4

Non-hierarchical classification

(An old activity in new shape)

Grouping different things, that is, *classification* is our fundamental intellectual activity: without this orientation in the surrounding world would be impossible. To mention an important example: the tool of communication, the language, is inseparable from classification, because naming things automatically implies some sort of categorization. The development of language is therefore simultaneous and interdependent with the refinement of classification[1]. Classification has a central organizing role in scientific disciplines that are characterized with an unusually high and troublesome diversity of the subject matter. Biology at the supraindividual level is a case in point; its history has always been entangled with the evolution of principles and methodology of classification.

The precise definition of classification in mathematics relies upon equivalence relations and sets. A classification is traditionally defined as a partition of objects into subsets (here: classes, groups or clusters) such that no object can belong to more than one class at the same time (the subsets are disjunct). This is still valid for the non-hierarchical or partititoning methods that are discussed in Subsections 4.1.1-4. However, the classical definition has undergone several modifications, allowing overlapping, fuzzy and hierarchical classifications – important extensions to be discussed afterwards.

It is an interesting property of many languages (for example, English and Hungarian), that the noun "classification" has double meaning: this word may refer to both the process of arranging things into groups, as well as to the result of that process. No confusions can arise from this, because it is usually obvious whether a series of operations or their end-product is meant in the given context. There is another source of ambiguity, however, which does cause problems. Since this is only the question of convention, clarification is unavoidable. In accordance with the literature of numerical taxonomy (e.g., Sneath & Sokal 1973), the process of classification will be understood here as a sequence of operations that create a *new set of*

1 Ability to classify is not only a human "privilege". Let us think, for example, of the animal world: recognition of members of the same species, distinguishing them from enemies and neutral species; and the distinction between edible, unedible and poisonous food are also classificatory activities.

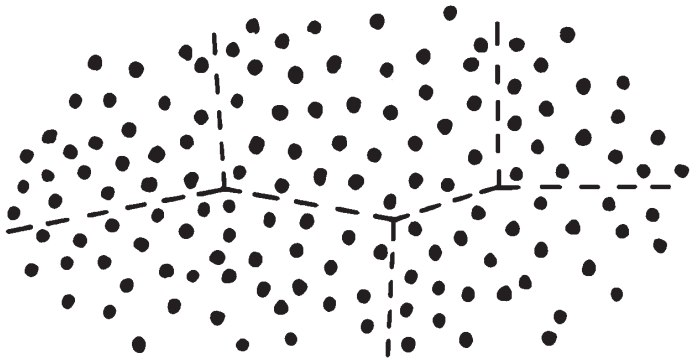

Figure 4.1. Division of trees in a relatively evenly dense forest stand into sectors in order to create an efficient system of access roads cannot be considered as a classification, because the structure of the forest is not influential in creating the classes.

classes, a completely new classification that did not exist before. To the contrary, in everyday speech as well as in certain fields of mathematics classification may often imply assignment of new objects into an appropriate group of an existing classification. This second activity is better distinguished under the term *identification* from the first. The distinction between these two objectives of classification becomes most obvious when we compare computerized algorithms designed to create new classifications with those made with the purpose of assigning objects into the "best" class. The first group of algorithms is often referred to as *cluster analysis* or simply *clustering*: its purpose is therefore the detection of groups in the data. The topic of identification, on the other hand, is closely related to the complex area of pattern recognition, and the above contrast is manifested in the pair *"unsupervised versus supervised pattern recognition"* (Therrien 1989).

Further requirement from the viewpoint of our discussion is that a classification should reflect inherent properties of the data by revealing group structure in the variable space. A simple subdivision of objects (*dissection*, Kendall 1966; Figure 4.1) into groups that do not reflect their distance or similarity relationships cannot be considered a 'true' classification. In this case external and practical, often completely data-independent criteria are forced upon the set of objects, such as subdividing a town into districts or dissecting a forest into management sectors. The dense and relatively evenly dispersed points of Figure 4.1 would be classified intuitively into a single group by anyone. Even spatial dispersion of objects and absence of gaps are not the only counter-indication of a classification; points randomly arranged in the multidimensional space also withstand any classificatory attempt, as shown in Figure 4.2a.

Then, under what circumstances could one talk about meaningful classifications? What kinds of inter-object relationships represent a minimum requirement of classifiability? Using Euclidean distance, or any other dissimilarity function defined in the previous chapter a classification can be characterized by two fundamental criteria: 1) internal cohesion of clusters, expressed using within-cluster distances, and 2) segregation or separation of clusters, measured by between-cluster distances (Gordon 1981). In an ideal case, both cohesion and separation

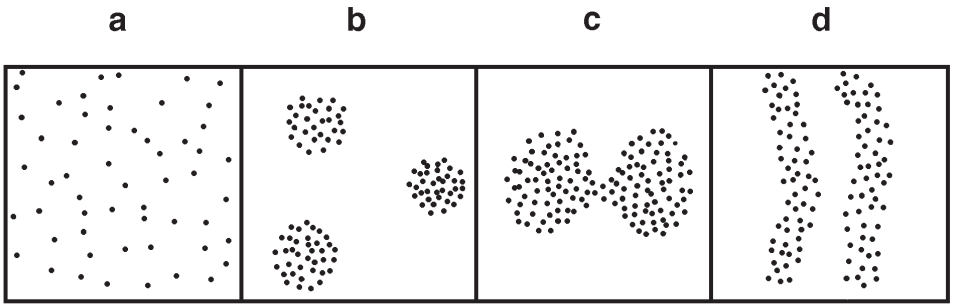

Figure 4.2. Special cases of clustering in a two-dimensional variable space. **a:** random arrangement, without real group structure, **b:** "ideal" case, with strong cohesion and clear separation of clusters, **c:** two classes with high cohesion and without segregation, **d:** elongated and well-separated point clouds with low internal cohesion.

are strong (Fig. 4.2b), a quite unequivocal situation in which all methods are expected to produce the same result. Such a grouping is usually easy to recognize in practice without use of any sophisticated analytical method, and the sole objective of clustering is therefore summarization and illustration of what is obvious anyway. More problems will occur if clusters are characterized by strong cohesion and weak separation (Fig. 4.2c). These are more or less detectable by the majority of methods, although the position of "transitional" or intermediate objects can be quite uncertain in the results. The other extreme is represented by the groups of Figure 4.2d, with clear-cut segregation and very low internal cohesion. Such clusters are the most difficult to recognize, as will be seen later. Of course, there are infinite ways of combining different levels of separation and cohesion, giving rise to problems we are usually concerned with in actual studies.

One would expect that numerical classification methods will try to optimize cohesion and separation of clusters simultaneously. Most procedures, however, do not treat these two complementing aspects of clustering equally: usually cohesion is measured directly and separation remains ignored. The algorithms of non-hierarchical classification are relatively simple, their introduction and understanding do not require deep knowledge of mathematics. It seems therefore justified that these methods are discussed prior to all other clustering procedures, which – on the other hand – does not mean that partitioning should be the first step in the complex methodological sequence of multivariate data exploration. In fact, the opposite is true in most cases: non-hierarchical clustering is used if other approaches have provided some information on data structure already.

4.1 Partitioning methods

The purpose of partitoning methods is to provide a conventional partition of m objects into k disjunct clusters (classes). In the literature, they are often referred to as *hard* or *crisp* partitioning, as opposed to fuzzy clustering (Section 4.3). By definition, an object can only belong to a single cluster and every cluster must have at least one element (otherwise there would be less than k clusters). The classificatory algorithm is usually iterative: an initial partition is im-

proved step by step in the analysis until no further improvement can be achieved. Defining an initial partition requires *a priori* specification of the number of classes. Assume that the "goodness" of the partition is measured by function *J*, whose value is to be decreased as much as possible in order to achieve further optimization of the result. Then, a very general partitoning algorithm will involve the following steps (Hartigan 1975, Therrien 1989):

1. Specify an initial partition into *k* clusters and compute the value of *J*.

2. Change the partition so as to decrease the value of *J* as much as possible, leaving *k* unchanged (that is, empty or new clusters cannot appear as a result of this change).

