Comparison of optimization algorithms for the design of a brushless DC electric machine with travel time minimization

T.D. Nguyen, V. Lanfranchi, C. Doc and J.P. Vilain
University of Technology of Compiègne / Electromechanical Laboratory of Compiègne (LEC)
Compiègne, France

Abstract—This paper presents DC brushless (DCBL) machine design methods using the three Matlab’s optimization algorithms. Instead of optimizing the machine’s design with respect to a classic criterion such as the minimization of losses, of weight or cost (usually at one operating point), our criterion will be the time it takes for the machine to drive a variable load to fulfill its trajectory. This will require the calculation of the machine’s behavior on the whole of its operating diagram (torque/speed diagram). A simplified analytical model of a DCBL machine is used in order to compare the performance of the three algorithms. The characteristics of the optimization results and the use of different algorithms in cascade will be discussed.

I. INTRODUCTION

In the literature, many articles have proposed different methods to optimize the design of electrical machines based on one operating point (torque/speed) ([1], [2], [3], [4]). In [4] an optimization method using the sequence of genetic and gradient-based algorithms has been proven to give the optimal solution with a small calculation time. This method is applied for the design of a DCBL motor where the criterion is the total losses at only one particular operating point.

One particularity of our problem is that the machine has to drive a variable load, which is in function of the rotor’s angular position (see Fig. 1). At first the load increases linearly, then it stays at the maximal position. The second particularity is our optimization criterion: the travel time. This “travel time” is defined as the time it takes for the machine’s load to fulfill its trajectory. This method is different from an optimization method at one operating point.

In this article, a simplified analytical model of a DCBL machine as well as a discretized travel time calculation method will be shortly presented, then the three optimization algorithms of Matlab (SQP, Direct Search and Genetic Algorithm) along with their advantages and drawbacks will be described. Finally the results obtained by different methods will be mentioned.

The DCBL machine in this article is a small one with the following fixed external dimensions: 70mm of length and 18mm of the outside radius.

II. ANALYTICAL MODEL OF A DCBL MACHINE

Synchronous surface permanent magnet machines are famous for their high compactness and their high massic torque. In this section, a linear DCBL machine’s analytical model (with a realistic flux density limit for the considered section) is presented. This model is validated and fine-tuned with help of FEM method (FLUX2D software).

A. Electromagnetic model

The main idea is to calculate the torque with torque constant from design parameters. In this analytical model, the torque constant $K$ is assumed to be equal to the EMF constant. An equivalent LRE electric circuit (Fig.2) is needed to calculate the behavior of the motor. The EMF is calculated by the following formula [5]:

$$ E = K \omega $$

(1)

where $\omega$ is the machine’s rotation speed.

The torque constant of a peripheral permanent magnet machine can be calculated by the following formula:

$$ K = 4\pi N B_p R_a L_{st} N_{enc} $$

(2)
where
- \( p \): number of pole pairs
- \( N \): number of conductors per slot
- \( N_{\text{enc}} \): number of slots per pole per phase
- \( B_g \): flux density in the air gap
- \( R_a \): average radius of the air gap
- \( L_{\text{st}} \): machine’s length

The flux density in the air gap \( B_g \) is calculated with help of the magnetic field circulation. It takes into account the geometric dimensions, the remanent flux density of permanent magnet and the saturated magnetic field.

\[
B_g = f(\text{geometric dimensions}, H_{\text{sat}}, B_{r})
\]  
(3)

In order to be able to simulate the motor’s behavior, we need to determine its inductance and resistance.

1. **Coil inductance calculation:**

   The net phase winding inductance in a machine is calculated by:

\[
L = L_g + L_s + L_e
\]  
(4)

where
- \( L_g \) is the air gap inductance:
  \[
  L_g = \frac{2\pi \mu_0 l_{\text{st}} R_a}{l_m + \mu_0 C_{\phi}} N^2
  \]  
(5)
- \( R_a \) is the rotor’s radius,
- \( l_a \) is the air gap’s width,
- \( l_m \) is the depth of permanent magnet and \( C_{\phi} \) is the relative magnet’s coverage.
- \( L_s \) is the slot leakage inductance

