VMODE: A HYBRID METAHEURISTIC FOR THE SOLUTION OF LARGE SCALE OPTIMIZATION PROBLEMS

Ernesto Díaz López[1], Amilkar Puris[2], Rafael Rafael Bello[3]
Universidad Central de Las Villas

ABSTRACT
Large scale continuous optimization problems have become increasingly common in real-world problems. The resolutions of these are computationally expensive, so the use of scalable and efficient algorithms is of particular interest. In this paper is proposed a hybrid algorithm, VMODE, which results from the combination of DE algorithm, known for its simplicity and efficiency and VMO, a population-based algorithm with encouraging results in continuous optimization. A comparison among the three algorithms is done using the 15 proposed functions for CEC-2013 (Special Session and Competition on Large-Scale Global Optimization) demonstrating the superiority of the algorithm VMODE.

KEYWORDS: Differential Evolution, Variable Mesh Optimization, Continuous Optimization, Large Scale Global Optimization.

RESUMEN
Los problemas de optimización continua de gran escala se han vuelto cada vez más común en los problemas del mundo real. La resolución de estos son computacionalmente costosos, por lo que el uso de algoritmos escalables y eficientes es de particular interés. En este trabajo se propone un algoritmo híbrido, VMODE, que resulta de la combinación del algoritmo Differential Evolution(DE), conocido por su sencillez y eficiencia y VMO, un algoritmo poblacional con resultados alentadores en la optimización continua. Se realiza una comparación entre los tres algoritmos utilizando las 15 funciones propuestas para CEC-2013 (Sesión Especial y Competición en la Optimización Global a Gran Escala) que demuestra la superioridad del algoritmo VMODE.

1. INTRODUCTION

Optimizing large scale continuous problems is a research area that has gained great importance recently. Many real problems in areas such as Engineering, Bioinformatics, Data mining, etc. can be solved as optimization problems. The resolution of these problems, due to its complexity and difficult to solve by traditional optimization methods, has been treated with the use of Evolutionary Algorithms (EA)[9],[3].

One representative evolutionary algorithm is the Differential Evolution (DE)[14].This algorithm is simple but highly efficient in solving global optimization problems. Its efficiency has been demonstrated in applications of neural networks [7], signal processing[13], electromagnetism[12], design of water distribution networks [18] and many others. It has also proven its efficiency in solving optimization problems with and without constraints [2],[16].

Variable Mesh Optimization (VMO)[11] is a recently proposed population-based metaheuristic for global optimization. It yielded competitive results when compared with outstanding state of-the-art schemes in continuous optimization. VMO features three search operators, one aimed at global exploration and two for local optima exploitation. Thus, VMO seems to be a promising multimodal problem solver.

The complexity and high dimensionality of the real problems to be addressed by the EA has led to its limits. This is the reason why many authors combine different algorithms looking to enhance the best of each method in order to complement each other, knowing this strategy as hybrid metaheuristic. A comprehensive description of these can be seen in [1],[17]. Following this methodology, in this work is decided to combine the VMO and DE algorithms so that the resulting mesh, at each iteration of VMO, serves as the initial population of DE and obtains a population of more quality. With this population VMO begins a new cycle.

The paper is organized as follows: In section II is provided a brief description of the VMO and DE algorithms. In section III is explained the proposed hybrid algorithm VMODE. Section IV contains the experimental results and performance analysis. Finally, conclusions are drawn in section IV.

[1] e-mail: ediaz@uclv.edu.cu
[2] e-mail: apuris@uteq.edu.ec
[3] e-mail: rbellop@uclv.edu.cu

2. BRIEF REVIEW OF VMO AND DE

VMO and DE algorithms that result in the hybrid metaheuristic VMODE are briefly described in this section

2.1. Variable Mesh Optimization (VMO)

VMO is a metaheuristic in which the population is distributed as a mesh. This mesh is composed of *P* nodes $(n_1, n_{2,} \cdots, n_P)$ that represent solutions in the search space. Each node is coded as a vector of *M* floating point numbers, $n_i = \left(v_1^i, v_2^i, \cdots, v_j^i, \cdots, v_M^i\right)$ that represent the solution to the optimization problem. In the search process developed by VMO, two operations are executed: the expansion and contraction processes. During the expansion, new nodes are generated in the direction of local extreme (mesh nodes with better quality in the neighborhoods), the global end (node with better quality obtained in the process) and to the border nodes (nodes nearest and furthest from the center of the search space base on the Euclidean distance). Based on an elitist strategy, nodes are ordered according to their quality (by their fitness value) in ascending order. Cleaning adaptive operator is then applied; each node is compared to its successors eliminating those that do not exceed a threshold. The value of this threshold is calculated as:

