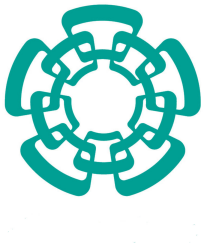


Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

CINVESTAV Tamaulipas

Método de clustering basado en optimización de índices de validez

Tesis que presenta:

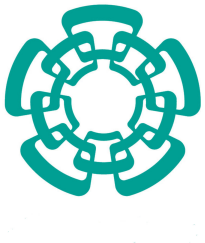
Melesio Crespo Sánchez

Para obtener el grado de:

**Maestro en Ciencias
en Ingeniería y Tecnologías
Computacionales**

Dr. Iván López Arévalo, Co-Director
Dr. Edwyn Javier Aldana Bobadilla, Co-Director

Cd. Victoria, Tamaulipas, México. Septiembre, 2017



Center for Research and Advanced Studies
of the National Polytechnic Institute

CINVESTAV Tamaulipas

A Clustering Method Based on Validity Indices Optimization

Thesis by:

Melesio Crespo Sánchez

as the fulfillment of the
requirement for the degree of:

**Master of Science
in Engineering and Computing
Technologies**

Dr. Iván López Arévalo, Co-Advisor
Dr. Edwyn Javier Aldana Bobadilla, Co-Advisor

Cd. Victoria, Tamaulipas, México. September, 2017

The thesis of Melesio Crespo Sánchez is approved by:

_____

Dr. Víctor Jesús Sosa Sosa

Dr. Ricardo Landa Becerra

Dr. Iván López Arévalo, Committee Co-advisor

Dr. Edwyn Javier Aldana Bobadilla, Committee Co-advisor

Cd. Victoria, Tamaulipas, México., September 19 2017

All we have to decide is what to do with the time that is given us.
Gandalf the grey.

Acknowledgements

- I especially thank my wife for her unconditional support and love in difficult times during this stage of my life. Thank you for cheering me up when things got tough and thought I could not achieve this.
- I also thank my advisors Dr. Iván López Arévalo and Dr. Edwyn Javier Aldana Bobadilla for guiding me during this work and thanks to the administrative personnel at CINVESTAV-Tamaulipas for their help during my stay.
- Thanks to my reviewers Dr. Victor Jesús Sosa Sosa and Dr. Ricardo Landa Becerra for their comments that served as feedback to improve this work.
- I thank CONACyT for the provided economic support which allowed me to concentrate in my studies and CINVESTAV-Tamaulipas for the opportunity to pursue graduate studies.

Contents

Contents i

List of Figures iii

List of Tables v

List of Algorithms vii

Resumen ix

Abstract xi

Nomenclature xiii

1 Introduction 1
- 1.1 Context . . . 1
- 1.2 Hypothesis . . . 3
- 1.3 Objectives . . . 3
- 1.4 Document organization . . . 3

2 Background 5
- 2.1 Clustering methods . . . 5
  - 2.1.1 Probabilistic-based models . . . 6
  - 2.1.2 Distance-based clustering . . . 7
    - 2.1.2.1 Flat clustering . . . 8
    - 2.1.2.2 Hierarchical clustering . . . 9
  - 2.1.3 Density-based clustering . . . 10
- 2.2 Validity indices . . . 12
  - 2.2.1 External indices . . . 13
  - 2.2.2 Internal indices . . . 14
- 2.3 Meta-heuristic optimization methods . . . 15
  - 2.3.1 Basic concepts in meta-heuristic techniques . . . 16
    - 2.3.1.1 Encoding . . . 16
    - 2.3.1.2 Objective function . . . 17
    - 2.3.1.3 Constraint handling . . . 17
  - 2.3.2 Single-solution based meta-heuristics . . . 18
    - 2.3.2.1 Basic concepts . . . 18
  - 2.3.3 Common single-solution based meta-heuristics . . . 19
  - 2.3.4 Population-based meta-heuristics . . . 20
    - 2.3.4.1 Basic concepts . . . 21

2.3.4.2 Evolutionary approaches . . . 22
2.3.5 Common population-based meta-heuristics . . . 22
2.4 Related work . . . 24

3 Methodology **27**
3.1 Defining the objective function . . . 29
3.2 Choosing a suitable meta-heuristic . . . 30
3.3 Problem encoding for clustering . . . 34
3.3.1 Binary encoding . . . 35
3.3.2 Integer encoding . . . 36
3.3.3 Real encoding . . . 37
3.3.4 Proposed encoding . . . 37
3.3.5 Determining the membership of $\vec{x}$ . . . 38
3.4 Clustering proposal . . . 39
3.5 Preliminary experiments . . . 41
3.5.1 Meta-heuristics parameters . . . 48
3.5.2 Algorithm's complexity . . . 50

4 Experimental process and results **53**
4.1 Generating synthetic datasets . . . 53
4.2 Experimental design . . . 55
4.3 Results and analysis . . . 56
4.3.1 Complementary analysis . . . 62

5 Conclusions and future work **67**

A Optimization problems **69**

B Validity indices **79**
B.1 Dunn and Dunn (DD) index . . . 79
B.2 Davies-Bouldin (DB) index . . . 80
B.3 Scattering and Dispersion (SD) index . . . 81

C Algorithms **83**

List of Figures

2.1 EM clustering result. . . . 7
2.2 Example of flat clustering (k-means algorithm). a) Input dataset X, b) Selection of representative elements for 3 clusters, c) Assignment of $\vec{x}$ to its nearest cluster C_i. d) Adjustment of the representative elements for clusters C_i. . . . 8
2.3 Cluster dendogram. . . . 9
2.4 Distance based CM applied to a non spherical dataset. . . . 11
2.5 Different Π proposed by a CM variating k. . . . 12
2.6 Single-solution based meta-heuristics approach. . . . 18
2.7 Local and global optimum values in a search space for a minimization problem. . . . 19
2.8 Population based meta-heuristics approach. . . . 21

3.1 Proposed clustering method. . . . 28
3.2 Count of the best fitness obtained 33
3.3 Performance of the selected meta-heuristic for unconstrained problems. The results show a different magnitude orders which are expressed as log_{10}. . . . 33
3.4 Performance of the selected meta-heuristic for constrained problems. The results show a different magnitude orders which are expressed as log_{10}. . . . 34
3.5 Encoded solutions positioned in the landscape of a two dimensional objective function. 35
3.6 Binary encoding. . . . 36
3.7 Integer encoding. . . . 36
3.8 Real number encoding. . . . 37
3.9 Membership function encoding with $r = 1, d = 2, k = 3$. . . . 38
3.10 Label decoding. . . . 39
3.11 Two dimensional dataset X 42
3.12 Instances of Π at iterations 1, 5, 18, 34, 38, and 47. . . . 43
3.13 Elements $\vec{x}$ of X projected in f_i. . . . 44
3.14 Discrimination after applying ρ. . . . 45
3.15 All membership functions determine Π. . . . 45
3.16 Π obtained by using the DB index. . . . 46
3.17 Two dimensional dataset X. . . . 47
3.18 . . . 47
3.19 Top 5 $\aleph$ performances for EGA. . . . 49
3.20 Top 5 $\aleph$ performances for DE. . . . 49

4.1 Examples of synthetic datasets in $\mathbb{R}^2$ with different values of k 55
4.2 Dunn and Dunn index comparative results. . . . 59
4.3 Davies-Bouldin index comparative results. . . . 60
4.4 SD index comparative results. . . . 61
4.5 Number of clusters obtained with Kmeans. . . . 62

4.6 Number of clusters obtained with SOM. 63
4.7 Number of clusters obtained with MCM-DE. 64
4.8 Number of clusters obtained with the MCM-EGA. 65

List of Tables

3.1 RMHC parameter settings 31
3.2 SA parameter settings 31
3.3 Holland parameter settings. 31
3.4 DE parameter settings. 32
3.5 EGA parameter settings. 32
3.6 Meta-heuristic parameter settings. 50

4.1 Dunn and Dunn index results. 57
4.2 Davies-bouldin index results. 57
4.3 SD index results. 58

List of Algorithms

1 Clustering algorithm based on validity index optimization. 41
2 Procedure initialize population. 83
3 Procedure evaluate population. 84
4 Procedure decode. 85
5 Procedure penalize. 86

Resumen

Método de clustering basado en optimización de índices de validez

por

Melesio Crespo Sánchez
CINVESTAV Tamaulipas
Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, 2017
Dr. Edwyn Javier Aldana Bobadilla, Co-Director
Dr. Iván López Arévalo, Co-Director

Clustering es una tarea importante en análisis de datos que nos permite encontrar aquellos elementos en un dataset no etiquetado que comparten propiedades en común, comúnmente llamados clusters. El proceso de encontrar dichos clusters es conocido como método de clustering. Dicho proceso es guiado típicamente por un criterio de similitud basado en una métrica o medida de distancia. La bondad de los resultados de clustering son evaluados típicamente mediante medidas de calidad conocidas como índices de validez. Un índice de validez mide las propiedades deseadas de los clusters. En este trabajo proponemos un método de clustering que es guiado directamente mediante un índice de validez. Ya que este enfoque implica un amplio espacio de soluciones, hacemos uso de una meta-heurística apropiada que explora dicho espacio eficientemente. Nuestra propuesta de clustering es invariante a un índice de validez en particular, esto resulta en un método generalizado de clustering que puede encontrar la mejor partición relativa a un amplio espectro de índices. A diferencia de métodos de clustering tradicionales, nuestra propuesta no carece de capacidad explicativa, ya que provee un modelo matemático para cada subconjunto en la partición, que nos permite asignar nuevos elementos al subconjunto más apropiado sin realizar nuevamente el proceso de búsqueda.

Abstract

A Clustering Method Based on Validity Indices Optimization

by

Melesio Crespo Sánchez
CINVESTAV Tamaulipas
Center for Research and Advanced Studies of the National Polytechnic Institute, 2017
Dr. Edwyn Javier Aldana Bobadilla, Co-advisor
Dr. Iván López Arévalo, Co-advisor

Clustering is an important task in data analysis that allows to find those elements in an unlabeled dataset that share common properties, the so-called clusters. The process to find such clusters is known as clustering method. Such process is typically guided by a similarity criterion based on a metric or proximity measure. The goodness of the clustering results are typically evaluated via a quality measure known as validity index. A validity index measures several desired properties of the clusters. We propose a clustering method in which the search process is directly guided by a validity index. Since this approach implies a large space solution, we use an appropriate meta-heuristic to efficiently explore such space. Our proposal is invariant to a particular validity index, this feature results in a generalized clustering method that can find the best partition relative to a wide spectrum of indices. Unlike traditional clustering methods, our proposal does not lack an explanatory capability, since it provides a mathematical model of each subset in the partition, that allows us to assign a new element to the most suitable subset without performing the search process again.

Nomenclature

CM	Clustering method
MCM	Meta-heuristic Clustering method
DE	Differential evolution
EGA	Eclectic genetic algorithm
TGA	Holland genetic algorithm with elitism
RMHC	Random mutation hill climbing
SA	Simulated annealing
$\vec{x}$	Numeric pattern
X	Numeric pattern dataset
Π	Partition of X (Clustering solution)
d	Dimensions of X
k	Number of clusters in Π
$\vec{s}$	Candidate solution
S	Set of candidate solutions
Q	Validity index
DB	Davies bouldin index
DD	Dunn and Dunn index
SD	SD index

1 Introduction

Nowadays, the advances on technology allow us to accumulate large amounts of data in different formats (e.g. plain text, structured databases, images, video, audio) which require to be stored, processed and analyzed, in order to obtain information and knowledge to support strategic decisions. The analysis of data implies to find a model that represents the behavior of a phenomenon. In order to find a model, many approaches have arisen. Several approaches make use of statistic methods as inference [86], fitting density function [81, 86], and regression [86].

1.1 Context

Other ways to find a suitable model that represents a phenomenon is via machine learning, in which there are many computational techniques that emulate a learning process from previous experience represented as an input dataset. Such learning can be based on two approaches: supervised and unsupervised [80] .

In a *supervised approach* a labeled dataset is used. The learning process is an iterative process

which finds a model whose outputs are as close as possible to the class labels of the dataset. It means, those outputs that minimize the error relative to prior information.

The *unsupervised approach*, aims to find those elements in an unlabeled dataset with common properties. This involves a similarity criterion, usually determined by a metric or proximity measure (e.g. Euclidean distance [2], Bhattacharyya [1], Mahalanobis [69]). Those elements that share common properties are called clusters. The process to find such clusters is known as clustering method (CM) [24, 54, 61, 80].

In the literature there are different CMs which are usually grouped into the following categories:

- Probabilistic-based models, such as Expectation Maximization *EM* [66].
- Distance-based methods, such as *k-means*, *fuzzy c-means*, *SLINK*, *CLINK*, *BIRCH*. [24, 38, 41, 42, 43, 72, 77, 80].
- Density-based methods, like *DBSCAN* [24, 60, 80].
- Meta-heuristic clustering [4, 11, 17, 91].

In most clustering methods, the number k of clusters must be explicitly specified. Choosing an appropriate value for k has important effects on the clustering results. An adequate selection of k should consider the shape and several desired properties of clusters. Although there is not a generally accepted approach to this problem, many methods attempting to solve it have been proposed [54, 63, 82].

Another concern is to determine the goodness of a clustering solution. Usually, the clusters are found via a CM and then evaluated via a quality criterion. Such criterion can be expressed as a mathematical function known as validity index [8, 61, 70, 73], which attempts to measure several desired properties of clusters.

1.2 Hypothesis

In this work, we hypothesize the next:

It is possible to find clusters by directly optimizing a given validity index rather than finding a set of clusters that optimize a similarity criteria and evaluating them later.

This involves a large solution space that must be explored efficiently, such large space can be expressed as the Stirling number of second kind [71]. In this regard we resort to meta-heuristic techniques.

1.3 Objectives

Based on the above, our main goal is to obtain a clustering method based on validity indices optimization to obtain the best clustering solution on an unlabeled numerical dataset X, in which case is mandatory:

1. To represent the clustering problem as an optimization problem.
2. To define an appropriate meta-heuristic to explore the large space of all possible solutions of the problem.

1.4 Document organization

The rest of this document is organized as follows: In Chapter 2 the background concepts and related works over this proposal are shown. In Chapter 3 the design of the proposed clustering method is described. Chapter 4 shows the experimental process and analysis over the obtained results. Finally, in Chapter 5 we point out the conclusions, advantages, disadvantages and future work of this clustering proposal.

2

Background

The faced problem in this research aims at two widely explored fields of computer science (Clustering and Optimization techniques) . This chapter describes several important concepts about different approaches of CMs and how the results of such methods are evaluated to have a quality measure of them via validity indices. Following our objectives, we review several approaches that tackle the clustering problem as an optimization problem, specially those that use meta-heuristic methods to explore the solution space.

2.1 Clustering methods

Clustering is considered a hard problem due to the lack of prior information about the structure of X and the membership of all elements $\vec{x} \in X$ to a given cluster. CMs have a broad field of applications such as: data summarization [7], learning, image segmentation [14], marketing [36], outlayer detection [31], biological data analysis [89], etc. In general the clustering problem can be stated as follows:

Given a dataset X to be clustered, clustering is a process that allows us to find a partition Π of the space of X into k regions, such that the elements that belong to them satisfy a similarity criterion.

Different approaches have been developed in this field, for instance: based on probabilistic models, distance or density based. Some of the commonly found clustering approaches are described below.

2.1.1 Probabilistic-based models

In this clustering approach the CM attempts to model clusters via a generative process which finds a Π based on a probabilistic model. At the beginning a specific probabilistic model is taken as reference (e.g. a mixture of Gaussian models) and at subsequent steps the parameters of the model are estimated using the Expectation Maximization (EM) algorithm [66].

The base of this approach is to obtain a mixture of probability distributions $\mathbb{P} = \{p_1, \ldots, p_k\}$ form which the elements in X are generated. The process is as follows:

- **Expectation step**. Determine the expected probability to assign $\vec{x}$ to each cluster using the current model parameters.

- **Maximization step**. Determine the optimal model parameters of each mixture of probabilities by using the obtained expected probabilities in the expectation step represented as weights α_i.

An example of this approach is illustrated in Figure 2.1, where a mixture of Gaussian models is used to generate a Π of 3 clusters over a dataset X with two dimensions (x_1, x_2).

The main advantage of the EM-models is that they can be used with different data types as long as the generative model is right chosen from the individual mixture components $\mathbb{P}$. Examples of this can be numerical data (by using a mixture of Gaussian [86]), categorical data (by using a Bernoulli model [86]) and time series data (by using a Hidden Markov Model [9]).

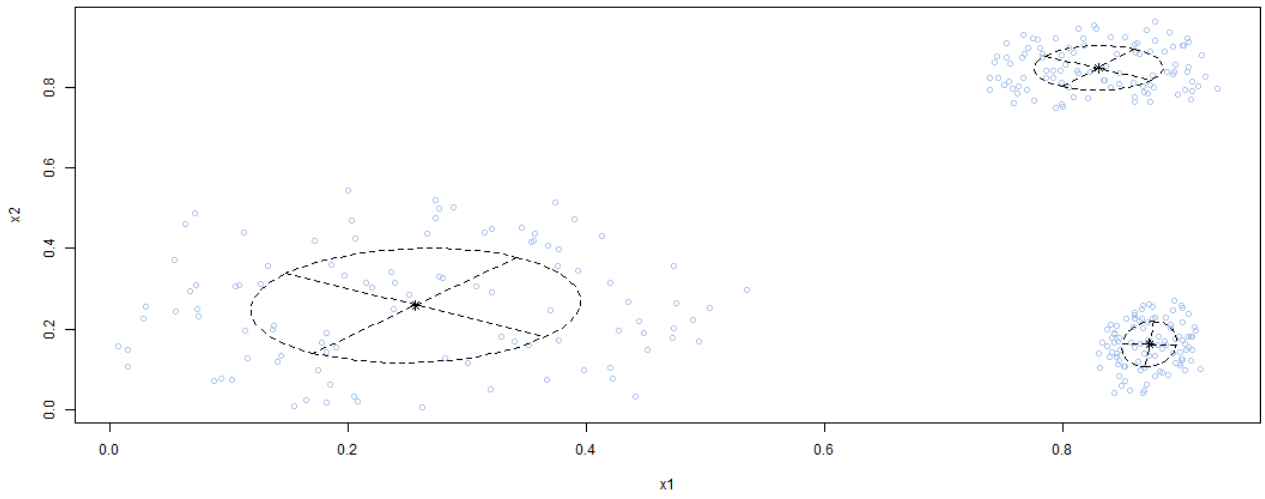


Figure 2.1: EM clustering result.

2.1.2 Distance-based clustering

This clustering approach is based on minimize a distance measure of the elements $\vec{x} \in C_i$ (compactness), and maximize the distance between clusters $C_i \in \Pi$ (separation) [24].

Many definitions of distance have been case of study and a wide research area have been developed around this concept [1, 12, 19, 23, 69]. The simplest and most common definition of distance is that defined by Euclid, which states that the shortest distance between two points a and b is a straight line, this is known as Euclidean distance. This is expressed in Equation 2.1.

$$dist(a,b) = \sqrt{\sum_{i=1}^{d} |a_i - b_i|^2} \tag{2.1}$$

where d are the dimensions of the elements a and b.

Euclidean distance is part of one family of metrics in a normed vector space known as Minkowski distance or norm L_p represented in Equation 2.2 [23].

$$L_p(a,b) = \left(\sum_{i=1}^{d} |a_i - b_i|^p \right)^{\frac{1}{p}} \tag{2.2}$$

where the value of p determines the type of distance that is being used. For instance $p = 1$ is City block or Manhattan distance, $p = \infty$ is Chevishev distance, and d are the dimensions of the elements a and b.

Distance-based CMs can be classified in two principal approaches: flat and hierarchical clustering.

2.1.2.1 Flat clustering

In this approach X is divided into clusters at a same level using a distance representation and representative elements $\vec{x}$ for each cluster C_i. During a defined number of iterations elements $\vec{x}$ are assigned to its closest C_i; after that the partition Π is induced by those assignments, such Π is readjusted following a distance criteria. An example of this is illustrated in Figure 2.2.

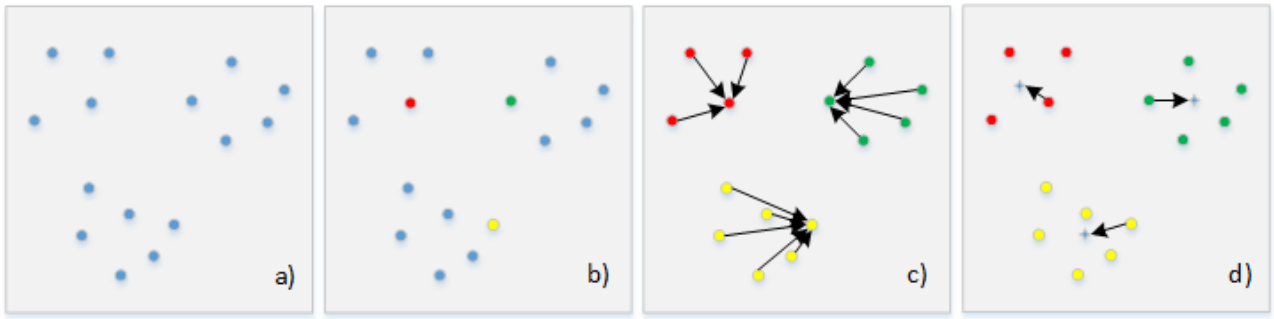


Figure 2.2: Example of flat clustering (k-means algorithm). a) Input dataset X, b) Selection of representative elements for 3 clusters, c) Assignment of $\vec{x}$ to its nearest cluster C_i. d) Adjustment of the representative elements for clusters C_i.

