Politecnico di Torino

01RGBRV

Optimization methods for engineering problems

Multi-objective NSGA-II/Pareto approach

Stefano Emilio CARIA

Gaetano DILEVRANO

Paolo RAGAZZO

Lorenzo SOLIDA

June, 7th 2022

Outline

- Multi-objective optimization
  - *Introduction*
  - *The Pareto front*
- NSGA-II
  - *from EMOA to NSGA-II*
  - *fast nondominated sorting procedure*
  - *diversity preservation*
  - *elitism*
- Library validation
  - *POL problem*
  - *Results*
- Case study
  - *SyR-e*
  - *Workflow*
  - *Original machine*
  - *Optimized machine (short optimization)*
  - *Optimized machine (long optimization)*
- Conclusions

Multi-objective optimization: introduction

When more than one objective function is defined, objectives can be:

- Coherent, if changing the decision variables in the way to improve one of the objective functions causes an improvement also in the other(s) objective function(s)
- Conflicting, if changing the decision variables in the way to improve one of the objective functions causes the other(s) to worsen

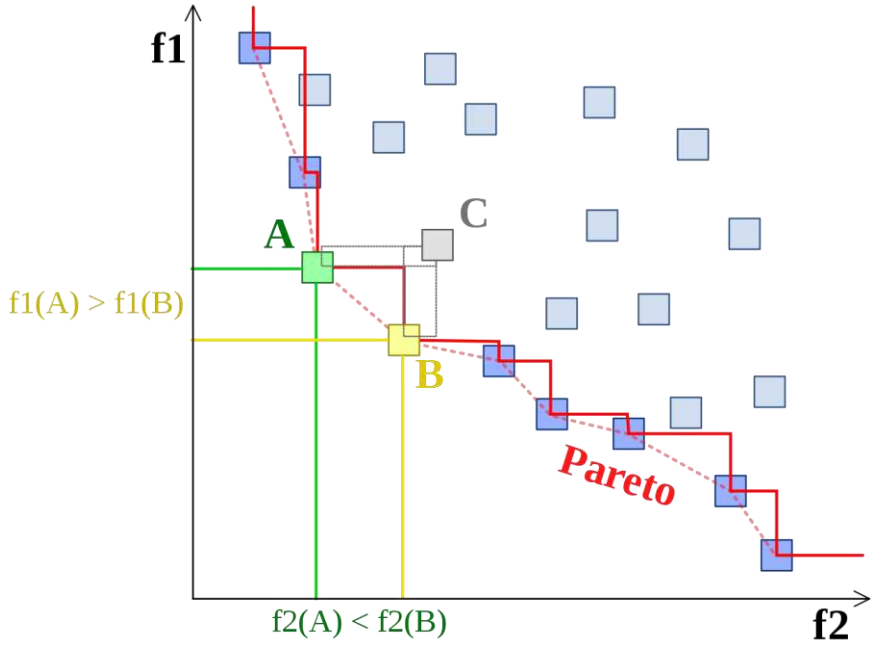
Typical example

- Operation cost VS Quality of the service

Multi-objective optimization: the Pareto front

When conflicting objectives arise, two strategies can be adopted:

- Classical optimization methods suggest converting the multiobjective optimization problem to a single-objective optimization problem by assigning a weight to every objective
- Multi-objective Optimization give rise to a set of Pareto-optimal solutions: the Pareto-optimal front includes individual optimal solutions and many other compromised solutions which are optimal with respect to certain trade-offs among the different objectives



In non-dominated sorting, an individual A is said to dominate another individual C, if and only if there is no objective of A worse than that objective of C and there is at least one objective of A better than that objective of C.

NSGA-II: from EMOA to NSGA-II

Evolutionary Multi-objective Optimization (EMO) algorithms

- population approach
- use of non-domination
- use of a diversity-preserving mechanism

Pros:

- ⊕ Set of solutions (Pareto front)
- ⊕ Well-diversified solutions

Cons:

- ⊖ Can be very computational expensive: $O(MN^3)$ computational complexity (M is the number of objectives, N is the population size)
- ⊖ Lack of elitism
- ⊖ Based on *"sharing parameter"* to ensure diversity in population

Nondominated Sorting Genetic Algorithm - II (NSGA-II) innovations:

- fast nondominated sorting procedure
- elitist-preserving approach
- parameterless algorithm for measure diversity

