International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:4, No:2, 2010

# Using Pattern Search Methods for Minimizing Clustering Problems

Parvaneh Shabanzadeh, Malik Hj Abu Hassan, Leong Wah June, Maryam Mohagheghtabar

*Abstract*—Clustering is one of an interesting data mining topics that can be applied in many fields. Recently, the problem of cluster analysis is formulated as a problem of nonsmooth, nonconvex optimization, and an algorithm for solving the cluster analysis problem based on nonsmooth optimization techniques is developed. This optimization problem has a number of characteristics that make it challenging: it has many local minimum, the optimization variables can be either continuous or categorical, and there are no exact analytical derivatives. In this study we show how to apply a particular class of optimization methods known as pattern search methods to address these challenges. These methods do not explicitly use derivatives, an important feature that has not been addressed in previous studies. Results of numerical experiments are presented which demonstrate the effectiveness of the proposed method.

*Keywords*—Clustering functions, Non-smooth Optimization, Non-convex Optimization, Pattern Search Method.

## I. Introduction

CLUSTERING is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). The clustering problem has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps in exploratory data analysis. However, clustering is a difficult problem combinatorially, and different approaches to this problem have been proposed and studied"[1].

In cluster analysis, we are given a finite set *B* of points in the *d*-dimensional space $R^d$, that is $B = \{b^1, \ldots, b^n\}$, where $b^j \in R^d, j = 1, \cdots, n$.

There are different types of clustering such as packing, partition, covering and hierarchical clustering [2]. In this paper we consider partition clustering, that is, the problem of distributing points of the set *B* into a given number *k* of separated subsets $B^i \neq \phi$ with respect to predefined criteria such that:

*(i)* $B^i \qquad i = 1, \cdots, k;$
*(ii)* $B^i \cap B^j = \phi \qquad i, j = 1, \cdots, k, \quad i \neq j;$
*(iii)* $B = \bigcup_{i=1}^{k} B^i.$

The sets $B^i, i = 1, \cdots, k$ are called clusters.

Suppose that each cluster $B^i, i = 1, \cdots, k$ can be recognized by its center (or centroid) $bc^i \in R^d, i = 1, \cdots, k$. Then the clustering problem can be reduced to the following optimization problem [3],[4]:

$$\min \psi(bc, w) = \frac{1}{n} \sum_{j=1}^{n} \sum_{i=1}^{k} w_{ij} \left\| bc^i - b^j \right\| \qquad (1)$$

$$S.t\ bc = \left(bc^1, \cdots, bc^k\right) \in R^{d \times k},$$
$$\sum_{i=1}^{k} w_{ij} = 1, \quad j = 1, \cdots, n,$$
$$w_{ij} = 0\ or\ 1,\ for\ j = 1, \cdots, n,\ i = 1, \cdots, k.$$

where $w_{ij}$ is the association weight of pattern $b^j$ with cluster *i*, if pattern *j* is allocated to cluster *i* then $w_{ij} = 1$ otherwise $w_{ij} = 0$ and $bc^i = \frac{\sum_{j=1}^{n} w_{ij} b^j}{\sum_{j=1}^{n} w_{ij}} \quad i = 1, \cdots, k.$

Here $\|.\|$ is an Euclidean norm and $w$ is an $n \times k$ matrix; also problem (1) is also known as minimum sum-of-squares clustering problem. This problem is a global optimization problem. Therefore different algorithms of mathematical programming can be applied to solve this problem. In [1], [5] up-to-date and good review of these algorithms including dynamic programming, branch and bound, cutting planes and etc are presented. Also different heuristics can be used for solving large clustering problems and k-means is one such algorithm. This is a very fast and famous algorithm in clustering and it gives good results when there are few clusters but worsens when there are many [2], [6].