The most common CMs in this approach are:

- *k-means*. In this CM the representative element of a cluster stand for a $\vec{x}$ formed by the mean of the feature values corresponding to the elements $\vec{x} \in C_i$. K-means is considered as the simplest and more used CM for data clustering [39, 40].

- *k-medians*. Unlike k-means, this approach contemplate the representative element $\vec{x}$ of a cluster not by the mean but the median of the feature values corresponding to the elements $\vec{x} \in C_i$. The advantage of this approach over k-means is that is not too sensitive to noise or outliers because the median is not as sensitive as the mean to extreme values. From this behavior this CM can be used as a method of outliers detection [32, 54].

- *k-medoids*. The representative element of a cluster in this CM is taken from the original $\vec{x} \in X$. Each iteration the representative elements are replaced with another $\vec{x}$ in order to check if the quality of Π is improven. Usually this CMs require a bigger quantity of iterations than k-means or k-medians to converge to a solution [3, 24, 68].

2.1.2.2 Hierarchical clustering

In this approach the partitions in a given X are represented by a graphic tree representations known as *Dendogram* in which the different partitions $k = \{1, \ldots, |X|\}$ vary in a hierarchical granularity [38, 44, 46, 72, 77]. Figure 2.3 illustrates an example of this hierarchy.

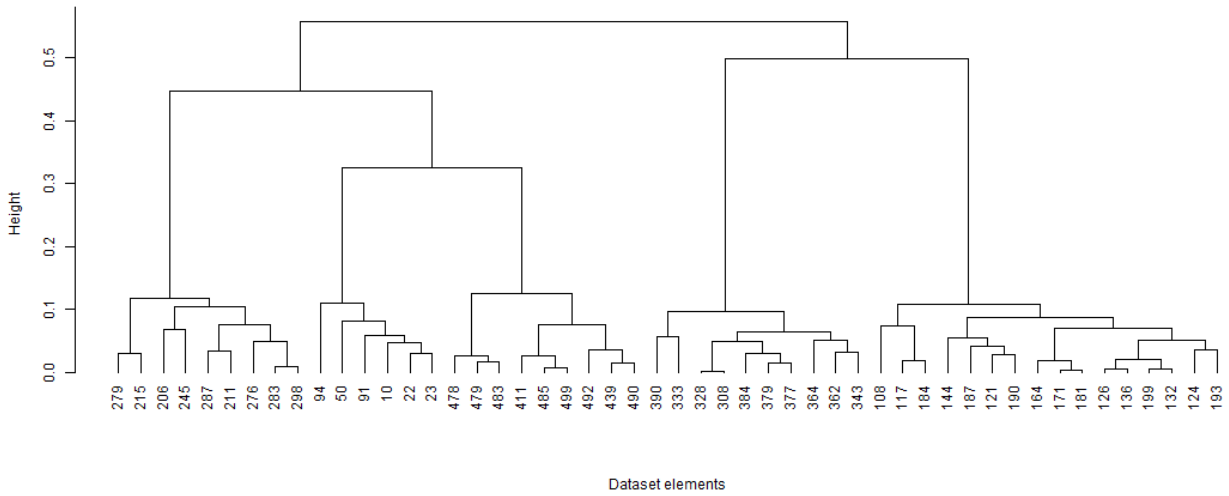


Figure 2.3: Cluster dendogram.

This representation can be built in a top-down or bottom-up form, such manner to build the

hierarchy is described as follows:

- *Agglomerative*. In these CMs, the bottom-up approach is used in which at the beginning each $\vec{x} \in X$ is considered as a cluster. Later each pair of clusters (C_i, C_j) are merged and this process continues until the whole X is considered a single cluster.

- *Divisive*. In the opposite case, the top-down approach here is used in which at the beginning the whole X is considered as a single cluster, then the resultant cluster is divided iteratively until each element $\vec{x}$ is considered as a cluster. To perform such division a flat clustering approach can be used at each level.

2.1.3 Density-based clustering

Most of the CMs described in the above sections assume that the elements $\vec{x} \in X$ were generated from a phenomena that follows a distribution. Nevertheless, this is not the case in some cases for real world observations; due to this assumption the majority of those CMs produce spherical clusters and cannot deal with data that follows non spherical distributions, an example of this is illustrated in Figure 2.4. CMs that discover clusters with arbitrary shapes have been developed known as *Density-based clustering*. This type of CMs are considered as nonparametric methods, since they do not assume a priori number of clusters and their distribution. To deal with this problem, density-based CMs define clusters as dense areas of elements $\vec{x}$ separated by sparse areas of them.

Thanks to this nature this kind of CMs are able to obtain clusters with arbitrary shapes. Some of the most common density-based CMs are:

- *DBSCAN*. This CM estimates the density of a region by counting those elements $\vec{x}$ in a given neighborhood and determine if two elements are connected if they are in each other's neighborhood. Also, an $\vec{x}$ is named *core point* if the neighborhood with radius Eps has at least $MinPts$ of elements. For instance, an element q is density-reachable from a core point

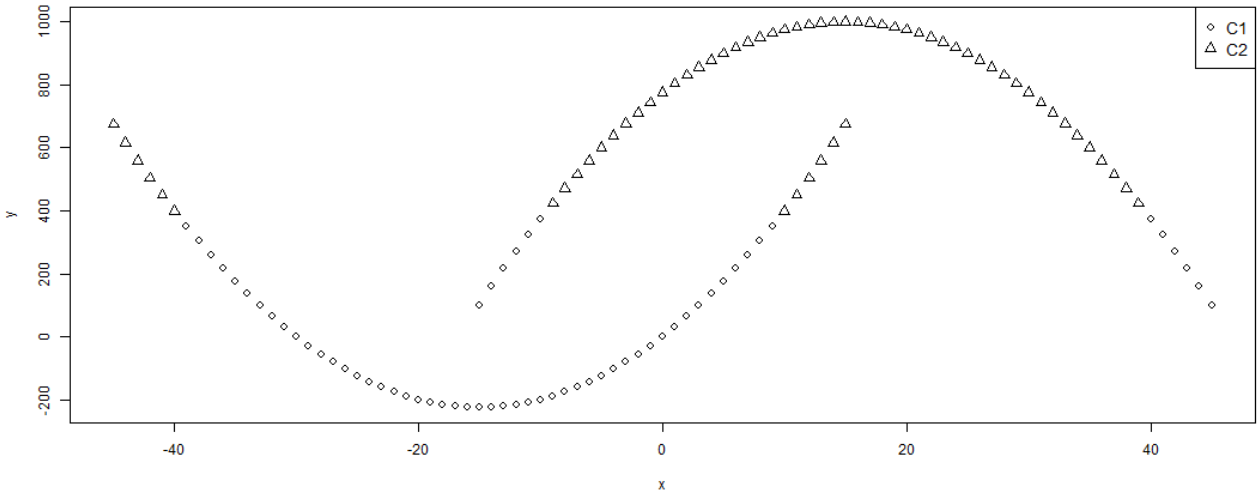


Figure 2.4: Distance based CM applied to a non spherical dataset.

p if q is in the Eps of p. Two elements p and q are density-connected if a third point r exists from which p and q are reachable. By this property DBSCAN considers a cluster as a set of density-connected elements which is maximal with respect to density-reachability [75].

- *DENCLUE*. This CM considers the concept of density-based clusters by the use of *influence functions*, which are mathematical models of an element $\vec{x}$ influence in its neighborhood. The density at a given point p is determined by the sum of the influences of all $\vec{x}$. Then, an element is said to be *density-attracted* to a *density-attractor*, if they are connected through a path of high-density points p. As mentioned, the density function at a point p is determined as the sum of the influence functions of all data points $\vec{x}$ at p. Density-attractors are points that correspond to a local maximum of the density function. A point p is density-attracted to a density-attractor q if this one can be reached from p through a path of points in a neighborhood of radius Eps from each other in the direction of a gradient. Then, a cluster is defined as a set of elements $\vec{x}$ that are density-attracted to one of the density-attractors where the density function exceeds a threshold [34].

- *CUBN*. In this CM the approach of finding a dense area of elements $\vec{x}$ is to find border points of those dense areas via an *erosion operation*, in a second step border and inner elements of the clusters are joined by the near distance between them. This *erosion operation* has a visual meaning for geometry; for instance, applying an erosion operation to an area will eliminate the roughness of the border and keep the basic shape of the cluster area. This CM can be considered as a mixture of density-based and distance-based clustering [88].

2.2 Validity indices

One of the most important tasks in clustering is the evaluation of the results. Such evaluation is relative to the desired properties of the found clusters. For instance in Figure 2.5 are illustrated three possible solutions for the same dataset. The optimal number of clusters and the arrangement of them depends on those conditions or criteria of what should be a good solution.

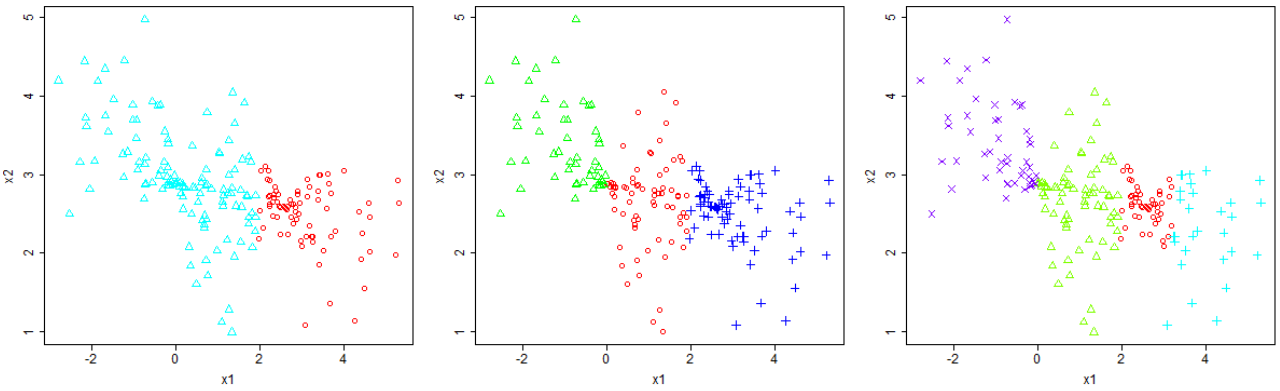


Figure 2.5: Different Π proposed by a CM variating k.

There are many efforts to quantify such criteria. These are commonly known as *validity indices*, which usually consider the following properties:

1. **Compactness**. This measure contemplates how close to each other are elements $\vec{x} \in C_i$. A common measure of compactness is the variance of clusters, which is commonly minimized

[2, 61].

2. **Separation** This measure determine how spaced are clusters between them [2, 61]. There are three principal approaches to measure this property:

   - *Single linkage*. It measures the distance between the closest elements $\vec{x}$ of the C_i and C_j clusters.
   - *Complete linkage*. It measures the distance between the most distant elements $\vec{x}$ of the C_i and C_j clusters.
   - *Comparison of centroids*. It measures the distance between the centers of the clusters C_i and C_j, commonly determined by the mean of the elements $\vec{x}$ in the same cluster.

Generally there are two types of validity indices that determine the quality of a clustering solution. Such types are described in the next subsections with some of their most representative validity indices.

2.2.1 External indices

In this approach the idea is to determine if the elements of X are randomly structured or not. To do so, this analysis is based on statistical tests, which leads to a high computational complexity. Thus, to accelerate this process Monte Carlo techniques can be used as a solution to this problem, however this does not return an exact measure [61].

Another way to reduce this complexity is with the comparison of Π with an a priori proposed partition $\Pi' = \{C_1, C_2, \ldots, C_k\}$, in which such comparison would be done by the following terms:

- *SS*: if both points belong to the same cluster of partition Π and to the same cluster of partition Π'.
- *SD*: if points belong to the same cluster of Π and to different clusters of Π'.

- *DS*: if points belong to different clusters of Π and to the same cluster of Π'.
- *DD*: if both points belong to different clusters of Π and to different clusters of Π'.

Let a, b, c and d be the number of pairs SS, SD, DS and DD, then $a+b+c+d = M$ where M is the maximum number of pairs in X (meaning $M = N(N-1)/2$ where $N = |X|$). By defining this the validity indices in this approach are those listed below:

- *Rand Statistic [57, 61]:* $R = (a+d)/M$
- *Jaccard Coefficient [57, 67]:* $JC = a/(a+b+c)$
- *Folkes and Mallows* [57, 61]: $FM = \sqrt{\frac{a}{a+b} \cdot \frac{a}{a+c}}$

These validity indices values are in $[0, 1]$ where a higher value determines a better Π.

2.2.2 Internal indices

In this approach the main idea is to use features inherent to X. Commonly this type of validity indices are used depending on the clustering structure where two principal cases can be found as follows:

Validating hierarchy of cluster schemes. To validate a Π with this type of structure a matrix called cophenetic matrix P_c is used, which represents the hierarchy produced by a hierarchical CM. The $P_c(i, j)$ element in the matrix represents the proximity at which two elements $\vec{x}_i$ and $\vec{x}_j$ in X are found in the same cluster. A common validity index that measure the similarity between this P_c and P (proximity matrix, which is a matrix of size $|X| \times |X|$ with the proximities between all pairs of elements in X) is called *Cophenetic Correlation Coefficient* [55, 61], defined as shown below:

$$CPCC = \frac{(1/M)\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} d_{ij}c_{ij} - \mu_p\mu_c}{\sqrt{[(1/M)\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} d_{ij}^2 - \mu_p^2][(1/M)\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} c_{ij}^2 - \mu_c^2]}}, -1 \leq CPCC \leq 1 \tag{2.3}$$

where $M = N \cdot (N-1)/2$ and $N = |X|$. Also, μ_p and μ_c are the matrices means of P and P_c respectively, and are defined by:

$$\mu_p = (1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} P(i,j), \mu_c = (1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} P_c(i,j) \tag{2.4}$$

moreover, d_{ij}, c_{ij} are the (i,j) elements of P and P_c matrices. This validity index represents a better similarity between the two matrices when its value is closer to 0.

Validating a single clustering scheme. The approach in this type of indices is to find a degree of agreement between clusters in a clustering scheme C, consisting on k clusters and its proximity matrix P. A common example of this approach is the Hubert's Γ statistic [37].

Since this validity index approach is the one used in this work some of the most common validity indices in this category are described in Appendix B.

2.3 Meta-heuristic optimization methods

Optimization problems have been a widely case of study during the last decades in which many approaches to solve them have been developed. Find optimal solutions is an essential task for many problems in the real world. For this, exact methods approaches guarantee the optimality of the solutions found by them, but sometimes this kind of techniques are not suitable because of such problem's complexity. Thus, exact methods can be used to solve small instances for some NP-hard problems [79].

Unlike exact methods, meta-heuristics allow to solve hard problems and offer good solutions in a reasonable time, however this approach does not guarantee the optimality of results. Meta-heuristics have been used in many applications such as engineering design, machine learning, system modeling, aerodynamics, cost reduction, physics, etc.

2.3.1 Basic concepts in meta-heuristic techniques

A meta-heuristic has three principal common elements described as follows:

- An encoding that represents a solution of the problem, commonly represented as a vector $\vec{s}$.
- An objective function that guides the search process.
- An adaptive process in which the solutions are changed in order to improve the fitness of solutions at previous iterations. The goodness of a meta-heuristic lies on its adaptive process, such element approach has been subject of study in the wide field of optimization. Some of the most common approaches are described in subsections 2.3.2 and 2.3.4.

2.3.1.1 Encoding

To represent a solution, the meta-heuristic involves a vector $\vec{s}$ in a set of candidate solutions S whose components quantify properties or variables of the problem [85]. Such representation must satisfy:

- **Completeness**: This refers to that all possible solutions must be able to be represented by $\vec{s}$.
- **Connexity**: This refers to the path that must exist between any pair of solutions in the search space, especially the optimal solution.
- **Efficiency:** $\vec{s}$ must be easy to manipulate by the algorithm since this directly affects the execution time and complexity of any algorithm.

Some examples of encodings for traditional families of optimization problems are listed below:

- **Binary encoding.** This type of encoding may be like $\vec{s} \in \mathbb{B}^l$, which is a vector of length l with elements in a binary alphabet $\mathbb{B} = \{0, 1\}$. Such encoding may be used in problems such as the knapsack problem, SAT problem, $0/1$ IP problems.

- **Discrete encoding.** This type of encoding may be like $\vec{s} \in \mathbb{N}^+$ with length l, in a numerical alphabet $\mathbb{N}^+$. Such encoding may be used in problems such as location problem, assignment problem.

- **Real encoding.** This type of encoding is commonly one of the type $\vec{s} \in \mathbb{R}^l$, where l denotes the length of the vector. This encoding may be used in problems like continuous problems, parameter identification, global optimization.

2.3.1.2 Objective function

Formally such function is defined as $f : \vec{s} \to \mathbb{R}$. It represents an absolute measure that allows a complete ordering of all solutions $\vec{s}$ of the search space, where the optimal solution is the minimal element (assuming the problem as one of minimization) [79, 85].

2.3.1.3 Constraint handling

Many real world problems can represent an extremely large search space, which can include unfeasible solutions. To guide a meta-heuristic to ignore this unfeasible search space, a set of constraints is commonly included. This constraints help to prune all this out of interest space for the user. Many continuous and or discrete optimization problems are constrained and is not a trivial work to deal with such constrains. For this, some of the most common strategies for the constraint handling are described below:

1. **Death penalty strategies**. In this approach, only the feasible $\vec{s}$ are considered during the search process and all other solutions are discarded. This approach of constraint handling is functional if the portion of unfeasible search space is very small [6].

2. **Penalizing strategies**. In this approach the unfeasible search space is also considered during the search process. To do this, the objective functions is complemented with a penalization function that penalizes the unfeasible solutions $\vec{s}$ [26].

3. **Repairing strategies**. This approach consists in a transformation of a $\vec{s}$ from unfeasible space to a $\vec{s}$ in the feasible space. To do this, a repairing procedure is applied to this unfeasible solutions to generate feasible ones [79]. Examples of repairing strategies may be:

   - Extreme strategies: One kind of extreme strategy is to set $\vec{s}$ to the limit it exceeds. A second kind of this strategy is to reinitialize $\vec{s}$ to a random value.
   - Intermediate: An example of intermediate strategy consists in reinitializing $\vec{s}$ to a point, which is the midway between its old value (before variation) and the bound being violated.

2.3.2 Single-solution based meta-heuristics

This kind of meta-heuristic techniques are based in improving a single solution $\vec{s}$. Its approach can be seen as "walks" trough the search space of the problem.

2.3.2.1 Basic concepts

In this meta-heuristic approach the search is performed in an iterative process in which one solution is evaluated per iteration, moving it in the search space. Single solution meta-heuristics apply generation and replacement procedures iteratively from the current $\vec{s}$. In the generation procedure, a set of candidate solutions S is generated from the current $\vec{s}$, which is commonly obtained by transformations of such $\vec{s}$. In the replace procedure, one $\vec{s}\,' \in S$ is selected to replace the actual solution $\vec{s}$. An example of this is illustrated in Figure 2.6 [79].

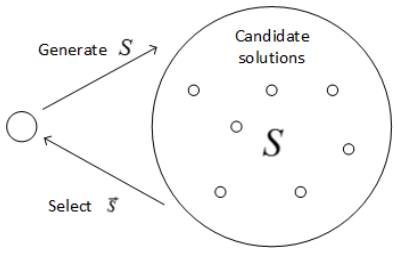


Figure 2.6: Single-solution based meta-heuristics approach.

Neighborhood. One essential concept in any single based meta-heuristic is the neighborhood definition, which structure plays an important role in the performance of the meta-heuristic. One of the most common definitions of the neighborhood is one that states:

The neighborhood $N(\vec{s})$ of a solution $\vec{s}$ in a continuous space is the hypersphere with center $\vec{s}$ and radius equal to ϵ with $\epsilon > 0$. Thus $N(\vec{s}) = \{\vec{s} \in S \mid ||\vec{s} - \vec{s}\,'|| < \epsilon\}$ where $||\vec{s} - \vec{s}\,'||$ is the Euclidean distance between the actual solution $\vec{s}$ and the proposed solution $\vec{s}\,'$ [79, 85].

Local optimum. Relative to a given neighborhood N in the search space, a solution $\vec{s}$ is a local optimum if it has better fitness than the rest of $\vec{s} \in N$. In a given problem a local optimum of one neighborhood N_1 may not be a local optimum for a neighborhood N_2, even so, a local optimum could be the global optimum. An example of this is illustrated in Figure 2.7.