NSGA-II: fast nondominated sorting procedure

Traditional approach

Each solution of the population is compared with every other solution to check if it's dominated

- total complexity is $O(MN^2)$
- this procedure allows to identify the first nondominated front

The procedure has to be iterated to identify next nondominated fronts, excluding already considered solutions

- worst case can require $O(MN^2)$ for each of the fronts

NSGA-II approach

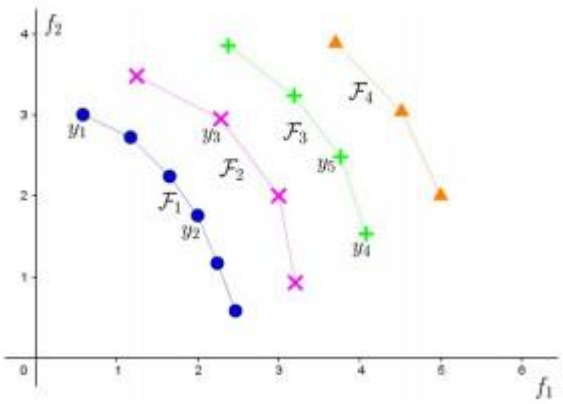
For each solution two entities are computed:

- domination count n_p, the number of solutions which dominate the solution p
- domination list S_p, a set of solutions dominated by solution p

This procedure is $O(MN^2)$ computational expensive

Build the nondominated fronts:

- start from solutions with $n_p = 0$: they represent first front
- follow their dominations list S_p and decrement the n_p counter for each q member
- each q solution with $n_p = 0$ is part of the new front



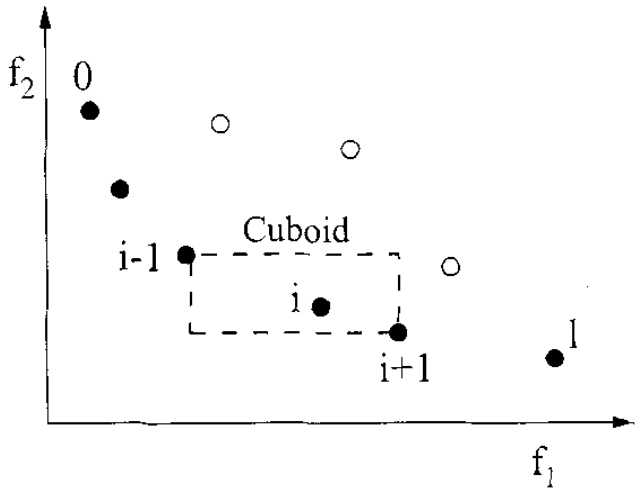
NSGA-II: diversity preservation

Traditional approach

Sharing function approach, based on σ_{share} parameter, related to the distance metric chosen to calculate the proximity measure between two population members

Cons:

- Performances depends on the chosen σ_{share} parameter
- Each solution must be compared with all other solutions in the population, the overall complexity of the sharing function approach is $O(N^2)$



NSGA-II approach

Crowding distance

- defined as average side length of the cuboid
- computational complexity related to sorting algorithm $O(MN \log N)$