Much better results have been obtained with metaheuristics, such as genetic algorithm and tabu search, simulated annealing [7]. We continue the effort to find a local optimizer for the clustering problem so that an algorithm for clustering based on non-smooth optimization techniques is developed in [5]. Here we introduce this algorithm, which calculates clusters step-by-step, gradually increasing the number of data clusters until stopping conditions are met. In this approach the clustering problem is moderated to an unconstrained optimization problem with non-smooth objective function. In the present article we have adapted and used pattern search method to solve this optimization problem and we are able to verify that pattern search methods have better performance than some of best known ways. Pattern search methods have been widely employed in many applications. This method needs an initial search that use heuristics to find good initial points near some promising local minima before running the pattern search algorithm, where it is used as a local optimizer for the continuous variables. They generate significantly fewer invalid solutions and also our numerical experiments show that pattern search methods are robust. In [8] an excellent introduction and survey of these methods can be found, which also contains numerous references. We will review these methods in more detail in section IV.

M. Hassan and L. June and P. Shabanzadeh are with the Institute for Mathematical Research (INSPEM), University Putra Malaysia (E-mail: malik@science.upm.edu.my, leongwj@putra.upm.edu.my, parvaneh.sh@inspem.upm.edu.my)
M. Mohagheghtabar is with the Department of Mathematics, University Kebangsaan Malaysia. (E-mail: maryamohaghegh@gmail.com )

Results of computational experiments using real-world datasets are presented and we compare our results with the best known solutions from the literature.

The article is organized as follows: In section II it is reviewed the non-smooth optimization approach to clustering and an algorithm for solving clustering problems. In section III it is introduced the pattern search method for solving optimization problems. In section IV it's presented discussion of the results of computational experiments and analysis them. Finally in section V it is given concluding remarks.

## II. Cluster Analysis

"Cluster analysis is the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity; intuitively patterns within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster"[1].

Consider a set *B* which consists of *n d*-dimensional vectors $b_j = (b_{j1}, \cdots, b_{jd}), j = 1, \cdots, n$. Assume that this set can be shown as the union of *k* clusters. Assume also, that each cluster can be shown by a point, which can be considered as the center of this cluster. In order to find a cluster we should find its center. Thus we would like to find *k* points which are centers of clusters. Assume that we have a set *BC*, consisting of *k* points $bc_1, \cdots, bc_k$. The distance $d(b_j, BC)$ from a point $b_j \in B$ to this set is defined by $d(b_j, BC) = \min_{i=1,\cdots,k} \|b_j - bc_i\|$. The deviation $d(B, BC)$ from the set *B* to the set *BC* is computed by the formula

$$d(B, BC) = \sum_{i=1}^{n} d(b_j, BC) = \sum_{i=1}^{n} \min_{i=1,\cdots,k} \|b_j - bc_i\|$$

Thus the cluster analysis problem can be written in the following problem of mathematical programming:

$$\min g(bc_1, \cdots, bc_k) = \sum_{j=1}^{n} \min_{i=1,\cdots,k} \|b_j - bc_i\|, \qquad (2)$$

*S.t* $(bc_1, \cdots, bc_k) \in R^{d \times k}$.

If $k > 1$, the objective function g in the problem (2) is non-convex and nonsmooth. We call g the cluster function. In [9] it is shown that problems (1) and (2) are equivalent. Note that the problem (1) contains both integer and continuous variables whereas in the nonsmooth global optimization formulation of the clustering problem (2) we have only continuous variables. Also, note that the number of variables in problem (1) is $(n + d) \times k$ while in problem (2) this number is only $d \times k$ and the number of variables does not depend on the number of instances. On the other hand in many real-world databases the number of instances *n* is significantly greater than the number of attributes *d*. All these conditions can be considered as advantages of the nonsmooth optimization formulation (2). Note that in clustering analysis, valid choice of the number of clusters is very important and it is difficult to identify a priori how many clusters are present in the set *B* under consideration; therefore to avoid this difficulty, a step-by-step calculation of clusters is presented and the optimization algorithm is discussed in the following subsection.

### A. An Optimization Algorithm for Solving Clustering Problems

We will describe the following algorithm for solving cluster analysis problems.

***Algorithm 1:*** *An algorithm for solving a cluster problem.*

***Step 1:*** *(Initialization). Select a tolerance $\epsilon > 0$ and an initial point $bc_0 = (bc_0^1, \cdots, bc_0^d) \in R^d$. Let $k = 1$, solve the minimization problem (2). Let $bc_1^* \in R^d$ be a solution to this problem and $g_1^*$ be the corresponding objective function value. Set $h = 1$.*

***Step 2:*** *(Computation of the next cluster center). Select a starting point and solve the following minimization problem:*

$$\min \overline{g}_h(y) = \sum_{j=1}^{n} \min \{\|bc_1^* - b_j\|, \cdots, \|bc_h^* - b_j\|, \|y - b_j\|\} \qquad (3)$$

*S.t* $y \in R^d$.