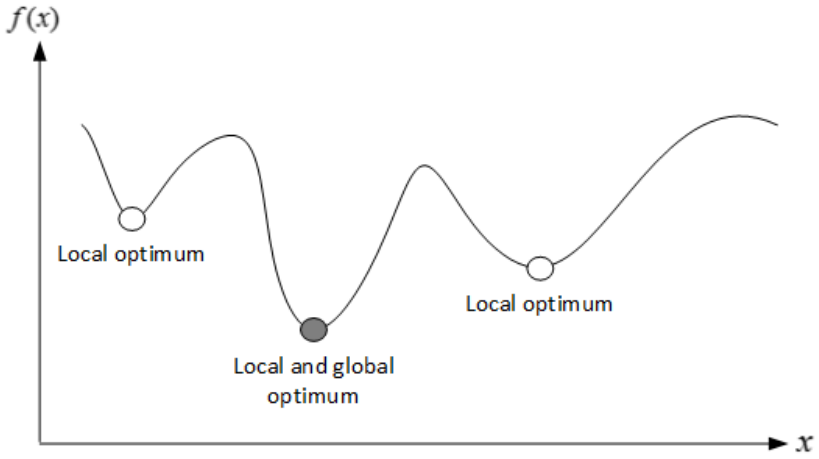


Figure 2.7: Local and global optimum values in a search space for a minimization problem.

2.3.3 Common single-solution based meta-heuristics

Some of the most common single-solution based meta-heuristic techniques are described below:

- **Simulated annealing (SA).** This meta-heuristic is based on the principles of statistical mechanics where an annealing process starts by heating and then slowly cooling a material in order to obtain a crystalline structure. SA algorithm simulates the energy changes in a system

subjected to a cooling process until it converges to an equilibrium state. The principal objective of this approach is to escape from local minimum solutions and so to delay the convergence. This is a memoryless algorithm since it does not use any information obtained during previous iterations during the search space [84].

- **Tabu search (TS).** One of the advantages of this method over simulated annealing is the use of memory, which stores information about the search process. TS acts like a local search algorithm in which all the neighborhood is explored in a deterministic manner. When a local optimum is found the next step is to choose a worse $\vec{s}$ than the current one. The best $\vec{s}$ in the neighborhood is selected as the new current $\vec{s}$ even when this one does not improve the fitness, and a new local search process is done again. To avoid cycles, TS discards the previously explored neighbors with a search history of the recently applied moves, which is called *tabu list*, in order to find the best $\vec{s}$ [27].

- **Random mutation hill climbing (RMHC).** This algorithm is based on a principle that has been used in many heuristics. First, a local search is performed to an initial solution. Then, at each iteration, a perturbation is applied to the $\vec{s}$ representing the local minimum. Finally, a local search is performed over the new perturbed $\vec{s}$. Some criteria must be satisfied in order to replace the current $\vec{s}$ with the new one [59].

2.3.4 Population-based meta-heuristics

The second relevant approach in meta-heuristics are population based algorithms. In this kind of techniques the main idea is to improve a set of candidate solutions S commonly named as *individuals* in an iterative process.

2.3.4.1 Basic concepts

In this approach a initial population of solutions S is generated. After this generation, during a given number of iterations or a stopping criteria is reached, a new population S' is generated which replaces the actual one [79, 85]. This process is illustrated in Figure 2.8.

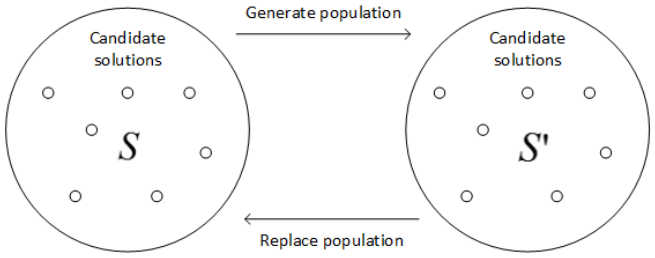


Figure 2.8: Population based meta-heuristics approach.

Commonly this type of algorithms are nature-inspired, and the process is memoryless since the two procedures (generation and replacement) are based only in the actual S. Population based meta-heuristics may be different in the way they perform the generation and replacement procedures. This leads to define the next principal concepts:

- **Generation of S.** This is the step in which the new population S' is generated. Different strategies have been proposed for this procedure to be perform, which are generally classified in two main categories:

  1. *Evolution based.* In this approach the new S is generated by the selection and reproduction of individuals $\vec{s} \in S$ using operations commonly known as *mutation* and *recombination* operators. This operators attempt to generate the new population S' considering different attributes of solutions $\vec{s} \in S$ [28, 64].

  2. *Blackboard based.* In this approach S participates in the construction of a shared memory which is the main input to generate S' [79].

- **Replacement of S.** In this step the replacement of the actual S is based in selecting the new solutions $\vec{s}$ from the union of S and S'. The most common strategy in this phase is replace the entire S by the generated population S'. Other strategies imply to use an elitist approach which provides the best individuals from the two sets [28, 64].

2.3.4.2 Evolutionary approaches

This population-based meta-heuristic approach has been applied to many real and complex problems. They are the most studied population based algorithms. It has been proven their success in solving optimization problems in domains such as continuous or combinational optimization, system modeling, control, engineering design and machine learning. Such approach have promoted the field known as *evolutionary computation* [5].

This kind of meta-heuristics are based on the natural evolution of species, which in the optimization field this is represented by a population S of individuals $\vec{s}$. At the beginning S is generated in a random manner. At each iteration (called *generation* in this approach) individuals $\vec{s}$ are selected from the population S following a selection process in which individuals with better fitness have higher probability to be chosen. In a second step, the selected individuals are reproduced following variation operations (e.g. *crossover* and *mutation*) in order to generate a new offspring S'. This process is repeated during a defined number of generations, which is the most common stopping criteria. Finally, as mentioned, a replacement scheme is followed in order to determine which individuals will survive for the next generation [28, 64].

2.3.5 Common population-based meta-heuristics

Some of the most common population-based meta-heuristic techniques are described below:

- **Genetic algorithms (GA).** This type of evolutionary algorithms are based on the adaptive process of natural genetic systems. Usually GAs are associated with the use of binary

representations $\vec{s} = \{0,1\}^l$ with length l. This approach applies a crossover operator over two individuals $\vec{s}$ in the population S during the generation phase. Also a mutation operator that modifies, in a random manner, the resultant individuals is applied over the population which promotes a diversity of solutions. GAs have a wide brand of variations, which are born from the original proposal of J. Holland [28, 64].

- **Differential evolution (DE).** This meta-heuristic works directly on the continuous space $\vec{s} \in \mathbb{R}^l$ where l denotes the length of the encoded solution $\vec{s}$. The principal idea of DE is the use of a vector of differences in the perturbation of S to generate S'. Each $\vec{s}$ is generated in a random manner at the beginning in a range $[x_{lo}, x_{hi}]$ representing lower and upper bounds for each element in $\vec{s}$. For the recombination phase the generation of the new individuals in the population S is based on a linear combination of three randomly selected individuals in which the distance concept plays an important role [76].

- **Ant colony optimization (ACO).** The main idea of this approach is to imitate the behavior of ants to solve problems. This type of algorithms mimic the principle of using simple communication mechanisms to find the shortest path between two points (e.g. actual $\vec{s}$ and best $\vec{s}$). During the trip a trail (pheromone) is left for other solutions to follow it. The larger amount of pheromone the more probability that the ant colony will select such path that leads to the best $\vec{s}$ [21].

- **Particle swarm optimization (PSO).** This is a stochastic population-based meta-heuristic which mimic natural swarm organisms behavior. This approach was designed for continuous optimization problems. The basic particle swarm algorithm consists of a set of particles (solution population S) moving around the search space where each particle has its own position and velocity. During the search process each particle adjusts its position toward the global optimum according to its best position visited and also the best position visited by the entire swarm. This implies a cooperation between all particles in S [47].

2.4 Related work

As mentioned, clustering methods attempt to make groups of elements similar between them. In general, the clustering problem involves a search process of a partition Π in a dataset X. Usually such process is driven by proximity criteria relative to a distance metric. In the last two decades the use of optimization methods such as meta-heuristics have been applied along with CMs to include other criteria (e.g. entropy or density fitting). This kind of CMs are commonly called *Meta-heuristic clustering*, which tackle the clustering problem as an optimization problem. These imply a greater flexibility of optimization algorithms. However, such methods also require more computational resources which increase their execution time. Some of these works are described below.

In 2001 Lin Yu Tseng et al. [83] proposed a genetic approach to the automatic clustering problem, which is composed of two stages: 1) The distance between each $\vec{x} \in X$ is computed in order to determine the average distance between each $\vec{x}$. After this, the adjacency matrix is computed to build a graph which denotes a set of initial clusters $C = \{c_1, c_2, \ldots, c_k\}$ that will not be separated. 2) Using a GA the initial clusters $c_i \in C$ are merged as a partition Π during the evolution process. The GA was guided by an objective function composed by the intra and inter cluster distances.

In 2008 Swagatam Das et al. [16] proposed Automatic Clustering Using an Improved Differential Evolution Algorithm. As mentioned in its name, this meta-heuristic CM uses DE to improve the population S of solutions $\vec{s}$. Here, each $\vec{s}$ is a vector of real numbers of dimension with $length = Kmax + Kmax \cdot d$ where $2 \leq K_{max} \leq |X|$. The first K_{max} entries are positive floating point numbers in $[0, 1]$, which control whether the corresponding cluster is to be considered in the solution or not for the induction of Π. The remaining entries are reserved for K_{max} cluster centers, each d dimensional. This enables the search process without giving a specific k number of clusters a priori. The evolution process was guided by the optimization of the Davies-bouldin index and the CS Measure proposed by Chou [13].

In 2010 Xiaoyong Liu et al. [57] proposed a clustering algorithm using the ant colony algorithm in which each solution $\vec{s}$ defines a set of cluster centroids $C = \{v_1, v_2, \ldots, v_k\}$. To infer Π in a solution $\vec{s}$ each $\vec{x} \in X$ is assigned to its nearest centroid. The improvement of the set of solutions S is guided by the the sum of square Euclidean distances between each $\vec{x}$ and its centroid. Also the use of validity indices such as Dunn index, Jaccard index and FM index was proven.

In 2012 Chih-Wei Wang et al. [87] proposed an Automatic Clustering method using Particle Swarm Optimization with Various Validity Indices. This CM is based on using the PSO algorithm. In this approach, each particle represents a solution $\vec{s}$ composed by k cluster centroids $C = \{v_1, v_2, \ldots, v_k\}$ in $[k_{min}, k_{max}]$; each particle may have different number of v, which enables the search process without giving a specific k number of clusters a priori; in a similar manner to the Swagatam Das's work [16]. The improvement of the particle population is guided by validity indices as objective function, which may vary depending on the used one.

In 2014 Leonardo Enzo Brito da Silva et al. [15] proposed the Clustering of the Self-Organizing Map using PSO and Validity Indices. This method uses PSO to train a SOM network [48] which is used to make clusters. Each solution $\vec{s}$ denotes those weights for the nodes in the SOM network. To do the clustering process $\vec{x}$ are grouped to its nearest neuron using an hypersphere approach instead of the voronoi regions approach. This enables at the end of the execution a neural network that may be used in the classification of new $\vec{x}$. To guide the search process the CDbw Index, Rand and Adjusted Rand Indices were used.

In 2015 Aldana Bobadilla et al. [10] proposed a clustering method which uses a genetic algorithm to find one partition that maximize the entropy [74] subject to a given set of possible constraints that represent the desired properties of the clusters.

Based on the above discussion, our proposal is also a meta-heuristic clustering approach, in which the problem is tackled as a general optimization problem. Many proposals proceeding in this fashion have arisen as shown above. In most methods of this type, a distinguishable element is the objective function. It involves different criteria associated to the properties of Π that must be

optimized. Since these criteria affect significantly the way in which the problem is encoded, usually the clustering methods using meta-heuristics lack versatility to change or include new criteria. To minimize this lack, we propose an encoding invariant of the objective function, that allows us to obtain a clustering method able to include different optimality criteria and a variety of clustering solutions. Unlike the most related works, the proposed encoding does not attempt to include those common variables associated to the problem (e.g. labels of the elements to be clustered, centroids of the clusters, etc.). Instead, such encoding includes a feasible mathematical model of Π defined as polynomial expression. From this model, the membership of all objects in X is determined. Throughout our method, a set of models is iteratively adapted until several optimality criteria (given by a validity index) are met. The best model will describe the best partition Π. From this model, unlike other clustering methods, our method is able to predict the membership of a new object without executing the search process of Π again.

3

Methodology

Based on the above discussion, we tackle the clustering problem as a general optimization problem wherein the partition Π that optimizes a validity index must be found. In general, Π is formed by k non-empty subsets of X. The number of different partitions in a dataset X may be expressed by the function $S(N;k)$ associated with the Stirling number of the second kind [71] defined as:

$$S(N;k) = \frac{1}{k!}\sum_{i=0}^{k}(-1)^{k-i}\binom{k}{i}i^N \tag{3.1}$$

where $N = |X|$ and k is the desired number of clusters in a partition Π. It implies that to find the optimal partition, a large space must be explored (for example, given $N = 50$ and $k = 2$, $S(N;k) \approx 5.63 \times 10^{14}$).

Based on the above, a mandatory step is to choose an appropriate meta-heuristic that allows us to explore efficiently the large solution space that such problem involves. The use of a meta-heuristic implies an appropriate representation of the problem. In this regard, an important concern is the length of such representation. Since many representation proposals depend on properties of the

dataset as the cardinality and the number of dimensions, we propose a novel representation that attempts to minimize such dependency. Attending the above elements the proposed method works as shown in Figure 3.1.

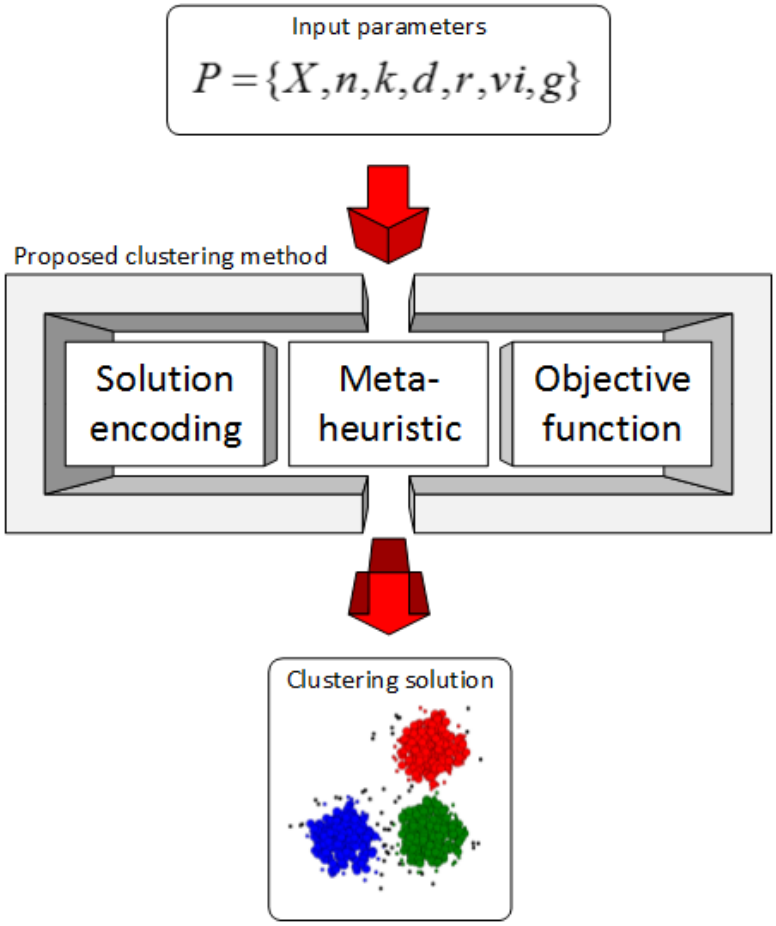


Figure 3.1: Proposed clustering method.

The proposed CM receives a set of parameters that includes the dataset X to be clustered and setting values which are described in the subsequent sections of this chapter. The proposed CM searches iteratively one clustering solution that optimizes a validity index. This method includes three principal components which are illustrated in Figure 3.1 and described as follows:

1. An objective function based on a validity index that allows us to evaluate the solutions proposed during the search process (described in Section 3.1).

2. A suitable meta-heuristic (described in Section 3.2) that allows to explore efficiently the large solution space.

3. An encoding in which the problem is represented (described in section 3.3).

3.1 Defining the objective function

As mentioned, the clustering problem involves a complex search process of an optimal partition Π on X. Typically this process is guided by a proximity measure. Instead, we hypothesize that the search process can be guided by a quality measure of Π. Such measure is given by a validity index [8, 70, 73]. In this case the problem is reduced to find a partition that optimizes the used validity index. This is an optimization problem whose objective function can be expressed as:

$$\begin{gathered} Optimize: \\ Q(\Pi) \\ \\ Subject\ to: \\ g_1(\Pi) \leq \epsilon_1 \\ g_2(\Pi) \leq \epsilon_2 \\ \vdots \\ g_n(\Pi) \leq \epsilon_n \end{gathered} \tag{3.2}$$

where $Q(\Pi)$ is a validity index as a function of an instance of Π. The functions $g_n(\Pi) \leq \epsilon_n$ represent a set of possible constraints that must be satisfied. From Equation 3.2 the problem to be solved is a constrained optimization problem. The constraints ensure that each instance of Π satisfies several conditions, in which case, such instance will be a feasible partition. To ensure such feasibility, a penalty function [90] can be applied to Q as follows:

$$Q(\Pi) = \begin{cases} Q(\Pi) & \Pi \in feasible\ region \\ Q(\Pi) + penalty() & otherwise \end{cases} \tag{3.3}$$

The function $penalty()$ allows us to prune the search space, favoring those instances of Π that satisfy the constrains g. There are many ways to define $penalty()$, for simplicity, we assume that the constraints have the same importance, in which case, the $penalty()$ function will be proportional to the number of those constraints that are met. This can be expressed as follows:

$$penalty(s, n) = K\left(1 - \frac{s}{n}\right) \tag{3.4}$$

where K is a large constant (e.g. $O(10^9)$), n is the number of constraints and s is the number of these that are satisfied [53].

3.2 Choosing a suitable meta-heuristic

Our discussion has outlined an optimization problem that involves a large solution space composed of all possible instances of Π. We point out that the search process of the optimal instance of Π can be driven by a validity index. Since not all indices guarantee conditions of continuity and convexity, a search approach as those based on descent gradient is unsuitable. For this reason, it is compulsory to use a computationally-intensive method that allows us to explore efficiently the solution space without considering the mentioned conditions such as meta-heuristic techniques. To choose a suitable meta-heuristic, we conducted several experiments that include a wide reservoir of optimization problems and a set of different meta-heuristics.

The reservoir of problems is composed of:

1. A set of unconstrained optimization functions typically used in benchmark analysis (see Appendix A). In this case the objective function is composed only by the function value.

2. A set of hard constrained optimization functions [20] (see Appendix A). In this case the objective function is composed by the function value plus a penalization as shown in Equation 3.3.

The set of meta-heuristics includes:

Non-evolutionary approaches:

- Random mutation hill climbing (RMHC) [79].
- Simulated annealing (SA) [79, 84].

Evolutionary approaches:

- Holland genetic algorithm with elitism (Holland) [29, 62, 79].
- Differential evolution DE/rand/1/bin (DE) [76, 78, 79].
- Eclectic genetic algorithm (EGA) [10, 25, 29, 62, 65].

The meta-heuristic parameters for each technique are shown in Tables 3.1, 3.2, 3.3, 3.4 and 3.5. Such parameter values were taken as recommended in the citations above.

Iterations:	100

Table 3.1: RMHC parameter settings

Initial temperature: (T)	500
Cooling update:	$T = \alpha T$
α:	0.99
Equilibrium state:	50 iterations

Table 3.2: SA parameter settings

Population size:	100 individuals
Crossover probability:	0.8
Mutation probability:	0.1
Generations:	100
Selection:	Elitist

Table 3.3: Holland parameter settings.

Population size:	100 individuals
CR:	0.9
F:	0.8
Generations:	100
Repair strategy:	Intermediate

Table 3.4: DE parameter settings.

Population size:	100 individuals
Crossover probability:	0.8
Mutation probability:	0.1
Generations:	100
Selection:	Deterministic

Table 3.5: EGA parameter settings.

Each combination of meta-heuristic and optimization function was executed 100 times. This results in a total of $15,500$ fitness values. To determine the performance of each meta-heuristic, since all meta-heuristic techniques were tested with the same problems a first approach was to count how many times each meta-heuristic obtained the best fitness values per optimization function. To do this the next process was performed:

- For an optimization problem A the average fitness obtained with each meta-heuristic is calculated.
- From the values obtained above, the mean μ and standard deviation σ was calculated.
- We gave a point to the meta-heuristic if the average fitness for problem A is in $[\mu - \sigma, \mu + \sigma]$.
- The previous steps were applied for the rest of the optimization problems.
- At the end the total of winning points for each meta-heuristic is obtained.

The total of this counting is shown in Figure 3.2.