3. If no reduction of *J* is possible, the analysis stops with the actual partition as the final result. Otherwise we continue the iterations in step 2.

The different procedures vary in defining the goodness function *J* and in the operations allowed in step 2 to modify the actual partition. A fundamental property of the above general algorithm is that the result may often be only a *local optimum*, i.e., not necessarily the best classification of the given objects into *k* groups according to the *J* criterion. It is quite possible that from a different initial partition an even smaller value of *J* can be reached. By the same token, the analysis may be "trapped" in very bad solutions as well. This problem can be circumvented by performing the iterations from tens of different initial partitions and retaining the "best" result. We can never be 100% sure that we have reached the absolute (global) optimum; to obtain this all the possible partitions should be checked, which is an unaccomplishable task for large values of *m*.

The partition can be modified in Step 2 in two ways:

- We examine for each object separately how its relocation from the actual group to every other group influences the value of *J*. Objects causing a decrease of *J* are relocated into the group for which this decrease is the maximum. It is possible that many or even all the objects need to be relocated in a single step, and we just hope that the resulting new value of *J* will be smaller than the former (cf. Therrien 1989).
- The object for which maximum decrease of *J* is achieved is selected and then relocated to the new group. This strategy implies a monotone decrease of *J*, and is definitely slower than the previous method.

4.1.1 The k-means method

The classical example of partitioning procedures is the *k-means method* and its many variants (e.g., Forgy 1965, Jancey 1966, MacQueen 1967). The standard algorithm is as follows:

1. An initial, even arbitrary partition of objects into *k* groups is selected.

2. The centroid (i.e., the mean for all variables) is calculated for each cluster.

3. The Euclidean distance of each object from the respective centroid is determined. The goodness of the partition is measured as the sum of squared distances:

$$J = \sum_{h=1}^{k} \sum_{j \to A_h}^{m_h} \sum_{i=1}^{n} (x_{ij} - z_{ih})^2 , \qquad (4.1)$$

where z_{ih} is the centroid ("mean") of cluster A_h for variable i, m_h is the number of objects in cluster A_h (the second summation is according to these objects), and n is the number of variables. J is the sum of squares, which may also be calculated from the pairwise distances of objects for each cluster (Formula 3.106). If we encounter objects whose relocation decreases the value of J, then they are moved to the new group and we return to Step 2. Otherwise, when there are no relocations, the iterations stop.

A "slow" version of the above algorithm allows only a single relocation in each pass. A simplified version ignores the computation of J, and each object is reassigned into the group whose centroid is the closest. (One may expect that this second procedure produces the same result as the first, but this is not always the case, as will be seen in Subsection 4.1.3). The procedure applies when the data can be averaged and the Euclidean distance is meaningful (ordinal and nominal variables are excluded, for example). The stronger the cohesion of clusters, as indicated by the minimum value of J, the better the partition. The segregation between clusters is not measured directly in this analysis.

The procedure best recognizes compact, ball-shaped point swarms ("convex clusters") in the multidimensional space. Elongated clusters may be dissected, however, even though their separation is pronounced. To see the behaviour of the method under controlled circumstances, it is worth considering Figure 4.3 which shows the results of k-means clustering for typical combinations of marked features of cohesion and separation. (This figure will be used in subsequent chapters as well. For example, the two-dimensional arrangements of points will allow comparison of hierarchical clustering strategies.) The value of k was different for each case, either arbitrary (for unstructured data) or chosen so as to comply with our preconceived ideas about groups. The structure-free, randomly arranged set of points was simply dissected along a diagonal-like line (Fig. 4.3a), whereas the clusters with high cohesion and explicit separation were distinguished with ease (Fig. 4.3b). The border between the contiguous clusters of Figure 4.3c was drawn between objects 13 and 14 (note that upon a very small perturbation of the data, point 14 is moved to the other group, showing the relative instability of such classifications). As mentioned above, the k-means procedure cannot recognize elongated clusters (Fig. 4.3d), and is in a great "trouble" when an arched cloud surrounds a small, compact group: both are dissected (Fig. 4.3e). For the almost completely regular arrangement of points lacking any group structure we can only get a dissection of points for any value of k (in Figure 4.3f the "solution" for $k = 2$ is shown).

The initial partition may be specified in the following ways:

- Random partition. Group membership is defined by chance, and usually the analysis needs many more steps to converge into the final solution than when the starting configuration is not arbitrary (see below).
- A classification derived from the data by a different method (e.g., a hierarchical classification "cut" at a given level, see Chapter 5). It is likely that we shall have fewer iteration steps, but there is a chance to end up with a local optimum.
- The user predetermines k seed objects, and each of the other objects is classified together with the closest seed. This initialization is the most straightforward if we wish

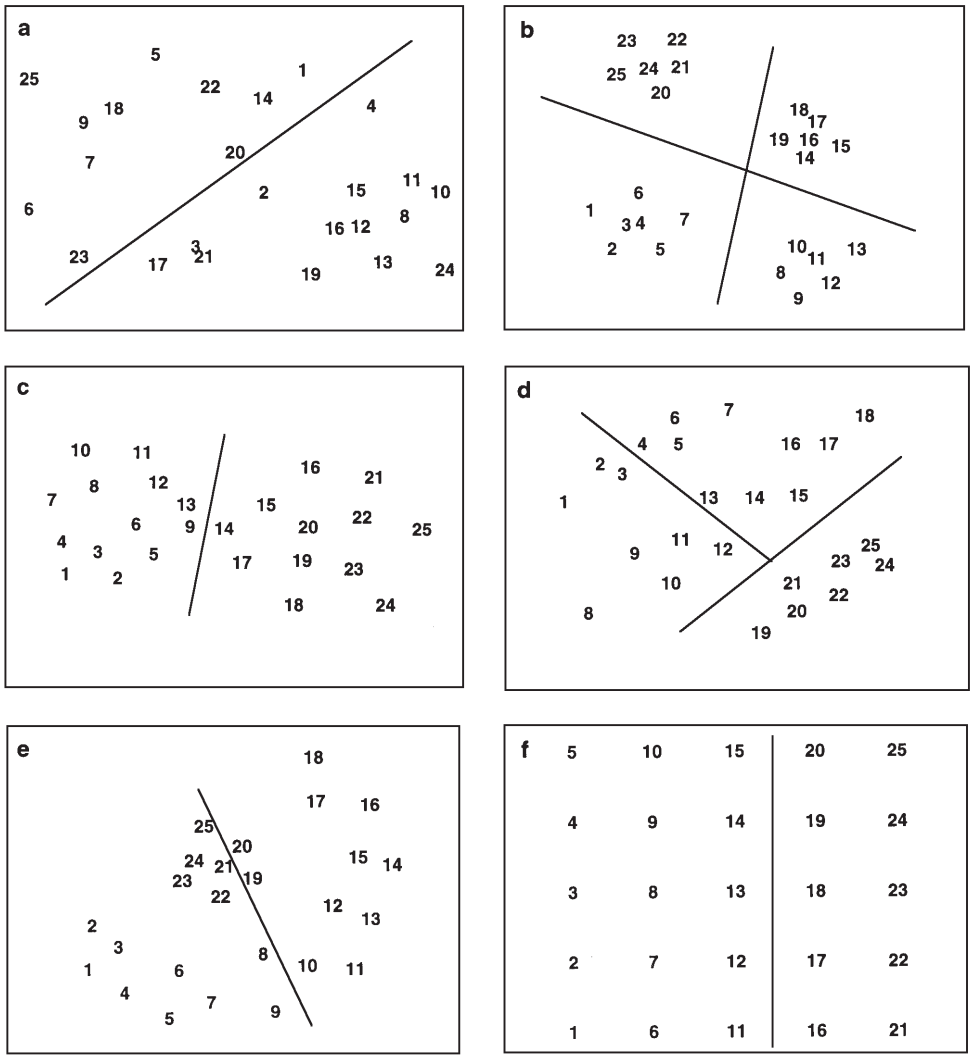

Figure 4.3. The results of *k*-means clustering for six illustrative two-dimensional arrangements with $m = 25$. Iterations were performed from 10 random initial partitions for each case, and the most optimal results are shown. The sum of squares are not reported here, because they are not commensurable with each other, even though m was constant. **a:** random configuration with $k = 2$, **b**: four "ideal" clusters, **c:** compact classes without segregation, **d:** three elongated point clouds illustrating low cohesion and fair separation, **e**: a small compact group surrounded by an arched point cloud, with apparent separation of clusters, **f:** an almost completely regular arrangement, partitioned at $k = 2$. The horizontal and vertical coordinates of points are given in Table A3 in Appendix A.

to find a partition that best fits a given set of typical objects. The possibility that the iterations stop with a local optimum also exists.

