CHAPTER 3. ClusDM (Clustering for Decision Making)

This chapter explains the new multi-criteria decision aid methodology we propose, called *ClusDM*, which stands for *Clus*tering for *D*ecision *M*aking. Its name comes from the use of clustering algorithms to solve the decision-making problem, as it will be explained in this chapter.

This methodology has been designed for dealing with heterogeneous data sets because there is a lack of MCDA tools for this type of problems. One of the key points of this method is that it can deal with different types of variables during all the stages of the decision-making analysis. As it has been explained in the previous chapter, the existing approaches perform a transformation of the original data into a common domain. In our method, we are always dealing directly with the data provided by the experts, in order to avoid the modification of the information available in those data.

Although we will explain our method as a ranking decision tool, it can also be used to solve selection decision problems. In fact, a selection problem can be seen as a subtype of ranking problems in which we are only interested in distinguishing the group of best alternatives.

In this chapter we will explain part of the ClusDM methodology. Before starting this explanation, section 3.1 is devoted to describe the scales we use. Then, in section 3.2 we give an outline of the ClusDM methodology, giving some details of the four stages of the process: Aggregation, Ranking, Explanation and Quality measurement. Section 3.3 is devoted to the explanation in detail of the aggregation stage. The rest of the stages will be explained in the following chapters.

3.1 Considerations on the scales in ClusDM

It has been reviewed in Chapter 2 that the evaluation of alternatives in relation to a given criterion can be done in many different scales. The most common MCDA methods deal with a single common scale. ClusDM is a general methodology that is able to handle heterogeneous criteria. In our design and implementation of the methodology, we have considered the following ones:

- quantitative or numerical scale
- ordered qualitative or ordinal scale (i.e preference values)
- non-ordered qualitative scale (i.e. nominal or categorical values)
- Boolean scale (i.e. binary values)

Although we have restricted ourselves to these types of values, we would like to note that any other type of value that has a distance function defined in its domain could also be used.

To operate on the values of these scales, in particular to compute similarities between pairs of values, some assumptions are needed on the semantics of the values. In the case of quantitative, categorical and Boolean scales, the definition of distances or similarities has been widely studied (we will review some possibilities in section 3.3.2). For the case of ordered qualitative values, we can find in the literature several approaches to the definition of the underlying semantics of the scale, which is the basis for the similarity and aggregation operations [Torra,2001].

Explicit semantics: A mapping exists that translates each linguistic term in a numerical or fuzzy value. Operations on the linguistic values are defined on terms of the corresponding operations in the numerical or fuzzy scale.

Implicit semantics: Operations are defined assuming an implicit mapping function from the original scale into a numerical one. The typical case is to replace each term by its position in its domain.

Operations restricted on the ordinal scale: New operations in a given scale are only defined in terms of operations axiomatically defined in that scale. Allowed operations are maximum, minimum, t-norm, t-conorm and operations defined from them.

Working on any of these settings present advantages and disadvantages:

- In the case of explicit semantics, operations are well defined and sound. However, the experts are required to supply additional information, in particular, they must provide a mapping for each scale.
- Implicit semantics provide easy to use operations but, instead, semantics is coded - and fixed- in the operators. Counterintuitive results can be obtained if the application does not follow the assumptions considered.

- Operators restricted on the ordinal scale also lead to sound results. Nevertheless, some of the basic operations are difficult to be defined by non-experienced users, as their meaning is sometimes difficult to grasp. This is the case of defining ordinal t-norms and t-conorms.

ClusDM uses a negation-based semantics. This can be seen as an alternative to the explicit semantics approach as it builds an explicit mapping from the set of linguistic terms into the unit interval. This mapping is inferred from a negation on the set of terms. This approach avoids the use of operators with coded semantics. Now the user is only required to supply a negation function instead of a complete explicit mapping from terms to numbers. This approach is easier for the experts because the negation of a term can be interpreted as its antonym, following [de Soto&Trillas,1999].

In the rest of this section we describe the negation functions we consider and how the semantics is inferred.

Negation based semantics for linguistic terms

Negation is a well-known operation in multi-valued logics that is defined over a set of ordered linguistic labels (i.e. terms) T=$\{t_0, ..., t_n\}$ (with $t_0<...<t_n$). It is axiomatically defined as a function from T to T, that satisfies the following conditions:

N1) if $t_i < t_j$ then $N(t_i) > N(t_j)$ for all t_i, t_j in T

N2) $N(N(t_i)) = t_i$ for all t_i in T

In fact, when these conditions hold, the set of ordered linguistic terms T completely determines the negation function. This is so because for each set of ordered linguistic terms T=$\{t_0, ..., t_n\}$ there exists only one negation function that satisfies N1 and N2 [Agustí et al.,1991]. This negation function is defined by:

$$N(t_i) = t_{n-i} \quad \text{for all } t_i \text{ in T}$$

According to this last result, when conditions N1 and N2 are required, the negation function assumes vocabularies where each term in the pair $< t_i, t_{n-i} >$ is equally informative. Although in decision making, equal informativeness is sometimes not adequate, it is not always possible for the expert to define an interval or a fuzzy set for each term because that would require a degree of accuracy that the expert cannot always supply. To allow non-equal informativeness without requiring experts to supply detailed information on the semantics of the terms, [Torra,1996] introduced a new class of negation functions over linguistic terms. With this approach an expert can provide additional information about the meaning of the terms in a more natural way. These new negation functions are defined from T to $\wp$(T) (i.e., parts of T) weakening conditions N1 and N2.

Definition 1. [Torra,1996] A function *Neg* from T to $\wp$(T) is a negation function if it satisfies:

C0) *Neg* is not empty and convex
C1) if $t_i < t_j$ then $Neg(t_i) \geq Neg(t_j)$ for all $t_i, t_j \in$ T
C2) if $t_i \in Neg(t_j)$ then $t_j \in Neg(t_i)$

In this definition C1 and C2 are generalisations, respectively, of N1 and N2. In fact C2 is a generalisation of N3 (given below) that is equivalent to N2.

N3) if $t_i = Neg(t_j)$ then $t_j = Neg(t_i)$

C0 is a technical condition. It means that for all t_i in T, $Neg(t_i)$ is not empty ($Neg(t_i) \neq \emptyset$) and convex (a subset X of T is convex if and only if for all t_x, t_y, t_z in T such that $t_x < t_y < t_z$ and t_x, $t_z \in$ X then $t_y \in$ X). In other words, C0 establishes that $Neg(t_i)$ is a non-empty interval of terms in T.

Now, let us turn into the semantics. For a vocabulary T, the semantics of a term is understood as a subset of the unit interval. Let $I(t_i)$ be the subset attached to term t_i; in this case the set $P = \{I(t_0),..., I(t_n)\}$ corresponds to the semantics of all terms in T. It is assumed that the sets recover the unit interval and that the intersection of any two sets is empty or punctual (if they are contiguous). That is, $\cup_{I \in P} I = [0,1]$ and $I(t_i) \cap I(t_j) = \emptyset$.

However, not all partitions in the unit interval are adequate as semantics for a set of linguistic labels. In fact, the relations among labels that a negation function establishes should also be true in the intervals in *P* when the negation in the unit interval is considered. In particular, the consistency of *P* in relation to the most usual negation function N(x)=1-x was mathematically defined. Informally, when consistency is required, the following two conditions hold: (i) the negation of all the elements of the interval attached to t_i belongs to the intervals attached to the negation of t_i; (ii) if $Neg(t_i) = \{t_{i0},...,t_{ik}\}$, then neither the term t_{i0} nor the term t_{ik} are "superfluous" in relation to the negation function. This latter condition means that there exists at least one element of the interval attached to t_i such that its negation belongs to $I(t_0)$ (respectively to $I(t_{ik})$). Given a negation function, there are several consistent semantics. In particular, the following one (which is the one we are going to use) is consistent with N(x)=1-x:

Definition 2. [Torra,1996] Let *Neg* be a negation function from T to $\wp$(T), according to Definition 1; we define P_{Neg} as the set $P_{Neg} = \{[m_0, M_0],...., [m_n, M_n]\}$ where

$$I(t_i) = [m_i, M_i] = \left[\frac{\sum_{t<t_i} |Neg(t)|}{\sum_{t \in T} |Neg(t)|}, \frac{\sum_{t \leq t_i} |Neg(t)|}{\sum_{t \in T} |Neg(t)|} \right] \quad \text{Eq. 3.1}$$

where |X| stands for the cardinality of the set X.

It is important to note that that the classical semantics is obtained when the negation function is restricted to satisfy $|Neg(t_i)| = 1$. In that case, $I(t_i) = [i/(n+1), (i+1)/(n+1)]$, which corresponds to having all the intervals with the same measure (i.e., the same precision). According to that, this approach extends the classical negation functions for multi-valued logics and relates them with the usual implicit semantics (note that the central point of the interval $I(t_i)$, $(i+1/2)/(n+1)$, is proportional to the position of the term t_i normalized in [0,1]: i/n.

3.2 The ClusDM methodology

In this section we will introduce a methodology for multi-criteria decision aid, which follows the utility-based model. As it has been explained in section 2.2, these multi-criteria decision methods distinguish two different stages: (1) the aggregation of alternatives and (2) their ranking. Our methodology follows the same strategy but we have included two additional stages: (3) an explanation stage to give semantics to the ranking obtained, and (4) an evaluation stage to measure the quality of the result. With these new stages we want ClusDM to be a useful decision aid more than a simple decision making procedure. That is, our goal is to give recommendations to the user rather than make an automatic decision.

Therefore, the ClusDM methodology distinguishes the following steps:

> STAGE 1. *Aggregation or Rating Phase*: The values of each alternative are analysed in order to find another evaluation for the alternative that allows us to compare it with the others and decide which one is the best.
>
> STAGE 2. *Ranking Phase*: The alternatives are compared and ranked on the basis of the value given in the aggregation phase.
>
> STAGE 3. *Explanation Phase*: In addition to the list of ordered alternatives, a qualitative term is attached to each alternative, in order to give some semantics to their relative position in the ranking in comparison to the positions of the ideal and nadir alternatives. So, the alternatives near the ideal will be denoted as "*optimum*" or "*very_good*" ones, the ones near

the nadir will be the "*very_bad*" options. The others will receive a term according to their values.

STAGE 4. *Quality Measurement Phase*: some quality measures are given, which can be useful for the decision maker in order to decide the reliability of the ranking.

In Figure 2 we can see a schema of the flow of data. We begin with a data matrix with *m* alternatives and *p* criteria. At the end, we have a qualified set of alternatives (each alternative has a linguistic term t_i that describes the appropriateness to be selected as a solution for the decision problem) and a report with additional information.

During the analysis of the decision matrix, the method extracts useful information for the decision-maker. All the details about this data and the way it is obtained will be included to this final report. The ClusDM methodology has been designed having in mind that the user will be reluctant to make a machine-based decision. He needs some guarantee of the quality of the ranking given by the system. ClusDM pretends to be a useful aid for decision makers supplying them all the useful knowledge that can be extracted form the data during the aggregation, ranking and explanation stages.

As it has been said in the introduction of this chapter, section 3.3 reviews the aggregation stage. The ranking phase is described in chapter 4 and the last two ones are explained in chapter 5.

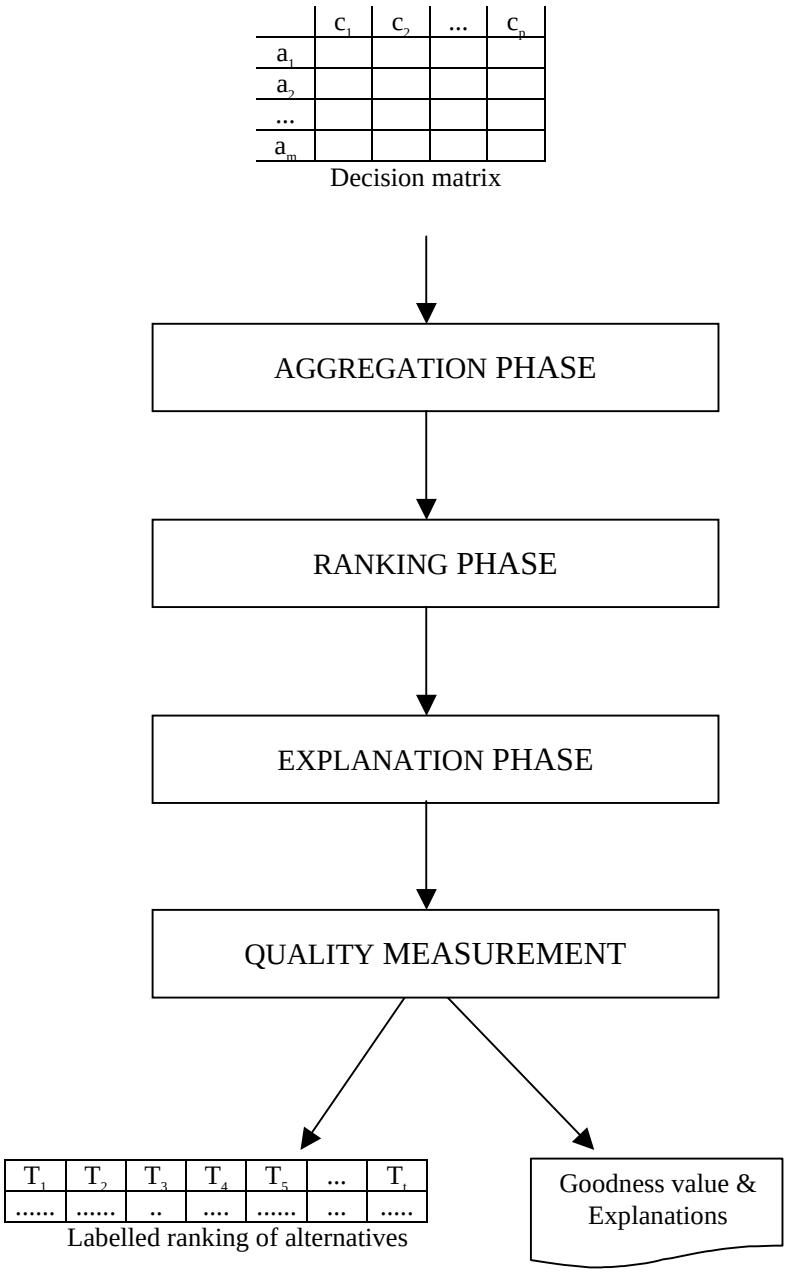

Figure 2. Stages of the ClusDM process

3.3 Aggregation

The first stage of the multicriteria decision process consists of aggregating the different values given to each alternative, and obtain a new one that synthesises the information provided by the individual criteria. When working with homogeneous values, the result of the aggregation stage is a new value of the same nature than the original ones. For example, the Weighted Average operator is usually applied to a set of numerical values, producing a new numerical value. However, when the criteria are heterogeneous, it is not obvious which should be the type of values of the result. This is so because not all the scales can give the same accuracy when describing the alternatives.

