Iterative hybridization of DE with Local Search for the CEC'2015 Special Session on Large Scale Global Optimization

Daniel Molina
Department of the Computer Science
University of Cadiz, Spain
Email: daniel.molina@uca.es

Francisco Herrera
Department of Computer Science
University of Granada, Spain
Email: herrera@decsai.ugr.es

***Abstract*—Continuous optimization is an important research field because many real-world problems from very different domains (biology, engineering, data mining, etc.) can be formulated as the optimization of a continuous function. Into continuous optimization, solving high-dimensional optimization problems, also called large scale optimization, is a difficult challenge by the huge expansion of the domain search with the dimensionality. In this paper we propose a new hybrid algorithm to tackle this type of optimization, combining a DE with a LS method in a iterative way. The sharing of the best solution between these components in combination with a memory making possible a more in-depth search in each one of them, allowing the algorithm to obtain good results. Experiments are carried out using a benchmark designed for large scale optimization, and the proposal is compared with other algorithms, showing that the algorithm is robust, obtaining good results specially in the most difficult functions.**

Keywords*—*Continuous Optimization, Large Scale Global Optimization, Hybridization, Differential Evolution, Memetic Algorithm.

I. Introduction

Continuous optimization is an important research field because many real-world problems from very different domains (engineering, data mining, etc.) can be formulated as the optimization of a continuous function. Evolutionary Algorithms (EAs) [1] are very effective algorithms for solving this type of problems, because they can obtain accurate solutions in complex problems without specific information about them, sometime important in real-world problems.

In recent years, with the appearing of new parallel processing possibilities, new challenges are getting introduced by massive data processing in optimisation, one of that is large scale optimisation. In this type of optimisation the solution space of the problem increases exponentially with the problem size, and a more efficient search strategy is required to explore the promising regions under the temporal restrictions or the limit of solution evaluations.

In recent years, there have appears new hybridization algorithms capable of obtain very good results in continuous optimization, like ICMAES-ILS [5] in continuous optimisation and MOS [2] in large scale optimisation. They alter different algorithms (usually using a self-adaptive mechanism) to obtain good result, using at least one very exploitative component. This type of design can be specially useful in high-dimensional component.

In this work we present a new algorithm for large scale global optimization. The main idea is to combine in a iterative way a DE and a LS method, combining the exploratory component of the DE with the exploitative factor of the LS method. Our proposal is characterized by several features: First, the DE is run in each iteration, the idea is not to apply only the most adapted algorithm, but to apply both of them in a complete way, using a more exploitative component without compromising the diversity into the search. Second, both algorithms share the current best solution for different reasons: The LS method to exploit it, and the DE to use it to guide the search (maintaining at the same time diversity into the population). Finally, inspired in previous algorithm [6] both components keep its state between iterations: DE using the same population, and the LS maintaining the adaptive step size between calls.

This work has the following structure: In Section II, the algorithm is described in detail. In Section III, they are briefly described the benchmark and experimental conditions used, and there are analyzed the results obtained by our proposal in comparison with other algorithm used as references. Finally, in Section IV main conclusions and future work are summarize.

II. Proposed Approach: Iterative Hybrid DE with LS

In this section we describe in detail the algorithm proposed: Iterative Hybrid DE with LS, IHDELS.

This algorithm contains several main concepts:

- It is an iterative algorithm that during the run it applies iteratively a DE and a LS method, exploring all the variables at the same time.
- The population is maintained between the different runs of the DE. However, the current solution is introduced into the population to allow that the improvement method could guide the DE search.
- There are two LS methods. In each iteration it is chosen which LS is going to be applied considering the improvement obtained by the previous one.
- The LS is not always applied to the best solution. When a solution is detected as a local optima, the LS is applied to different solution of the population.

978-1-4799-7492-4/15/$31.00

In Algorithm 1 we can observe the complete scheme. In the following, we are going to describe more deeply the different components of the algorithm.