\[
L_s = N_m (2N)^2 \left[ \frac{\mu_0 d_s L_{\text{st}}}{3 \omega_{sb}} + \frac{\mu_0 d_{ls} L_{\text{st}}}{2 \omega_{sb}} + \frac{\mu_0 d_{sh} L_{\text{st}}}{\omega_{so}} \right]
\]  
(6)

where \( N_m \) is the number of slots.
- \( L_e \) is the end turn inductance

\[
L_e = \frac{N_m \mu_0 d_{s} \tau_{cp} N^2}{2} \ln \left( \frac{\tau_{cp} \sqrt{\pi}}{2A_{s}} \right)
\]  
(7)

where \( A_s \) is the cross sectional area of the air gap, the other variables are presented in Fig.3.

2. **Resistance calculation:**

   The total resistance of the motor is calculated from the total length of the conductor in the stator, this is a classic calculation and won’t be presented here.

B. **Mechanical calculation**

   Once the magnetic and the electric models have been made, the maximal torque can be calculated. From this maximal torque, the minimal radius of the drive shaft is calculated thanks to a material resistance formula, then the inertia of the whole rotor is calculated.

III. **OBJECTIVE FUNCTION CALCULATION**

In order to obtain the optimal solution for an optimization problem, one has to define an objective function along with its optimization variables and constraints if necessary. In this article, the objective function will calculate the travel time with three stages: acceleration, constant speed and acceleration. Because of the size of our machine, some dimensions have been fixed in order to satisfy dimensional requirements of the application. Table I presents some fixed and optimization variables. Among these variables, \( p \) and \( N_{\text{enc}} \) are discrete variables with the following constraints (due to the size of the machine):

- \( 1 \leq p \leq 3 \)
- \( 1 \leq N_{\text{enc}} \leq 2 \)

Unfortunately, Matlab’s optimization algorithms cannot handle directly problems with discrete values, thus the optimization procedure is the following:

- Initializing the value of these two variables (6 possible combinations thanks to the above constraints).
- Proceeding the optimization with three rest variables using different algorithms available in Matlab.

In this article, only results with the initialization \( p = 3 \) and \( N_{\text{enc}} = 1 \) are detailed.

One can also notice that the variable \( N \) (number of conductors per slot) is also an integer value. The difference of this variable and the two above discrete variables is that \( N \) can vary in a large range of values. That’s why we opted to do an optimization as if \( N \) were

<table>
<thead>
<tr>
<th>Table I</th>
<th>VARIABLE</th>
<th>DESCRIPTION</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_{\text{ext}} )</td>
<td>machine’s length</td>
<td>70mm</td>
<td></td>
</tr>
<tr>
<td>( R_{\text{ext}} )</td>
<td>exterior radius of the stator</td>
<td>18mm</td>
<td></td>
</tr>
<tr>
<td>( L_{\text{st}} )</td>
<td>slot opening width</td>
<td>3mm</td>
<td></td>
</tr>
<tr>
<td>( L_{\text{g}} )</td>
<td>air gap’s width</td>
<td>0.3mm</td>
<td></td>
</tr>
<tr>
<td>( N )</td>
<td>number of conductor per slot</td>
<td>optim. variable (3)</td>
<td></td>
</tr>
<tr>
<td>( N_{\text{enc}} )</td>
<td>number of slot per pole per phase</td>
<td>optim. variable (1)</td>
<td></td>
</tr>
<tr>
<td>( p )</td>
<td>number of pole pairs</td>
<td>optim. variable</td>
<td></td>
</tr>
<tr>
<td>( d_{s} )</td>
<td>permanent magnet’s depth</td>
<td>optim. variable</td>
<td></td>
</tr>
<tr>
<td>( d_{c} )</td>
<td>interior radius of the stator</td>
<td>optim. variable</td>
<td></td>
</tr>
</tbody>
</table>
continuous; the solution given by the optimizer for N will then be rounded to the closest integer value. Finally, the objective function value will be re-evaluated with the rounded value of N. This technique can lead to a problem called “integrality gap” which is that the optimal solution, with continuous variables, respects all the constraints whilst the neighbor points which correspond to discrete variables do not; and so the final solution, with discrete values, is far away from the solution given by the optimizer. In our case, no constraints are connected to N (cf. V), this technique is applicable.