We have implemented a system, called Radames, which allows the aggregation of many different data representation structures (e.g. data matrices, trees, vectors). The case studied in this thesis concerns the aggregation of vectors describing an alternative. In particular, we work with a data matrix whose rows are vectors with qualitative or heterogeneous values. For the rest of cases (e.g. numerical or Boolean data), the most well-known aggregation operators have been studied and implemented [Valls, 1997].

For qualitative or heterogeneous value we propose the use of the ClusDM methodology to obtain a new qualitative criterion. That is, ClusDM can be seen as a MCDA methodology or as an aggregation or fusion operator.

In ClusDM, the result of the first stage is a qualitative non-ordered vocabulary, although after the ranking and explanation stages it will become an ordered preference qualitative criterion. The selection of a qualitative preference scale is based on the comparison of the different scales we are considering: numerical, qualitative (preferences or categories) and Booleans. The most informative type is the numerical one, and the least informative is the Boolean one. Qualitative values are in the middle, the greater the cardinality of their domains; the more differences can be stressed. In fact, sometimes Boolean can be considered as a qualitative variable with two values in the domain.

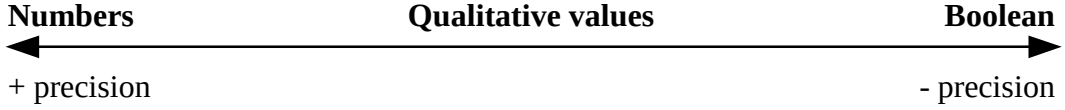

Figure 3. Precision of the different types of values

The transformation of one scale into another has two different effects (see Figure 3). On one hand, the translation of numbers into terms (or Booleans) implies a reduction of information because different numbers will be transformed into the same term. On the other hand, transforming qualitative values into numerical ones implies substituting a term by a number. The subsequent operations with this number will treat it as a precise value, which is introducing error because the number is only an interpretation of a term that is actually covering an interval of values.

Considering that changes from one type of representation to another produces a loss of some kind of information, we decided to take a position in the middle. Thus, the result of ClusDM will be a qualitative term describing each alternative.

After studying qualitative domains, we have seen that the linguistic terms of a vocabulary define a partition on the set of alternatives, because the alternatives that take the same value are indistinguishable, according to the expert. Therefore, we can formulate our aggregation goal as: to obtain a new partition of the set of alternatives having into account all the information provided by the criteria (i.e. experts). Each cluster in this partition will correspond to a new linguistic value in the domain of the new social (i.e. agreed) criterion [Valls, 2000a].

To obtain a partition (i.e. a non-overlapping set of clusters) we can use clustering methods. During the clustering process the objects form groups according to their similarity, which is measured comparing the values of the alternatives for the different criteria. To find these groups or clusters, each object is compared to the others.

We have studied the application of clustering to qualitative and heterogeneous data sets. In the next section, there is a brief overview of clustering techniques, making special emphasis on the ones that are more appropriate to be used as an aggregation operator. Section 3.3.2 explains how to obtain the aggregation of the alternatives in the decision matrix by means of a clustering tool called *Sedàs*.

Although we will concentrate on our clustering system *Sedàs*, any other clustering technique could be applied. In any case, it is important to note that this aggregation method does not hold the condition of irrelevant alternatives[4] [Arrow,1963], because (using clustering) it is not possible to obtain the consensus value of an alternative without taking into account the rest.

3.3.1 Review of Clustering methods

Clustering methods are traditional techniques to obtain a partition of a set of objects [Everitt,1977], [Jain&Dubes,1988]. A clustering process has two phases (Figure 4):

[4] This condition is usually satisfied by the aggregation methods in MAUT.

(a) The construction of a *similarity* matrix that contains the pairwise measures of proximity between the alternatives. Several similarity or dissimilarity functions can be used. Each one has different properties, and it is not possible to determine which is the best for a particular set of data. In [Anderberg,1973] and [Baulieu,1989] there is a review of some of these measures and their interrelationships.

(b) The construction of a set of clusters, in which similar objects belong to the same cluster. Many different methods have been developed [Jain&Dubes,1988]. Up to now, it is impossible to define a way to choose neither the best method, nor the best for a particular problem. These methods can are divided into two families:

- *Hierarchical Agglomerative clustering methods*: clusters are embedded forming a tree. The root is the most general cluster, which contains all the objects (i.e. alternatives), and the leaves are the most specific groups, that contain a unique alternative.

- *Partitioning clustering methods*: clusters are mutually exclusive. They are generated optimising a 'clustering criterion'.

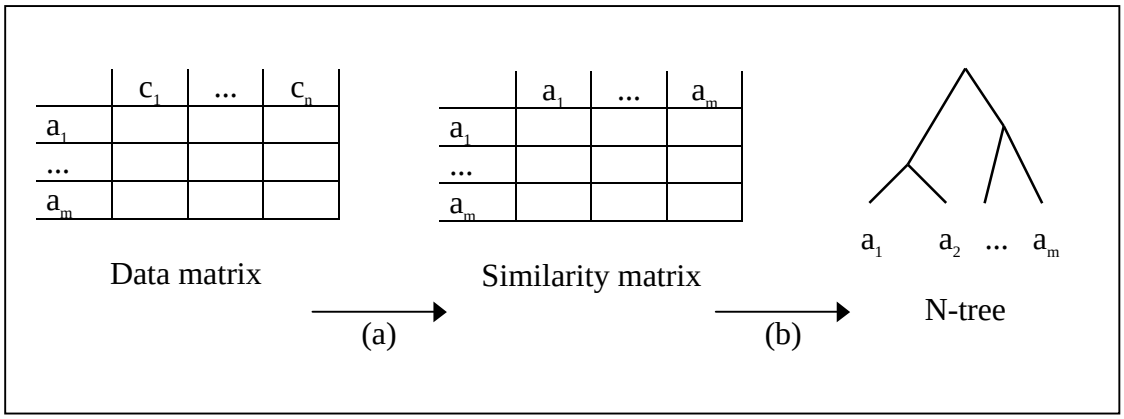

Figure 4. Clustering process

We will follow the hierarchical agglomerative apoproach. That is, once the similarity relation is defined for each pair of alternatives in the data matrix, the clustering will proceed to build a tree. A tree is a nested sequence of partitions over the set of alternatives. Formally,

Definition 3. [Gordon, 1987] A tree over a set of alternatives A is defined as a set τ of subsets of A that satisfies the following conditions:

1. $A \in \tau$
2. $\varnothing \notin \tau$
3. $\{a_i\} \in \tau$ for all $a_i \in A$
4. $M \cap N \in \{\varnothing, M, N\}$ for all $M, N \in \tau$

With this conditions we can have binary or n-ary trees, although usually clustering trees are forced to be binary (each node has only two children). The use of binary trees is justified in terms of the facility with which these structures are obtained and treated. However, binary trees are not as much close to the knowledge they represent as n-trees.

The clustering process, besides of returning the set of nodes of the clustering tree, assigns to each node a cohesion value, h_α, of the cluster it represents. This value corresponds to a measure of similarity of the last union (i.e. when all the subclusters have been gathered to form the cluster that the node represents). Therefore, for any pair of alternatives (a_i, a_j) that belongs to the cluster α, the following condition is fulfilled: $d(a_i, a_j) \leq h_\alpha$, where d is the dissimilarity function (i.e. the opposite of the similarity) used to compare the alternatives during the clustering process.

As it will be seen in the next section, we have focused on the study of a particular subset of clustering methods known as SAHN [Sneath&Sokal,1973]: Sequential, Agglomerative, Hierarchical and Non-overlapping methods. The clustering algorithm for these methods can be summarised as follows:

```
STEP 0. Construction of the initial similarity matrix

STEP 1. Selection of the alternatives (i.e. objects) that are
        more similar. Those alternatives will form the new
        cluster

STEP 2. Modification of the similarity matrix as follows:
     2.1.  Elimination of the alternatives that belong to the new
           cluster
     2.2.  Insertion of the new cluster in the similarity matrix
     2.3.  Calculation of the similarity between the new cluster
           and the rest of objects (using the clustering
           criterion)

STEP 3. Repeat steps 1-2 until we have a single cluster
```

At step 1, the method can gather only two objects (in this way we build a binary tree) or gather all those alternatives with maximum similarity (so we obtain a n-tree). With respect to the clustering criterion that appears in step 2.3, it is used to recalculate the similarity matrix when a new cluster has been created. There are different approaches, such as the Single Linkage, the Ward's method, the Centroid Clustering analysis, etc. (see [Everitt,1977] for more details). Some of them will be reviewed in the next section.

As it has been said, the result of the clustering process is a tree. Trees are generally pictured using dendrograms (see Figure 5). A monotonic dendrogram is the graphical representation of an ultrametric (i.e. cophenetic) matrix. More formally, a dendrogram is defined as a rooted terminally-labeled weighted tree in which all terminal nodes are equally distant from the root [Lapointe&Legendre,

1991]. The weights of this tree are given by the heights h_α, which correspond to the cohesion values of the clusters α. So, for a tree τ with M,N∈τ (two internal nodes), the following property is fulfilled: if M∩N≠∅, $h_M \leq h_N \leftrightarrow M \subset N$.

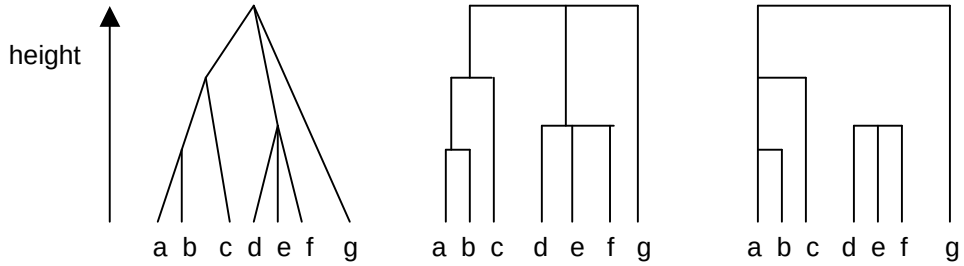

Figure 5. Three different formats for representing dendrograms

Alternative characterisations of a dendrogram can be found in the literature. Gordon [Gordon, 1987] states that a necessary and sufficient condition for a monotonic dendrogram is that the set h_{ij} satisfies the ultrametric condition:

$$h_{ij} \leq \max(h_{ik}, h_{jk}) \text{ for all } a_i, a_j, a_k \in A$$

where h_{ij} is the height of the internal smallest cluster to which both alternatives a_i and a_j belong.

Nevertheless, some of the trees generated by the clustering criteria do not fulfil this ultrametric condition. So, they are not monotonous. They are said to present *inversions* or *reversals*. For example, in
Figure 6 we can see that clusters α=(g,h) and β=(i,j) merge at a level lower that the level at which α was created.

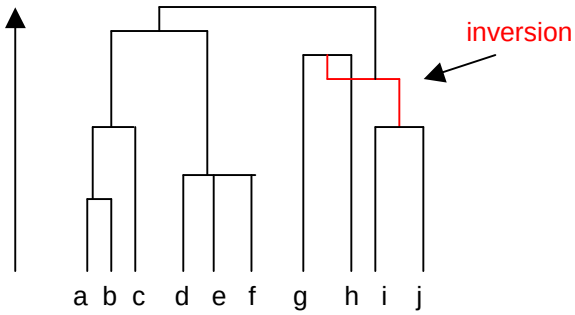

Figure 6. Dendrogram of a non-monotonous tree

Non-monotonous trees may cause problems when the tree is cut in order to obtain a partition of the set of alternatives.

Partitions are obtained making a horizontal cut of the tree at a particular height. The height at which the tree is cut determines the abstraction level achieved. Increasing the cut level we obtain a smaller number of bigger (more general) clusters.

3.3.2 Our generic clustering system: *Sedàs*

We have implemented a generic SAHN clustering system, called *Sedàs* [Valls et. al., 1997]. All the scales mentioned in section 3.1 are allowed in *Sedàs*: numerical, ordered qualitative preferences, categorical and Boolean. However, any other scale with a subtraction function defined in its domain can be included in the system. *Sedàs* has been incorporated to the *Radames* system, in order to be used as an aggregation operator.

The interface allows the user to choose from a list of similarity functions and a list of clustering techniques the most adequate to each particular data set. The system includes, among others, the following classic weighted dissimilarity functions. Being v_{ij} the value of the i-th criterion of alternative a_j, and v_{ik} the value of the i-th criterion of alternative a_k, we can calculate the dissimilarity $d(a_j,a_k)$ using:

- Distance based on Differences

$$\frac{\sum_{i=1}^{p}\left(v_{ij}-v_{ik}\right)}{p} \qquad \text{Eq. 3.2}$$

- Manhattan Distance

$$\sum_{i=1}^{p}\left|v_{ij}-v_{ik}\right| \qquad \text{Eq. 3.3}$$

- Mean Character Difference (M.C.D.)

$$\frac{\sum_{i=1}^{p}\left|v_{ij}-v_{ik}\right|}{p} \qquad \text{Eq. 3.4}$$

- Taxonomic or Euclidean Distance

$$\sqrt{\frac{\sum_{i=1}^{p}\left(v_{ij}-v_{ik}\right)^2}{p}} \qquad \text{Eq. 3.5}$$

- Minkowski Distance

$$\sqrt[r]{\sum_{i=1}^{p}\left|v_{ij}-v_{ik}\right|^r} \qquad \text{Eq. 3.6}$$

- Pearson Correlation Coefficient

$$\frac{\sum_{i=1}^{p}\left(v_{ij}-\overline{v_i}\right)\left(v_{ik}-\overline{v_k}\right)}{\sqrt{\sum_{i=1}^{p}\left(v_{ij}-\overline{v_i}\right)^2\cdot\sum_{i=1}^{p}\left(v_{ik}-\overline{v_k}\right)^2}} \qquad \text{Eq. 3.7}$$

This dissimilarity functions have been generalised to be applied to numerical, ordered qualitative, categorical and Boolean data [Valls et. al., 1997]. For a numerical criterion with range [a,b], we put the values into the unit interval [0,1] before applying the dissimilarity function. Ordered qualitative values are translated into numbers in [0,1] using their negation-based semantics. The difference v_{ij} - v_{ik} for categorical values takes only two possible values: 0 if they are different or 1 if they are equal. Finally, the Boolean values are treated as categorical ones. This functions can also be adapted to consider different weights for the different variables (i.e. criteria, attributes) [Gibert&Cortés, 1997].