Algorithm 1 General Scheme

1: $population \leftarrow random(dim, popsize)$
2: $initial\_solution \leftarrow (upper + lower)/2$
3: $current\_best \leftarrow LS(initial\_solution)$
4: Define probabilities $prob[LS] \leftarrow \frac{1}{|LS|}$.
5: $best\_solution \leftarrow current\_best$
6: **while** $totalevals < maxevals$ **do**
7: $current\_best \leftarrow SaDE(population, current\_best)$
8: $previous \leftarrow current\_best.fitness$
9: $improvement \leftarrow previous - current\_best.fitness$
10: Select LS using a probability $prob[LS]$.
11: $current\_best \leftarrow LS(population, current\_best)$
12: $improvement[LS] \leftarrow improvement[LS] + 1$
13: $countLS \leftarrow countLS + 1$
14: **if** $countLS = Freq_{LS}$ **then**
15: Update probabilities of LS using Eq. 1
16: $countLs \leftarrow 0$
17: **end if**
18: **if** $better(current\_best, best\_solution)$ **then**
19: $current\_best \leftarrow best\_solution.$
20: **end if**
21: **if** Must be restarted **then**
22: $current\_best \leftarrow random(population)$
23: **end if**
24: **end while**

A. Differential Evolution

In order to explore the domain search, a DE is used. The DE used is the SaDE [9] algorithm. SaDE is adequate for our exploratory component by several reasons:

- It has several crossover operators that introduce several diversity levels. In the most exploitative crossover methods, the best current solution focuses the search but there are other crossovers that explore without taking in account the best solution. This is interesting for us, because it introduces quickly the improvement obtained by the LS method but maintaining at the same time diversity into the population.
- It has a self-adaptive behaviour and it does not require additional parameter. The only required parameter is the population size.

This algorithm is well-known by its good results, and it was also used in other hybrid algorithms like MOS [2], but instead of be used only a certain number of times, in our proposal the DE is run in each iteration, and maintaining the same population. In our experiments the DE is continuously improving the current best solution.

B. Local Search Application

As the component of LS method, several methods are used. In each iteration one of them is chosen using a probability value that depends on the previous $Freq_{LS}$ iterations. Initially, each LS has the same probability of been chosen, and for each iteration it is stored for each one of them the improvement in fitness obtained by the LS. After $Freq_{LS}$ iteration, the probability of choosing each LS Method is obtained by Equation 1.

$$P_{LS_M} = \frac{I_{LS_M}}{\sum_{m \in LS} I_{LS_m}} \quad (1)$$

Where I_{LS_M} is defined as

$$I_{LS_M} = \sum_{i=1}^{Freq_{LS}} \text{Improvement obtained by } LS_M$$

The idea is to apply more the LS method that gives the best results, to adapt the LS to the function to obtain best results [8].

Although this model can be used with any combination of LS method, only two are used: The first one is the L-BFGS-B [7], that uses an approximation of the gradient (using only the fitness) to improve the search; and MTS LS-1 algorithm [10], specially designed for large scale optimization. These two LS methods can complement each other, because they are very different.

The final LS parameters values of each application of MTS LS-1 are used as the initial LS parameter values when it is applied again to the same solution, making equivalent applying several times the LS to the same solution than applying it once with a greater intensity.

C. Restart Mechanism

An interesting element of the proposal is the restart mechanism. When it is decided that the solution cannot be improved because the current best solution is stuck in a local optimum, a new current solution is selected to be improved by the LS. Instead of selecting randomly a solution, the algorithm choses randomly a solution of the population used by the DE. The idea is to chose a relative good solution in the population but still not improved by the LS method.

The problem with the restart mechanism is to decide when it should be used. In the proposal the restart mechanism was only applied when during a complete iteration the current best result could not be improved (considering both the DE algorithm and the LS method).

III. Experimental Results

Experiments are carried out with the benchmark functions and experimental conditions indicated for the Special Session on Large Scale Global Optimisation [4]. This benchmark is composed by 15 continuous optimization functions for dimension 1000 and with different degrees of separability, from completely separable functions to fully non-separable functions:

- Fully separable functions: $f_1 - f_3$.
- Partially separable functions: with a separable subcomponent ($f_4 - f_7$) and without separable subcomponents ($f_8 - f_{11}$).
- Overlapping functions: $f_{12} - f_{14}$.
- Non-separable functions: f_{15}.