- The seed objects are selected randomly, so we start essentially with a random partition.
- The starting k seed objects are those falling furthermost apart in the n-dimensional space. The first seed is the object most distant from the centroid of all the others, the second seed object is the most distant one from the first, the third seed object has a maximum sum of distances from the previous two, and so on, up to k. This mode of initialization may be sensitive to the presence of "atypical" elements (hardly classifiable outliers) in the data.
- We start from an optimal partition containing k-1 classes such that the seed of the kth cluster is the object with the highest distance from its own centroid (Hartigan 1975). The multiple partitioning technique discussed in Subsection 4.1.3 utilizes this initialization in each main step.

Other possibilities to begin with are discussed in Anderberg (1973: 157-160).

A flexible modification of the k-means procedure is the ISODATA method (Ball & Hall 1965), in which k is not fixed so rigorously as above (the number of classes is allowed to change under certain circumstances), and cluster segregation is also measured directly. The price to be paid is that specification of further parameters is required, introducing more arbitrary elements into the analysis. To perform ISODATA, one needs to specify the minimum cluster size (smaller classes are disregarded so k is deacreased). In addition, the most preferred value of k is to be given in advance. If this number is significantly exceeded during iterations, then the algorithm will try to amalgamate the closest clusters. In the opposite case, when the number of classes tends to be too small, the most "heterogeneous" classes will be split. Thresholds for amalgamation and split are also arbitrary values, defined in form of minimum separation and maximum within-class sum of squared deviations. Due to the relatively high number of input parameters, the algorithm of ISODATA is fairly complicated and is not detailed here (for more information, see Therrien 1989, pp. 219-222).

4.1.2 A general, coefficient-independent partitoning method

The k-means procedure, as we have seen, has certain limitations, for example, the operation of averaging raw data must be meaningful. An additional shortcoming is that the sum of squares measures only within-cluster cohesion, and cluster separation is not considered directly. A more generally applicable partitioning precedure is obtained by adapting the following redefinition of the J criterion. Let AVG_b be the mean (average) of all within-cluster dissimilarities, and AVG_e be the mean dissimilarity of object pairs that do not belong to the same cluster. Formula 3.111 already gave us the average for one cluster; its extension to k clusters yields the following:

$$AVG_b = \sum_{i=1}^{k} \sum_{g \to A_i} \sum_{h \to A_i} DIS_{gh} \Big/ \sum_{i=1}^{k} m_i (m_i - 1) \Big/ 2, \qquad (4.2)$$

whereas the average of between-cluster dissimilarities is calculated by a more "frightening" formula:

$$AVG_e = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \sum_{g \to A_i} \sum_{h \to A_j} DIS_{gh} \Big/ \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} m_i m_j \,. \tag{4.3}$$

AVG_b is thus a measure of internal cohesion, whereas AVG_e expresses the overall segregation among clusters. *DIS* can in fact be any of the dissimilarity functions discussed in Chapter 3, for example, those developed for ordinal or mixed data apply with no problem. The function measuring the goodness of a partition is defined as the ratio of cohesion and segregation, and is denoted by *G*:

$$G = \frac{AVG_b}{AVG_e} \,. \tag{4.4}$$

That is, the larger the external dissimilarities compared to the internal ones, the better the partition[2]. For a completely random classification, *G* will take a value around 1 ($G>1$ corresponds to the extremely "bad" situation with within-cluster dissimilarities larger than the between-cluster measures). As the internal dissimilarities decrease and external values increase, the value of *G* will approximate zero. *G* can be considered as a very general measure of the goodness of partitions, which is independent of the coefficient of dissimilarity used. Its further advantage is that classifications based on different coefficients become directly comparable, since *G* is insensitive to the range of the coefficients.

The partitioning algorithm of the *k*-means method can also be used to optimize *G*: in each iteration step the object giving maximum reduction of *G* is relocated. As initial classifications we can consider only those not requiring calculation of cluster centroids.

Figure 4.4 illustrates the performance of the method for the sample data, using Euclidean distance (chosen to allow comparison with the results of *k*-means clustering). In cases **a, b** and **d** the results are the same as for *k*-means clustering, so these are not shown again. In case **c** the difference is caused by a single object: 14 belongs to the left group, showing that transitional objects are difficult to classify unambiguously. For case **e**, a better result was obtained because at least the compact central group remained intact. In the regular case (**f**), the result could not be other than an arbitrary dissection of points.

What *k*-means clustering did not allow becomes possible now: the *G* values are directly comparable so we may evaluate the relative goodness of all the six classifications. The best value resulted for case **b**, of course ($G = 0.23$), a score much lower than for the next best classification, the two non-separated clusters of case **c** ($G = 0.48$). The classifiability criterion value is even higher for the other examples, because cluster cohesion is diminished (for **d,** $G = 0.52$, for **e** we have $G = 0.56$). It is striking that the random case (**a**) produces the same value, for the first two decimal digits, as case **e** ($G = 0.56$). As one would expect, the least classifiable set of objects is represented in **f**, producing the highest criterion value, $G = 0.64$.

There are mathematically more sophisticated methods considering "inner" and "outer" distances simultaneously. However, these apply to Euclidean models only. Several authors proposed to decompose the matrix of sum of squared deviations into two parts, the "between-class", **B** and the "within-class", **W** components. Then, a partition is sought for which the largest eigenvalue (Roy criterion) or the trace (Hotelling criterion, see Anderberg 1973) of matrix $\mathbf{W}^{-1}\mathbf{B}$ is the maximum. As Gordon (1981) points out, use of these criteria favours clusters of approximately equal size. Other procedures could also be mentioned, but they assume stringent conditions (e.g., multivariate normality) which rarely satisfy.

2 Ratio 4.4 has long been used to measure the goodness of classifications *a posteriori* (Hartigan 1975), but as a clustering criterion it was first used by Podani (1989a), in hierarchical clustering as well (see Subsection 5.2.4).

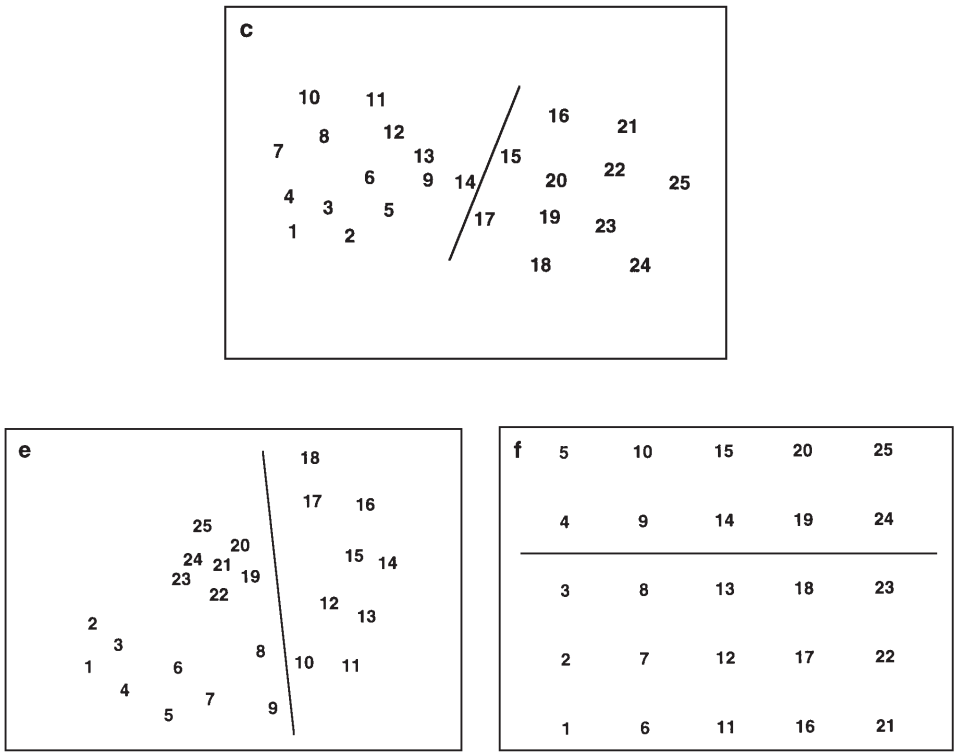

Figure 4.4. Results of the coefficient-independent partitoning method for the sample data. Partitions differing from the result of *k*-means clustering (see Fig. 4.3) are displayed only.

