A Hybrid Optimization Algorithm for Electric Motor Design

Mokhtar Essaid[1(✉)], Lhassane Idoumghar[1], Julien Lepagnot[1], Mathieu Brévilliers[1], and Daniel Fodorean[2]

[1] IRIMAS, University of Haute-Alsace, 68093 Mulhouse, France
{mokhtar.essaid, lhassane.idoumghar, Julien.lepagnot, Mathieu.brevilliers}@uha.fr
[2] Department of Electrical Machines and Drives, Technical University of Cluj-Napoca, 400114 Cluj-Napoca, Romania
daniel.fodorean@emd.utcluj.ro

Abstract. This paper presents a hybrid algorithm employed to reduce the weight of an electric motor, designed for electric vehicle (EV) propulsion. The approach uses a hybridization between Cuckoo Search and CMAES to generate an initial population. Then, the population is transferred to a new procedure which adaptively switches between two search strategies, i.e. one for exploration and one for exploitation. Besides the electric motor optimization, the proposed algorithm performance is also evaluated using the 15 functions of the CEC 2015 competition benchmark. The results reveal that the proposed approach can show a very competitive performance when compared with different state-of-the-art algorithms.

Keywords: Electric motors · Hybridization · Cuckoo search · CMAES

1 Introduction

Metaheuristics are widely used to solve complex optimization problems within a reasonable time. According to [3], most of metaheuristics have the common following characteristics: they are nature-inspired, and based on stochastic components (randomness). Besides, a balance should be preserved between the diversification and exploitation phases [5]. Otherwise, metaheuristics would suffer from falling in local optima (weak diversification) or from slow convergence (weak exploitation). To overcome these limitations, hybridization between different algorithmic components appears to be an appropriate choice. Hybrid metaheuristics have attracted a lot of attention from researchers to solve optimization problems. It has been proven that choosing an appropriate combination of algorithmic concepts can lead to a successful solving of many hard optimization problems [2].

In this paper, we propose a hybrid algorithm to optimize the weight of electric motors within electric vehicles. The hybridization is based on a CMAES-enhanced Cuckoo Search (CS) which provides a well distributed initial population. Then, the population is transferred to a surrogate model-based LSHADE to optimize the solutions obtained so far.

Y. Shi et al. (Eds.): ICCS 2018, LNCS 10861, pp. 501–517, 2018.
https://doi.org/10.1007/978-3-319-93701-4_39

LSHADE is a recent version of differential evolution algorithm (DE) that incorporates success-history based parameter adaptation [17] with Linear Population Size Reduction (LPSR) [18]. The main motivation for using LSHADE as the main optimizer is its high adaptability when solving many optimization problems [1]. In addition to that, LSHADE ranked first at the CEC 2014 competition. However, our approach relies on a modification of this algorithm, where we aim to improve its exploration capability by integrating the K-means clustering algorithm to generate the central individuals of a given population. Then, we apply a Lévy Flight movement on them to generate new individuals. The proposed approach is evaluated using the 15 test functions of the CEC 2015 benchmark and it is compared with LSHADE, DE and Cuckoo Search (CS). This experimentation shows that our proposition obtains very competitive results on the electric motor problem at hand and on the CEC 2015 benchmark.

The rest of this paper is organized as follows. In Sect. 2, the electric motor problem is discussed. In Sect. 3, the algorithmic components of our proposition are presented. In Sect. 4, our approach is explained in details. Then, the results are discussed in Sect. 5. Finally, Sect. 6 concludes the paper.

2 Electric Motor Design

Here the application under study is introduced. The hybrid optimization approach will be applied to optimize the design of an electric motor used to propel an electric vehicle (EV). Concerning the EV application one can observe that from the first hybrid vehicle in series production, the Toyota Prius Hybrid, have past about 20 years. At that time, the permanent magnet synchronous machine (PMSM) used to propel the hybrid vehicle was designed at 6 000 r/min. The PMSM structure from today's Prius Hybrid has a double speed, 12 000 r/min. Moreover, for all other manufacturers, the electric motors used for the traction present a speed level in the same range: BMW-i3 is running at a top speed of 11 400 r/min, Renault-Zoe at 11 300 r/min, etc. The idea of increasing the speed of the electric propulsion is due to the improved power density (i.e., power/weight ratio) of the high speed traction motor: the higher the speed of the motor, the better the power density. Since the power density is improved, meaning that for the desired out power, the weight of the machine is reduced, the operation range (the main drawback of electric vehicles today), meaning the autonomy of the electric car, can be increased [6, 7]. Also, by reducing the weight and consequently the volume of the electric motor, one could consider to increase the capacity of the supplying battery, again, with a clear advantage on terms of vehicle's autonomy.

