10th Brazilian Congress on Computational Intelligence (CBIC'2011), November 8 to 11, 2011, Fortaleza, Ceará Brazil

MULTIMODAL OPTIMIZATION AS A CLUSTERING TASK: GENERAL FORMULATION AND APPLICATION IN THE CONTEXT OF PARTICLE SWARM

Paulo S. Prampero[1,2], Levy Boccato[1], Romis Attux[1]

1 - Department of Computer Engineering and Industrial Automation, FEEC – University of Campinas (Unicamp) Campinas, Brazil [1]

2 - Federal Institute of Science, Technology and Education of São Paulo – Campus Salto, Salto, Brazil [2]

prampero@cefetsp.br, lboccato@dca.fee.unicamp.br, attux@dca.fee.unicamp.br

Abstract – In this paper, we propose a formulation of the optimization problem as a clustering task, which has the potential of allowing an interchange between approaches belonging to both fields of research and may also allow a straightforward treatment of multimodal scenarios. The proposal is tested using the classical k-means algorithm and a particle swarm algorithm as an accessory tool, and the results indicate the relevance of the clustering-based formulation in terms of search space exploration capability.

Keywords – clustering-based optimization, multimodal optimization, magnetic particle swarm, k-means algorithm.

1 Introduction

The problem of iterative optimization can be understood as one of consecutively sampling, in a controlled way, a cost function in order that one or more satisfactory maximizing or minimizing solutions is obtained. This problem can be addressed with different degrees of information concerning the cost function: classical nonlinear optimization methods typically require knowledge of its derivatives, whereas many metaheuristics simply demand function evaluations.

An interesting perspective emerges when an optimization problem is formulated as one resulting from an underlying (abstract) clustering problem. In simple terms, this would amount to considering the cost function to be maximized as representing a data probability density, thereby allowing a clustering approach to be employed. The chosen clustering method, ideally, would be capable of finding multiple centers, i.e., the position of the multiple peaks of the probability density, which would correspond to finding multiple optima of the focused cost function. Thus, the inherently multimodal character of the clustering process can be duly taken advantage of. Notice that this formulation is, in essence, completely distinct from the application of clustering methods to the control of a population of solutions associated with a search tool, as proposed, for instance, in [3].

Although the outlined idea can be implemented with the aid of any clustering method, we will test it, in this work, using the classical and structurally simple k-means algorithm. The clustering-based approach can be used as a standalone search method, but, in this work, we will preferentially treat it as an additional stage within the *modus operandi* of another technique, the recently proposed magnetic particle swarm algorithm. A number of tests within representative scenarios will be carried out, and the effectiveness of the clustering-based formulation will be analyzed.

The paper is structured as follows. In Section 2, we describe the fundamentals of the clustering problem, as well as the k-means algorithm. Additionally, we introduce the basic formulation of an optimization problem through a clustering perspective. Section 3 presents the magnetic particle swarm with k-means, whereas, Section 4 brings the obtained results. Finally, Section 5 concludes the work by presenting our main observations and future perspectives.

2 Clustering and Optimization

Clustering can be safely regarded as a most relevant problem within unsupervised learning and statistical data analysis. Basically, clustering is the classification of unlabelled data into groups, called clusters, based on a similarity criterion. This task of identifying classes of similar observations from the whole collection of data is useful in a myriad of contexts, like data mining, information retrieval, image segmentation and pattern classification [10] [11].

One of the simplest algorithms to solve the clustering problem is the so-called k-means [10] [12]. It starts by defining k points as cluster centers, or centroids, in the input space. Then, each input pattern is assigned to the cluster whose center it is closest to. Once all patterns have been assigned to clusters, the position of each center is updated taking into account the cluster current membership. These two basic steps – assignment and center relocation – are repeated until a convergence criterion is met (e.g., there is no reassignment of any pattern from one cluster to another).

It is possible to observe that, in order that k-means be properly applied, some important aspects must be taken into account: 1) the representation of a given datum or pattern: in general, it consists of a vector of d measures $\boldsymbol{x} = (x_1 \ldots x_d)$; 2) the similarity

measure, which is strongly related to the pattern representation. 3) The number of centers k, which also establishes the number of clusters (classes) in which the observations will be grouped.