The analytical model helps obtain the torque constant K, the inertia of the whole rotor $\bar{J}_{\text{rotor}}$, the resistance $R$ and inductance $L$ of the machine. We will use these values to determine the dynamics (acceleration, speed and displacement).

Since we have a variable load, which depends on the angular position of the drive shaft, it is difficult to have a direct formula to calculate the travel time of the machine. We choose the method of discretizing the time into small intervals and calculate the displacement and speed of the machine in each interval. Figure 4 presents the calculation principle of the objective function. At the iteration 1, all values are set to be 0. Then at each iteration k, we execute the following calculations:

- The electromotive force of the motor:
  \[ EMF(k) = K \\Omega (k-1) \]  
  \[ \text{(8)} \]

- The current I which flows in the machine:
  \[ I(k) = \frac{U-EMF(k)+L \frac{\text{d}I(k-1)}{\text{d}t}}{R+\frac{L}{\text{d}t}} \]  
  \[ \text{(9)} \]
  where $dt$ is time step.

- The resistive torque $\text{Torque}_{\text{Res}}(k)$ at each iteration is determined by Fig.1.

- The Torque given by the motor at each iteration is calculated by
  \[ \text{Torque}(k) = K I(k) \]  
  \[ \text{(10)} \]
  \[ \text{If the torque developed by the motor is superior to the resistive torque, the speed of the machine} \]
  \[ \\Omega (k) \text{ is calculated by} \]
  \[ \\Omega (k) = \Omega (k-1) + \frac{\text{Torque}(k)-\text{Torque}_{\text{Res}}(k)}{I_{\text{rotor}}+I_{\text{load}}} \text{dt} \]  
  \[ \text{(11)} \]

where $I_{\text{rotor}}$ and $I_{\text{load}}$ are the inertia of the rotor and the load (included the transmission system) seen by the motor. Otherwise, the travel time increases without the increase of displacement which will bring us to the limit time $(f=0.4 \text{ s})$.

- The angular position is calculated by:
  \[ \theta(k) = \theta(k-1) + \Omega(k)dt \]  
  \[ \text{(12)} \]

- And finally the travel time at iteration k is determined by
  \[ \text{time}(k) = \text{time}(k-1) + dt \]  
  \[ \text{(13)} \]
  At the end of each iteration, the angular position is compared with the final angular position of the motor, if the two values are equal, the calculation will stop and the value of the objective function is the value of time(k). Another comparison is conducted here to see if the travel time exceeds the limit time, if yes, the calculation is stopped, and the objective function is given the value 2.

IV. INTRODUCTION TO MATLAB’S OPTIMIZATION ALGORITHMS

A. Sequential Quadratic Programming (SQP)

This is a gradient-based optimization method. At each iteration, the gradient of the objective function and a linear step are calculated in order to calculate the variables for the next iteration.
According to [6] the stopping criteria of this algorithm are:

- The distance of the two consecutive points (input vectors of design parameters) is less than \( X \) tolerance ToIX.
- The change of the objective function between 2 iterations is less than function tolerance TolFun.
- The constraint violation is more than TolCon, which represents the maximum value by which parameter estimates can violate a constraint and still allow successful convergence.
- The number of iterations exceeds MaxIter.
- A rapid convergence is the advantage of this algorithm. However, the drawbacks are:
  - To use this algorithm, we must specify a starting point; moreover this point must not be far away from the global minimum unless the algorithm will converge to a local minimum. So to use this algorithm, we must have an idea on the zone where the optimal solution could be. To overcome this drawback, the multi-start technique can be used but this also means that the computing time will increase.
  - This algorithm can only be used when the objective function is continuous and differentiable, which is not always the case as seen in fig5.

B. Direct Search (DS)

Direct search methods belong to a class of optimization methods that do not compute derivatives. Examples of direct search methods are the Hooke and Jeeves’ pattern search [7], the Nelder-Mead Simplex method [8], the Dennis and Torczon’s parallel direct search algorithm PDS [9] and the Box method [10]. In Matlab, the pattern search method is used.