***Step 3:*** *(Modification of all cluster centers). Let $y_{h+1}^*$ be a solution to problem (3). Take $bc_{h+1}^0 = (bc_1^*, \cdots, bc_h^*, y_{h+1}^*)$ as a new starting point and solve the following minimization problem:*

$$\min \overline{g}_{h+1}(bc) = \sum_{j=1}^{n} \min_{i=1,\cdots,h+1} \|bc_i - b_j\| \qquad (4)$$

*S.t* $bc = (bc_1, \cdots, bc_{h+1}) \in R^{d \times (h+1)}$.

***Step 4:*** *(Stopping criterion). Let $bc_{h+1}^*$ be a solution to the problem (4) and $g_{h+1}^*$ be the corresponding value of the objective function. If*

$$\frac{g_h^* - g_{h+1}^*}{g_1^*} < \epsilon$$

*then stop, otherwise set $h = h + 1$ and go to Step 2.*

Note that in Step 1 the center of the entire set *B* will be calculated, that is, in this case the problem (2) is a convex programming problem. In step 2, we consider previous *h* cluster centers to be known, and estimate the center of next $(h + 1)$-th cluster. Note also, the number of variables in problem (3) is *d* which is lesser than if we calculate all cluster centers simultaneously. In Step 3 the modification of all $h + 1$ cluster centers is carried out. It should be noted since the starting point $bc_{h+1}^0$ calculated in step 2 is not far from the solution to problem (4); therefore it takes only a reasonable number of iterations to calculate it and this tact reduce the computational time for solving problem (4). It is obvious that $g_h^* \geq 0$ for all $h \geq 1$ and the sequence $\{g_h^*\}$ is decreasing, that is, $g_{h+1}^* \leq g_h^*$ for all $h \geq 1$. This condition implies that after $\overline{h} > 0$ iterations the stopping criterion in Step 4 will be satisfied. When one tries to apply Algorithm 1, one of the important questions is the choice of the tolerance $\epsilon > 0$. Large values of $\epsilon$ can result in the appearance of large clusters while small values can produce small and artificial clusters. Results studied in [5] show that appropriate values for $\epsilon$ are $[10^{-1}, 10^{-2}]$. We will explain an optimization method for solving problems (3) and (4) in section III.

## III. PATTERN SEARCH METHOD

Pattern search methods are a class of optimization methods known as direct search methods. Direct search methods are the ways for solving optimization problems that do not need any information about the gradient of the objective function in contrast to more traditional optimization methods that use information about the gradient or higher derivatives to search for an optimal point, a direct search algorithm searches a set of points around the current point, looking for one where the value of the objective function is lower than the value at the current point. These methods have a rich history in science and engineering where they have been used to many problems. An excellent introduction and survey of these methods is explained in [8], which also contains many references. An attractive characteristic of the pattern search method is that it is simple and easy to implement and it only needs the ability to evaluate the function at a point.

### A. Generalized Pattern Search Algorithms

Several generalizations of the simple pattern search method have been suggested and studied previously [8], [10], [11], [12]. These methods fall under the general classification of generating set search (GSS) methods, which are categorized by using multiple search directions computed from a generating set. One example from this class of methods is generalized pattern search (GPS) method. We study the notation for GPS presented in [13], [14], and demonstrate a very brief description of the algorithm.

Briefly at each step, the algorithm searches a set of points, called a mesh, around the current point-the point computed at the previous step of the algorithm. The mesh is formed by adding the current point to a scalar multiple of a set of vectors called a pattern. If the pattern search algorithm finds a point in the mesh that improves the objective function at the current point, the new point becomes the current point at the next step of the algorithm.

Any instance of a GPS algorithm needs the following parameters:

- Initial point $y_0 \in R^n$ (with finite value $g(y_0)$ ) and mesh size parameter $\Delta_0 > 0$ in *R*.
- Non-singular generating matrix $D \in R^{n\times n}$ and a $n \times p$ matrix $Z \subset Z^{n\times p}$ whose columns form a positive spanning set.
- Mesh updates parameters $\alpha \in Q$ and $\omega^-, \omega^+ \in Z$ with $\alpha > 1, \omega^+ \geq 0, \omega^- \leq -1$.