4.1.3 Multiple partitioning

A priori specification of the number of clusters may be avoided by a recursive application of partitioning methods that require choice of this value beforehand (e.g., *k*-means clustering). This complex strategy represents a transition towards hierarchical classification (Chapter 5). The set of objects is first subdivided into two groups, then the centroid of a new class is determined in order to shift to three clusters, and so on until the pre-specified maximum number of clusters, k_{max}, is reached. The name of the procedure was proposed by André (1988). A brief description of the underlying algorithm is as follows:

1. All objects belong to the same class at the outset. The centroid is determined and we identify the object which falls furthest apart from that centroid. This is considered as the seed point of a new cluster. Thus $k = 2$.

2. This step is practically a complete *k*-means clustering analysis: every object is relocated to the nearest cluster based on its distance from the centroids. Then, new centroids need to be calculated, which will probably necessitate further relocations. The iterations continue as long as the clusters are changed. Otherwise, iterations stop so that every object belongs to the closest cluster, and we output the result for *k* clusters.

3. k is increased by one. If it is no larger than k_{max} then the object falling farthest apart from its own centroid is identified and then considered as the seed point of the new class. Return to step 2. If $k > k_{max}$ multiple partitioning stops.

The above algorithm was tested on the sample data sets. Note that in this case it is not the sum of squares we attempt to minimize directly, which is the only source of potential differences from the results of k-means clustering. In case **c**, for example, object 14 is put into the left group (similarly to the coefficient-independent method, Fig. 4.4c). However, upon thorough scrutiny of the data we may realize that the same object could also be moved into the right-hand group (as in k-means clustering), because this new position changes the two centroids such that object 14 is in an optimal group again. So, if distances from centroids are measured, two or more alternative solutions may emerge. The chance for such ambiguity is much lower when sum of squares is minimized: in this example object 14 is better positioned into the right group (go back to Figure 4.3c). We conclude therefore that the two variants of the same strategy responded differently to the presence of a controversial object.

The result of multiple partitioning is a hierarchical classification if the new cluster obtained for k+1 comes about by subdividing some cluster we had before for k, and this is true of all numbers of clusters used in the analysis. This is exactly what happened for case **b**; for three values of k the following partitions were produced:

$k = 2$ {1 - 19} {20 - 25}
$k = 3$ {1 - 7} {8 - 19} {20 - 25}
$k = 4$ {1 - 7} {8 - 13} {14 - 19} {20 - 25}

(for $k = 4$ the result is the same as in Fig. 4.3b). Contrarywise, in example **d** two values of k lead to classes that cannot be nested:

$k = 2$ {1 - 11, 13} {12, 14 - 25}
$k = 3$ {1 - 7, 13} {8 - 12, 19} {14 - 18, 20 - 25}

(see Figure 4.5**d** for k=3, where the analysis was stopped). A possible explanation of such ambiguities is that classifiability of objects is in question for the given values of k (André 1988), as is the case for example **d**: the elongated clusters cannot be recognized by this classification strategy. The result of multiple partitioning differs from the previous partitions for cases **e** and **f** as well.

In the above algorithm, successive subdivisions of some cluster produced the refined partition (see Chapter 5 for the divisive hierarchical methods). We can of course proceed in the opposite direction: the objects are first arranged into k_{max} clusters. When convergence is reached, the two clusters whose centroids are the nearest are amalgamated. This partition into k_{max}–1 clusters is impoved by relocations, and the analysis continues with further amalgamations (e.g., Beale 1969, Wishart 1978). This procedure has properties similar to the agglomerative hierarchical clustering methods.

4.1.4 Quick partitioning of large sets of objects

In computerized realizations of methods discussed thus far the maximum number of objects is strongly determined by the amount of available memory. For a standard PC with 640kbytes of RAM this usually means that we cannot classify more than 3-400 objects. This may be a serious limitation if we consider that in certain fields of biology we can easily have several magnitudes more, say two-hundred thousand objects to be classified. Landsat photographs, for example, are usually composed of very large numbers of units (pixels), and the recognition and analysis of patterns in these pictures would be impossible without classification. Even though memory limitations could be alleviated by continuous reading operations on magnetic

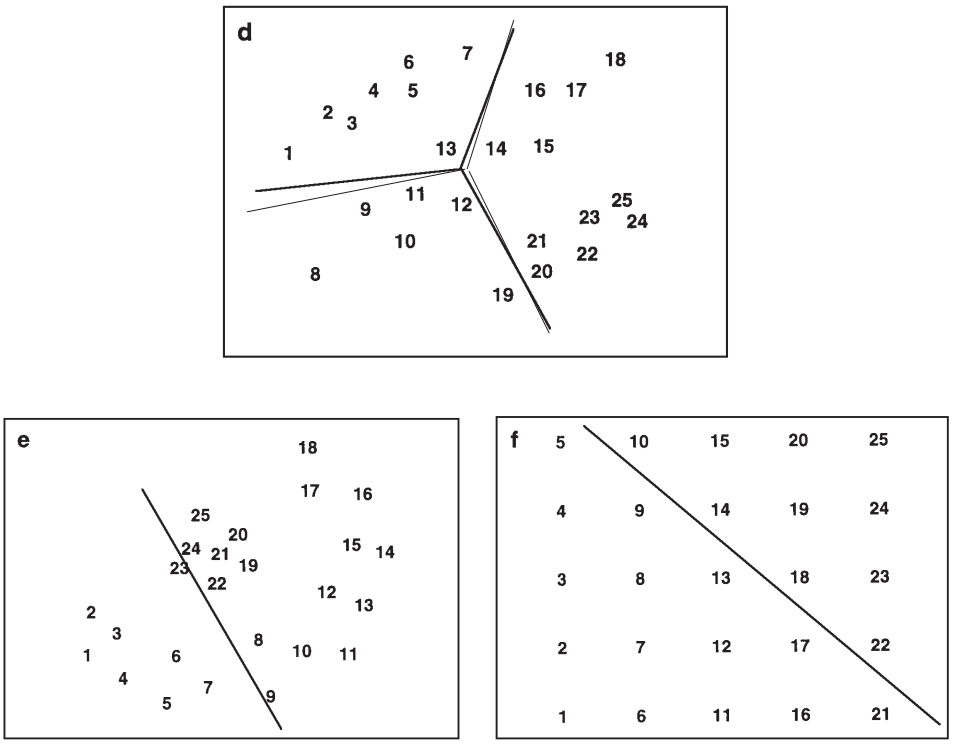

Figure 4.5. Results of multiple partitioning for the sample data. In each step, relocation was based on the distances from cluster centroids, rather than on within-cluster sum of squares. For cases **a** and **b**, the final partition is the same as in Fig. 4.3, for case **c** the result is the same as in Fig. 4.4.

storage media, the traditional methods would require extremely long running time. Procedures facilitating relatively quick classification of very large sata sets are best suited to such problems. Increasing speed has its own consequences, of course, because there will be a very small chance that quick clustering produces globally optimal partitions. Very often, the result depends greatly upon the order in which the objects are presented for analysis. The advantage is, on the other hand, that very large sets are reduced to a few hundred clusters, each of which can be represented by one of its objects in further analyses – not only partitioning but also any procedures that follow in the subsequent chapters. Note that hierarchical clustering and ordination can be even more constrained by available memory than non-hierarchical clustering.