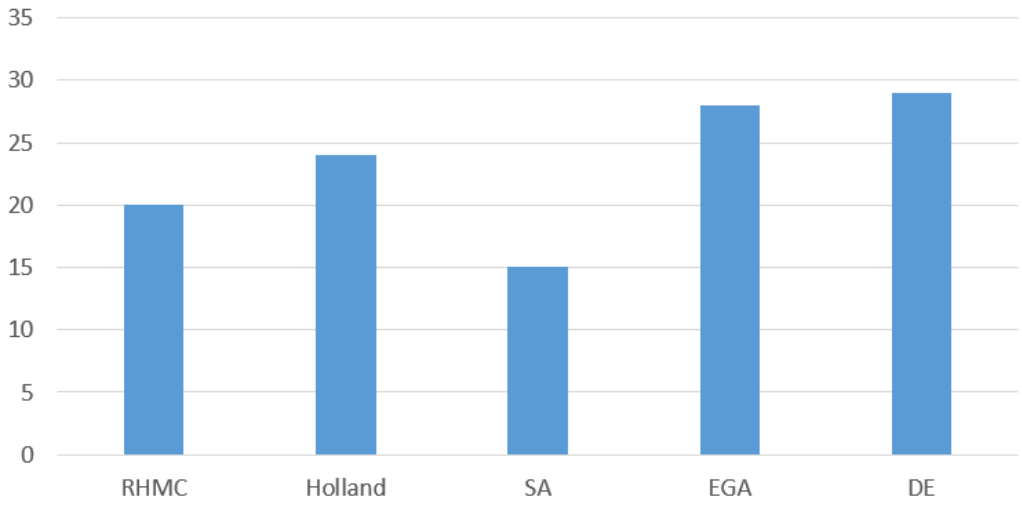


Figure 3.2: Count of the best fitness obtained

The performance of a meta-heuristic is also determined by the average fitness f' obtained from all functions (assuming minimization) at the i^{th} iteration, this shows an overview of the convergence of each meta-heuristic, which is also an important factor. Such performance is illustrated in Figure 3.3 and 3.4.

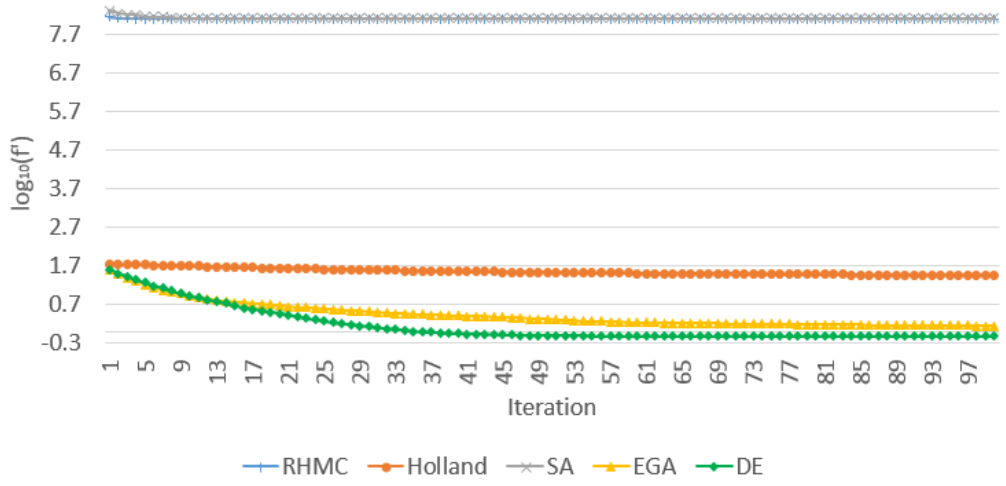


Figure 3.3: Performance of the selected meta-heuristic for unconstrained problems. The results show a different magnitude orders which are expressed as log_{10}.

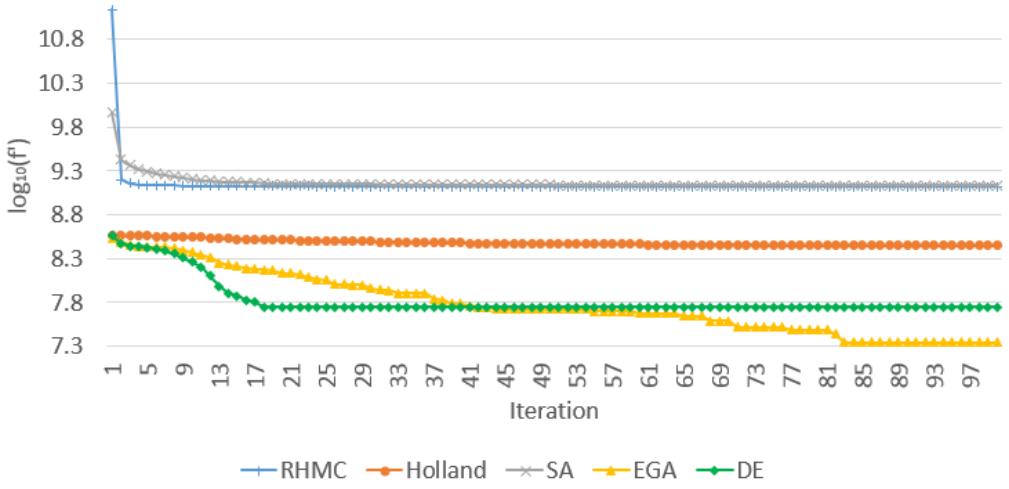


Figure 3.4: Performance of the selected meta-heuristic for constrained problems. The results show a different magnitude orders which are expressed as log_{10}.

In general, this results showed that DE and a variation of genetic algorithm called Eclectic Genetic Algorithm (EGA), are best. Similar studies that point to EGA as the best option relative to other GAs (not including DE) can be found in [52]. Based on these results, we consider EGA and DE as the best options to solve our problem.

3.3 Problem encoding for clustering

As mentioned, meta-heuristics are computationally-intensive techniques in which a set of candidates solutions are iteratively adapted until an optimality criterion is met. A candidate represents a possible solution of the problem. The way in which a candidate is represented is called solution encoding. Formally such representation is a vector of the form $\vec{s} \in S$ whose components represent variables or attributes of the problem. For instance, when $\vec{s} \in \mathbb{B}^l$ for $\mathbb{B} = \{0, 1\}$ the encoding is a binary string of length l. An example is illustrated in Figure 3.5. There exist other possible representations such as $\vec{s} \in \mathbb{R}^l$, whose set of solutions belongs to the real numbers. In the following sections, we present different approaches in the context of the clustering problem.

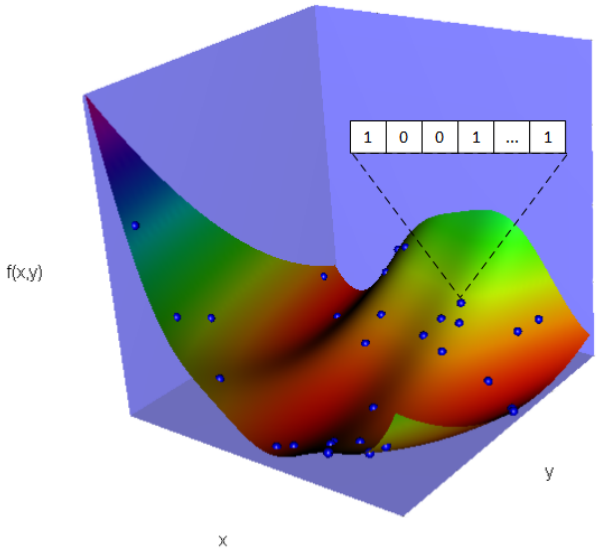


Figure 3.5: Encoded solutions positioned in the landscape of a two dimensional objective function.

3.3.1 Binary encoding

A typical encoding of a partition Π is a binary string of length $l = |X|$. In this case, many interpretations of each digit are possible. For instance, in [35, 45, 50, 51] the digits with value 1 indicate those elements in X that are the cluster's centroids (see Figure 3.6). Based on these centers, the membership of the remains elements in X is determined.

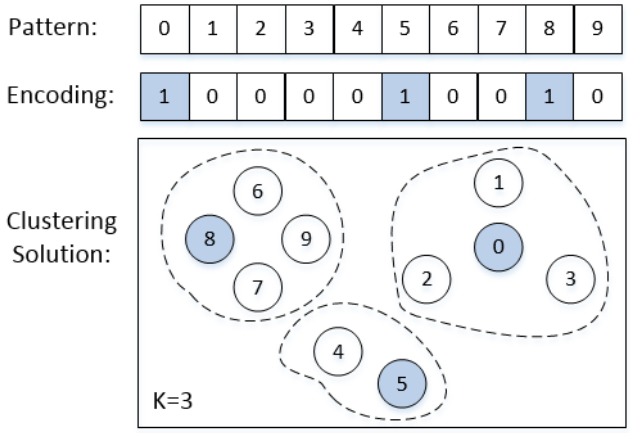


Figure 3.6: Binary encoding.

3.3.2 Integer encoding

In a typical case, Π is a vector $\vec{s} \in \mathbb{Z}^{+l}$ where $l = |X|$. Like binary encoding, many interpretations of the elements of $\vec{s}$ are possible. A common interpretation is one in which the i^{th} element in $\vec{s}$ represents the cluster label of the i^{th} element in X [35, 45, 49, 56]. This is shown in Figure 3.7.

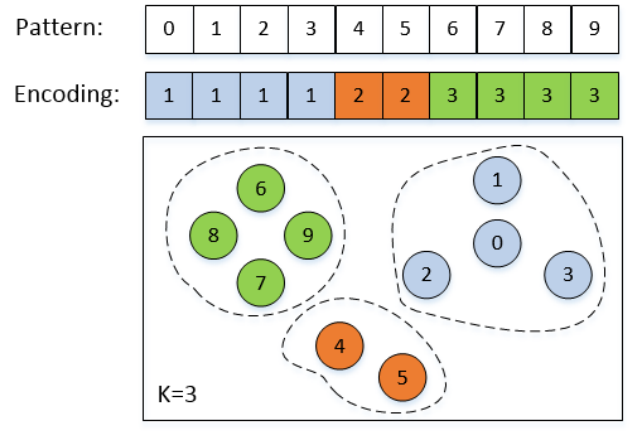


Figure 3.7: Integer encoding.

3.3.3 Real encoding

In this approach the vector $\vec{s} \in \mathbb{R}^l$, the value of l depends on different variables of the problem that are encoded. For example, in Figure 3.8 is shown a proposal in which $l = (d * k_{max}) + k_{max}$, where k_{max} is an upper bound of the number of clusters that must be found. The first $d * k_{max}$ elements represent a set of k_{max} centroids. The remains elements represent an activation threshold that determines when a centroid is feasible for the solution (e.g. if such value is greater than 0.5 the i^{th} centroid is feasible) [16, 35, 45].

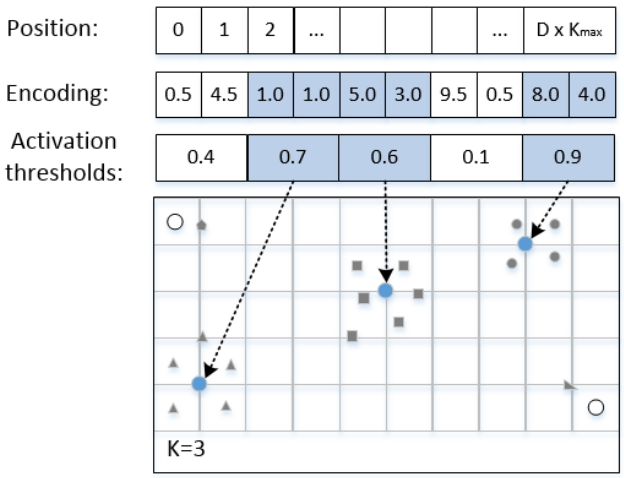


Figure 3.8: Real number encoding.

3.3.4 Proposed encoding

In our proposal, the encoding is based on a real approach in which case the vector $\vec{s} \in \mathbb{R}^l$. Unlike the traditional representations, $\vec{s}$ does not define Π in terms of labels of the elements in X or centroids. In our proposal $\vec{s}$ attempts to represent the clusters as mathematical functions. Based on these functions, it is possible to determine the cluster to which $\vec{x} \in X$ belongs. We use functions defined by a polynomial expression of the form:

$$f(\vec{x}) = \sum_{i=0}^{r} \sum_{j=1}^{d} \alpha_{ij} x_j^i \tag{3.5}$$

where r is the polynomial degree, α_{ij} represents the coefficient of the term in which x_j is power of i. In what follows a function of this type is called **membership function**. The vector $\vec{s}$ will represent a set of k membership functions, in which case the length of $\vec{s}$ is expressed as $l = ((r \cdot d) + d) \cdot k$. This implies an encoding as the one shown in Figure 3.9.

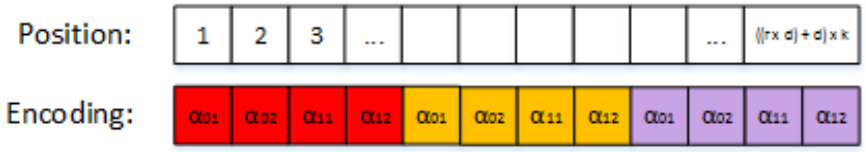


Figure 3.9: Membership function encoding with $r = 1, d = 2, k = 3$.

Unlike the typical proposals (see Figure 3.6, 3.7 and 3.8) the length of our encoding proposal does not depend on variables or properties of the clustering problem as centroids or membership labels. Instead, our proposal encodes a mathematical model for each cluster in which case the length of $\vec{s}$ depends directly on d, r and k. This results in an encoding invariant to the cardinality of X. Additionally, the models encoded by $\vec{s}$ are non-linear functions (see Equation 3.5) that can induce clusters with non-spherical shapes or non-convex hull.

3.3.5 Determining the membership of $\vec{x}$

To determine the membership of $\vec{x}$ to cluster C_i, a membership function f_i (encoded in $\vec{s}$) is applied. Then, the result of such application is used as parameter of a discriminant function ρ whose result will be a binary value that will allow to decide if $\vec{x} \in C_i$. As discriminant function, we use a sigmoid function (commonly used in neural networks [33]). The above process is illustrated in Figure 3.10.

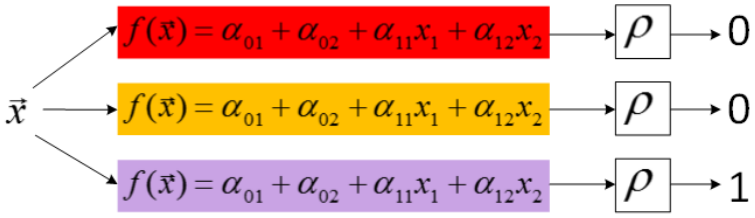


Figure 3.10: Label decoding.

The above gives rise to the following remarks:

1. $\vec{x}$ **does not belong to any cluster.** This means that there is not a membership function whose domain includes $\vec{x}$. In this case, we assign $\vec{x}$ to an artificial cluster denoted as C_{null}.

2. $\vec{x}$ **belongs to more than one cluster.** In this case, $\vec{x}$ is assigned to the cluster represented by the first activated membership function.

3.4 Clustering proposal

Based on the above elements, we discuss about the process in which the best instance of Π can be found.

Let S be a set of vectors $\vec{s}$ encoded in accordance to subsection 3.3. Since $\vec{s}$ induces a partition Π on X, such partition is evaluated via a validity index Q. Based on encoding proposal, given a partition Π, the set C_{null} must be empty. We have included an additional constraint where the cardinality of i^{th} cluster must be greater or equals to 1. Thus, the objective function (see Equation 3.2) can be rewritten as:

$$
\begin{aligned}
&Optimize: \\
&Q(\Pi(\vec{s})) \\
&subject\ to: \\
&|C_i(\Pi(\vec{s}))| \geq 1 \\
&|C_{null}(\Pi(\vec{s}))| = 0 \\
&\vec{s} \in \mathbb{R}^l
\end{aligned}
\tag{3.6}
$$

If $\Pi(\vec{s})$ does not satisfy the constraints, the function $Q(\Pi(\vec{s}))$ is penalized in accordance to Equation 3.3.

For all $\vec{s} \in S$ the function $Q(\Pi)$ is determined. Depending on the chosen meta-heuristic, random alterations and recombination processes are executed to get better instances of Π. This process is repeated until a number of iteration G is reached or a stopping criterion is satisfied. At the end the set S will contain those instances of Π that exhibit the best value of Q. The above is summarized in Algorithm 1 (See Appendix C for a detailed explanation of the implied procedures in Algorithm 1).

Algorithm 1 Clustering algorithm based on validity index optimization.

Finds the best Π of X based on a given validity index Q.

Input parameters:

X: Set of $\vec{x}$ to be clustered.

n: Number of individuals in the solution population.

k: Number of clusters.

d: Dimensions of X.

r: Degree for the membership functions.

1: $S \leftarrow \text{INITIALIZEPOPULATION}(n, k, d, r)$ ▷ Population of encoded clustering solutions $\vec{s}$.

2: $\vec{Q} \leftarrow \text{EVALUATEPOPULATION}(k, r, d, X, S)$ ▷ Vector of fitness values.

3: **while** a stopping criterion is not reached **do**

4: $\text{RECOMBINE}(S)$

5: $\vec{Q} \leftarrow \text{EVALUATEPOPULATION}(k, r, d, X, S)$

6: **end while**

7: **return** $\Pi(\vec{s})$ with the best Q.

3.5 Preliminary experiments

To validate the functionality of our proposal, the Algorithm 1 was designed, implemented and executed with a dataset $X \in \mathbb{R}^2$ that represents a linearly separable problem, as is illustrated in Figure 3.11. For this experiment, we set $k = 3$ and $r = 1$. This preliminary experimentation was performed in order to determine if there was a convergence in the search of the best validity index value and a correct Π of X was obtained.

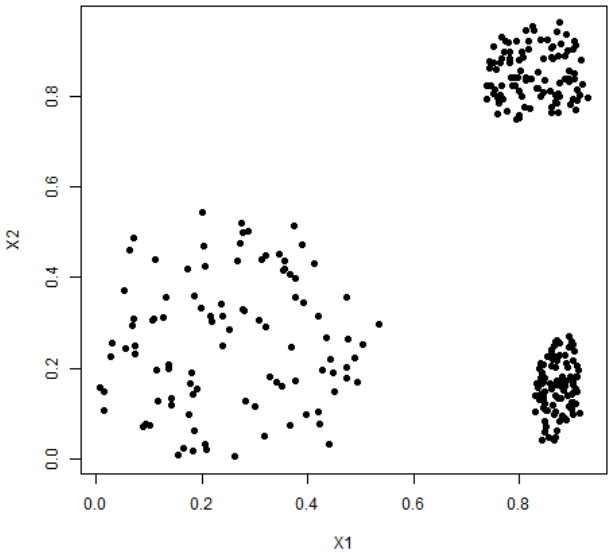


Figure 3.11: Two dimensional dataset X

The Algorithm 1 was executed using Dunn and Dunn index (see Appendix B), with a parameter settings as shown in Table 3.4 for DE and Table 3.5 for EGA. In Figure 3.12 are shown the fittest instances of Π throughout the adaptive process of S. At the end, the set of hyperplanes defined by the membership functions allow us to define a suitable partition on X (at least in this example).

