SyR-e
User Manual

Simone Ferrari

simone.ferrari@polito.it

September 2, 2022

Contents

Chapter 1

Introduction

1.1 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 on SourceForge and, from September 2020, on GitHub. The principle of operation of SyR-e is represented in Figure 1.1. A Matlab script realizes a parameterized drawing of a synchronous machine as a .fem file that is quickly analyzed 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.
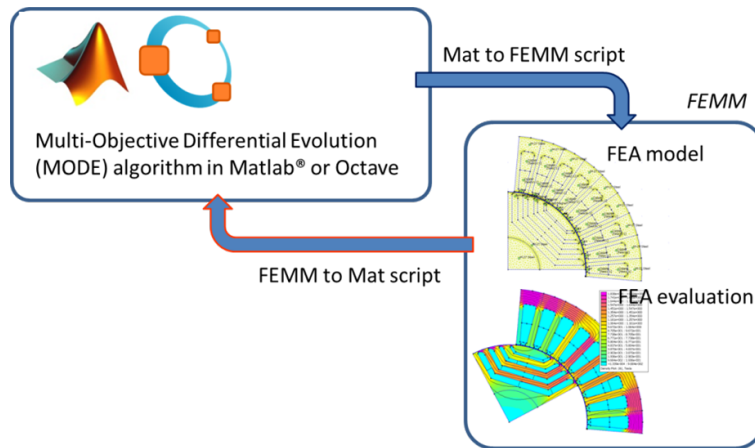
Figure 1.1: Data flow to evaluate machine performances

1.2 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 years and used to realize several designs and prototypes. 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:

- to investigate SyR rotor geometries with no prejudices from the existing literature;

- 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. It was shown that multi objective differential evolution (MODE) can guarantee superior performances in terms of speed of convergence and quality of the 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. In the latest versions, other geometries are added, as the Fluid geometry, the SPM motors and V-type IPM motors.

Chapter 2

Getting Started

2.1 SyR-e Installation

To use SyR-e, the code must be downloaded from one of the repositories and unzipped. The Matlab or Octave path must be the unzipped folder. SyR-e is equipped with two graphical user interfaces (GUIs): one for the design and FEA analysis (main GUI) and the other for magnetic model manipulation (identified with the acronym MMM). To use SyR-e it is just necessary to launch from Matlab one of the two GUI, realized with Matlab AppDesigner.

2.2 FEMM Installation

SyR-e requires FEMM for the FEA simulations, including OctaveFEMM. This is installed automatically with FEMM, typically in the directory
'C:\femm42\mfiles\'.
    This path is automatically added from SyR-e to the Matlab/Octave path once launching one of the GUI, or with the function setupPath. If FEMM in installed in other folders, it is necessary to manually add the folder to Matlab/Octave path.

2.3 SyR-e File Format

SyR-e save two files each motor *motorname.mat* and *motorname.fem*. The former contains all the information of the motor, regarding the geometry, ratings, performance, materials,. . . while the latter is the FEMM model, with the geometry of the motor.

2.4 System Requirements

The system requirements can be checked launching the function `checkRelease()`. For convenience, they are reported also in the following.

2.4.1   Matlab Version

Actual Matlab version adopted for SyR-e development is the 2021b. SyR-e can operate with Matlab versions not older than 2016b. For previous versions, some problems can arise.

2.4.2   Matlab Packages

To use all the capabilities of SyR-e, this Matlab packages are suggested:

- Simulink: dynamic simulations (syreDrive)

- Parallel Computing Toolbox: for the parallel computing of FEA simulations

- PDE Toolbox: structural analysis and mass computation

- Curve Fitting Toolbox: used for some post-processing

2.4.3   Octave Version

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.

2.4.4   FEMM Version

It is suggested to operate with the last FEMM version available. At the time of this document, the last version available is the one released 21Apr2019. Older version than the 25Feb2018 cannot be used.

2.4.5   Back Compatibility

In general, back-compatibility of the SyR-e files is guaranteed. Once the old project is loaded in one of the GUIs, a back-compatibility check is performed, and the missing information are added. It is hardly suggested to save the motor, in order to have an updated project.

2.5   Motor Examples

In the SyR-e path it is included the folder *motorExamples*, with three examples of motors (complete with flux maps), designed with SyR-e. They are:

- mot_01 : it is a SyR motor, and the default SyR-e motor.

- RAWP : it is a SyR motor designed from an induction motor stator. Further information on this motor can be found here.

- THOR : it is a PM-SyR motor, with custom geometry loaded (originally used an old SyR-e parametrization). Further details on the motor can be found here.

Figure 2.1: Geometric tab of SyR-e GUI

- TeslaModel3_custom : it is a IPM V-type motor similar to the rear axle e-motor of Tesla Model 3 electric car. It uses custom geometry, with the rotor cross-section retrieved from available online pictures.

2.6 Utilities

On the main GUI there is a tab (reported in Figure 2.1) related to some useful links. These utilities are:

- Export of the motor model to dxf file or other commercial FEA softwares.

- Check of the minimum system requirements in terms of Matlab version and toolboxes and FEMM version.

- Control of the parallel pool.

- Link to the SyR-e documentation and references.

- Link to the MMM GUI.

- Link to syrmDesignExplorer.

Chapter 3

Design eMotor with SyR-e

In the following, the operation with GUI (and so Matlab) will be illustrated. Besides the availability of the GUIs, it is always possible to work without GUI, both for Matlab and Octave users. Some glimpse of non-GUI operations will be given in the last sections of this document. To design an eMotor with SyR-e the main GUI must be launched, with the command `GUI_Syre`. This GUI is divided into 8 tabs, dedicated for precise operations (that will be described in the following). When the GUI is launched, a default motor is loaded.

3.1 Motor and Geometric Definitions

The main parameters of the motor and the stator and rotor geometric definitions are done with the first two tab of the GUI, reported in Figure 3.1 and Figure 3.2.