Let *g* be the objective function. Starting from an initial guess $y_0$ and an initial value $\Delta_0$ of the step length control parameter, a GPS method generates a sequence of iterates $\{y_k\}$ such that $g(y_{k+1}) \leq g(y_k)$ . Each trial point where the algorithm evaluates *g* must lie on the current mesh, defined by

$$M_k = \{y_k + \Delta_k H z : z \in N^p\}, where \quad H = DZ \qquad (5)$$

and where $\Delta_k > 0$ is the mesh size parameter at iteration *k*. At each iteration, trial points can be produced in two steps. The difference between the two steps is in the way the trial points are elected. In the first step, called the Search, any finite strategy can be applied to find a mesh point that gives a lower objective function value than the incumbent as long as only finitely many trial points are selected. When a trial point $y_{k+1} \in M_k$ satisfying $g(y_{k+1}) < g(y_k)$ is found, then $y_{k+1}$ is said to be an *improved mesh point*, and the Search Step can be stopped.

At whatever time the Search Step fails to generate an improved mesh point, a second step, called the Poll, is performed before the iteration is completed. In this phase the objective function is evaluated at the neighboring mesh points to see if a lower value can be found there. The set of neighboring mesh points (called the Poll Set) is made using a positive spanning matrix $H_k$ composed of columns of the finite set *H* (defined in Eq.(5) ):

$$Pollset : \{y_k + \Delta_k h : h \in H_k\} \subset M_k. \qquad (6)$$

If the poll set includes a trial point $y_{k+1} \in M_k$ such that $g(y_{k+1}) < g(y_k)$ then, as in the Search Step $y_{k+1}$ is said to be an *improved mesh point*, and the Poll Step can be stopped.

If both the Search and the Poll steps do not succeed to generate an improved mesh point, then the current incumbent solution $y_k$ is said to be a *mesh local optimizer* (i.e., its objective function value is less than or equal to that of all points from the poll set). Then the algorithm updates the mesh by setting the mesh size parameter $\Delta_{k+1} = \alpha^{\omega_k}\Delta_k$ where $\omega_k \in Z$ and

$$\omega^- \leq \omega_k \leq -1 \qquad (7)$$

and therefore $0 < \alpha^{\omega_k} < 1$.Otherwise if either the Search or Poll step generates an improved mesh point, then the new point $y_{k+1} \neq y_k$ has a strictly lower objective function value, then the mesh size parameter is kept the same or is increased, and the process is repeated. It follows that if $y_{k+1}$ is an improved mesh point, then the later iterates can never return to previous iterates: $y_j \neq y_l$ and $g(y_j) > g(y_l)$ for any $j \leq k$ and $l > k$. The roughening of the mesh follows the rule $\Delta_{k+1} = \alpha^{\omega_k}\Delta_k$ where $\omega_k \in N$ and

$$0 \leq \omega_k \leq \omega^+ \qquad (8)$$

and therefore $\alpha^{\omega_k} \geq 1$. These generalizations allow for great freedom in using the GPS method and can have a significant impact on the efficiency of the algorithm. Then the GPS method can be described as follows:

***Algorithm 2:*** A basic GPS algorithm

1) *Initialization:* Let $y_0$ be such that $g(y_0)$ is finite, and let $M_0$ be the mesh on $R^n$ defined by $\Delta_0 > 0$ and let $H_0, \omega^-, \omega^+, \alpha, D, H, Z$ satisfy the requirements given above. Set the iteration counter k to *0*.
2) *Search and Poll Step:* Execute the Search and possibly the Poll steps (or only part of them if an improved mesh point is found on the mesh defined by (5)).
   a) *Optional Search:* Evaluate *g* on a finite subset of trial points on the mesh $M_k$.
   b) *Local Poll:* Evaluate *g* on the poll set defined in (6).
3) *Parameter Update:* If the Search or Poll step generated an improved mesh point, $y_{k+1} \in M_k$ for which $g(y_{k+1}) < g(y_k)$, then update $\Delta_{k+1} \geq \Delta_k$ according

to rule (8). Otherwise, $g(y_k) \leq g(y_k + \Delta_k h)$ for all $h \in H_k$ and so $y_k$ is a mesh local optimizer; set $y_{k+1} = y_k$, update $\Delta_{k+1} < \Delta_k$ according to rule (7). Increase $k \leftarrow k + 1$ and go back to the Search and Poll step.

