CHAPTER 2

Partitioning Around Medoids (Program PAM)

1 SHORT DESCRIPTION OF THE METHOD

When partitioning a set of objects into k clusters the main objective is to find clusters, the objects of which show a high degree of similarity, while objects belonging to different clusters are as dissimilar as possible. Of course, many methods exist that try to achieve this aim. The algorithm used in the program PAM is based on the search for k *representative objects* among the objects of the data set. As evoked by their name, these objects should represent various aspects of the structure of the data. In the cluster analysis literature such representative objects are often called *centrotypes*. In the PAM algorithm the representative objects are the so-called *medoids* of the clusters (Kaufman and Rousseeuw, 1987). After finding a set of k representative objects, the k clusters are constructed by assigning each object of the data set to the nearest representative object.

To illustrate the algorithm used in PAM (a detailed description is given in Section 4) let us consider the data set represented in Figure 1. This data set contains 10 objects ($n = 10$) each characterized by two variables ($p = 2$), called x and y. The x and y values are given in Table 1.

Suppose the data set must be divided into two subsets or clusters ($k = 2$). In the algorithm one first considers possible choices of two representative objects and then constructs the clusters around these representative objects. As an example, suppose objects 1 and 5 are the selected representative objects. In Table 2 the dissimilarities from each of the objects to the two selected objects are given, as well as the smallest of these two dissimilarities and the corresponding representative object. (In this example we have simply used Euclidean distances between the points on

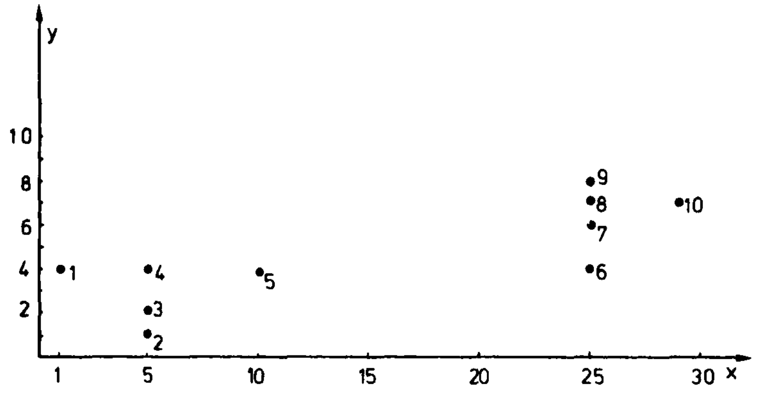


Figure 1 Two-dimensional example with 10 objects.

Table 1 Coordinates of the Objects of the Example of Figure 1

Number	x Coordinate	y Coordinate
1	1.0	4.0
2	5.0	1.0
3	5.0	2.0
4	5.0	4.0
5	10.0	4.0
6	25.0	4.0
7	25.0	6.0
8	25.0	7.0
9	25.0	8.0
10	29.0	7.0

the plot.) The average dissimilarity is 9.37. This value measures the tightness of the clusters and therefore the quality of the clustering.

In Table 3 the assignment is carried out for the case objects 4 and 8 are selected as representative objects. The clusterings associated with these two pairs of representative objects are shown in Figure 2.

The average dissimilarity for the case objects 4 and 8 are selected is 2.30, which is considerably less than the value of 9.37, found when 1 and 5 were the representative objects. In Section 4 an algorithm will be described, which for a given k makes it possible to select k representative objects which yield a very low average dissimilarity, and therefore a "good" partition. The clustering found with this algorithm is the same one as in

Table 2 Assignment of Objects to Two Representative Objects

Object Number	Dissimilarity from Object 1	Dissimilarity from Object 5	Minimal Dissimilarity	Closest Representative Object
1	0.00	9.00	0.00	1
2	5.00	5.83	5.00	1
3	4.47	5.39	4.47	1
4	4.00	5.00	4.00	1
5	9.00	0.00	0.00	5
6	24.00	15.00	15.00	5
7	24.08	15.13	15.13	5
8	24.19	15.30	15.30	5
9	24.33	15.52	15.52	5
10	28.16	19.24	19.24	5
			Average 9.37	