A complete information can be found in [4].

The algorithm is run 25 times for each function until the maximum fitness evaluation, FEs, is achieved. For every function, MaxFEs = 3.0e+06. Also, the current best fitness obtained is shown for different FEs: 1.2e+05, 3.0e+05, and 6.0e+06.

The parameters used in the proposal are indicated in Table I. As we can observe, the number of parameters is very reduced. It is maintained the same ratio of effort invested in each component, and they are run during 50000 evaluations each (thus each one of them is run 30 times). These parameters have not tuned, thus it is possible that the algorithm could obtain better results with other parameter values. It should be studied as future work.

TABLE I: Parameters values used in IHDELS

Parameter	Description	value
DE popsize	population size	100
FE_DE	Evaluations for each DE run	50000
FE_LS	Evaluations for each LS run	50000
Freq_{LS}	Frequency of updating the probabilities of each LS	10
MTS initial size	Initial step size for MTS	20

In Table II are shown the results obtained for IHDELS and in Table III we compare the results obtained against the obtained by the reference algorithm, DE-CC-CG [12]. From Table III we can observe that our algorithm have no particular good results in full-separable problems, in $f_2 - f_3$, in comparisons with DE-CC-CG. This is not surprising, considering that DE-CC-CG, like other algorithms in the comparisons are specially designed for large scale dimensionality, exploring a group of variable at the same time. This partial exploration, that our proposal does not do, produces very good solutions in separable functions. On the contrary, for overlapping function and non-separable function ($f_{12} - f_{15}$), IHDELS is clearly superior.

In comparisons with more modern algorithm, in Tables IV and V we compare our algorithm in median and best with other more modern algorithms proposed in previous editions of the same competitions: SACC [11] and MOS [2]. In the first table we can observe that, comparing with SACC, our algorithm has better results in 8 functions, and the difference is very relevant in many of them, like f_4,f_7, f_{11} or f_{14}. On the contrary, the results of our algorithm are still worse than obtained with MOS, but we have to consider that MOS is possible the state-of-the-art in Large Scale Global Optimization [3]. It is remarkable that there are functions for which our algorithm obtain the best results (in median and best) of all compared algorithms: f_8, f_{11} or f_{14}.

TABLE IV: Median Results with FEs=3e+06

	IHDELS	DE-CC-CG	SACC	MOS
F1	4.80e-29	2.00e-13	0.00e+00	0.00e+00
F2	1.27e+03	1.03e+03	5.71e+02	8.36e+02
F3	2.00e+01	2.85e-10	1.21e+00	9.10e-13
F4	3.09e+08	2.12e+10	3.66e+10	1.56e+08
F5	9.68e+06	7.28e+14	6.95e+06	6.79e+06
F6	1.03e+06	6.08e+04	2.07e+05	1.39e+05
F7	3.18e+04	4.27e+08	1.58e+07	1.62e+04
F8	1.36e+12	3.88e+14	9.86e+14	8.08e+12
F9	7.12e+08	4.17e+08	5.77e+08	3.87e+08
F10	9.19e+07	1.19e+07	2.11e+07	1.18e+06
F11	9.87e+06	1.60e+11	5.30e+08	4.48e+07
F12	5.16e+02	1.03e+03	8.74e+02	2.46e+02
F13	4.02e+06	3.36e+10	1.51e+09	3.30e+06
F14	1.48e+07	6.27e+11	7.34e+09	2.42e+07
F15	3.13e+06	6.01e+07	1.88e+06	2.38e+06

