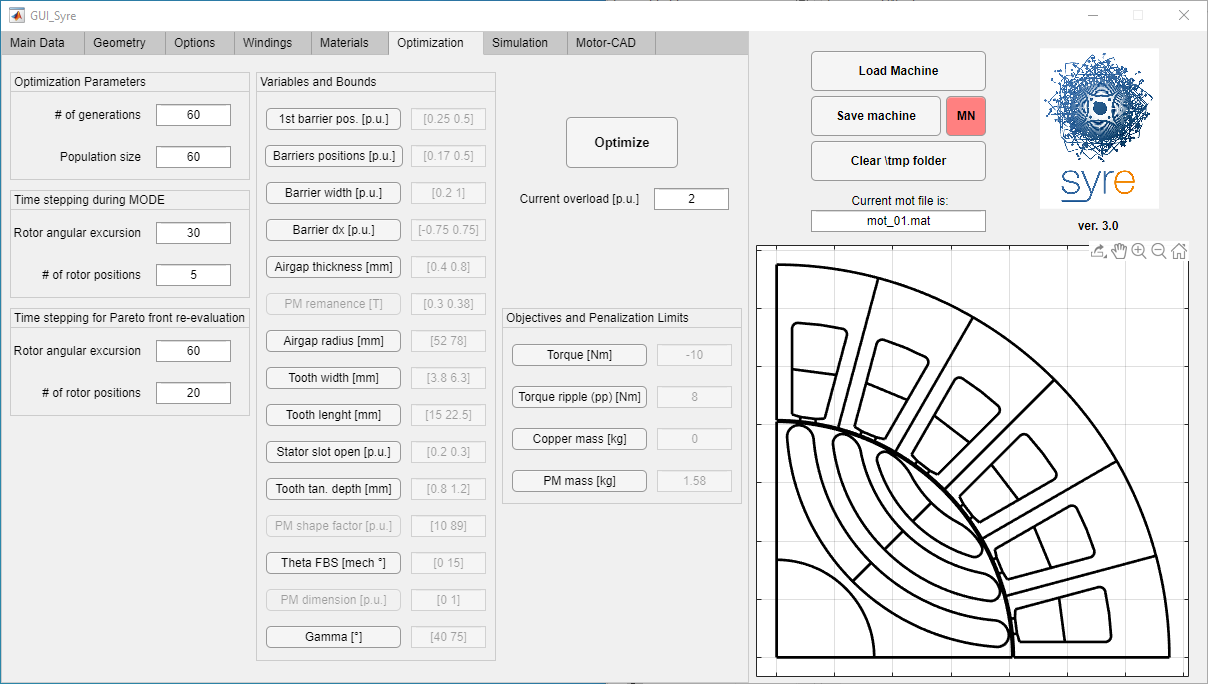


Figure 9 - Optimization window

## SyR-e FEA Simulations

The post processing is executed using the last SyR-e window. The main parameters to enter are the ”rotor angular excursion” that is the angular span of the static time stepping in electrical degrees, the “current phase angle” in *dq* coordinates and the “number of rotor positions” that is the number of equally spaced rotor positions that will be simulated over the previously defined angular excursion. Moreover, the “current load” is the per unit current level used in the analysis, being the current *i*0 defined on the basis of the admissible joule losses at stall the base current value. Permanent magnets having residual flux density specified by “Br” can be included in the rotor barriers (Br can be extracted from the temperature characteristic too). Note that the radial ribs are not re-calculated considering permanent magnet mass during the post processing because the analysis will be executed starting from an existing .fem file.