Table 3 Assignment of Objects to Two Other Representative Objects

Object Number	Dissimilarity from Object 4	Dissimilarity from Object 8	Minimal Dissimilarity	Closest Representative Object
1	4.00	24.19	4.00	4
2	3.00	20.88	3.00	4
3	2.00	20.62	2.00	4
4	0.00	20.22	0.00	4
5	5.00	15.30	5.00	4
6	20.00	3.00	3.00	8
7	20.10	1.00	1.00	8
8	20.22	0.00	0.00	8
9	20.40	1.00	1.00	8
10	24.19	4.00	4.00	8
			Average 2.30	

Table 3, but the two representative objects are 3 and 8 and in this way the average dissimilarity is 2.19.

There are basically two ways of entering the data in PAM. The most common way is by means of a matrix of measurement values. The rows of this matrix represent the objects and the columns correspond to variables, which must be on an interval scale. An example of such data is given in Table 1.

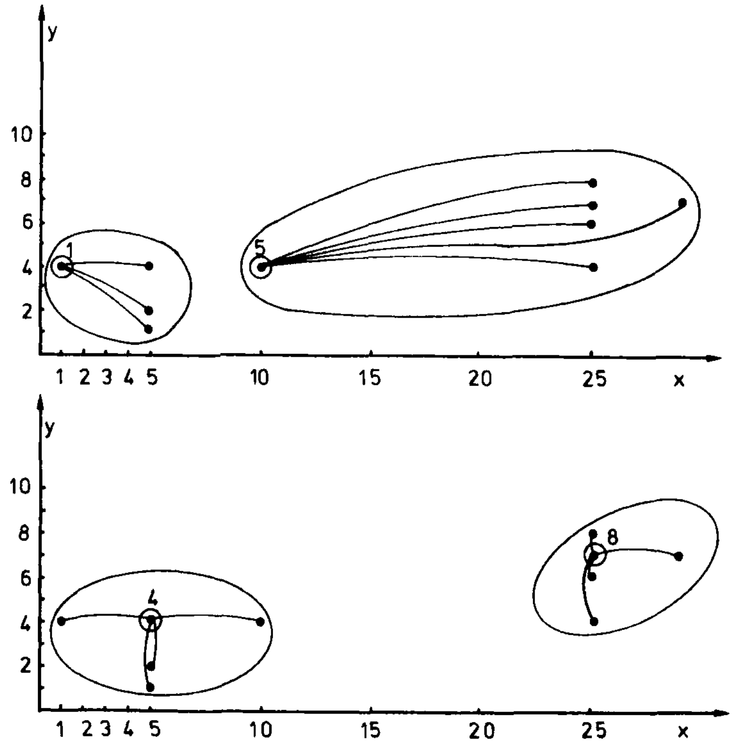


Figure 2 Clusterings corresponding to the selections described in Tables 2 and 3.

Alternatively the program can be used by entering a matrix of dissimilarities between objects. Such dissimilarities can be obtained in several ways. Often they are computed from variables that are not necessarily on an interval scale but which may also be binary, ordinal, or nominal. (This can be done by using the program DAISY of Chapter 1.) It also happens that dissimilarities are given directly, without resorting to any measurement values.

In many clustering problems one is particularly interested in a characterization of the clusters by means of typical or representative objects. These are objects that represent the various structural aspects of the set of objects being investigated. There can be many reasons for searching for representative objects. Not only can these objects provide a characterization of the clusters, but they can often be used for further work or research, especially when it is more economical or convenient to use a small set of k objects instead of the large set one started off with. In the method used in the program PAM the representative object of a cluster is its medoid, which we define as that object of the cluster for which the average dissimilarity to all the objects of the cluster is minimal. (In fact, the name PAM comes from *P*artitioning *A*round *M*edoids.) As the objective is to find k such objects, we call this the *k-medoid method* (in the literature one also encounters the name k-median which to us seems less appropriate, as confusion can arise with the classical notion of median).

Another typical aspect of PAM is that it provides a large number of statistics by which a thorough investigation of the clustering results is made possible. Particularly worth mentioning are the medoids of the clusters (with their coordinates), the diameters and separations of the clusters, and also a graphical representation of the clusters by means of so-called silhouettes.

