Optimal Design of Electrical Machines: Mathematical Programming Formulations *

Sonia Cafieri[1] Leo Liberti[2] Frédéric Messine[3]
Bertrand Nogarede[4]

[1] *École Nationale de l'Aviation Civile (ENAC, Lab. MAIAA), 7 avenue E. Belin F-31055 Toulouse France, Email:sonia.cafieri@enac.fr*

[2] *LIX, École Polytechnique, 91128 Palaiseau, France. Email:liberti@lix.polytechnique.fr*

[3] *ENSEEIHT-IRIT, Université de Toulouse, 2 rue C. Camichel, 31071 Toulouse, France. Email:messine@n7.fr*

[4] *ENSEEIHT-LAPLACE, Université de Toulouse, 2 rue C. Camichel, 31071 Toulouse, France. Email:nogarede@n7.fr*

Abstract

The optimal design of electrical machines can be mathematically modeled as a (mixed-integer) nonlinear optimization problem. We investigate the impact of different mathematical formulations on the results obtained using a local optimization solver which is well-known in the engineering community: *MatLab*'s `fmincon` function. Our analysis is based on six different mathematically equivalent formulations for the same problem of the design of an electrical machine without slot. Our results underline the important impact that formulation differences may have on solver performance even on a small example of design.

Keywords: analytical model, formulation, modeling, local optimization, inverse problem, design, electrical machine

*The second and third author were partially supported by grant ANR 07-JCJC-0151 "ARS".

1 Introduction

The design of electromechanical actuators is known as an *inverse problem*, i.e. from the characteristic values given by the schedule of conditions (for example the torque), obtain the *structure*, the *dimensions* and the *material compositions* of the actuator constitutive parts, [1, 2, 13]. One is usually interested in performing an optimal design where a given criterion is optimized (for example, the volume of the magnet is minimized). The interest of the *electromagnetic actuators design* combining *optimization algorithms* and *analytical models* has in fact already been widely shown in the literature [10, 9, 5, 14, 15, 16, 17]. The *inverse problems* of electromechanical actuator design are more general than *optimal dimensioning problems* [2]. These problems must be formulated as mixed-constrained optimization problems, see [1, 2] for complete formulations. Optimization problems of this kind are in general quite difficult to solve to global optimality. Exact global optimization solvers can however be adapted for their solution. Methods based for example on interval analysis have been proposed in [2, 11, 9, 10]. Other recent methods for global optimization of mixed-integer problems can be applied as well. In practice, however, the picture is not as simple: recent powerful solvers are not always publically available (this is the case for interval analysis based solvers) or do not interface to a modelling environment for easy using. In practice, in the context of engineering applications, one usually prefers to resort to well known and easy to use optimization solvers [13, 14, 15, 17]. In this sense, a very successful solver is provided by the *MatLab*'s `fmincon` function [8]. It is based on Active-Set and Sequential-Quadratic Programming (SQP) methods with computation of the Hessian from Quasi-Newton techniques [3, 4] and finds local optima of nonconvex nonlinear optimization problems with continuous decision variables. This solver does not represent the state of the art for nonlinear nonconvex optimization and, being a local solver, does not provide global solutions. However, it usually provides good quality (local) solutions which are often considered satisfactory in practical contexts.

Even though this kind of solver is quite easy to use, the mathematical features of the optimization problem may have an impact on the practical efficiency of the solver. It is known that the same optimization problem can be often formulated in different ways, sharing some properties such as optima and feasible regions. Furthermore, the formal description of optimization problems has an impact on the applicability and efficiency of the corresponding solution methods. Indeed, the study of reformulations is an active research area in the optimization community [6, 7]. The aim of this paper is to investigate the efficiency and reliability of standard local optimization

solvers when handling different mathematical formulations. We illustrate the impact of such different formulations on solver performance. While providing any advancement on the global solution of engineering design problems is out of the scope of this paper, we consider a local solver which is widely used in the engineering community for optimal design, with the aim of providing guidelines for designers in practical engineering applications.

Our application testbed is a slotless electrical rotating permanent magnet machine. This example was first presented in [14] and also studied in a lot of papers such as [11, 14, 9, 17]. The analytical equations of the considered design problem, which come from approximations of Maxwell's equations (taken in the quasi-static mode) and mechanical considerations, can be found in [11, 12, 14]. These equations are recalled below:

$$\Gamma_{em} = \frac{\pi}{2\lambda}(1 - K_f)\sqrt{k_r \beta E_{ch} E} D^2 (D + E) B_e \tag{1}$$

$$E_{ch} = A J_{cu} = k_r E J_{cu}^2 \tag{2}$$

$$p = \frac{\pi D}{\Delta_p} \tag{3}$$

$$K_f = 1.5 p \beta \frac{e + E}{D} \tag{4}$$

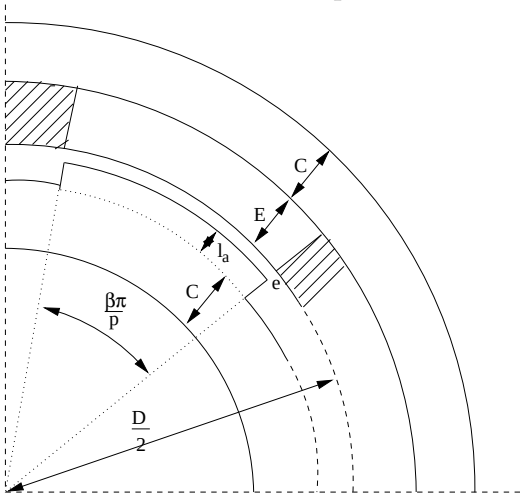
$$C = \frac{\pi \beta B_e}{4 p B_{iron}} D \tag{5}$$