For the above reasons our interest was to find the best suited motor topology, capable to run at higher speeds. We have established the desired output power, which is 20 kW, while the motor is supplied from a battery of 380 Vdc. In Fig. 1(a) is presented the 3D view of the high speed PMSM which was designed to run at 22 000 r/min and where one can identify the main components of the active parts of the machine, meaning: the stator and rotor parts, the inset permanent magnets (PMs) with the appropriate polarity. All the calculation of the obtained performances are made on the length of the machine (Lm in Fig. 1(a)). The analysis with respect to the obtained performances of the designed highs-speed PMSM is carried out with a numerical finite

element method (FEM). Based on this one will get the electromagnetic and mechanical performances. This analysis stands for validation of the analytical design of the machine and will also be used to verify the optimized results proposed by our optimization approach. In Fig. 1(b) is presented, in several slices on the length of the machine, the flux density repartition while the machine is running at high speed. Based on this analysis we can see which machine's iron regions present a risk of saturation. Since we are using very good steel, we can conclude that the machine is not oversaturated.

The structure is designed to have only 2 magnetic poles. Since the frequency of the supply is proportional to the machine's magnetic poles and since the iron losses of the machine are proportional to the square of the frequency, it means that the machine efficiency is drastically affected by the frequency/poles values. Thus, having the lowest possible number of poles will offer the possibility to reduce at maximum the frequency of the machine, and finally to have the most appropriate magnetic configuration. And from this point on, we can consider to optimize the structure itself in order to obtain the best power density (power/weight ratio).

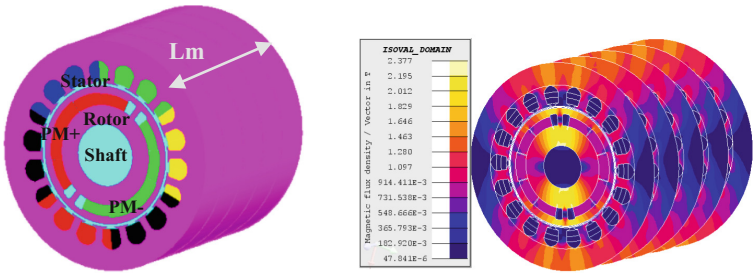

Fig. 1. (a) 3D view of the considered high-speed PMSM; (b) Numerical FEM analysis for flux density distribution on the high-speed PMSM

3 Approach Components

We provide in this section a brief presentation of the algorithmic components used in our proposition.

3.1 CMA-ES

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a class of evolutionary algorithms where the new population is generated by sampling from a probability distribution constructed during the optimization process. CMA-ES is briefly explained in Algorithm 1.

Algorithm 1: Pseudocode of CMA-ES algorithm

1. $\lambda \leftarrow$ number of samples per iteration
2. $\mu \leftarrow$ number of recombination points
3. Initialize state variables m, σ, $C = I$, $p_\sigma = 0$, $p_c = 0$
4. Repeat
5. For $i = 1$ to λ do
6. $x_i^{(t+1)} \leftarrow$ sample i^{th} solution according to (1)
7. $f_i \leftarrow$ evaluate i^{th} solution
8. End for
9. Sort the new solutions and find the first μ solutions
10. $m^{(t+1)} \leftarrow$ update the mean value according to (2)
11. $p_c^{(t+1)} \leftarrow$ update anisotropic evolution path according to (4)
12. $C^{(t+1)} \leftarrow$ update the covariance matrix according to (5)
13. $p_\sigma^{(t+1)} \leftarrow$ update isotropic evolution path according to (6)
14. $\sigma^{(t+1)} \leftarrow$ update the step size using isotropic path length according to (7)
15. Until (stopping condition = true)

In this algorithm, solutions are generated using a multivariate normal distribution N with mean m and a covariance C. According to [11], a new solution x^{t+1} is generated as follows:

$$x^{t+1} = m^t + \sigma^t N(0, C^t) \tag{1}$$

$$m^t = \sum_{i=1}^{\mu} w_i x_{i:\lambda}^t \tag{2}$$

$$w_i = \log\left(\mu + \frac{1}{2}\right) - \log(i), \sum_{i=1}^{\mu} w_i = 1 \tag{3}$$

where m^t is the weighted mean of the μ best solutions, $x_{i:\lambda}^t$ is the t[th] ranked individual, λ is the number of samples, σ^t is the step size parameter. Besides, a covariance matrix C^t is adapted using an evolution path p_c^{t+1}. It is generated with the following equation:

$$p_c^{t+1} = (1 - c_c)p_c^t + \sqrt{c_c(2 - c_c)}\frac{\sqrt{\mu}}{\sigma^t}\left(m^{t+1} - m^t\right) \tag{4}$$

$$C^{t+1} = (1 - c_{cov})C^t + c_{cov}p_c^{t+1}(p_c^{t+1})^T \tag{5}$$

where c_c and $c_{cov} \in [0, 1]$ are learning rates for p_c^{t+1} and C^{t+1} respectively.

Moreover, the step size parameter is updated through the evolution path p_σ^{t+1} as below:

$$p_\sigma^{t+1} = (1 - c_\sigma)p_\sigma^t + \sqrt{c_\sigma(2 - c_\sigma)}\sqrt{\mu}B^t m^{t+1} \tag{6}$$

where c_σ is a learning rate controller, and B^t is the normalized eigenvectors of C^t. Then, σ^{t+1} is updated as follows:

$$\sigma^{t+1} = \sigma^t \exp\left(\frac{\left|\left|p_\sigma^{t+1}\right|\right| - T_n}{d_\sigma T_n}\right) \tag{7}$$

$$T_n = \sqrt{n}\left(1 - \frac{1}{4n} + \frac{1}{21n^2}\right) \tag{8}$$

where n represents the problem dimension and $d_\sigma > 1$ is a damping parameter.