In the case of real-valued data, the similarity between a pattern $\boldsymbol{x}$, and a particular center $\boldsymbol{c_i}$. $i = 1,\ldots, k$, is frequently computed using the Euclidean distance. Moreover, each center $\boldsymbol{c_i}$ is updated according to the following expression:

$$\boldsymbol{c_i} = {}^{1}/_{|Cluster_i|} \sum_{\boldsymbol{x} \in Cluster_i} \boldsymbol{x}, \tag{1}$$

where $Cluster_i$ is the set of the patterns $\boldsymbol{x}$ assigned to the i-th center and $|A|$ represents the cardinality of set A. Notice that each center actually corresponds to the centroid of the cluster, or, in other words, to the mean vector of that cluster (hence the name *k-means*).

The k-means algorithm is very popular, as it is relatively easy to implement and may be applied even to large data sets. However, the method has a major drawback: its convergence strongly depends on the initial condition, i.e., starting with different sets of centers may lead to a distinct outcome. Another critical aspect is that an inappropriate choice of the number of clusters k may yield poor results. Nonetheless, this method has been successfully employed in various domains, like computer vision and geostatistics [4] [5]. Additionally, several methods have drawn inspiration from the original k-means algorithm, among which we mention the fuzzy c-means clustering [6] [7]. Another interesting application of k-means, under the guise of the Lloyd algorithm, emerges in the context of the rate distortion theory, more specifically, in the bit allocation problem [8] [9].

In this work, we are particularly interested in a version of the k-means algorithm adapted to perform online clustering, which requires that the clusters be adapted as the data are collected [10]. In this case, for each new pattern $\boldsymbol{x}$, the closest center $\boldsymbol{c_i}$ is identified, and then updated as follows:

$$\boldsymbol{c_i}(n+1) = \boldsymbol{c_i}(n) + \mu(\boldsymbol{x} - \boldsymbol{c_i}(n)), \tag{2}$$

where μ, often called learning rate, defines the amplitude of the movement toward the pattern $\boldsymbol{x}$. Note that the idea of adjusting the center to be more akin to the pattern establishes a connection with the self-organizing maps and the competitive learning framework [13].

Given the initial position of the centers, the k-means algorithm tends to spread them over the regions of the input space occupied by the samples $\boldsymbol{s_i}$. Thus, the final configuration of the center set depends on the concentration of data along the space. In fact, it is possible to observe that, the more crowded a certain portion of the input space is, the higher becomes its attraction force, so that the closest center will constantly move in its direction. Conversely, if the number of samples in a region is small, only in a few instances the closest center will move toward this region. Another relevant observation is that the clusters associated with crowded regions have a smaller radius, i.e., their centers represent a reduced area of the input space, whereas those associated with regions presenting a small concentration of data have a larger radius, which means that their centers represent a wider area of the space. The online version of the k-means algorithm is used in this work to test the proposed approach, as described in the next section.

2.1 Optimization as a Clustering Task

Suppose we want to find the minima or maxima of a specific function f($\boldsymbol{x}$). Apparently, this optimization problem has no direct connection with the clustering problem described in the previous section, and, therefore, the k-means algorithm should not be able to perform this task. However, one specific feature of this technique is particularly interesting in the context of optimization: as emphasized in Section 2, the centers are spread according to the data concentration, occupying different regions of the input space. In this sense, we can say that k-means creates and preserves a certain degree of diversity within the repertoire of the centers. This evidence encourages the application of k-means as part of a multimodal optimization process.

Now, consider that we draw a set of samples $\boldsymbol{s_i}$ from the search space associated with the function f($\boldsymbol{x}$), and the corresponding values f($\boldsymbol{s_i}$), $i = 1,\ldots,N_p$, are available. In this case, it is possible to employ the k-means algorithm. Unfortunately, the straightforward application of k-means considering the samples of the search space will probably allocate the centers in regions which are not associated with the global optimum of the function, since the movement of the centers depends solely on the position of the samples, not on their quality.