$$\varepsilon_j = \begin{cases} \dfrac{range(a_j, b_j)}{4} & if\ c < 0.15\%C \\ \dfrac{range(a_j, b_j)}{8} & if\ 0.15\%C \le c < 0.3\%C \\ \dfrac{range(a_j, b_j)}{16} & if\ 0.3\%C \le c < 0.6\%C \\ \dfrac{range(a_j, b_j)}{50} & if\ 0.6\%C \le c < 0.8\%C \\ \dfrac{range(a_j, b_j)}{100} & if\ c \ge 0.8\%C \end{cases}$$

where C and c denote a maximum number of fitness evaluations allowed and the current number of fitness evaluations. In addition, the range (a_j, b_j) denotes the domain boundaries of each component.
The node generation process at each cycle comprises the following steps:

1. Randomly generate P nodes for the initial mesh.
2. Generate nodes toward the local best.
3. Generate nodes toward the global best.
4. Generate nodes from nodes in the mesh frontier.

The method includes the following parameters:
• Number of nodes in the initial mesh (P).
• Maximum number of mesh nodes in each cycle (T), where 3 • P = T.
• Size of neighborhood (k).
• Stop condition (S).

Algorithm 1 shows the pseudocode of the VMO model presented in [10].

Algorithm 1 VMO's Pseudocode

```
1. begin
2.   Randomly generate P nodes for the initial mesh
3.   Select the global best in the initial mesh
4.   repeat
5.     for each node in initial mesh do
6.       Find its closest k nodes by their spatial locations
7.       Select the best neighbor as per the fitness values
8.       if current node is not the local best then
9.          Generate a new node toward the local best
10.      end if
11.    end for
```

12. **for each** node in initial mesh but the global best **do**
13. Generate a new node toward the global best
14. **end for**
15. Generate nodes from nodes in the mesh frontier (up to T nodes in the total mesh)
16. Sort nodes according to their fitness values
17. Apply the adaptive clearing operator
18. Select P best nodes to build the initial mesh for the next iteration
19. If needed, randomly generate new nodes so as to complete the initial mesh for the next iteration
20. **until** stop criterion (S) is met
21. **end**

2.2 Differential Evolution (DE)

DE is a simple yet powerful evolutionary algorithm for solving continuous optimization problems. This follows the procedure of any evolutionary algorithm. The initial population contains NP individuals who are represented by D-dimensional vectors $x_i, \forall i \in \{1, \dots, NP\}$, where D is the number of decision variables. The initial population is generated randomly according to a uniform distribution.

Mutation: For each target vector $x_{i,G}$ at generation G an associated mutant vector $v_{i,G} = \{v_{i1,G}, v_{i2,G}, \cdots, v_{iD,G}\}, \quad i = 1,2, \cdots, NP$ is generated by using one of the following strategies [4],[10]:

DE/rand/1: $v_{i,G} = x_{i1,G} + F \cdot (x_{i2,G} - x_{i3,G})$

DE/best/1: $v_{i,G} = x_{best,G} + F \cdot (x_{i1,G} - x_{i2,G})$

DE/current to best/1: $v_{i,G} = x_{i,G} + F \cdot (x_{best,G} - x_{i,G}) + F \cdot (x_{i1,G} - x_{i2,G})$

DE/best/2: $v_{i,G} = x_{best,G} + F \cdot (x_{i1,G} - x_{i2,G}) + F \cdot (x_{i3,G} - x_{i4,G})$

DE/rand/2: $v_{i,G} = x_{i1.G} + F \cdot (x_{i2,G} - x_{i3,G}) + F \cdot (x_{i4,G} - x_{i5,G})$

where $i = 1, 2, \dots, NP$ y $i_1, i_2, .. i_5$ are random and mutually different indices in the range $[1, NP]$. $F \in [0, 2]$ is a mutation scale factor. Vector $x_{best,G}$ is the best in the generation G.

Crossover: After the mutation phase, the "binominal" crossover operation is applied to each pair of the generated mutant vector $v_{i,G} = (v_{i1,G}, \dots, v_{iD,G})$ and its corresponding target vector $x_{i,G} = (x_{i1,G}, \dots, x_{iD,G})$. The binomial operation is defined as follows [15]:

$$u_{ij,G} = \begin{cases} v_{ij,G}, & if\ (rand_j(0,1) \le CR\ \ or\ (j = j_{rand}), \\ x_{ij,G} & otherwise\ j = 1, \dots, D \end{cases}$$

CR is a crossover control parameter or factor within the range [0, 1] and j_{rand} is a randomly chosen integer in the range $[1, NP]$ to ensure that the trial vector contains at least one parameter of the mutant vector.