$$B_e = \frac{2 l_a P}{D \ln\left(\frac{D+2E}{D-2(l_a+e)}\right)} \tag{6}$$

where Γ_{em} is the *electromagnetic torque* (from an energetic calculation, see Eq. (1)); $D(m)$ is the *bore diameter*, λ the *diameter over length ratio*, $E(m)$ the *winding thickness*, β the *polar arc factor*, k_r a *coefficient of occupation*; the global heating up of the winding is rather roughly modeled by E_{ch} (function of current electric loading A and J_{cu}, see Eq. (2)); $J_{cu}(A/m^2)$ is the *current areal density*; p is the *number of pole pairs*, Δ_p the *polar step* (p is linked to Δ_p by Eq. (3)); K_f is a *semi-empiric magnetic leakage coefficient* (established by numerical simulations, see Eq. (4)); $e(m)$ is the *thickness of the mechanical air-gap*; $C(m)$ is the *thickness of yoke* (Eq. (5) is obtained by neglecting interpolar leakages and armature reaction flux); B_{iron} is the *magnetic field in the iron*; $B_e(T)$ is the *no-load magnetic radial flux density* (see Eq. (6)); $l_a(m)$ is the *thickness of the permanent magnets*, P the *magnetic polarization*. For this study, we fix $\Gamma_{em} = 10N.m$, $P = 0.9T$, $k_r = 0.7$, $B_{iron} = 1.5T$, $E_{ch} = 10^{11}A/m$ and the polar step $\Delta_p = 0.1m$ as in [14]. The other parameters can vary inside the following intervals: $D(m) \in [0.01, 0.5]$, $\lambda \in [1, 2.5]$, $l_a(m) \in [0.003, 0.05]$, $E(m) \in [0.001, 0.05]$, $C(m) \in [0.001, 0.05]$, $\beta \in [0.8, 1]$, $B_e(T) \in$

$[0.1, 1]$, $J_{cu}(A/m^2) \in [10^5, 10^7]$, $K_f \in [0.01, 0.3]$, $e(m) \in [0.001, 0.005]$ and $p \in \{1, \cdots, 10\}$.
The motor structure is presented in Figure 1.

Figure 1: Structure of the considered permanent magnet machine



The addressed design problem can be formulated as an optimization problem by minimizing or maximizing a given criterion under constraints given by equations (1)-(6).

Note that, even though we focus on a simple specific design problem for the sake of illustration, the same analysis may be generalized to more complex examples formulated as nonlinear nonconvex constrained problems.

The rest of this paper is organized as follows. In Section 2, we propose six equivalent mathematical formulations of the optimal design problem of a slotless electrical rotating permanent magnet machine. In Section 3, we first computationally compare the six proposed formulations in the continuous case, i.e., with a fixed value for p and we discuss about their efficiency when the optimization problem is solved using the solver `fmincon` of *MatLab*. We show that numerical performances are different depending on the formulation. The the effect of changing starting points on the optimization results is also discussed. We then summarize some results when p is free on three mixed-integer formulations. Some concluding remarks are given in Section 4.

2 Mathematical formulations

In this paper, the addressed optimization problem has the following general form:

$$\mathcal{P}: \begin{cases} \min\limits_{\substack{x\in\mathbb{R}^n \\ y\in\mathbb{R}^m}} & f(x,y) \\ s.t. & \\ & g_i(x,y) \leq 0, \forall i \in \{1,\cdots,p\}, \\ & h_i(x,y) = 0, \forall i \in \{1,\cdots,q\}, \\ & y_i = A_i(x, y_{J_i}), \forall i \in \{1,\cdots,m\}, \\ & \underline{x_i} \leq x_i \leq \overline{x_i}, \forall i \in \{1,\cdots,n\}, \\ & \underline{y_i} \leq y_i \leq \overline{y_i}, \forall i \in \{1,\cdots,m\}. \end{cases}$$

where $J_i \subseteq \{1,\cdots,m\} \setminus \{i\}$. Moreover, y_i depends explicitly or implicitly on x by recursive calls to A_j functions and there is no cycle in the definition of y_i. Hence, $y_i = A_i(x, y_{J_i}) = A_i^R(x)$; as a vectorial notation, we use: $y = A^R(x)$.

By replacing the occurrences of y_i in $(\mathcal{P})$ by $A_i^R(x)$, we obtain the following reformulation:

$$\mathcal{R}: \begin{cases} \min\limits_{x\in\mathbb{R}^n} & f(x, A^R(x)) \\ s.t. & \\ & g_i(x, A^R(x)) \leq 0, \forall i \in \{1,\cdots,p\}, \\ & h_i(x, A^R(x)) = 0, \forall i \in \{1,\cdots,q\}, \\ & \underline{x_i} \leq x_i \leq \overline{x_i}, \forall i \in \{1,\cdots,n\}, \\ & \underline{y_i} \leq A_i^R(x) \leq \overline{y_i}, \forall i \in \{1,\cdots,m\}. \end{cases}$$

This reformulated problem $(\mathcal{R})$ of $(\mathcal{P})$ is obtained by removing m variables (all the variables y) and changing m equality constraints ($y_i = A_i(x, y_{J_i})$) to $2m$ inequality ones ($\underline{y_i} - A_i^R(x) \leq 0$ and $A_i^R(x) - \overline{y_i} \leq 0$).

The two optimization problems $(\mathcal{P})$ and $(\mathcal{R})$ are *mathematically equivalent* because they provide the same solution which corresponds to the global minimum of the two problems: all realizable solutions of $(\mathcal{P})$ are realizable solutions of $(\mathcal{R})$ and reciprocally, moreover the objective function values are equal (because $y = A^R(x)$).

From equations of the design problem such as those defined in (1)-(6), and by introducing the minimization of the volume of the magnets: $V_m = \pi\beta l_a \frac{D}{\lambda}(D-2e-l_a)$, we can generate some distinct but equivalent formulations of our design problem, as described above by problem $(\mathcal{R})$. The differences concern some variables which are replaced by functions and equality constraints by inequality ones. This yields six formulations.