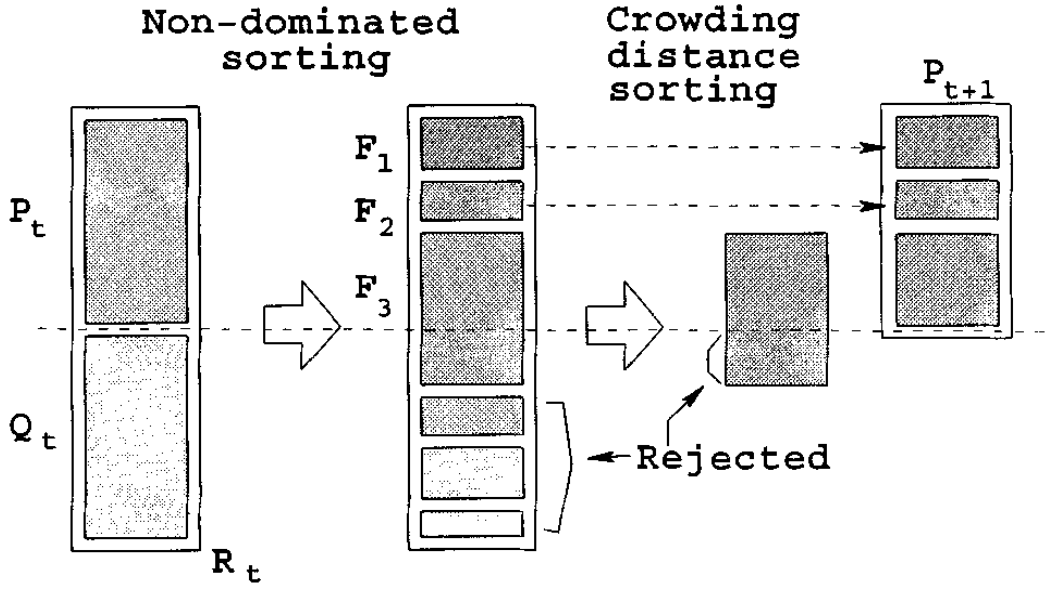
Crowded-comparison operator

- between two solutions with differing nondomination ranks, the solution with the lower (better) rank is preferred
- if both solutions belong to the same front, the solution that is located in a lesser crowded region is preferred

NSGA-II: elitism

NSGA-II algorithm

1. A random parent population P_0 is created
2. Population P_0 is sorted according with nondomination
3. Binary tournament selection, recombination, and mutation operators are used to create a offspring population Q_t of size N
4. Combined population $R_t = P_t \cup Q_t$ is formed. The population R_t is of size $2N$
5. Population R_t is sorted according with nondomination
6. Members for next generation are selected with 2 rules:
   - solutions belonging to the best nondominated front F_1 are of best solutions and must be chosen firstly. Then, solutions from the F_2 front are chosen next, followed by solutions from the F_3 front, and so on, until N solutions are selected
   - to choose exactly population members, solutions of the last front are sorted using the crowded comparison operator in descending order and choose the best solutions needed to fill all population slots
7. Procedure is repeated from step 3



Library validation: POL problem

To validate the chosen library, we decided to test with a benchmark literature problem, the *POL problem*.

The objective functions are:

$$f_1(x_1, x_2) = 1 + (A_1 - B_1)^2 + (A_2 - B_2)^2$$

$$f_2(x_1, x_2) = (x_1 + 3)^2 + (x_2 + 1)^2$$

where:

$$A_1 = 0.5 \cdot \sin 1 - 2 \cdot \cos 1 + \sin 2 - 1.5 \cdot \cos 2$$

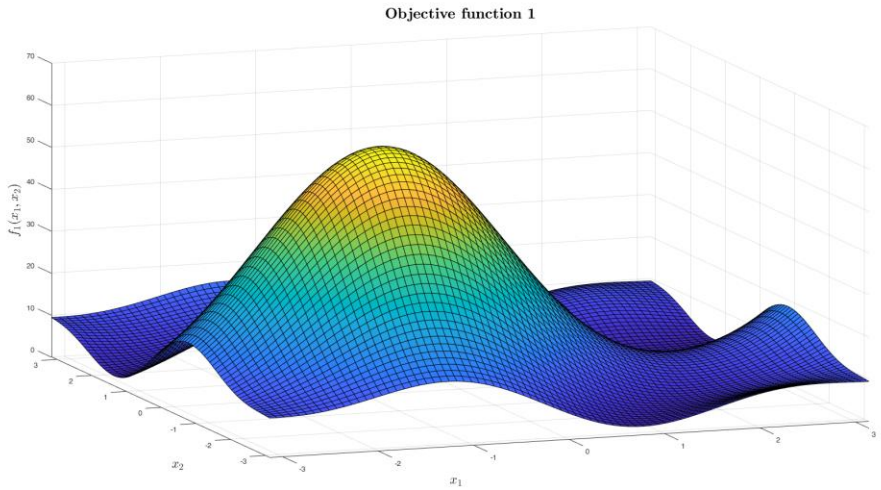
$$A_2 = 1.5 \cdot \sin 1 - \cos 1 + 2 \cdot \sin 2 - 0.5 \cdot \cos 2$$