Each step of the GPS algorithm can be specified even further. For example, in step 1 the lengths of the vectors in the generating set can take on any values between specified lower and upper bounds. Note the search strategy is the key to effectiveness. Especially in practice it allows the use of heuristic and surrogate methods to explore the domain of the variables. For example, one might apply a few generations of a genetic algorithm on the mesh to g or any other physics-based approach that seems suitable. Also using the search method reduces the total function count-the number of times the objective function was evaluated.

## IV. RESULTS AND DISCUTION

To verify the efficiency of the proposed algorithms a number of numerical experiments with real-world data sets have been carried out on a PC, Intel(R) Core2Duo CPU,1.95GB of RAM. In our calculations, we apply Algorithm 2 for solving problems of clustering. As mentioned before, mesh set plays the main role in GPS algorithm. Therefore we apply some methods in order to form patterns (directions) in mesh set, for both steps (poll and search). The patterns are the Positive basis *2n* and the Positive basis $n+1$ selected in regular form and random, where *n* is the number of independent variables for the objective function. Positive basis *2n* regular pattern is formed using the positive and negative of the linearly independent identity vectors $\{e_i, -e_i | i = 1, \cdots, n\}$. As mentioned earlier, linearly independent identity vectors can be generated using a random permutation of an n-by-n linearly independent lower triangular matrix that is regenerated at each iteration (as mentioned in before part). So we tested 5 type of algorithm GPS for solving clustering problem:

- *GPS1* is the Algorithm 2 that in search step, positive basis random $n + 1$ is applied for directions of mesh set and positive basis random $2n$ is applied for directions of mesh set in poll step.
- *GPS2* is the Algorithm 2 that in search step, positive basis regular $n + 1$ is applied for directions of mesh set and positive basis regular $2n$ is applied for directions of mesh set in poll step.
- *GPS3* is the Algorithm 2 that genetic algorithm is used for search step, and positive basis regular $2n$ is applied for directions of mesh set in poll step.
- *GPS4* is the Algorithm 2 that Nelder-Mead algorithm is used for search step, and positive basis regular 2n is applied for directions of mesh set in poll step.
- *GSP5* is the Algorithm 2 that search step is empty, and positive basis regular 2n is applied for directions of mesh set in poll step.

Also, we consider four standard test problems to compare our algorithm with the following heuristics and metaheuristics: the tabu search method *(TS)*, a genetic algorithm *(GA)* and the simulated annealing *(SA)* method, the k-means algorithm *(K-M)*. We use the results obtained by using *TS, GA, SA* and *K-M* in [15] for comparison. It should be noted that *SA, GA* and *TS* have been applied to problem (1) which is equivalent to the nonsmooth optimization formulation of a clustering problem. In Tables we report the best known value for the global minimum. The error *E* is considered as

$$E = \frac{(\bar{g} - g_{opt})}{g_{opt}}$$

where $g_{opt}$ and $g$ indicate the best known solution and the function value obtained by the algorithm respectively. In these tables Q represents the number of clusters and the following datasets are used in numerical experiments and they can be found in [4]:

- German towns database ($m = 59, n = 2$) ;
- First Bavarian postal zones data set ( $m = 89, n = 3$) ;
- Second Bavarian postal zones data set ($m = 89, n = 4$);

The results presented in Table I show that for this data set, *GPS1* achieve the same or better results than all algorithms mentioned except for $Q = 2$. *TS* and *GA* and *SA* are approximately better than *GPS2* and *GPS3* but the result of *GPS2* is better than *K-M* and also *GPS3* is better than *K-M* except for $Q = 2$. The results presented in Table I explain that *GPS4* for $Q = 2, 4$ is the same as *TS,GA,SA* but for $Q = 5$, *GPS4* is better than these methods and for $Q = 3$ *GPS4* is worse. Also *GPS4* is better than *K-M*. The results from *GPS5* and TS are similar except for $Q = 3$; also the results of *GPS5* and *GA* and *SA* are same except for $Q = 3$. But result of *GPS5* is better than *K-M* except for $Q = 3$.