The first formulation comes directly from the equations (1)-(6) of the design problem. Hence, we obtain:

$$F1: \begin{cases} \min\limits_{\substack{(D,\lambda,\cdots)\in\mathcal{D}\subset\mathbb{R}^{10} \\ p\in\{1,\cdots,10\}}} V_m(D,\lambda,...) = \pi\beta l_a \dfrac{D}{\lambda}(D-2e-l_a) \\ \text{s.t.} \\ \quad \text{constraints defined by Equations (1)-(6)} \end{cases}$$

Formulation F1 is a nonlinear, nonconvex optimization problem, the nonlinearities arising in the analytical formulæ expressing the objective function and the constraints. The variables $D(m)$, λ, $l_a(m)$, $E(m)$, $C(m)$, β, $B_e(T)$, J_{cu}, K_f and $e(m)$ are continuous, while p is integer. Hence, the problem is a mixed-integer nonlinear program (MINLP).

We remark that the problem is badly scaled, because some parameters such as J_{cu} have large values and the others, such as $D, e, \ldots$, are definitely small. However, this seems to have no impact when a solver like `fmincon` is used on all the formulations considered in this paper. We also note that the integer variable p will be fixed to a constant in the first part of Section 3 to provide a continuous nonlinear problem (NLP) and p will be an integer variable in subsection 3.4.

The following formulations are reformulations of F1 which are mathematically equivalent.

First, we consider B_e as an auxiliary function (depending on D, l_a, E, e) which returns $\frac{2l_aP}{D\ln\left(\frac{D+2E}{D-2(l_a+e)}\right)}$ and whose value is bounded by 0.1 and 1; in fact, in all the expressions of Formulation F1, B_e could be directly replaced by $\frac{2l_aP}{D\ln\left(\frac{D+2E}{D-2(l_a+e)}\right)}$. Thus, from F1 we remove one variable and its corresponding equality constraint, and we replace this constraint with two inequality constraints which impose lower and upper bounds on the value of B_e. We obtain the following formulation:

$$F2: \begin{cases} \min\limits_{\substack{(D,\lambda,\cdots)\in\mathcal{D}\subset\mathbb{R}^{9} \\ p\in\{1,\cdots,10\}}} V_m(D,\lambda,...) = \pi\beta l_a \dfrac{D}{\lambda}(D-2e-l_a) \\ s.t. \quad \text{constraints defined by Equations (1)-(5)} \\ \qquad 0.1 \le B_e(.) \le 1 \end{cases}$$

where $B_e(.)$ is an auxiliary function and returns $\frac{2l_aP}{D\ln\left(\frac{D+2E}{D-2(l_a+e)}\right)}$ (or where $B_e(.)$ is directly replaced in Formulation F2 by its corresponding expression).

The two formulations only differ in size (number of variables and constraints). The first one, F1, has 10 continuous variables: $D, \lambda, l_a, E, C, \beta, J_{cu}, K_f, e, B_e$

and 6 equality constraints, while the second one, F2, has 9 continuous variables: $D, \lambda, l_a, E, C, \beta, J_{cu}, K_f, e$, 5 equality constraints and 2 inequality constraints involving B_e. Additional constraints are given in both cases by the bounds on the variables. We remark that any solution (global minimum) of F1 is also a solution of F2 and conversely.

We now proceed to eliminate another variable and considering it as a function. Again, the equality constraint corresponding to the selected variable is replaced by two inequality constraints which impose lower and upper bounds on its value. Considering C as a function which returns $\frac{\pi\beta B_e}{4pB_{iron}}D$, we have:

$$F3: \begin{cases} \min\limits_{\substack{(D,\lambda,\cdots)\in\mathcal{D}\subset\mathbb{R}^8 \\ p\in\{1,\cdots,10\}}} V_m(D,\lambda,...) = \pi\beta l_a \dfrac{D}{\lambda}(D-2e-l_a) \\ s.t. \quad \text{constraints defined by Equations (1)-(4)} \\ \qquad 0.1 \leq B_e(.) \leq 1 \\ \qquad 0.001 \leq C(.) \leq 0.05 \end{cases}$$

Formulation F3 has 8 continuous variables, 4 equality constraints, 4 inequality constraints and bounds on variables.

Considering now K_f as a function which returns $1.5p\beta\frac{e+E}{D}$, we obtain the following formulation:

$$F4: \begin{cases} \min\limits_{\substack{(D,\lambda,\cdots)\in\mathcal{D}\subset\mathbb{R}^7 \\ p\in\{1,\cdots,10\}}} V_m(D,\lambda,...) = \pi\beta l_a \dfrac{D}{\lambda}(D-2e-l_a) \\ s.t. \quad \text{constraints defined by Equations (1)-(3)} \\ \qquad 0.1 \leq B_e(.) \leq 1 \\ \qquad 0.001 \leq C(.) \leq 0.05 \\ \qquad 0.01 \leq K_f(.) \leq 0.3 \end{cases}$$

Formulation F4 has 7 continuous variables, 3 equality constraints, 6 inequality constraints and bounds on variables.

Another possibility to reformulate the problem is to introduce a new variable y which replace a nonlinear term appearing in the first equality constraint, and add the corresponding constraint to the formulation. Let y

be equal to $\sqrt{\beta E}$. We can reformulate the problem as follows:

$$F5: \begin{cases} \min\limits_{\substack{(D,\lambda,\cdots)\in\mathcal{D}\subset\mathbb{R}^8 \\ p\in\{1,\cdots,10\}}} V_m(D,\lambda,...) = \pi\beta l_a \dfrac{D}{\lambda}(D-2e-l_a) \\ s.t. \quad \Gamma_{em} = \frac{\pi}{2\lambda}(1-K_f(.))\sqrt{k_r E_{ch}} y D^2 \times \\ \qquad\qquad\qquad\qquad (D+E)B_e(.) \\ \qquad y^2 = \beta E \\ \qquad \text{constraints defined by Equations (2),(3)} \\ \qquad 0.1 \leq B_e(.) \leq 1 \\ \qquad 0.001 \leq C(.) \leq 0.05 \\ \qquad 0.01 \leq K_f(.) \leq 0.3 \end{cases}$$

Note that now the square root term appearing in the first constraint is a constant term. Note also that y has a positive value because of the bounds on variables β and E. Formulation F5 has 8 continuous variables, 4 equality constraints, 6 inequality constraints and bounds on variables.