If the decision matrix has missing values (that is there are some unknown values), the system is able to calculate the similarity among the pairs of objects. If v_{ij} or v_{ik} are unknown, *Sedàs* can operate in two modes: a) the rest of values of this criterion are used to calculate the average value, which is used instead of the unknown value; b) this criterion is ignored in the comparison of the two alternatives, a_j and a_k, so p is decreased in 1 unit because we are dealing with less criteria.

Using the data in the similarity matrix, *Sedàs* executes the algorithm explained in the previous section. In step 2.3, a clustering criterion is needed to compare the new-created cluster with the rest of elements of the similarity matrix. To determine the similarity of this new element with respect to the others, many methods have been defined. Some of the most known approaches are available in our system, such as:

- Single Linkage or Nearest Neighbour

This criterion considers that the dissimilarity value between a new cluster α and an object[5] o_k is equal to the minimum distance between the objects in the cluster and the object outside o_k.

$$d(\alpha, o_k) = \min_{o_i \in \alpha} d(o_i, o_k)$$

Graphically,

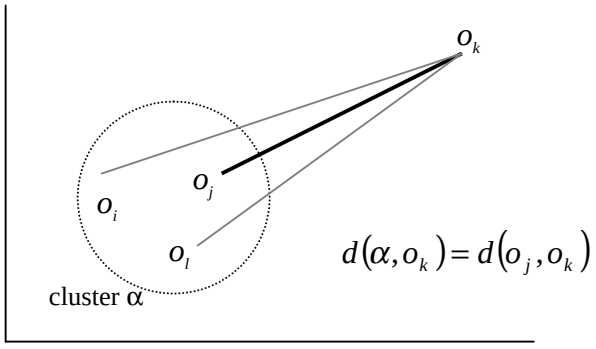

Figure 7. Single Linkage

- Complete Linkage or Furthest Neighbour

This criterion assumes a similar behaviour than the Single Linkage, however, it considers that the dissimilarity value between a new cluster α and an object o_k is equal to the maximum distance between the objects in the new cluster and the object outside it, o_k.

$$d(\alpha, o_k) = \max_{o_i \in \alpha} d(o_i, o_k)$$

Graphically,

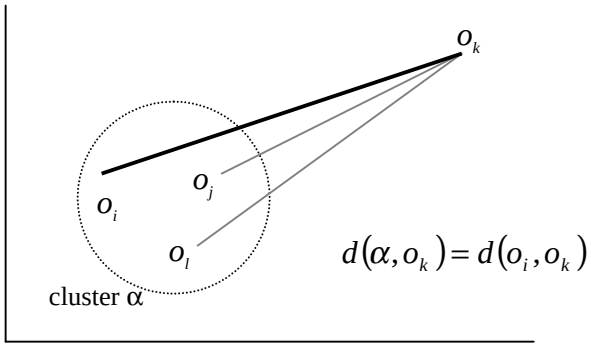

Figure 8. Complete Linkage

[5] An object can be a single alternative or a cluster generated in a previous step.

- Arithmetic Average

A measure in between of the two previous ones is the one known as Arithmetic Average criterion. It takes as a dissimilarity value between a new cluster α and an object o_k, the average distance between the objects in the cluster and the object outside o_k.

$$d(\alpha, o_k) = \frac{\sum_{o_i \in \alpha} d(o_i, o_k)}{|\alpha|}$$

- Centroid Clustering

This approach is based on the calculation of the prototype of each cluster. Let us denote as o_α the prototype of the cluster α. This prototype or centroid is defined as follows: $o_\alpha = (\overline{c_1}, \overline{c_2}, ..., \overline{c_p})$, where $\overline{c_i}$ is the average value of the criterion c_i considering the alternatives that belong to α. Using this prototype or centroid, the distance between the new cluster and an outside object, o_k, is defined.

$$d(\alpha, o_k) = d(o_\alpha, o_k)$$

This averaging function needed to calculate the prototype of the cluster depends on the type of scale. In Table 4 we can see some examples of averaging functions for the scales we are dealing with:

Scale	**Functions**
Numerical	Arithmetic average, Weighted Arithmetic average, OWA
Categorical	Max-min, Voting Techniques, Averages (translating terms into numbers)
Boolean	Voting

Table 4. Some averaging functions to build prototypes

In the case of qualitative domains with a negation function, we propose the translation of the values into numbers and the application of a numerical averaging operator. We recommend the use of the Weighted Arithmetic average or the OWA operator, depending on the kind of weights we are interested to apply.

- Median Cluster Analysis

This criterion established that the dissimilarity between a cluster α (formed by the union of objects o_i and o_j) and the object o_k (which does not belong to α) is the length of the bisectrix of the angle corresponding to o_k, considering a triangle formed by these three objects. This is illustrated in Figure 9.

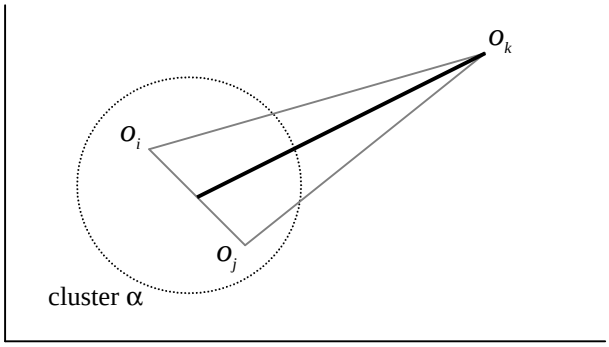

Figure 9. Median Cluster Analysis

$$d\left(\left[o_i, o_j\right], o_k\right) = \sqrt{\frac{1}{2}d^2\left(o_i, o_k\right) + \frac{1}{2}d^2\left(o_j, o_k\right) - \frac{1}{2}d^2\left(o_i, o_j\right)}$$

For n-trees, this definition can be generalized as follows:

$$d\left(\alpha, o_k\right) = \sqrt{\frac{1}{2}\max_{o_i \in \alpha} d^2\left(o_i, o_k\right) + \frac{1}{2}\min_{o_i \in \alpha} d^2\left(o_i, o_k\right) - \frac{1}{2}d^2\left(a, b\right)}$$

$$\text{where } a = \max_{o_i \in \alpha} d^2\left(o_i, o_k\right) \text{ and } b = \min_{o_i \in \alpha} d^2\left(o_i, o_k\right)$$

That is, we build a triangle using two of the objects that belong to the cluster: the one that is nearest to o_k and the one that is furthest with respect to o_k.

Using these clustering criteria, *Sedàs* is able to generate n-ary trees. We decided to discard the binary approach in order to avoid the arbitrary choice of two elements to be joined when there are several with the same similarity. Moreover, with this method we eliminate the chaining of clusters that have the same distance between them.

Not all these clustering criteria produce monotonous trees. In particular, the Centroid and the Median Cluster Analysis methods may generate trees with inversions. So, when *Sedàs* generates a partition P from the tree, it checks that the clusters in the partition are mutually exclusive, that is, $M \cap N = \varnothing$ for all $M, N \in P$.

For instance, the partition induced in Figure 10, P={(a,b,c),(d,e,f),(g),(h),(g,h,i,j),(i,j)}, is not correct because does not hold this condition.

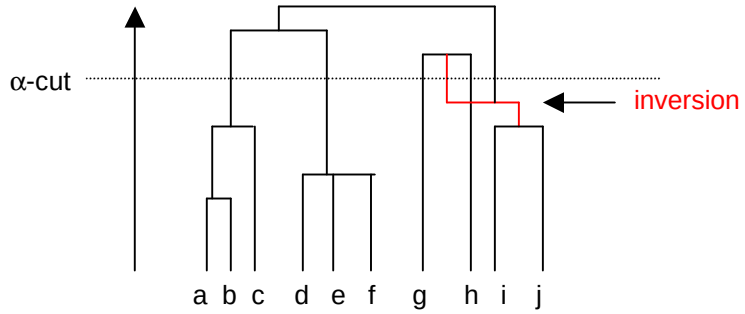

Figure 10. Making a cut in a tree with inversions

3.3.3 Using *Sedàs* as an aggregation operator

We have studied and compared the trees obtained using different similarity functions and clustering criteria [Valls et. al., 1997]. The main conclusion reached is that clustering criterion has less influence on the structure of the tree generated than the similarity function. In Table 5 we can see the comparison of different trees obtained from the same data matrix with several clustering criteria and similarity functions. The table give the distance between pairs of trees. We have used the distance defined in [Newmann,1986] and [Barthélémy&McMorris,1986]:

$$d_\tau(\tau,\tau') = |\tau \cup \tau'| - |\tau \cap \tau'|$$

Looking at the distances between trees, we can see that the distance is highly related to the similarity function used rather than to the classification method. This is reflected by means of small distances between trees obtained with the same similarity function (Differences or Mean Character Difference), and greater distances when different similarity functions are considered. We can see, for example, that when we choose the similarity function Differences (Dif), the trees obtained by means of the Arithmetic Average (Dif_a) and the Median procedure (Dif_m) have a distance of 8. On the other hand, when Arithmetic Average is considered with several similarity functions we have d_τ(Dif_a, MCD_a)=13. Notice that the distances in the upper right frame are greater than the others in the same column/row.

	Dif_a	Dif_m	Dif_s	MCD_a	MCD_m	MDC_s
Dif_a	0	8	9	13	17	13
Dif_m		0	13	17	19	17
Dif_s			0	12	16	12
MCD_a				0	6	0
MCD_m					0	6
MCD_s						0

Symbols glossary
Dif: Differences Distance
MCD: Mean Character Difference
a: Arithmetic average
m: Median procedure
s: Single Linkage

Table 5. Distances between trees

Assuming that the selection of the clustering criterion does not causes great differences in the structure of a tree if the similarity between the elements is well established, we recommend the use of the ***Centroid Clustering*** criterion for aggregating the values of the alternatives. The rationale for this decision is that this method is based on the concept of prototype. The prototype is the pattern of the cluster, and it is used to determine the relation from one cluster to the other clusters and objects analysed. As it will be seen in the next chapters, the following stages of the ClusDM methodology are also based on the prototype of the clusters in the partition obtained after the cutting of the tree. For this reason, we consider that it is appropriate that the aggregation stage also works with prototypes.

After fixing the clustering criterion to the use of the Centroid Clustering, we studied the most usual similarity functions:

- the Differences distance may compensate a negative difference in one criterion with a positive difference in another one. This is an important drawback since two different objects can be considered as equal if the differences compensate each other;
- the Manhattan distance is based on a city made of blocks, so the distance between two opposite corners of a building is the length of the two streets you have to walk to arrive to the other side;
- the Taxonomic distance considers that if you have to cross a square from one corner to the opposite one, you can walk through the square. So, the distance between these two opposite corners is the length of the line that crosses them;
- the Minkowski distance is a generalization of the Taxonomic distance that considers more than 2 criteria, but the properties are the same;
- the Pearson Correlation Coefficient is based on the lineal relations between alternatives. It measures the correlation between two alternatives comparing their values to the average for each criterion. Some dimensional properties on the data set are required for applying this distance [Sneath&Sokal, 1973].

Having into account that the goal of our methodology is to be able to deal with heterogeneous data sets. As it has been said in chapter 1, it is particularly interesting the case of having qualitative preference criteria with different

vocabularies. For this reason, we recommend the use of the Manhattan distance. The basis of this similarity function is more appropriate to the characteristics of a qualitative domain because when we compare two linguistic terms, we will use a numerical translation of this terms, however, the number represents an interval (like one face of a building block) instead of a single point (like the corner of a square).

Although in this point of the explanation we are suggesting the use of the Manhattan distance together with the Clustering criterion, we must remember that the ClusDM methodology is more general, and these are only some parameters that can be changed.

In our system, *Sedàs*, these parameters are required to build the n-tree. As it has been said, to obtain a partition this tree is cut at an appropriate level. In our case, this level is determined by the number of clusters we want to obtain. Remember, that each of these clusters must receive a different term in the vocabulary of the new preference criterion. So, the number of clusters is proportional to the length of the vocabulary. In general, 7 it is said to be the ideal number of terms that a person is able to handle [Miller,1956], however, this number might not be adequate in some cases.

We propose to use the lengths of the vocabularies of the criteria provided by the experts to have an idea of the number of clusters we are looking for. Using this criterion, *Sedàs* takes a number of clusters as close as possible to the number of linguistic terms used in the criteria. If there is no qualitative criterion, then a good approximation is to take $max(1,log_2 d)$, where d is the number of different values considering all alternatives. This value is based on the proposal of [Dougherty et. al.,1995]: they define the best number as the maximum of 1 and $2*log_{10} d$. However, this approximation gives a number of labels too small, which implies losing too much information. After making different tests, we recommend the use of the logarithm base 2.

Despite of being interested in a partition, it is also useful to know the complete tree of clusters, which is giving us the relation among the alternatives at different levels. Looking into the subclusters of a particular cluster we can obtain a more precise clustering of the alternatives, which allows us to distinguish different categories inside a cluster. On the other hand, if we look at higher clusters in the tree, we can see the similarities among the clusters of our partition.

Finally, once the alternatives have been aggregated in clusters, *Sedàs* automatically assigns a symbolic name to each cluster. This partition and the prototype of each of its clusters are the inputs of the following stage: Ranking.

CHAPTER 4. Ranking stage

The ranking of the alternatives is applied after the aggregation of the values in the decision matrix. In general, the ranking procedure depends on the type of result provided by the previous stage. In our case, the aggregation produces a set of clusters and each cluster can be represented by a prototype alternative, which is built according to the values of the alternatives that belong to the cluster, as it has been explained in chapter 3.

Therefore, the goal of this stage is to determine automatically the preference among the clusters, that is, their ranking. In this way, at the end of the process, the class at the first position of the ranking will contain the most preferred alternatives (according to the new overall criterion). To obtain these preferences on the clusters, their prototypes will be used.

The study of different ranking techniques have brought us to distinguish two different situations:

CASE A: All the criteria in the decision matrix are expressing preferences over the alternatives. That is, each criterion is giving an order of the alternatives according to some preference opinion or property.

CASE B: The criteria are expressing different views of the data, which can be preferences or just descriptive properties (e.g. educational degree, job, and age).

The first case is the one that is usually studied in MCDA research [Vincke,1992]. Nevertheless, sometimes there are descriptive properties that should also be taken into account in the decision making process.