Selection: The fitness value of each trial vector $f(u_{i,G})$ is compared to that of its corresponding target vector $f(x_{i,G})$ in the current population. If the trial vector has smaller or equal fitness value (for minimization problem) than the corresponding target vector, the trial vector will replace the target vector and enter the population of the next generation. The operation is expressed as follows:

$$x_{i,G+1} = \begin{cases} u_{i,G}, & if\ f(u_{i,G}) \le f(x_{i,G}), \\ x_{i,G} & otherwise\ i = 1, \dots, NP \end{cases}$$

The DE algorithm is outlined below:

Algorithm 2 Differential Evolution

1. **begin**
2. Initialize population
3. Evaluate initial population(fitness function)

```
4.  for i=0 to max-iteration do
5.    Select random trial vectors
6.    Create offspring population(mutation, crossover)
7.    Evaluate offspring population
8.    Merge parent and offspring population
9.    if an offspring is better than its parent then
10.     Replace the parent by offspring in the next generation(selection)
11.   end if
12.  end for
13. end
```

3. THE PROPOSED ALGORITHM: VMODE

The VMODE metaheuristic employs VMO as the main core and inserts the DE algorithm in order to enhance the initial mesh of the next iteration. The use of DE was decided to improve the quality of the population at the end of the cleaning process done by VMO. The DE algorithm does not generate a random initial population but takes as its initial population the matrix resulting from the cleaning operation performed by VMO, giving out a population with higher quality individuals whose VMO starts a new iteration. Algorithm 3 shows a schematic of the algorithm.

Algorithm 3 VMODE's pseudocode

```
1. begin
2.  Randomly generate P nodes for the initial mesh
3.  Select the global best in the initial mesh
4.  repeat
5.    for each node in initial mesh do
6.      Find its closest k nodes by their spatial locations
7.      Select the best neighbor as per the fitness values
8.      if current node is not the local best then
9.        Generate a new node toward the local best
10.     end if
11.   end for
12.  for each node in initial mesh but the global best do
13.     Generate a new node toward the global best
14.  end for
15.  Generate nodes from nodes in the mesh frontier
       (up to T nodes in the total mesh)
16.  Sort nodes according to their fitness values
17.  Apply the adaptive clearing operator
18.  Select P best nodes to build the initial mesh for the
     next iteration
19.  DE call using VMO population
20. until stop criterion (S) is met
21. end
```

4. EXPERIMENTAL STUDIES

4.1. Experimental Setup

For the comparison of VMODE with VMO and DE, a recently proposed benchmark test suite for the CEC-2013 Special Session and Competition on Large Scale Global Optimization [8] has been utilized. This benchmark has been designed with the aim of providing a suitable evaluation platform for testing and comparing large-scale global optimization algorithms.

These are divided into 4 categories:

1. Fully-separable functions (f_1, f_2, f_3)
2. Two types of partially separable functions:

   a. Partially separable functions with a set of non-separable subcomponents and one fully-separable subcomponents;. (f_4, f_5, f_6, f_7)
   b. Partially separable functions with only a set of non-separable subcomponents and no fully-separable subcomponent. (f_8, f_9, f_{10}, f_{11})
3. Functions with overlapping subcomponents: the subcomponents of these functions have some degree of overlap with its neighboring subcomponents. There are two types of overlapping functions:
   a. Overlapping functions with conforming subcomponents. (f_{12}, f_{13})
   b. Overlapping functions with conflicting subcomponents.(f_{14})
4. Fully-non-separable functions. (f_{15})

To perform the experiments it followed the same methodology used in CEC-2013 Special Session and Competition on Large Scale Global Optimization[8]; each algorithm performs 25 runs for each of the functions with the problem dimension D = 1000 and a maximum of evaluations of the objective function, Max_FE = $3x10^6$.

4.2. VMO parameters

- Initial population size(P), *P =100*
- Maximum number of individuals (T) after the mesh expansion. T = *3 * P = 300*
- The number of mesh nodes in the neighborhood(k), *k=3*

4.3. DE parameters

- F = 0.85
- CR = 0.5
- Mutation strategy *DE/best/1*

To achieve a balance between both algorithms, the number of iterations of DE has been limited to 20, taking into account the number of evaluations of the objective function to be performed therein.