In the main tab is possible to define main motor quantities, like number of pole pairs, number of stator slots, stator, rotor and shaft radius, airgap and stack length and motor type. The available motor types are:
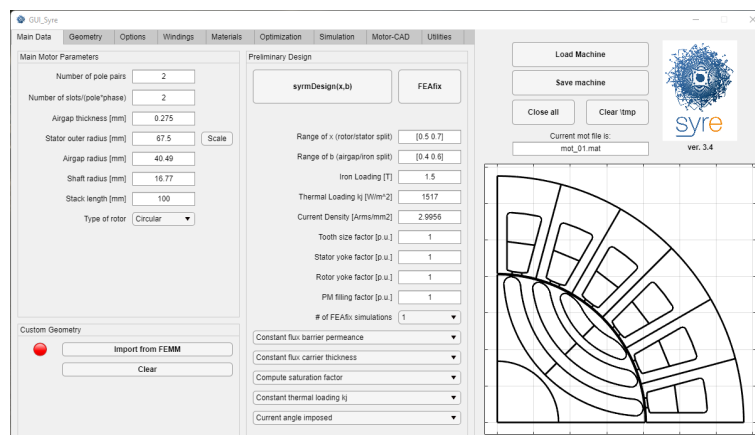
- Circular

- Seg

Figure 3.1: Main tab of SyR-e GUI
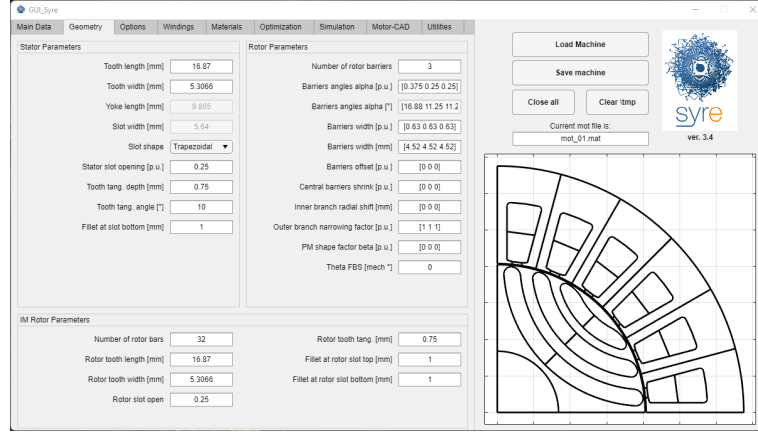
Figure 3.2: Geometric tab of SyR-e GUI

- ISeg (not recommended for new projects, use Seg geometry with modified parameters)

- Fluid

- SPM

- V-type (not recommended for new projects, use Seg geometry with modified parameters)

The first four are for SyR and PM-SyR motors and will generate motors with SyR dq axis convention (PMs along the $-q$ axis), while SPM and V-type will generate motors with PM motor dq axis convention (PMs along the $+d$ axis). Moreover, it is possible to change the axis convention from the Simulation tab.

Stator and rotor dimensions are defined in the second tab of the main GUI, reported in Figure 3.2. The stator definition is quite common, while for rotor definition, the p.u. framework is adopted, and some input changes based on the type of selected rotor.

3.1.1 SyR Rotor Definitions

The main parameters for SyR rotors are the number of flux barriers, their position along the airgap and the barrier thickness, defined according to Figure 3.3. Please note that the barriers are numbered from the outer to the inner.

Other parameters include:

- Barrier offset, defined in p.u. and represent a shift of the barriers along the q-axis, as described in Figure 3.4

- Central barrier shrink, expressed in p.u. allows to transform a Seg geometry to a V-type geometry (when set to 1)

- Inner branch radial shift in mm, allows to create "deep" flux barriers, without moving the barriers edges
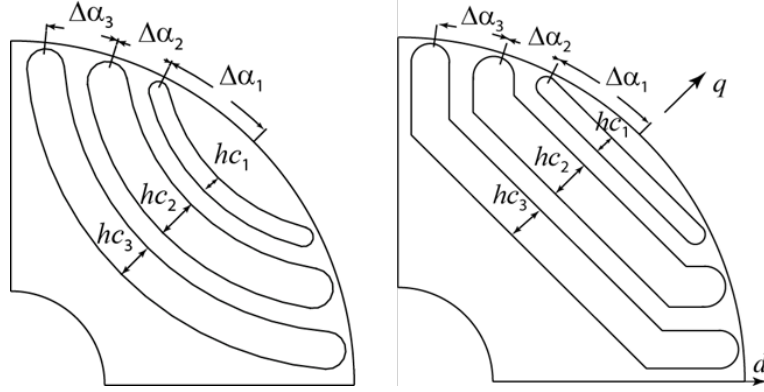
Figure 3.3: Definition of the rotor barrier position aong the airgap and the barrier thickness

Figure 3.4: Definition of the flux barrier shift

- Outer branch narrowing factor in p.u, allows to reduce the thickness of the outer branch of the Seg barriers

- FBS angle, expressed in mechanical degrees allows to make the rotor asymmetric and reduce torque ripple (see here and here for details)

3.1.2   SPM Rotor Definitions

For SPM rotors, the input are the PM angular span (expressed in electrical degrees) and the PM thickness. Additional parameters are the number of PM segment along the circumference, that allows to discriminate between parallel and radial magnetization as depicted in Figure 3.5, and the PM shape factor, expressed in p.u., that allows to change the PM thickness as shown in Figure 3.6.

3.1.3   IPM V-type Rotor Definitions

For the V-type rotor, similar parametrization of SyR motor are followed. The only big difference is the definition of PM shape factor, that change the PM angle, going from a straight magnet if PM shape factor is zero, to PMs parallel to pole limit if PM shape factor is equal to one. Further information can be found in the *2019 07 22 - IPM Vtype geometry parameters.pptx.*
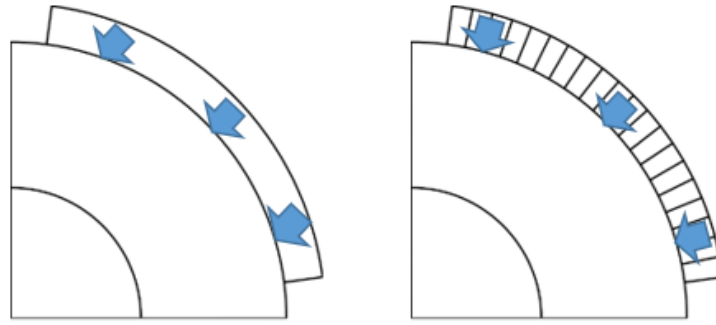
Figure 3.5: Difference between parallel (left) and radial (right) PM magnetization
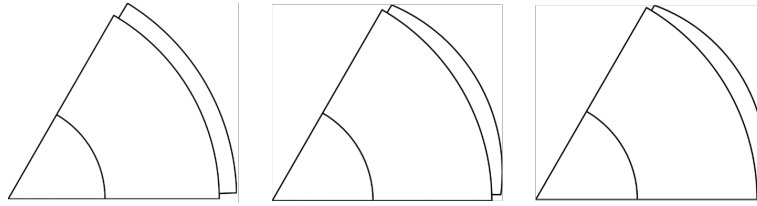
Figure 3.6: Effect of PM shaping factor, equal to 1 / 0.5 / 0.33 from left to right

This geometry is going to be dismissed. The suggestion to create V-type IPM motors is to use Seg geometry, imposing *Central Barrier Shrink=ones(1,nlay)* (see TeslaModel3_custom motor model).

