Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 349 (2020) 49–67

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

A Multi-Objective Optimization Algorithm for Center-Based Clustering

Jared León[1] Boris Chullo–Llave[2] Lauro Enciso–Rodas[3]
José Luis Soncco–Álvarez[4]

Department of Informatics
Universidad Nacional de San Antonio Abad del Cusco
Cusco, Peru

Abstract

Center-based clustering is a set of clustering problems that require finding a single element, a center, to represent an entire cluster. The algorithms that solve this type of problems are very efficient for clustering large and high-dimensional datasets. In this paper, we propose a similar heuristic used in Lloyd's algorithm to approximately solve (EMAX algorithm) a more robust variation of the k-means problem, namely the EMAX problem. Also, a new center-based clustering algorithm (SSO-C) is proposed, which is based on a swarm intelligence technique called Social Spider Optimization. This algorithm minimizes a multi-objective optimization function defined as a weighted combination of the objective functions of the k-means and EMAX problems. Also, an approximation algorithm for the discrete k-center problem is used as a local search strategy for initializing the population. Results of the experiments showed that SSO-C algorithm is suitable for finding maximum best values, however EMAX algorithm is better in finding median and mean values.

Keywords: Center-Based Clustering, Approximation Algorithms, EMAX, Multi-Objective Optimization, Social Spider Optimization.

1 Introduction

Clustering is usually considered as the most important problem in unsupervised learning. Like any other unsupervised problem, it involves searching for patterns and structures in unlabeled data. The goal of a clustering algorithm is to group similar objects into sets called *clusters*. Due to the nature of the problem, it appears in many research areas such as data compression, image analysis, bioinformatics, and data mining.

[1] Email: jared.leon.m@gmail.com / jared.leon@unsaac.edu.pe
[2] Email: boris.chullo@unsaac.edu.pe
[3] Email: lauro.enciso@unsaac.edu.pe
[4] Email: jose.soncco@unsaac.edu.pe

https://doi.org/10.1016/j.entcs.2020.02.012

In essence, clustering is not a well defined task, but a very general problem to be solved. For that reason, many well defined problems and algorithms have been proposed for this task. There is a great variety of different concepts and definitions of what a cluster is and how it can be formed. There is also no general convention about what types of data can be clustered. So, all that diversity led to several models of clustering [2]. A very popular one is called center-based model.

Center-based clustering is the task of representing an entire cluster by an element called the "center" of the cluster. Although there are many variations of this problem, the basic idea is to make what is called strict partitioning clusterings, where each element belongs to exactly one cluster. The most well-known center-based clustering algorithm is k-means or Lloyd's algorithm [23]. This is one of the most used algorithms probably because of its simplicity and the reasonably good results it provides in most cases. In spite of that, the main problem with the algorithm is the effect that present outliers in the dataset have on it [19].

Many algorithms have been proposed for reducing the outlier effect in the k-means algorithm. However, few of them are strict partitioning clustering algorithms, that is, many of them try to identify the outliers and remove them. In this paper, a robust center-based clustering algorithm, SSO-C, is presented for strict partitioning.

The SSO-C algorithm tries to optimize two center-based problems. One of them is k-means, the other one is EMAX, a closely related problem that requires finding a more robust clustering solution. The two problems are condensed in a single-objective optimization function considering the correlation of the solutions of both problems. The function is to be optimized with a global optimization algorithm called Social Spider Optimization [10] (SSO). Also, an approximation algorithm for the k-center problem is used as local search for generating initial solutions, that is, initial places for starting the search. The SSO-C algorithm is a more general version of a previously proposed algorithm for center-based clustering [27] called SSO-A. In the experiments, four algorithms were evaluated on thirteen datasets: k-means, EMAX, SSO-A, and SSO-C. Six synthetic datasets were generated and seven real others were taken. All the datasets have the true labels of the points, so a clustering metric called Adjusted Mutual Information [28] was used to evaluate the quality of each prediction. A number of executions of each algorithm were performed on each dataset, reporting the mean, median and highest score.

The paper is organized as follows. Section 2 explains the basic background concerning the k-means problem and Lloyd's algorithm, it also includes a review of the Social Spider Optimization algorithm. Section 3 defines a more robust variation of the k-means problem and uses a similar heuristic in Lloyd's algorithm to approximately solve it. Then, the algorithm is presented as a solution for a multi-objective optimization of both problems. The local search strategy used as initialization for few starting points is also explained. Section 4 shows the experimental results of the execution performed on four center-based algorithms. The four algorithms were executed with synthetic and real datasets. Finally, the conclusions are presented in section 6.

2 Background

2.1 K-means

K-means is commonly known as a clustering algorithm rather than an optimization problem. In this work, the term is used interchangeably according to the context. The k-means optimization problem is defined as:

Let $\mathcal{S}$ be a set of n points $\{x_1, x_2, \ldots, x_n\}$ in a metric space $(\mathbb{R}^d, \ell_2)$ where d defines the dimension space and ℓ_2 represents the Euclidean metric, and let $k > 1$ be an integer. Find a set $C = \{c_1, c_2, \ldots, c_k\}$ of k elements in $\mathbb{R}^d$ and a matrix $\mathbf{A} = [a_{ij}]$ of size $n \times k$ such that the following is minimized:

$$J_1(C, \mathbf{A}) = \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} \|x_i - c_j\|_2^2 \tag{1}$$

subject to:

$$a_{ij} \in \{0, 1\} \text{ for all } i, j \tag{2}$$

$$\sum_{j=1}^{k} a_{ij} = 1 \text{ for all } i. \tag{3}$$

The first parameter of the problem, the set C, contains k points in the same space as $\mathcal{S}$. Those points are called the centers. The purpose of each center is to represent a cluster, so that every point in $\mathcal{S}$ is to be assigned to a unique center and therefore, to a unique cluster. Matrix $\mathbf{A}$ defines those assignments. A point x_i is said to be assigned to center c_j if and only if the element a_{ij} is equal to one. Also, (3) ensures that every element of the set $\mathcal{S}$ is assigned to exactly one center of C.

The objective is to find the set of centers C and the assignments matrix $\mathbf{A}$ so that the sum of the squared Euclidean distances from each point to the center it is assigned to is minimized.

The k-means problem was shown to be NP-Hard [24]. However, there is a well known heuristic to approximate a solution for the problem. The heuristic consists of iteratively solving the following sub-problems:

- P_1: Fix $A = \hat{A}$ and solve the reduced problem $J_1(\hat{A}, C)$.
- P_2: Fix $C = \hat{C}$ and solve the reduced problem $J_1(A, \hat{C})$.

P_1 and P_2 can easily be solved in polynomial time using the following lemmas.