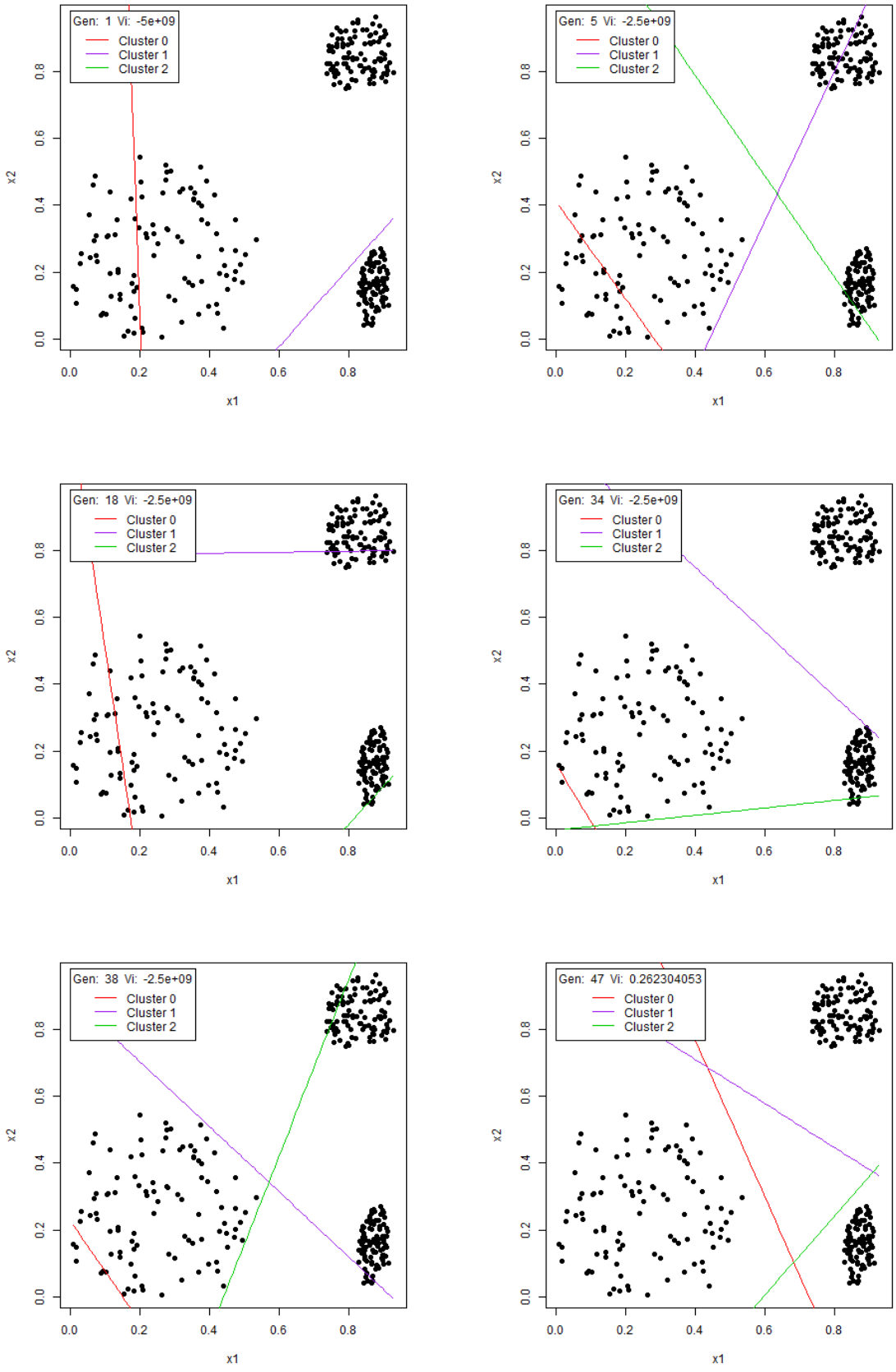


Figure 3.12: Instances of Π at iterations 1, 5, 18, 34, 38, and 47.

For completeness, we illustrate the process in which each membership function (encoded in $\vec{s} \in S$) determines those elements in X that belong it, as follows:

Let $F = \{f_1, f_2, \ldots, f_k\}$ be the set of membership functions encoded by an instance of $\vec{s}$.

1. For all $\vec{x} \in X$ the value i^{th} function $f_i(\vec{x})$ was determined. This value allows us to obtain a projection as it is illustrated in Figure 3.13.

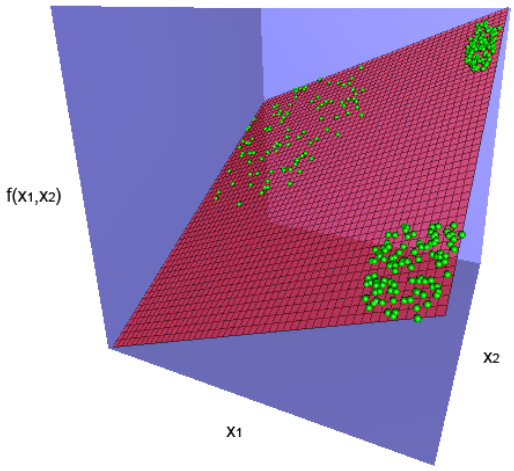


Figure 3.13: Elements $\vec{x}$ of X projected in f_i.

2. The values $f_i(\vec{x})$ were used to determine the value of the sigmoid function $\rho(f_i(\vec{x}))$ which induces a separability among elements of X. This is illustrated in Figure 3.14.

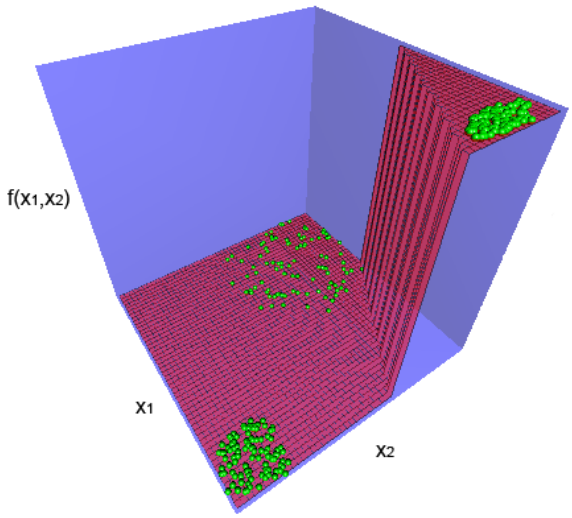


Figure 3.14: Discrimination after applying ρ.

3. In Figure 3.15 are shown the values of all membership functions F after the discriminant function ρ was applied.

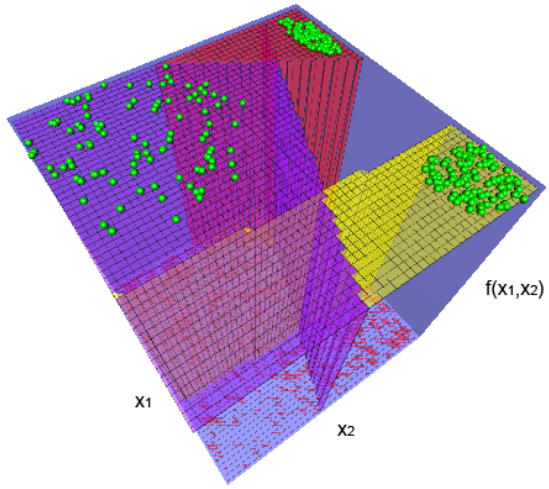


Figure 3.15: All membership functions determine Π.

For the same dataset, we carried out an additional experiment using the Davies Bouldin index (see Appendix B). The partition obtained is illustrated in Figure 3.16.

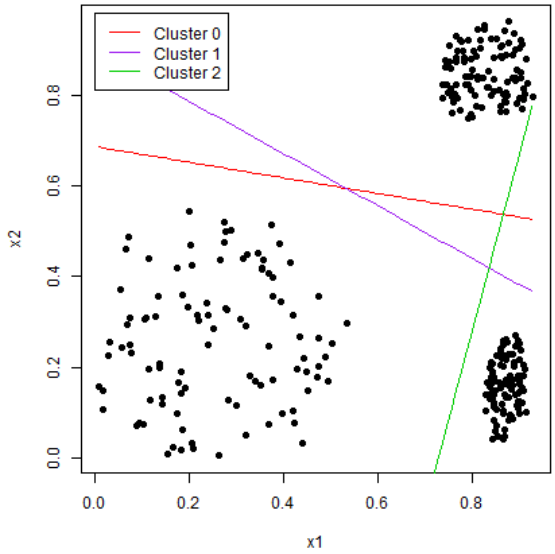


Figure 3.16: Π obtained by using the DB index.

A second preliminary experiment was performed with a dataset that represents a non-linearly separable problem, which is illustrated in Figure 3.17. For this experiment, we set $k = 5$, y $r = 3$.

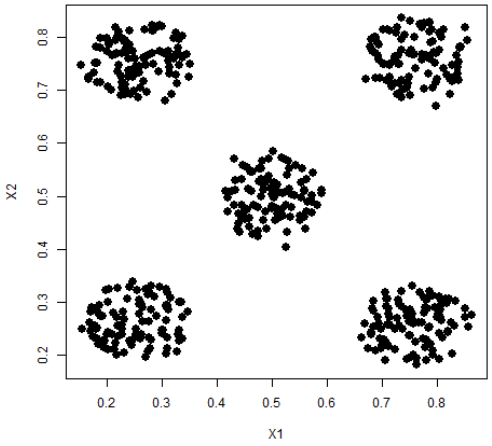
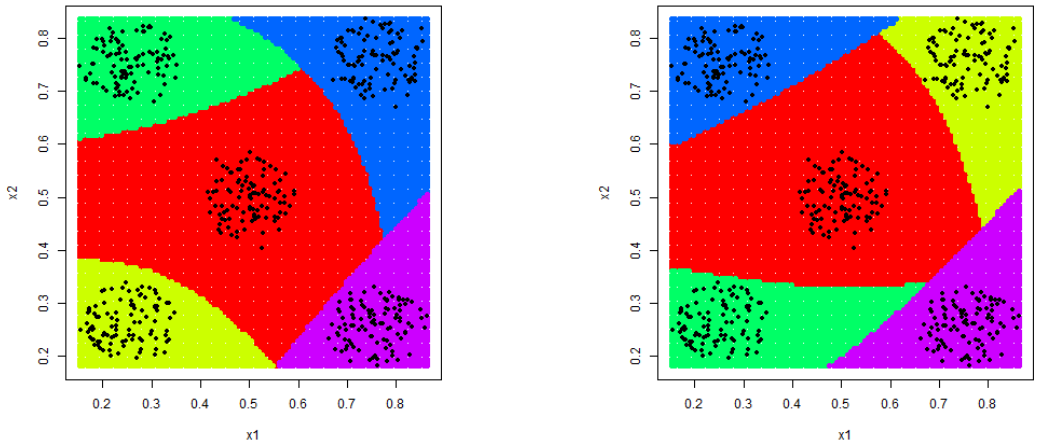


Figure 3.17: Two dimensional dataset X.

In Figure 3.18a and 3.18b are shown the best Πs obtained when DD and DB indices were used.



(a) Π obtained with DD index. (b) Π obtained with DB index.

Figure 3.18

This earlier results show that our proposal is promissory. We can see that such proposal is invariant to the validity index. This allows us to explore more validity indices in order to obtain a

large variety of solutions.

3.5.1 Meta-heuristics parameters

A fundamental factor to determine the performance of any meta-heuristic is the parameters setting. In the literature, many proposal attempting to solve this problem have arisen [30, 58, 76, 79]. However most of them have important dependencies associated to the properties of the problem that disable the generalization of the optimal parameters. For this reason, we conducted an experimental process to determine the most suitable parameters in function of the characteristics of our problem. This process is summarized as follows:

1. A setting parameters was defined as N-tuple of the form $\aleph = [P_1, P_2, \ldots, P_N]$ that includes those variables P_i that are necessary to execute a given meta-heuristic (e.g. population size, crossover and mutation probabilities for EGA and population size, CR and F for DE).

2. A threshold for each parameter $P_i \in \aleph$ was determined in order to define the space of those possible instances of $\aleph$.

3. From such space, a random set (uniformly distributed) of 300 instances of $\aleph$ was obtained.

4. For each instance in the random set, a representative clustering problem is solved 30 times via the Algorithm 1 using EGA and DE.

5. From each series of 30 executions per instance $\aleph$, the average fitness Q' at each meta-heuristic iteration is obtained.

In Figure 3.19 and 3.20 are illustrated the adaptive process of the top five instances of $\aleph$ that achieved the best Q'. Based on these results, we consider a suitable setting as one instance of $\aleph$ with the fastest convergence and the best value of Q'. Such instances are shown in Table 3.6 for EGA and DE.

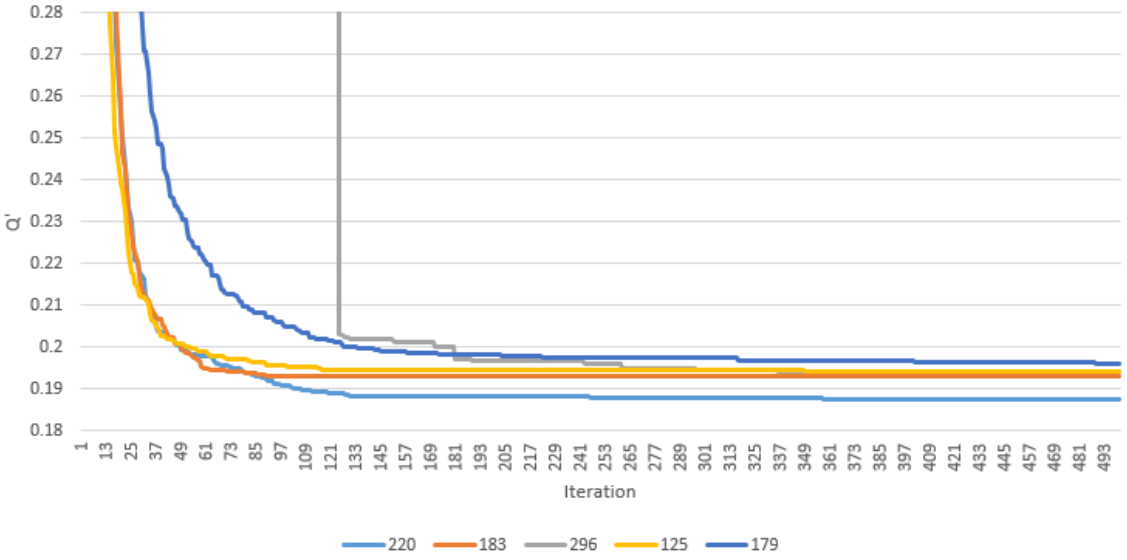


Figure 3.19: Top 5 א performances for EGA.

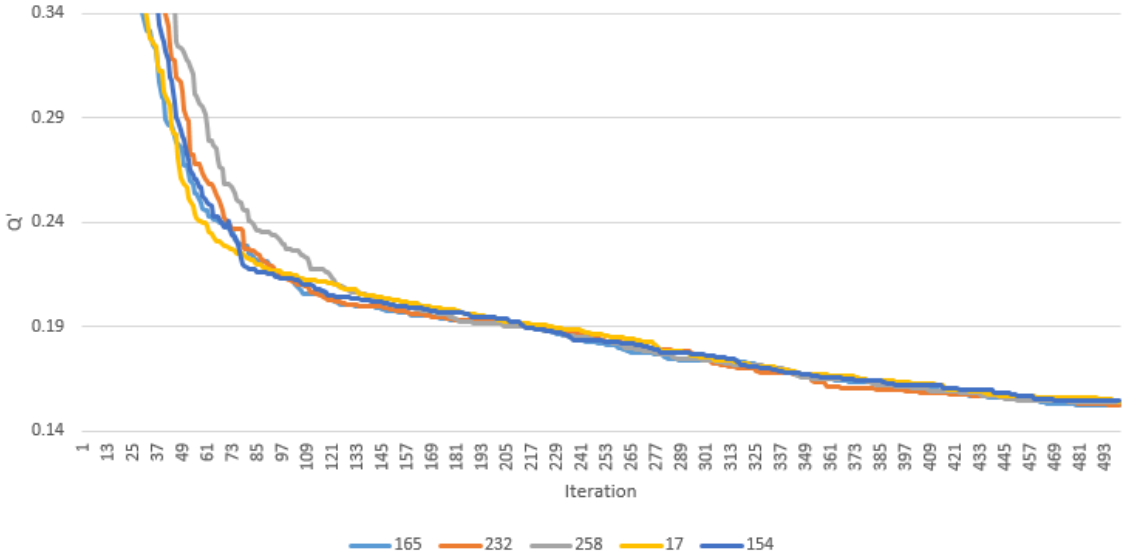


Figure 3.20: Top 5 א performances for DE.

EGA		DE	
Best configuration instance:	$\aleph_{220}$	Best configuration instance:	$\aleph_{165}$
Population size:	83 individuals	Population size:	86 individuals
Crossover probability:	0.6594	CR:	0.0161
Mutation probability:	0.0109	F:	0.3062
Static parameters			
Generations:	500	Generations:	500
Selection:	Deterministic	Repair strategy:	Intermediate

Table 3.6: Meta-heuristic parameter settings.

3.5.2 Algorithm's complexity

In meta-heuristic clustering the complexity is associated to the length of the encoding solutions, which usually depends on the values of k, d and $|X|$. An important concern arises when the value of $|X|$ is large (for example in the order of 1×10^4). To avoid this problem, the proposed encoding (see Subsection 3.3.4) removes such dependency. In this case, our algorithm depends only on k, d and the additional parameter r that represents the degree of the membership functions. Thus, the order of complexity of our algorithm can be expressed as a function of the form $O(k, r, d) = k((r+1) \cdot d)$ obtained from the process of inducing a Π from a given solution $\vec{s}$. Evidently the value of this function is affected by the number N of candidate solutions in S; in which case the order of complexity can be expressed as:

$$O(k, r, d) = [k((r+1) \cdot d)] \cdot N \tag{3.7}$$

The above expression does not include the complexity of the validity index. Under such consideration the order of complexity of our algorithm can be rewritten as:

$$O(k, r, d) = [k((r + 1) \cdot d)] \cdot N + O_{vi}() \tag{3.8}$$

where O_{vi} is the order of complexity of any validity index. In Appendix B we have included the complexity expression of the used indices. Finally, an important variable that affects the complexity is included: the number of iterations G. Therefore, the complexity can be expressed as:

$$O(k, r, d) = G \cdot [[k((r + 1) \cdot d)] \cdot N + O_{vi}()] \tag{3.9}$$

4

Experimental process and results

Based on the results obtained in the preliminary experiments, it has been demonstrated that it is possible to obtain a mathematical model of a cluster by the use of meta-heuristic techniques guided by validity indices. In this chapter is described the followed procedure to determine the performance of the proposed CM which is referred as MCM in what follows.

4.1 Generating synthetic datasets

In order to determine the performance of any clustering approach, datasets that represent clustering problems must be obtained. Many cluster methods attempt to solve particular clustering problems that others cannot solve. Such problems have special properties, which can be given as a priori information to the clustering method. To determine the effectiveness, in this case, test datasets are intentionally created for the purpose of satisfying such properties and conditions, commonly associated to the distribution, arrangement or shape of the clusters that must be found. However, this fact inhibits a generalization about the performance of the clustering method.

The performance of a CM must be relative to ability of other methods to solve the same set of problems. A generalization of the performance will be possible as long as these problems represent a random sample of all clustering problems in a wide numerical space. A systematic process it has been followed to generate numerical datasets in this space. Each dataset contains elements grouped into k clusters which are generated via a set of parametric functions as described in what follows:

Let k, $\aleph_i$ and $\mathbb{F}$ the number of clusters, the size of the i^{th} cluster and a set of generator functions respectively. A cluster will be a set of d-dimensional vectors generated as follows:

1. From $\mathbb{F}$, a subset of functions $f_i : \mathbb{R} \to \mathbb{R}$ are randomly chosen $(i = 1, 2, ..., d)$.
2. A vector of the form $\vec{v} = [f_1(x_1), f_2(x_2), ..., f_d(x_d)]$ is generated. The values of x_i are drawn randomly from the domain of f_i.
3. The step 2 is repeated varying the value of x until $\aleph_i$ vectors have been obtained.

The above process is executed until k clusters have been obtained. In Figure 4.1, examples of clustering problems in $\mathbb{R}^2$ with different values of k are illustrated. We take advantage of the reservoir of functions described in Appendix B to define previously the set $\mathbb{F}$ and generate a total of 95 datasets with $k = 2, 3, \ldots, 20$.

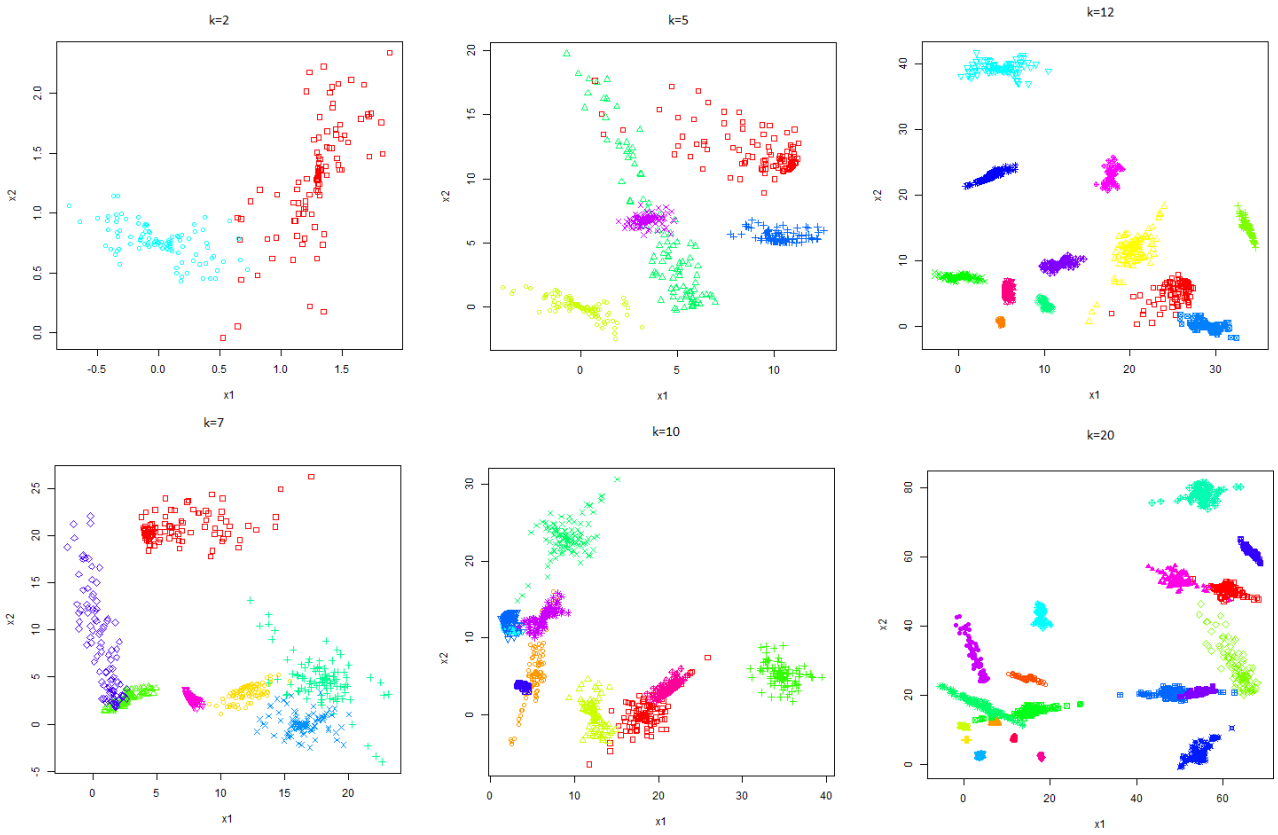


Figure 4.1: Examples of synthetic datasets in $\mathbb{R}^2$ with different values of k

4.2 Experimental design

Since it is assumed that the distribution of the validity indices used in the performance test is unknown and because of the random nature in which the used datasets described in the previous section were generated; we rely on the central limit theorem [86]. For this, the experiment design was performed as follows:

1. For each of these datasets a clustering solution Π is obtained via a clustering method A.
2. For each solution the value of an index Q is calculated and denoted as Q_i.
3. The steps 1-2 are repeated until $i = 31$ per dataset.