3.1.4 Induction Motor Rotor Definition

Induction motors are added to SyR-e, at the moment just for simulation purposes and without design equations. The stator is defined as for synchronous machines, while the rotor geometry has a dedicated tab, where is possible to define number of rotor bars and slot and tooth dimensions. Dealing with simulations, just single points and flux maps are supported (with the IM controlled in Field Oriented Control).

3.1.5 Custom Geometry Motor

Besides the parametric definition of the geometry, SyR-e allows the simulation of custom rotor geometry. To do so, the steps are:

- Create and save a SyR-e motor as close as possible to the custom geometry, in order to have the material blocks setted correctly in FEMM

- Open the FEMM project and modify the geometry. DXF file is possible to import, but be careful about the boundary condition setting, winding settings and material settings

- Once the FEMM file is saved, it is possible to push the button *Import from FEMM* in the main tab. Once import is completed, the led will be green
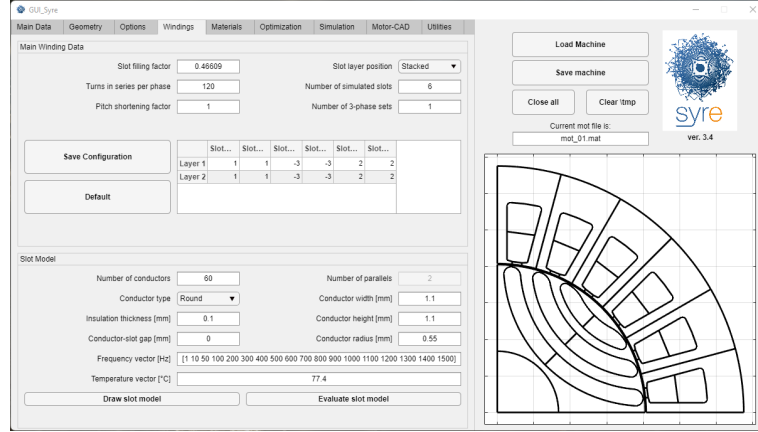
Figure 3.7: Winding tab in the main GUI

and the custom motor geometry will be colored on the SyR-e GUI. Examples of a custom geometry are THOR and TeslaModel3_custom motors in the *motorExamples* folder.

3.1.6 Radial Geometrical Scaling

SyR-e allows an easy radial scaling of the motor by using the button *Scale* close to the stator outer radius input. If this button is enabled, it is possible to change the outer radius and all the radial dimensions will scale accordingly.

3.2 Winding Definition

The windings are defined in the respective tab, reported in Figure 3.7. The winding is automatically generated thanks to a custom *Koil* version, realized for SyR-e. Multi-three-phase motors can be realized by changing the number of three-phase sets. In this case, the number of slots per pole pair in the main tab must be adjusted. It is possible to change the number of simulated slots and have custom winding by changing the winding definition in the tab and then the button *Save Configuration*. The numbers define the phase on each layer and each slot, while the sign define the current direction.

3.2.1 Slot Model and AC Loss Simulation

In the bottom of the tab there is the section related to slot model and AC loss factor computation. It is possible to set the shape and number of conductor to fit in the slot, starting from the slot filling factor, or set the slot filling factor equal to `NaN` and set the conductor number and dimensions. The model will be created in FEMM (Time Harmonic solver) and can be computed for a matrix of frequency and temperature, defined from the proper inputs. Parallel computing is adopted by default.
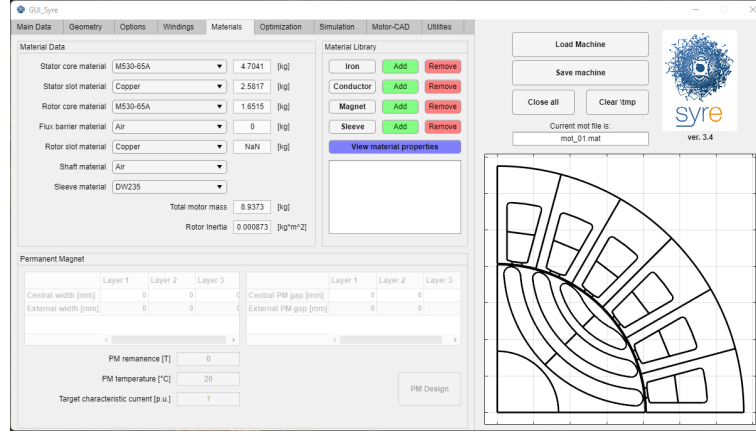
Figure 3.8: Materials tab of the main GUI

3.3   Material Definition

Materials are defined in the respective tab, reported in Figure 3.8. It is possible to define the material for the different sections of the motor. With the green and red buttons is possible to add or remove materials for each class (iron, conductor, magnet and sleeve) and with the blue button it is possible to see the material properties. Motor mass and rotor inertia is computed according to material properties and geometry.

3.4   Thermal Parameters

Thermal parameters are included in the top section of the Options tab of the main GUI, reported in Figure 3.9. It is possible to use three different inputs to compute the rated current: thermal loading k_j [W/m²], rated loss [W] and RMS slot current density J [A/mm²]. One quantity is the input and the other are computed online. Together with this, housing temperature and target average winging temperature must be set. With this data, a simple lumped parameters thermal model (see here for reference) computes the winding temperature and the rated current to get the target thermal loading.

3.5   Structural Parameters

SyR-e embeds also a simple structural model adopted during the design. The two main structural parameters are the overspeed and the minimum mechanical toler-ance, all reported in a dedicate window in the Options tab. It is possible to include also rotor sleeve, by imposing its thickness (and eventually design a rib-less motor), but all the procedures related to this feature are under development.

At the bottom of this tab there is also the ribs design windows. There is possible to define the thickness of the tangential and radial ribs (neglecting the overspeed parameters), decide if the radial ribs must be split (just for Seg geometry) and change fillets and inclination of the ribs.

Figure 3.9: Options tab of the main GUI

Table 3.1: Mesh resolution definition

	During Optimization	During Post-Processing
General mesh resolution	$\frac{Mesh\_MOOA}{p}$	$\frac{Mesh}{p}$
Airgap mesh resolution	$\frac{1}{p}$	$\frac{Mesh}{p \cdot Mesh\_MOOA}$

3.6 Mesh Control

The mesh size control for FEMM is done through two parameters in the Options tab: `Mesh` and `Mesh_MOOA`. The mesh is defined in a different way for the airgap and if the optimization is running or not. The detail of the mesh resolution is reported in Table 3.1

3.7 Preliminary Design

There are three tools that can be adopted for preliminary design: *syrmDesign* and *FEAfix* buttons (in the main tab) and *PM Design* (in the material tab).