3.2 Cuckoo Search

Cuckoo search (CS) is an optimization algorithm that simulates brood parasitism of cuckoo birds [19]. These birds lay their eggs in other bird's nests. When host birds find the foreign eggs, they will either throw them, or abandon the nest. Following this model, each egg represents a solution, and each new solution is represented by a cuckoo egg. Cuckoo bird replaces bad eggs in the host nest with better feasible eggs. Moreover, CS algorithm simulates the food foraging process of many animals and insects [8]. Commonly, the foraging path of animals is a random walk where the next move depends on the actual location and the transition probability to the next location. However, Lévy flight random walk is proved to be more efficient for searching [8]. In CS, a balanced combination of local random walk and a global random walk is obtained through a switching parameter P_a. The local random walk is written as:

$$x_i^{t+1} = x_i^t + s \otimes H(P_a - \epsilon) \otimes \left(x_j^t - x_k^t\right) \tag{9}$$

where $\otimes$ represents entry-wise multiplications x_j^t, x_k^t are solutions randomly selected and H is heaviside function. ϵ and s are random numbers generated from a uniform distribution. The global random walk is handled using Lévy flights:

$$x_i^{t+1} = x_i^t + a \otimes \text{Lévy}(\beta) \tag{10}$$

where

$$a = a_0 \otimes \left(\left(x_j^t - x_i^t\right)\right) \tag{11}$$

$$\text{Lévy}(\beta) = \frac{u}{|v|^{1/\beta}} \tag{12}$$

a_0 is a step size scaling factor and β is Lévy Flights exponent. Finally, u and v are two numbers with zero means and associated variance. Indeed, using Lévy Flights and combining local and global search capabilities makes CS a very efficient algorithm. *CS* pseudo-code is presented in Algorithm 2.

Algorithm 2: Pseudocode of CS algorithm	
1.	Initialize a population of n host nests x_i *(i=1,2.... n)*
2.	While stop criterion is not reached
3.	Get a cuckoo (say *i*) randomly and generate a new solution according to (10)
4.	Evaluate its fitness F_i
5.	Choose randomly a nest among n (say *j*)
6.	If ($F_i < F_j$)
7.	Replace *j* by the new solution
8.	End
9.	Abandon a fraction (p_a) of worse nests
10.	Build new nests according to (9)
11.	Keep the best solutions among the current nests and the new generated ones
12.	Rank the solutions and find the current best
13.	End while

3.3 K-means

K-means is a widely used clustering algorithm. The parameter K is a given integer representing the number of centers. The algorithm assigns each point from a given set of points to the nearest center among the K centers [14]. Algorithm 3 presents the pseudo-code of K-means.

Algorithm 3: Pseudocode of K-means algorithm	
1.	Input: A data set D
2.	Output: K clusters
3.	Choose k centers C_1, C_2, ... C_k randomly from n points $(X_1, X_2, .. X_n)$
4.	Assign point X_i, $i=1,2, ..., n$ to the nearest center C_j, $j \in \{1, 2, ..., k\}$
5.	Compute new cluster centers as follows:
6.	$C_i^* = \frac{1}{\lvert C_i \rvert} \sum_{X_j \in C_i} X_j,\ i = 1, 2, \ldots, k$
7.	Stop if termination criterion is satisfied. Otherwise, continue from step 3

3.4 LSHADE

LSHADE is an adaptive version of differential evolution algorithm (*DE*). It incorporates success-history-based parameter adaptation [17] with Linear Population Size Reduction (LPSR) [18]. The convergence performance of LSHADE is improved by using the mutation strategy current to-pBest/1/bin [20] to generate mutant vectors. It is proved that this strategy is very efficient for the generation of high quality individuals [20].

Current to-pBest/1/bin is expressed as follows:

$$v_{i,g} = x_{i,g} + F\left(x_{best,g} - x_{i,g}\right) + F\left(x_{r1,g} - x_{r2,g}\right) \tag{13}$$

where $x_{best,g}$ is a randomly selected parent from the top best individuals of the current individuals. To maintain the diversity of the population, an archive A is proposed. The parent solutions that are not selected are added to this archive. Moreover, success-history-based parameter adaptation is a mechanism used to store successful CR, F values that performed well in the past generations. After the generation of a new

trial vector u_i, it is compared with its parent. If u_i is better, then the CR and F parameters are stored in the sets S_{CR}, S_F respectively. Finally, the memories M_{CR} and M_F are updated using these successful parameters according to the following equations:

$$M_{CR} = \begin{cases} meanw_A(S_{CR}) & if\ S_{CR} \neq \emptyset \\ M_{CR} & otherwise \end{cases} \tag{14}$$

$$M_F = \begin{cases} meanw_L(S_F) & if\ S_F \neq \emptyset \\ M_F & otherwise \end{cases} \tag{15}$$

where $meanw_A$ is the weighted mean and $meanw_L$ is the Lehmar mean.