It is important to keep in mind that, in a clustering task, the density or concentration of points controls center adaptation, while, in an optimization problem, the value of the function in each point of the space is the information we use to guide the search for the global optimum. Hence, aiming to obtain a repertoire of centers occupying the best regions of the search space in terms of the value of the function f($\boldsymbol{x}$), ideally including the region associated with the global optimum, the set of samples $\boldsymbol{s_i}$ should be such that a significant number of samples lie in the most promising regions of the search space, and only a small number of samples are taken from the other portions of the search space. In other words, the density of the samples should reflect the "quality of the function". However, we do not know *a priori* which regions of the search space are promising or not, otherwise, the original optimization problem would be easily solved.

In this work, we propose a strategy to circumvent this problem by including the information about sample cost or fitness in the process of center adaptation. The idea is to emulate the effect of the density of points in a particular region of the search

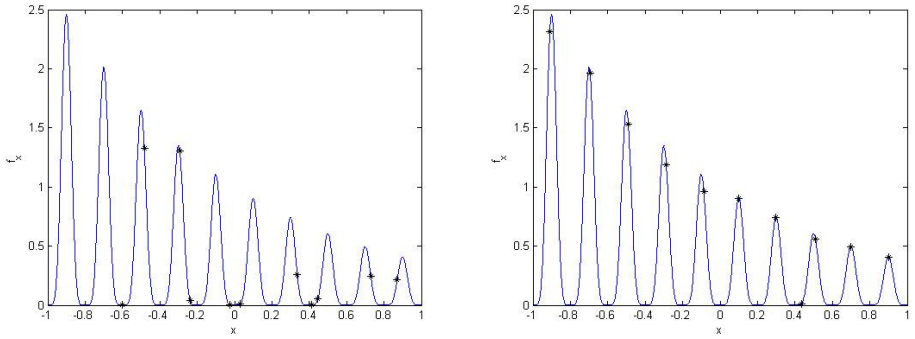
space through the use of a step-size proportional to the quality of that region. Let $\boldsymbol{s}$ and f(**s**) denote the current sample and its fitness value, respectively. Now, the closest center $\boldsymbol{c_i}$ is updated according to the following expression:

$$\boldsymbol{c_i}(n+1) = \boldsymbol{c_i}(n) + \alpha.abs(f(\boldsymbol{s}))\big(\boldsymbol{s} - \boldsymbol{c_i}(n)\big), \tag{3}$$

where α is a scaling factor, and *abs(.)* denotes the absolute value.

As remarked in the previous section, the centers are attracted to determined regions of the input space depending on their data concentration: the more samples lie in a specific region, the stronger is the attraction experienced by the centers. Employing the fitness value of the sample as the step size, as displayed in Equation (3), a similar behavior is observed: the higher the fitness, the bigger is the step taken by the closest center toward it, which emulates steps frequently given in the direction of a crowded region of the space. Thus, if the sample lies in a poor region of the search space, the closest individual will suffer a small displacement, which means that it is not encouraged to explore that region. On the other hand, if the sample lies in a promising region, the closest individual will move significantly in its direction.

This modified version of k-means works similarly to that presented in Section 2 – the only difference refers to the way the position of the closest center is updated. In order to illustrate the behavior of the proposed method, Figure 1 displays the initial and the final repertoire of centers considering a multimodal one-dimensional function and using 1000 samples of the search space.



(a) **(b)**

Figure 1 – Initial (a) and final configuration of the centers (b).

As we can observe, the individuals have been spread along all the peaks of the function, which leads to the desired performance in terms of diversity in the final population. On the other hand, it is important remark that this method does not perform local refinement, which means that the centers are not exactly positioned in the top point of each region. Additionally, the use of the k-means may cause a strong dependence with respect to the initial condition. However, as the approach will be used here together with another search method, the Magnetic Particle Optimization (MPSO) algorithm, the effect of these drawbacks should be significantly attenuated.