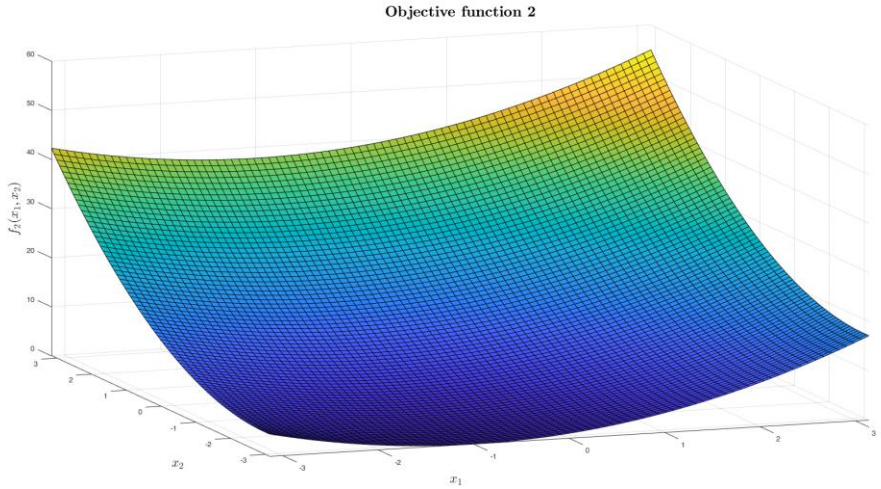
$$B_1 = 0.5 \cdot \sin x_1 - 2 \cdot \cos x_1 + \sin x_2 - 1.5 \cdot \cos x_2$$

$$B_2 = 1.5 \cdot \sin x_1 - \cos x_1 + 2 \cdot \sin x_2 - 0.5 \cdot \cos x_2$$

The variable field of existence is:

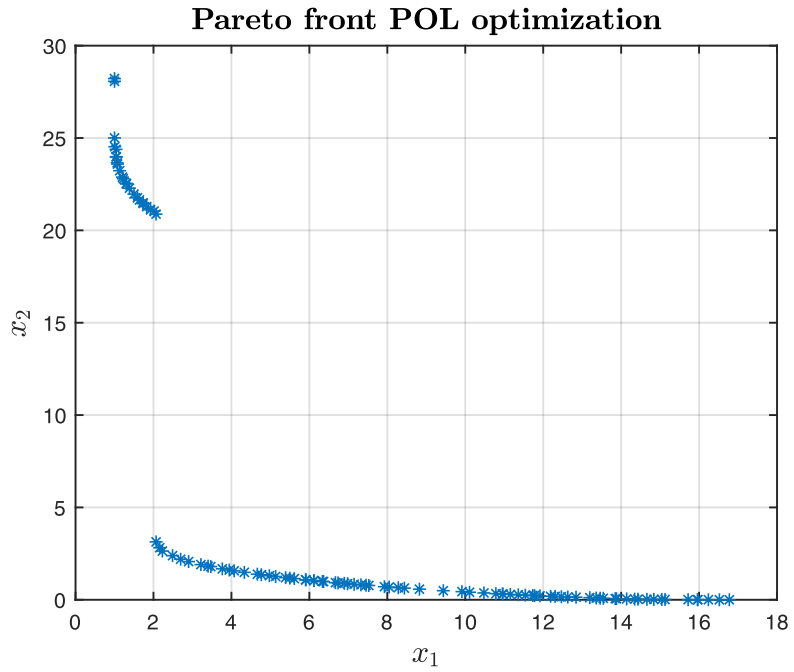
$$x_1, x_2 \in [-\pi, \pi]$$



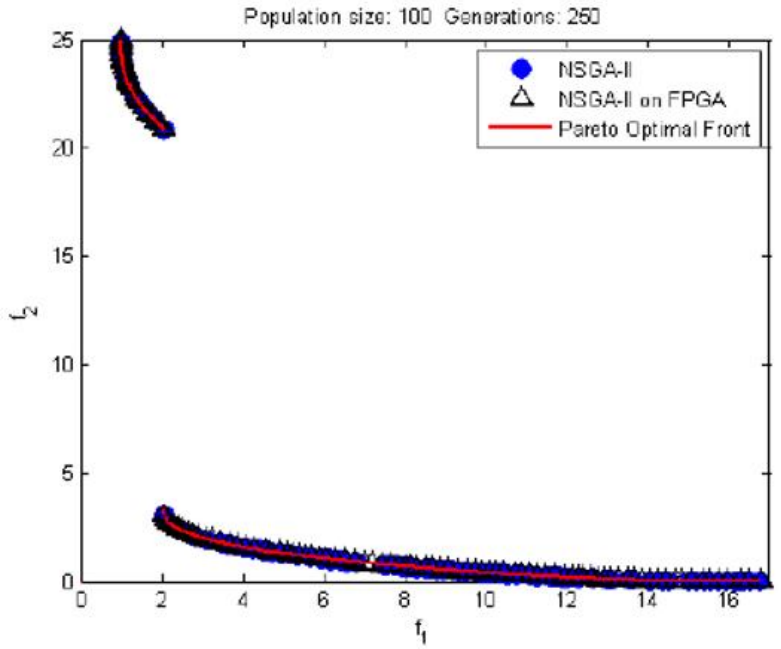


Library validation: results

Results are coherent with literature



Results of the optimization from our library with a population size of 100 and 100 generations