There are seven types of post-processing which SyR-e can perform, selectable from the Evaluation Type menù (see “2019 07 22 - Evaluation Type”):

* Single point evaluation/sensitivity analysis
* Flux map
* Demagnetization analysis
* Demagnetization curve
* Characteristic current computation
* Flux density analysis
* Current offset simulation
* Airgap force computation

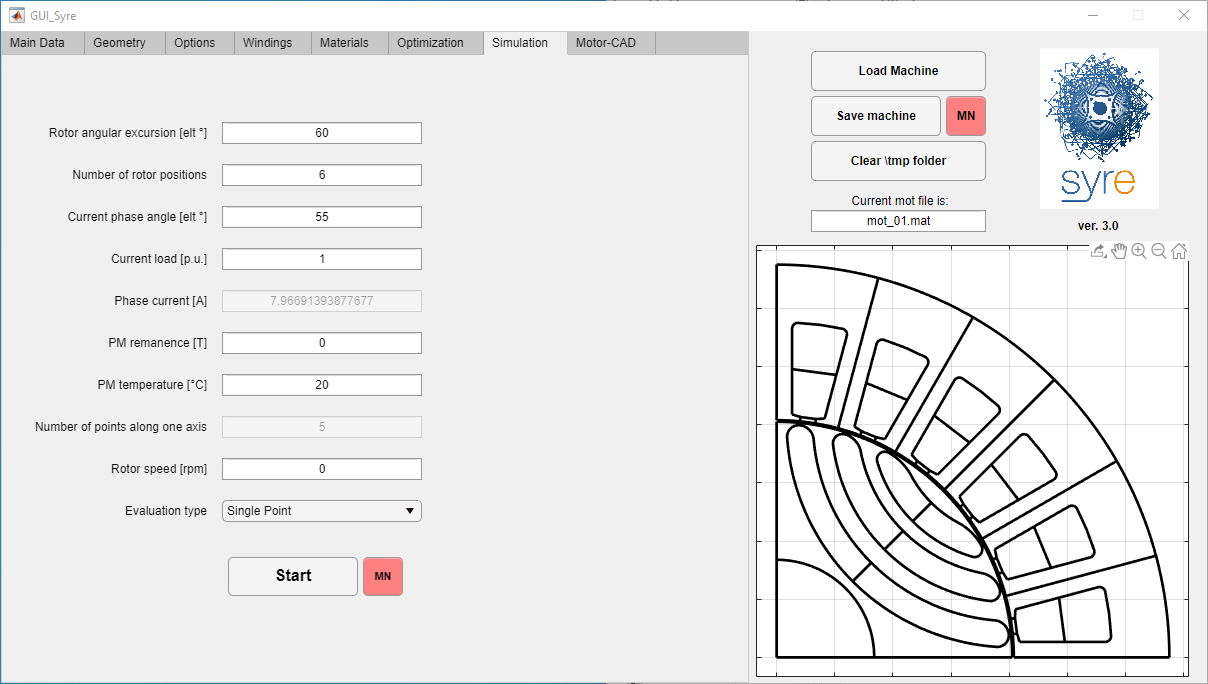


Figure 12 - Post Processing window

### Single point evaluation

This is the standard evaluation of SyR-e: the machine is simulated with one condition of current, with the set rotor angular excursion and number of rotor position. The results of the evaluation are the fluxes, the torque and the internal power factor. If the current amplitude and angle are vectors, SyR-e performs a sensitivity analysis according to the selected inputs. If parallel pool is licensed and enabled, simulations are run in parallel.

### Flux and torque maps evaluation

In this evaluation, a grid of id and iq is used to evaluate the motor. The grid limit are set with the current amplitude. If Current Load is a number, the test grid is square, in the correct quadrant for the machine under simulation. If it is a two element vector, the domain is rectangular, according to [IdMax IqMax] for SyR and PM-SyR machines. If the input is a four element vector, the map is done in the domain [IdMin IdMax IqMin IqMax]. The number of point for each dimension of the grid is defined by the “number of points in [0 Imax]” input.

### Demagnetization analysis

In this case, a selected current level and PM temperature is simulated and the quantity of PM demagnetized is shown.