TABLE V: Best Results with FEs=3e+06

	IHDELS	DECC-CG	SACC	MOS
F1	0.00e+00	1.75e-13	0.00e+00	0.00e+00
F2	1.27e+03	9.90e+02	2.88e+02	7.40e+02
F3	2.00e+01	2.63e-10	9.24e-14	8.20e-13
F4	1.03e+08	7.58e+09	8.48e+09	1.10e+08
F5	5.88e+06	7.28e+14	3.36e+06	5.25e+06
F6	1.00e+06	6.96e-08	1.57e+05	1.95e+01
F7	1.33e+04	1.96e+08	1.72e+06	3.49e+03
F8	5.36e+10	1.43e+14	1.47e+14	3.26e+12
F9	3.27e+08	2.20e+08	2.29e+08	2.63e+08
F10	9.05e+07	9.29e+04	1.38e+07	5.92e+02
F11	5.59e+06	4.68e+10	8.12e+07	2.06e+07
F12	2.63e-13	9.80e+02	2.43e+02	2.22e-01
F13	1.90e+06	2.09e+10	6.72e+08	1.52e+06
F14	9.41e+06	1.91e+11	8.21e+07	1.54e+07
F15	1.41e+06	4.63e+07	1.26e+06	2.03e+06

IV. Conclusions

In this work we have proposed a new algorithm for large scale optimization by the hybridization of a DE with a LS method iteratively. The algorithm combines the global exploration of the DE with the exploitative factor of the LS to continuously improve the results.

We have tested the proposal using the benchmark proposed by the Special Session on Large Scale Global Optimization [4], and compared with several algorithms, giving us as result that our algorithm is very competitive against the majority of algorithms used in the comparative, in particular for more complex functions. In future work, we will update the algorithm to increase its effectiveness in more separable functions.

Acknowledgment

This work was supported by Research National Projects TIN2012-37930-C02-01, TIN2013-47210-P and Research Regional Projects P08-TIC-04173, P12-TIC-2958.

TABLE II: Experimental Results with IHDELS

		F1	F2	F3	F4	F5	F6	F7	F8
FEs = 1.2e+05	best	3.50e-04	1.77e+03	2.00e+01	7.96e+09	8.83e+06	1.03e+06	1.25e+08	6.31e+13
	median	1.29e+01	1.77e+03	2.00e+01	1.77e+10	1.12e+07	1.05e+06	2.86e+08	1.74e+14
	Worst	1.48e+06	1.80e+04	2.04e+01	4.14e+10	1.76e+07	1.06e+06	5.61e+08	5.31e+14
	mean	2.01e+05	4.99e+03	2.01e+01	2.15e+10	1.24e+07	1.05e+06	3.09e+08	2.09e+14
	std	4.03e+05	6.79e+03	1.54e-01	8.96e+09	2.88e+06	9.42e+03	1.14e+08	1.39e+14
FEs = 6.0e+05	Best	7.42e-29	1.26e+03	2.00e+01	1.53e+09	6.07e+06	1.02e+06	2.92e+06	3.12e+12
	Median	1.81e-23	1.27e+03	2.00e+01	2.24e+09	1.02e+07	1.04e+06	6.90e+06	1.70e+13
	Worst	7.05e+03	1.56e+04	2.04e+01	4.20e+09	1.48e+07	1.05e+06	3.12e+07	6.21e+13
	Mean	4.70e+02	2.78e+03	2.01e+01	2.36e+09	1.02e+07	1.03e+06	1.10e+07	2.22e+13
	Std	1.82e+03	4.53e+03	1.08e-01	6.64e+08	2.18e+06	1.39e+04	8.43e+06	1.49e+13
FEs = 3.0e+06	Best	0.00e+00	1.27e+03	2.00e+01	1.03e+08	5.88e+06	1.00e+06	1.33e+04	5.36e+10
	Median	4.80e-29	1.27e+03	2.00e+01	3.09e+08	9.68e+06	1.03e+06	3.18e+04	1.36e+12
	Worst	3.94e-27	1.40e+03	2.03e+01	5.46e+08	1.32e+07	1.05e+06	8.03e+04	2.52e+12
	Mean	4.34e-28	1.32e+03	2.01e+01	3.04e+08	9.59e+06	1.03e+06	3.46e+04	1.36e+12
	Std	1.23e-27	6.98e+01	1.36e-01	1.07e+08	2.03e+06	1.95e+04	1.33e+04	6.85e+11