Lemma 2.1 ([20]) *Given the k-means problem, let $\hat{A} = \{a_{ij}\}$ be fixed. Then the function J_1 is minimized if and only if each center of C is defined as the mean of the points of $\mathcal{S}$ assigned to it. That is, only if*

$$c_j = \frac{\sum_{i=1}^{n} a_{ij} x_i}{\sum_{i=1}^{n} a_{ij}} \text{ for all } j. \tag{4}$$

Lemma 2.2 ([20]) *Given the k-means problem, let $\hat{C} = \{c_j\}$ be fixed. Then the function J_1 is minimized if and only if each point of $\mathcal{S}$ is assigned to the closest center of C to it. That is, only when matrix A has the following entries:*

$$a_{ij} = \begin{cases} 1 & \text{if } j = \arg\min_k \|x_i - c_k\|_2 \\ 0 & \text{otherwise} \end{cases} \quad \text{for all } i, j. \tag{5}$$

Solving P_1 means defining the set of centers as the centroids of the clusters. Solving P_2 means partitioning the set $\mathcal{S}$ according to the Voronoi diagram induced by the set of centers C. Informally, a Voronoi diagram is a partition of the plain into cells induced by a set of fixed points where every point inside a cell is closer to the fixed point that generated such cell than to any other fixed point. See [15] for more on Voronoi diagrams. The algorithm usually starts by solving problem P_2 fixing the centers with random values. The resulting algorithm is often called the k-means algorithm due to its popularity, also known as Lloyd's algorithm, named after Stuart Lloyd [23]. Several methods for better initialization of the centers have been proposed, being one of the most known the k-means++ algorithm [4]. The convergence of the k-means can be shown, however, there are no guarantees of finding the global minimum [18].

2.2 Social Spider Optimization

Optimization problems can sometimes be difficult to solve in a closed form. In this work, an optimization problem will be faced and a global optimization algorithm will be needed for solving it. One of the recent swarm intelligence algorithms that had good results when compared with state-of-the-art algorithms is the Social Spider Optimization algorithm (SSO) [10]. Swarm intelligence is a collective intelligence of groups of simple agents dealing with behaviors of swarms [13].

The SSO algorithm is based on the simulation of cooperative behavior of social-spiders. The algorithm assumes that the entire search space is a communal web, where each position of a spider represents a solution for the problem. Each spider i receives a weight w_i which represents its solution quality. The information transmitted through the communal web is encoded in form of vibrations. The vibration made by spider j and perceived by spider i is modeled according to:

$$Vib_{ij} = w_j \cdot e^{-\|s_i - s_j\|_2^2}. \tag{6}$$

Where the vibration intensity decreases with the squared distance of the spiders involved. The algorithm distinguishes between male and female agents. The entire process consists on iteratively emulating three cooperative operators: Female Cooperative Operator, Male Cooperative Operator, and Mating Mating Operator. When performing each of these operators, the spiders change their positions according to bio-inspired laws in order to explore the search space and find a better solution.

The algorithm was shown to have better performance in run-time and solution quality than other state-of-the-art global optimization algorithms when evaluated with a set of well known benchmark functions.

3 Proposal

3.1 EMAX

The k-means algorithm is one of the simplest and most popular algorithms for unsupervised machine learning due to its center-based nature. However, center-based algorithms usually make assumptions of the data such as: clusters are assumed to be hyperspherical and evenly sized.

For practical usage, the fact that a solution to the k-means problem makes a partition of the original set induced by the means has a disadvantage: atypical values have a very large influence over the centers positioning because of the sum of squared distance in the objective function. Big distances have greater impact on the objective function in proportion to small distances because the penalization that each point gets grows with the square of the distance from a fixed point. So if a point has distance $2d$ to the center it is assigned to, it will have a much greater penalty than the double of the penalty that has a point with distance d to the center it is assigned to. This behavior becomes more important when the set to be clustered contains many outliers, because an outlier is generally abnormally far of the typical points and in consequence, of the ideal center. If this point is assigned to a center that is surrounded by typical points, it will cause its penalization to be large, making the position of the center to change significantly in the next iteration to reduce the penalization. This is illustrated in Fig. 1 in which an outlier generates a readjustment of the center point to decrease the value of the objective function. In the same figure, it is also shown a special point (geometric median) which we will talk about later.

A common approach to solve the outlier effect and to create a more robust algorithm is to change the objective function with the following expression:

$$\sum_{i=1}^{n}\sum_{j=1}^{k} a_{ij}\|x_i - c_j\|_1, \tag{7}$$

where the the set $\mathcal{S}$ no longer belongs to $(\mathbb{R}^d, \ell_2)$, but to $(\mathbb{R}^d, \ell_1)$, being ℓ_1 the 1-norm or the Manhattan norm. This function has the advantage of being more robust in an outlier scenario because it minimizes the within cluster error with respect to the 1-norm distance metric, as opposed to the squared 2-norm distance metric. Problems P_1 and P_2 stay the same for this new problem. So, this allows

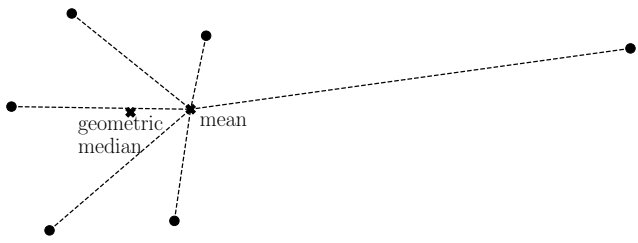

Figure 1. A center being affected by an outlier.

using the heuristic mentioned above. When calculating the new centers, the median in each dimension is taken and then combined. Also, the points of $\mathcal{S}$ are assigned according to the Voronoi diagram induced by the set of centers C with respect to the 1-norm distance metric. This variation of the k-means algorithm is usually called the k-medians algorithm [21].

An intermediate algorithm between k-means and k-medians can be developed making a small variation in the objective function. Let's call this new problem the EMAX problem and define it as follows:

Let $\mathcal{S}$ be a set of n points $\{x_1, x_2, \ldots, x_n\}$ in a metric space $(\mathbb{R}^d, \ell_2)$ where d defines the dimension space and ℓ_2 represents the Euclidean metric, and let $k > 1$ be an integer. Find a set $C = \{c_1, c_2, \ldots, c_k\}$ of k elements and a matrix $A = [a_{ij}]$ of size $n \times k$ such that the following is minimized:

$$J_2(C, A) = \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} \|x_i - c_j\|_2 \tag{8}$$

subject to:

$$c_j \in \mathbb{R}^d \text{ for all } j \tag{9}$$

$$a_{ij} \in \{0, 1\} \text{ for all } i, j \tag{10}$$

$$\sum_{j=1}^{k} a_{ij} = 1 \text{ for all } i. \tag{11}$$

As can be observed, the only difference between the k-means problem and this new one is the objective function. Function J_1 takes the within cluster sum of squared Euclidean distances. On the other hand, function J_2 takes the within cluster sum of Euclidean distances.