### Demagnetization curve

This simulation aims to evaluate the demagnetization current function of temperature. The input is just the temperature vector, while the current is computed through an iterative cycle for each temperature.

### Characteristic current

The characteristic current is computed through an iterative cycle for each geometry with PM for each PM temperature considered.

### Flux density analysis

In this case, the simulation is similar to the first one, but the flux density waveforms at the airgap, in the middle of the stator teeth and in the middle of the stator yoke, for each rotor position are extracted.

### Current offset simulation

This simulation is similar to the first one, but a common mode current is added to the phase currents.

### Airgap force computation

This is the same simulation of single operating point, but the force along the airgap is extracted.

### Iron loss computation

There are two evaluation type for iron loss computation, similar to operating point and flux maps. Iron loss can be computed using MagNet model, or through FEMM simulations (with a dedicated post-processing included in SyR-e code).

# Offline operation

## Manipulation of the simulations results

This set of function (stored in the folder syreManipulateMM) allow to manipulate the result of the flux and torque maps evaluation. Further information in the Gui\_Syre\_MMM documentation.

## SyR-e export

Standard SyR-e files are in .mat format and .fem format. There are some features to do the automatic export of the motor from SyR-e to other software. The export currently supported are:

* File .dxf: is useful to export the draw in other CAD or FEA software, like MagNet, SolidWorks, AutoCAD, and other. The function is in \syreExport and is called SyreToDxf.
* MagNet: is like the post-processing with FEMM, but use MagNet. It is included in the GUI with the red buttons “MN” to save a MagNet file and simulate in MagNet (only single point and map, with loss)
* Motor-CAD: export available from the last window.

# Working without GUI

Octave users cannot use the GUI and must find their way to the direct use of the major scripts of SyR-e. Plus the use of SyR-e without GUI can be useful for Matlab users which want to automatize some function of SyR-e. Here the expert user can find the advanced function that allow to use SyR-e without GUI.

## manual\_dataSet.m

In this script produce a dataSet structure, similar to the one produced by the GUI. Is the input for the no-GUI users.

## DrawMachineScript(dataSet,pathname,filename)

This function allow to save the motor described by the structure dataSet in the selected pathname and with the selected filename. If no input are used, the dataSet is evaluated by manual\_dataSet.m, the directory is the SyR-e directory and the name is “newmachine.mat”.

## MODEstart(pathname,filename)

With this function, you can run the optimization. If you run the function without inputs, the base setting are the ones stored in manual\_dataSet.m. Else, if you put a correct pathname and filename, the base motor will be the selected motor, and in MODEstart you can change the parameters of the optimization.