3 Magnetic Particle Swarm Optimization with K-means

The Magnetic Particle Swarm Optimization (MPSO) algorithm combines the essential features of Particle Swarm Optimization (PSO) approach to elements of the behavior of dipoles in a magnetic field. The original PSO approach, developed by Kennedy and Eberhart in 1995 [14], employs a population of particles - each one corresponding to a potential solution to a given problem – which are capable of moving in the search space and of "recalling" favorable positions visited by themselves and by their neighbors. The MPSO algorithm [1], on the other hand, is a method that uses elements of the behavior of the aforementioned dipoles[1] to establish search mechanisms, a strategy that led to interesting results for a number of benchmark scenarios [15]. In this work, the k-means optimizer, described in Section 2.1, is applied to initialize the population of the MPSO, thereby promoting the expected synergy between optimization and clustering. The obtained centroids are expected to lie in promising regions of the search space, hopefully near to the global optimum of the cost function, so that the MPSO shall begin with a more adapted population, as the k-means positioned the centers taking into account the quality of several samples. The combined use of MPSO and the k-means optimizer forms the main algorithm we shall analyze, even though, as remarked before, the clustering-based formulation introduced is this work can be virtually employed with any optimization technique, including genetic and immune-inspired algorithms, as a tool to enhance its multimodal search capability.

In the following, we discuss the proposed combination between MPSO and k-means.

[1] *We will use the term "particle" while referring to a magnetic dipole.*

3.1 The Proposed Hybrid Algorithm

In the following, we will describe some aspects and settings of the MPSO algorithm in a very succinct way, for two reasons: 1) the use of this particular search method is not a *sine qua non* condition for building a clustering-based optimization approach and 2) a full description would probably require the abridgement of sections containing original material, which did not seem to be reasonable as the reader can be directly referred to [1] for more details.

Here, the MPSO starts from a population composed by the k-centroids found by the k-means optimizer, described in Section 2.1. The maximum radius of the region of repulsion of each particle [1] is calculated using a measure corresponding to half the minimal distance between centroids. The minimum radius is set throughout this work as the thousandth part of the maximum radius. This choice is heuristic, but it is important to point out that, as discussed in [1][2], the MPSO is not particularly sensitive to this specific parameter. The existence of the repulsion region aims, in simple terms, to prevent the particles from getting excessively close to one another.

Then, each particle searches for a better position generating points inside the repulsion region: when this happens, the particle stores the found solution and also the direction along which it has moved. In the next step, the algorithm generates a new point on this line. If this new point is worse than the original one, two perpendicular points are created and investigated (these points are generated inside the repulsion region): the best perpendicular point shall be adopted if it is better than the current point, and, in this case, the perpendicular line will also be adopted. If there are no points along the perpendicular line that are better than the current point, the particle discards the current direction and a new point is randomly generated within the repulsion region. The general structure of the method is shown in Algorithm 1.

Algorithm 1: Magnetic Particle Swarm Algorithm (with k-means)

```
k-means()
while the termination criteria are not satisfied do
    Move particles()
    Verify confronts()
end while
```

The function k-means() was explained in section 2.1, and, as already indicated, the other functions of algorithm were explained in [1].

4 Experiments and Results

In this section, we present the results obtained with the proposed approach, which combines the k-means optimizer, described in Section 2.1, with the MPSO algorithm. In order to analyze the benefits of the clustering-based formulation, the performance of the hybrid MPSO is compared to that obtained with the standard MPSO and to that obtained using only the k-means optimizer. The experiments were carried out considering five benchmark cost functions, all of which are well-known in the literature [15], using different search space sizes.

a) Rosenbrock Function

$$f(x) = \sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2, \quad (4)$$

where, $x \in [-30,30]^n$ and the global optimum is x*=[1,1,…,1].

b) Griewank Function

$$f(x) = 1 + 0.00025 \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\frac{x_i}{\sqrt{i}}, \quad (5)$$

where, $x \in [-600,600]^n$. The global optimum is x*=[0,0,…,0].

c) Rastrigin Function

$$f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)], \quad (6)$$

where, $x \in [-5.12, 5.12]^n$ and the global optimum is x*=[0,0,…,0].

d) Schaffer Function

$$f(x) = \sum_{i=1}^{n-1} \left[0.5 + \frac{\left(sin^2\left(\sqrt{x_{i+1}^2 + x_i^2} \right) - 0.5 \right)}{\left(0.001 * \left(x_{i+1}^2 + x_i^2 \right)^2 + 1 \right)} \right] \quad (7)$$

where, $\in [-10,10]^n$.

e) Schwefel Function

$$f(x) = -\sum_{i=1}^{n} x_i \sin\sqrt{|x_i|}, \quad (8)$$

where x ∈ [-500,500]^n and the global optimum is x*=[420.97,…,420.97].