Until now, there was no formal definition of the EMAX problem or the algorithms used to solve it in the literature. However, the EMAX problem was informally approached in the past and some algorithms attempting to approximate it were implemented as part of clustering packages [8]. Also, a global optimization approach for the problem was proposed in [27] using Social Spider Optimization to directly approximate it. This algorithm will be used in the experiments with the name of SSO-A.

The EMAX problem can be handled in another way: the same heuristic used for the k-means problem can be used to approximately solve the EMAX problem. First, P_1 and P_2 can be defined in the same way:

- P_1: Fix $A = \hat{A}$ and solve the reduced problem $J_2(\hat{A}, C)$.
- P_2: Fix $C = \hat{C}$ and solve the reduced problem $J_2(A, \hat{C})$.

Then, as in the k-means algorithm, both sub-problems can be iteratively solved taking the solution from the previous iteration. As before, the way of minimizing J_2 is when the set C is fixed is the same as in the k-means algorithm.

Lemma 3.1 *Given the EMAX problem, let $\hat{C} = \{c_j\}$ be fixed. Then the function J_2 is minimized if and only if each point of $\mathcal{S}$ is assigned to the closest center of C to it. That is, only when matrix A has the following entries:*

$$a_{ij} = \begin{cases} 1 & \text{if } j = \arg\min_k \|x_i - c_k\|_2 \\ 0 & \text{otherwise} \end{cases} \quad \text{for all } i, j. \tag{12}$$

Proof ($\Longrightarrow$) Since the center assignment for a point is unique and independent of any other, if $J_2(\hat{C}, A)$ is minimum, then for each point x_i the following expression is minimum.

$$\sum_{j=1}^{k} a_{ij} \|x_i - c_j\|_2 \tag{13}$$

Then by (10) and (11), (13) is equal to $\|x_i - c_j\|_2$ for a specific j. Consequently, this can only be true when j gets the value that minimizes the expression.

($\Longleftarrow$) If $\|x_i - c_j\|_2$ is minimum, then for every x_i (13) is minimum because of (10) and (11). This implies that $J_2(\hat{C}, A)$ is minimum for every x_i. □

This solves sub-problem P_2. The way of solving problem P_1 is however, not so trivial.

Lemma 3.2 *Given the EMAX problem, let $\hat{A} = \{a_{ij}\}$ be fixed. Then the function J_2 is minimized if and only if each center of C is defined as the geometric median of the points of $\mathcal{S}$ assigned to it. That is, only if*

$$c_j \in \arg\min_{c \in \mathbb{R}^d} \sum_{i=1}^{n} a_{ij} \|x_i - c\|_2 \text{ for all } j. \tag{14}$$

Proof ($\Longrightarrow$) The location of a center does not affect the points that are not assigned to it, so a center is independent of any other. If $J_2(C, \hat{A})$ is minimum, then for every center c_j the following expression is also minimum:

$$\sum_{i=1}^{n} a_{ij} \|x_i - c_j\|_2. \tag{15}$$

Therefore, c_j must be a point that minimizes (15).

($\Longleftarrow$) If c_j is a center that minimizes (15), then the objective function is also minimum as it is the sum of independent minimums. So $J_2(C, \hat{A})$ is minimum for every c_j. □

As can be seen, solving sub-problem P_1 requires finding a point that minimizes the sum of Euclidean distances between a set of fixed points and it. This point is known as the geometric median. Finding the geometric median of a set of points is also known as the Fermat-Weber problem. In spite of the ancient nature of the problem, there are few theoretical guarantees to solve it. The problem was first studied in the XVII century by Pierre de Fermat for the case of three points [22], and

despite the elegant construction of the geometric median using ruler and compass by Evangelista Torricelli, there is not a similar construction for a larger number of points. In fact, it was shown that even for five points, the geometric median is not expressible by radicals over the rationals [5] and there is also no exact algorithm that solves this problem using arithmetic operations and kth roots. The approximation problem has been widely studied for a great number of points, giving as result many polinomial time algorithms for the approximate geometric median problem. In [9], a compilation of many of these algorithms is given and a nearly linear time algorithm is proposed for finding the geometric median of a set with arbitrary precision.

Many algorithms have been proposed as an approximation scheme for finding the geometric median. The majority of them exploits the fact that the sum of distances from the searched point to all the others is a convex function since the distance to a specific point is also convex. Also, it was proven that the geometric median is unique when the points are not collinear [26]. Since in most cases the function is convex and has a single minimum, using a searching algorithm that iteratively decreases the sum of distances can be thought of "safe" because it cannot get trapped in a local optimum and will probably reach the global minimum.

A simple and very popular search algorithm for finding the geometric median is Weiszfeld's algorithm [29]. The algorithm iteratively creates a new estimate of the geometric median. The algorithm fails to converge when one of its estimates falls on one of the points. In practice, the algorithm converges in almost all cases. Given a set $p_1, p_2, \ldots, p_t$ of t points, the algorithm first creates an initial estimate $c^{(0)}$ for the geometric median ensuring that this point is different from any of the given points. Then, the algorithm iteratively updates the current estimate $c^{(h)}$ to $c^{(h+1)}$ using the following rule:

$$c^{(h+1)} = \left(\sum_{i=1}^{t} \frac{p_i}{\|p_i - c^{(h)}\|_2} \right) \Bigg/ \left(\sum_{i=1}^{t} \frac{1}{\|p_i - c^{(h)}\|_2} \right). \tag{16}$$

An adopted approach in this work is using Weiszfeld's algorithm to calculate the geometric median. This algorithm can be replaced with any other that achieves the same objective. When using Weiszfeld's algorithm in the EMAX algorithm, the initial estimate for the geometric median is the mean of the given points.

Finally, a fairly natural criterion for the convergence of the algorithm is, like in the k-means algorithm, when clusters do not change anymore. The clusters can be easily recovered given the set C and the matrix A. The entire procedure of EMAX is shown in Algorithm 1.

Algorithm 1 The EMAX algorithm

Input: *A set* $\mathcal{S} \in (\mathbb{R}^d, \ell_2)$, k.
Output: *A clustering* $(C_1, C_2, \ldots, C_k)$ *of* $\mathcal{S}$.

1: $C \leftarrow$ *set of initial centers*
2: ***while*** *convergence criterion not reached* ***do***
3: *Assign each* $x_i \in \mathcal{S}$ *to* c_j *if* $j = \arg\min_k \|x_i - c_k\|_2$

4: **for** $j = 1$ **to** k **do**
5: *Let* $P = \{p_1, p_2, \ldots, p_m\}$ *be the set of points of* $\mathcal{S}$ *assigned to* c_j
6: $c_j \leftarrow$ *geometric median of* P
7: *Let* $(C_1, C_2, \ldots, C_k)$ *be* k *different empty clusters*
8: **for** i, j *such that* x_i *is assigned to* c_j **do**
9: *Place* x_i *in* C_j
return $(C_1, C_2, \ldots, C_k)$

The values of the centers declared in line 1 of the algorithm can be randomly initialized. As in k-means, the algorithm iteratively solves P_1 and P_2 assigning each point to its closest center and then re-calculating the center to be the geometric median (using Weiszfeld's algorithm) until the assignments do not change anymore. This last step aims to create a more robust algorithm than k-means algorithm when dealing with multiple outliers in the dataset. Problems k-means and EMAX can be simultaneously solved using multi-objective optimization. This is very suitable since both problems have the same parameters and conditions. The only difference between them is the objective function.