		F9	F10	F11	F12	F13	F14	F15	
FEs = 1.2e+05	Best	5.02e+08	9.31e+07	2.69e+09	1.49e+03	3.58e+09	2.36e+10	2.75e+07	
	Median	7.34e+08	9.39e+07	7.15e+09	1.82e+03	1.20e+10	6.81e+10	5.95e+07	
	Worst	1.00e+09	9.79e+07	3.65e+10	3.96e+03	2.34e+10	2.73e+11	1.08e+08	
	Mean	7.25e+08	9.43e+07	9.55e+09	2.07e+03	1.06e+10	8.96e+10	5.34e+07	
	Std	1.17e+08	1.34e+06	7.38e+09	6.41e+02	5.74e+09	6.30e+10	2.18e+07	
FEs = 6.0e+05	Best	4.73e+08	9.16e+07	3.41e+08	8.61e+02	1.72e+08	7.87e+07	8.20e+06	
	Median	7.24e+08	9.35e+07	4.55e+08	1.24e+03	7.32e+08	1.53e+08	1.72e+07	
	Worst	8.72e+08	9.45e+07	6.03e+08	2.51e+03	1.39e+09	4.38e+08	2.29e+07	
	Mean	6.96e+08	9.31e+07	4.51e+08	1.44e+03	7.29e+08	1.67e+08	1.49e+07	
	Std	1.14e+08	9.06e+05	6.76e+07	4.74e+02	3.34e+08	8.35e+07	5.88e+06	
FEs = 3.0e+06	Best	3.27e+08	9.05e+07	5.59e+06	2.63e-13	1.90e+06	9.41e+06	1.41e+06	
	Median	7.12e+08	9.19e+07	9.87e+06	5.16e+02	4.02e+06	1.48e+07	3.13e+06	
	Worst	8.44e+08	9.26e+07	2.23e+07	9.11e+02	5.15e+06	2.90e+07	4.58e+06	
	Mean	6.74e+08	9.16e+07	1.07e+07	3.77e+02	3.80e+06	1.58e+07	2.81e+06	
	Std	1.30e+08	9.17e+05	4.12e+06	3.30e+02	9.72e+05	5.11e+06	1.01e+06	

TABLE III: Comparisons of Results against the Reference algorithm (DE-CC-CG) for FEs=3.0e+06

		F1	F2	F3	F4	F5	F6	F7	F8
DECC-CG	Best	1.75e-13	**9.90e+02**	**2.63e-10**	7.58e+09	7.28e+14	**6.96e-08**	1.96e+08	1.43e+14
	Median	2.00e-13	**1.03e+03**	**2.85e-10**	2.12e+10	7.28e+14	**6.08e+04**	4.27e+08	3.88e+14
	Worst	2.45e-13	**1.07e+03**	**3.16e-10**	6.99e+10	7.28e+14	**1.10e+05**	1.78e+09	7.75e+14
	Mean	2.03e-13	**1.03e+03**	**2.87e-10**	2.60e+10	7.28e+14	**4.85e+04**	6.07e+08	4.26e+14
	Std	1.78e-14	2.26e+01	1.38e-11	1.47e+10	1.51e+05	3.98e+04	4.09e+08	1.53e+14
IHDELS	Best	**0.00e+00**	1.27e+03	2.00e+01	**1.03e+08**	**5.88e+06**	1.00e+06	**1.33e+04**	**5.36e+10**
	Median	**4.80e-29**	1.27e+03	2.00e+01	**3.09e+08**	**9.68e+06**	1.03e+06	**3.18e+04**	**1.36e+12**
	Worst	**3.94e-27**	1.40e+03	2.03e+01	**5.46e+08**	**1.32e+07**	1.05e+06	**8.03e+04**	**2.52e+12**
	Mean	**4.34e-28**	1.32e+03	2.01e+01	**3.04e+08**	**9.59e+06**	1.03e+06	**3.46e+04**	**1.36e+12**
	Std	1.23e-27	6.98e+01	1.36e-01	1.07e+08	2.03e+06	1.95e+04	1.33e+04	6.85e+11