Finally, we consider again $y = \sqrt{\beta E}$ and we add this variable to formulation F1, obtaining a new formulation with 11 continuous variables and 7 equality constraints:

$$F6: \begin{cases} \min\limits_{\substack{(D,\lambda,\cdots)\in\mathcal{D}\subset\mathbb{R}^{11} \\ p\in\{1,\cdots,10\}}} V_m(D,\lambda,...) = \pi\beta l_a \dfrac{D}{\lambda}(D-2e-l_a) \\ s.t. \quad \Gamma_{em} = \frac{\pi}{2\lambda}(1-K_f)\sqrt{k_r E_{ch}} y D^2 (D+E) B_e \\ \qquad E_{ch} = AJ_{cu} = k_r E J_{cu}^2 \\ \qquad y^2 = \beta E \\ \qquad \text{constraints defined by Equations (2)-(6)} \end{cases}$$

These six formulations vary in size from 7 to 11 continuous variables, from 3 to 7 equality constraints and from 0 to 6 inequality constraints.

Comparing the six formulations above, we note that when the number of variables increases, the nonlinearities of the equations decrease yielding simpler optimization problems but with more variables. Since the solver performance is roughly directly proportional to both the number of variables and the number of nonlinearities in the objective and constraints, a natural trade-off situation arises. Note that this behavior is independent of the specific considered problem and the same considerations apply to other possibly more general design problems having nonlinear optimization formulations like $(\mathcal{P})$.

In the following section, we numerically compare the proposed formulations and discuss about their efficiency when a local optimization solver is used to solve the optimal design problem. We also consider two other

objective functions for the addressed problem:

$$V_u = \pi \frac{D}{\lambda}(D + E - e - l_a)(2C + l_a + e + l_a) \quad (7)$$

$$P_j = \pi \rho_{cu} \frac{D}{\lambda}(D + E) E_{ch} \quad (8)$$

where V_u represents the volume of the active parts and P_j the losses by joule effects with $\rho_{cu} = 0.018 \times 10^{-6}$.

3 Computational comparison between formulations

Different mathematical formulations of a given problem may share the same properties, such as for example feasible region and optima. However, some of them are easier to solve than others. The same optimization solver can perform differently depending on the formulation. For this reason, a considerable amount of work is often devoted to investigate efficient *reformulations*, see e.g. [7].

In this section, we computationally compare formulations F1 to F6 and evaluate their impact on the performance of the deterministic local optimization solver `fmincon` of *MatLab v7*. In a first series of experiments, the integer variable p is fixed to the constant value 5 (known to be optimal), so that formulations F1 to F6 become continuous optimization problems. We employ *MatLab*'s `optimset` function to fix the following parameters for `fmincon`: (i) the maximum number of iterations and of function evaluations is fixed to 30000, (ii) the tolerance on the value of the function is fixed to 10^{-10}, (iii) the tolerance on the solution point is fixed to 10^{-6}, (iv) the tolerance on the constraints satisfaction is fixed to 10^{-7} and (v) `FunValCheck` is fixed to '`on`', which generates an error if values become complex during the computation (this can occur in equations (1)-(6) when computing the logarithm of a negative number). In this last case (v), we have to use *MatLab*'s exception handling mechanism (the `try` and `catch` functions). The considered tolerances correspond to standard settings in optimization solvers.

The performance of a local method such as `fmincon` often depends on the choice of a starting point (values initially assigned to the decision variables). This may be given by the existing design configuration in case one exists, otherwise it is very difficult to choose it appropriately. Depending on this choice, we have three possibilities: (i) the algorithm converges to a local solution; (ii) the algorithm progresses slowly towards a local optimum, but finally exceeds the maximum allowed number of iterations or function

evaluations; (iii) the algorithm fails for a number of other reasons, with no meaningful answer. An aprioristic choice of starting point guaranteeing the occurrence of case (i) is usually impossible. We try and make our results "starting point independent" by implementing a simple multistart approach from 1000 randomly generated starting points (random values are generated using *MatLab*'s `rand` function). We then record the percentage of starting points yielding a local and a global optimum.

To generate starting points, two strategies are possible. In the first one, random values are generated for each variable of Formulation F6 (corresponding to the formulation having the maximum number of variables) and the same values are used for all Fi, $i = 1, \ldots, 5$. In the second one, a random value is generated for the 7 continuous variables of Formulation F4 (corresponding to the formulation having the minimum number of variables) and corresponding values are computed for problem entities which are treated as functions (in formulations F1 to F3, F5, F6). For example, in Formulation F2 the value of C and K_f can be computed using auxiliary functions defined for Formulation F4 on the basis of the values of D, λ ... which are randomly generated.

3.1 Numerical comparisons using V_m

In this subsection, we compare the results of our Multistart method on the six optimization problems F1 to F6 defined in Section 2. The value of p is fixed to 5.