4.1 Results

In the experiments, both versions of the MPSO used 30 particles, one per search space dimension in the first series of tests, and were allowed to evaluate 1,000,000 (one million) times each cost function. The k-means used 1000 points per particle to generate the initial points of the hybrid MPSO, which means that this technique employs *a priori* 30,000 cost function evaluations.

The k-means optimizer generated 30 centroids using 1,000,000 points uniformly distributed over the search space; the centroid nearest to best point is moved to the best point. Therefore, the best point found becomes a centroid. The algorithms were run 30 times for each case and the mean and standard deviation of the best attained costs were calculated, as shown in Tab. I.

TABLE I. MEAN AND STANDARD DEVIATION VALUES OF THE BEST COST VALUE – SEARCH SPACE WITH 30 DIMENSIONS.

Functions		Algorithms		
	Global	K-Means	Hybrid MPSO (with k-means)	MPSO
Rosenbrock	0,00	49488304.62 ± 8804981.23	**0.03 ± 0.10**	0.17 ± 0.20
Griewank	0,00	291.02 ± 30.81	**0.00 ± 0.00**	**0.00 ± 0.00**
Rastrigin	0,00	300.60 ± 7.01	**0.00 ± 0.00**	**0.00 ± 0.00**
Schaffer		15.86 ± 0.13	**27.30 ± 1.08**	25.05 ± 2.66
Schwefel	-12569,49	-4961.20 ± 153.38	**-12569.41 ± 0.09**	-12569.40 ± 0.12

Table I shows that the use of the k-means optimizer effectively led to a performance improvement insofar as the MPSO is concerned, but was not able per se to obtain expressive results, which is, in part, a consequence of its inability to perform local refinement.

In order to provide more insight on the statistical relevance of the obtained results, the following figures present the evolution of the fitness of the best particle and also of the average fitness. The curves of the Hybrid MPSO start after those of the MPSO because of the 30,000 points used by the k-means optimizer.

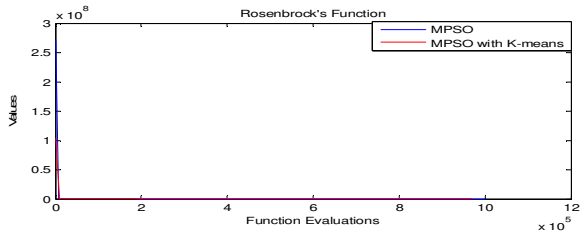


Figure 2: The convergence of the best particle.

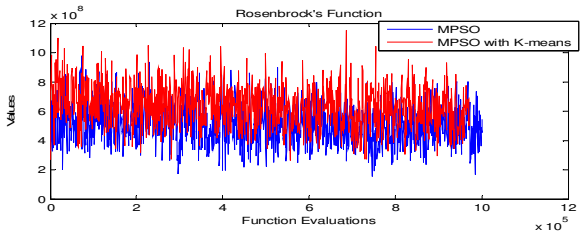


Figure 3: The convergence of mean.

In Fig.2, it is noticeable that the best particle converges very quickly in both cases, but the Hybrid MPSO, as expected, had a profile of average fitness that indicates a more significant population diversity, as shown in Fig. 3. The same appreciation can be extended to Figs. 4 – 7. Finally, Figs. 10 and 11 reveal, once more, the positive effect of the k-means optimizer in terms of exploration.

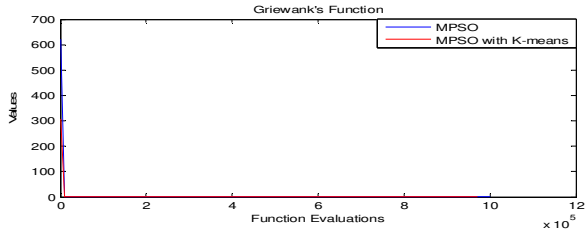


Figure 4: The convergence of the best particle.

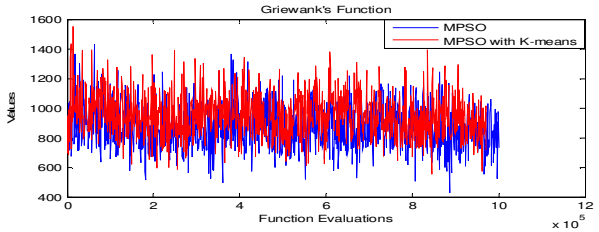


Figure 5: The convergence of mean.