For further details about LSHADE, we refer the reader to [18].

3.5 Radial Basis Function (RBF)

Surrogate models can be used as approximation models for the cost functions of optimization problems [10]. There are several surrogate models in the literature such as polynomial Response Surface Model RSM, Kriging and Radial Basis Function (RBF). In our proposition, RBF has been chosen thanks to its acceptable accuracy and to its relative simplicity compared to other surrogate models. A brief description of the RBF surrogate model is presented. Assuming that we have n given points $x_1, x_2, \ldots, x_n$ whose true function values $f(x_i)$ are known. In this method, an interpolant is used as follows:

$$s(x) = \sum_{i=1}^{k} w_i \emptyset(|x - x_i|) + b^T x + a, \quad x \in R^D \tag{16}$$

where $w = (w_1, w_2, \ldots w_n)^T \in R^n$, $b \in R^D$, $a \in R$, and $\emptyset$ is a cubic basis function. Moreover, the parameters w, b, a are obtained by solving the following system of linear equations:

$$\begin{pmatrix} \Phi & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} w \\ c \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix} \tag{17}$$

where Φ is the $n \times n$ matrix with $\Phi_{i,j} = \emptyset\left(\left\|x_i - x_j\right\|_2\right), c = (b^T, a)^T$,

$$F = (f(x1), f(x2), \ldots, f(xn))^T$$

and

$$P^T = \begin{pmatrix} x_1 & x_2 & \ldots & x_n \\ 1 & 1 & \ldots & 1 \end{pmatrix} \tag{18}$$

For more details about RBF surrogate model, the reader can refer to [10].

4 The Proposed Approach

An initialization strategy can play an important role on the algorithm performance. Several works have investigated the effect of initialization techniques in finding the global optima such as [12]. In our approach, a CMA-ES-enhanced CS initialization technique is proposed to provide well-distributed points over the search space. Our goal is to investigate the high capability of CS in exploring the search space as well as the CMA-ES, which is able to provide high quality solutions with a limited number of evaluations. First, CMA-ES algorithm is run for a certain number of evaluations. Then, the produced solution is provided along with a randomly generated population to CS procedure. The goal is to speed up the convergence rate of CS and lead the search towards well-distributed individuals. Afterwards, the population *pop* of *pop_size* individuals obtained from the last iteration of CS is used to train the RBF model. Thereafter, it will be provided to the main procedure of our proposition.

Since it is important for a metaheuristic to show a good balance between exploration and exploitation, we propose a technique to handle this issue. For each generation of the main procedure, it is decided with a probability *PLV* whether the algorithm performs the global search procedure, or whether one iteration of the modified LSHADE is performed instead as a local search procedure. The global search procedure consists of a clustering algorithm and a Lévy Flight perturbation. The clustering is used for its high capability of avoiding redundant search points during the search process [15] as well as its efficiency in performing global search [8, 9]. The clustering is considered as a search operator that exploits the whole information of the population to generate a certain number of centers (new individuals). In our approach, the K-means algorithm is chosen because it has experimentally shown a superior performance than other algorithms as the fuzzy c-means (FCM) which is demonstrated in the results. K-means is exploited in order to generate K central individuals of the current population. Then, the central individuals are relocated to potentially more promising areas of the search space using Lévy Flight perturbation. The Lévy Flight movement is applied according to:

$$step\_size_i = 0.001 * step_i * (z_i - best)\ i = 1, 2, \ldots K \tag{19}$$

where $step_i$ is generated according to (11), *best* is the best solution so far and z is the set of the central individuals. Then, the new trial individuals are generated as follows:

$$z_i = z_i + step\_size_i \tag{20}$$

As a replacement strategy, the best *pop_size* individuals of $pop \cup z$ are selected for the next iteration. Then, *PLV* parameter is decreased to progressively shift the search process from exploration to exploitation. This parameter is updated according to the following equation:

$$PLV = \max\left\{0, PLV - \frac{current\_fes}{\max\_fes}\right\} \tag{21}$$

Algorithm 4: One generation of our SLSHADE algorithm

Given a population of *pop_size* individuals
Given S_{CR}, S_F
Compute M_{CR}, and M_F according to (14), (15)
Generate a mutant population pop_a according to (13)
Generate a mutant population pop_b according to (22)
Evaluate approximately pop_a and pop_b using RBF surrogate model
Choose the population containing the best solution
For each x_i in the population
  Use binomial crossover to generate the trial vector u_i
  Evaluate u_i using the actual objective function
  If f(u_i) < f(x_i)
    $x_{i+1} \leftarrow u_i$, $A \leftarrow x_i$, $S_{CR} \leftarrow CR_i$, $S_F \leftarrow F_i$
  Else
    $x_{i+1} = x_i$
  End If
End for
Reduce the population by using LPSR

The decreasing procedure makes the approach more exploitative during the search process by enhancing the possibility of performing the exploitation procedure.