		F9	F10	F11	F12	F13	F14	F15	
DECC-CG	Best	**2.20e+08**	**9.29e+04**	4.68e+10	9.80e+02	2.09e+10	1.91e+11	4.63e+07	
	Median	**4.17e+08**	**1.19e+07**	1.60e+11	1.03e+03	3.36e+10	6.27e+11	6.01e+07	
	Worst	**6.55e+08**	**1.73e+07**	7.16e+11	1.20e+03	4.64e+10	1.04e+12	7.15e+07	
	Mean	**4.27e+08**	**1.10e+07**	2.46e+11	1.04e+03	3.42e+10	6.08e+11	6.05e+07	
	Std	9.89e+07	4.00e+06	2.03e+11	5.76e+01	6.41e+09	2.06e+11	6.45e+06	
IHDELS	Best	3.27e+08	9.05e+07	**5.59e+06**	**2.63e-13**	**1.90e+06**	**9.41e+06**	**1.41e+06**	
	Median	7.12e+08	9.19e+07	**9.87e+06**	**5.16e+02**	**4.02e+06**	**1.48e+07**	**3.13e+06**	
	Worst	8.44e+08	9.26e+07	**2.23e+07**	**9.11e+02**	**5.15e+06**	**2.90e+07**	**4.58e+06**	
	Mean	6.74e+08	9.16e+07	**1.07e+07**	**3.77e+02**	**3.80e+06**	**1.58e+07**	**2.81e+06**	
	Std	1.30e+08	9.17e+05	4.12e+06	3.30e+02	9.72e+05	5.11e+06	1.01e+06	

REFERENCES

[1] T Bäck, D B Fogel, and Z Michalewicz, editors. *Handbook of Evolutionary Computation*. IOP Publishing Ltd., Bristol, UK, 1997.

[2] A. LaTorre, S. Muelas, and J.-M. Pena. Large scale global optimization: Experimental results with mos-based hybrid algorithms. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 2742–2749, June 2013.

[3] Antonio LaTorre, Santiago Muelas, and Jos-Mara Pea. A comprehensive comparison of large scale global optimizers. *Information Sciences*, In press, 2015.

[4] X. Li, K. Tang, M. Omidvar, Z. Yang, and K. Qin. Benchmark functions for the cec'2013 special session and competition on large scale global optimization. Technical report, Evolutionary Computation and Machine Learning Group, RMIT University, Australia, 2013.

[5] Tianjun Liao and T. Stutzle. Benchmark results for a simple hybrid algorithm on the cec 2013 benchmark set for real-parameter optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 1938–1944, June 2013.

[6] D. Molina, M. Lozano, and F. Herrera. Ma-sw-chains: Memetic algorithm based on local search chains for large scale continuous global optimization. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8, July 2010.

[7] José Luis Morales and Jorge Nocedal. Remark on algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound constrained optimization. *ACM Trans. Math. Softw.*, 38(1):7:1–7:4, December 2011.

[8] Yew-Soon Ong and A.J. Keane. Meta-lamarckian learning in memetic algorithms. *Evolutionary Computation, IEEE Transactions on*, 8(2):99–110, April 2004.

[9] A.K. Qin, V.L. Huang, and P.N. Suganthan. Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization. *IEEE Transactions on Evolutionary Computation*, 13(2):398–417, April 2009.

[10] Lin-Yu Tseng and Chun Chen. Multiple trajectory search for large scale global optimization. In *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 3052–3059, June 2008.

[11] Fei Wei, Yuping Wang, and Yuanliang Huo. Smoothing and auxiliary functions based cooperative coevolution for global optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 2736–2741, June 2013.

[12] Zhenyu Yang, Ke Tang, and Xin Yao. Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*, 178(15):2985 – 2999, 2008. Nature Inspired Problem-Solving.