2 HOW TO USE THE PROGRAM PAM

The k-medoid method can be applied by using the program PAM. This program was written in Fortran and runs on an IBM PC, XT, AT, or compatible computer with at least 256K of internal storage. In Section 4 some details of the program are described and the Appendix discusses the implementation of the program.

The program PAM is operated entirely interactively. The data specifications and options must be entered by keyboard. The data set (consisting of the measurements or the dissimilarities) can be input by the keyboard or from a file located on a floppy or hard disk. If they are input from the keyboard, there is an option for saving them in a file. The output can be sent either to the screen, to the printer, or to a file which is then written on floppy or hard disk. In Section 2.1 we will describe how the program is operated. In Section 2.2 the output of the clustering results will be detailed. The case of missing measurement values, which is not of interest to all readers, is discussed in Section 2.3.

2.1 Interactive Use and Input

In order to run the program the first thing to do is to insert the diskette containing the file PAM.EXE. To load and run the program the user only has to type A:PAM in case the diskette is in drive A. The user must then press the key ENTER. (The carriage return or ENTER key, which instructs the computer to read the information, must follow each answer or input of data. It will not be mentioned hereafter.) The program then generates the following screen:

```
PARTITIONING AROUND MEDOIDS

DO YOU WANT TO ENTER MEASUREMENTS ? (PLEASE ANSWER M)
OR DO YOU PREFER TO GIVE DISSIMILARITIES ?
(THEN ANSWER D) : m
```

Note that the user-supplied information (answers or data) is underlined.

The user now must select one of the two types of data to be used for the clustering algorithm. If he chooses measurements, the program will itself compute the dissimilarities before performing the clustering. This results in several additional options concerning the treatment of the measurements. We will suppose measurements were chosen in order to illustrate these options.

After the choice of input data type has been made, the following message appears:

```
THE PRESENT VERSION OF THE PROGRAM CAN HANDLE UP TO 100
OBJECTS.
(IF MORE ARE TO BE CLUSTERED, THE ARRAYS INSIDE THE PRO-
GRAM MUST BE ADAPTED)
HOW MANY OBJECTS ARE TO BE CLUSTERED ?
-----------------------------------------
PLEASE GIVE A NUMBER BETWEEN 3 AND 100 : 10
```

The user then enters the number of cases in his data set. Note that there are limits on the size of the data set that can be handled. (These are due to the central memory limitation of the computer.) Also note that if an answer of less than 3 or more than 100 is given, the program reiterates the question. The next message concerns the number of clusters:

```
CLUSTERINGS WILL BE CARRIED OUT IN K1 TO K2 CLUSTERS.
PLEASE ENTER K1 : 2
PLEASE ENTER K2 : 2
```

If the values $K1$ and $K2$ are not satisfactory, special messages are given and they have to be entered again.

Subsequently, if input of measurements was chosen (see the first question), specific information related to the measurements must be entered. The questions and prompts are now discussed. (Note that these do not appear if input of dissimilarities was chosen.) The user first specifies the total number of variables in his data set:

```
THE PRESENT VERSION OF THE PROGRAM ALLOWS TO ENTER UP TO
80 VARIABLES, OF WHICH AT MOST 20 CAN BE USED IN THE ACTUAL
COMPUTATIONS.
(IF MORE ARE NEEDED, THE ARRAYS INSIDE THE PROGRAM MUST BE
ADAPTED.)
WHAT IS THE TOTAL NUMBER OF VARIABLES IN YOUR DATA SET ?
----------------------------------------------------------
PLEASE GIVE A NUMBER BETWEEN 1 AND 80 : 5
```

As with the number of objects there are also limits on the number of variables. If a number is entered outside the range, a message appears and the question is restated.