3.7.1 syrmDesign and FEAfix

This tool was originally created for the SyR motors design at constant iron flux density and thermal loading. It allows to compute a design plane where torque and power factor are plotted function of two dimensionless parameters. Each point on this design plane represent a different motor and it is possible to select the best design, based on the target performance and the inputs. In the years, the approach is improved, adding the FEAfix button, that allows to correct the analytical equations at the base of the design plane with few selected FEA simulations (the number of FEA simulations is selectable from the GUI). Further details can be found here. The last improvement deals with the PM motors, and the extension of the syrmDesign/FEAfix approach for PM-SyR and IPM machines.

The design options that can be selected from the GUI are:
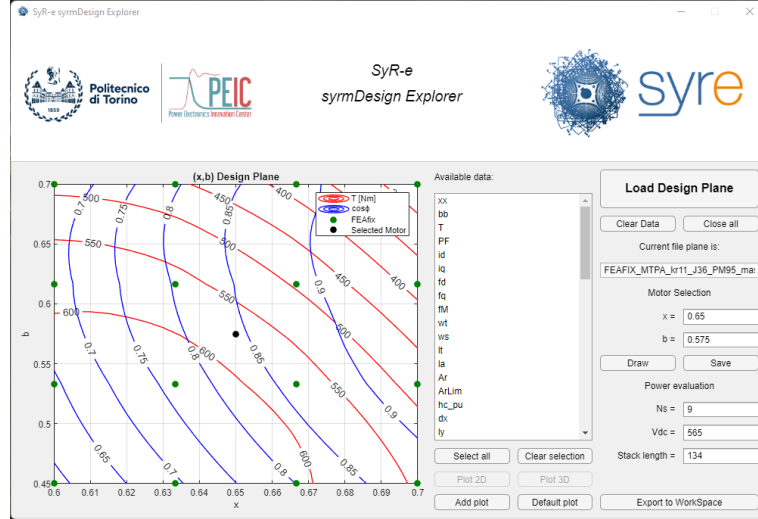
Figure 3.10: Graphical User Interface of syrmDesignExplorer tool

- flux barrier design: select if all the barriers must have the same thickness (useful for PM-SyR motors) or have same permeance (suggested for SyR motors);

- flux carrier design: select how to split the rotor iron thickness, if with $\Delta x = 0$, have constant flux carrier thickness, or thickness proportional to sine wave;

- use or not use the saturation factor, computed with simple model;

- select the thermal input, if the design plane is obtained with constant k_j or constant J;

- correction of the current angle during the FEAfix process (searching for MTPA).

The result of the syrmDesign/FEAfix evaluation is the picture of the design plane, that include more data than the plotted. It is possible to open this figures with the *syrmDesignExplorer* GUI, reported in Fig. 3.10 and see all the performance figures computed from the model, select new motors and scale the design plane (length and turns).

3.7.2 PM Design

This procedure allows to design the PMs of PM-SyR and V-type motors targeting a defined characteristic current. The tool is on the bottom of the Material tab and allows to manually change the PM size or automatically design the PM sizes, with a FEA iterative procedure described here.

3.8 Design Optimization

The design optimization is performed with MODE algorithm and is setup in the Optimization tab, reported in Figure 3.11. The tab is divided into 5 windows, to:

Figure 3.11: Optimization tab of the main GUI

- Define the optimization size, so number of generations and number of individual for each generation

- Define the time stepping during MODE (raw)

- Define the time stepping during the re-evaluation of the Pareto front after MODE ends (fine)

- Select the optimization variables and boundaries

- Select the optimization objectives and penalization limits.

Dealing with the optimization process, it is carried out at constant thermal loading (i.e. constant copper loss, since the stator outer radius is constant). The current angle can be included in the optimization variables, in this way the MTPA is automatically found during optimization. To improve the convergency, penalization is done on the quantities that are bigger than some limits, defined as penalization limits. If the limit is negative, the respective quantity is maximized (example: minimization of minus maximum torque), while if the penalization limit is positive, the quantity is minimized (torque ripple). Details on optimization can be found, for example here and here. The results of the optimization are saved in the results folder as a file and one folder containing all the motors on the Pareto front of the optimization.

It is possible to perform a structural check on each motor, by enabling the *Mechanical Stress Control*. This add a further objective related to structural simulations, that penalize the structurally-weak motors.

A last parameter is the optimization type. Selecting *Design*, the design space will be explored, while the mode *Refine*, the optimization variables will change around the actual motor parameters.

Chapter 4

Simulate eMotor with SyR-e/FEMM

The main FEA simulation engine adopted with SyR-e is FEMM. The FEA simulations are controlled from the Simulation tab of the main GUI, reported in Figure 4.1.

The first two inputs are related to the definition of the FEA simulation and the time stepping solver, while the other inputs can be enabled or disabled based on the Evaluation type selection. The possible FEA simulations are:

- Single Point

- Flux Map

- Characteristic Current

- HWC Short-Circuit Current

- Demagnetization Curve

- Demagnetization Analysis

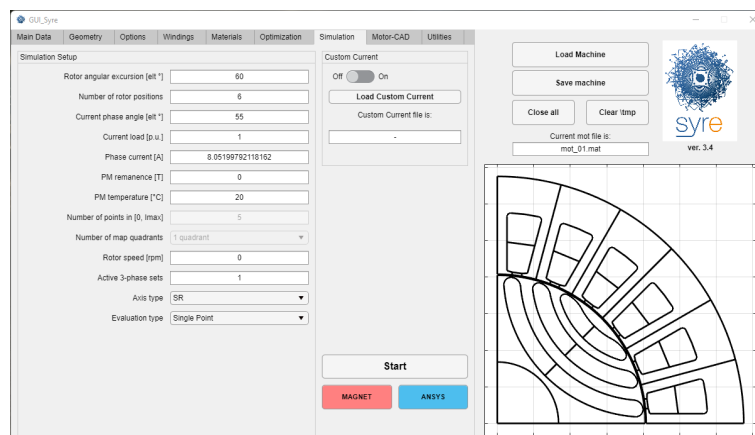- Flux Density Analysis

Figure 4.1: Simulation tab of the main GUI

- Current Offset

- Airgap Force

- Iron Loss – Single Point

- Iron Loss – Flux Map

- Structural Analysis

It is possible to select also the axis type convention used for FEA simulations. There are two possible choices:

- *SR*: typical of SyR motors, with the (possible) PM flux linkage aligned along the $-q$ axis;

- *PM*: typical of PM motors, with the (possible) PM flux linkage aligned along the d axis.

For multi-three-phase motor, it is possible to activate / de-activate each single 3-phase sets with the input *Active 3-phase sets* (1 means that the set is active, 0 means that the set is open).