Results are shown in Tables 1 and 2 depending on whether the starting points are generated from Formulation F6 or from Formulation F4 respectively (the same base of the 1000 starting points is used for all the formulations). We compare, for each formulation, the percentage of local minima and the percentage of best local minima found, the best and the worst values of local minimum found, the best CPU time, the average CPU time, the average CPU time corresponding to successful running (i.e., providing a local minimum), and the worst CPU time. It appears that the percentage of failure is quite high. Indeed, for all the considered formulations, the percentage of local minima found is always lower than 50% and slightly higher than this percentage in one case only. On the last line of the tables ($x0 = mid$), we report the local minima which are obtained (or not) using the middle point of the variable bounds taken as a starting point; '—' denotes no local minima. In Table 1, only two formulations provide a local minimum using the middle of the bounds as a starting point compared to Table 2 where four formulations achieve the convergence. Nevertheless, all these local minima are far from the best one (one order of magnitude of difference). This confirms that

a lot of care must be paid in the choice of a good starting point. When some values are computed starting from values assigned to variables on which they depend (Table 2), the percentage of local minima found is higher. In particular, the formulations with more variables, F1 and F6, provide the highest percentage of local minima found. This remark yields to show that the formulations with more variables F1 and F6, where the nonlinearity decrease, associated with starting points made from components which are partially randomly generated and partially computed from other components using auxilary functions, provide the most efficient Multistart technique: about 50% of chance to provide a local minimum compared to less than 45% for all the 10 others (see Tables 1 and 2), and about 5% to find the best local minima compared to less than 4.6% for the other formulations excepted F5; note that a minimum is considered equal to the best one if their corresponding optimal value are numerically close.

CPU times are in general practically acceptable (lower than 30 seconds). The average CPU time is however increased by the time spent in unsuccessful runs, where convergence is not achieved because of a worse choice of the starting point. We note that there is no impact on the CPU-time if the formulation has more variables than another one.

Table 1: Numerical results by minimizing V_m in a Multistart method: random case

	F1	F2	F3	F4	F5	F6
% local min	40.4%	44.2%	43.7%	44.6%	44.0%	41.6%
% best min	3.7%	3.3%	2.8%	4.6%	4.8%	3.4%
best min value	7.3589e-5	7.3586e-5	7.3586e-5	7.3568e-5	7.3597e-5	7.3568e-5
worst local min	1.1443e-3	1.1215e-3	1.1220e-3	1.0526e-3	1.0976e-3	1.1065e-3
best CPU-time	0.02s	0.00s	0.00s	0.00s	0.00s	0.00s
avrg CPU-time	0.50s	0.24s	0.27s	0.18s	0.17s	0.42s
avrg-scf CPU-time	0.12s	0.14s	0.13s	0.14s	0.14s	0.13s
worst CPU-time	20.88s	27.61s	33.25s	23.02s	30.86s	23.94s
$x0 = mid$	—	8.4162e-4	8.4163e-4	—	—	—

In Tables 3 and 4 we provide the best found solutions for each formulation. We remark that Formulation F4 produces the minimal value for the objective function V_m; this is understandable because F4 is the most compact formulation. We note also that all computed objective function values are different except for F2 and F3 in Table 3, this minimum point was also

Table 2: Numerical results by minimizing V_m in a Multistart method: partially random case

	F1	F2	F3	F4	F5	F6
% local min	49.7%	45.2%	44.7%	44.6%	45.5%	50.2%
% best min	4.8%	2.9%	3.1%	4.6%	5.0%	5.5%
best min value	7.3586e-5	7.3586e-5	7.3598e-5	7.3568e-5	7.3606e-5	7.3586e-5
worst local min	1.1512e-3	1.0382e-3	1.0327e-3	1.0526e-3	1.0077e-3	1.0261e-3
best CPU-time	0.02s	0.00s	0.02s	0.02s	0.02s	0.02s
avrg CPU-time	0.75s	0.17s	0.23s	0.18s	0.16s	0.61s
avrg-scf CPU-time	0.12s	0.13s	0.13s	0.14s	0.14s	0.13s
worst CPU-time	20.00s	26.66s	30.91s	23.09s	30.84s	12.98s
$x0 = mid$	8.0291e-4	7.7114e-4	7.7112e-4	—	—	8.7127e-4

found using formulations F1, F2 and F6 in Table 4. Thus, as we found 8 different minima, we can conclude that there exist a lot of solutions which provide close values for the objective function V_m. This kind of Multistart method could be adapted to provide a large set of minimal points.

Note that Formulation F4 provides the same results in the two tables 3 and 4 because we take exactly the same starting point in both cases, as it is the most compact formulation, so there are no components of the starting point that can be deduced from others.

In [14], the considered design problem with formulation F1 was first modeled and solved using a local search algorithm. These first solutions were clearly local minima because in [17] new better solutions were found. In [11], a deterministic global optimization solver was used for the first time (with a restriction to 4 digits on the accuracy of parameters). The objective function values in Table 4 are the same as the one corresponding to the global solution provided in [11]. Thus, some global minimizers are found here using a multistart technique with the fmincon procedure. The same remarks could be done for the results involving the objective functions V_a and P_j presented in next sections.

3.2 More numerical comparisons using V_u and P_j

In order to summarize all the computations that we performed, we present in the following and until the end of the paper, only the two most important formulations: F1, which comes directly from equations (1)-(6), and F4, which

Table 3: Comparison between local minima: completely random case

Best local min	F1	F2	F3	F4	F5	F6
$D^* =$	0.159155	0.159155	0.159155	0.159155	0.159155	0.159155
$\lambda^* =$	2.5	2.5	2.5	2.5	2.5	2.5
$l_a^* =$	0.003	0.003	0.003	0.003	0.003	0.003
$E^* =$	0.002549	0.003419	0.003419	0.002897	0.003542	0.002997
$C^* =$	0.005100	0.004556	0.004556	0.004853	0.004496	0.004790
$\beta^* =$	0.8	0.8	0.8	0.8	0.8	0.8
$B_e^* =$	0.382466	0.341724	0.341724	0.363953	0.337209	0.359256
$J_{cu}^* =$	7485729.14	6464050.67	6464050.68	7021833.60	6350604.11	6904057.59
$K_f^* =$	0.149734	0.182640	0.182640	0.163657	0.186849	0.167446
$e^* =$	0.001422	0.001426	0.001426	0.001444	0.001414	0.001445
V_m^*	7.3589e-5	7.3586e-5	7.3586e-5	7.3568e-5	7.3597e-5	7.3568e-5