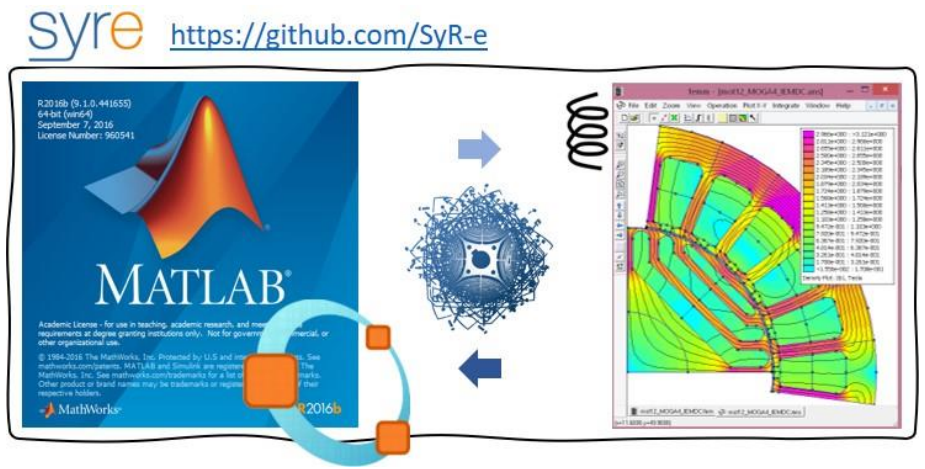


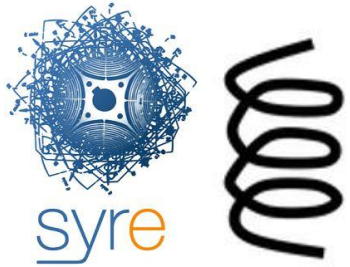
Results from Kok, Gonzalez et al. *"An FPGA-based Approach to Multi-Objective Evolutionary Algorithm for Multi-Disciplinary Design Optimisation"* (2011)

Case study: SyR-e

SyR-e stands for Synchronous Reluctance – evolution and is an open-source Matlab-based environment for synchronous motor design and evaluation

- It is available on GitHub and SourceForge, new public version online
  https://github.com/SyR-e/syre_public/releases/tag/v3.1
- Covers SyR, PM-SyR, IPM and SPM machine types
- Uses FEMM for 2D magnetostatic FEA
- Linked with commercial CADs: Simcenter MagNet and Ansys Motor-CAD

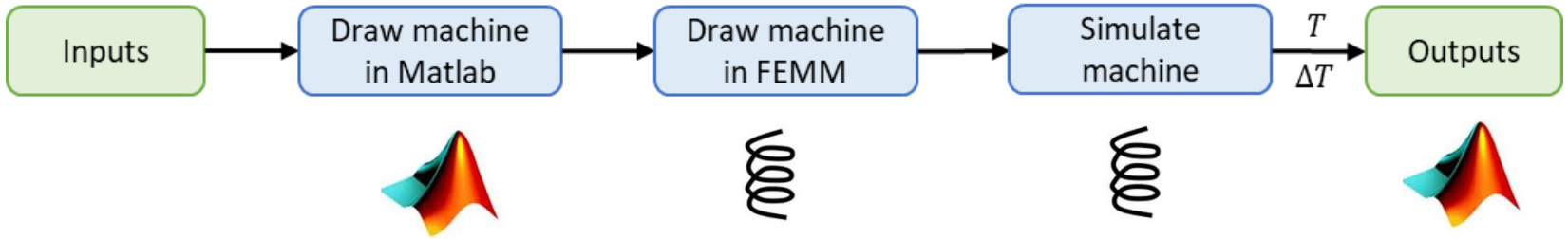




Case study: workflow

The objective function is modified by implementing FEMM simulations of e-motors.