As a local search technique, a proposed surrogate model-based LSHADE (SLSHADE) algorithm is used, which is depicted in Algorithm 4. LSHADE is an adaptive version of DE that proved its high capability in solving a wide range of complex optimization problems. Its parameter adaptation strategy allows achieving an efficient exploitation [4]. In SLSHADE, we modify the mutation strategy by using an additional mutation equation and a simple switching technique to choose the best mutation operator for the next generation. The proposed mutation equation is as follows:

$$v_{i,g} = \left(x_{i,best} - x_{i,g}\right) + step\_size_i \tag{22}$$

where $step\_size_i$ is computed according to (11), and $x_{i,best}$ is the i_{th} best solution in the current population. To save the limited budget of evaluations, we use the RBF for an approximated evaluation. After applying the two search operators (Eqs. (13) and (22)), RBF is used to approximate the fitness of the two mutant populations. Afterwards, the population that contains the best solution is chosen to perform the binomial crossover of LSHADE. To provide a fair approximation, the RBF model is updated using the current population after each quarter of the available budget. The pseudo-code of the full proposition is depicted in Algorithm 5.

Algorithm 5: Pseudocode of the proposed approach

```
Archive A ← ∅
S_CR ← ∅, S_F ← ∅ // sets of successful CR and F in the previous generation
Set all values in M_CR, M_F to 0.5
Generate a solution S_cmaes using CMA-ES for a number of evaluation α
Perform CS to generate pop of pop_size individuals for a number of evaluation ρ
Train the RBF surrogate model using each individual in pop and its fitness
While current_fes < Budget
    If (current_fes < max_fes) and (rand < PLV)
      For i=1:T
          Generate K central individuals of pop using K-means
          Generate a step size for each center using Lévy flights
          Generate new individuals according to (20)
          Evaluate individuals and replace the bad individuals of the population
          current_fes ← current_fes + K
      End for
      Decrease PLV according to equation (21)
    Else
      Perform SLSHADE (pop, S_CR, S_F)
    End if
    Update the RBF model using the current population after each quarter of the budget
End while
```

5 Experimental Results

This section presents an evaluation of our proposition on the problem at hand. We compare it with LSHADE as well as CS [19] and DE [16]. Moreover, we evaluated our algorithm on the 15 functions of the CEC 2015 test suite [13] in order to provide difficult test cases. The parameter setting of the compared algorithms are given in Table 1.

Table 1. Parameter setting of the compared algorithms

Algorithm	Parameters	Electric motor design problem	CEC 2015
Our proposition	α	$D * 1000$	$D * 1000$
	ρ	$D * 1000$	$D * 1000$
	pop_size	200	200
	k	*pop_size*/4	*pop_size*/4
	PLV	0.8	0.8
	Budget	10^6	$D * 10000$
	max_fes	2/3 * Budget	1/3 * Budget
	T	100	1
CS	P_a	0.25	0.25
	b	1.5	1.5
DE	*CR*	0.5	0.5
	F	0.5	0.5
LSHADE	r^{init}	18	18
	r^{arc}	2.6	2.6
	p	0.11	0.11
	Size of archive *A*	6	6

5.1 Comparison on CEC 2015 Benchmark

The comparison has been carried out using the CEC 2015 test suite. The benchmark contains 15 functions to be minimized with a limited budget of 10000 * D evaluations, where D is the dimension of the search space. Table 2 presents in detail the characteristics of CEC 2015 benchmark. The performance is evaluated using $D = 30$ and we performed 30 runs for each function.

Table 2. Problem definitions for the CEC 2015 competition on learning-based real-parameter single objective optimization

No	Function	Range	F_i
1	Rotated high conditioned elliptic function	$[-100, 100]^D$	100
2	Rotated cigar function	$[-100, 100]^D$	200
3	Shifted and rotated Ackley's function	$[-100, 100]^D$	300
4	Shifted and rotated Rastrigin's function	$[-100, 100]^D$	400
5	Shifted and rotated Schwefel's function	$[-100, 100]^D$	500
6	Hybrid function 1	$[-100, 100]^D$	600
7	Hybrid function 2	$[-100, 100]^D$	700
8	Hybrid function 3	$[-100, 100]^D$	800
9	Composition function 1	$[-100, 100]^D$	900
10	Composition function 2	$[-100, 100]^D$	1000
11	Composition function 3	$[-100, 100]^D$	1100
12	Composition function 4	$[-100, 100]^D$	1200
13	Composition function 5	$[-100, 100]^D$	1300
14	Composition function 6	$[-100, 100]^D$	1400
15	Composition function 7	$[-100, 100]^D$	1500

To demonstrate the importance of each component, an experimentation has been conducted to compare the proposed algorithm with other variants. In the first variant, the initialization method is removed and it is called "*variant-1*". In the second variant, the switching technique to the global search procedure is disabled. The algorithm becomes a hybridization between the proposed initialization method and SLSHADE. Moreover, to demonstrate the clustering impact in producing new individuals, K-means algorithm has been replaced by the FCM clustering algorithm "*variant-3*".