Table 4: Comparison between local minima: partially random case

Best local min	F1	F2	F3	F4	F5	F6
$D^* =$	0.159155	0.159155	0.159155	0.159155	0.159155	0.159155
$\lambda^* =$	2.5	2.5	2.5	2.5	2.5	2.5
$l_a^* =$	0.003	0.003	0.003	0.003	0.003	0.003
$E^* =$	0.003419	0.003419	0.003549	0.002897	0.003633	0.003419
$C^* =$	0.004556	0.004556	0.004493	0.004853	0.004454	0.004556
$\beta^* =$	0.8	0.8	0.8	0.8	0.8	0.8
$B_e^* =$	0.341724	0.341724	0.336951	0.363953	0.334045	0.341724
$J_{cu}^* =$	6464050.65	6464050.67	6344113.56	7021833.60	6271081.34	6464050.68
$K_f^* =$	0.182640	0.182640	0.187094	0.163657	0.189877	0.182640
$e^* =$	0.001426	0.001426	0.001413	0.001444	0.001404	0.001426
V_m^*	7.3586e-5	7.3586e-5	7.3598e-5	7.3568e-5	7.3606e-5	7.3586e-5

is the most compact one. Moreover, two distinct techniques are applied to F1 to generate its starting point: F1(rnd) indicates that all the components of the starting point are randomly generate and F1(cmp) indicates that the starting point is partially randomly generated and some components are computed from others. In the following tables, the best and worst values of local minima and the best and worst CPU-times used for one local minimization step of the Multistart are reported as in the previous subsection.

In this subsection, two different objective function are minimized: V_u and P_j defined in (7) and (8).

In Table 5, results using the Multistart method with the objective function V_u (7) are given. Similarly, in Table 6, results for the objective function P_j (8) are presented. The best found local minima are respectively given in Tables 7 for V_u and 8 for P_j.

We observe that results are similar to the ones of the previous subsection. This seems to confirm that the Formulation F1 associated with partially randomized starting points (denoted by F1(cmp)) provides the best results in terms of percentage to converge to a local solution, while the percentage to find the best local minimum are still low for V_m but quite high (about 20%) for P_j. In these two cases too, the minimum points are all different.

These remarks show that the objective function does not have a deep impact in the considered optimal design problems.

Table 5: Numerical results by minimizing V_u in a Multistart method

	F1(rnd)	F1(cmp)	F4
% local min	41.0%	49.8%	45.1%
% best min	2.4%	2.2%	2.4%
best min value	5.3632e-4	5.3631e-4	5.3635e-4
worst local min	1.7000e-3	1.6494e-3	1.7543e-3
best CPU-time	0.02s	0.00s	0.02s
avrg CPU-time	0.44s	0.73s	0.13s
avrg-scf CPU-time	0.13s	0.12s	0.20s
worst CPU-time	19.00s	12.03s	24.09s
$x0 = mid$	—	1.5392e-3	—

Note that only Formulation F1(cmp) provides a local minimum when the middle of the bounds on the variables is used as a starting point (and some components are computed from the others). Note also that these two local minima are not the best minima found (in Table 5, this local minimum is close to the worst case).

Table 6: Numerical results by minimizing P_j in a Multistart method

	F1(rnd)	F1(cmp)	F4
% local min	44.5%	50.2%	46.1%
% best min	19.7%	19.6%	21.6%
best min value	5.7810e+1	5.7810e+1	5.7810e+1
worst local min	8.5743e+1	7.0680e+1	6.9911e+1
best CPU-time	0.02s	0.00s	0.02s
avrg CPU-time	0.28s	0.55s	0.09s
avrg-scf CPU-time	0.10s	0.09s	0.12s
worst CPU-time	12.41s	20.45s	12.42s
$x0 = mid$	—	58.057	—

Table 7: Comparison between local minima using V_u

Best local min	F1(rnd)	F1(cmp)	F4
$D^* =$	0.159155	0.159155	0.159155
$\lambda^* =$	2.5	2.5	2.5
$l_a^* =$	0.003	0.003	0.003
$E^* =$	0.003373	0.003419	0.003419
$C^* =$	0.004580	0.004556	0.004556
$\beta^* =$	0.8	0.8	0.8
$B_e^* =$	0.343463	0.341724	0.341724
$J_{cu}^* =$	6507727.58	6464050.67	6464050.67
$K_f^* =$	0.181054	0.182640	0.182640
$e^* =$	0.001429	0.001426	0.001426
V_u^*	5.3632e-4	5.3631e-4	5.3635e-4

Table 8: Comparison between local minima using P_j

Best local min	F1(rnd)	F1(cmp)	F4
$D^* =$	0.159155	0.159155	0.159155
$\lambda^* =$	2.5	2.5	2.5
$l_a^* =$	0.047767	0.042697	0.040992
$E^* =$	0.001429	0.001429	0.001429
$C^* =$	0.007328	0.007250	0.007870
$\beta^* =$	0.845289	0.803578	0.870069
$B_e^* =$	0.520176	0.541294	0.542699
$J_{cu}^* =$	10000000	10000000	10000000
$K_f^* =$	0.181863	0.193636	0.227066
$e^* =$	0.003137	0.003685	0.004109
P_j^*	57.8101	57.8101	57.8101

3.3 Comparison between the six formulations and the three objective functions

Figures 2, 3, 4 and 5 summarize for the six proposed formulations and the three objective functions the results obtained in terms of percentage of convergence to a local minimum (Figures 2 and 3) and of percentage of times when the best local minimum is found (Figures 4 and 5), using totally or partially randomized starting points.

From Figure 2, we can see that Formulation F4 (which is the most compact one) provides the best results for V_m and V_u, and Formulation F3 provides the best results for P_j. Comparing the six formulations on each of the considered objective functions, it appears that close results were obtained for formulations F2, F3, F4 and F5, while formulations F1 and F6 provide a percentage which is about 3% lower. Contrarily on Figure 3 (when some components of the starting points are computed from others), F1 and F6 appear clearly to be the most efficient formulations for our Multistart method for all the three objective functions, providing a percentage about 5% higher than all other formulations.