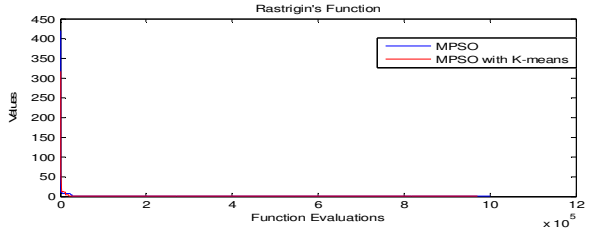


Figure 6: The convergence of the best particle.

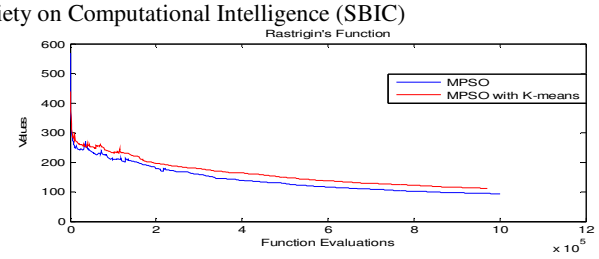


Figure 7: The convergence of mean.

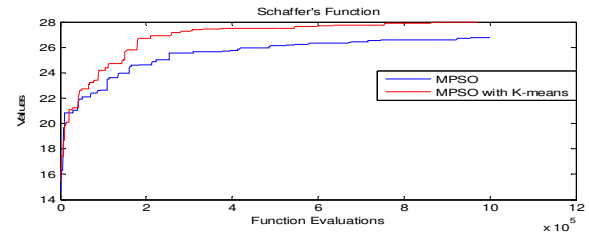


Figure 8: The convergence of the best particle.

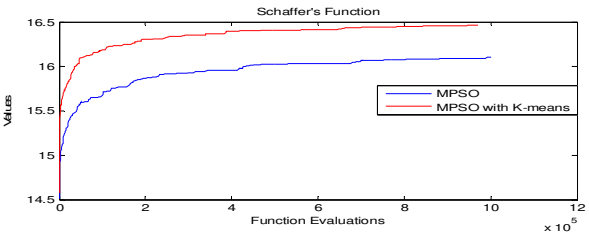


Figure 9: The convergence of mean.

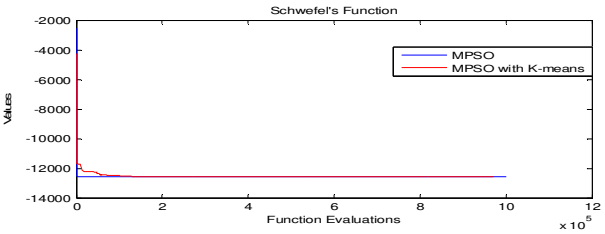


Figure 10: The convergence of the best particle.

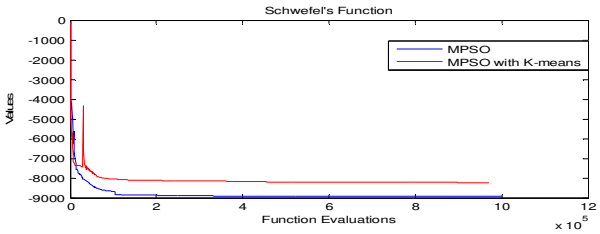


Figure 11: The convergence of mean.

The following series of tests is more challenging, as it is based on a search space of 1000 dimensions. The settings are the same used in the previous series, and the results are in Tab. II. Notice that the k-means optimizer was not considered here in view of its performance in the previous series of tests.

TABLE II. MEAN AND STANDARD DEVIATION VALUES OF THE BEST COST VALUE - SEARCH SPACE WITH 1000 DIMENSIONS.