Each column from Table 3 shows best, mean and standard deviation of each algorithm for each function. The best fitness found for each function is in bold. Mean results that are significantly better than the ones of the other algorithms, according to the Kruskal-Wallis statistical test at 95% confidence level followed by a Tukey-Kramer post hoc test are also in bold. The results presented in Table 4 show a superior performance of our proposition over the other algorithms. It can significantly outperform LSHADE in 6 functions, DE and CS in 15 functions. The comparison with the three variants reveals that our proposition could achieve better performance as well. It can outperform variant-1 in 8 functions which shows the importance of the proposed initialization method. Similarly, disabling the global search has a major effect on the algorithm. *Variant-2* performs significantly worse than the proposition in 8 functions.

Table 3. Comparison of CS, DE, LSHADE and our proposition on CEC 2015 test suite

		CS	DE	LSHADE	Variant-1	Variant-2	Variant-3	Our proposition
F1	Best	4.06E+06	1.84E+07	**100**	100.00	100.00	100.00	**100**
	Mean	5.88E+06	3.25E+07	100.00	100.00	100.00	100.00	**100**
	Std	1.09E+06	6.62E+06	6.34+E−05	0.02	8.6e−04	4.2e−03	0
F2	Best	1.00E+11	1.39E+04	**200**	200.00	200.00	200.00	**200**
	Mean	1.00E+11	7.50E+06	**200**	200.00	200.00	200.00	**200**
	Std	0	4.78E+04	0	3.07e−14	2.63e−14	7.46e−15	0
F3	Best	320.80	320.60	320	320.03	320.05	320.04	**319.99**
	Mean	320.90	320.68	320.05	320.07	320.09	320.09	**319.99**
	Std	0.03	0.03	0.01	0.01	0.01	0.01	0
F4	Best	525.86	532.89	404.98	405.98	408.97	406.97	**403.99**
	Mean	575.17	547.86	**410.04**	**410.25**	415.76	415.18	**410.19**
	Std	19.99	8.26	2.04	2.58	7.01	4.08	2.19
F5	Best	3 499.36	5 368.68	**1 315.40**	1490.93	1992.54	1899.11	1 618.87
	Mean	4 053.09	5 940.68	**1 768.50**	**1865.59**	2370.67	2348.35	2 147.12
	Std	193.60	234.93	200.44	181.26	195.42	199.84	229.57
F6	Best	5.64E+04	8.84E+05	673.18	676.37	**654.09**	663.72	712.41
	Mean	1.08E+05	2.20E+06	1 021.14	1014.76	861.93	**796.49**	928.25
	Std	2.89E+04	8.09E+05	215.63	230.03	129.32	88.51	151.84
F7	Best	711.84	712.73	705.85	705.56	704.928	705.07	**703.15**
	Mean	713.41	713.71	**706.84**	**706.84**	**706.24**	**706.15**	**706.40**
	Std	0.91	0.36	0.56	0.63	0.68	0.55	0.94
F8	Best	5 384.50	1.34E+05	810.82	811.42	806.97	**801.21**	811.71
	Mean	9 497.06	4.12E+05	906.48	909.34	**852.84**	**838.76**	**859.89**
	Std	1 941.79	1.39E+05	68.79	80.78	43.78	39.22	62.67
F9	Best	1 004.23	1 003.27	**1 002.37**	1002.59	1002.39	1002.42	1 002.43
	Mean	1 004.79	1 003.50	**1 002.81**	**1002.85**	**1002.72**	**1002.75**	**1 002.80**
	Std	0.24	0.15	0.18	0.14	1.3e−01	0.14	0.15
F10	Best	1.22E+04	1.01E+05	1 413.77	1265.26	1170.70	1152.72	**1 100**
	Mean	2.38E+04	5.13E+05	**1 691.50**	**1609.82**	**5330.32**	**1474.494**	**1563.60**
	Std	4864.28	1.81E+05	171.69	223.07	17629.47	200.49	178.13
F11	Best	1 428.42	1 694.51	1 500	1500	1400.75	1400.72	**1 400.7**
	Mean	1 443.50	1 874.25	1 515.42	1519.98	**1410.13**	**1406.101**	**1 401.10**
	Std	9.86	125.368	23.28	27.34	9.26	8.03	0.56
F12	Best	1 305.67	1 306.58	1 303.42	1303.53	1303.24	1303.24	**1 303.2**
	Mean	1 306.46	1 307.71	**1 304.02**	**1303.95**	**1303.83**	**1303.78**	**1 304**
	Std	0.36	0.55	0.26	0.23	0.27	0.23	0.38
F13	Best	1 300.02	1 300.02	1 300.02	1300.02	1300.02	1300.02	**1 300**
	Mean	1 300.02	1 300.02	1 300.02	1300.02	1300.02	1300.02	**1 300**
	Std	0	0	0	2.1e−04	2.75e−04	2.57e−04	0
F14	Best	3.33E+04	3.49E+04	3.25E+04	34167.93	32486.61	32499.62	**1 500**
	Mean	3.39E+04	3.51E+04	3.44E+04	34832.79	34085.43	34132.58	**1 500**
	Std	402.35	117.80	735.88	208.95	667.58	625.16	0
F15	Best	1 606.34	1 600.00	**1 600**	1600.00	1600.00	1600.00	**1600**
	Mean	1 607.50	1 600.00	**1 600**	1600.00	1600.00	1600.00	**1600**
	Std	0.59	0	0	0	4.22e−14	0	0

Variant-3 shows slightly better performance when compared to *variant-1* and *variant-2*. However, the proposition can significantly outperform it in 7 functions. These results demonstrates the impact of clustering. It is observed that both clustering methods can significantly improve the performance of the proposed algorithm.