Concerning the percentage of best found local minima, from Figures 4 and 5 we can see that similar results are obtained for all the formulations and all the objective functions.

Figure 2: Percentages of found local minima: totally randomized case

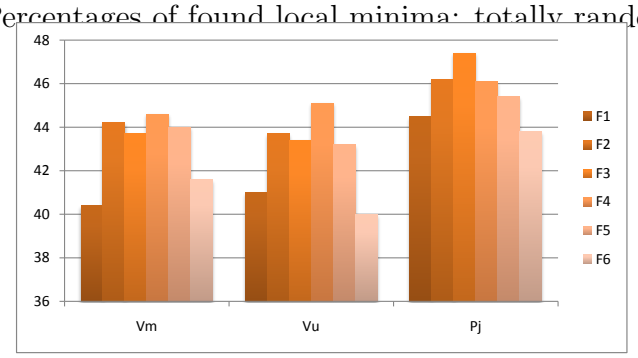


Figure 3: Percentages of found local minima: partially randomized case

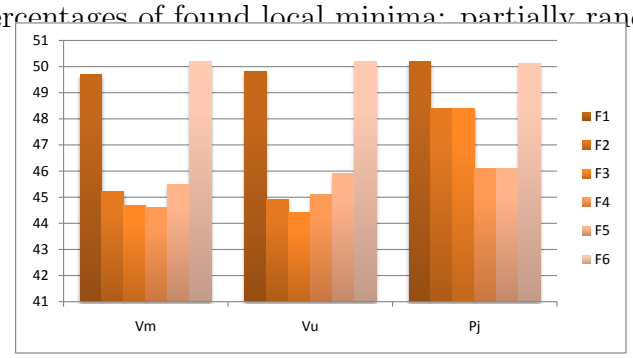


Figure 4: Percentages of best found minima: totally randomized case

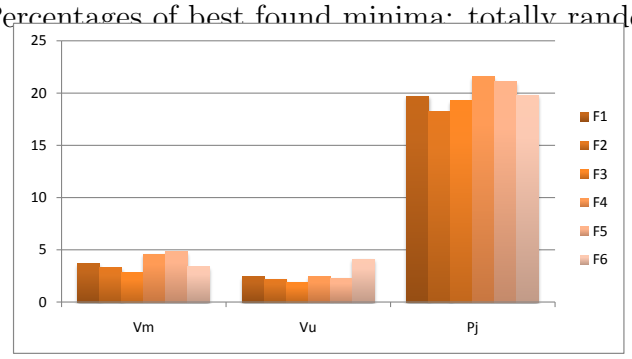
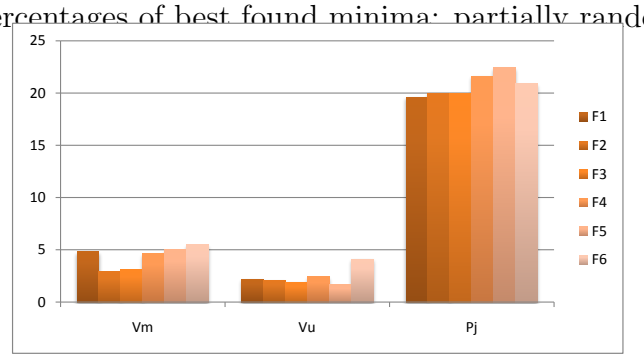


Figure 5: Percentages of best found minima: partially randomized case



3.4 Computational comparison with p free

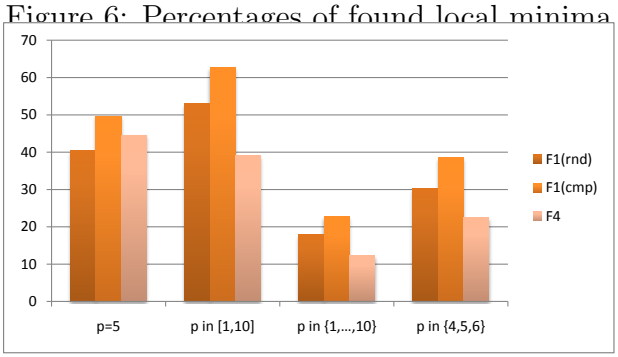
In the considered optimal design problem, the number of pole pairs p has an integer value and can be considered as a variable of the problem. In fact, algorithms such as the one implemented in `fmincon` of *MatLab*, need continuous formulations and twice differentiable functions. In some papers of the literature (see [14, 17]), p is considered as a continuous variable and the user has to convert the obtained real value of p into an integer one. Previously, we considered p as a fixed parameter ($p = 5$).

Some ways to deal with the integer variable p could be the following: (i) p is converted into a real variable $p \in [1, 10]$ and its optimal value is rounded into an integer one by the user (all the other variables must be adjusted); (ii) p is treated as a real variable but an equality constraint $(p-1) \times (p-2) \times \cdots \times (p-10) = 0$ is added; (iii) we first solve case (i) and following the optimal value of p (denoted p^*) we add the constraint $(p - ([p^*] - 1)) \times (p - [p^*]) \times (p - ([p^*] + 1)) = 0$, where $[p^*]$ indicates that the closest integer number of p^* is taken into account.

In this study, we focus on two formulations only, the first one where all entities are considered as variables of the problem F1 and the one where there is the minimum number of variables F4. Results with F1 are again obtained using random generated values for all variables as well as random values for the 7 variables of F4 and values computed from this 7 values for the other variables (as we did in subsection 3.2). Since, as noticed in the previous section, no relevant differences in the behavior of the considered formulations can be observed using a different objective function like V_u or P_j, we focus in the following only on the minimization of the volume V_m of the magnet.

In Figures 6 and 7, we compare the formulations in terms of percentage of found local minima and percentage of best found local minima respectively, considering the different choices of p discussed above. Thus, these figures summarize the results obtained via the multistart method associated with *fmincon*.