4.4. Results Analysis

The values obtained by the algorithms for each of the 15 functions after running the 25 runs with dimension 1000 is shown in Table 1. For each one, the average (AVG) and Standard Deviation (Dev_Std) are shown. There are noted in bold the best results for each function.
The values show that the VMODE algorithm outperforms the others in all functions except f_6 and f_{10}, which is only beaten by the DE algorithm. Both functions are derived from the Ackley function.

TABLE 1. Experimental results for dimension D = 1000 and 25 runs. Best results in bold.

	DE AVG Dev Std	VMO AVG Dev Std	VMODE AVG Dev Std
f1	3.51E+06	5.98E+01	**1.29E-03**
	3.07E+06	6.53E+01	**1.27E-03**
f2	8.70E+03	5.12E+01	**5.53E+03**
	6.17E+02	7.13E+00	**3.96E+02**
f3	1.02E+01	6.18E+00	**3.70E-04**
	1.11E+00	1.69E-01	**1.37E-04**
f4	7.76E+11	2.83E+11	**9.13E+09**
	1.84E+11	3.62E+10	**3.73E+09**
f5	7.28E+14	8.27E+14	**7.28E+14**
	0.00E+00	3.46E+13	**0.00E+00**
f6	**3.96E+03**	4.48E+05	2.15E+05
	2.93E+03	4.34E+04	4.33E+04
f7	6.22E+09	5.27E+08	**3.43E+06**
	1.56E+09	2.19E+08	**2.71E+05**

f8	8.78E+15	1.58E+15	**6.94E+13**
	3.57E+15	4.58E+14	**7.22E+13**
f9	7.25E+08	1.22E+09	**7.68E+08**
	2.67E+07	2.06E+08	**1.03E+08**
f10	**2.78E+05**	1.24E+07	9.10E+06
	4.98E+05	7.35E+06	3.05E+06
f11	1.07E+11	6.01E+09	**1.66E+08**
	6.32E+10	1.05E+09	**3.80E+07**
f12	4.74E+07	1.40E+05	**4.45E+03**
	4.72E+07	7.20E+04	**4.61E+03**
f13	8.15E+10	1.24E+10	**2.46E+07**
	1.74E+10	2.89E+09	**3.28E+06**
f14	6.81E+11	7.60E+10	**9.54E+07**
	3.21E+11	3.77E+10	**1.13E+07**
f15	4.11E+08	2.36E+07	**1.10E+07**
	1.48E+09	3.66E+06	**1.65E+06**

4.5. Statistical Analysis

To make a comparative analysis among the algorithms, they are initially sorted according to their average ranking. The calculation was applied separately to each of the categories in which the functions are divided [8] As shown in Table 2 the best ranking algorithm for all categories was the VMODE. It is further appreciated that with the increasing complexity of the functions VMODE also increases quality.

TABLE 2. Average rankings of algorithms.

Functions	DE	VMO	VMODE
f1-f3	3.00	1.66	1.33
f4-f7	2.13	2.50	1.38
f8-f14	2.42	2.28	1.28
f15	3.00	2.00	1.00
all	2.50	2.19	1.29

With the intention of determining if significant differences exist between the results of the proposed algorithm VMODE for a dimension of 1000 and VMO and DE algorithms, a Holm's test was applied. Table 3 shows the results obtained by employing a Holm's test with VMODE as a control algorithm. In both cases the p-value is smaller than α / i so that the hypothesis of equality is rejected (R).

TABLE 3. Holm test results

I	Algorithm	*p*	*α/i*	Hypothesis
2	DE	0.001	0.025	R
1	VMO	0.013	0.05	R

To complement the above multiple comparisons of all pairs of algorithms there were performed the Holm's test and Shaffer's test. These values were obtained by following the procedure in [5],[6] .Table 4 shows the results. Both tests agree on the differences between the VMO and DE algorithms with VMODE. The table also shows that the VMO and DE algorithms have no significant difference and in the comparison among them the hypothesis is accepted (A).

TABLE 4. Values of *p* for Holm y Shaffer tests

I	Algorithm	*P*	*Holm*	Shaffer	Hypothesis
3	DE vs VMODE	0.001015	0.017	0.017	R
2	VMO vs. VMODE	0.013710	0.025	0.05	R

1	DE vs. VMO	0.411313	0.05	0.05	A

In general, it can be confirmed that the VMODE algorithm achieves better results and that behavior is most noticeable in the more complex functions.