In the following sections we will explain the ranking methodology used in the two different cases. A formal definition of the method is done at the beginning of the section, to continue with the explanation of how to apply each method to the ranking of clusters.

4.1 Ranking using Principal Components Analysis

The ranking in CASE A is done using the multivariate statistical method called *Principal Components Analysis* (PCA). To obtain a good ranking with PCA, criteria are required to be correlated with each other. This situation happens when the criteria are the opinions of different experts about the alternatives. Although the experts may have different points of view, if it is possible to define "the best ranking" for the set of alternatives, and experts really know the decision problem, there is supposed to be a high degree of correlation.

The method of *Principal Components* [Pearson, 1901] obtains linear transformations of a set of correlated variables such that the new variables are not correlated. This is a useful technique for statistical analysis of multivariate data, in particular, to describe the multivariate structure of the data.

Although the Principal Components Analysis is usually a descriptive tool, it can be also used for other purposes. For example, PCA can be applied to obtain a ranking of observations [Zhu, 1998].

In this section, we will explain in detail the mathematical basis of a Principal Components Analysis. We will see some properties that are interesting for its use as a ranking tool. Furthermore, we will define some measures and procedures to interpret the results. Finally, we will detail how PCA must be applied to the ranking phase in a multicriteria decision problem.

4.1.1 How to perform a Principal Components Analysis

Considering that we have a data matrix, X, where the alternatives are defined in a certain basis, the PCA will make a change in the basis, so that, the new space is defined by orthogonal axes. However, PCA is not applied directly to the matrix X [Jackson,1991]. We use a $p \times p$ symmetric, non-singular matrix, M.

Principal Components are generated one by one. To find the first principal component we look for a linear combination of the variables that has maximum sample variance. Then, the second vector will be obtained with the same goal subject to the fact of being orthogonal to the first vector, and so on. The solution to this maximisation problem is based on the fact that the matrix M can be reduced to a diagonal matrix L by premultiplying and postmultiplying it by a particular orthonormal matrix U. This diagonalisation is possible because M is a $p \times p$ symmetric, non-singular matrix.

$$U'MU = L$$

With this diagonalisation we obtain p values, l_1, l_2, ..., l_p, which are called the characteristic roots or eigenvalues of M. The columns of U, u_1, u_2, ..., u_p, are called the characteristic vectors or eigenvectors of M. Geometrically, the values of the characteristic vectors are the direction cosines of the new axes related to the old.

Having the set of data, X, described by p variables, x_1, x_2, …, x_p, we can obtain the eigenvectors corresponding to this data and produce new p uncorrelated variables, z_1, z_2, …, z_p. The transformed variables are called the *principal components* of x.

The new values of the alternatives are called z-scores, and are obtained with this transformation:

$$z = U'x^* \qquad \text{Eq. 4.1}$$

where x^* is $p \times 1$ vector that has the values of an alternative after some scaling.

4.1.2 Types of Principal Components Analysis

The matrix M, from which the principal components are obtained, is defined as described in Eq.4.2.

$$M = \frac{Y'Y}{n} \qquad \text{Eq. 4.2}$$

Different types of principal components analysis exist according to the definition of variable Y in terms of X. Here we underline the three different possibilities [Jackson,1991].

- **Product matrix**
  The first approach consists in taking $Y = X$, that is, perform the analysis from the raw data. However, there are not many inferential procedures that can be applied in this case.

- **Covariance matrix**
  The second approach consists in centring the data, so that $Y = X - \overline{X}$. In this case, we scale the data to be distances from the mean (which is actually a translation of the points). Thus we transform the variables such that all of them have mean equal to 0, which makes them more comparable. It is important to notice that, in this case, the matrix M obtained is the covariance matrix of X.

In the calculation of the covariances the mean is subtracted from the data, so it is not necessary to do it in advance. Then, we obtain the principal components using Eq.4.1, where x^* will be the result of subtracting the mean from the data values.

$$z = U'\left[x - \bar{x}\right] \qquad \text{Eq. 4.3}$$

where x is a $p \times 1$ vector that has the values of an alternative on the original variables, and $\bar{x}$ is also a $p \times 1$ vector that has the mean of each variable.

The covariance matrix is denoted *S* and it is calculated as follows:

$$S = \begin{bmatrix} s_1^2 & s_{12}^2 & \dots & s_{1p}^2 \\ s_{21}^2 & s_2^2 & \dots & s_{2p}^2 \\ \dots & \dots & \dots & \dots \\ s_{p1}^2 & s_{p2}^2 & \dots & s_p^2 \end{bmatrix} \qquad \text{Eq. 4.4}$$

where s_i^2 is the variance of x_i, and the covariance of (x_i, x_j) is calculated as follows:

$$s_{ij} = \frac{n\sum x_{ik} x_{kj} - \sum x_{ik} \sum x_{jk}}{n(n-1)}.$$

PCA based on the covariance matrix is widely applied because the inferential procedures are better developed for this kind of matrix than for any other situation [Jackson,1991]. However, there are some situations in which the covariance matrix should not be used: (i) when the original variables are expressed in different units or (ii) when the variances are different (even though the variables are in the same units). The use of a covariance matrix in these two situations will give undue weight to certain variables (i.e. those that have a large range of values or a large variance).

- **Correlation matrix**
  To avoid the weighting of certain variables, we can work with variables with a common deviation equal to 1. This is obtained by centring and standardising the variables. So, the matrix *M* is, in this case, the correlation matrix of *X*.
  The correlation matrix, denoted by *R*, is computed as follows:

$$R = D^{-1}SD^{-1} \qquad \text{Eq. 4.5}$$

where D is the diagonal matrix of standard deviations of the original variables:

$$D = \begin{bmatrix} s_1 & 0 & ... & 0 \\ 0 & s_2 & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & s_p \end{bmatrix} \qquad \text{Eq. 4.6}$$

The use of correlation matrices is also very common and it is usually the default option in some computer packages (e.g. Minitab). Inferential procedures for this type of matrices are also well defined.

In this case, the z-scores are obtained using Eq.4.3 but using standardised values for x^*. That is, we have to subtract the mean to the data and divide it by the standard deviation. Then, we must multiply it by the eigenvectors.

$$z = U' D^{-1}\left[x - \bar{x}\right] \qquad \text{Eq. 4.7}$$

As it has been previously said, the results obtained with each type of scaling are different. For example, the eigenvectors, *U*, and the z-scores, *z*, are different. In fact, there is no one-to-one correspondence between the principal components obtained from a correlation matrix and those obtained from a covariance matrix.

Other types of vectors can be derived from the characteristic vectors (U-vectors) obtained either with the covariance or the correlation matrix. We are interested in the V-vectors, which properties will be described in the next section. The transformation of the characteristic vectors is done in order to obtain principal components in other scales, in which other properties are fulfilled.

V-vectors are the ones obtained with the following transformation:

$$V = UL^{1/2} \qquad \text{Eq. 4.8}$$

$$\text{i.e.} \quad v_i = u_i \sqrt{l_i} \qquad \text{Eq. 4.9}$$

$$\text{i.e.} \quad v_{ij} = u_{ij} \sqrt{l_i} \qquad \text{Eq. 4.10}$$

Giving weights to the variables:

To give different importance to each variable, we must adjust the matrix used in the PCA (either the correlation or the covariance matrix) using a diagonal matrix with the weights of each variable. Then, the matrix used for the multivariate analysis will be:

$$X \cdot \begin{bmatrix} weight_{c1} & 0 & ... & 0 \\ 0 & weight_{c2} & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & weight_{cp} \end{bmatrix} \qquad \text{Eq. 4.11}$$

4.1.3 Properties

Let us describe the properties of the results obtained in the two most popular PCA approaches: the covariance matrix and the correlation matrix.

PCA based on covariances:

The U-vectors are orthonormal; that is, they are orthogonal and have unit length. Therefore, they are scaled to unity (i.e. the coefficients of these vectors will be in the range [-1,1]). Using these vectors we produce principal components that are uncorrelated and have variances equal to the corresponding eigenvalues. The contribution of each variable to the formation of the i-th principal component is given by the magnitude of the coefficients of u_i, with the algebraic sign indicating the direction of the effect [Dillon&Goldstein, 1984].

V-vectors are also orthogonal but they are scaled to their roots. In this case, the principal components will be in the same units as the original variables. The variances will be equal to the squares of the eigenvalues.

Interpretation of principal components is often facilitated by computing the component loadings, which give the correlation of each variable and the respective component. So, the loading for the j-th variable on the i-th principal component is:

$$\frac{u_{ij}\sqrt{l_i}}{\sqrt{S_{jj}}} \qquad \text{Eq. 4.12}$$

Note that the numerator is actually $v_{ij.}$

PCA based on correlations:

The properties of U-vectors are the same as the ones explained for the case of the covariance matrix. Therefore, the interpretation of their coefficients is the same.

With regard to the V-vectors, in this case, they hold important property: their coefficients show the correlations between the principal components and the original variables, because the variances of the standardised variables are all equal to 1. Thus, if the coefficient v_{ij} is equal to 1 it means that the *i*-th principal component and the *j*-th variable are positively correlated, and if v_{ij} is equal to -1 they are negatively correlated. However, we lose the property of obtaining z-scores in the domain of the original variables.

4.1.4 Stopping rule:

The Principal Component Analysis allows us to reduce the multidimensionality of the data, and represent the information of the initial data set in a *k*-space smaller than the original (with *p* variables), that is, $k<<p$. In the *k*-space the data is easily interpretable. However, the determination of which should be the value *k* is not straightforward. The larger *k* is, the better the fit of the PCA model; the smaller *k* is, the simpler the model will be.

There are different stopping criteria (see [Jackson, 1991]). They are based in the fact that the characteristic roots, $l_1, l_2, ..., l_p$, are decreasingly ordered, that is, $l_1 > l_2 > ... > l_p$. That means that the first characteristic vector is the one that accounts for a higher proportion of variability. These stopping criteria range from methods that evoke formal significance tests to less formal approaches involving heuristic graphical arguments.

For the covariance input, the stopping criteria are usually related to the statistical significance of the eigenvalues. However, for the correlation matrix, these statistical testing procedures no longer apply.

An alternative approach consists of more ad hoc criteria. For example, the cumulative percentage of the variance extracted by successive components, or the Jolliffe's criterion (called Broken Stick), which consists of selecting the *k* vectors, u_j, such that $l_j > g_j$, where g_j is:

$$g_j = \frac{1}{p}\left(\sum_{i=j}^{p}(1/i)\right) \qquad \text{Eq. 4.13}$$

An adaptation of this formula to the case of having variables with uniform distributions is:

$$g_j' = \frac{(n-j+1)(p+j+1)}{\sum_{i=1}^{p}(n-i+1)(p-i+1)} \qquad \text{Eq. 4.14}$$

For the case of the correlation matrix, this variance approach lacks clear meaning, because the standardisation of the data produces a dimensionless standard score space, where the sum of the eigenvalues is equal to the number of variables, *p*. The most frequently used extraction approach in this case is the selection of the components whose eigenvalues are greater than one. The rationale for this criterion is that any component should account for more "variance" than any single variable (remember that variances are equal to 1 because data have been centred and standardised).

4.1.5 Interpretation of the results

A Principal Components Analysis is usually performed for descriptive purposes. In this framework, it is useful to know the global variance of the data we are studying. There is a direct relation between the sum of the original variances and the sum of the characteristic roots obtained with the PCA.

$$Tr(L) = l_1 + l_2 + l_3 + \ldots l_p \qquad \text{Eq. 4.15}$$

In the case of doing the PCA with the correlation matrix, it holds that $Tr(L) = p$ because the variables have been previously standardised.

The value $Tr(L)$ is used to calculate the proportion of the total "variance" attributable to the i-th component, which is $l_i/Tr(L)$.

Another measure that is interesting is the contribution of each observation, *j*, to the formation of a particular component, *i*, denoted $CTR_i(j)$. With this information, we can detect observations that if they were removed from the analysis, the result would be the same. These observations have low contribution values.

$$CTR_i(j) = \frac{z_i^2(j)}{l_i} \qquad \text{Eq. 4.16}$$

We can also measure the cosine of the angle between an alternative *j* and the component *i*, which gives us an idea of the quality of the representation of the alternative if it is projected into the i-th component.

$$\cos_i^2(j) = \frac{z_i{}^2(j)}{d^2(j,G)} \qquad \text{Eq. 4.17}$$

being *d* the Euclidean distance between the observation *j* and the centre of gravity (which is 0 if the data is standardised).

Graphically,

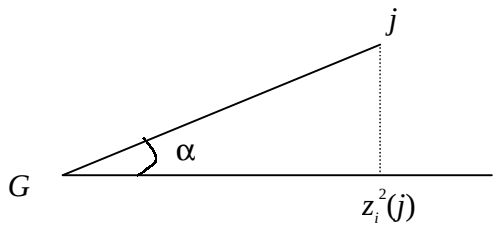

Figure 11. Measuring the quality of representation of alternative *j*

The measure $\cos_i^2(j)$ is, actually, the square of the cosine of the angle α in Figure 11. If we denote as A the distance between *j* and *G*, and B is the distance between $z_i^2(j)$ and *G*, we can see that when Eq.4.17 is equal to 0, A and B are perpendicular, and if Eq.4.17 is equal to 1 then A=B, so *j* is the same as $z_i^2(j)$, which means that there is no loss of information in the change between one space and the other.

We can define a measure of the quality of the representation of a particular observation *j* in a *k*-space (formed with the *k* first components). The maximum value of *QLT* (quality) is 1, which means that the observation is completely representable with the *k* components.

$$QLT_k(j) = \sum_{i=1}^{k} \cos_i^2(j) \qquad \text{Eq. 4.18}$$

On the other hand, it is very interesting to know the meaning of the new space defined by the eigenvectors obtained in the PCA in terms of the initial variables. It is

possible to make a dual analysis with X^T, that is, transposing the data matrix, with which we consider the variables as rows (as observations) and the individuals as columns. Then, using the PCA, we obtain an *m*-space where we can represent the variables in terms of a set of uncorrelated axis (that represent uncorrelated observations). An important property is that this *m*-space is related to the *p*-space obtained with matrix *X*. With this relation, we can use the *p*-space to represent the variables without having to perform the second analysis.