- The inputs of the function are the variables values
- The machine geometry is varied accordingly to the input variables and the geometry feasibility checked
- The machine is draw in FEMM by means of Matlab scripts and ActiveX
- The FEMM model is simulated and the results collected in Matlab



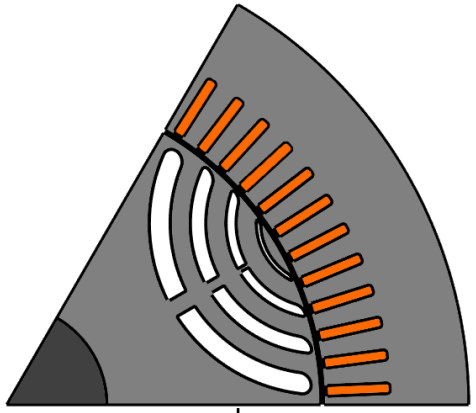
Case study: original machine

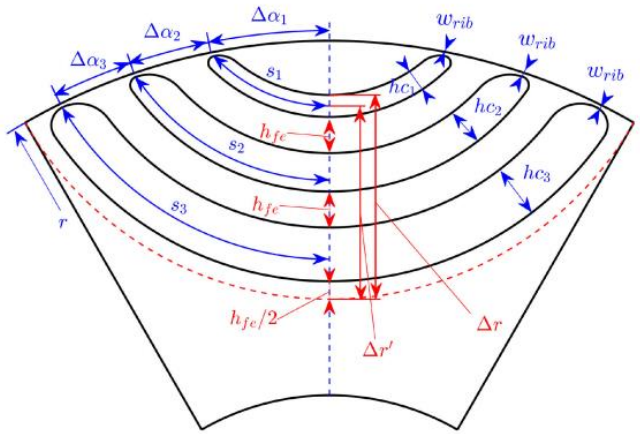
The optimization process presented before, it thus applied to a **synchronous reluctance machine** for automotive application.

- The rotor geometry is optimized by means of 3 degrees of freedom for each barrier
- The average torque is maximized and the ripple minimized.

Variables name	Number of variables
Barriers position	4
Barriers width	4
Barriers offset	4
Current phase angle	1
Total	13
Objectives	
Mechanical torque	[Nm]
Pk-pk torque ripple	[Nm]

Initial machine





- Torque: 535 Nm
- Ripple: 70 Nm

Case study: optimized machine (short optimization)

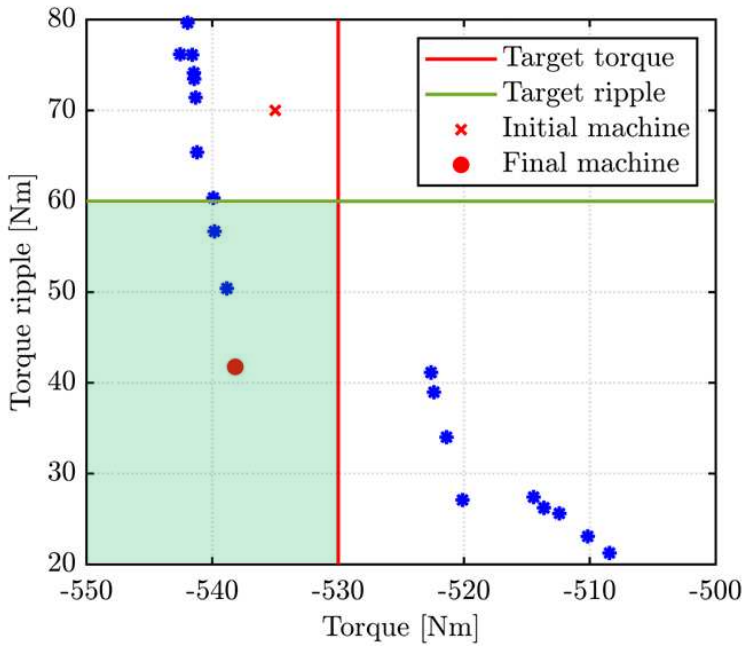
The optimization process presented before, it thus applied to a **synchronous reluctance machine** for automotive application.