In order to make it possible to cluster the same set of objects using different sets of variables, the program allows the choice of a subset of variables to be used for the clustering. The choice of variables must be made by the user:

```
HOW MANY VARIABLES DO YOU WANT TO USE IN THE ANALYSIS ?
-------------------------------------------------------
(AT MOST 5) : 2
```

Note that in the text this quantity is denoted by *p*. When the number of selected variables equals the total number of variables in the data set, a table is given in which each variable is to be identified by a label of at most 10 characters. On the other hand, when not all variables are uesd, the *position* of the selected variables must also be given. For our example, the following appears on screen:

```
VARIABLE TO BE USED:           POSITION   LABEL (AT MOST 10
                                          CHARACTERS)
------------------------------ ↓↓↓↓ ---↓↓↓↓↓↓↓↓↓↓---------
NUMBER 1 :                        2       x-coordina
NUMBER 2 :                        3       y-coordina
```

For each variable the position in the measurement matrix (column number) and its label must be entered on the same line. Note that after the position has been entered, blanks must be used to reach the area reserved for the label.

Also in this table checks on the input are carried out. For example, it is not allowed to enter the same variable twice. Finally, the user has the option of standardizing the measurements and a choice is given between two dissimilarity measures.

DO YOU WANT THE MEASUREMENTS TO BE STANDARDIZED (PLEASE ANSWER YES) OR NOT? (THEN ANSWER NO) .... no

If this option is selected, the data are standardized as follows. First the mean (m_f) and the mean absolute deviation (s_f) of the fth variable are calculated:

$$m_f = \frac{1}{n}(x_{1f} + x_{2f} + \cdots + x_{nf})$$

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \cdots + |x_{nf} - m_f|)$$

Then, each measurement x_{if} is replaced by the standardized value z_{if} defined as

$$z_{if} = \frac{x_{if} - m_f}{s_f} \tag{1}$$

which is often called a z-score. The advantages and disadvantages of such standardization were discussed in Section 2 of Chapter 1.

DO YOU WANT TO USE EUCLIDEAN DISTANCE ? (PLEASE ANSWER E) OR DO YOU PREFER MANHATTAN DISTANCE ? (THEN ANSWER M) . . e

Let us recollect that the Euclidean distance between two objects i and j is given by

$$d(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2} \tag{2}$$

while their Manhattan distance corresponds to

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}| \tag{3}$$

(For the interpretation of these formulas see Section 2 of Chapter 1.) If a standardization is carried out, the x_{if} are replaced by z_{if} in the last two expressions. The answers to the following questions control the output of the program.

```
PLEASE ENTER A TITLE FOR THE OUTPUT (AT MOST 60 CHARACTERS)
------------------------------------------------------------
Data of Table 1

DO YOU WANT LARGE OUTPUT ? (PLEASE ANSWER YES)
OR IS SMALL OUTPUT SUFFICIENT ? (THEN ANSWER NO)
(IN THE LATTER CASE NO DISSIMILARITIES ARE GIVEN) ......... yes

DO YOU WANT GRAPHICAL OUTPUT (SILHOUETTES) ? PLEASE AN-
SWER YES OR NO ........................................ yes

DO YOU WANT TO ENTER LABELS OF OBJECTS ? PLEASE ANSWER YES
OR NO ................................................. no
```

If labels of objects are to be entered this is made possible by the following screen:

```
EACH LABEL MAY CONSIST OF AT MOST 3 CHARACTERS
OBJECT          LABEL
------------- ↓↓↓
NUMBER 1 :      AAA
NUMBER 2 :      BBB
NUMBER 3 :      CCC
NUMBER 4 :      DDD
NUMBER 5 :      EEE
```

If the labels are not provided by the user, they are constructed by the program as three-digit integers such as 001, 002, ... in which the leading zeroes are maintained. The following questions concern the input of the data.

```
DO YOU WANT TO READ THE DATA IN FREE FORMAT ?
THIS MEANS THAT YOU ONLY HAVE TO INSERT BLANK(S) BETWEEN
NUMBERS.
(NOTE : WE ADVISE USERS WITHOUT KNOWLEDGE OF FORTRAN
FORMATS TO ANSWER YES.)
MAKE YOUR CHOICE (YES / NO) : no

YOUR DESIRED FORTRAN FORMAT IS :
(BETWEEN BRACKETS AND AT MOST 60 CHARACTERS, e.g. (2F3.0,F1.0))
(5F5.1)
```

This format is chosen by the user to input the measurements or dissimilarities. It must be entered between brackets. (For the actual data only Fortran *F* formats and *E* formats are allowed because the dissimilarities are processed as real numbers. For ease of use *X* and / are also allowed. Details about formats are described in the Appendix.)