For completeness the above procedure was executed using the proposed method with the EGA and DE meta-heuristics. Also, the k-means [40] and SOM [48] methods were executed in order to have a comparison against conventional CMs. All these experiments were executed using the Dunn and Dunn, Davies-Bouldin and SD validity indices (See Appendix B). Such combinations of CMs and validity indices result in a total of $35,340$ executions which are enough to approximate a normal distribution.

The execution of this experimental phase was made using a server with the following properties:

- Two Unit rack server.
- Four processors Intel R Xeon R Processor E7-4830 v3, 12 Cores (30M Cache, 2.10. GHz) Intel R QPI 8 GT/s speed, FCLGA2011 (a total of 48 cores).
- 128 GB (16X8GB) DDR3-1600 1.35V 2Rx4 ECC REG DIMM memory.

4.3 Results and analysis

After the execution of the experiments described in the section above the results are described as follows:

In Table 4.1 the summary of the experiment results using the Dunn and Dunn index is shown. Since the definition of this index states that a higher value denotes a better clustering quality, the higher Q values were obtained by the MCM as it can be seen in column *Max*. However this may denote only one result in the execution series. In the μ column the mean of all obtained Q's is shown, in here, also the higher Q's were obtained by the MCM. Column σ denotes the standard deviation of all Q's in where it can be observed that the MCM shows a higher σ, this behavior suggest that more variety of Πs are obtained derived from the better exploration of the search space rather than k-means and SOM whose solutions may fall in local optimal.

Algorithm	μ	σ	Min	Max
MCM-EGA	0.0341	0.0020	0.0009	0.3014
MCM-DE	0.0475	0.0024	0.0004	0.3014
KMEANS	0.0120	0.0009	0.0006	0.0923
SOM	0.0094	0.0006	0.0004	0.0949

Table 4.1: Dunn and Dunn index results.

Table 4.2 shows those results for the experimental process with the Davies-bouldin index, here a lower value of Q denotes a better Π; as the previous index the better Q value was obtained by the MCM as is shown in column *Min*. The same effect is observed in σ in which a wider solution spectrum is shown by MCM-EGA and MCM-DE, however for this validity index k-means shows a lower μ than MCM-DE which suggest that for this index MCM-DE does not have a better performance than k-means. Nevertheless, by using the MCM with the EGA shows a better performance than k-means and SOM.

Algorithm	μ	σ	Min	Max
MCM-EGA	0.3780	0.0130	0.0849	1.0360
MCM-DE	0.4593	0.0277	0.0803	1.9888
KMEANS	0.3979	0.0059	0.1516	0.5911
SOM	0.4603	0.0055	0.1509	0.7448

Table 4.2: Davies-bouldin index results.

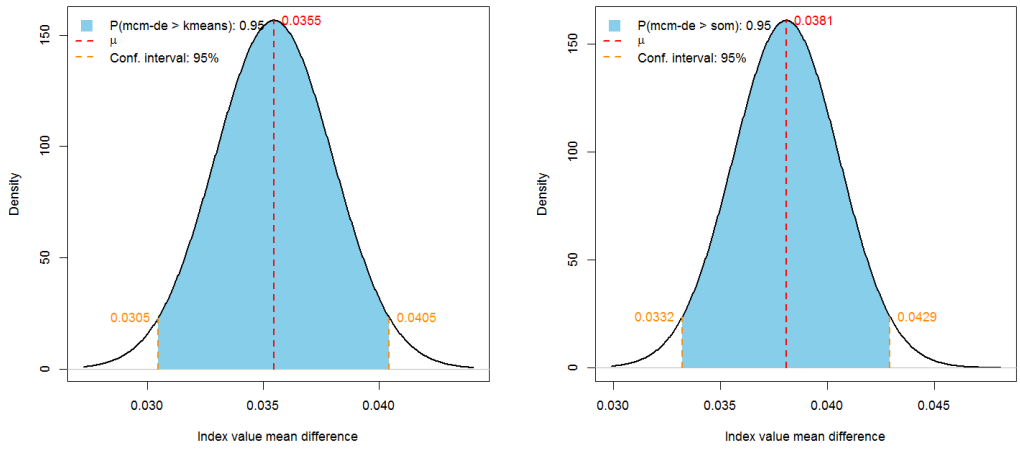
For the SD index the Table 4.3 shows the summary of the experimental results, like the previous index a lower Q denotes a better Π. Once again, the lower values in column *Min* were obtained by the MCM. However, k-means obtained a better μ than the obtained by MCM-DE as with the previous validity index. For the EGA, the MCM showed a better μ rather than the rest of the CMs.

Algorithm	μ	σ	Min	Max
MCM-EGA	0.5649	0.1741	0.0915	11.6182
MCM-DE	0.9895	0.4376	0.0852	13.2721
KMEANS	0.9338	0.0722	0.1675	5.2469
SOM	24.6529	1.8210	3.4092	135.7860

Table 4.3: SD index results.

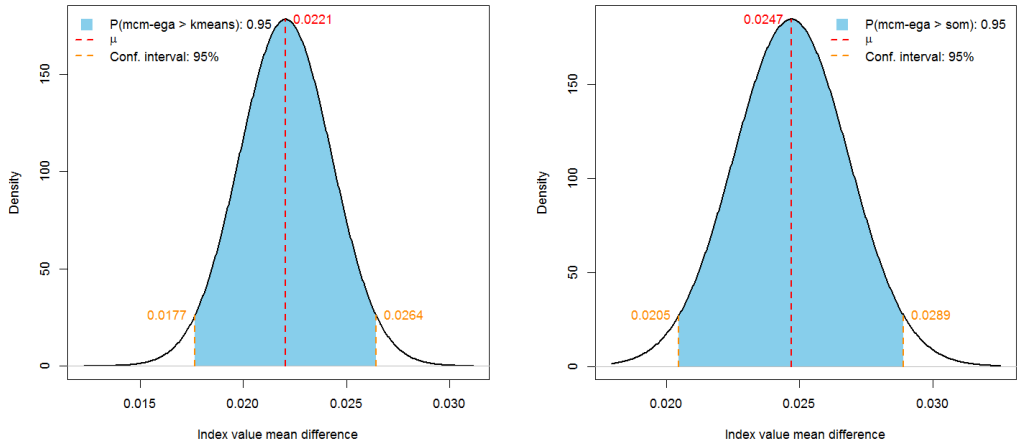
To add statistical significance to the results presented above let the Figure 4.2, 4.3 and 4.4 be the distribution difference between two means [86]. This statistical analysis shows the probability that the MCM get better values of Q against k-means and SOM, for both DE and EGA. For this, a confidence interval of 95% was established which delimits the area of interest under the curve.

Figure 4.2 shows the results for the Dunn and Dunn index in which, for DE (4.2a and 4.2b) and EGA (4.2c and 4.2d) the MCM obtains Πs with a better quality with a probability of 95% rather than using k-means or SOM. Since the Dunn and Dunn index is a maximization problem, the area of interest under the curve is denoted by the area in $\mathbb{R}^+$.



(a) Probability that MCM-DE outperforms Kmeans.

(b) Probability that MCM-DE outperforms SOM.



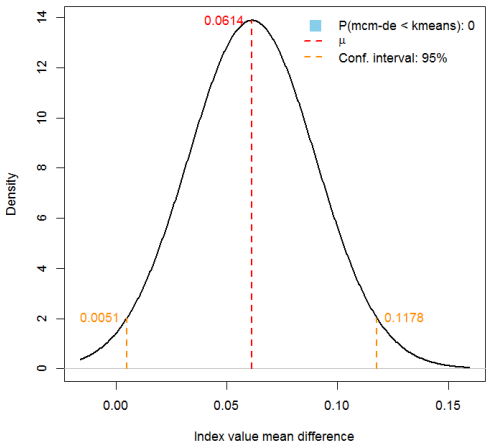
(c) Probability that MCM-EGA outperforms Kmeans.

(d) Probability that MCM-EGA outperforms SOM.

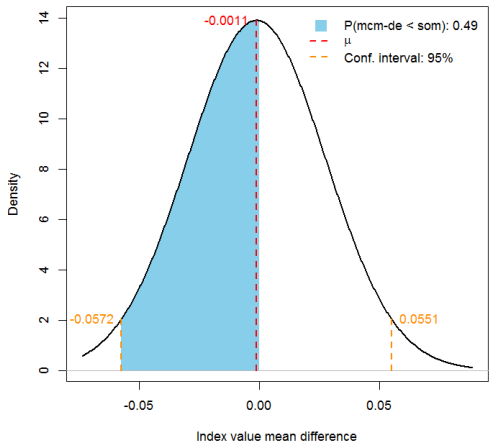
Figure 4.2: Dunn and Dunn index comparative results.

Figure 4.3 shows the results for the Davies bouldin index. Since this index is a minimization problem, the area of interest under the curve is denoted by the area in $\mathbb{R}^-$. Here, a better performance of the MCM is reflected by using the EGA (4.3c and 4.3d) since this gets higher probability of outperforms k-means and SOM. The results of DE against k-means (4.3a) suggest that at least for this validity index the MCM will never get better Q in the Π obtained rather than using k-means. As

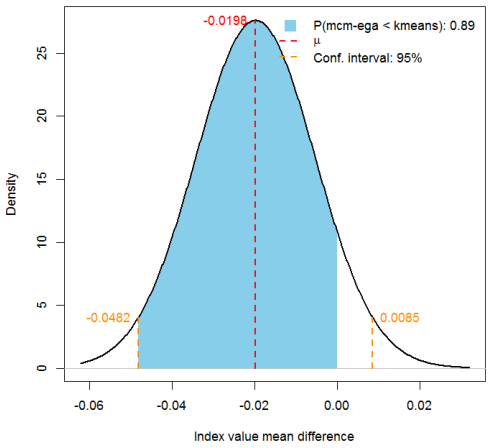
for SOM (4.3b), it makes no difference using the MCM with DE or using SOM, since the probability of getting the same Q is almost the same.



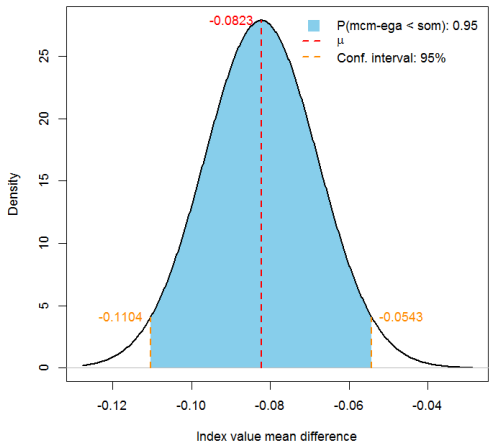
(a) Probability that MCM-DE outperforms Kmeans.



(b) Probability that MCM-DE outperforms SOM.



(c) Probability that MCM-EGA outperforms Kmeans.

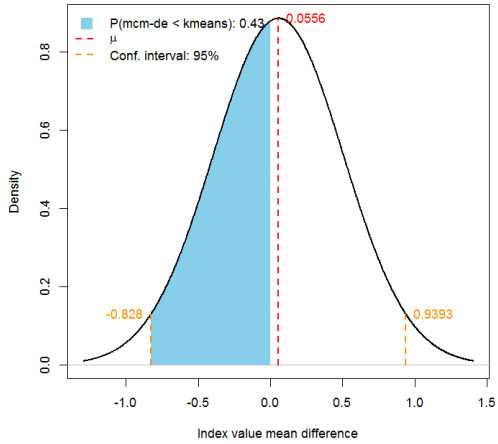


(d) Probability that MCM-EGA outperforms SOM.

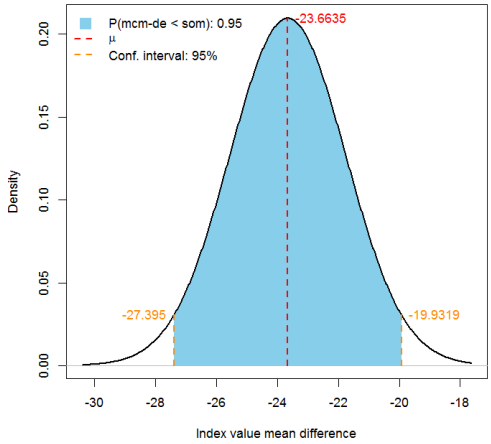
Figure 4.3: Davies-Bouldin index comparative results.

Figure 4.4 shows the results for the SD index. The SD index is a minimization problem, in which case the area of interest under the curve is denoted by the area in $\mathbb{R}^-$. Here, as the previous index, a better performance of the MCM is reflected by using the EGA (4.4c and 4.4d) since this gets

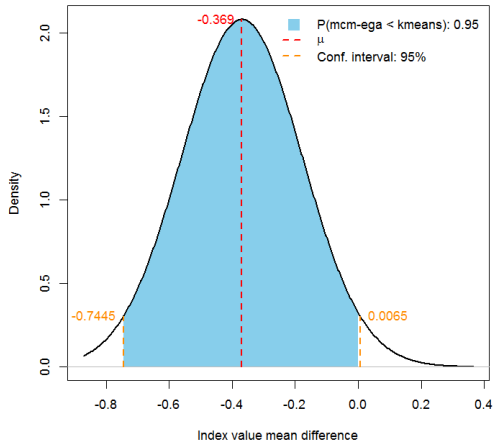
higher probability of exceeding k-means and SOM. In the case of using DE against k-means (4.4a) the experimental results show no difference in using DE or k-means since they reflect almost the same probability of getting the same Q. For DE against SOM (4.4b) a higher probability of getting Π with better Q is shown by the MCM.



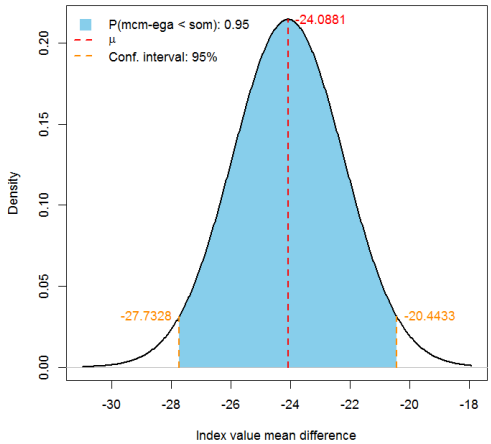
(a) Probability that MCM-DE outperforms Kmeans.



(b) Probability that MCM-DE outperforms SOM.



(c) Probability that MCM-EGA outperforms Kmeans.



(d) Probability that MCM-EGA outperforms SOM.

Figure 4.4: SD index comparative results.

4.3.1 Complementary analysis

One important aspect of a CM is the number of clusters obtained by any method. Since sometimes CMs may fall on local optimal solutions the k clusters found by a CM may not be the required k in the input parameter of the algorithm. In such case Figures 4.5, 4.6, 4.7 and 4.8 show this analysis over experimental results with the three used validity indices.

Figure 4.5 shows the k clusters obtained for k-means. As is shown, for both values, average ($+$) and mode ($\times$) are fairly distanced from the expected k value ($\square$). This suggests that k-means is a good CM for finding a Π with a desired k even when this input value may be high.

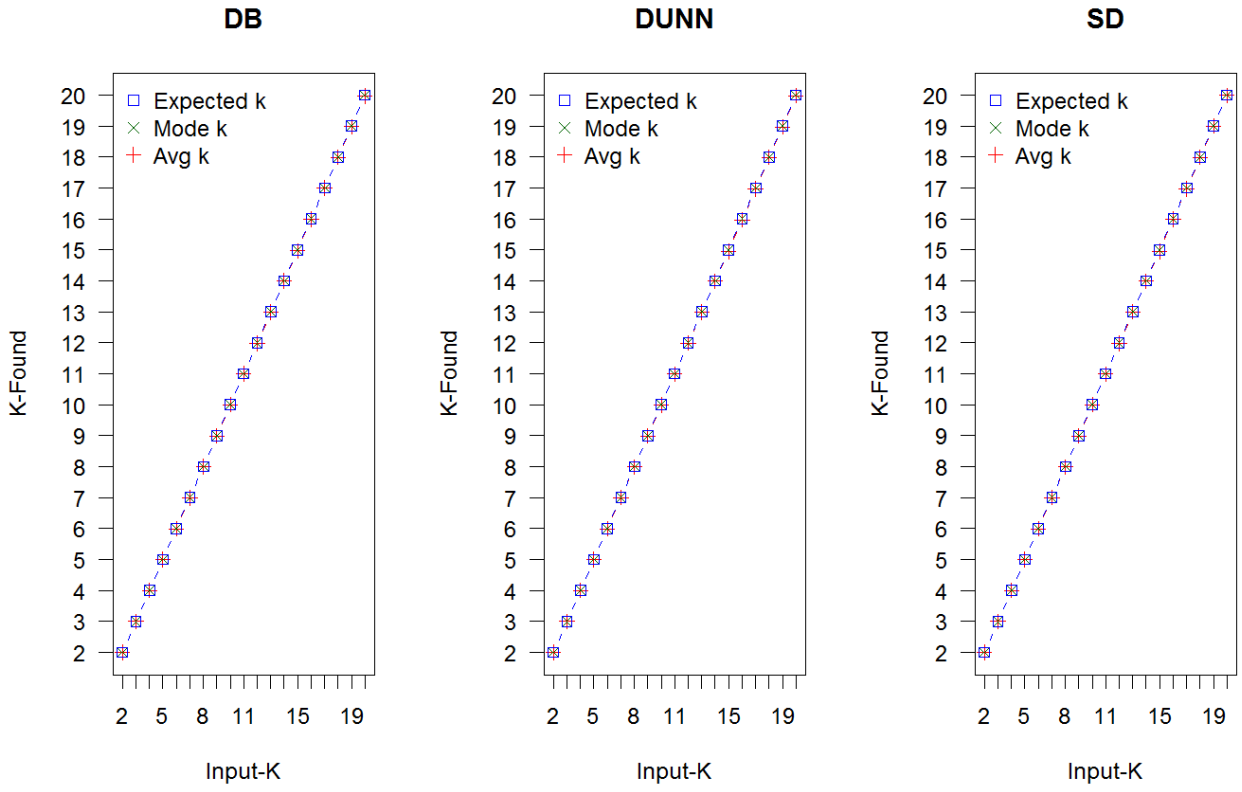


Figure 4.5: Number of clusters obtained with Kmeans.

Figure 4.6 shows those k obtained by SOM. For this CM, it can be seen that even for simple problems of finding $k = 3$ clusters the returned k for both, average and mode, may vary which is not a desired behavior on a CM.