The fundamental strategy of quick clustering is that the data are read from disk object by object, so that the complete data matrix need not be stored in RAM. The basic type of such procedures is the *leader algorithm* (Hartigan 1975), which runs through all the objects in a single pass according to the following steps:

1. A distance or dissimilarity function (*DIS*) is selected in accordance with the type of data. Most of the functions treated in Chapter 3 are suitable for this purpose. In addition, we specify a threshold T for *DIS*, which pre-determines the size (more precisely, the "diameter") of quick classes to be detected during the analysis.

2. Object 1 is dedicated to be the leader object of cluster 1. Let j be used as the index of the objects, that is $j = 1...m$. The number of clusters is $k = 1$ and j is also equal to 1.

3. Increase j by 1. If $j>m$, the analysis stops.

4. Evaluate all existing clusters from 1 to k. If the distance of object j from the leader object of a cluster is smaller than T, then the object is assigned to the first such cluster and the analysis returns to step 3.

5. If object j has distances larger than T from all leader objects, this object is declared to be the leader of a new cluster, and k is increased by 1. Return to step 3.

The advantage of the above method lies in its high speed, but the dependence of the final result upon the intial sequence of objects is undesirable (object 1 is always leader). The latter problem is solved if the leader element is selected at random from the set of unclassified objects. This will increase computing time, however, because we must run through the data more than once: as many times as the final number of clusters. Another difficulty is that the firstly created clusters will be generally larger than the subsequent ones. A possible reason is that some objects may be "trapped" in the hollows left among the clusters already defined, as illustrated by Figure 4.6 for two dimensions. If we introduce a second threshold, T_2, which is slightly larger than the first, the undesirably small clusters may be amalgamated with the near-

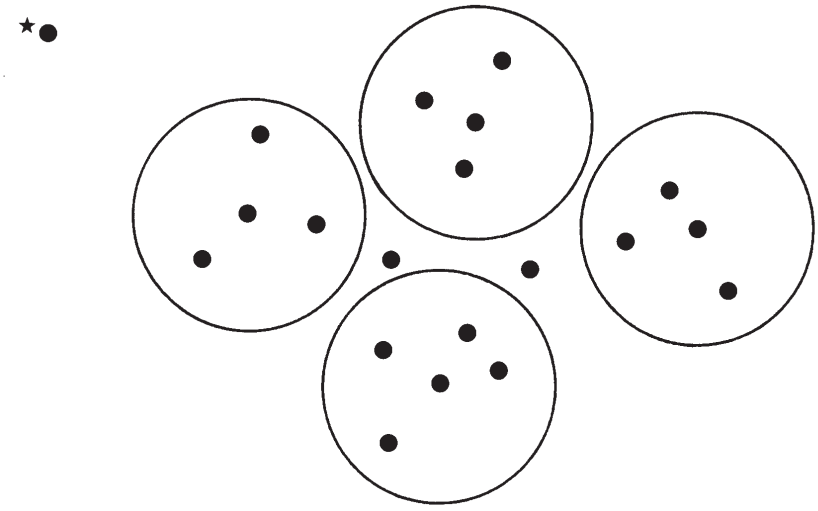

Figure 4.6. A disadvantage of quick partitioning is that some objects may only serve as seed points of singleton clusters, giving the false impression of being outliers. A true outlier, falling far away from all clusters, is found in the top left corner and is marked by an asterisk.

est large cluster (COMPCLUS method, Gauch 1979, 1980). Clusters that remain still very small may be rightly considered as *outliers*, whose classification is difficult if not impossible (such as the object marked with an asterisk in Fig. 4.6). The user must play a little bit with the actual value of T within a certain range which is unknown *a priori*: if T is defined to be too small, we shall have too many clusters, adding very little to our kowledge on group structure. For very large values of T, quick partitioning may end up with a single cluster containing all the objects. Several trials are needed therefore with various values of T (and of T_2 for COMPCLUS) in order to find a balanced and sufficiently small number of clusters.

The algorithm of CLUSLA (Louppen & van der Maarel 1979) combines the above rapid procedure with iterative relocations: objects falling closer to a new leader are relocated. A transitional procedure between the leader algorithm and multiple partitioning may also serve the purpose of quick clustering (Hartigan 1975). In this, the algorithm of multiple partitioning is modified so that no relocations are allowed in any step. The first leader object may be the one closest to the centroid of all data, the second may be the object falling the farthest from the first. All the other objects are assigned to the closest leader object. In the next step, for three clusters, we find the object which has the largest distance from its own leader, and designate this as the leader of group three. The analysis proceeds in similar manner up to the maxmum value of k.

4.2 Overlapping clusters

Figures 4.3c and 4.4c illustrate situations when cluster membership is not clear cut: object 14 could be classified into either of the two, otherwise quite apparent groups. As we have seen already, use of distance from the centroid as the clustering criterion equally favours the alternative assignments. Therefore, it seems reasonable to get rid of the axiomatic limitation of crisp partitions, that is the mandatory disjunctness of clusters, and declare this intermediate object to be the member of both clusters! In this way an *overlapping* classification is created. Methods designed for this purpose were proposed first by Jardine & Sibson (1968) under the heading "B_k clustering", in order to achieve a more realistic classification of objects for which the solution of hard partitioning may not be unique. According to their definition, a series of classifications may be obtained for a set of objects for $k = 1, 2, 3...$, in which any two clusters may overlap with each other in at most k–1 objects. Hard partitions are therefore B_1 classifications, whereas the above example with object 14 assigned to both clusters represents a B_2 classification. (This k parameter should not be confused with the number of clusters required in k-means clustering. The alphabet is apparently not long enough, because the literature insists on using the letter k for both families of methods!)

The algorithm of B_k clustering is more intricate than the most complex hard partitioning procedures (see Ling 1972 and Rohlf 1975b) and we have space only for a brief summary of the main steps. All objects are considered as vertices of a graph in which two vertices are connected if the similarity of the corresponding objects exceeds a threshold T. Then, the so-called maximum complete subgraphs are searched for; these are subgraphs containing the maximum number of objects for which edges exist in all possible pairs. Of the subgraphs those are selected to represent overlapping clusters which do not intersect in more than k-1 vertices (i.e., they agree in k–1 vertices at most). Such a situation is exemplified for $k = 3$ in Figure 4.7. The search may be continued for descreasing values of T, yielding an overlapping hierarchical classification (cf. next chapter). Of course, the value of k may also be changed systematically, clearly illustrating the usual dilemma of data analysts: many alternative B_k classifications ex-

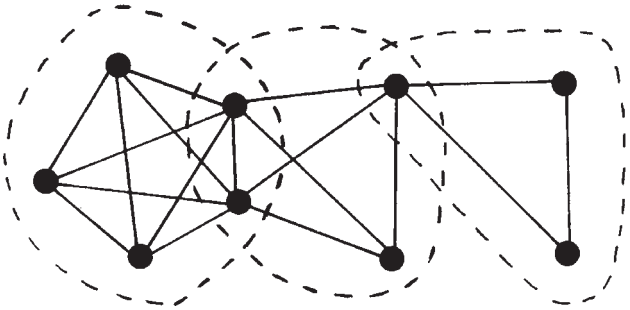

Figure 4.7. Illustrating Jardine-Sibson's B_k clustering with a graph. Each of the three complete subgraphs represents a cluster, two of them overlapping at the level of $k = 3$, that is, in maximum two objects.

ist for the same set of objects. In addition, visual display of results is quite cumbersome for large m, so that most authors suggest not to use the method as a standard routine of data exploration. Instead, a relatively more recent group of procedures, the fuzzy clustering methods discussed right below are recommended.