Functions		Algorithms	
	Global	Hybrid MPSO (with k-means)	MPSO
Rosenbrock	0,00	0.00 ± 0.00	2.64 ± 1.59
Griewank	0,00	0.00 ± 0.00	0.00 ± 0.00
Rastrigin	0,00	0.02 ± 0.02	0.00 ± 0.00
Schaffer		723.52 ± 47.93	893.08 ± 169.31
Schwefel	-418982,9	-418982.88 ± 0.00	-418955.24 ± 32.55

In this case, the k-means optimizer does not improve the performance of the MPSO in all scenarios, as in the 30-dimensional context. Notwithstanding, the performance is improved performance for the Rosenbrock and Schwefel functions. The reason for this performance is, in our opinion, that the 30,000 points employed by the k-means to generate 30-centroids have a smaller impact for a 100-dimension search space, albeit the results, express in terms of the best solution, do not imply a smaller capability with respect to exploration.

5 Conclusions

In this work, we proposed a formulation of the optimization problem as a clustering task, which has the potential of allowing multimodal search to be performed in a organic and straightforward way. The idea took the shape of an algorithm called k-means optimizer, although any other clustering technique could have been, in principle, applied.

To test the proposal, we employed it as part of a hybrid version of the magnetic particle swarm algorithm (MPSO), a technique that, in its original version, had already proved itself to be efficient in multimodal domains. A standalone k-means optimizer was also tested, but its performance was not satisfactory due to an inherent lack of a local refinement mechanism. In summary, the results reveal that there was a general trend for performance improvement in terms of multimodal search

potential with the accessory use of the clustering-based approach. This indicates the relevance of further investigating the clustering-based formulation, probably taking into account distinct clustering and optimization algorithms and also analyzing in more detail the question of sensitivity to the various required settings.

Acknowledgements

The authors thank FAPESP for the financial support.

6 References

[1] Paulo S. Prampero and Romis Attux, "Magnetic Particle Swarm Optimization", **Proceedings of the IEEE Swarm Intelligence Symposium (SIS2011)**, Paris, France, 2011.

[2] Paulo S. Prampero and Romis Attux, "Magnetic Particle Swarm Optimization with Estimation of Distribution", **Proceedings of the IEEE Congress on Evolutionary Computatio**n (CEC2011), New Orleans, USA, 2011.

[3] A. Passaro, A. Starita, "Clustering particles for multimodal function optimization". **In: Proc. GSICE/WIVA**. (2006) published on CD, ISSN 1970-5077.

[4] T. Zhang, "Filter-based training pattern classification for spatial pattern simulation", **PhD thesis**, Stanford University, (2006).

[5] M. Honarkhah, J. Caers, "Stochastic simulation of patterns using distance-based pattern modeling", **Mathematical Geosciences**, 42, 487-517, (2010).

[6] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters", **Journal of Cybernetics**,3, 32-57, (1973).

[7] J. C. Bezdek, "Pattern recognition with fuzzy objective function algorithms", **Plenum Press**, New York, (1981).

[8] S. P. Lloyd, "Least squares quantization in PCM", IEEE **Transactions on Information Theory**, 28 (2), 129-137, (1982).

[9] M. Sabin, R. Gray, "Global convergence and empirical consistency of the generalized Lloyd algorithm", IEEE **Transactions on Information Theory**, 32 (2), 148-155, (1986).

[10] R. O. Duda, P. E. Hart and D. G. Stork, "Pattern classification", **John Wiley & Sons**, 2nd edition, (2001).

[11] A. K. Jain, M. N. Murty, P. J. Flynn, Data clustering: a review, **ACM Computing Surveys**, 31, 264-323, (1999).

[12] J. McQueen, "Some methods for classification and analysis of multivariate observations", In **Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability**, 281-297, (1967).

[13] T. Kohonen, "Self-organizing maps", **Springer**, Berlin, (1995).

[14] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," Proceedings of IEEE **International Conference on Neural Networks,** pp. 1942–1948, 1995.

[15] Molga M. and Smutnicki C., "Test functions for optimization needs" -(2005), available at http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf

[16] J. Kennedy, R. C. Eberhart and Y. SHI, "Swarm Intelligence," San Francisco: Morgan Kaufmann/ **Academic Press**, 2001.

[17] A. P. Engelbrecht, "Computational intelligence: an introduction" 2nd ed. **Jonh Wiley and Sons, Chichester**, England, 2007, pp. 237-260.