3 CONCLUSIONS

It was introduced the VMODE algorithm, a hybrid metaheuristic that combines the VMO and DE algorithms. The benchmark functions of the CEC-2013 Special Session and Competition on Large Scale Global Optimization [16] were used for statistical tests that permitted the comparison among the algorithms.
The statistical results exposed the highest performance of the VMODE in most functions, especially in the more complex ones.
The study of new variants of this algorithm, as well as a parallel version, represents our directions for future work.

RECEIVED SEPTEMBER, 2014
REVISED JUNE, 2015

REFERENCES

[1] BLUM, C., PUCHINGER, J., RAIDL, G. R. & ROLI, A. (2011):Hybrid Metaheuristics in Combinatorial Optimization: A Survey. ***Appl. Soft Comput.*** 11**,** 4135–4151.

[2] BREST, J., GREINER, S., BOŠKOVIĆ, B., MERNIK, M. & ŽUMER, V. (2006):Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. ***IEEE Trans Evol Comput*** 10(6)**,** 646–657.

[3] DUQUE, T. S. P., SASTRY, K., DELBEM, A. C. B. & GOLDBERG, D. E. (2007): Evolutionary Algorithm for Large Scale Problems. in ***Intell. Syst. Des. Appl***. 819–822. doi:10.1109/ISDA.2007.114

[4] FEOKTISTOV, V. (2006):*Differential evolution*. Springer US.

[5] GARCÍA, S., FERNÁNDEZ, A., LUENGO, J. & HERRERA, F. (2010):Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. ***Inf. Sci. (Ny).*** 180**,** 2044–2064.

[6] GARCÍA, S., MOLINA, D., LOZANO, M. & HERRERA, F. (2007):Un estudio experimental sobre el uso de test no paramétricos para analizar el comportamiento de los algoritmos evolutivos en problemas de optimización. ***Proc. 2007 Congr. Español sobre Metaheurísticas, Algoritm. Evol. y Bioinspirados, MAEB 2007*** 275–285.

[7] ILONEN, J., KAMARAINEN, J.-K. & LAMPINEN, J. (2003):Differential Evolution Training Algorithm for Feed-Forward Neural Networks. ***Neural Process. Lett.*** 17**,** 93–105.

[8] LI, X. *et al.* (2013):Benchmark Functions for the CEC'2013 Special Session and Competition on Large-Scale Global Optimization. ***Gene*** 1–21. at <http://goanna.cs.rmit.edu.au/~xiaodong/cec13-lsgo/competition/cec2013-lsgo-benchmark-tech-report.pdf>

[9] LI, X. and YAO, X. (2012):Cooperatively coevolving particle swarms for large scale optimization. ***Evol. Comput. IEEE Trans.*** 16**,** 210–224.

[10] PRICE, K., STORN, R. M. & LAMPINEN, J. A. (2005):Diferential Evolution : A Practical Approach to Global Optimization. ***Nat. Comput. Ser.*** 519–524.

[11] PURIS, A., BELLO, R., MOLINA, D. & HERRERA, F. (2011): Variable mesh optimization for continuous optimization problems. ***Soft Comput.*** 16**,** 511–525.

[12] ROCCA, P., OLIVERI, G. & MASSA, A. (2011):Differential Evolution as Applied to Electromagnetics. ***Antennas Propag. Mag.*** *IEEE* 53**,** 38–49.

[13] STORN, R. (1995):Differential Evolution Design of an IIR-Filter with Requirements for Magnitude and Group Delay. in ***Proc. IEEE Int. Conf. Evol. Comput.*** 1–15.

[14] STORN, R. & PRICE, K. (1997):Differential evolution-A simple and efficient adaptive scheme for global optimization over continuous spaces. ***J. Glob. Optim.*** 11**,** 341–359.

[15] STORN, R., PRICE, K., RAINER, S. & KENNETH, P. (1997):Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. ***J. Glob. Optim.*** 11**,** 341–359.

[16] TAKAHAMA, T., & SAKAI, S. (2005): Constrained optimization by ε constrained particle swarm optimizer with ε-level control. ***Soft Computing as Transdisciplinary Science and Technology*** 1019-1029. Springer Berlin Heidelberg

[17] TALBI, E., PUBLISHERS, K. A. & TALBI, G. A (2002):Taxonomy of Hybrid Metaheuristics. ***J. heuristics*** 8**,** 541–564.

[18] VASAN, A. & SIMONOVIC, S. (2010):Optimization of Water Distribution Network Design Using Differential Evolution. ***J. Water Resour. Plan. Manag.*** 136**,** 279–287.