Once we have the variables represented together with the observations, we can use the measures Eq.4.16, Eq.4.17, and Eq.4.18 to infer the meaning of the principal components. In [Volle,1985] there are some guidelines about the process to follow for the interpretation of the new axes, in the case of using the correlation matrix. Note that if we calculate the projection of the variable x_j into the i-th component, $z_i(x_j)$, we can write the contribution and cosine in terms of the V-vectors, because $z_i(x_j) = v_i(x_j)$.

$$CTR_i(x_j) = \frac{v_i^2(x_j)}{l_i} = u_i^2(x_j) \qquad \text{Eq. 4.19}$$

We can see that the contribution of a variable to the i-th component is given by the square of the U-vector obtained when performing the PCA of *X*. The sign of u_i says if it has contributed positively of negatively.

On the other hand, with respect to Eq. 4.17, the distance of each variable to the centre of gravity is 1 (because the data has been standardised). So, the cosine is equal v_i and also it is equal to the correlation between the variable and that component. If $\cos^2_i(x_j)$ is near to 1, x_j can explain the meaning of the axis, because it is really well represented by this axis. In addition, if v_i is near to 1, x_j is positively correlated with the component (and if v_i is -1, it is negatively correlated).

$$CORR_i(x_j) = v_i^{\,2}(x_j) \qquad \text{Eq. 4.20}$$

Finally, there are some measures for the global correlation of the initial variables. One of them is the calculation of the determinant of the covariance or the correlation matrix. In the case of the correlation matrix, *R*, the determinant is sometimes referred to as the "scatter coefficient" [Jackson, 1991]. This coefficient is bounded between 0 (all of the variables are perfectly correlated) and *p* (all of the variables are uncorrelated).

$$|R| = l_1 \cdot l_2 \cdot \ldots \cdot l_p \qquad \text{Eq. 4.21}$$

Another measure is the addition of the individual correlation of each variable to the first component, but having into account the sign of their direction (positive if it has the same direction than the component, and negative otherwise). If all the variables are positively correlated, the sum is equal to the first eigenvalue l_i, so the percentage of correlation is l_i/p.

4.1.6 Application of the PCA to rank order

The principal components found with a PCA can be used to rank the observations [Slottje et. al., 1991]. In the simplest case, we have a set of highly correlated variables and the stopping criterion selects only one component to represent the data. Then, the projections of the observations in this component, z_1, completely define an order among them.

In the case of needing more than one component to represent the information of our set of data, we can combine the components considering the proportion of variance explained by each one. In [Zhu, 1998] the position of each alternative a_j is given by:

$$POS_j = \sum_{i=1}^{k} \frac{l_i}{p} \left| z_{ij} \right| \qquad \text{Eq. 4.22}$$

In this expression, all the values of the observations in the original variables must be positive. If this is not the case, some adjustments must be introduced to Eq.4.22 (see [Zhu,1998]).

We propose to use the Principal Components Analysis to rank the alternatives only if one component is enough to represent our data. If more than one component are needed, the interpretation of the result is far more complicated to automatize. In addition, the measure that qualify the goodness of a ranking obtained with the PCA can only be applied for the case of having the projection of the alternatives in one component (this will be explained in more detail in chapter 5). Therefore, when the first component is not enough to represent the data and perform the ranking, we will use an alternative procedure based on the similarity to an ideal alternative, which is explained in section 4.2.

Now, we are going to see in detail the process that must be followed to obtain the rank order of the partition of alternatives that we have got in the clustering phase. We want to mention here, that usually the PCA is used as a descriptive tool for an statistical expert that knows how to interpret the results in each of the different steps of the process. However, we want to include PCA in a decision-making method that can be implemented and executed automatically to obtain the ranking of the

alternatives without the help of any expert in PCA. For this reason, we have studied in depth this statistical procedure and have selected some measures that can provide a useful knowledge to the decision maker without having to know the insights of this statistical method [Valls&Torra,2002].

First of all, we have to decide which type of PCA to use. As we have seen, there are different ways of performing a PCA depending on the kind of matrix from which we obtain the eigenvectors and eigenvalues. We propose to use the correlation matrix because it will allow us to have variables with different variances. Remember that, in our decision-making framework the variables are the criteria[6], which can have different types of values and different domains.

In the moment of having to perform the ranking, we have the following information available: a data matrix with the alternatives described according to a set of criteria, the grouping of this alternatives into similarity classes and, finally, the prototype of each class (in terms of the same criteria). With the prototypes we can build another matrix, *B*, of the form:

	Criterion 1	*...*	*Criterion p*
Prototype Class A			
...			
Prototype Class G			

Table 6. Prototypes matrix, denoted by *B*

Then, we have two data matrices that can be used to obtain the first principal component: the original data matrix, *X*, and the prototypes matrix, *B*. In principle, PCA could be performed in each of the two matrices. However, the second one has a very short number of objects (between 4 and 9, which are the usual cardinalities of linguistic vocabularies). This is not good for PCA, which is a technique to be used when the number of variables (i.e. criteria) is smaller than the number of alternatives (i.e. classes or objects). Moreover, the values in the matrix of prototypes have not been provided by the experts, they are the result of some computation over the original values, which can introduce error in the interpretation of the result. So, although the objects that we want to rank are the ones in matrix *B*, we should not perform the PCA directly with these data. The PCA will be done in the original data matrix, and then, the prototypes of the classes will be introduced in the new space in order to be ranked.

[6] In the data matrix we can have criteria given by a single expert or by different experts.

We can distinguish 5 steps in the process of applying the Principal Components Analysis to our data. These steps must be followed sequentially. At the end, we will have a ranking of the classes and some values that will be used to measure the goodness of the result, and to infer the relations among the variables (i.e. preference criteria).

STEP 1 – Apply the Principal Components Analysis to the data matrix. Obtain the eigenvalues, l_i, eigenvectors, u_i and V-vectors, v_i.

STEP 2 – Check if the first component is enough to perform the ranking. To decide whether it is enough or not, we must apply a stopping criteria (section 4.1.4) and see if the number of selected components is one or greater. As we are working with correlation matrices, we propose to use the criteria that selects those vectors that account for more than a 1% of variance, that is, l_i>1.

If we need more than one principal component to represent our data, we will execute step 4 (to obtain some additional information) and end.

STEP 3 – Use the first V-vector to know the meaning of the first component. A value near zero means that the variable has no influence in the interpretation of the component, while the higher the absolute value of the variable, the more the component is saying the same than the variable. We can apply Eq.4.20 to calculate the relation between each variable and the first axis and find the variables with higher correlation.

Once, we have got the variables that can explain the meaning of the axis, we need to know if they are positively or negatively correlated, this can be found looking directly into the V-values of the first axis, v_1. The sign indicates the direction of the variable in relation to the component. This is particularly interesting because we must determine which is the direction of the first component in order to know which are the best alternatives. In our case, all the variables are expressing preferences, where the higher the value, the more preferred the alternative is. Thus, the sign of coefficients of v_1 should be the same if all the criteria agree in giving the same kind of preference (good or bad) to the same alternatives. When a criteria is saying the contrary than the others, its sign will be the opposite of the others. In case of having a set of positively correlated variables of similar dimension to the set of negatively correlated variables, we will stop the MCDA process because the direction of the first component cannot be established.

STEP 4 – Calculate the contribution of each variable to the formation of the first principal component (Eq.4.19). If a variable did not contributed to the formation of the first axis, it means that this variable does not give any useful information for the determination of the axis to be used in the ranking.

When a variable highly contributes to the second principal component and not to the first one, we can say that this variable is in contradiction (it is perpendicular) to our social axis, which is the first one.

If a variable does not contribute to any axis, it means that it can be eliminated from the analysis and the result would not be significantly different.

STEP 5 – Find the z-scores of the prototypes in the first principal component, z_1, using (Eq.4.3), where x^* are the columns of the prototypes matrix. Before, these values have been centred and standardised.

The z-scores tell us the position of each class into a line, which defines a total order among them. The direction of the director vector of this line determines which is the best and worse position. This direction has been found in step 3. Thus, the ranking of the classes we were looking for is already set.

If the process finishes successfully, in step 5 we have obtained the z-scores in the first principal component, z_1. However, the values of z_1 do not belong to a predefined real interval. To be used in the following stages of the MCDM process, we need to know the position of the clusters in the [0,1] interval. To perform this scaling for a given prototype, j, we use Eq.4.23.

$$z_{01}(j) = \frac{z_1(j)}{z_1(a_{ideal}) - z_1(a_{nadir})} \qquad \text{Eq. 4.23}$$

The a_{ideal} is a fictitious alternative that takes the best possible value for each criterion. If this alternative existed, it will be the most preferred by the decision maker. On the other hand, the a_{nadir} is a fictitious alternative with the worst possible value for each criterion.

4.2 Ranking based on the similarity to the Ideal alternative

The second procedure, denoted as CASE B in the description of the ranking phases, corresponds to the situation in which criteria are not correlated enough. For this case, we propose the application of another ranking technique based on similarity functions. Due to the distinct opinions of the experts (or criteria suppliers) or the incomparable meaning of the criteria, we will need a separable measure, which compares the objects criterion by criterion.

We assume that for each criterion there is a single value of its domain, v_{ij}, which is the best. That is, if alternatives were only described with this criterion, the ones with value v_{ij} will be selected by the decision maker. With the values v_{ij} we build an ideal alternative, denoted a_{ideal}, which is the one that has the best value for each criterion. This ideal alternative is the same one considered in the previous section to locate alternatives in the [0,1] interval.

The ranking is based on the comparison of prototypes with respect to the ideal alternative. The alternatives that belong to the class whose prototype is nearer to a_{ideal} are the best ones. To compare them we must use a similarity measure, like the ones used during the clustering process.

With this approach, the position in $\Re$ of a cluster is given by:

$$z(j) = similarity\left(prototype_j, a_{ideal}\right) \qquad \text{Eq. 4.24}$$

where the lower the z, the better the cluster is.

A similar approach is the one known as TOPSIS (Technique for Order Preference by Similarity to Ideal Solution), developed by Yoon and Hwang [Hwang&Yoon,1981]. TOPSIS is based on the concept that the selected alternative should have the shortest distance from the ideal solutions and the farthest distance from the negative-ideal (nadir) solution. Therefore, they define a measure of the relative closeness to the ideal as:

$$C_{i*} = \frac{\sqrt{\sum_{j=1}^{p}\left(v_{ij} - v_{j-}\right)^2}}{\sqrt{\sum_{j=1}^{p}\left(v_{ij} - v_{j*}\right)^2} + \sqrt{\sum_{j=1}^{p}\left(v_{ij} - v_{j-}\right)^2}}$$

That is, they calculate the Euclidean distance between the alternative a_i and the ideal, defined as $a_{ideal} = (v_{1*}, v_{2*}, ..., v_{p*})$, and the Euclidean distance between the alternative a_i and the nadir one, $a_{nadir} = (v_{1-}, v_{2-}, ..., v_{p-})$. Then the ranking of the alternatives in found according to the preference rank order of C_{i*}.

Using the TOPSIS approach, if we have two alternatives with same similarity to the ideal, the one that is furthest from the nadir is the one considered as best than the other one. If we represent it in a two-dimensional space (Figure 12), we can see, that the alternative more distant to the nadir is the one that has a greater difference in the values given by the two criteria (*a* is considered as 0.5 for one criterion and 0.8 for the other). Their corresponding closeness preference values according to TOPSIS will be: $C_{a*} = 0.64$ and $C_{b*} = 0.62$. So, the best one is *a*.

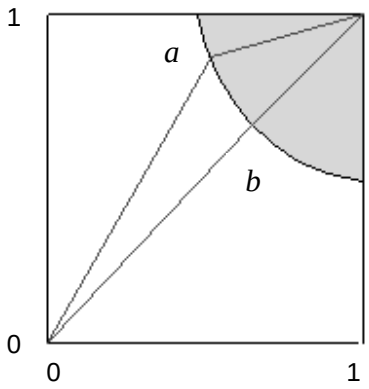

Figure 12. Ranking of alternatives with TOPSIS

However, this approach does not have into account the agreement between the criteria. Under our point of view, alternative *a* is as good as *b* with respect to the goal of achieving the values of the ideal solution. The difference between them is related to the knowledge we have about their goodness. For this reason, we propose to consider them as equal and give extra knowledge to the decision maker about the trustworthiness of their position in the preference ranking. As it will be explained in more detail in the next chapter, our confidence on *b* is greater than on *a*, because the two criteria give the same value to *b*, whereas our knowledge about *a* is that it can be as good as 0.8 indicates, or it can be not so good as 0.5 says. For this reason, the ranking method we propose only compares the prototypes with the ideal alternative.

Moreover, after studying the properties and behaviour of different similarity measures to rank the clusters, we propose the use of the Manhattan distance if we have qualitative criteria in our decision matrix. The Manhattan distance (Eq.3.3) is

appropriate when working with numbers that represent linguistic terms, as it has been argued in section 3.3.3, where it has been recommended to be used in the aggregation process.

If no qualitative criterion is considered, we recommend to apply the same measure used in the first stage, so that the same conditions apply during all the process (this is, to avoid different similarity functions in the same process because each similarity function has different properties).

4.2.1 Application of the similarity-based ranking

As said, this ranking procedure will be used in case of having non-correlated preference criteria or descriptive criteria with a non-ordered domain. The information provided by the aggregation stage is the same than in the PCA ranking: a data matrix with the alternatives described according to a set of criteria, the grouping of this alternatives into similarity classes and, finally, the prototype of each class in terms of the same criteria.

To find the ranking, we start with the prototypes of the clusters. For each prototype we measure the similarity (or distance) to the *ideal* alternative. The result will indicate a degree of preference of a particular cluster.

Repeating this distance measurement for all the prototypes we obtain a numerical degree of preference of all clusters (we denote by *z*(*j*) the numerical value of the j-th cluster Eq. 4.24). Using these values we can determine an order among the clusters.

Now, we have got a rough approximation of the position of the clusters in a numerical interval [a,b]. As we have explained in section 4.1.6, the values that the following stages require must be in the [0,1] interval. For this reason we must apply the same transformation function that was indicated for the PCA method, Eq. 4.23, which is reproduced here:

$$z_{01}^{*}(j) = \frac{z(j)}{z(a_{ideal}) - z(a_{nadir})}$$

In this case, z(a_{ideal}) will be 0 because the distance between the ideal solution and itself is 0. Moreover, the values we obtain will be ordered from best to worse, that is, the alternative with a lower z_{01}^{*} will be the best one, whereas in the PCA ranking the

ordering was the opposite. For this reason the following transformation is applied to the z_{01}^{*} values.

$$z_{01} = 1 - z_{01}^{*} \qquad \text{Eq. 4.25}$$

After these calculations, the result of the ranking stage for case B is the same than case A: we have obtained a totally ordered set of clusters. This leads to an ordered partition of the alternatives. This ordered partition defines a new qualitative ordered criterion.