Subsequently, information must be provided concerning the status of input and output. Several options make it possible to use the keyboard, screen, printer, and files on disk.

```
PLEASE GIVE THE NAME OF THE FILE CONTAINING THE DATA (e.g.
TYPE A:EXAMPLE.DAT)
OR TYPE KEY IF YOU PREFER TO ENTER THE DATA BY KEYBOARD.
WHAT DO YOU CHOOSE ? key
```

If the data are to be entered by keyboard, there is an option for saving them on a file.

```
DO YOU WANT TO SAVE YOUR DATA ON A FILE ?
PLEASE ANSWER YES OR NO: yes
ON WHICH FILE DO YOU WANT TO SAVE YOUR DATA ?
(WARNING : IF THERE ALREADY EXISTS A FILE WITH THE SAME
           NAME THEN THE OLD FILE WILL BE OVERWRITTEN.)
TYPE e.g. B:SAVE.DAT ................................ c:test.dat
```

The output can be directed toward the screen, a printer, or a file:

```
WHERE DO YOU WANT YOUR OUTPUT ?
--------------------------------
   TYPE CON IF YOU WANT IT ON THE SCREEN
OR TYPE PRN IF YOU WANT IT ON THE PRINTER
OR TYPE THE NAME OF A FILE (e.g. B:EXAMPLE.OUT)
(WARNING : IF THERE ALREADY EXISTS A FILE WITH THE SAME
           NAME THEN THE OLD FILE WILL BE OVERWRITTEN.)
WHAT DO YOU CHOOSE ? ........................... c:test.res
```

When measurements are to be entered, PAM makes provision for the occurrence of missing values in the data.

```
CAN MISSING DATA OCCUR IN THE MEASUREMENTS ?
PLEASE ANSWER YES OR NO : no
```

An affirmative answer to this question results in several prompts which, as they are not of interest to all users, will be discussed in Section 2.3.

Finally, the data specifications and chosen options are displayed:

```
DATA SPECIFICATIONS AND CHOSEN OPTIONS
--------------------------------------
TITLE : Data of Table 1
THERE ARE 10 OBJECTS
LABELS OF OBJECTS ARE NOT READ
INPUT OF MEASUREMENTS
LARGE OUTPUT IS WANTED
GRAPHICAL OUTPUT IS WANTED (SILHOUETTES)
CLUSTERINGS ARE CARRIED OUT IN 2 TO 2 CLUSTERS

THERE ARE 5 VARIABLES IN THE DATA SET,
   AND 2 OF THEM WILL BE USED IN THIS ANALYSIS
THE MEASUREMENTS WILL NOT BE STANDARDIZED
EUCLIDEAN DISTANCE WILL BE USED
THERE ARE NO MISSING VALUES
THE INPUT FORMAT FOR THE MEASUREMENTS IS
(5F5.1)
THE DATA WILL BE READ FROM THE KEYBOARD
THE DATA WILL BE SAVED ON FILE : C:TEST.DAT
YOUR OUTPUT WILL BE WRITTEN ON : C:TEST.RES

ARE ALL THESE SPECIFICATIONS OK ? YES OR NO : yes
```

If the user answers no to this question, the program returns to the first line (concerning the choice of measurements or dissimilarities). Otherwise it proceeds with the input of the actual data.