From an initial point \( M_i \), the algorithm will create a mesh with the central point is \( M_i \) then the value of the objective function is evaluated at each node of the mesh. The node \( M_{\text{min}} \) corresponding to the lowest function value is determined. If \( M_{\text{min}} \) and \( M_i \) are not the same point, \( M_{\text{min}} \) becomes the initial point for the next iteration. Moreover in the next iteration an expansion step to form a new mesh is carried out in which the size is expanded by some multiple, usually 2. If \( M_{\text{min}} \) and \( M_i \) are the same point, then a contraction step to form a new mesh is carried out in which the size is reduced by some multiple, usually 1/2. The algorithm stops when one of the stopping criteria is encountered. According to [6], Direct Search stops when one of these conditions is satisfied:

- The mesh size is less than Mesh tolerance.
- The change in the objective function from one successful poll to the next successful poll is less than Function tolerance.
- The distance of the two consecutive points (input vectors of design parameters) is less than X tolerance.
- The number of iterations exceeds MaxIter.
- The total number of objective function evaluations reaches the value of Max function evaluations.
- The change in the objective function from one successive iteration is less than X tolerance.
- A population is generated at each iteration. The population will approach one optimal solution.
- The next point in the sequence is selected by a deterministic calculation. The selection of the next population is partly random.

Because the Direct Search only uses function comparisons, no derivative is needed, this is an advantage compared to the SQP method when encountering problems which are difficult to compute their derivatives. However, as SQP methods there is still a drawback of this method: the initial point must be specified. This point must not be far from the global optimum unless the algorithm will converge to a local optimum.

C. Genetic Algorithm (GA)

This third optimization algorithm imitates the natural biological evolution. From an initial population of “individuals”, through selection, crossover and mutation, the algorithm will converge to the global optimum. At each iteration, the following procedure is conducted:

1) Choose initial population
2) Evaluate the fitness of each individual in the population
3) Repeat until termination:
   - Select best-ranking individuals to reproduce
   - Breed new generation through crossover and/or mutation (genetic operations) and give birth to offspring
   - Evaluate the individual fitnesses of the offspring
   - Replace worst ranked part of population with offspring

The difference between this algorithm and SQP and Direct Search (column standard algorithms) is presented in table II.

According to [6], this algorithm will stop when:

- The number of generations reaches the value of Max Generations.
- The running time in seconds equal to Time limit.
- The value of the fitness function for the best point in the current population is less than or equal to Fitness limit.
- The weighted average change in the fitness function value over Stall generations is less than Function tolerance.
- The algorithm stops if there is no improvement in the objective function during an interval of time in seconds equal to Stall time limit.
- The algorithm runs until the weighted average change in the fitness function value over Stall generations is less than Function tolerance.

The advantage of this algorithm, from the point of view of a user, is that we don’t have to specify the initial population, the algorithm itself will do this job. So it is particularly adapted to problems where there is no a priori
knowledge about the zone where the optimal solution is. But some drawbacks must be mentioned here:

- It is difficult to determine the stopping conditions.
- It is time-consuming because at each iteration, a population of individuals is evaluated.

To conclude, using SQP, Direct Search or Genetic Algorithm depends on the objective function’s nature and on the a priori knowledge that we have about the optimal solution.

V. RESULTS

We have applied the three algorithms to our objective function with the idea to test which one is the best fitted to this problem.

Since the different variables vary in different ranges which are not homogenous, they are normalized. The normalization consists of transforming the variation of these variables into the interval [0, 1]. The transformation is just a linear one in which the lower bound corresponds to 0 and the upper bound corresponds to 1. This helps the variation of all variables be homogenous.

The following constraints are posed to make sure that the final result is adequate:

- The drive shaft radius must be smaller than the interior radius of the stator minus the depth of the magnet and the air gap.
- The slot area must be positive
- The magnet’s depth must be inferior to a ratio of the interior radius of the stator. This ratio is determined to insure that the centrifugal and tangential forces do not pull the magnets out of the rotor.