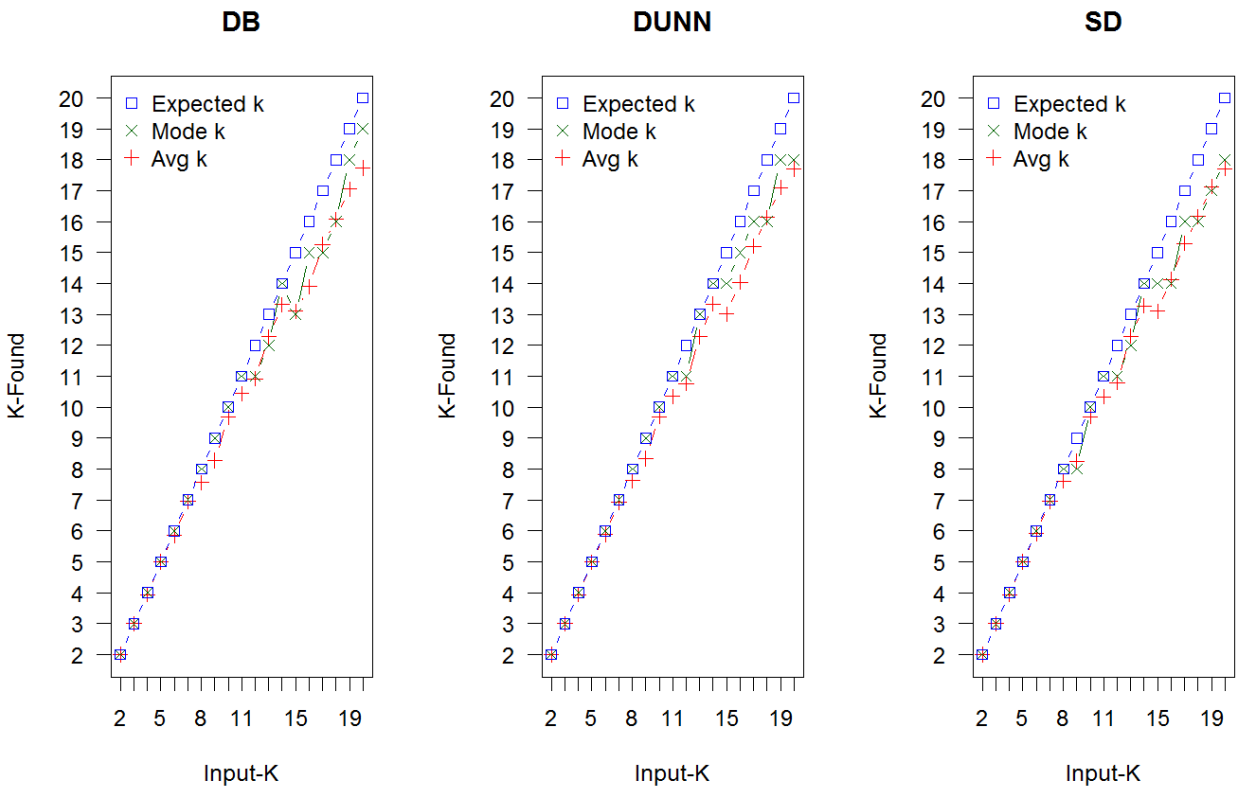


Figure 4.6: Number of clusters obtained with SOM.

The results for the MCM using DE are shown in Figure 4.7 in which a good performance in average and mode for the k found clusters is reflected from $2 \leq k \leq 6$ for the three used validity indices. This behavior is due to the implied complexity of the problem in which the meta-heuristic tries to fix the cluster model denoted by the membership functions, such problem gets more difficult as the value of k arises. Since all the executions of the MCM were done with the same number of generations this suggests that the meta-heuristic may need more generations to find the best Πs with values of $k \geq 6$.

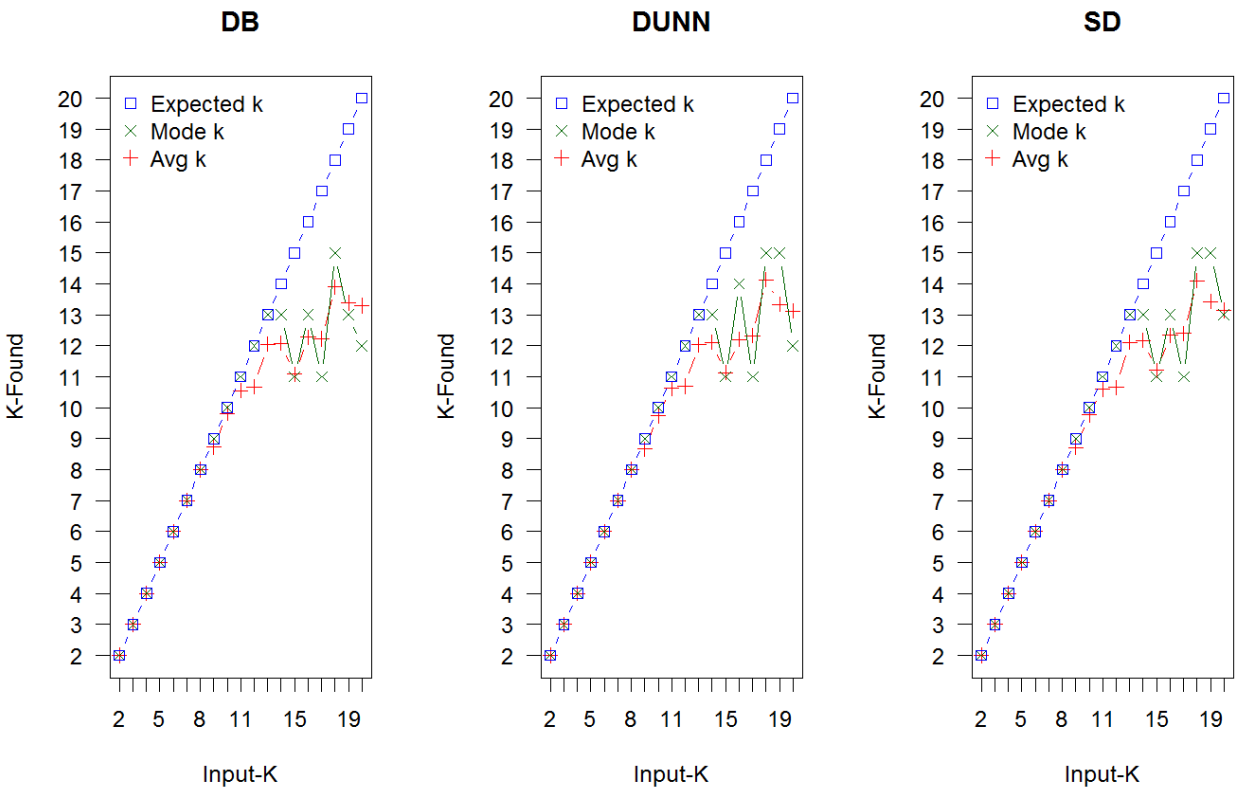


Figure 4.7: Number of clusters obtained with MCM-DE.

The results for the MCM using the EGA are shown in Figure 4.8 where, as the DE, good average performance was obtained also with $2 \leq k \leq 6$; however in this case, as the value of k arises the average k returned is fairly separated from the expected k. One important aspect in this result is that all values shown for the mode k obtained are the expected ones, this reflects the effectiveness of the EGA in solving harder problems of partitioning, which are better than the obtained with the DE and SOM. This behavior is the same for all used validity indices.

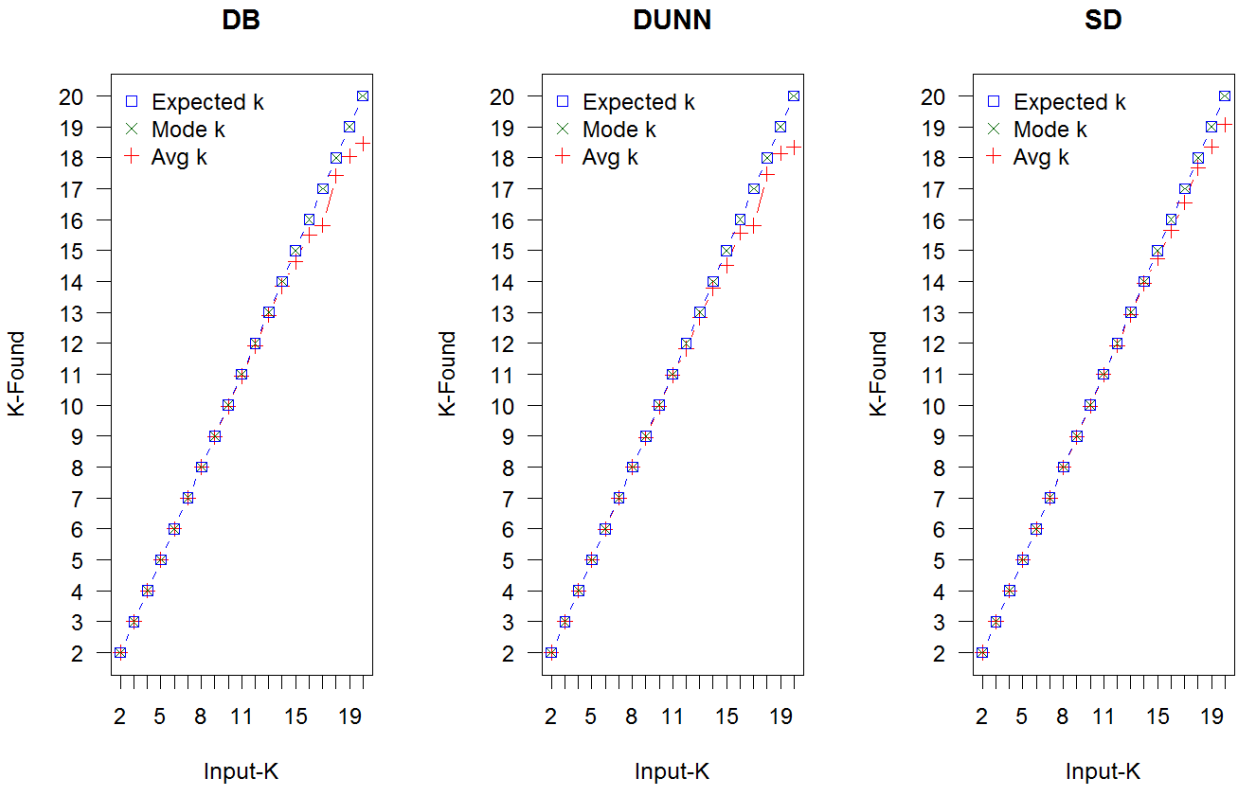


Figure 4.8: Number of clusters obtained with the MCM-EGA.

5

Conclusions and future work

We have proposed a meta-heuristic CM whose adaptive process is driven by a validity index. The experiments showed that a set of different solutions can be obtained based on a set of validity indices. Our method guarantees better solutions (relative to a given index) than those achieved by traditional methods such as k-means and SOM. Based on a probabilistic approach, we found that in most cases the proposed CM outperforms k-means and SOMs. Additionally, a set of earlier experiments showed that in those cases where there is a linear separability, any method is suitable. When such separability did not exist, a partial order among methods arose, wherein the highest order was the MCM. This allows us to assert that the MCM is specially suitable to obtain optimal partitions on datasets that involve hard clustering problems.

Since our method is invariant to any index, this is able to obtain a wide spectrum of clustering solutions. The above is consequence of a novel encoding proposal. Although the MCM was tested for a common set of validity indices, it is sufficiently versatile to include any other index, from which the best Π on X will be found. It is possible to propose new validity indices that evaluate desired

properties of the clusters, based on prior information about the problem that a X represents. For instance, given a gray scale image, we may want to find k clusters of pixels in which the intensity of the pixels is as minimal as possible. Defining a proper index, the MCM is able to find the best Π on the image that satisfy the desired properties. In this sense, the MCM can be considered a general CM that does not depend on a single criterion to guide the search of the clusters.

The main disadvantage of our method is its computational complexity. As important elements of such complexity, we can point out 1) the complexity of the index and 2) parameters associated to the used meta-heuristic (number of individuals, iterations). To solve this disadvantage, we can resort to techniques of parallel computing defining those components of our algorithm that can be executed in a concurrent way. A secondary disadvantage is the type of data that the MCM accepts, since this method accepts datasets of elements represented as numerical vectors, a transformation of the elements in the dataset must be done in case these ones are of any other type.

As contribution, unlike traditional methods, our proposal offers an explanatory model (given by a polynomial representation) of the partition Π on X that allows us to determine the membership of a new object (a vector $\vec{x}$ not necessarily belonging to X) without executing the search process of Π again. Another advantage of this method is the independence of the dataset's distribution for the CM functionality, since this one is measured by the used validity index. As a proof of this an implementation of the MCM had been done during the experimental process to determine the effectiveness of this one.

Finally, we consider that the proposed polynomial described in Equation 3.5 can be extended by adding terms that allow to increase freedom degrees to obtain a better representation of the clusters in Π.

A

Optimization problems

Unconstrained objective functions:

1. Function A

$$f(x) = \left[1 - \left(\frac{11}{2}x - \frac{7}{2}\right)^2\right] \cdot \left[\cos\left(\frac{11}{2}x - \frac{7}{2}\right) + 1\right] + 2 \tag{A.1}$$

   Input domain: $x \in [0, 1]$

   Global maximum: $f(x) = 4$

2. Hansen

$$f(x_0, x_1) = \sum_{i=0}^{4}(i+1)\cos(ix_0 + i + 1)\sum_{j=0}^{4}(j+1)\cos((j+2)x_1 + j + 1) \tag{A.2}$$

   Input domain: $|x_i| \leq 10$

   Global minimum: -176.54

3. DeJong

$$f(x) = \sum_{i=1}^{n} x_i^2 \tag{A.3}$$

Input domain: $-5.12 \leq x_i \leq 5.12, i = 1, \ldots, n$

Global minimum: 0

4. Axis parallel hyper ellipsoid

$$f(x) = \sum_{i=1}^{n} i \cdot x_i^2 \tag{A.4}$$

Input domain: $-5.12 \leq x_i \leq 5.12, i = 1, \ldots, n$

Global minimum: 0

5. Rotated hyper ellipsoid

$$f(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \tag{A.5}$$

Input domain: $-65.536 \leq x_i \leq 65.536, i = 1, \ldots, n$

Global minimum: 0

6. Rosenbrock

$$f(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right] \tag{A.6}$$

Input domain: $x_i \in [-2.048, 2.048], i = 1, \ldots, n$

Global minimum: 0

7. Rastringin

$$f(x) = 10n + \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) \right] \tag{A.7}$$

Input domain: $-5.12 \leq x_i \leq 5.12, i = 1, \ldots, n$

Global minimum: 0

8. Schwefel

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

Input domain: $-500 \leq x_i \leq 500, i = 1, \ldots, n$

Global minimum: 0

9. Griewangk

$$f(x) = \frac{1}{1400} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{A.9}$$

Input domain: $-600 \leq x_i \leq 600, i = 1, \ldots, n$

Global minimum: 0

10. Sum of different power

$$f(x) = \sum_{i=1}^{n} |x_i|^{i+1} \tag{A.10}$$

Input domain: $-1 \leq x_i \leq 1, i = 1, \ldots, n$

Global minimum: $-418.9828n$

11. Ackley

$$f(x) = -a \exp\left(-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(cx_i)\right) + a + \exp(1) \tag{A.11}$$

Input domain: $-32.768 \leq x_i \leq 32.768, i = 1, \ldots, n$

Recommended values: $a = 20, b = 0.2, c = 2\pi$

Global minimum: $-418.9828n$

12. Langermann

$$f(x) = \sum_{i=1}^{m} c_i \exp\left(-\frac{1}{\pi}\sum_{j=1}^{n}(x_j - A_{ij})^2\right) \cos\left(\pi \sum_{j=1}^{n}(x_j - A_{ij})^2\right) \tag{A.12}$$

Input domain: $x_i \in [0, 10], i = 1, \ldots, n$ Recommended values: $m = 5, c = (1, 2, 5, 2, 3)$ and:

$$A = \begin{pmatrix} 3 & 5 \\ 5 & 2 \\ 2 & 1 \\ 1 & 4 \\ 7 & 9 \end{pmatrix}$$

Global minimum: -4.1

13. Michalewicz

$$f(x) = -\sum_{i=1}^{n} \sin(x_i) \sin^{2m}\left(\frac{ix_i^2}{\pi}\right) \tag{A.13}$$

Input domain: $0 \leq x_i \leq \pi, i = 1, \ldots, n$

Recommended values: $m = 10$

Global minimum:

for $n = 2$, $f(x) = -1.8013$

for $n = 5$, $f(x) = -4.687658$

for $n = 10$, $f(x) = -9.66015$

14. Branin

$$f(x) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1 - t)\cos x_1 + s \tag{A.14}$$

Input domain: $x_1 \in [-5, 10], x_2 \in [0, 15]$

Recommended values: $a = 1, b = \frac{5.1}{4\pi^2}, c = \frac{5}{\pi}, r = 6, s = 10$ and $t = \frac{1}{8\pi}$

Global minimum: 0.397887

15. Easom

$$f(x) = -\cos x_1 \cos x_2 \exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2) \tag{A.15}$$

Input domain: $x_1 \in [-100, 100], x_2 \in [-100, 100]$

Global minimum: -1

16. Goldstein-price

$$\begin{aligned} f(x) = &[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \\ &\times[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \end{aligned} \tag{A.16}$$

Input domain: $x_1 \in [-2, 2], x_2 \in [-2, 2]$

Global minimum: 3

17. Six-hump camel back

$$f(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \tag{A.17}$$

Input domain: $x_1 \in [-3, 3], x_2 \in [-2, 2]$

Global minimum: -1.0316

18. Drop Wave

$$f(x) = -\frac{1 + \cos(12\sqrt{x_1^2 + x_2^2})}{0.5(x_1^2 + x_2^2) + 2} \tag{A.18}$$

Input domain: $x_i \in [-5.12, 5.12], i = 1, 2$

Global minimum: -1

19. Shubert

$$f(x) = \left(\sum_{i=1}^{5} i\cos((i+1)x_1 + i)\right)\left(\sum_{i=1}^{5} i\cos((i+1)x_2 + i)\right) \tag{A.19}$$

Input domain: $x_i \in [-5.12, 5.12], i = 1, 2$

Global minimum: -186.7309

20. Schaffer

$$f(x) = 0.5 + \frac{\sin^2(x_1^2 + x_2^2) - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2} \tag{A.20}$$

Input domain: $x_i \in [-100, 100], i = 1, 2$

Global minimum: 0

21. McCormick

$$f(x) = \sin(x_1 + x_2) + (x_1 - x_2)^2 - 1.5x_1 + 2.5x_2 + 1 \tag{A.21}$$

Input domain: $x_1 \in [-1.5, 4], x_2 \in [-3, 4]$

Global minimum: -1.9133

22. Moved axis parallel hyper-elipsoid

$$f(x) = \sum_{i=1}^{n} 5_i x_i^2 \tag{A.22}$$

Input domain: $x_i \in [-5.12, 5.12], i = 1, 2, \ldots, n$

Global minimum: 0

Constrained objective functions:

1. Lamparas

$$f(x) = 15x_1 + 10x_2 \tag{A.23}$$

Input domain: $x_i \geq 0, i = 1, 2$

Subject to:

$c_1 : \frac{x_1}{3} + \frac{x_2}{2} \leq 100$

$c_2 : \frac{x_1}{3} + \frac{x_2}{6} \leq 80$

Global maximum: 3750

2. Constraint A

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \tag{A.24}$$

Input domain: $x_1 \in [-0.5, 0.5], x_2 \leq 1$

Subject to:

$c_1 : x_1 + x_2^2 \geq 0$

$c_2 : x_1^2 + x_2 \geq 0$

Global minimum: 0.25

3. Constraint B

$$f(x) = -x_1 - x_2 \tag{A.25}$$

Input domain: $x_1 \in [0, 3], x_2 \in [0, 4]$

Subject to:

$c_1 : x_2 \leq 2x_1^4 - 8x_1^3 + 8x_1^2 + 2$

$c_2 : x_2 \leq 4x_1^4 - 32x_1^3 + 88x_1^2 - 96x_1 + 36$

Global minimum: -5.5079

4. Constraint C

$$f(x) = (x_1 - 10)^3 + (x_2 - 20)^3 \tag{A.26}$$

Input domain: $x_1 \in [13, 100], x_2 \in [0, 100]$

Subject to:

$c_1 : (x_1 - 5)^2 + (x_2 - 5)^2 - 100 \geq 0$

$c_2 : -(x_1 - 6)^2 - (x_2 - 5)^2 + 82.81 \geq 0$

Global minimum: -6961.8138

5. Constraint D

$$f(x) = 0.01x_1^2 + x_2^2 \tag{A.27}$$

Input domain: $x_1 \in [2, 50], x_2 \in [0, 50]$

Subject to:

$c_1 : x_1 x_2 - 25 \geq 0$

$c_2 : x_1^2 + x_2^2 \geq 0$

Global minimum: 5

6. Constraint E

$$f(x) = (x_1 - 2)^2 + (x_2 - 1)^2 \tag{A.28}$$

Subject to:

$c_1 : -x_1^2 + x_2 \geq 0$

$c_2 : x_1 + x_2 \leq 2$

Global minimum: 1

7. Constraint F

$$f(x) = \exp(x_1 x_2 x_3 x_4 x_5) \tag{A.29}$$

Input domain:

$x_i \in [-2.3, 2.4], i = 1, 2$

$x_i \in [-3.2, 3.2], i = 3, 4, 5$

Subject to:

$c_1 : \sum_{i=1}^{5} x_i^2 = 10$

$c_2 : x_2 x_3 - 5 x_4 x_5 = 0$

$c_3 : x_1^3 + x_2^3 = -1$

Global minimum: 0.053949

8. Constraint G

$$f(x) = -x \cdot sgn(x) \tag{A.30}$$

Subject to:

$c_1 : -1 \leq x \leq 2$

Global minimum: 2

9. Constraint H

$$f(x) = 5x_1 + 4x_2 \tag{A.31}$$

Input domain:

$x_i \geq 0, i = 1, 2$

Subject to:

$c_1 : x_1 + 2x_2 \leq 6$

$c_2 : 6x_1 + 4x_2 \leq 24$

$c_3 : -x_1 + x_2 \leq 1$

Global maximum: 21

B

Validity indices

A validity index is used to measure the quality of a clustering results obtained through of a clustering method. Some of the most used are described in what follows:

B.1 Dunn and Dunn (DD) index

The Dunn and Dunn validity index attempts to identify compact and well separated clusters [22]. This is defined by following equation for a specific number of clusters.

$$D = min_{i=1,..,k} \left\{ min_{j=i+1,...,k} \left\{ \frac{d(C_i, C_j)}{max_{m=1,...,k} diam(C_m)} \right\} \right\} \tag{B.1}$$

where k is the number of clusters, $d(C_i, C_j)$ is the dissimilarity function between two clusters C_i and C_j defined as:

$$d(C_i, C_j) = min_{x \in C_i, y \in C_j} d(x, y) \tag{B.2}$$

and $diam(c_m)$ is the diameter of the m-th cluster which may be considered as a measure of dispersion of the clusters and defined as follows:

$$diam(C_m) = max_{x,y \in C_m} \{d(x,y)\} \tag{B.3}$$

If the data set contains well-separated clusters, the distance between clusters is usually large and the diameter of the clusters is expected to be small. Therefore a large value means better clustering results.