From Table II we can see that $GPSi \;\; i = 1, \cdots, 5$ are better than all algorithms except $GPSi \;\; i = 2, 3, 4$ are similar with *GA*.

The results presented in Table III explain that for this data set, *GPS1* achieves better results than the *K-M* and *SA* and *TS* except for $Q = 5$ and the same with *GA*. The results obtained by our experiment *GPS2* and *GPS3* are the same, however they are better than those obtained by *K-M* and *SA* and TS and also these are the same with *GA* except for $Q = 5$ that *GPS2* is better. The results from *GPS4* for $Q = 2$ show that it is almost the same for all other algorithms and for $Q = 3$, *TS* and *GA* and *SA* are better than *GPS4*; also for $Q = 4$ it is the same with *TS* and *GA* but it is better than *K-M* and *SA*, and also for $Q = 5$ the result of *TS* is better than *GPS4*. Therefore it is almost better than *GA*, *SA*, *K-M*. The results of table III show that *TS* is better than *GPS5* and results of *GPS5* and *GA* are the same except for $Q = 5$ that *GPS5* is better and also *GPS5* is better than *SA* and *K-M*. Based on the results presented in Tables [I, II, and III], we can conclude that at least for these three data sets, $GPSi \;\; i = 1, \cdots, 5$ work better than the k-means algorithm and achieve closer, similar and sometimes better results than the genetic and simulated annealing and tabu search algorithms for solving clustering problem.

## V. CONCLUSION

In this paper is studied an algorithm for solving a clustering problem where the problem is treated as a non smooth,

TABLE I
RESULTS FOR GERMAN TOWNS DATA SET

| $Q$ | $f_{opt}$ | *TS* | *GA* | *SA* | *K-M* | *GPS1* | *GPS2* | *GPS3* | *GPS4* | *GPS5* |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 | $2.1426 \times 10^4$ | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 3 | $7.009 \times 10^3$ | 0.00 | 0.00 | 0.29 | 1.45 | 1.45 | 0.29 | 18.08 | 0.35 | 18.08 |
| 4 | $49.601 \times 10^3$ | 0.00 | 0.00 | 0.00 | 0.55 | 0.00 | 0.24 | 0.23 | 0.00 | 0.00 |
| 5 | $39.453 \times 10^3$ | 0.00 | 0.15 | 0.15 | 2.75 | $-1.86$ | 0.00 | 0.14 | $-1.86$ | 0.00 |

TABLE II
RESULTS FOR FIRST BAVARIAN POSTAL ZONES DATA SET

| $Q$ | $f_{opt}$ | *TS* | *GA* | *SA* | *K-M* | *GPS1* | *GPS2* | *GPS3* | *GPS4* | *GPS5* |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 | $60.255 \times 10^{10}$ | 0.00 | 0.00 | 0.00 | 7.75 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 3 | $29.451 \times 10^{10}$ | 23.48 | 23.48 | 23.48 | 23.48 | 0.00 | 23.48 | 23.48 | 23.44 | 23.48 |
| 4 | $10.447 \times 10^{10}$ | 18.14 | 0.00 | 0.39 | 66.88 | 0.00 | 0.00 | 0.00 | 0.00 | $-0.28$ |
| 5 | $59.762 \times 10^9$ | 33.35 | 0.00 | 40.32 | 35.32 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

TABLE III
RESULTS FOR SECOND BAVARIAN POSTAL ZONES DATA SET

| $Q$ | $f_{opt}$ | *TS* | *GA* | *SA* | *K-M* | *GPS1* | *GPS2* | *GPS3* | *GPS4* | *GPS5* |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 | $9.9080 \times 10^9$ | 0.00 | 144.25 | 144.25 | 144.25 | 144.28 | 144.28 | 144.28 | 144.28 | 144.28 |
| 3 | $7.3987 \times 10^9$ | 0.00 | 0.00 | 77.77 | 106.79 | 0.00 | 106.78 | 106.78 | 106.78 | 0.00 |
| 4 | $5.5908 \times 10^8$ | 0.00 | 0.00 | 9.13 | 303.67 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 5 | $4.0379 \times 10^8$ | 5.76 | 15.76 | 18.72 | 446.13 | 15.76 | 0.00 | 0.00 | 15.76 | 0.00 |