3.2 Multi-objective optimization

Solving a multi-objective optimization requieres very different approaches than those used to solve single objective problems. A common criterion when solving multi-objective optimization problems is to search for the Pareto frontier. The objective is to minimize J_1 and J_2 simultaneously subject to the same conditions. To illustrate the Pareto frontier, let's suppose there were found six feasible solutions to both problems: $s_1, \ldots, s_6$ having different values when evaluated with J_1 and J_2 as shown in Fig. 2.

In multi-objective optimization, it is not usual to find a feasible solution that

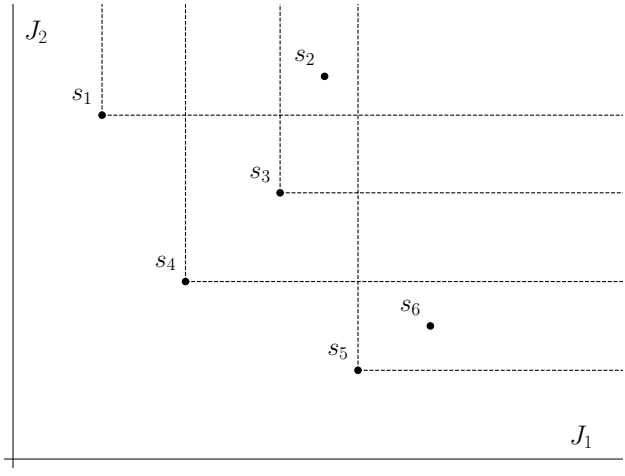

Figure 2. A set of feasible solutions for J_1 and J_2. The Pareto optimal solutions are s_1, s_4 and s_5 since they are not dominated by any other solution.

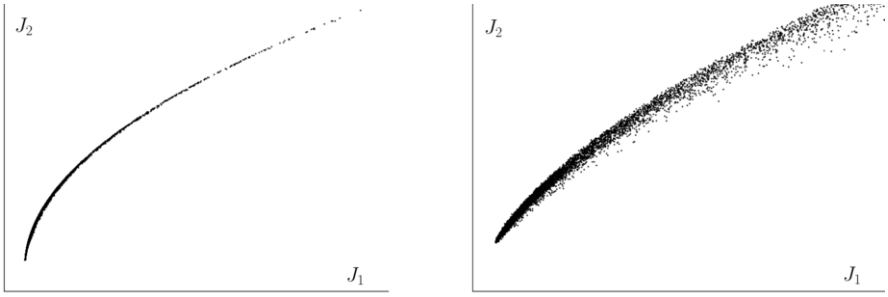

Figure 3. Left: behavior of solutions when evaluated by J_1 and J_2. Right: The same plot seen near minimum values.

minimizes all objectives (in this case, J_1 and J_2). Therefore, the task is to find the Pareto optimal solutions: those feasible solutions that cannot be improved by any other solution without degrading at least one of the other objectives. Coming back to the example, it is clear that s_1 is a better candidate when evaluated with the first objective, i.e. $J_1(s_1) < J_1(s_2)$. This is also true when s_1 is evaluated with the second one, i.e. $J_2(s_1) < J_2(s_2)$. Thus, we say that s_2 is dominated by s_1. We see also see that s_3 and s_2 are dominated by s_4 and that s_6 is dominated by s_5, but s_4 is not dominated by s_1 although it is a better candidate when evaluated with J_1, i.e. $J_1(s_1) < J_1(s_4)$ because s_4 is better when evaluated with J_2, i.e. $J_2(s_4) < J_2(s_1)$. The Pareto optimal solutions are those solutions that are not dominated by any other solution. In this example, the Pareto optimal solutions are s_1, s_4 and s_5.

In order to optimize J_1 and J_2 simultaneously, it is helpful to visualize a similar plot of feasible solutions. This plot is shown in Fig. 3.

As can be seen in the plot, those problems are highly related. In many cases, a solution near the minimum of both problems is not dominated by the majority of the other solutions. This fact simplifies the problem, as we can take the solutions that minimizes an objective that takes into account both objectives. This way, we can convert a multi-objective optimization problem into an optimization problem with a single objective. A common approach taken is to redefine the objective function taking a weighted sum of the objectives involved. So, the objective function that handles both problems is defined as:

$$\begin{aligned} \mathcal{F}(C, A) &= \alpha J_1 + \beta J_2 \\ &= \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} \left(\alpha \|x_i - c_j\|_2^2 + \beta \|x_i - c_j\|_2 \right). \end{aligned} \tag{17}$$

Where $\alpha, \beta > 0$ and $\alpha + \beta = 1$. Also, the conditions of the original problems stay the same. In this way, we can find good solutions for both problems with a single optimization task.

The problem, as is presented, is not suitable for a continuous optimization algorithm since the domain of search is not continuous and many optimization algorithms require the gradient of the function in each iteration. Some other also require

the value the function takes in specific points that, in this case, could not be defined. We can define the objective function just in terms of C using the observation that, as in previous cases, the optimal strategy is to assign each point x_i of $\mathcal{S}$ to the closest center of C to x_i. The proof of this fact is trivial and follows immediatly from Lemmas 2.2 and 3.1. So, from this point, the values of the elements of matrix A will permanently be defined similarly as in previous cases:

$$a_{ij} = \begin{cases} 1 & \text{if } j = \arg\min_k \|x_i - c_k\|_2 \\ 0 & \text{otherwise} \end{cases} \quad \text{for all } i, j. \tag{18}$$

The objective function is now totally dependent on C:

$$\mathcal{F}_1(C) = \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} \left(\alpha \|x_i - c_j\|_2^2 + \beta \|x_i - c_j\|_2 \right). \tag{19}$$

The Social Spider Optimization algorithm described in section 2.2 will be used for minimizing $\mathcal{F}_1$. To make the objective function suitable for the algorithm, the parameter C of the function can be "converted" into a single variable such that $\mathcal{F}_1 \colon \mathbb{R}^{kd} \to \mathbb{R}$. Each spider will be a kd-dimensional vector, being its structure:

$$(\underbrace{c_{11}, c_{12}, \ldots, c_{1d}}_{c_1}, \ldots, \underbrace{c_{k1}, c_{k2}, \ldots, c_{kd}}_{c_k}) \in \mathbb{R}^{kd}.$$

Being this a minimization problem, the values of $best_{\mathrm{s}}$ and $worst_{\mathrm{s}}$ (inherent of the algorithm) are re-defined in the following way:

$$best_{\mathrm{s}} = \min_{i=1}^{N} \mathcal{F}_1(\mathrm{s}_i), \tag{20}$$

$$worst_{\mathrm{s}} = \max_{i=1}^{N} \mathcal{F}_1(\mathrm{s}_i). \tag{21}$$

Being N the number of spiders in the colony. The lower and upper bounds for the position values will be chosen taking into account that the position of a center of C cannot be outside the limits established by the points of $\mathcal{S}$. We can handle this with the following rule: if c is a parameter of $\mathcal{F}_1$, then the minimum value of the ith component of c will be the minimum value of the component $((i-1) \mod k) + 1$ among all points of $\mathcal{S}$. Similarly, the maximum value of the ith component of c will be the maximum value of the component $((i-1) \mod k) + 1$ among all points of $\mathcal{S}$.