CHAPTER 5. Explanation and Quality stages

The outcome of the ranking stage is an ordered set of clusters, where each cluster is defined in terms of several alternatives. This cluster has also associated a value in the [0,1] interval corresponding to a rough approximation of its position on the "social axis". In this section we describe how to associate a linguistic term to each cluster (and, therefore, to each alternative). The linguistic terms will replace the numerical rough approximations computed in the previous stage. To complete the process and obtain a new qualitative preference criterion, we must establish not only the vocabulary but also the negation-based semantics of this criterion.

In the first part of this chapter, the complete methodology to build the new qualitative criterion is explained. Several algorithms have been developed in order to deal with all the special situations and obtain a good vocabulary with an appropriate semantics. This is very important because these are the tools that we give to the user to understand the result of the decision making process.

The second part of the chapter is devoted to the evaluation of the goodness of this new criterion, which we have called: the quality measurement stage. This goodness is calculated from the information provided at the different stages of the process: the aggregation through clustering, the ranking (with the Principal Components Analysis or with the Similarity calculation) and the vocabulary building. Many different factors are analysed and included in a final qualitative measure of the trustworthiness of the resulting criterion. However, we also recommend having into account not only the final qualification but also the partial quality measures of each stage.

5.1 Giving semantics to the ordered set of clusters

The main goal of this phase of the process is to give meaning to the ordered qualitative domain of the new-created criterion. At this stage, the values of this domain are terms artificially generated in the first stage. We want to change these terms by others that have a meaning easily understandable for the decision maker.

We propose a new method to select the most appropriate linguistic terms to describe each cluster of alternatives. With these terms we build the vocabulary and semantics of the new overall criterion.

The vocabulary can be obtained from the ones used by the different preference criteria in the data matrix, or it can be given by the user. Once we have the set of possible terms to be used, we apply a new assignation procedure to select the best term for each cluster. During this process, we can split up some terms to obtain others with a finer semantics, that is, to generate more precise terms. The new linguistic labels are obtained using linguistic hedges.

When the selection of the terms to be used has been done, the new vocabulary has been established. The next step consists of giving the semantics to these terms that is, building the negation function over this vocabulary.

5.1.1 The vocabulary of the result

To determine which is the most appropriate set of terms to be used in the new criterion, we distinguish two different situations:

CASE C: The decision maker provides a vocabulary to be used in this stage. This vocabulary must consist of a finite ordered set of terms and a negation function over these terms.

CASE D: No vocabulary is given by the decision maker. Then, the system has to choose one of the vocabularies of the criteria provided by the experts when they have filled the decision matrix.

We believe that the less parameters the user has to define when running a decision support system, the more encouraged to use it he will be. The large amount of information required to the decision maker may be a counterpart for its use in daily situations. For this reason, we will only consider CASE C when there is no possibility

to describe the result with the vocabularies of the original criteria. For example, in Table 7 we have that the three criteria are not appropriate for expressing a preference ranking over the alternatives. Thus, the user should provide a vocabulary like the one in the last row.

	lowest value . . . *largest value*
Weight	lean, thin, normal, corpulent, fat, overweighted
Distance	same_place, close, near, far, remote, outlying
Waiting time	very_short, short, acceptable, long, very_long
Preference	terrible, bad, not-recommendable, acceptable, recommendable, good, very_good, ideal

Table 7. Qualitative vocabularies of the criteria vs. preference vocabulary for the raking

We can see that the vocabularies in Table 7 are ordered sets of terms, but the higher value does not necessary mean that it is the desired value. For example, concerning the weight, we may prefer a corpulent person than a fat or a normal one.

In CASE D or when some of the vocabularies of the criteria are already expressing preferences over the alternatives, we can use their values to qualify the clusters of alternatives without having to ask to the decision maker. In this case, we have the problem of choosing a vocabulary among the possible ones. We have defined a distance measure between ordered qualitative vocabularies, d_v, based on the fact that each vocabulary is a set of bounded closed non-overlapping intervals in [0,1].

First, we define a *centre function* as a function that assigns to each value x_i in [0,1] another value in [0,1] that is the value of the central point of the interval $(m,M]$ to which x_i belongs to. This centre function is a left continuous step function.

Having two vocabularies, V_A and V_B, we denote A and B their corresponding centre functions, such that, for any $x \in [0,1]$,

$$A : x \to a_x$$
$$B : x \to b_x$$

where a_x is the central point of the interval of A to which x belongs, and b_x is the central point of the interval of B to which x belongs.

Then, we define a measure of similarity between vocabularies as follows:

$$d_v(V_A, V_B) = d_v(A, B) = \left[\int_0^1 d^2(a_x, b_x)dx\right]^{1/2} \quad \text{Eq. 5.1}$$

where $d^2(a_x, b_x) = (a_x - b_x)^2$.

It can be easily seen that $d(a_x, b_x) = \sqrt{(a_x - b_x)^2}$ is the Euclidean distance between two points.

Theorem: $d_v(V_A, V_B)$ is a distance.

Proof.

(1) Positivity.

According to the definition of $d_v(V_A, V_B)$, the result cannot be negative, $d_v(V_A, V_B) \geq 0$.

Let's proof that if $d_v(V_A, V_B) = \left[\int_0^1 d^2(a_x, b_x)dx\right]^{1/2} = 0$ then $V_A = V_B$

We will show that when $d_v(V_A, V_B) = 0$, for any $x \in (0,1]$, $d^2(a_x, b_x) = 0$, which means that a_x and b_x are always equal ($V_A = V_B$).

Let us suppose that there exists $x' \in (0,1]$, such that $d^2(a_{x'}, b_{x'}) = (a_{x'} - b_{x'})^2 \neq 0$, as A and B are left-continuous step functions, for any $x' \in (0,1)$, there exists an $x'' \in (0,1)$, $x'' < x'$ such that $(a_x - b_x)^2 = (a_{x'} - b_{x'})^2$ for any $x \in [x'', x']$. So,

$$\int_0^1 d^2(a_x, b_x)dx \geq \int_{x''}^{x'} d^2(a_x, b_x)dx = \int_{x''}^{x'} (a_x - b_x)^2 dx = \int_{x''}^{x'} (a_{x'} - b_{x'})^2 dx = (a_{x'} - b_{x'})^2 (x'' - x')$$

as $(a_{x'} - b_{x'})^2 \neq 0$ and $(x'' - x') > 0$, we have that the previous expression is positive, i.e., $\int_0^1 d^2(a_x, b_x)dx \geq (a_{x'} - b_{x'})^2 (x'' - x') > 0$, which contradicts the original assumption $d_v(V_A, V_B) = \left[\int_0^1 d^2(a_x, b_x)dx\right]^{1/2} = 0$.

So, it is not possible to find any $x' \in (0,1]$ such that $d^2(a_{x'}, b_{x'}) = (a_{x'} - b_{x'})^2 \neq 0$. Therefore, $a_{x'} = b_{x'} \forall x' \in [0,1]$, i.e. $V_A = V_B$

(2) Symmetry.

For any V_A, V_B,

$$d_v(V_A, V_B) = d_v(A, B) = \left[\int_0^1 d^2(a_x, b_x)dx\right]^{1/2} = \left[\int_0^1 d^2(b_x, a_x)dx\right]^{1/2} = d_v(B, A) = d_v(V_B, V_A)$$

since $d^2(a_x, b_x) = (a_x - b_x)^2$ is symmetric.

(3) Triangle inequality.

We want to show that $d_v(V_A, V_B) \leq d_v(V_A, V_C) + d_v(V_C, V_B)$.

We know that $d(a_x, b_x) \leq d(a_x, c_x) + d(c_x, b_x) \ \forall x \in [0,1]$, because it is a distance.

From this inequality we can also have, $d^2(a_x, b_x) \leq (d(a_x, c_x) + d(c_x, b_x))^2$ or $d^2(a_x, b_x) \leq d^2(a_x, c_x) + d^2(c_x, b_x) + 2d(a_x, c_x) \cdot d(c_x, b_x)$.
So, if we introduce the bounded integral in each operand, $\int_0^1 d^2(a_x, b_x)dx \leq \int_0^1 d^2(a_x, c_x)dx + \int_0^1 d^2(c_x, b_x) + 2\int_0^1 d(a_x, c_x) \cdot d(c_x, b_x)dx$, the inequality is also true.

Since $\int_0^1 d(a_x, c_x) \cdot d(c_x, b_x)dx \leq \left(\int_0^1 d^2(a_x, c_x)dx \cdot \int_0^1 d^2(c_x, b_x)dx\right)^{1/2}$, we have that

$$\int_0^1 d^2(a_x, b_x)dx \leq \int_0^1 d^2(a_x, c_x)dx + \int_0^1 d^2(c_x, b_x) + 2\left(\int_0^1 d^2(a_x, c_x)dx \cdot \int_0^1 d^2(c_x, b_x)dx\right)^{1/2}$$

or $\int_0^1 d^2(a_x, b_x)dx \leq \left[\left(\int_0^1 d^2(a_x, c_x)dx\right)^{1/2} + \left(\int_0^1 d^2(c_x, b_x)\right)^{1/2}\right]^2$ which is exactly the triangle inequality property:

$$\left[\int_0^1 d^2(a_x, b_x)\right]^{1/2} \leq \left[\int_0^1 d^2(a_x, c_x)\right]^{1/2} + \left[\int_0^1 d^2(c_x, b_x)\right]^{1/2}$$

□

This distance measure take values in [0,0.25], being 0 the value indicating that two vocabularies are identical, and being 0.25 the maximum distance value for two different criteria. This maximum is obtained when the intervals of the two negation-based vocabularies are completely different.

Proof.

The difference between the centers of two overlapping intervals reaches its limit when these intervals are large and are positioned far from one to each other. The maximum length of the intervals is achieved having the minimum number of terms. That is, having a vocabulary with only 1 term, and the other one with 2 terms (Figure 13).

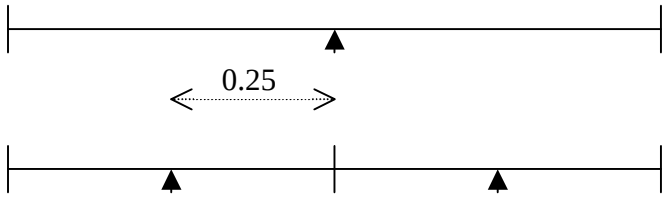

Figure 13. Maximum distance between overlapping negation-based intervals

In this situation, the maximum difference of the centers is 0.25 for all the points in the domain [0,1]. Therefore, for all x in [0,1], we have

$$d^2(a_x, b_x) = 0.25^2 = 0.0625$$

which can be substituted to d_v to obtain the maximum distance value:

$$d_v(V_A, V_B) = \left[\int_0^1 0.0625 dx\right]^{1/2} = 0.25$$

□

We apply the distance d_v to measure the similarity between each vocabulary given by the experts and the result of the ranking phase, which is a set of ordered names of clusters.

For each vocabulary of the criteria we have a negation function that allow us to obtain the interval $(m,M]$ corresponding to each term (using Eq.3.1). Obtaining the centre of this interval (i.e. a_x) is straightforward. Moreover, for each cluster we know the position of the prototype in the interval [0,1]. Being b_x the centres of the intervals, it

is possible to know the boundaries of the intervals. Therefore, we have all the information needed to calculate the integral in expression Eq. 5.1.

The criterion whose vocabulary is the most similar to the set of clusters is selected to be used to explain the meaning of those clusters.

5.1.2 Assigning the most appropriate term to each cluster

Once we have the final vocabulary selected (or provided by the user), we have to assign a term of this vocabulary to each class. This term will describe the suitability of the cluster for the decision problem. Moreover, we can only use each term once, because if more than one cluster receives the same term, they will be indistinguishable.

We have a method to solve this selection problem. Some intuitive assumptions have been considered:

- no cluster with a position, z_{01}, lower than 0.5 will receive a positive term
- no cluster with a position, z_{01}, higher than 0.5 will receive a negative term
- if a cluster is near the centre, 0.5, it will receive the neutral term
- the neutral term, if exists, will have a negation equal to itself

With this requirements, we have developed the following procedure that divides the vocabulary into three parts: positive terms (those with a preference higher to 0.5), negative terms (those with a preference lower than 0.5) and the neutral term (the one whose negation is itself, and its value is 0.5). For knowing the position see the semantics induced by the negation function (Eq.3.1).

According to the negation function it is possible that the selected vocabulary does not have any neutral term. In this case we will have the vocabulary divided into two sets, instead of three.

The procedure has 6 steps:

1. Find the cluster with corresponding z-value equal to $0.5 \pm \xi$, which will be denoted $C_{neutral}$
2. If it exists then assign to it the neutral term, $T_{neutral}$ (if the vocabulary does not have a neutral term, it will be provided by the user). For further calculations, consider that $C_{neutral}$ is positioned in z_{01}=0.5.
3. Divide the clusters into two groups:
   Positive Clusters (positioned between 0.5 and 1) and

Negative Clusters (positioned between 0 and 0.5)

4. Divide the vocabulary into two groups:
   Positive Terms (following $T_{neutral}$) and
   Negative Terms (preceding $T_{neutral}$)

5. If the granularity of any group is smaller than the number of clusters of the corresponding group, apply the algorithms `Making_new_labels` and `Make_names` until we have the same number of terms than clusters.

6. Apply the algorithm `Explain_result` to the 2 groups independently.

Two additional algorithms have been defined in order to sort out two particular steps of this assignation process [Valls&Torra, 2000b]. Firstly, we will see the algorithm to assign terms than are able to explain the result (i.e. the alternatives according to the clusters). The inputs to the algorithm are the set of ordered clusters and the set of ordered terms to be used to qualify the clusters.

```
Algorithm Explain_result is
 k := number of clusters to be explained
 if k=number of terms then
     Assign these k terms to the k clusters
 else
     Take the best cluster of the set (C_best)
     While k>0 do
        Take all those terms in the vocabulary that have at least k-1 worse
         terms [t_b..t_a]. Moreover, t_a should not be better than any
         previously assigned label.
        If similarity(C_best, Ideal) belongs to one of the intervals of the
         terms in [t_a..t_b] then
           C_best takes the term corresponding to this interval
        else
           if similarity(C_best, Ideal)>I(t_a) then
                   C_best takes t_a (the best possible label)
           elsif similarity(C_best, Ideal)<I(t_b) then
                   C_best takes t_b (the worst possible label)
           end if
        end if
```

```
        k := k-1;
        If  k = number of terms that follow the assigned term then
            Assign these k labels to the k remaining clusters
            k := 0;
        else
          Take the cluster that follows C_best in the ranking, and call it C_best
        end if
      end while
  end if
end algorithm.
```

This method pretends to give the most appropriate term to each cluster maintaining always the ranking among them. However, we suppose that the decision maker is particularly interested in knowing which are the best alternatives, because he is trying to make a good decision. Thus, we start the process with the selection of the most suitable term for the first cluster in the ranking, provided that we leave enough terms for the rest of clusters.