Table 4. Comparison using Kruskal-Wallis test on CEC 2015 test suite

vs	Our proposition	D = 30
LSHADE	+(better)	1
	−(worse)	6
	=(no sig)	8
CS	+(better)	0
	−(worse)	15
	=(no sig)	0
DE	+(better)	0
	−(worse)	15
	=(no sig)	0
Variant-1	+(better)	1
	−(worse)	8
	=(no sig)	6
Variant-2	+(better)	0
	−(worse)	8
	=(no sig)	7
Variant-3	+(better)	1
	−(worse)	7
	=(no sig)	7

5.2 Comparison on HS-PMSM

The paper's objective is to reduce HS-PMSM weight and by that, increase EV's autonomy. The problem at hand is a multi-objective problem, where the objective functions are as follows:

1. The first objective function concerns the mass of the electric motor m_{atot}:

$$m_{atot} = m_{cooper} + m_{stat} + m_{rot} + m_{pm} \tag{23}$$

   where m_{cooper} is the cooper mass, m_{stat} is the stator iron mass, m_{rot} is the rotor iron mass, and m_{pm} is the magnets mass.
2. The second objective function is to maximize the output power density. It is written as follows:

$$P_{out} = P_{in} + \sum losses \tag{24}$$

   where P_{out} is the output power density, P_{in} is the input density, and the sum of losses mainly contains the mechanical, iron and copper loss component.

Table 5. The problem constraints

Parameter	Symbol	Unity	Variation limits
Output power	P_{out}	W	[19995; 20005]
Current consumption	I_s	A	[20; 56]
Motor torque	T_m	Nm	[8.5; 8.6]
Motor's efficiency	η	-	[0.9; 0.99]
Motor's power factor	PF	-	[0.81; 0.99]
Rotor inner diameter	Dir	mm	[22; 70]
Slot filling factor	τ	-	[0.1; 0.5]

The two objective functions are aggregated to obtain the following new objective function which will be optimized using the proposed algorithm:

$$\min j(x) = -\frac{P_{out}}{m_{atot}} + penality \tag{25}$$

where

$$penality = 10^4 \sum_{i=1}^{7} C_i \tag{26}$$

$C_i = 0$ if the constraint i is satisfied, 1 otherwise.

The set of constraints are presented in Table 5. There are 8 variables for the optimization problem at hand, i.e. 8 geometrical parameters controlling the electric motor structure. The parameters are presented in Table 6.

Table 6. The geometrical parameters for the weight optimization

Symbol	Description	Variation limits
Dis	Inner stator diameter	[50; 80] mm
hjr	Rotor yoke height	[7; 15] mm
histm	Tooth isthmus	[0.5; 2] mm
hjs	Stator yoke height	[8; 15] mm
wt	Tooth width	[3.5; 8] mm
gap0	Air-gap length	[0.5; 1.5] mm
hmp	PM height	[4; 8] mm
Lm	Machine's length	[100; 160] mm

To conduct a fair comparison, the proposed algorithm has been run 30 times. We collected the best, the mean, the median, the worst, and the standard deviation of each algorithm. It is observed from Table 7 that our proposition obtains the best solution compared to the other algorithms.

Table 7. Results on the real problem after 30 runs

	CS	DE	L-SHADE	*Variant-1*	*Variant-2*	*Variant-3*	Our proposition
Best	−3.308e+03	−3.156e+03	−3.197e+03	**−3.397e+03**	−3.318e+03	−3.380e+03	**−3.397e+03**
Mean	−3.131e+03	−2.910e+03	−3.044e+03	−3.114e+03	−3.202e+03	−3.188e+03	**−3.397e+03**
Median	−3.179e+03	−3.024e+03	−2.816e+03	−2.937e+03	−3.283e+03	−3.210e+03	**−3.397e+03**
Worst	−3.034e+03	−2.848e+03	−2.797e+03	−2.935e+03	−3.130e+03	−3.085e+03	**−3.397e+03**
Std	88.52	91.23	107	112.10	49.03	58.36	**0**

Besides, a stable performance is achieved, since the proposition could obtain the best solution in each run. The comparison between the 3 variants reveals that variant-2 could obtain the best results. It demonstrates clearly the importance of the proposed initialization method. *Variant-3* shows an inferior performance when compared to the proposition. Thus, it can be concluded that each component of our proposition tends to be effective and the combination as a whole leads to a successful algorithm. Further details about the optimal solution found by our proposition are depicted in Table 8. Regarding the optimized obtained results, the proposed algorithm could achieve an important gain of 28% in the mass. Moreover, it could achieve a gain of 17% and 29% decreasing the mechanical loss and the iron loss stator respectively.