During the execution of the algorithm, the spiders generate vibrations that are perceived by other colony members. The vibration perceived by spider i as a result of the information emitted by spider j is modeled through (6). This behavior can be observed in Fig. 4.

As can be observed, the vibration intensity perceived by some spider decreases with the square Euclidean distance to the spider that generated it. Depending on the distances of the points in $\mathcal{S}$, the distances between pairs of spiders can

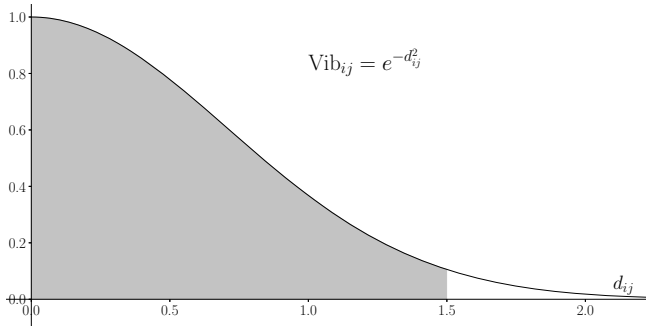

Figure 4. The behavior of the vibration intensity is a function of the squared Euclidean distance between spiders.

be significantly big; making communication impossible for them as the vibration intensities would effectively be zero. There is no direct approach to handle this problem in the original SSO algorithm. A proposed strategy to fix this problem is to consider the maximum distance possible when calculating the vibrations to be 1.5. For this to be true, each distance d_{ij} is replaced with d'_{ij} according to the following replacement:

$$d'_{ij} = 1.5 \cdot \frac{d_{ij}}{d_{\max}}. \tag{22}$$

Where $d_{\max}$ is the maximum distance between spider i and other spider in the colony. This normalization was chosen because approximately 80% of the area on the right of the vibration function is between 0 and 1.5.

A desirable property of the spiders is that they can cover a great part of the search space with the initialization. Depending on the initial positions, some spiders have greater probability than others of arriving to a local optimum. Initializing the positions of few spiders near some local optimums of a related function to $\mathcal{F}$ may improve convergence. To do this, and considering that outliers are the center of attention; we will assume that the set $\mathcal{S}$ has a natural property in its structure to solve a useful problem in facility location and clustering. The problem is called k-center and the property, Perturbation Resilience.

The k-center problem is to find a set $C = \{c_1, \ldots, c_k\} \subseteq \mathcal{S}$ of centers as to minimize:

$$J_3(C) = \max_{x \in \mathcal{S}} \left(\min_{c \in C} \|x - c\|_2 \right). \tag{23}$$

The objective function of the k-center problem is considerably different from the k-means and EMAX problems. The function is also dependent only on C as each point is to be assigned to the closest center. The set of centers must belong to $\mathcal{S}$, so it is a combinatorial optimization problem by its nature.

Perturbation resilience [7] is a notion of stability assumed for set $\mathcal{S}$. This property implies that if distances between points change due to a small perturbation, the optimal solution for k-means is the same. Formally, ρ is called an α-perturbation

of a distance function if for all $x, y \in \mathcal{S}$,

$$\|x - y\|_2 \leq \rho(x, y) \leq \alpha\|x - y\|_2. \tag{24}$$

Then, the pair $(\mathcal{S}, \ell_2)$ satisfies α-perturbation resilience for k-center if for any α-perturbation of the distance function, the optimal k-center clustering does not change.

This is a special natural assumption because, when the distances between points increases by a constant factor, the outlier effect becomes bigger, specially when evaluated with J_2. So, this assumption decreases this possibility.

The following theorem is helpful when trying to find optimal solutions for the k-center problem under perturbation resilience.

Theorem 3.3 ([6]) *Given a clustering instance $(\mathcal{S}, \ell_2)$ satisfying α-perturbation resilience for k-center, and a set C of k centers which is an α-approximation factor for k-center. Then the Voronoi partition induced by C is the optimal clustering.*

This result states that an α-approximation algorithm for k-center will give an optimal solution assuming perturbation resilience for $\mathcal{S}$ under the Euclidean distance metric. Exact and approximations algorithms have been designed for the k-center problem [1]. Optimal algorithms in approximation factor and run-time have also been found. In [17], it was shown that there is no polynomial time algorithm with an approximation factor less than 2 for the k-center problem unless P $\neq$ NP; also, a 2–approximation greedy algorithm for the problem was presented. The algorithm successfully finds a 2-approximation solution if the triangular inequality holds, which is the case for $(\mathcal{S}, \ell_2)$.

With this 2-approximation greedy algorithm we can generate solutions for the k-center problem as a local search strategy for some individuals of the initial population. The entire procedure of the SSO-C algorithm is shown in Algorithm 2.

Algorithm 2 The SSO-C algorithm

Input: *A set* $\mathcal{S} \in (\mathbb{R}^d, \ell_2), k, \alpha, n_s, n_{it}$.
Output: *A clustering* $(C_1, C_2, \ldots, C_k)$ *of* $\mathcal{S}$.

$\mathrm{s} \leftarrow (\underbrace{\overrightarrow{0}, \overrightarrow{0}, \ldots, \overrightarrow{0}}_{n_s})$
for $j = 1$ ***to*** k ***do***
  $C \leftarrow$ *Greedy k-center*$(\mathcal{S}, k)$
  $\mathrm{s}_i \leftarrow \overrightarrow{C}$
$\beta \leftarrow 1 - \alpha$
$\mathcal{F}_1(C) = \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} \left(\alpha\|x_i - c_j\|_2^2 + \beta\|x_i - c_j\|_2\right)$
$x \leftarrow SSO(\mathcal{F}_1(C), \mathrm{s}, n_s, n_{it})$
Get $C = \{c_1, c_2, \ldots, c_k\}$ *from* x
Let $(C_1, C_2, \ldots, C_k)$ *be* k *different empty clusters*
for i, j *such that* x_i *is assigned to* c_j ***do***
  Place x_i *in* C_j

return $(C_1, C_2, \ldots, C_k)$

The Algorithm 2 starts by creating n_s empty spiders in line 1. Lines 2 to 4 initialize the value of the first k spiders with a 2-approximation solution for the k-center problem. As the value of α is provided as input, the value of β is calculated in line 5. Then, in lines 6 and 7 the objective function is defined and the values of the empty spiders are initialized with uniform random values in the SSO algorithm. The global optimization is then performed and the best spider recovered. The clusters are then obtained in lines 10 and 11. The output of the algorithm is a clustering of set $\mathcal{S}$.