4.1   Static Magnetic Solver

The FEA simulation are performed in FEMM with a static magnetic solver. The rotor movement is emulated by running several simulations at different rotor position, emulating a continuous rotor rotation. Phase currents are imposed in accordance to rotor position and the input *dq* currents. The trade-off between accuracy and computational time is done by selecting a proper angular span of the rotation and the number of rotor position that must be simulated. Usually, for standard three-phase distributed winding motors, 60 electrical degrees rotation is enough (symmetry are exploited) and 30 rotor positions gives good harmonic resolution, while 6 rotor positions can get the average values.

4.2   Single Operating Point Simulation

The simplest FEA simulation is the single point simulation. It consists in the evaluation of a single (i_d, i_q) point, defined from current amplitude (in p.u. of the rated current) and current angle, computed from the d-axis. The results folder is named *T_eval_xxAx_xxdxx_xxdeg* with information of current amplitude, current angle and PM temperature (substitute "xx" with numbers). It is possible to input a vector of current amplitude and angles, running a sensitivity analysis (with parallel computing). In this case, an additional result folder will be created with the vectors resulting from this analysis. The name of the result folder is *senseOut_yyyymmddThhmmss* with information of date and time of the simulation.

4.2.1 Custom Phase Currents

It is possible to simulate custom phase currents (with current ripple from inverter, for example), by loading a proper file with the current waveform. This file should include 4 variables: `ia`, `ib`, `ic` and `th`, that are the phase currents and rotor position in one electrical period. This applies also for iron loss computation.

4.3 Flux Maps Evaluation

In this case the flux and torque maps over a regular (i_d, i_q) domain are computed. The domain is defined with current amplitude and number of current levels (valid for both axes). It is possible also to select the number of quadrants to simulate: 1, 2 or 4. The simulated quadrant change based on the rotor geometry:

- Circular, Seg, ISeg, Fluid use the SyR convention, so:
  - 1Q: $i_d > 0$ / $i_q > 0$
  - 2Q: $i_d > 0$
  - 4Q: full plane

- SPM and V-type use PM motor convention, so:
  - 1Q: $i_d < 0$ / $i_q > 0$
  - 2Q: $i_q > 0$
  - 4Q: full plane

The results folder is named *F_map_nnAnxnnAn_xxdeg_nQ*, with indication of the current limits, PM temperature and number of quadrants (change *n*). with numbers).

4.4 Iron Loss Evaluation

Iron loss can be computed for single point or flux maps. For both cases, the folder name is the same of other cases, with *_ironLoss* at the end. Iron loss are computed in FEMM by simulating the complete rotation and with good number of rotor positions. All the mesh data for each simulation must be downloaded from FEMM, so the computational time is relevant. The results are the stator and rotor hysteresis and eddy-current iron loss, and the PM loss, for a given simulation speed. The modified Steinmetz equation are adopted. Further information on the iron loss computation in SyR-e can be found here.

4.5 PM Motor Analysis

The PM motor analysis includes three types of simulations in SyR-e, specifically intended for PM motors. They are:

- Computation of the characteristic current function of the PM temperature

- Computation of the demagnetization current function of the PM temperature

- Check of demagnetized area for a given demagnetizing current and PM temperature

The first two analysis are based on iterative procedures, described here.

Dealing with the demagnetization analysis, all the mesh nodes of the PMs are imported in Matlab and the flux density along the magnetization direction is considered. To assess the demagnetization limit, 1% of the PM volume demagnetized is tolerated.

4.6 Short-Circuit Analysis

This analysis allows to estimate the peak short-circuit current that could occur during three-phase symmetric short-circuit. This estimation is identified with the acronym HWC that stands for "Hyper-Worst-Case", because phase resistance and all the other loss terms are neglected. The HWC current is function of the pre-fault point, that is the input of the analysis. After the single point simulation is performed, the HWC current is found with an iterative process (details available here). Also in this case, it is possible to input a vector of current amplitude-angle, so several pre-fault conditions are considered (reducing the number of iterations).

4.7 Specific FEA Analysis

Some other specific FEA analysis can be carried out. They consist of a single point operation, with additional inputs or output and include:

- **Flux density analysis**: the waveform of airgap, tooth and yoke flux densities for each rotor position simulated are exported

- **Current offset simulation**: allows to run the FEA simulation with a phase current offset (homopolar current) of 10% of the current amplitude

- **Airgap force computation**: exports also the radial force along the airgap.

4.8 Structural Analysis

Structural analysis is performed using the PDE Toolbox in Matlab and is still under development. The only input is the rotor speed, while mesh and boundary conditions are automatically set. All the rotor structure is modeled as a unique body, with material properties adapted if the region in steel, PMs, rotor bars or sleeve. For PMs and rotor bars, the Young's module is set as $\frac{1}{100}$ of steel Young's module to avoid structural support of this sections.

The mesh size is imposed to be between one tenth and one time the minimum mechanical tolerance (smaller mesh on the model edges), while the boundary condition is set to have zero displacement along the shaft surface and along the pole sides. This approximation is not formally correct (the correct boundary condition should be periodic), but is easier to set and gives similar results for the considered geometries.

Dealing with rotor sleeve, the simulation is still under development and validation against more complex FEA solvers.

The rotor geometry is loaded from FEMM, allowing structural validation also for custom geometries.

Chapter 5

Export to other FEA Software

The export to other (commercial) FEA software is possible, but not the complete features are supported.

5.1   DXF Export

SyR-e support the export to dxf file, through the function `syreToDxf()`. The motor file can be selected and the dxf geometry is exported in a new file. Please, note that just the geometry is exported, and not boundary definitions, blocks definitions, materials, and so on.

5.2   SIMCENTER MagNet Export

The export to Simcenter MagNet is almost fully supported, through the GUI. There are two red buttons, close to *Save* button and to *Start* simulation button, named **MN**. These two commands allows the export and simulation in Simcenter Magnet respectively. In this case, the FEA model is completely set-up (i.e. materials, windings, boundaries,:.) and the *Transient with Motion* simulation is performed, mainly to get iron loss.

5.3   Ansys Motor-CAD Export

Ansys Motor-CAD export is managed from the relative tab in the main GUI, reported in Figure 5.1. Also in this case, the model is completely set-up, and it is possible to run both electromagnetic and thermal simulations.