Figure 6: Percentages of found local minima

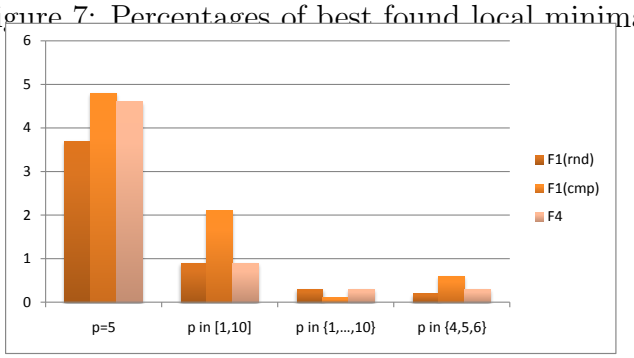


From Figure 6, one can see that Formulation F1 with recomputed values for the starting point appears to be the most efficient to find the largest number of local solutions, with the different strategies to deal with p. Thus, as the values of the percentage of best found local solution are low, we can have more confidence on the local solutions found by this formulation in order to provide the global solution.

It is also evident, from Figure 6 and specially from Figure 7, that considering p as a variable, as expected the problem becomes more difficult to solve and consequently the percentages of solutions found and best found solution are low. The fourth proposed strategy, based on a lower-degree polynomial constraint, allows to obtain higher percentages, though very low, when p is treated as an integer variable.

In Figure 7, it is very clear that, using Formulation F1 with some computed values for the starting point, a higher number of starting points allows to obtain the best found local solution. Thus, as the values of the percentage of best found local solution are low (except for $p \in \{1, \cdots, 10\}$ but the minimal value for F1(cmp) is the lowest one), we can have more confidence on the local solutions found by this formulation in order to provide the global solution.

Figure 7: Percentages of best found local minima



4 Conclusion

We have presented in this paper six mathematical formulations of an optimal design problem. We have shown that, even though the formulations are mathematically equivalent, their numerical performances are different when an optimization solver is used. We mainly discussed the impact of reformulations on a classical local solver based on Active-Set/SQP/Quasi-Newton algorithm, thus showing that the designer must take care about the formulation of the problem in order to make more efficient the use of this kind of local algorithms, such as the one proposed by *MatLab* in `fmincon`.

First, we have discussed the solutions found using the six formulations, considering the problem as a continuous one (fixing the value of the integer variable present in the formulation), in a multistart setting where the solution is computed at each step by `fmincon`. The first formulation F1, with 10 continuous variables, provides the best percentage of local solutions found, specially when the starting point is computed by giving random values to some of its components and computing the other values from these ones. Second, we have discussed about strategies to deal with integer variables, such as the number of pole pairs. Again Formulation F1 (with starting points partially generated) appears to be the most efficient in providing the best local solution.

These remarks could be extended to more general design problems when a local solver is used: formulations of type $(\mathcal{P})$, with starting points only depending on variables x, seem to be more easier to solve and provide better results than their possible reformulations of type $(\mathcal{R})$.

References

[1] E. Fitan, F. Messine, and B. Nogarede. A general analytical model of electrical permanent magnet machine dedicated to optimal design. *COMPEL - International Journal for Computation and Mathematics in Electrical and Electronic Engeneering*, 22(4), 2003.

[2] E. Fitan, F. Messine, and B. Nogarede. The electromagnetical actuators design problem: a general and rational approach. *IEEE Transactions on Magnetics*, 40(3):1579–1590, 2004.

[3] P.E. Gill, W. Murray, M.A. Saunders, , and M.H. Wright. Procedures for optimization problems with a mixture of bounds and general linear constraints. *ACM Transactions on Mathematical Software*, 10:282–298, 1984.

[4] P.E. Gill, W. Murray, and M.H. Wright. *Numerical Linear Algebra and Optimization*. Addison Wesley, 1991.

[5] S. Huang, M. Aydin, and T.A. Lipo. A direct approach to electrical machine performance evaluation: Torque density assessment and sizing optimisation. In *International Conference on Electrical Machines*, volume Art. 235, 2002.

[6] L. Liberti. Reformulations in mathematical programming: Definitions and systematics. *RAIRO-Operations Research*, 43(1):55–86, 2009.

[7] L. Liberti, S. Cafieri, and F. Tarissan. *Reformulations in Mathematical Programming: a Computational Approach*, volume 203 of *Studies in Computational Intelligence*, pages 153–234. Springer, Berlin, 2009.

[8] Mathworks. fmincon documentation.

[9] F. Messine. Deterministic global optimization using interval constraint propagation techniques. *RAIRO-Operations Research*, 38(4):277–294, 2004.

[10] F. Messine. *A Deterministic Global Optimization Algorithm for Design Problems*, pages 267–294. Kluwer, 2005.

[11] F. Messine, B. Nogarede, and J.L. Lagouanelle. Optimal design of electromechanical actuators: a new method based on global optimization. *IEEE Transactions on Magnetics*, 34(1):299–307, 1998.

[12] T.J.E. Miller. *Brushless Permanent-Magnet and Reluctance Motor Drives*. Clarendon Press, Oxford, 1989.

[13] P. Neittaanmki, M. Rudnicki, and A. Savini. *Inverse Problems and Optimal Design in Electricity and Magnetism*. Clarendon Press, Oxford, 1996.

[14] B. Nogarede, A.D. Kone, and M. Lajoie-Mazenc. Optimal design of permanent-magnet machines using analytical field modeling. *Electromotion*, 2(1):25–34, 1995.

[15] L. Rao, W. Yan, and R. He. Mean field annealing (mfa) and optimal design of electromagnetic devices. *IEEE Transactions on Magnetics*, 32:1218–1221, 1996.

[16] G.R. Slemon and X. Liu. Modeling and design optimization of permanent magnet motors. *Electric Machines and Power Systems*, pages 71–92, 1992.

[17] F. Wurtz, J. Bigeon, and C. Poirson. A methodology and a tool for the computer-aided design with constraints of electrical devices. *IEEE Transactions on Magnetics*, 32:1429–1432, 1996.