4.3 Fuzzy clustering

In classification studies, we are often faced with ambiguous situations when certain objects cannot be assigned to clusters unequivocally. Figure 4.3c has illustrated this already, and we examined a possibility to circumvent the problem in the previous section. We pointed out, however, that overlapping classifications are difficult to interpret for many clusters and objects, and the results cannot be displayed without other analytical tools, such as ordinations. It is to be realized, therefore, that discrete methods are not necessarily the proper choice if the cluster structure in the data is unclear. The results are more interpretable and – what is more important – they give a more realistic picture on data structures if the criteria for cluster memberships are much more relaxed than in B_k clustering. Zadeh's (1965) revolutionary ideas on fuzzy sets will give us the starting impetus. Contrary to classical set theory it is allowed that an object belongs to several subsets such that the degree of belonging may differ considerably. In fuzzy classifications, the membership of each object in each cluster is expressed by weights with the constraint that the sum of weights should equal exactly 1 for each object. (This condition may remind us of probabilities, because the sum of probabilities is also 1 in a complete system of events. The analogy is remote, however, because weights reflect the affinity of objects to clusters, rather than probabilities of belonging.) A classification is represented by a matrix whose rows are the objects and the columns are clusters, and each score is the weight according to:

$$\mathbf{U} = \{ u_{jc} \}, \quad j = 1,\ldots,m, \quad c = 1,\ldots,k, \quad \text{and}$$

$$\Sigma_c u_{jc} = 1, \text{ for all } j \tag{4.5}$$

(the number of clusters, k, is predefined by the investigator, just like for k-means clustering). The question is how to derive such a matrix of weights?

The simplest and most widely applied fuzzy clustering method is the fuzzy *c*-means procedure (Bezdek 1981, 1987, Marsili-Libelli 1989). In this, the so-called *fuzzy sum of squares* is minimized:

$$FSSQ = \sum_{j=1}^{m} \sum_{c=1}^{k} u_{jc}^{f} \, d_{jc}^{2}, \tag{4.6}$$

where

$$d_{jc}^{2} = \sum_{i=1}^{n} (x_{ij} - v_{ic})^{2} \tag{4.7}$$

is the distance between object *j* and the cetroid of cluster *c,* and f (>1) is the *coefficient of fuzziness*. The larger the value of *f,* the fuzzier the partition, that is the less crisp the boundary between clusters. The value of f is to be specied in advance, not only *k*, requiring arbitrary decision but at the same time offering yet another possibility to analyze our data more flexibly by successively changing the parameters of the analysis.

The centroid of clusters is determined according to:

$$v_{ic} = \frac{\sum_{j=1}^{m} u_{jc}^{f} \, x_{ij}}{\sum_{j=1}^{m} u_{jc}^{f}}. \tag{4.8}$$

The main steps of the analysis are summarized in the following algorithm:

1. The initial partition is specified by selecting the *k* furthermost seed objects. Other initialization, such as those mentioned in Subsection 4.1.1, may also be conceived.

2. Starting cluster membership weights for each object *j* are calculated such that they are proportional to the distances from the centroid, with condition (4.5) obeyed.

3. New weights are computed using the following equation:

$$u'_{jc} = \frac{1}{\sum_{h=1}^{k} \left(\frac{d_{jc}}{d_{jh}} \right)^{2/(f-1)}}, \tag{4.9}$$

If $d_{jc} = 0$, that is the centroid of cluster *c* coincides with object *j*, then $u_{jc} = 1$ and all other weights are zero.

4. Calculate new centroids using Formula 4.8.

5. The analysis stops if the difference between the values in cycle *q*, and those obtained in the previous cycle *q–1* does not exceed a user-specified threshold ε:

$$\varepsilon = \max_{j} \max_{c} |u_{jc}^{(q)} - u_{jc}^{(q-1)}|. \tag{4.10}$$

Table 4.1. Results of fuzzy *c*-means clustering for the points of Fig. 4.3c with $k = 2$ and $f = 1.5$.

Object	Cluster 1	Cluster 2
1	.9839	.0161
2	.9819	.0181
3	.9973	.0027
4	.9948	.0052
5	.9901	.0099
6	.9556	.0444
7	.9940	.0060
8	.9979	.0021
9	.9536	.0464
10	.9810	.0190
11	.9723	.0277
12	.9915	.0085
13	.9676	.0324
14	**.5050**	**.4950**
15	.0804	.9196
16	.0547	.9453
17	.1951	.8049
18	.0460	.9540
19	.0023	.9977
20	.0003	.9997
21	.0173	.9827
22	.0012	.9988
23	.0018	.9982
24	.0190	.9810
25	.0104	.9896

The condition to arrest the iterations is thus based on the maximum change achieved between two successive steps. If ε is exceeded, we return to step 3. Otherwise, the weights determined in the last cycle are output as the final result.

To illustrate the procedure, the set of points in Figure 4.3c is classified using the following parameters: $k = 2$, $f = 1.5$ and $\varepsilon = 0.01$. This threshold was reached very early, after the fourth iteration. In the result most objects have strong affinity to one cluster, as depicted by the high number of weights larger than 0.9 (Table 4.1). Object 14 causing so much trouble for us in hard clustering has almost identical weights for the two clusters (boldface in the table), indicating its transitional position.

In evaluating fuzzy classifications, simple inspection of cluster membership weights is usually insufficient. There are several possibilities to detect the "optimum" number of clusters, for example. First of all, Bezdek's (1974, 1981) *coefficient of partition* is to be mentioned:

$$F_k = \sum_{j=1}^{m} \sum_{c=1}^{k} (u_{jc})^2 \,/\, m, \tag{4.11}$$

which ranges from $1/k$ to 1. For different values of k, this function takes relative maxima where k equals the optimum. The range of F_c depends on k, however, a problem easily solved by extending it to [0,1] using the coefficient given by

$$F'_k = \frac{kF_k - 1}{k - 1}. \tag{4.12}$$

Clustering efficiency may also be measured by Dunn's *partiton entropy:*

$$H = -\frac{1}{m} \sum_{j=1}^{m} \sum_{c=1}^{k} u_{jc} \log u_{jc}, \tag{4.13}$$

Its standardized form is:

$$H' = \frac{H}{(1 - k \,/\, m).}. \tag{4.14}$$

For different values of k, the minimum of Function 4.14 is found, thus facilitating the detection of the "optimal" number of clusters in the data.

> For the case of Figure 4.3b with four obvious clusters we run fuzzy c-means clustering with k = 2, 3, 4, 5 and 6, continuously increasing the value of f as well (f = 1.2; 1.5; 2.0; 2.5; and 3.0). The relationship between the number of clusters and the partition coefficient and partition entropy is shown in Figure 4.8. As expected, the partition coefficient reaches the maximum for k = 4 irrespective of the value of f (although the maximum is less pronounced with f = 1.2). Interestingly, the curve of partiton entropy is influenced by f as well: for very fuzzy classifications (f > 2) the minimum was obtained for k = 2, and the expected result yields for the less fuzzy classifications. This example suggests that partition coefficient is a more reliable indicator of the number of clusters than partition entropy.

The so-called *coefficient of separation* is closely related to the partition coefficient:

$$\Omega = \sum_{j=1}^{m} \sum_{c=1}^{k} u_{jc}^2 \tag{4.15}$$

Its value ranges between m/k and m. The closer it is to m, the harder the partition because one weight for each object approximates unity. In an extreme situation, all objects reach the weight of 1 for one cluster, showing that the hard partitions are in fact special cases of the more general family of fuzzy partitions.