5.4   Ansys Maxwell Export

The export in Ansys Maxwell works is at the very first development stage. In principle, it works as Simcenter MagNet export, with dedicated buttons (*Ansys*, in blue) to save the model and launch FEA simulations. Since it is at the very beginning development stage, not all the simulations can be performed in Ansys
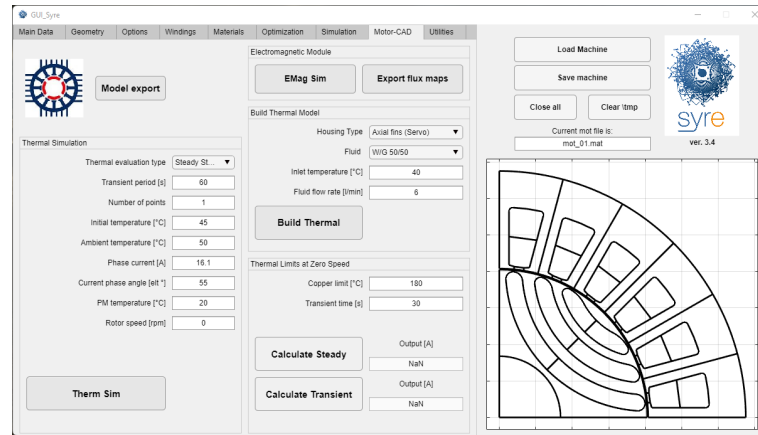
Figure 5.1: Motor-CAD tab in main GUI

Maxwell from SyR-e, but it is always possible to launch the FEA model outside the SyR-e environment.

Chapter 6

Magnetic Model Manipulation

The routines for the manipulation of the flux maps was included in SyR-e from its first release. However, there was a collection of scripts without an intuitive interface to operate and deeply understand their working. In April 2020, a second GUI of the SyR-e project is released, ruling and organizing the flux maps elaboration routines and an easy and understandable method to operate with motor model.

This GUI is identified with the acronym MMM, that stands for Magnetic Model Manipulation. From the MMM GUI there is not possible to launch FEA simulations, but just load simulated data and post-process them.

6.1   Getting Started with MMM GUI

The MMM GUI is launched with the command `GUI_Syre_MMM`. The GUI is composed of 6 tabs:

- Main: used to load models and do simple manipulations, like inverse maps, control trajectories computation, ...

- Scale & Skew: used to compute flux maps of scaled and skewed motors;

- Torque-Speed: used to compute the characteristic in the torque-speed domain;

- syreDrive: interface with Simulink and the dynamic model simulation;

- Waveform: used to compute waveform as single working point (from $dq\theta$ maps) and transient short-circuit;

- Thermal: used to compute thermal limits (under development).

The main window of MMM GUI is reported in Figure 6.1. On the right a table with the motor rating is reported and always visible, while on the right top of the GUI, there are the buttons to load and save models.
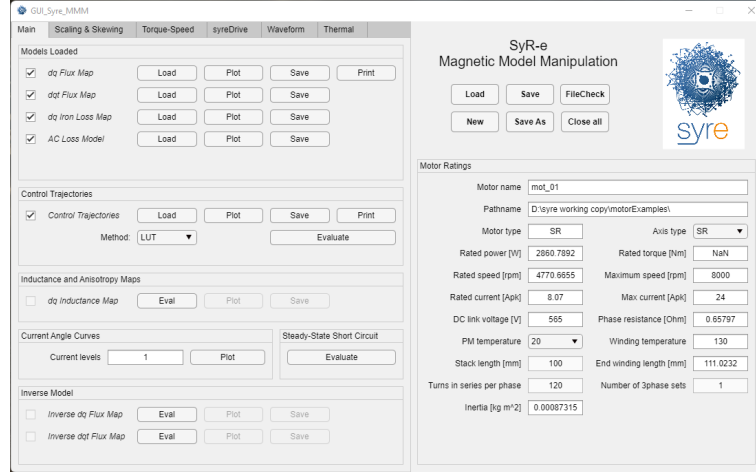
Figure 6.1: Main window of the MMM GUI

6.1.1   Data Structure

The main data structure behind the MMM procedures is called `motorModel` and it is saved in the motor .mat file from the MMM GUI (not from the main GUI!). The `motorModel` structure is composed from different substructures, that can be empty if the respective model in not loaded. They are:

- `data`: contains the ratings of the motor;

- `FluxMap_dq`: contains the flux maps of the motor in the *dq* domain (fundamental model);

- `FluxMap_dqt`: contains the *dqθ* flux maps of the motor, so function of *dq* currents and rotor position;

- `IronPMLossMap_dq`: contains the iron loss and PM loss maps, function of *dq* current and at given speed, with also the coefficients to scale in speed;

- `acLossFactor`: contains the AC loss factor, function of temperature and frequency;

- `controlTrajectories`: contains the control trajectories, like Maximum Torque per Ampere (MTPA), Maximum Torque per Voltage (MTPV), and so on;

- `IncInductanceMap_dq`: contains the incremental inductance maps, useful for the control system simulation;

- `FluxMapInv_dq`: contains the inverse flux maps in the *dq* domain, so currents and torque function of *dq* flux linkages;

- `FluxMapInv_dqt`: contains the inverse flux maps in the *dqθ* domain;

- `TnSetup`: contains the set-up for the torque-speed evaluations, so operative limits and efficiency maps;

- `SyreDrive`: contains the setup for *syreDrive*, the export to dynamic model simulation;

- `WaveformSetup`: contains the setup for the extraction of waveform from $dq\theta$ model and transient short-circuit computation;

- `tmpScale`: contains the scaling factor, used during the scaling procedure;

- `tmpSkew`: contains the data for skewing, used during the skewing procedure;

- `Thermal`: contains the data for thermal computation (under development).

6.1.2   New, Load, Save and Check Model

The main model operations are done with the buttons at the top right of the GUI. There is six operations that could be done:

- *Load*: load a new motor model. The target is load the .mat file of the motor, and eventually add flux maps and other manipulations.

- *New*: allows to create an empty `motorModel` structure, useful if experimental flux maps must be loaded and post-processed, and in general, if the SyR-e file is not available.

- *Save*: allows to update the current motor file with the motorModel structure.

- *Save As*: allows to save the motor model, with the `motorModel` structure. New file must be created.

- *File Check*: allows to check the motor file, if it is somethings corrupted (under development).

- *Close all*: close all the figures.

6.1.3   Temperatures Manager

It is possible to manage flux maps at different PM temperatures thanks to the PM temperature selector in the ratings section. Loading a new PM temperature will create a cache file in the MMM results folder, with all the PM temperature maps saved, and the data will be loaded from there.

6.2   Load Data

The flux map data are loaded from the top section of the main tab. Four kind of models can be loaded:

- *dq Model*: is the fundamental flux map, function of *dq* currents;

- *dqtMap Model*: is the $dq\theta$ flux map, with the rotor position dependency;

- *Iron Loss Model*: is the iron and PM loss map model, function of *dq* current and computed at single rotor speed, with the speed-scaling factors;

- *AC Loss Model*: is the AC loss factor, function of frequency and temperature.

There are three operations that can be done on this models, that are represented with the three buttons close to the model name:

- *Load*: load the model, from a coherent data structure (FEA results from SyR-e simulations are correct);

- *Load*: allows to plot the selected model and eventually save the pictures;

- *Save*: allows to save the model data (just the selected) in a file format compatible with the old SyR-e version.