This algorithm needs a set of terms equal or larger than the set of clusters. If the vocabulary selected does not have enough terms, we have designed an method to create new terms using the ones that we have in the vocabulary. The key idea is to split some terms up and use a qualifier to distinguish the two new parts. So, the problem is reduced to the selection of the labels most adequate to be split.

As we have some information (given by the negation function) about the meaning of the labels in a vocabulary, we can use it to guide the process. A label that has more than one label in its negation indicates that there are slight differences between some of the alternatives assigned to it, in some sense, there is a gradation in the meaning of the label, and each degree corresponds with a label in the negation. Under this interpretation, this label is a candidate to be split up.

```
algorithm Making_new_labels is

  repeat
      {t_left, t_right}:= split the most suitable label, t_k
         T := remove t_k from T
         T := add t_left and t_right to T
  until we have the desired number of terms

end algorithm.
```

We assume that the labels in a vocabulary cover all the possible values in [0,1]. Each label t_i corresponds to a fixed interval $[m_i, M_i]$, as in Figure 14.

The splitting method begins by looking for the possible cut points. This is done with the help of the negation function, which is used to calculate the numeric intervals of each label. Then, these values are projected into the opposite part of the domain [0,1] to find out which labels have more specific meanings.

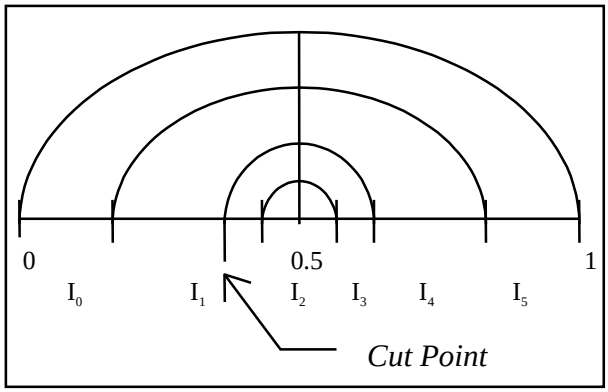

Figure 14. The negation procedure for generating new terms

Once we have got the cut points, we apply each one of these cuts to the vocabulary separately. Thus, we obtain a new possible vocabulary for each cut point. Then, each new vocabulary is compared to the ordered names of the clusters (the result of the second stage) using the distance we have defined, d_v. The vocabulary that is closer to the partition is chosen, and two new labels are obtained from the one we have split. If we already do not have enough labels, we repeat the process of applying the cut points but now they are applied to this new vocabulary.

However, it is possible to have some situations where the negation cannot produce the number of new terms required [Valls&Torra, 1999a]. For example, when the negation function is the classical one, we cannot obtain any new term because all have the same dimension. Then, if the clusters obtained are concentrated on one side of the vocabulary (if they are mainly good or bad), we will have a lack of terms.

In this particular case, where the negation-based semantics cannot help, the solution proposed consists of identifying the term that has a larger number of clusters to explain, and split it up. This process can be repeated until we have produced the desired number of terms.

When a term is selected to be split, t_i, we have to divide its corresponding interval $[m_i, M_i]$ and obtain $[m_i, c_i]$ and $[c_i, M_i]$. In order to obtain the most accurate cut point, c_i, we propose to use the information of the position of the clusters.

Let us suppose that we have 3 clusters (α, β and γ) with the following z_{01} positions after the ranking (see Figure 15):

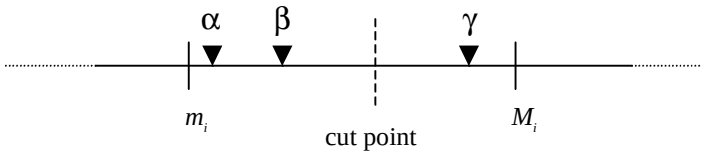

Figure 15. Selected cut point for the interval $[m_i,M_i]$

The most suitable cut point is the one between the two clusters that are more distant from each other. That is, if in Figure 15 the distance between α and β is 0.05 and between β and γ is 0.15, we decide to break up the interval just in the middle between β and γ, since the meaning of the two clusters is more different than the meaning of β with respect to α.

Each time we split a term up, two new terms are needed. The method to create new terms for the new intervals in a vocabulary is not trivial, because they should be in accordance with the rest. For this reason, we do not invent them, we introduce linguistic hedges (e.g. *very*, *not-so*, ...) in order to distinguish the different grades in the meaning of the term.

To keep the structure of the qualitative vocabularies, we have decided that the neutral label (if exists) it is never split up, since its meaning is that its negation is itself, and an split will end with this property. The rest of the vocabulary can be divided in two sets: T_{inf} and T_{sup}. T_{inf} has the labels that are smaller than the neutral value, and T_{sup} the ones that are greater than the neutral value. Then, the process is the following:

```
algorithm make_names is
   if t∈T_inf then
         if t has not been previously split then
           return {very-t, t }  being very-t < t
         else  /* this means that very-t exists */
           return  { t, not-so-t }  being t < not-so-t
         end if
   else  /* t∈T_sup */
         if t has not been previously split then
           return { t, very-t }   being t  < very-t
         else  /* this means that very-t exists */
           return  {not-so-t, t }  being not-so-t < t
         end if
   end if
end algorithm.
```

We express the grades in the meaning, introducing a new more precise term that uses the modifiers *very* or *not-so*.

This algorithm assumes that we will only cut a term once or twice. That is, we will not generate more than 3 terms from a single one. We consider that if more than 3 terms must be obtained, we should ask the decision maker (i.e. the user) in order to obtain more appropriate terms.

Regarding the global process presented in this section, it may produce bad results if there are some clusters whose positions are very close (we should consider that a difference of only the 20% of the length of the term is problematic). This situation indicates that we have two clusters that are very similar in relation to the ranking position (given by the Principal Components Analysis or by the Similarity-based Ranking) but whose elements were not considered similar enough to be assigned to the same class, during the clustering process (the aggregation stage). This is a problematic situation, since the ranking methods have not distinguished the goodness of the two different clusters in relation to the ideal alternative. However, the quality measures that we have defined (which will be detailed in section 5.2.2) will give us some idea of the trustworthiness of the ranking obtained. If the degree of quality is under some threshold, the decision maker can decide to stop the process, or to ignore the values finally given to these conflicting clusters.

5.1.3 Building the negation function of the new criterion

Once we have got a set of terms, possibly adapted to fit the consensus partition, we have to study their semantics. If the consensus partition were identical to the expert's one, the meaning of the terms would not change, but this will usually not be the case. The meaning of the terms has to be built knowing the alternatives that each term is now describing.

Following the approach based on negation functions, the meaning of each term is going to be expressed using the negation. Moreover, this is also the form in which experts have supplied their knowledge. So, they are supposed to be familiar with the negation concepts and notation. Therefore, it will be an easy and comprehensible form to express the meaning of the new terms.

To calculate the new negation function, first we have to attach a numerical interval in [0,1] to each label, $I(t_i)$. The disjoint intervals are built with the positions z_{01} of the clusters into the first principal component. Using the fuzzy approach for linguistic labels, we can say that the labels have a triangular membership function

[Yuan&Shaw,1995] (except in the extremes), so the z-value is taken as the point of the label where the membership value reaches 1.

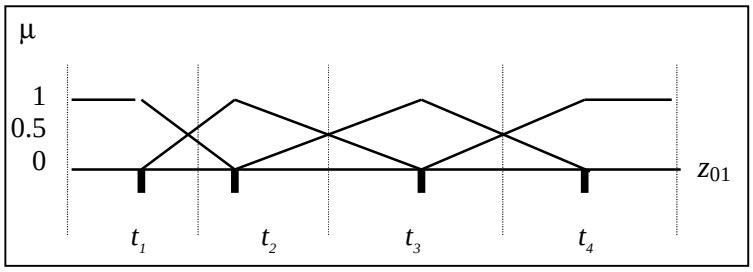

Figure 16. Fuzzy partition used to establish the semantics

If some of the terms of the vocabulary have not been used to explain the clusters obtained in the previous stages, we include a new imaginary cluster with a prototype positioned in the centre of the interval corresponding to this term. Then, the negation function is built with the real and imaginary prototypes. The additional prototypes are located in the centre in order to try to avoid the changes in the limits of the labels that are not used in the result, since we do not have any information about what should be their meaning in the new criterion.

In order to keep the neutral term centred in 0.5, we begin the process of building the fuzzy sets from the middle. If the two neighbour prototypes are not located at the same distance from 0.5, we take the nearest prototype location as the boundary of the support of the fuzzy set of the term. For example, in Figure 17 we can see 3 prototypes (marked with a bold line), the one in the left is the closest to the neutral class, so this establishes the point where the membership to the neutral cluster ends in the left. Since the similarity function of the neutral term must be symmetrical, we have that the end of the membership function in the right is established at the same distance to the centre than the prototype in the left.

Once the fuzzy set of the neutral term has been fixed, we continue with the rest of the membership functions as explained before.

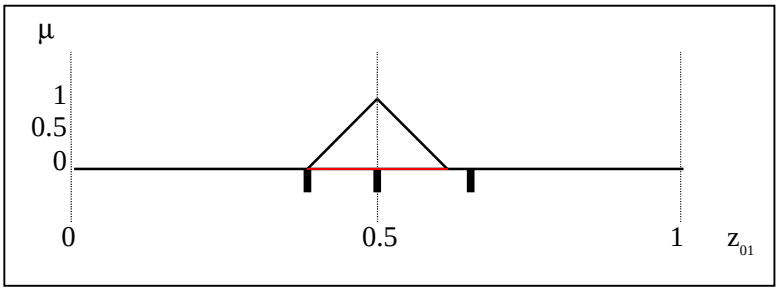

Figure 17. Negation for the neutral term

It can be observed that, in general, the middle point between two consecutive projections is the one that has membership equal to 0.5. These are the points usually corresponding to the limits of intervals, as in the example of Figure 18.

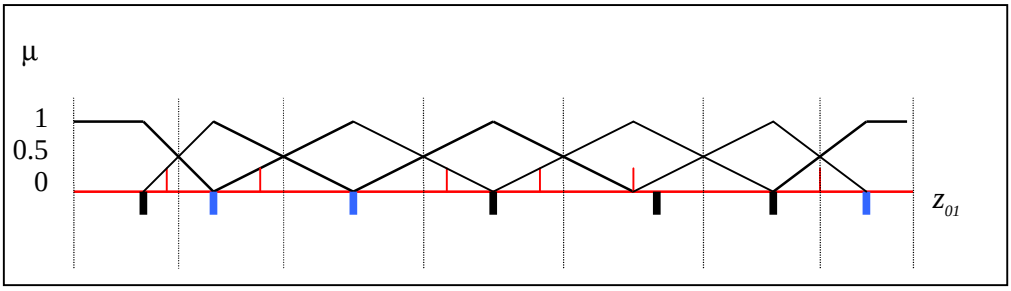

Figure 18. Fuzzy sets corresponding to an example with 4 clusters (the blue marks correspond to imaginary prototypes for unused terms, the black ones are the real positions of the clusters after the ranking, the red line corresponds to the distribution of the terms according to the original negations)

Once each term has its corresponding interval in the new criterion, $I(t_i)$, the negation of each one can be computed as:

$$Neg(t_i) = \{ t_j \mid I(t_j) \cap 1\text{-}I(t_i) \neq \varnothing \} \qquad \text{Eq. 5.2}$$

where $1\text{-}I(t_i)$ is the interval between $1\text{-}\max(I(t_i))$ and $1\text{-}\min(I(t_i))$.

Using Figure 18 we will follow an example of the negation function generation. We will see that some problems appear, and we will present some methods to sort them out.

Let us take that the original vocabulary is $\{l_1, l_2, l_3, l_4, l_5, l_6, l_7\}$, and its semantics is given by the negation function we have in the first column of Table 8. The second

column shows the interval corresponding to each label according to this semantics (using Eq.3.1).

Original Negation	Original Intervals
Neg (l_1) = {l_7}	I (l_1) = [0.0, 0.11]
Neg (l_2) = {l_6}	I (l_2) = [0.11, 0.22]
Neg (l_3) = {l_5, l_6}	I (l_3) = [0.22, 0.44]
Neg (l_4) = {l_4}	I (l_4) = [0.44, 0.56]
Neg (l_5) = {l_3}	I (l_5) = [0.56, 0.67]
Neg (l_6) = {l_2, l_3}	I (l_6) = [0.67, 0.89]
Neg (l_7) = {l_1}	I (l_7) = [0.89, 1.0]

Table 8. Semantics for the example with 7 terms

Now we look at the positions of the clusters. Let us suppose that we have obtained 4 clusters. In Table 9 we have the positions of the real (black) and additional (blue) prototypes.

Class	Positions in [0,1]	Term
7^{th}	0.09	l_1
6^{th}	0.16	l_2
5^{th}	0.33	l_3
4^{th}	$0.48 \rightarrow 0.5$	l_4
3^{rd}	0.71	l_5
2^{nd}	0.80	l_6
1^{st}	0.94	l_7

Table 9. Positions of the 4 clusters in the example

After applying the methodology to build the new intervals for the terms, we obtain the result shown in Table 10. The first column is the result of the interval generation based on the fuzzy membership functions. The second column is the opposite interval corresponding to each term, which is calculated doing 1-x_i. Finally, the third column gives the negation induced by these intervals, considering that a difference of 0.02 in the value of the borders is not significant. In general, if we have 7 terms in the vocabulary each one covers a 14% of the domain, so a 0.02 is only 1/7 of the length of a term. However, this value could be changed according to the characteristics of the application domain or the decision maker opinion.