The pairwise separation of clusters b and c may be expressed using the distances betwen their fuzzy centroids:

$$\delta_{bc} = \frac{\sum_{i=1}^{n} (v_{ib} - v_{ic})^2}{\max_j (u_{jb}\, d_{jb}) + \max_j (u_{jc}\, d_{jc})}. \tag{4.16}$$

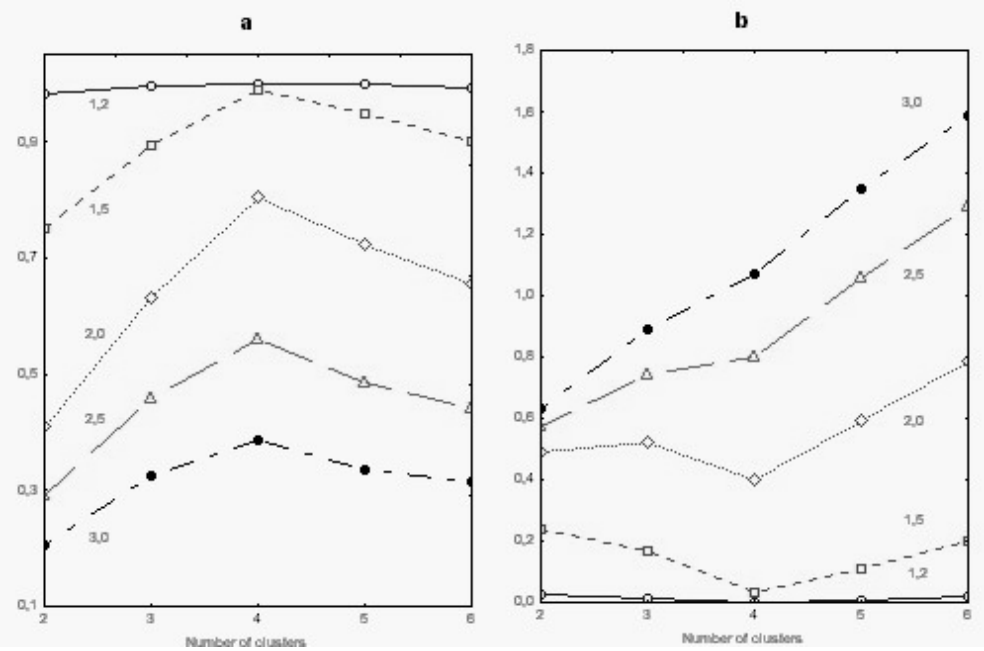

Figure 4.8. The change of **a**) the partition coefficient (4.12) and **b**) the partition entropy (4.14) over the number of clusters for different values of *f* in the fuzzy clustering of points of Figure 4.3b.

The tabulated result of fuzzy clustering may be visualized graphically in a coordinate system with axes representing clusters and coordinates corresponding to the membership weights. Since the sum of weights is 1 for each object, the points fall onto a hyperplane, similarly to raw data points standardized by the total. (In Figure 2.9c, the full symbols drawn onto the "diagonal" may illustrate a fuzzy clustering for two groups. The points in the fuzzy clustering of Table 4.1 also fall to a diagonal line, most of them at the endpoints whereas point 14 very close to the middle – but we feel that this is far too obvious making any illustration unnecessary.) On the plane only two clusters can be shown at a time, of course, yet such an ordination-like diagram (Chapter 7) does help us to interpret the results. In fact, for three dimensions there is an alternative possibility for display, the *ternary plot*. The points are on an equilateral triangle, which may be shown in two dimensions. The vertices correspond to the clusters, and the closer a point to a vertex, the less equivocal its group membership. If all three weights happen to be 0.33 for an object, it will get into the centroid of the triangle, showing the maximally uncertain class membership of that object. If two weights are 0.5 and the third is zero, the point falls onto the normal bisector of the respective side of the triangle.

The ternary plot is illustrated by the fuzzy clustering of the raw *Iris* data (Table A2) in Figure 4.9, for two values of the coefficient of fuzziness. The analysis assumes three groups, because we distinguish *a priori* between three different taxa. As seen in the diagram, for low values of *f* (*f* = 1.25) the separation of the species is high, since most points coincide, although

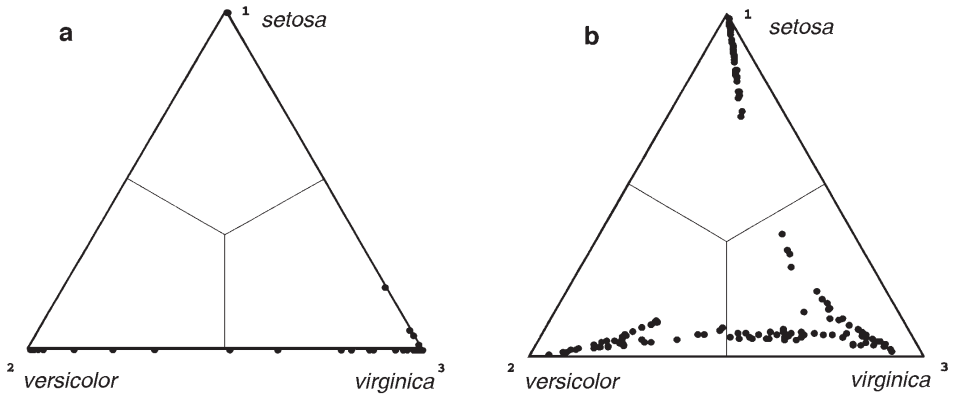

Figure 4.9. Ternary plot illustrating fuzzy clustering of three *Iris* species (Table A2) with two values of the coefficient of fuzziness, **a:** $f = 1.25$; **b**: $f = 2.5$.

an initial transitional series between *Iris versicolor* and *virginica* appears (Fig. 4.9a). For a higher value ($f = 2.5$), this series becomes more continuous, and some individuals will take intermediate positions between *setosa* and *virginica* as well. Figure 4.9b can be interpreted by saying that some *virginica* individuals have certain affinity towards *versicolor*, whereas the others have higher resemblance to *setosa*. Note that ternary plots can be used in any case when our objects are described in terms of three variables and the total for each variable is 1 (i.e., we have standardized the data by the total).

4.4 Literature overview

A classical description and deep characterization of partitioning methods are presented by Anderberg (1973) and Hartigan (1975). The book by Späth (1980) devotes even more space to non-hierarchical classification with ample examples. Everitt (1980) is another useful reference; its special merit being that attention is called to unresolved problems (a more recent edition of the book is from 1993). Mirkin (1996) is very informative for those wishing to get insight into a more mathematical approach to partitoning. Other books on clustering are less useful in this regard, because they place emphasis upon hierarchical methods (for example, Clifford & Stephenson 1975 and Gordon 1981). As far as biological applications are concerned, partitions should be mentioned mostly in the context of ecology and phytosociology (see e.g., Orlóci 1978, André 1988, Jancey 1974). Gauch (1982) attributes primary role to non-hierarchical classifications in quick clustering of large data sets and gives many useful references for further reading. Of the non-standard methods fuzzy clustering is best-treated in Bezdek (1981, 1987), although Equihua (1990) and Marsili-Libelli (1989) may also be consulted for ecological applications. The relationship between non-hierarchical classification and pattern recognition is revealed in much detail by Therrien (1989).

4.4.1 Computer program packages

Program lists of various non-hierarchical methods are found in several books, especially in those published about fifteen or more years ago, such as Hartigan 1975, Anderberg 1973,

Table 4.2. Non-hierarchical clustering options in some program packages.

	BMDP 7	Statistica	SYN-TAX
k-means methods	++	++	++
coefficient-independent method			++
multiple partitioning			++
quick clustering			++
fuzzy *c*-means procedure			+

Orlóci 1978, Späth 1980, and – for **COMPCLUS** – Gauch 1979. More recently program lists rarely appear in publications because most users prefer easily executable, user-friendly routines in which the cluster analysis code itself represents a negligible part as compared to the user interface modules. The availability of clustering algorithms mentioned in this chapter is summarized by Table 4.2.

4.5 Imaginary dialogue

Q: *I am afraid that all procedures treated in this chapter will recognize ball-shaped clusters only, and elongated point clouds will not be detected in the multidimensional space. Can you recommend a procedure that detects, say, the elongated and arched clusters seen in Figures 4.3d-e? I would say that these clusters apparently exist and are well-separated, but none of your methods has found them.*

A: This is a very good point, because I was satisfied in the above discussion by showing to you how unexpected the results of certain clustering methods can be for some example data with otherwise known properties. There are of course methods capable of revealing sausage-shaped or arched clusters, but they will be discussd in the next chapter because of their hierarchical nature. I can tell you in advance, however, that it is the single link procedure, whose underlying principles appear in certain non-hierarchical methods as well (e.g., in Orlóci's TRGRPS procedure, 1976b, 1978). Partitions can be easily derived from hierarchical classifications, as you will see it later...