Once the objective function and the constraint function (which contains inequality expressions and is the same for all the three algorithms that we proceed subsequently) have been made, the optimization procedure can be carried out. The three optimization variables can be presented by a vector \( X = [X_1, X_2, X_3] \) where \( X_1 \) is the normalized stator’s interior radius, \( X_2 \) is the normalized magnet’s depth and \( X_3 \) is the normalized number of conductors per slot. Each normalized variable \( X_i \) varies in the interval \([0, 1]\). The other geometric or electric variables are either fixed or dependent on these three optimization variables.

1) Firstly, we will use the SQP method with the Matlab function fmincon. As explained in the section IV-A, this method can only be used when the objective function is continuous and differentiable. This method cannot be applied to our problem because of the particularities of our objective function (figure 5): a big jump at the border of the valid zone and a flat invalid zone. Several tests with SQP confirmed this.

2) The search for an optimal configuration of our motor goes on with the utilization of “Direct Search” method. Our starting point is \( X_0 = [0.8182, 0.75, 0.75] \) which corresponds to \( R_{int-stator} = 11 \) mm, \( t_{m} = 6.5 \) mm and \( N = 11 \), and the stopping criterion is that tolerance over the change of \( X \) is \( 10^{-6} \). We obtained the final value \( X_f = [0.9312, 0.3335, 0.2540] \). We can see the evolution of the objective function throughout the optimization procedure with the figure 6. A comparison between the starting configuration (\( X_0 \)) and the final configuration (\( X_f \)) is presented in the figure 7. The Direct Search algorithm is robust: the starting configuration (figure 7(a)) chosen out of constraint on permanent magnet’s depth allows to reach a final solution which respects all the constraints (figure 7(b)).

3) Finally, we try out the Genetic Algorithm. As presented above, the algorithm does not require a starting point. The stopping criteria are the following:

- The population size is 30.
- Number of generation: 30
- Tolerance on the objective function : \( 1e^{-3} \).

With these criteria, we obtained the vector \( X_f = [0.5830, 0.2818, 0.5222] \) (see figure 8(a)); the value of the objective function is \( 0.1775 \).

Cascading method using SQP and GA has been presented in [11]. Here the result from the cascading method of Direct Search and Genetic Algorithm, which means using Direct Search, with the solution given by Genetic Algorithm as the starting point, will be presented. The final result is illustrated in the figure 8(b), the obtained vector is \( X_f = [0.583, 0.2804, 0.5226] \) and the value of the objective function is \( Fval = 0.17745 \), which is better than the result given by using GA only (0.17745 compares to 0.1775). There is only a small difference...
between the two configurations shown in figure 8.

![Image](image.png)

**Figure 8.** Optimal motor’s configurations obtained by GA and GA-DS in cascade

VI. CONCLUSION

From the results, we can see that in order to satisfy the travel time constraint, the motor can have either a high torque (large interior radius of the stator as shown in the case of Direct Search method) or low rotor inertia (smaller interior radius of the stator as shown in the case of Genetic Algorithm). There is not necessarily one unique good result when optimizing travel time: two optimal solutions can be found, one with a high torque and the other with low rotor inertia. In order to choose the definitive solution, other criteria must be taken into consideration such as losses, costs or heating (out of the scope of this article).

Concerning optimization algorithms, SQP algorithm is not suitable with our particular problem. However it is possible to use either Direct Search or Genetic Algorithm to obtain the optimal solution.

To be able to reach the global optimal solution with Direct Search method, one has to have an a priori knowledge on the zone where the optimal solution could be. This has been shown very clearly in our case: the result obtained by Direct Search has the value of the objective function of 0.18905 (fig 6), whilst the objective function’s value for the case of Genetic Algorithm is 0.1775, which means that the starting point for Direct Search does not lie near the optimal solution. This starting point leads to a local optimum.

Another aspect that we have to mention here is that the calculating time for Direct Search is much shorter than the calculating time for Genetic Algorithm (350 seconds compare to 13000 seconds). To overcome this limit, we can use the Genetic Algorithm first, with relatively less strict stopping criteria in order to reduce the calculating time, to obtain the starting point for Direct Search. This cascading utilization of the two algorithms will help to have fully the advantages and avoid the drawbacks of the two algorithms.

REFERENCES