In addition, the button *Print* allows to print the model in the .c format, ready for the drive control system. The models can be deleted by de-checking the check-box that indicate which model or elaboration are loaded.

6.3   Simple Flux Maps Manipulation

All the simple flux maps manipulations can be carried out from dedicated windows in the main tab of the MMM GUI. They are:

- Computation of the control trajectories;

- Computation of the incremental inductance maps;

- Computation of the inverse model;

- Plot the current angle dependency plots;

- Compute the steady-state short-circuit condition, function of speed.

6.3.1   Computation of the Control Trajectories

The main control trajectories that are computed are the Maximum Torque per Ampere (MTPA) and the Maximum Torque per Voltage (MTPV), that is equivalent to the maximum torque per flux linkage (if loss are neglected). The computation is quite simple and fast. The only parameter that is needed is to select if the trajectories can be expressed as raw Look-Up Tables (LUTs) or must be fitted.

Once the control trajectories are evaluated, it is possible to plot them, save or print in the .c format, compatible with control systems.

6.3.2   Computation of the Incremental Inductance Maps

The incremental inductance maps can be useful for some control algorithms. They are computed as the gradient of flux linkage divided by current gradient. The results are four maps (one each incremental inductance), plus an additional map representing the incremental anisotropy.
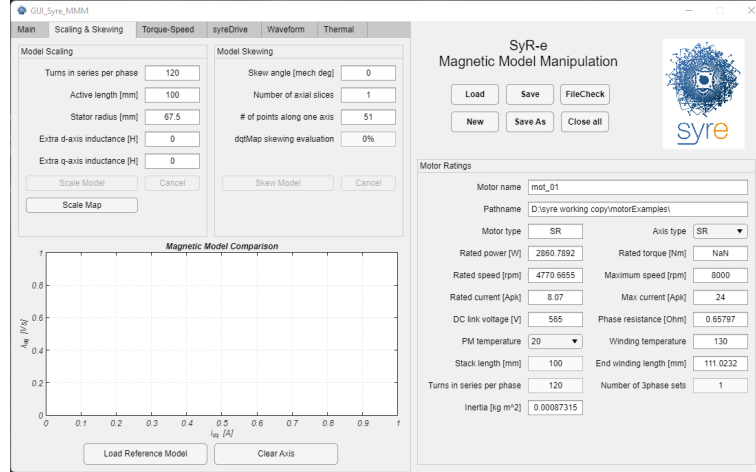
Figure 6.2: Scaling and skewing tab in MMM GUI

6.3.3 Computation of the Inverse Model

Inverse flux maps express currents and torque function of the dq flux linkages and are useful for the dynamic model simulation. The results are rectangular maps in the (λ_d, λ_q) domain, so a data loss at the borders is accepted. Further details on the elaboration can be found here. The process can be performed both for dq and $dq\theta$ models.

6.3.4 Plot of the Current Angle Dependency

This quick and simple manipulation allows to see some behaviors typical of synchronous reluctance machines. The plots of torque, torque ripple, flux linkage amplitude and angle and power factor, function of the current angle and at fixed current amplitude are plotted and eventually saved. It is possible to select the current levels, expressed in p.u. of the rated current.

6.3.5 Computation of the Steady-State Short-Circuit Conditions

From the fundamental flux maps it is possible to compute the steady-state short-circuit condition, function of the rotor speed. The procedure is explained here and the computation is performed with the rated phase resistance (possible to change value or temperature) and up to the maximum motor speed.

6.4 Scaling and Skewing Flux Maps

The scaling and skewing procedure allows the computation of similar motors with very limited computational effort, since no FEA simulations are performed. The two operations are controlled from the dedicated tab in the MMM GUI, reported in Figure 6.2.

The two operations cannot be done in parallel. It is possible to start editing the parameters and press the button for computation to see the results. Then the motor must be saved (not *Save*, but *Save As*) or it is possible to come back to the original model with the button *Cancel*.

Further information about the scaling and skewing procedure can be found here and here.

6.4.1 Motor Scaling

The parameters that can be changed with scaling procedure are:

- number of turns in series per phase

- active length

- stator outer radius

- additional inductance (3D effects) on d and/or q axis.

The scaling procedure acts on the dq and $dq\theta$ flux maps and iron loss. The AC model is deleted, but a new motor is saved, with the FEMM model, so further FEA simulations can be performed.

It is also possible to compute a performance map function of stack length and number of turns and constant inverter limits, by selecting *Scaling Map*. This features enables the optimization of the selected design.

6.4.2 Motor Skewing

The parameters that must be selected are the skew angle and the number of axial slices. If the $dq\theta$ model is loaded, so torque ripple is computed. Further details can be found here.

6.5 Torque-Speed Computations

The computation of the torque-speed behaviors of the motor are done in the dedicated tab, reported in Figure 6.3.

There are two procedures that can be followed, to get the operating limits and the efficiency maps.

6.5.1 Operating Limit Computation

The operating limits computation is a very fast and simple procedure that identify the operative limits function of the speed, given the voltage and current limits and neglecting all the loss terms except for the phase resistance.

Current limits can be set from the input window in per-unit of the rated current, while the voltage limit is taken from the DC link voltage, without margin.

The results of the computation are torque, power, current, voltage and flux linkage function of speed, together with the control locus on the (i_d, i_q) plane.
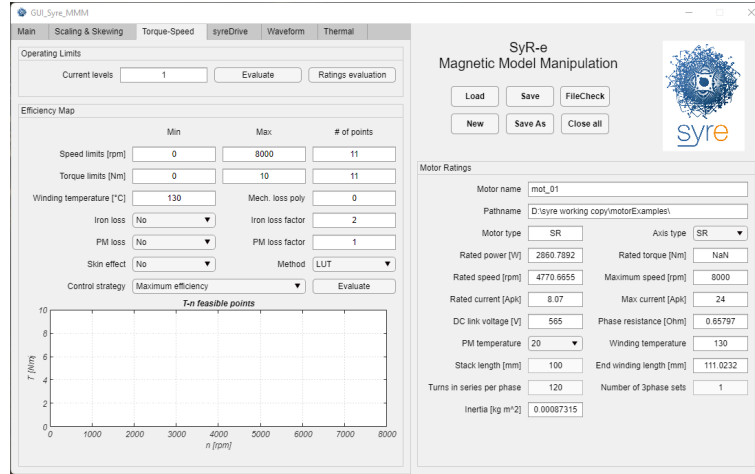
Figure 6.3: Torque-Speed tab of MMM GUI

6.5.2 Efficiency Map Computation

The efficiency map computation is more complete and heavy than the operating limits computation. In this case, a regular grid on the (T, n) plane is defined and explored. Copper loss are always considered (also at temperature different from the rated) and it is possible to account also for iron loss, PM loss, AC loss and mechanical loss. The first three terms are related to the loaded model, and for iron and PM loss, it is also possible to define a correction factor. Dealing with the mechanical loss, it is defined as a polynomial function of the speed in rpm. The control strategy can be selected (maximum efficiency or MTPA) and voltage limit is taken from the DC link voltage without margins.