Q: *It is also of primary concern to me whether partitioning can only be achieved by iterative methods, in fact "trial and error" searches, which rarely produce unanimous results? Are you not aware of any method which will always find the optimal partition?*

A: Most classification problems are very difficult or impossible to solve by a simple algorithm that always leads to a unique result and is very efficient at the same time. If you insist on seeking for the absolute optimum, you can only be sure if all the possible results are examined. Fortunately, there are some exceptions, such as the *"branch and bound"* algorithm (Grötschel & Wakabayashi 1990) which detects the absolute minimum of the sum of squares for a few dozen or so objects, but the method requires lot of computing effort and does not apply to larger data sets.

Q: *Is there any method which will always produce unique results? Is it really important that the final result be unique?*

A: From a mathematical viewpoint, algorithmic uniqueness is an important criterion. In applied studies, however, this is not necessarily so. We can get sufficient answers to our questions by *heuristic searches* which do not guarantee absolute optimality. Gauch (1982) provides some arguments in favour of heuristic methods. A most remarkable point is that biological data collection and analysis are burdened in each stage by many subjective elements so that it is self-deception to thrive for perfectness in algorithmic requirements. Nonetheless, we should choose the best-defined procedures whenever possible.

Q: *How many ways can we arrange m objects into k classes?*

A: The number of different partitions into k clusters is obtained by the Stirling-formula of the first kind as follows:

$$S = \frac{1}{k!} \sum_{i=0}^{k} (-1)^{k-i} \binom{k}{i} i^m \,. \tag{4.17}$$

You can now verify that 20 objects (not too many in actual studies) can be arranged into two non-empty clusters in 524287 ways! (Actually, the formula simplifies to $S = 2^m/2 - 1$ for $k = 2$; just recall your studies in elementary combinatorics!)

Q: *I am intrigued by another, apparently important thing: for most methods we have to specify in advance the number of clusters. This seems to introduce a great deal of arbitrariness into the analysis. We may of course "play" a lot with this number in an attempt to find a "meaningful" result, but this is cumbersome and does not remove the subjectivity of the investigator.*

A: Let me give you a more detailed answer to this question, since the number of clusters present in the data is a central issue of numerical classification. I cannot give you a full account here, but in the later chapters I will return to this topic.

Yes, the methods discussed above must be used in a "trial and error" style in order to depict the data structure appropriately. This is not a serious disadvantage, because modern computers are fast and efficient enough to do the "playing" in reasonable time and with the least effort. We must admit, however, that these methods of non-hierarchical clustering do not really stand on their own in the extensive area of exploratory data analysis. In most cases, complementary analysis by other kinds of methods, such as hierarchical clustering and ordination, need to be performed simultaneously. Ordinations, for example, will help you to visualize the shape of a multidimensional point swarm (you will see later, how), and its comparison with partitions may be extremely informative. A hierarchical classification is series of partitions, and there are quite few techniques designed to find the optimum in this sequence (see Subsection 5.5.3).

But I do not want to let you be fully disappointed. I can tell you that there are new developments in this area suggesting that partitioning methods themselves will provide the desired solution. As a biologist, you will be most delighted to see that mathematicians have invented long ago the so-called "genetic algorithms" (Holland 1975, Goldberg 1989) which appear suitable for the purpose. (Perhaps, the term "evolutionary algorithm" would reflect more faithfully what is going on, but this is not critical, of course.) The point is that a "*population*" of possible final results is generated by simulation, we define a "*fitness"* function expressing the viability of the individuals of that population (i.e., the goodness of alternative classifica-

tions), and we design a set of transitional rules by which the individuals of the population may change (i.e., *mutation* is also possible). Individuals having a tendency to improve the fitness are retained and allowed to "reproduce" themselves, whereas those having detrimental properties are selected out: they become "extinct". The mechanisms of evolution run by themselves for some time, and then the user examines the fittest individuals of the final population. If the evolutionary processes are allowed to operate for long enough, then there will a be high chance to find individuals with maximum fitness which cannot be improved any further. (And here you see the big difference between artificial and biological evolution: the latter will never end as long as there is life on Earth.) In order to modify partitions in this manner, you need a new term which, in contrast with the *k*-means procedure (where the centroids are not real objects, only averages), will represent a cluster by one of its objects, and all the other objects are classified according to their distances from this (*k-medoid* method, Lucasius et al. 1993). Each individual of the population may be described in terms of a "chromosome", that is a string of *m* values, 0-s and 1-s. Having 1 in the *i*-th position of the chromosome means that the object is a medoid, whereas 0 means that the object has to be assigned to the nearest medoid. Thus, the chromosome describes a classification whose goodness can be measured in many ways (Moraczewski et al. [1995] proposed to use the stress function [7.66] applied extensively in nonmetric multidimensional scaling). On the chromosome, point mutations are allowed to happen, and even crossing overs are possible. The fittest subset of the new individuals is retained. These procedures are in an experimental stage of development, because the frequency of mutations and crossing overs, the size of the starting population and the specification of other parameters strongly influence the efficiency of the algorithm (see the study by Moraczewski et al. 1995). There is no doubt that in the future such evolutionary algorithms will appear even in commercial program packages.

Q: *This was a very illuminating detour for me, showing how interesting research problems appear in a field appearing at first glance – I am sorry to say that – not very exciting. And now let me turn back to the examples you showed: I have some comments. It is remarkable that the three procedures compared produced identical results only for the random case and for the four "obvious" clusters (a and b). The latter is all right, because those clusters are ideal groups, but it is unclear to me why did they yield the most diverging results for the regularly dispersed objects (case f)?*

A: This is the point: the regular arrangement, that is when the objects fall – with some noise added – onto a square grid (see last two columns of Table A3), fails to satisfy any criteria of classifiability. Since it did not escape your attention I was probably successful to show that discrepancy of results is some indication that cluster structure is lacking. The opposite is not true, however, because in the random case – as you discovered – the results did not differ, yet there are no real clusters in the data.

Q: *I have the impression that your coefficient-independent partitoning procedure outperformed the other two. In any case...*

A: Let me interrupt you! Do not let yourself be misled by the examples! No matter how illuminative they appeared, they did not prove anything, and any statement that "method A is always better than B" is false. A right conclusion from the illustrative examples is perhaps that you should never be satisfied with a single result and it is always advisable to examine your

data with as many methods as possible. This is not a big deal with the present-day computer technology...

Q: *Oh yes, but then what should I do with that big bunch of results produced by those different procedures? Do you really mean that one should always do lots of alternative analyses?*

A: This observation is straight to the point – as several times before. I have to make you wait until the 9th chapter which is fully devoted to this problem.

Q: *No matter what you propose as a solution, I dare say that space series will be involved again!*

A: You bet! The last example of this chapter implicitly referred to nothing else but a series: by changing the coefficient of fuzziness we obtain a series of classifications. Although only two values were examined (Fig. 4.9) you could see that gradual changes of *f* will lead to a classification series which tells much more on the cluster structure of objects than any arbitrarily selected value by itself. Nevertheless, there are many options for further evaluation, but be patient.

Q: *Well, just let me ask one more question: in which fields you consider non-hierarchical classification to be extremely useful or even indispensable?*

A: For example, in vegetation mapping. The map itself, in which vegetation types are depicted by different colors, is a classification. The taxonomist may also be interested to distinguish among taxonomic categories of the same rank within a population. But there are more peculiar applications of these methods: in Canada types of shoplifting behaviour were identified by partitioning (McShane & Noonan 1993), in Japan winning tricks of sumo wrestlers were categorized in a similar approach (sorry, I cannot remember the reference; it was an oral conference contribution). Non-hierarchical classification helps people to organize themselves in practically any aspect of everyday and scientific life.