Table 8. The best geometrical parameters with the optimized factors

Symbol	Original motor	Optimized motor	Gain %
m_{atot}	8.2513 kg	5.8885 kg	**+28.63**
P_{out}	20000 W	20005 W	**+0.25e−3**
P_{out}/m_{tot}	2.42 kW/kg	3.39 kW/kg	**+28.653**
Iron loss stator	225.73 W	158.9 W	**+29.60**
Mechanical loss	352.69 W	292.15 W	**+17.16**
Efficiency	0.9596	0.9607	**+1.01**
Power factor	0.8187	0.8100	**−1.06**
Dis	63 mm	66.7 mm	
hjr	10.5 mm	9.3 mm	
histm	1.5 mm	1 mm	
hjs	11.8 mm	9.8 mm	
wt	5 mm	4 mm	
gap	1 mm	0.9 mm	
hmp	6 mm	4 mm	
Lm	135 mm	100 mm	

6 Conclusion

The paper has presented a successful hybridization to solve numerical optimization problems. The proposition consists in combining 2 state-of-art algorithms as an initialization method. Then, the produced population is transferred to the main procedure. The latter switches between the global and the local search procedures giving

progressively the priority to the local search procedure to adaptively enhance exploitation in the algorithm. The proposition has been tested on CEC 2015 test suite and on the optimization of an electric motor. The obtained results have shown a stable and competitive performance compared to other state-of-art algorithms. Besides, a superior performance of K-means over FCM clustering algorithm has been noticed in the global search procedure. Thus, as a future work, we aim to justify this superiority by conducting an experimentation integrating visualization tools, in order to analyze the behavior of each clustering method. We also aim to integrate recent landscape analysis strategies to switch between the search operators (local and global search) in order to investigate their influence on the algorithm performance.

References

1. Awad, N.H., Ali, M.Z., Suganthan, P.N., Reynolds, R.G.: An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark prolems. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 2958–2965 (2016)
2. Blum, C., Puchinger, J., Raidl, G.R., Roli, A.: Hybrid metaheuristics in combinatorial optimization: a survey. Appl. Soft Comput. **11**(6), 4135–4151 (2011)
3. Boussaid, I., Lepagnot, J., Siarry, P.: A survey on optimization metaheuristics. Inf. Sci. **237**, 82–117 (2013)
4. Brest, J., Maucec, M.S., Boskovic, B.: iL-SHADE: improved L-SHADE algorithm for single objective real-parameter optimization. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 1188–1195 (2016)
5. Crepinsek, M., Liu, S.H., Mernik, M.: Exploration and exploitation in evolutionary algorithms. ACM Comput. Surv. **45**(3), 1–33 (2013)
6. Fodorean, D., Idoumghar, L., N'diaye, A., Bouquain, D., Miraoui, A.: Simulated annealing algorithm for the optimisation of an electrical machine. IET Electr. Power Appl. **6**(9), 735–742 (2012)
7. Fodorean, D., Idoumghar, L., Szabo, L.: Motorization for an electric scooter by using permanent-magnet machines optimized based on a hybrid metaheuristic algorithm. IEEE Trans. Veh. Technol. **62**(1), 39–49 (2013)
8. Gao, W., Yen, G.G., Liu, S.: A cluster-based differential evolution with self-adaptive strategy for multimodal optimization. IEEE Trans. Cybern. **44**(8), 1314–1327 (2014)
9. Halder, U., Das, S., Maity, D.: A cluster-based differential evolution algorithm with external archive for optimization in dynamic environments. IEEE Trans. Cybern. **43**(3), 881–897 (2013)
10. Han, Z.H., Zhang, K.S.: Surrogate-based optimization. In: Real-World Applications of Genetic Algorithms. InTech (2012)
11. Hansen, N., Muller, S.D., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol. Comput. **11**(1), 1–18 (2003)
12. Kazimipour, B., Li, X., Qin, A.K.: A review of population initialization techniques for evolutionary algorithms. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 2585–2592 (2014)
13. Liang, J., Qu, B., Suganthan, P., Chen, Q.: Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization. Technical report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China, Nanyang Technological University, Singapore (2014)

14. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297. University of California Press, Berkeley, California (1967)
15. Pence, I., Cesmeli, M.S., Senel, F.A., Cetisli, B.: A new unconstrained global optimization method based on clustering and parabolic approximation. Expert Syst. Appl. **55**, 493–507 (2016)
16. Storn, R., Price, K.: Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. **11**(4), 341–359 (1997)
17. Tanabe, R., Fukunaga, A.: Evaluating the performance of shade on CEC 2013 benchmark problems. In: 2013 IEEE Congress on Evolutionary Computation, pp. 1952–1959 (2013)
18. Tanabe, R., Fukunaga, A.S.: Improving the search performance of SHADE using linear population size reduction. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 1658–1665 (2014)
19. Yang, X.S., Deb, S.: Cuckoo search via Lévy flights. In: 2009 World Congress on Nature and Biologically Inspired Computing (NaBIC). IEEE (2009)
20. Zhang, J., Sanderson, A.: JADE: adaptive differential evolution with optional external archive. IEEE Trans. Evol. Comput. **13**(5), 945–958 (2009)