Intervals from fuzzy sets	Opposite interval	Negation induced
I (l_1) = [0.0, 0.125]	[1.0, 0.875] ≅ I(l_7)	Neg (l_1) = {l_7}
I (l_2) = [0.125, 0.245]	[0.875, 0.755] ≅ I(l_6)	Neg (l_2) = {l_6}
I (l_3) = [0.245, 0.33]	[0.755, 0.67] ≅ I(l_5)	Neg (l_3) = {l_5}
I (l_4) = [0.33, 0.67]	[0.67, 0.33] = I(l_4)	Neg (l_4) = {l_4}
I (l_5) = [0.67, 0.735]	[0.33, 0.265] ≅ I(l_3)	Neg (l_5) = {l_3}
I (l_6) = [0.735, 0.87]	[0.265, 0.13] ≅ I(l_2)	Neg (l_6) = {l_2}
I (l_7) = [0.87, 1.0]	[0.13, 1.0] ≅ I(l_1)	Neg (l_7) = {l_1}

Table 10. Result of the semantics generation

Notice that, in this example, we have obtained a new criterion with the classical negation function. In Figure 19 we can see the distribution of the intervals according to the new semantics against the original distribution. As we can see, the new intervals are more suitable to explain the clusters, because each cluster belongs to a different interval.

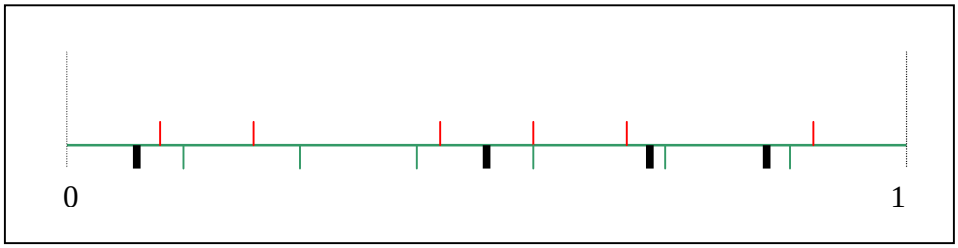

Figure 19. Comparison between the old (up) and new (down) intervals

It is worth to note that once we have established the negation function of the new criterion, the intervals induced by this negation may be slightly different to the ones we have used to build the function. In Table 11 we can see the intervals obtained (with Eq.3.1) from the classical negation function. These values can be compared to the ones calculated from the positions of the clusters according to the ranking, which are the ones in the first column of Table 10.

New Negation	New Intervals
Neg (l_1) = {l_7}	I (l_1) = [0.0, 0.143]
Neg (l_2) = {l_6}	I (l_2) = [0.143, 0.286]
Neg (l_3) = {l_5}	I (l_3) = [0.286, 0.428]
Neg (l_4) = {l_4}	I (l_4) = [0.428, 0.571]
Neg (l_5) = {l_3}	I (l_5) = [0.571, 0.714]
Neg (l_6) = {l_2}	I (l_6) = [0.714, 0.857]
Neg (l_7) = {l_1}	I (l_7) = [0.857, 1.0]

Table 11. Negation function for the new criterion

5.2 Quality Measurement

In this section we define some quality measures that can be useful for the user in order to decide the reliability of the result. In many applications where fusion techniques are required, it is interesting to know to what extent the result of the process is acceptable. In addition, if the person that is executing the process is a non-specialised end user, the ignorance about the way the result is obtained often causes a mistrust feeling, and the consequent abandon of the system to continue doing the processes by hand.

For this reason, we have studied in detail the techniques applied at each stage of this new method. In the rest of the section we will define some quality measures that use the information available at the different stages.

5.2.1 The quality of the aggregation

Remember that our aggregation method is based on a hierarchical agglomerative clustering method. At each step of the process, we find out new clusters with a lower cohesion value. This cohesion value, h_α, is an upper threshold of the similarity values between any two alternatives in the class. So that, for any cluster α,

$$h_\alpha \geq d(a_i, a_j) \qquad \text{Eq. 5.3}$$

being (a_i,a_j) any pair of alternatives that belong to this cluster α.

At the end of the clustering, we can measure the global level of cohesion in the r clusters of the selected partition with Eq. 5.4. This is the first part of the *goodness* value of the aggregation stage (i.e. G_{Agg1}). According to this definition, $0 < G_{Agg1} \leq 1$, where 1 is the best value, which is obtained when the differences between the objects in the clusters are small.

$$G_{Agg1} = 1 - \frac{\sum_{i=1}^{r} h_i}{r} \qquad \text{Eq. 5.4}$$

Another interesting value to consider is the dimension of the clusters. The alternatives that belong to the same cluster cannot be distinguished by the user, because all of

them will receive the same linguistic term (i.e. category). Therefore, it is appropriate to have all clusters with similar number of objects. Entropy has been used in aggregation to evaluate dispersion of weights [Marichal,1999b]. Here, defining R_i with Eq. 5.6, we can consider the use of entropy [Shannon&Weaver,1949] to measure how much of the information is explained by each cluster. The maximum is achieved if all the clusters explain the same amount of information, that is, we have the same number of alternatives in each one.

$$G_{Agg2} = -\frac{1}{\ln r}\sum_{i=1}^{r} R_i \ln R_i \qquad \text{Eq. 5.5}$$

where r is the number of clusters in the ranking. R_i corresponds to the proportional cardinality of the i-th cluster with respect to the total number of alternatives, m, which can be seen has the probability that a random alternative a_k belongs to the cluster C_i.

$$R_i = \frac{cardinality(C_i)}{m} \qquad \text{Eq. 5.6}$$

If R_i is 0, the measure G_{Agg2} is undefined. However, this is not possible since we do not have empty clusters. We have that this quality measure (to be maximised) is limited as follows: $0 < G_{Agg2} \leq 1$.

If we are dealing with a multi-criteria selection problem, we can also inform the decision maker about the goodness of the first cluster in the ranking. In this case, it is interesting to have got a small cluster in the best position, in order to not have many alternatives indistinguishable, which may not be very helpful for the decision maker.

Having into account this last remark, we have defined the goodness of the aggregation stage subject to the dimension of the best cluster, C_{best}. That is, if the number of alternatives in this cluster is greater than the expected number of terms, we decrease the quality of this stage as it is shown in Eq. 5.7.

$$G_{Agg} = \begin{cases} \dfrac{G_{Agg1} + G_{Agg2}}{2} & \text{if } R_{C_{best}} \leq r \\ \dfrac{G_{Agg1} + G_{Agg2} - \dfrac{R_{C_{best}}}{2}}{2} & \text{if } R_{C_{best}} > r \end{cases} \qquad \text{Eq. 5.7}$$

5.2.2 The quality of the ranking

The evaluation of this stage depends on the characteristics of the decision problem, which will determine the use of the Principal Components Analysis or the use of a Similarity Function.

Ranking based on PCA

In the application of the PCA, some of the values obtained during the process are also useful to interpret the final result. Different measures are well defined in PCA literature [Jackson,1991]. We have studied the use of these measures to qualify the ranking of alternatives in a decision-making framework. Then, we have defined a goodness measure (Eq. 5.8) that takes into account the quality of the representation of the clusters by the first principal component, as well as, the agreement of the criteria (or experts) in relation to the first component.

$$G_{PCA} = \frac{\dfrac{\sum_{j=1}^{p} s \cdot CORR_1^2(x_j)}{p} + \dfrac{\sum_j QLT_1(j)}{\text{number of clusters}}}{2} \qquad \text{Eq. 5.8}$$

where s depends on the direction of the first component. If the x_j is positively correlated to the first component, $s = 1$. Otherwise, $s = -1$.

The best value of G_{PCA} is 1. The worst value is 0, which would correspond to a situation where the clusters were not well represented and the criteria did not agree with the first component.

In the numerator, the first addend is measuring the correlation of the variables, using equation Eq.4.20. The second addend is related to the quality of representation of the clusters, which is measured using Eq. 4.18, which can be rewritten as Eq. 5.9 for the case of a single component. If a cluster obtains a value near to 0, it means that it is bad represented by the first component, if the value is 1, the cluster is perfectly explained by the axis.

$$QLT_1(j) = \frac{z_i^{\,2}(j)}{d^2(j,G)} \qquad \text{Eq. 5.9}$$

being d the Euclidean distance between the alternative j and the centre of gravity (0 in our case, because we work with the correlations matrix).

In addition to the goodness measure, there are other interesting information values that should be given to the decision maker. The first one regards to the agreement between the experts or criteria analysed. As it has been explained in Chapter 4, the elements of the eigenvector are giving the contribution of each variable to the formation of its corresponding axis. Therefore, we can detect when a criterion differs from the social opinion, just looking into the values of the first eigenvector. If one of them is significantly smaller than the others, we can conclude that this criterion is significantly different from the consensus.

Another indicator is based on the quality of the projection of the clusters into the principal component using Eq.4.17. This allows the user to discover objects that can not be synthesised because the experts do not agree in their descriptions. In this situation, as the aggregation is not possible, this group of alternatives[7] is removed from the study taking an "unknown" label. This "unknown" label, in case it exists, is taken from the set of terms that experts provided; otherwise, a predefined linguistic label is used.

In Figure 20 we can see a graphical representation of the PCA result for the case of two variables. In this case, this quality value will detect those clusters that may have been positioned in a point that does not represent their real relation to the other ones (like cluster D). This will happen if the criteria give different opinions about alternatives in the class. So, with this method we can tell the user which alternatives are the conflicting ones.

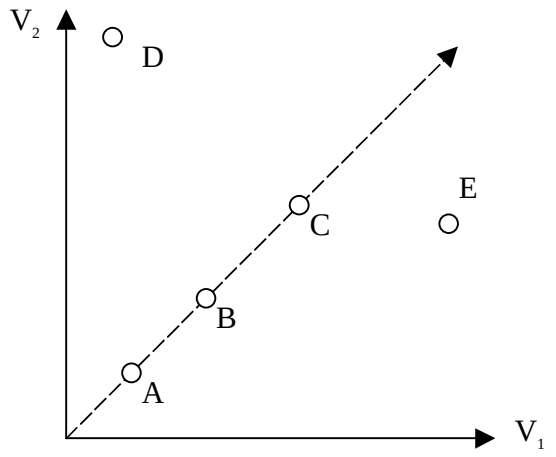

Figure 20. First principal component for a two-variable matrix

[7] Usually these groups are small.

If the two variables give the same value to the alternatives, the clusters formed will be positioned in an axis that will be in the middle of the two variables, like clusters A, B and C. Alternatives that are described with different values will not be in this line. For example, alternatives in D are bad (low value) according to criterion V_1 and good (high value) according to criterion V_2. On the other hand, alternatives in E are good for V_1 but only acceptable for V_2.

Similarity-based Ranking

When the ranking is based on the similarity to an *Ideal* alternative, other quality measures have to be designed. In this case, we can have two clusters with equal similarity values but that they are quite different from one to the other. That is, the distance to the *Ideal* is the same but due to different criteria. Then, we propose to give some additional information to the decision maker about in which criteria the alternatives do not have the desired value.

In addition to this information, we have defined a goodness measure based on the agreement between the criteria for each cluster.

$$G_{Sim} = 1 - \frac{\sum_{i=1}^{r} s_i}{r} \qquad \text{Eq. 5.10}$$

where the value that we are adding is based on the measurement of the dispersion (i.e. standard deviation, Eq. 5.11) of the values of the prototype of each cluster. The maximum value of this goodness measure is 1, which is given if all the clusters have dispersion equal to 0.

$$s_i = \sqrt{\frac{\sum_{j=1}^{r} \left(x_j - \overline{x_j}\right)^2}{r-1}} \qquad \text{Eq. 5.11}$$

5.2.3 The quality of the explanation stage

After the complete definition of the new criterion (i.e. c_{new}), we can evaluate the goodness of the new vocabulary and semantics. We should see if this new vocabulary could be misinterpreted. That is, if we are using some words that the decision maker will understand with a different meaning, we can induce him to an error. So, we propose to compare the new criterion with the ones in the initial decision matrix that

have some terms in common with it, $C_{common}=\{c_i, c_j, ..., c_k\}$. Obviously, the vocabulary from which we have generated the new one will be in this set.

We propose to use the distance d_v to measure the differences in the meaning of the terms in each vocabulary. The larger the differences (remember that the distance d_v gives values in [0,0.25]), the more confusing the result may be. Therefore, when the result is 1, we have a perfect correspondence between the terms in all the experts.

$$G_{Terms} = 1 - \frac{\sum_{c_i \in C_{common}} d_v(c_{new}, c_i) \Big/ 0.25}{cardinality(C_{common})} \qquad \text{Eq. 5.12}$$

Once we have given a linguistic term to each cluster, we evaluate their appropriateness. The position of each cluster before and after the explanation stage can be compared. The ranking stage provides a numerical position in [0,1] for each set of alternatives, z_{01}, which is used to select the most appropriate label from the vocabulary. After the explanation process, the position of some clusters may have changed due to the different meaning of the terms. That is, the intervals induced by the negation function may not have the cluster at the centre of the interval.

$$G_{Neg} = 1 - \frac{\sum_{j=1}^{r} \left| z_{01}(j) - (m(j) + M(j))/2 \right|}{r} \qquad \text{Eq. 5.13}$$

This measure compares the position of the alternatives before and after the introduction of the negation-based semantics. Being j the prototype of one cluster, $[m(j), M(j)]$ is the interval corresponding to the term assigned to this cluster using the new negation function.

Finally, we can define a global goodness measure for the whole ClusDM process.

$$G_{ClusDM} = \omega_1 G_{Agg} + \omega_2 G_{Rank} + \omega_3 G_{Terms} + \omega_4 G_{Neg} \qquad \text{Eq. 5.14}$$

where ω_i are the degrees of importance given to each step of the decision making process. For example, increasing ω_1 the user may indicate that obtaining very good and compact clusters is the best option, although it implies a change in the vocabularies and semantics. These weights must hold that $\sum \omega_i = 1$.

The ClusDM methodology pretends to be a useful recommender tool for decision makers. Our main aim has been to present the results using a linguistic vocabulary easily understandable by the user. The different goodness values can be used by the decision maker to have an idea of the quality of the different stages of the process. In addition, the overall goodness value can be also understood as the weight attached to the new preference criterion obtained.

Apart from that, our method is able to provide some additional information during the execution of the multiple criteria analysis. The importance of providing additional explanations of the results obtained with the decision model is a problem frequently considered in the Artificial Intelligence community [Papamichail,1998]. In our case, the information provided by ClusDM to the decision maker is the following:

- Which alternatives receive conflicting values from the different criteria. Those alternatives are identified during the ranking stage and do not appear in the final ranking given to the user. However, they should be presented to the decision maker in order to allow him to be aware of these special cases and perform an appropriate action if required.
- Which is the general degree of agreement (i.e. correlation) between the criteria or experts.
- Which criteria (i.e. experts) do not sufficiently agree with the result given by the system. However, this value is only available when the PCA ranking is possible.

In chapter 7, we will see some application examples where this additional information plays an important role.