3.3 Datasets

For experimental purposes, a set of six synthetic datasets were generated. Their characteristics are detailed below. Key information of the datasets is shown in Table 1. A plot of the synthetic datasets can be seen in Fig. 5. Also, a set of seven real datasets were taken from the UCI Machine Learning Repository [14]. Some information about the real datasets is shown in Table 2.

The first synthetic dataset is relativeley simple for a basic clustering algorithm. The dataset contains three clearly separated clusters. The second synthetic dataset contains two close clusters with two far outliers. The third synthetic dataset contains nine clusters generated according to a multivariate Gaussian probability distribution using the mean μ and covariance matrix Σ as specified in Table 1. The fourth synthetic dataset contains three clusters generated according to a Gamma distribution for each dimension using the shape a and rate b as specified in Table 1. Finally, the fifth and sixth datasets were generated according to the multivariate Gaussian probability distribution using the μ and Σ shown in Table 1.

4 Experiments

The four algorithms: k-means, EMAX, SSO-A [27], and SSO-C were evaluated with six synthetic and seven real datasets from UCI Machine Learning Repository [14]: Iris, Vowel Indian, Crude Oil, Balance Scale, Breast Cancer, and Wine.

The value of the parameter α in the SSSO-C algorithm was fixed to be 0.2. This way, the algorithm gives the second problem about 80% of the total priority. For evaluating the cluster quality, the Adjusted Mutual Information metric [28] was used with the labels predicted by the clustering algorithms and the true labels. The range of the metric varies from -1 to 1, being 0 a random prediction of the clusters and 1 a perfect prediction (negative values can arise in certain circumstances). For each algorithm, a total of fifty executions were performed on every dataset. For each dataset, the maximum, median, and mean values (using the AMI metric) of the executions are shown in Table 3, where the best values of the row are highlighted in bold.

Numerical experiments were conducted using the C Language (gcc version 8.1.0

	Space	Points per cluster		
Data 1	$\mathbb{R}^2$	Cluster 1	50 points	
		Cluster 2	50 points	
		Cluster 3	50 points	
Data 2	**Space**	**Points per cluster**		
	$\mathbb{R}^2$	Cluster 1	40 points	
		Cluster 2	40 points	
Data 3	**Space**	**Points per cluster**	$\boldsymbol{\mu}$	$\boldsymbol{\Sigma}$
	$\mathbb{R}^2$	Cluster 1	100 points	$(-2, 2)$	$\begin{bmatrix} 1.69 & 0 \\ 0 & 1.69 \end{bmatrix}$
		Cluster 2	100 points	$(0, 2)$	
		Cluster 3	100 points	$(2, 2)$	
		Cluster 4	100 points	$(-2, 0)$	
		Cluster 5	100 points	$(0, 0)$	
		Cluster 6	100 points	$(2, 0)$	
		Cluster 7	100 points	$(-2, -2)$	
		Cluster 8	100 points	$(0, -2)$	
		Cluster 9	100 points	$(2, -2)$	
Data 4	**Space**	**Points per cluster**	$\boldsymbol{a}$	$\boldsymbol{b}$
	$\mathbb{R}^2$	Cluster 1	100 points	2	$1/2$
		Cluster 2	100 points	2	1
		Cluster 3	100 points	2	$2/3$
Data 5	**Space**	**Points per cluster**	$\boldsymbol{\mu}$	$\boldsymbol{\Sigma}$
	$\mathbb{R}^2$	Cluster 1	200 points	$(1, 1)$	$\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$
		Cluster 2	200 points	$(2, 2)$	
		Cluster 3	200 points	$(3, 3)$	
		Cluster 4	200 points	$(1, 3)$	
		Cluster 5	200 points	$(3, 1)$	
Data 6	**Space**	**Points per cluster**	$\boldsymbol{\mu}$	$\boldsymbol{\Sigma}$
	$\mathbb{R}^2$	Cluster 1	400 points	$(2, 2)$	$\begin{bmatrix} 1.44 & 0 \\ 0 & 1.44 \end{bmatrix}$
		Cluster 2	400 points	$(6, 6)$	
		Cluster 3	400 points	$(6, 2)$	

Table 1
Information about the synthetic datasets

Dataset	**Iris**	**Vowel**	**Oil**	**Balance**	**Cancer**	**Wine**	**Glass**
Dimension	4	3	5	4	30	13	8
Clusters	3	6	3	3	2	3	6
Points	150	871	56	625	569	178	214

Table 2
Information about the real datasets

built by MinGW-W64) on a 3.40GHz Intel Core i7-6700 with 16GB of RAM memory and running Windows 10 version 1809 as operating system.

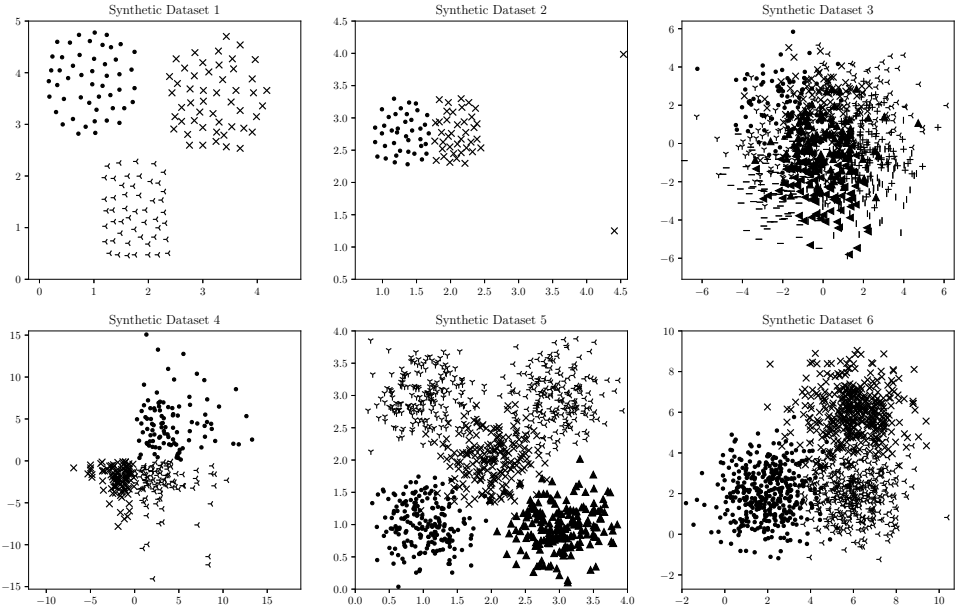

Figure 5. Synthetic datasets used to perform the experiments.