The results are matrices function of torque and speed that reports all the loss terms, the current (phase and magnetizing), voltage and efficiency of the motor.

Further information on the procedure can be found here.

6.6 Export to Dynamic Model Simulator

The export to dynamic model in Simulink is a feature called *syreDrive* and it is controlled from the dedicated tab of the MMM GUI, reported in Figure 6.4.

From the tab, it is possible to select:

- type of model: average model (with S-function) or Instantaneous model, with circuital modeling;

- type of control: if current, torque or speed control;

- type of motor model: if fundamental (based on *dq* maps) or with harmonics (based on *dqθ* maps);

- converter data, like ON threshold and internal resistance of the modules and the dead time;

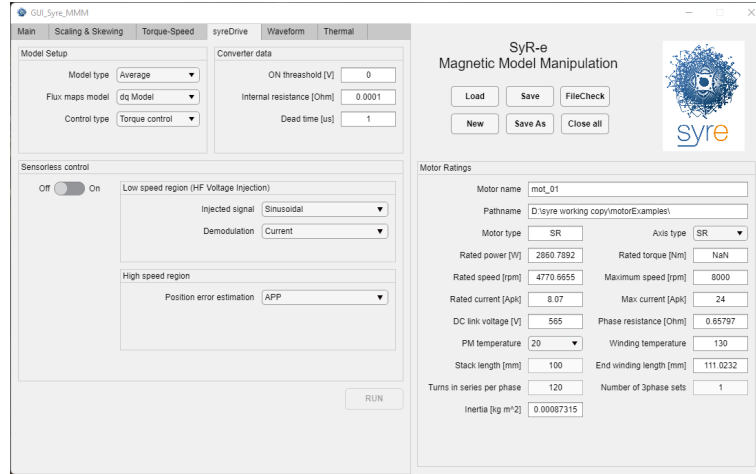- type of sensorless control, if needed
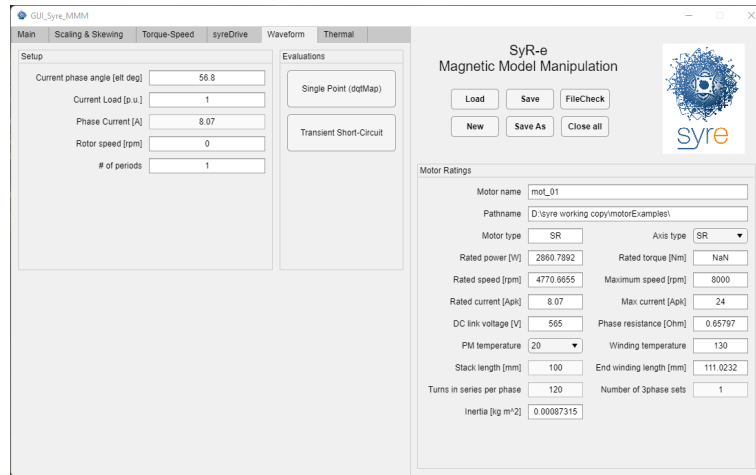
Figure 6.4: syreDrive tab of the MMM GUI

Figure 6.5: Waveform tab of the MMM GUI

At the moment, the export is done in Matlab/Simulink, but work on other dynamic model solvers is ongoing.

6.7 Waveform Computation

The waveform tab, reported in Figure 6.5, collect two post-processing that allows to get waveform of the single operating point and the transient short-circuit evolution.

The inputs are similar to the Simulation tab of the main GUI, and two buttons define the two elaboration. The single point waveform is based on the retrieve of the waveform from the $dq\theta$ map model and give similar results of the FEA simulation. On the other side, the transient short-circuit computation evaluate the evolution of the three-phase short-circuit in transient, starting from the selected pre-fault condition and for a number of cycles defined from the input. Further details available here

Chapter 7

Working without GUIs

It is possible to use SyR-e scripts without GUI. This is usually done from Octave users or expert users that need to run automatic routines. In general is always possible to launch functions that are in the call-back of the GUIs, but in some cases, it could be difficult to run some data check. It is always needed to load the motor model and resolve the back-compatibility issues. Here a (non complete) list of functions useful to operate with SyR-e without GUIs:

- `setupPath` : add all the folders needed to operate with SyR-e.

- `OpenSaveOCT` : open a motor model, edit some fields and save it.

- `OptimizeOCT` : launch the optimization process.

- `SimulateOCT` : launch FEA simulations

- `eval_operatingPoint` : evaluation of the single operating point with FEA.

- `eval_fluxMap` : evaluation of flux maps with FEA.

- `MMM_load` : create `motorModel` structure from SyR-e file

- `MMM_eval_AOA` : compute control trajectories.

- `MMM_scale` : scale motor model.

- `MMM_skew` : skew motor model.

- `MMM_eval_inverseModel_dq` : evaluate the inverse model.

- `MMM_eval_shortCircuitTransient` : evaluate the transient short-circuit.

- `MMM_eval_OpLim` : evaluate the operating limits.

- `MMM_MaxTw` : evaluate the efficiency map.

7.1 Custom Features

In the main SyR-e repository is included the folder *syreCustomFeatures*. This path and all the subfolders are added to the Matlab path as all the other SyR-e functions. Here it is possible to add custom functions to operate with SyR-e scripts and do extra post-processing. There are two examples included in the SyR-e release, that deals with additional post-processing of operating point simulation and a benchmark for timing, that is also an example of scripting.

In general this function are not included in the GUI environment, but can be used as standard Matlab functions.

7.1.1 Single Point additional Post-Processing

This feature add some extra post-processing to the results of the single point evaluation. There are two functions that can be launched:

- `plot_FFT_singt` : compute and plot the FFT of torque and *dq* flux linkage waveform;

- `plot_vectorDiagram_singt` : plot the vector diagram of the simulated point.

7.1.2 Timing Test

This folder contain the script `autoSyRe_TimingTest`, that launch some FEAfix simulations and a flux maps, recording the computational time on a logfile. This could be also used as example of scripting using SyR-e functions.

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.

Contacts and References

SyR-e is an open source project, originally born from a cooperation between Politecnico di Bari and Politecnico di Torino. During the years, several professors, researchers and students contribute to the project.

At the moment, SyR-e is mainly maintained from the Power Electronics Innovation Center (PEIC) members, at the Politecnico di Torino (PEIC@PoliTO).