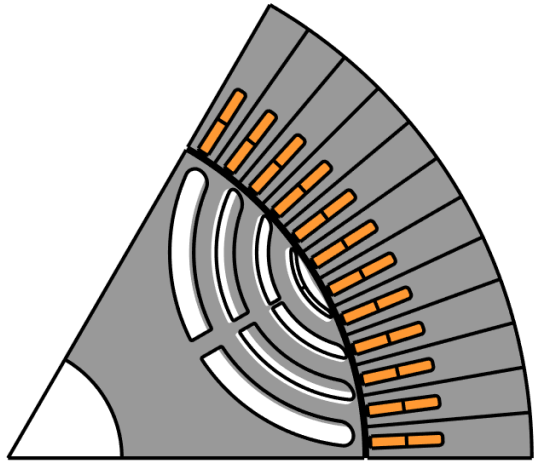
Computation time ≈ 4.5 h

ReFreeDrive

Variables name	Number of variables
Barriers position	4
Barriers width	4
Barriers offset	4
Current phase angle	1
Total	13
Objectives	
Mechanical torque	[Nm]
Pk-pk torque ripple	[Nm]

20 generations with 20 elements





Lines: initial version
Colored: final version

Case study: optimized machine (long optimization)

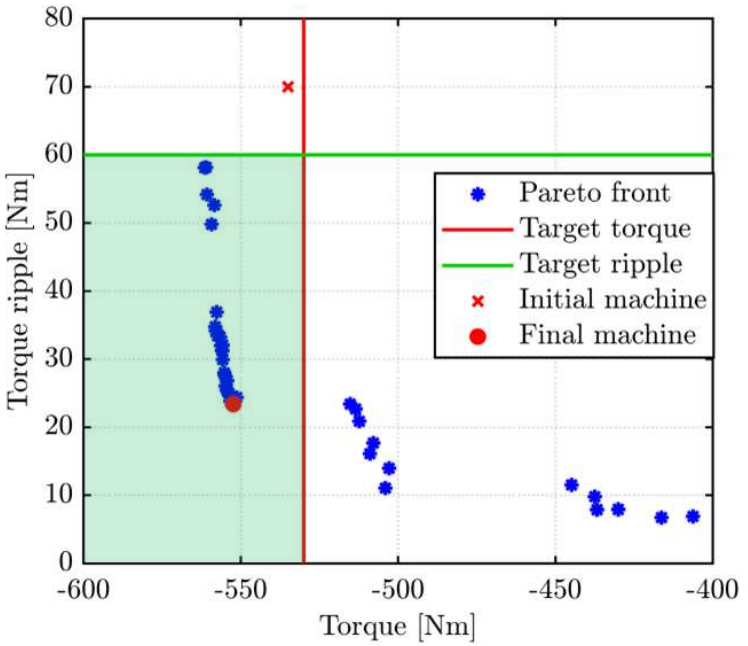
The optimization process presented before, it thus applied to a **synchronous reluctance machine** for automotive application.

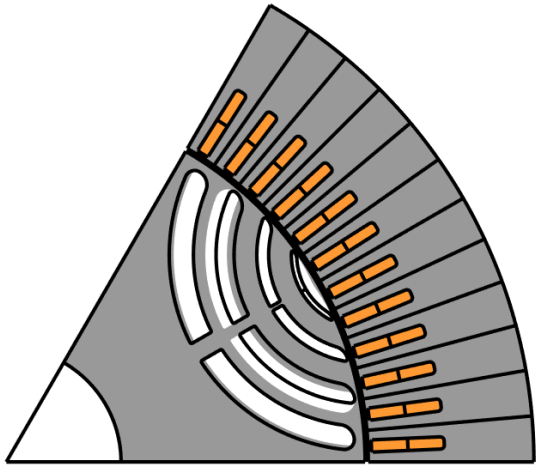
Computation time ≈ 16 h

ReFreeDrive

Variables name	Number of variables
Barriers position	4
Barriers width	4
Barriers offset	4
Current phase angle	1
Total	13
Objectives	
Mechanical torque	[Nm]
Pk-pk torque ripple	[Nm]

40 generations with 40 elements





Lines: initial version
Colored: final version

Conclusions

- The algorithm is employed to fulfill the specification of a traction motor
- Aims to optimize the rotor geometry changing 12 (+1) parameters
- The object functions are torque and torque ripple
- The procedure can be easily extended to stator parameters (yoke width, slot dimensions…) and other objective functions can be added (power factor, structural validation…)

Politecnico di Torino

THANK YOU FOR YOUR ATTENTION

Stefano Emilio CARIA

Gaetano DILEVRANO

Paolo RAGAZZO

Lorenzo SOLIDA