non convex optimization problem. The proposed algorithm calculates clusters step by step and it allows the decision maker to easily change the number of clusters according to the criteria considered by the nature of decision making issue not incurring the clear costs of the increased complexity of the solution procedure. The generalized pattern search (GPS) method has been applied in different ways to solve the non smooth optimization problems of clustering algorithm. Our evaluations show that GPS optimization methods are promising candidates for clustering of datasets. Evaluations of the numerical experiments is presented in this paper show that the GPS method exhibits a better performance than some efficient heuristic algorithms used for cluster analysis and also metaheuristic approaches to a global optimization at least for the datasets is used in this paper.

## ACKNOWLEDGMENT

The corresponding author would like to express gratitude to Prof.Dr.Adil Bagirov for his useful guide guide and providing datasets for computational experiments.

## REFERENCES

[1] A. K. Jain, M. N. Murty and P. J. Flynn, *Data clustering: a review*, ACM Comput. Surv., Vol. 31, No. 3. , pp. 264-323, Sep. 1999.
[2] P. Hansen, B. Jaumard, *Cluster analysis and mathematical programming*, Mathematical Programming, 79(1-3), 191-215, 1997.
[3] H. H. Bock, *Clustering and neural networks*, in: A. Rizzi, M. Vichi, H.H. Bock (Eds.), Advances in Data Science and Classification, Springer, Berlin, 1998, pp.265-277.
[4] H. Spath, *Cluster Analysis Algorithms*, Ellis Horwood Limited, Chichester, 1980.
[5] A. M. Bagirov, A. M. Rubinov, and N. V. Soukhoroukova, J. Yearwood, *Supervised and unsupervised data classification via nonsmooth and global optimization*, TOP:Span. Oper. Res. J. 11 (1), 1-93, 2003.
[6] P. Hansen, E. Ngai, and B. K. Cheung, N. Mladenovic, *Analysis of global k-means, an incremental heuristic for minimum sum-of squares clustering*, Research Paper G-2002-43, GERAD, Montreal, Canada, 2002.
[7] C. R. Reeves, *Modern Heuristic Techniques for Combinatorial Problems* .London, Blackwell, 1993.
[8] T. G. Kolda, R. M. Lewis, V. Torczon, *Optimization by direct search: new perspectives on some classical and modern methods*. SIAM Rev., 45 (2003), pp.385-482.
[9] H. H. Bock, *Automatische Klassifikation*, Vandenhoeck and Ruprecht, 1974,Gottingen.
[10] V. Torczon, *On the convergence of pattern search algorithms*, SIAM J. Optim. 7 (1997) 1-25.
[11] R. M. Lewis, V. Torczon, *Pattern search algorithms for linearly constrained minimization*, SIAM Journal on Optimization 10(3), 917-941 (2000)
[12] J. E. Dennis Jr., V. Torczon, *Direct Search Methods on Parallel Machines*, SIAM Journal on Optimization 1, 448 ,1991.
[13] C. Audet, J. E. Dennis Jr., *Analysis of generalized pattern searches,* SIAM J. Optim. 13, 889-903, 2003.
[14] C. Audet, *Convergence results for generalized pattern search algorithms are tight*, Optim. Eng. 5 ,101-122, 2004.
[15] K. S. Al-Sultan, M. M. Khan, *Computational experience on four algorithms for the hard clustering problem*, Pattern Recognition Letters 17, 295-308, 1996.

**Malik Hj. Abu Hassan (Prof., Dr.)** received the BSc. (Hons) in Mathematics from University Malaya and MSc degrees in Industrial Mathematics from Aston University UK and he received the PhD degree in Applied Mathematics from Loughborough University UK. He is Professor of the Department of Mathemathics and Computational Information in University Putra Malaysia. He held didactic activity both undergraduate and post graduate level. He interest focuses on large scale unconstrained optimization problems, domains of attractions of non autonomous systems, LQ problem for descriptor systems and predator-prey population models including time delay and harvesting.