# References

1. F. CUPERTINO, G. PELLEGRINO, C. GERADA, “Design of Synchronous Reluctance Motors with Multi-Objective Optimization Algorithms”, IEEE Transactions on Industry Applications, DOI: 10.1109/TIA.2014.2312540.
2. G. PELLEGRINO, F. CUPERTINO, C. GERADA, “Automatic Design of Synchronous Reluctance Motors focusing on Barrier Shape Optimization”, IEEE Transactions on Industry Applications, DOI: 10.1109/TIA.2014.2345953.
3. PALMIERI M., PERTA M.; CUPERTINO F., PELLEGRINO G. “High-Speed Scalability of Synchronous Reluctance Machines Considering Different Lamination Materials”, IEEE IECON 2014, the 40th Annual Conference of the IEEE Industrial Electronics Society, 29 October - 1 November Dallas (USA).
4. M. GAMBA, G. PELLEGRINO, F. CUPERTINO, “Optimal Number of Rotor Parameters for the Automatic Design of Synchronous Reluctance Machines”, ICEM 2014, International Conference ion Electrical Machines, 2-5 September 2014, Berlin, Germany.
5. PALMIERI M., PERTA M.; CUPERTINO F., PELLEGRINO G. “Effect of the numbers of slots and barriers on the optimal design of synchronous reluctance machines”, OPTIM 2014, International Conference on Optimization of Electrical and Electronic Equipment, Brasov, Romania 22-24 May 2014, pp. 260-267.
6. CUPERTINO F, PELLEGRINO G, ARMANDO E, GERADA C “A SyR and IPM machine design methodology assisted by optimization algorithms”, IEEE ECCE 2012, Energy Conversion congress & EXPO. Raleigh (USA), 15-20 September 2012, pp. 3686-3691.
7. G. PELLEGRINO, F. CUPERTINO: “FEA-based multi-objective optimization of IPM motor design including rotor losses”, IEEE ECCE 2010, second IEEE Energy Conversion Congress and Exposition, Atlanta, Georgia, USA, September 12-16, 2010.
8. F. CUPERTINO, G. PELLEGRINO: “IPM motor rotor design by means of FEA-based multi-objective optimization”, IEEE ISIE 2010, Bari, Italy, 4-7 July 2010.
9. A. Zamuda, J. Brest, B. Boskovi ˇ c, and V. ´ Zumer, “Differential Evolution for Multiobjective Optimization with Self Adaptation,” in 2007 IEEE Congress on Evolutionary Computation CEC 2007. IEEE Press, 2007, pp. 3617–3624.
10. A. Boglietti, A. Cavagnino, M. Lazzari and M. Pastorelli, "A simplified thermal model for variable-speed self-cooled industrial induction motor," in IEEE Transactions on Industry Applications, vol. 39, no. 4, pp. 945-952, July-Aug. 2003.
11. I. Boldea, T. Fulcao, T.A. Lipo, L. Malesani, T.J.E. Miller, A. Vagati, “Synchronous Reluctance Motors and Drives, A New Alternative,” Tutorial Course
12. C. Lu and G. Pellegrino “A Simple Design Method for Surface-mounted PM Machines for Traction Application”, Proc. of IEEE ECCE 2016, 18-22 Sept. Milwaukee (USA).
13. G. Y. Sizov, D. M. Ionel and N. A. O. Demerdash, "Modeling and Parametric Design of Permanent-Magnet AC Machines Using Computationally Efficient Finite-Element Analysis," in *IEEE Transactions on Industrial Electronics*, vol. 59, no. 6, pp. 2403-2413, June 2012.
14. C. Lu, S. Ferrari, G. Pellegrino, M. Davoli, C. Bianchini “Parametric Design Method for SPM Machines Including Rounded PM Shape”, *IEEE Energy Conversion Congress and Exposition (ECCE)*, Cincinnati, OHIO, Oct. 1–5, 2017
15. S. Ferrari, G. Pellegrino, M. Davoli and C. Bianchini, "Reduction of Torque Ripple in Synchronous Reluctance Machines through Flux Barrier Shift," *2018 XIII International Conference on Electrical Machines (ICEM)*, Alexandroupoli, 2018, pp. 2290-2296.
16. S. Ferrari and G. Pellegrino, "FEA-Augmented Design Equations for Synchronous Reluctance Machines," *2018 IEEE Energy Conversion Congress and Exposition (ECCE)*, Portland, OR, 2018, pp. 5395-5402.
17. S. Ferrari and G. Pellegrino, "Torque Ripple Minimization of PM-assisted Synchronous Reluctance Machines via Asymmetric Rotor Poles," in press, *2019 IEEE Energy Conversion Congress and Exposition (ECCE)*, Baltimore, MD, 2019.
18. S. Ferrari, G. Pellegrino, M. Z. M. Jaffar and I. Husain, "Computationally Efficient Design Procedure for Single-Layer IPM Machines," *2019 IEEE International Electric Machines and Drives Conference (IEMDC)*, San Diego, CA, 2019.

# Acknowledgements

SyR-e was made possible thanks to the contribution of several colleagues, students and friends that collaborated to the development of the project.

A special credit goes to Irene Bedino, author of the original design of the SyR-e logo.