Synthetic datasets						Real datasets					
Dataset		**SSO-A**	***k*-means**	**EMAX**	**SSO-C**	**Dataset**		**SSO-A**	***k*-means**	**EMAX**	**SSO-C**
S. Dataset 1	Maximum	**1.000**	**1.000**	**1.000**	**1.000**	Iris	Maximum	0.793	0.748	0.787	**0.829**
	Median	**1.000**	**1.000**	**1.000**	**1.000**		Median	0.713	0.748	0.757	**0.762**
	Mean	**1.000**	**1.000**	**1.000**	**1.000**		Mean	0.716	0.743	0.763	**0.765**
S. Dataset 2	Maximum	**1.000**	0.854	**1.000**	**1.000**	Vowel Indian	Maximum	0.517	0.477	0.509	**0.517**
	Median	**1.000**	0.802	**1.000**	**1.000**		Median	0.451	0.459	**0.490**	0.453
	Mean	0.964	0.813	0.664	**1.000**		Mean	0.445	0.463	**0.493**	0.452
S. Dataset 3	Maximum	0.353	0.341	0.352	**0.366**	Crude Oil	Maximum	0.250	0.236	**0.272**	0.250
	Median	0.336	0.340	0.342	**0.342**		Median	0.204	0.182	**0.272**	0.182
	Mean	0.336	0.338	**0.343**	0.343		Mean	0.201	0.190	**0.249**	0.189
S. Dataset 4	Maximum	0.806	0.770	0.813	**0.814**	Balance Scale	Maximum	**0.274**	0.160	0.134	0.253
	Median	0.761	0.770	**0.813**	0.790		Median	**0.214**	0.115	0.117	0.109
	Mean	0.719	0.770	**0.813**	0.784		Mean	**0.223**	0.106	0.114	0.113
S. Dataset 5	Maximum	0.888	0.885	0.882	**0.898**	Breast Cancer	Maximum	0.453	0.422	**0.458**	0.440
	Median	0.863	0.881	0.882	**0.884**		Median	0.351	0.422	**0.458**	0.374
	Mean	0.852	0.882	**0.882**	0.882		Mean	0.322	0.422	**0.458**	0.369
S. Dataset 6	Maximum	0.770	0.756	0.758	**0.779**	Wine	Maximum	**0.432**	0.423	0.427	**0.432**
	Median	0.752	0.754	0.755	**0.755**		Median	0.413	**0.423**	0.419	**0.423**
	Mean	0.748	0.754	0.755	**0.757**		Mean	**0.419**	0.408	0.414	0.417
						Glass	Maximum	0.301	0.316	0.319	**0.389**
							Median	0.177	0.274	**0.319**	0.318
							Mean	0.165	0.290	0.319	**0.320**

Table 3
Results of the experiments performed over the four algorithms

5 Discussion

From Table 3, the following can be observed: EMAX has better results than *k*-means in almost all datasets regarding maximum, median, and mean values; SSO-C has better results than SSO-A in almost all datasets regarding maximum, median,

Dataset	Control	i	Algorithm	Rank	p-value	α/i	Dataset	Control	i	Algorithm	Rank	p-value	α/i
	SSO-C	3	***k*-means**	3.63	1.29E-14	0.017		EMAX	3	**SSO-A**	3.18	1.48E-15	0.017
S. Dataset 2	(Rank:	2	**EMAX**	2.49	9.95E-04	0.025	Vowel	(Rank:	2	**SSO-C**	2.96	1.03E-12	0.025
	1.64)	1	**SSO-A**	2.24	2.00E-02	0.050		1.12)	1	***k*-means**	2.74	3.51E-10	0.050
	SSO-C	3	**SSO-A**	3.08	4.04E-05	0.017		EMAX	3	***k*-means**	3.17	4.30E-17	0.017
S. Dataset 3	(Rank:	2	***k*-means**	2.76	4.00E-03	0.025	Oil	(Rank:	2	**SSO-C**	3.17	4.30E-17	0.025
	2.02)	1	EMAX	2.14	6.40E-01	0.050		1.00)	1	**SSO-A**	2.66	1.28E-10	0.050
	EMAX	3	**SSO-A**	3.23	1.59E-16	0.017		SSO-A	3	***k*-means**	3.34	2.58E-19	0.017
S. Dataset 4	(Rank:	2	***k*-means**	3.22	2.20E-16	0.025	Balance	(Rank:	2	**SSO-C**	3.02	9.49E-15	0.025
	1.10)	1	**SSO-C**	2.45	1.71E-07	0.050		1.02)	1	**EMAX**	2.62	5.76E-10	0.050
	EMAX	3	**SSO-A**	3.56	4.58E-11	0.017		EMAX	3	**SSO-A**	3.32	2.58E-19	0.017
S. Dataset 5	(Rank:	2	***k*-means**	2.54	8.00E-03	0.025	Cancer	(Rank:	2	**SSO-C**	3.23	5.78E-18	0.025
	1.86)	1	SSO-C	2.04	4.80E-01	0.050		1.00)	1	***k*-means**	2.45	1.96E-08	0.050
	EMAX	3	***k*-means**	2.87	3.16E-04	0.017		SSO-A	3	***k*-means**	3.09	6.54E-04	0.017
S. Dataset 6	(Rank:	2	**SSO-A**	2.86	3.66E-04	0.025	Wine	(Rank:	2	EMAX	2.47	3.10E-01	0.025
	1.94)	1	SSO-C	2.33	1.30E-01	0.050		2.21)	1	SSO-C	2.23	9.30E-01	0.050
	SSO-C	3	**SSO-A**	3.51	2.76E-15	0.017		EMAX	3	**SSO-A**	3.92	3.39E-21	0.017
Iris	(Rank:	2	***k*-means**	3.22	1.22E-11	0.025	Glass	(Rank:	2	***k*-means**	2.66	4.87E-06	0.025
	1.47)	1	EMAX	1.80	2.00E-01	0.050		1.48)	1	SSO-C	1.94	7.00E-02	0.050

Table 4
Results of the Holm Test

and mean values. When all algorithms are considered, SSO-C showed bests results in almost all datasets regarding the maximum values. Also, for median and means values, EMAX and SSO-C have the best results the same number of cases (7 out of 13).

Statistical tests were performed (according to [11], see also [16] and [12]) over the results of the algorithms (samples of 50 elements) in the following way: (a) First, the Friedman test is performed to test the null hypothesis that algorithms, used in the experiments, have the same performance, (b) When the null hypothesis of Friedman test is rejected, then the Holm test is performed to test the null hypothesis that a control algorithm has the same performance regarding some other algorithm.