Index complexity:

$$O(X,k) = \left(\frac{|X|}{k}\right)^2 \binom{k}{2} + k\binom{\frac{|X|}{k}}{2} + 1 \tag{B.4}$$

B.2 Davies-Bouldin (DB) index

A similarity measure R_{ij} between the clusters C_i and C_j is defined based on a measure of dispersion of a cluster denoted as s_i and a dissimilarity measure between two clusters denoted as d_{ij}. The R_{ij} index is defined to satisfy the following conditions:

1. $R_{ij} \geq 0$
2. $R_{ij} = R_{ji}$
3. if $s_i = 0$ and $s_j = 0$ then $R_{ij} = 0$
4. if $s_j > s_k$ and $d_{ij} = d_{ik}$ then $R_{ij} > R_{ik}$
5. if $s_j = s_k$ and $d_{ij} < d_{ik}$ then $R_{ij} > R_{ik}$

These conditions state that R_{ij} is non-negative and symmetric. It is given by:

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{B.5}$$

The dissimilarity measure d_{ij} and the dispersion s_i are defined as follows:

$$d_{ij} = d(v_i, v_j) \tag{B.6}$$

$$s_i = \frac{1}{|C_i|} \sum_{x \in C_i} d(x, v_i) \tag{B.7}$$

where v_i is the centroid of the cluster C_i . The DB index measures the average of similarity between each cluster an its most similar one [18]. The lower value of DB means better clustering result due to the clusters have to be compact and separated. The value of the index is calculated as follows:

$$DB = \frac{1}{k} \sum_{i=1}^{k} R_i \tag{B.8}$$

where $R_i = max_{i=1,...,k,\, j \neq i} \{R_{ij}\},\ i = 1, ...k$

Index complexity:

$$O(X, k) = 2k \left(\frac{|X|}{k} + 1 \right) + 3 \binom{k}{2} + k + 1 \tag{B.9}$$

B.3 Scattering and Dispersion (SD) index

The Scattering and Dispersion validity index is defined based on the concepts of the average scattering for clusters and total separation between clusters [61]. Let $\sigma(C_i)$ be the vector variance of a i-th cluster and $\sigma(X)$ the vector variance of the set X such that the average scattering is given by:

$$Scat = \frac{1}{k} \sum_{i=1}^{k} \frac{\parallel \sigma(C_i) \parallel}{\parallel \sigma(X) \parallel} \tag{B.10}$$

The dispersion or total separation between clusters is given by following equation:

$$Dis = \frac{D_{max}}{D_{min}} \sum_{i=1}^{k} \left[\sum_{j=1}^{k} |v_i - v_j| \right]^{-1} \tag{B.11}$$

where $D_{max} = max(|v_i - v_j|)$ and $D_{min} = min(|v_i - v_j|)\ \forall i,\ j \in \{1, 2, 3, \ldots, k\}$ are the maximum and minimum distance between cluster centroids v_i respectively. Now, we can define a validity index based on equations above, as follows

$$SD = \alpha \cdot Scat + Dis \tag{B.12}$$

where α is a weighting factor equal to Dis value in case of maximum number of clusters. Lower SD index means better cluster configuration, it means that the clusters are compact and separated.

Index complexity:

$$O(X, k) = 3K^2 + 3K + 9|X| + 2\binom{K}{2} + 6 \tag{B.13}$$

C

Algorithms

Algorithm 2 Procedure initialize population.

Initialize the population for the clustering problem.

Input parameters:

n: Number of individuals in the population.

k: Number of clusters to group the given patterns.

d: Dimensions of the dataset.

r: Degree for the membership functions.

1: **procedure** INITIALIZEPOPULATION(n, k, d, r)

2: $S \leftarrow \emptyset$

3: $S \leftarrow$ Generate n individuals $\vec{s} \in \mathbb{R}^l$ where $l = ((r \cdot d) + d) \cdot k$

4: **return** S

5: **end procedure**

Algorithm 3 Procedure evaluate population.

Evaluate the population of a given clustering optimization problem.

Input parameters:

k: Number of clusters codified in the individual.

r: Degree of a membership function.

d: Dimensions of the dataset.

X: Set of patterns to cluster.

S: Population of individuals to be evaluated.

1: **procedure** EVALUATEPOPULATION(k, r, d, X, S)
2: $\quad \vec{Q} \leftarrow \{\emptyset\}$ ▷ Fitness vector for the population S.
3: $\quad$ **for all** $\vec{s} \in S$ **do** ▷ For each individual in the population.
4: $\quad\quad \vec{y} \leftarrow$ DECODE$(k, r, d, X, \vec{s})$
5: $\quad\quad \Pi \leftarrow \{X, \vec{y}\}$
6: $\quad\quad Q_i \leftarrow Q(\Pi) +$ PENALIZE$(\vec{y}, k)$
7: $\quad$ **end for**
8:
9: $\quad$ **return** $\vec{Q}$
10: **end procedure**

Algorithm 4 Procedure decode.

Obtains the vector of labels $\vec{y}$ from an individual.

Input parameters:

k: Number of clusters codified in the individual.

r: Degree of a membership function.

d: Dimensions of the dataset.

X: Set of patterns to cluster.

$\vec{s}$: Individual to be decoded.

1: **procedure** DECODE$(k, r, d, X, \vec{s})$
2: $\vec{y} \leftarrow \{y_i | \forall y = -1 \; and \; |\vec{y}| = |X|\}$ ▷ Vector of labels with a default label for all patterns.
3: **for all** $\vec{x} \in X$ **do**
4: **for** $mf \in \vec{s}$ **do** ▷ Where mf is each block of $((r \cdot d) + d)$ coefficients α in $\vec{s}$ representing the k^{th} membership function.
5: $f(\vec{x}) \leftarrow \sum_{a=0}^{r} \sum_{b}^{d} \alpha_{ab} x_b^a$
6: **if** SIGMOID$(f(\vec{x})) \geq 0.5$ **then**
7: $y_i \leftarrow l$ ▷ Assign the label l of mf where $l \in \mathbb{Z}^+$.
8: **break**
9: **end if**
10: **end for**
11: **end for**
12: **return** $\vec{y}$
13: **end procedure**

Algorithm 5 Procedure penalize.

Calculate the penalization to apply based on the satisfied restrictions s in the vector of labels $\vec{y}$. Which are: 1, $\vec{x} \notin C_{null}$ (where $\vec{x} \in C_{null}$ are those $\vec{x}$ with label $y_i = -1$), plus k (there is at least one $\vec{x}$ per cluster C_i).

Input parameters:

$\vec{y}$: Vector of labels, one label per $\vec{x} \in X$.

k: Number of cluster labels that should be in $\vec{y}$, different from a default label.

1: **procedure** PENALIZE($\vec{y}, k$)
2: $K \leftarrow 10^9$ ▷ Penalization constant.
3: $n \leftarrow k + 1$ ▷ Total of constraints.
4: $s \leftarrow 1$ ▷ Total of satisfied constraints.
5: **for** $l \leftarrow 1, |\vec{y}|$ **do**
6: **if** $y_i = -1$ **then**
7: $s \leftarrow s - 1$
8: break
9: **end if**
10: **end for**
11: **for** $l \leftarrow 1, k$ **do**
12: **for** $i \leftarrow 1, |\vec{y}|$ **do**
13: **if** $y_i = l$ **then**
14: $s \leftarrow s + 1$
15: break
16: **end if**
17: **end for**
18: **end for**
19: $penalty \leftarrow K\left(1 - \frac{s}{n}\right)$
20: **return** $penalty$
21: **end procedure**

Bibliography

[1] A., B. (1946). On a measure of divergence between two statistical populations. *Indian J. Stat.*, 7:401–406.

[2] Aggarwal, C. C., Hinneburg, A., and Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*, pages 420–434. Springer.

[3] Aggarwal, C. C. and Reddy, C. K. (2013). *Data clustering: algorithms and applications*. Chapman and Hall/CRC.

[4] Ammar, A., Elouedi, Z., and Lingras, P. (2016). Rough probabilistic meta-clustering of retail datasets. *Fuzzy Sets and Systems*, 286:173–196.

[5] Bäck, T., Fogel, D., and Michalewicz, Z. (1997). Handbook of evolutionary computation. *Release*, 97(1):B1.

[6] Back, T., Hoffmeister, F., and Schwefel, H.-P. (1991). A survey of evolution strategies. In *Proceedings of the fourth international conference on genetic algorithms*, volume 2. Morgan Kaufmann Publishers, San Mateo.

[7] Ball, G. H. and Hall, D. J. (1967). A clustering technique for summarizing multivariate data. *Systems Research and Behavioral Science*, 12(2):153–155.

[8] Bharill, N. and Tiwari, A. (2014). Enhanced cluster validity index for the evaluation of optimal number of clusters for fuzzy c-means algorithm. In *IEEE International Conference on Fuzzy Systems*, pages 1526–1533.

[9] Blunsom, P. (2004). Hidden markov models. *Lecture notes, August*, 15:18–19.

[10] Bobadilla, E. A. and Morales, A. K. (2015). A clustering method based on the maximum entropy principle. *Entropy*, 17:151–180.

[11] Caruana, R., Elhawary, M., Nguyen, N., and Smith, C. (2006). Meta clustering. In *Sixth International Conference on Data Mining*.

[12] Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1.

[13] Chou, C.-H., Su, M.-C., and Lai, E. (2004). A new cluster validity measure and its application to image compression. *Pattern Analysis and Applications*, 7(2):205–220.

[14] Chuang, K.-S., Tzeng, H.-L., Chen, S., Wu, J., and Chen, T.-J. (2006). Fuzzy c-means clustering with spatial information for image segmentation. *computerized medical imaging and graphics*, 30(1):9–15.

[15] da Silva, L. E. B. and Costa, J. A. F. (2014). Clustering of the self-organizing map using particle swarm optimization and validity indices. In *Neural Networks (IJCNN), 2014 International Joint Conference on*, pages 3798–3806. IEEE.

[16] Das, S., Abraham, A., and Konar, A. (2008). Automatic clustering using an improved differential evolution algorithm. *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans*, 38(1):218–237.

[17] Das, S., Abraham, A., and Konar, A. (2009). *Metaheuristic clustering*, volume 178. Springer.

[18] Davies, D. L. and Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, 2:224–227.

[19] Deza, M.-M. and Deza, E. (2006). *Dictionary of distances*. Elsevier.

[20] Digalakis, J. G. and Margaritis, K. G. (2002). An experimental study of benchmarking functions for genetic algorithms. *International Journal of Computer Mathematics*, 79(4):403–416.

[21] Dorigo, M., Birattari, M., and Stutzle, T. (2006). Ant colony optimization. *IEEE computational intelligence magazine*, 1(4):28–39.

[22] Dunn, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104.

[23] Edward, S. (2010). Clustering (xu, r. and wunsch, dc; 2008)[book review]. *IEEE Pulse*, 1:74–76.

[24] Gan, G., Ma, C., and Wu, J. (2007). *Data Clustering Theory, Algorithms and Applications*. SIAM.

[25] Garcia, A. J. and Flores, W. G. (2016). Automatic clustering using nature-inspired metaheuristics: A survey. *Applied Soft Computing*, 41:192–213.

[26] Gen, M. and Cheng, R. (1996). A survey of penalty techniques in genetic algorithms. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 804–809. IEEE.

[27] Glover, F. (1989). Tabu search—part i. *ORSA Journal on computing*, 1(3):190–206.

[28] Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning, 1989. *Reading: Addison-Wesley*.

[29] Goldberg, D. E. (2005). *Genetic Algorithms in search, Optimization and Machine Learning*. Addison Wesley.

[30] Grefenstette, J. J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on systems, man, and cybernetics*, 16(1):122–128.

[31] Guan, Y., Ghorbani, A. A., and Belacel, N. (2003). Y-means: A clustering method for intrusion detection. In *Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on*, volume 2, pages 1083–1086. IEEE.

[32] Han, J. and Kamber, M. (2006). *Data Mining: Concepts and Techniques*. Elsevier.

[33] Haykin, S. S., Haykin, S. S., Haykin, S. S., and Haykin, S. S. (2009). *Neural networks and learning machines*, volume 3. Pearson Upper Saddle River, NJ, USA:.

[34] Hinneburg, A. and Gabriel, H.-H. (2007). Denclue 2.0: Fast clustering based on kernel density estimation. In *International symposium on intelligent data analysis*, pages 70–80. Springer.

[35] Hruschka, E. R., Campello, R. J., Freitas, A. A., et al. (2009). A survey of evolutionary algorithms for clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 39(2):133–155.

[36] Huang, J.-J., Tzeng, G.-H., and Ong, C.-S. (2007). Marketing segmentation using support vector clustering. *Expert systems with applications*, 32(2):313–317.

[37] Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1):193–218.

[38] J., F., T., H., and R., T. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

[39] J., M., L.M., L. C., and Eds., N. J. (1967). Some methods for classification and analysis of multivariate observations. In *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.

[40] Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31:651–666.

[41] J.C., B. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer.

[42] J.C., D. (1973). A fuzzy relative of the isodata process and its use in detecting vompact well-separated clusters. *Journal of Cybernetics*, 3:32–57.

[43] J.C., D. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4:95–104.

[44] Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

[45] Jose-Garcia, A. and Gomez-Flores, W. (2016). Automatic clustering using nature-inspired metaheuristics: A survey. *Applied Soft Computing*, 41:192–213.

[46] Karypis, G., Han, E.-H., and Kumar, V. (1999). Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75.

[47] Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning*, pages 760–766. Springer.

[48] Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69.

[49] Krishna, K. and Murty, M. N. (1999). Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439.

[50] Krovi, R. (1992). Genetic algorithms for clustering: a preliminary investigation. In *System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on*, volume 4, pages 540–544. IEEE.

[51] Kuncheva, L. I. and Bezdek, J. C. (1997). Selection of cluster prototypes from data by a genetic algorithm. In *Proc. 5th European congress on Intelligent techniques and soft computing*, pages 1683–1688.

[52] Kuri-Morales, A. F., Aldana-Bobadilla, E., and López-Peña, I. (2013). The best genetic algorithm ii. In *Mexican International Conference on Artificial Intelligence*, pages 16–29. Springer.

[53] Kuri-Morales, A. F. and Gutiérrez-García, J. (2002). Penalty function methods for constrained optimization with genetic algorithms: A statistical analysis. In *Mexican International Conference on Artificial Intelligence*, pages 108–117. Springer.

[54] L., R. and O., M. (2005). *Clustering methods in Data Mining and Knowledge Discovery Handbook*. Springer.

[55] Lee, K. M., Lee, K. M., and Lee, C. H. (2012). Statistical cluster validity indexes to consider cohesion and separation. In *Fuzzy Theory and it's Applications (iFUZZY), 2012 International Conference on*, pages 228–232. IEEE.

[56] Liu, H., Li, J., and Chapman, M. A. (2003). Automated road extraction from satellite imagery using hybrid genetic algorithms and cluster analysis. *Journal of Environmental Informatics*, pages 40–47.

[57] Liu, X.-y. and Fu, H. (2010). An effective clustering algorithm with ant colony. *Journal of Computers*, 5(4):598–605.

[58] Lobo, F., Lima, C. F., and Michalewicz, Z. (2007). *Parameter setting in evolutionary algorithms*, volume 54. Springer Science & Business Media.

[59] Lourenço, H. R., Martin, O. C., and Stutzle, T. (2003). Iterated local search. *International series in operations research and management science*, pages 321–354.

[60] M., E., H.P., K., J., S., and X., X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

[61] M., H., Y., B., and M., V. (2001). On clustering validation techniques. *Journal of Intelligent Information Systems*, 17:2/3:107–145.

[62] Melanie, M. (1999). *An Introduction to Genetic Algorithms*. Cambridge.

[63] Milligan, G. W. and Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2):159–179.

[64] Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.

[65] Morales, A. K. and Quezada, C. V. (1998). A universal eclectic genetic algorithm for constrained optimization. In *Proceedings of the 6th European congress on intelligent techniques and soft computing*, volume 1, pages 518–522.

[66] Müller, E., Günnemann, S., Assent, I., and Seidl, T. (2009). Evaluating clustering in subspace projections of high dimensional data. *Proceedings of the VLDB Endowment*, 2(1):1270–1281.

[67] Niwattanakul, S., Singthongchai, J., Naenudorn, E., and Wanapu, S. (2013). Using of jaccard coefficient for keywords similarity. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, page 6.

[68] Park, H.-S. and Jun, C.-H. (2009). A simple and fast algorithm for k-medoids clustering. *Expert systems with applications*, 36(2):3336–3341.

[69] P.C., M. (1936). On the generalized distance in statistics. In *Proceedings of the National Institute of Sciences (Calcutta)*, volume 2, pages 49–55.

[70] Ren, Y., Wang, L., and Guan, W. (2015). Approaches to cluster validity index via mahalanobis metric. In *27th Chinese Control and Decision Conference*, pages 6480–6484.

[71] Rennie, B. C. and Dobson, A. J. (1969). On stirling numbers of the second kind. *Journal of Combinatorial Theory*, 7(2):116–121.

[72] S., G., R., R., and K., S. (1998). Cure: An efficient clustering algorithm for large databases. In *ACM SIGMOD International Conference on Management of Data*, pages 73–84.

[73] Saha, S. and Bandyopadhyay, S. (2007). New cluster validity index based on fuzzy granulation-degranulation criterion. In *Advanced Computing and Communications International Conference*, pages 353–358.

[74] Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

[75] Smiti, A. and Eloudi, Z. (2013). Soft dbscan: Improving dbscan clustering method using fuzzy set theory. In *Human System Interaction (HSI), 2013 The 6th International Conference on*, pages 380–385. IEEE.

[76] Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359.

[77] T., Z., R., R., and M., L. (1996). Birch: An efficient data clustering method for very large databases. In *ACM SIGMOD international Conference on Management of Data*, pages 103–114.

[78] Tagawa, K. and Ishimizu, T. (2010). Concurrent differential evolution based on mapreduce. *International Journal of computers*, 4(4):161–168.

[79] Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons.

[80] Tan, P.-N., Steinbach, M., and Kumar, V. (2014). *Introduction to Data Mining*. Pearson.

[81] Theodoridis, S. and Koutroumbas, K. (2008). *Pattern Recognition*. Elsevier.

[82] Tibshirani, R., Walther, G., and Hastie, T. (2001). Estimating the number of clusters in a

data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423.

[83] Tseng, L. Y. and Yang, S. B. (2001). A genetic approach to the automatic clustering problem. *Pattern Recognition*, 34(2):415–424.

[84] Van Laarhoven, P. J. and Aarts, E. H. (1987). Simulated annealing. In *Simulated Annealing: Theory and Applications*, pages 7–15. Springer.

[85] Vasant, P. M. (2012). *Meta-heuristics optimization algorithms in engineering, business, economics, and finance*. IGI Global.

[86] Walpole, R. E., Myers, R. H., Myers, S. L., and Ye, K. (2012). *Probabilidad y estadística para ingeniería y ciencias.* Person.

[87] Wang, C.-W. and Hwang, J.-I. G. (2012). Automatic clustering using particle swarm optimization with various validity indices. In *Biomedical Engineering and Informatics (BMEI), 2012 5th International Conference on*, pages 1557–1561. IEEE.

[88] Wang, L. and Wang, Z.-O. (2003). Cubn: A clustering algorithm based on density and distance. In *Machine Learning and Cybernetics, 2003 International Conference on*, volume 1, pages 108–112. IEEE.

[89] Wittkop, T., Emig, D., Lange, S., Rahmann, S., Albrecht, M., Morris, J. H., Böcker, S., Stoye, J., and Baumbach, J. (2010). Partitioning biological data with transitivity clustering. *Nature methods*, 7(6):419–420.

[90] Yeniay, Ö. (2005). Penalty function methods for constrained optimization with genetic algorithms. *Mathematical and Computational Applications*, 10(1):45–56.

[91] Zeng, Y., Tang, J., Garcia-Frias, J., and Gao, G. R. (2002). An adaptive meta-clustering approach: Combining the information from different clustering results. In *Proceedings of the IEEE Computer Society Conference on Bioinformatics*.