For performing the statistical test the `CONTROLTEST` package (https://sci2s.ugr.es/sicidm) was used with a significance level of $\alpha = 0.05$ as default. The null hypothesis of the Friedman test was rejected in all datasets, except in the Synthetic Dataset 1. Table 4 presents the results of the Holm test, where algorithms in bold represent those cases where the null hypothesis was rejected.

From Table 4, the following can be observed: EMAX is the control algorithm in 7 cases out of 12, then, when EMAX is compared to SSO-C, the null hypothesis is rejected just in 4 cases out of 7, that is, EMAX is the best performing algorithm with a statistically significant difference in just 4 cases.

6 Conclusions

In this paper, two algorithms were presented. In the first one (EMAX algorithm), a similar heuristic used in the k-means algorithm is used to solve the EMAX problem. The second one (SSO-C algorithm) is based on the Social Spider Optimization algorithm for approaching multi-objective problems. The SSO-C algorithm is used to minimize the weighted sum of two objective functions: those defined in the k-

means and EMAX problems. For the SSO-C algorithm, an approximation algorithm (for the k-center problem) was used in order to initialize a small part of the initial population.

Results of the experiment showed that the EMAX algorithm outperforms the k-means algorithm, also, the SSO-C algorithm outperforms the SSO-A algorithm. When all algorithms were compared, SSO-C showed to be suitable for finding best maximum values, but when it comes to median and mean values, EMAX is the one with best scores. This led us to perform statistical tests (Friedman and Holm tests) in order to discover which one is better: SSO-C or EMAX. In the results of the Holm test, EMAX appears as the control algorithm in 7 cases out of 12, of which, EMAX has statistically significant difference regarding SSO-C in just 4 cases.

As a future work, we plan to perform several experiments with more sophisticated datasets, with higher dimensionality and size. Also, we plan to develop other initialization procedures (such as Opposition-Based Learning [25,3]) for SSO-C in order to improve convergence results and robustness. For the EMAX problem, we plan to prove convergence and NP Hardness.

References

[1] Agarwal, P. K. and C. M. Procopiuc, *Exact and approximation algorithms for clustering*, Algorithmica **33** (2002), pp. 201–226.

[2] Aggarwal, C. C. and C. K. Reddy, "Data Clustering: Algorithms and Applications," Chapman & Hall/CRC, 2013, 1st edition.

[3] AlQunaieer, F., H. Tizhoosh and S. Rahnamayan, *Opposition based computing a survey*, in: *Proc. IEEE Int. Conf. Neural Networks*, 2010, pp. 1098–7576.

[4] Arthur, D. and S. Vassilvitskii, *K-means++: The advantages of careful seeding*, in: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07 (2007), pp. 1027–1035.

[5] Bajaj, C., *The algebraic degree of geometric optimization problems*, Discrete & Computational Geometry **3** (1988), pp. 177–191.

[6] Balcan, M.-F., N. Haghtalab and C. White, *k-center clustering under perturbation resilience*, in: *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, Leibniz International Proceedings in Informatics (LIPIcs) **55** (2016), pp. 68:1–68:14.

[7] Bilu, Y. and N. Linial, *Are stable instances easy?*, Comb. Probab. Comput. **21** (2012), pp. 643–660.

[8] Cardot, H., *Fast algorithms for robust estimation with large samples of multivariate observations. estimation of the geometric median, robust k-gmedian clustering, and robust pca based on the gmedian covariation matrix.* (2017).

[9] Cohen, M. B., Y. T. Lee, G. Miller, J. Pachocki and A. Sidford, *Geometric median in nearly linear time*, in: *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16 (2016), pp. 9–21.

[10] Cuevas, E., M. A. Díaz Cortés and D. A. Oliva Navarro, "A Swarm Global Optimization Algorithm Inspired in the Behavior of the Social-Spider," Springer International Publishing, 2016 pp. 9–33.

[11] Demšar, J., *Statistical comparisons of classifiers over multiple data sets*, The Journal of Machine Learning Research **7** (2006), pp. 1–30.

[12] Derrac, J., S. García, D. Molina and F. Herrera, *A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms*, Swarm and Evolutionary Computation **1** (2011), pp. 3–18.

[13] Du, K.-L. and M. N. S. Swamy, "Introduction," Springer International Publishing, 2016 pp. 1–28.

[14] Dua, D. and E. Karra Taniskidou, *UCI machine learning repository* (2017).

[15] Fortune, S., *Handbook of discrete and computational geometry*, in: *Computing in Euclidean Geometry – Lecture Notes Series on Computing – Vol. 1*, CRC Press, Inc., 1997 pp. 377–388.

[16] García, S. and F. Herrera, *An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons*, Journal of Machine Learning Research **9** (2008), pp. 2677–2694.

[17] Gonzalez, T. F., *Clustering to minimize the maximum intercluster distance*, Theoretical Computer Science **38** (1985), pp. 293–306.

[18] Hartigan, J. A. and M. A. Wong, *Algorithm AS 136: A K-Means clustering algorithm*, Applied Statistics **28** (1979), pp. 100–108.

[19] Hautamäki, V., S. Cherednichenko, I. Kärkkäinen, T. Kinnunen and P. Fränti, *Improving k-means by outlier removal*, in: *Image Analysis* (2005), pp. 978–987.

[20] Huang, Z., *Extensions to the k-means algorithm for clustering large data sets with categorical values*, Data Mining and Knowledge Discovery **2** (1998), pp. 283–304.

[21] Jain, A. K. and R. C. Dubes, "Algorithms for Clustering Data," Prentice-Hall, Inc., 1988.

[22] Krarup, J. and S. Vajda, *On Torricelli's geometrical solution to a problem of Fermat*, IMA Journal of Management Mathematics **8** (1997), pp. 215–224.

[23] Lloyd, S., *Least squares quantization in pcm*, IEEE Transactions on Information Theory **28** (1982), pp. 129–137.

[24] Mahajan, M., P. Nimbhorkar and K. Varadarajan, *The planar k-means problem is NP-hard*, Theoretical Computer Science **442** (2012), pp. 13 – 21.

[25] Tizhoosh, H. and M. Ventresca, "Oppositional Concepts in Computational Intelligence," Springer Berlin Heidelberg, 2008.

[26] Vardi, Y. and C.-H. Zhang, *The multivariate l1-median and associated data depth*, Proceedings of the National Academy of Sciences **97** (2000), pp. 1423–1426.

[27] Vera Olivera, H., J. L. Soncco-Álvarez and L. Enciso-Rodas, *Social spider algorithm approach for clustering*, in: *Proceedings of the 3rd Annual International Symposium on Information Management and Big Data*, 2016, pp. 114–121.

[28] Vinh, N. X., J. Epps and J. Bailey, *Information theoretic measures for clusterings comparison: Is a correction for chance necessary?*, in: *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09 (2009), pp. 1073–1080.

[29] Weiszfeld, E., *Sur le point pour lequel la somme des distances de n points donnes est minimum*, Tohoku Mathematical Journal, First Series **43** (1937), pp. 355–386.