Let us first consider the situation of input of measurements. For the example of Table 1, if the data are entered in the preceding format, (5F5.1), the following screen will appear:

```
PLEASE ENTER YOUR DATA FOR EACH OBJECT

THE 5 MEASUREMENTS FOR OBJECT 001 :
  1.0  1.0  4.0  24.0  0.6
THE 5 MEASUREMENTS FOR OBJECT 002 :
  2.0  5.0  1.0  26.5  -0.3
THE 5 MEASUREMENTS FOR OBJECT 003 :
  3.0  5.0  2.0  27.0  0.4
THE 5 MEASUREMENTS FOR OBJECT 004 :
  4.0  5.0  4.0  23.5  0.2
THE 5 MEASUREMENTS FOR OBJECT 005 :
  5.0  10.0  4.0  24.5  -0.6
```

```
THE 5 MEASUREMENTS FOR OBJECT 006 :
  6.0  25.0  4.0  25.5  0.1
THE 5 MEASUREMENTS FOR OBJECT 007 :
  7.0  25.0  6.0  26.0  0.3
THE 5 MEASUREMENTS FOR OBJECT 008 :
  8.0  25.0  7.0  23.5  -0.2
THE 5 MEASUREMENTS FOR OBJECT 009 :
  9.0  25.0  8.0  22.0  0.0
THE 5 MEASUREMENTS FOR OBJECT 010 :
  10.0  29.0  7.0  24.5  -0.3
```

Note that the first variable happens to be the number of the observation and that only the second and third variables are to be used in this particular analysis.

Let us now assume that the data consist of a set of dissimilarities. These are usually listed as in the matrix

$$\begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & 0 & 2 & 6 & 10 & 9 \\ b & 2 & 0 & 5 & 9 & 8 \\ c & 6 & 5 & 0 & 4 & 5 \\ d & 10 & 9 & 4 & 0 & 3 \\ e & 9 & 8 & 5 & 3 & 0 \end{array} \qquad (4)$$

For example, the dissimilarity between objects a and d equals 10. Observe that the matrix has nonnegative entries, zeroes on the diagonal, and is symmetric. Therefore, only the lower triangular part of the n-by-n matrix of dissimilarities is entered.

Be careful to delete the upper triangular part of the matrix, and also the diagonal, because otherwise wrong values might be read (especially if more than one line is used per object). We chose to read the lower triangular matrix instead of the upper one, because this makes it very easy to add objects to the data set by simply adding lines at the end of the data file.

When entering the dissimilarities by keyboard, the following screen appears:

```
FOR OBJECT J, ENTER THE DISSIMILARITIES TO OBJECTS
1,2,...,(J - 1)

DISSIMILARITY BETWEEN OBJECTS BBB AND AAA :
  2.0
THE 2 DISSIMILARITIES FOR OBJECT CCC :
  6.0  5.0
```

```
THE 3 DISSIMILARITIES FOR OBJECT DDD :
 10.0  9.0  4.0
THE 4 DISSIMILARITIES FOR OBJECT EEE :
 9.0  8.0  5.0  3.0
```

Conversely, the dissimilarities may also be read from a file which should then be organized as

```
 2.0
 6.0  5.0
10.0  9.0  4.0
 9.0  8.0  5.0  3.0
```

In some applications, similarities are given instead of dissimilarities. These cannot be used as input to the program but must first be converted to dissimilarities; this can, for example, be done by means of the program DAISY described in Chapter 1.

It is important to note that the input file requirements of PAM are completely identical to those of FANNY (Chapter 4), AGNES (Chapter 5), and DIANA (Chapter 6). This means that any input file constructed for either program will also run on the others. Also, the interactive dialogue is extremely similar for all four programs.

2.2 Output

In this section the output generated by PAM is described. Where necessary it is illustrated with the output of some test data. The output discussed in this section is obtained from a standard usage of the program, without missing values. (An example with missing values is given in Section 2.3. Outputs associated with special situations and error messages are discussed in the Appendix.)

The output of PAM is divided into five parts, two of which are not always obtained (parts c and e):

a. Identification of the Program and the Data Set

The phrase "PARTITIONING AROUND MEDOIDS" shows that the program PAM was used to obtain this output. The name of the data set is then printed as it was provided by the user.

b. Data Specifications and Chosen Options

Number of objects

Options concerning input and output:

- reading labels or not
- input of dissimilarities or measurements

- large or small output
- graphical output or not
- smallest and largest number of clusters

If measurement values are entered:

- number of variables in the data set
- which variables to be used in the analysis
- standardization or not
- the selected dissimilarity measure
- the absence or presence of missing values

The input format for measurements or dissimilarities.

Options for input and output (choice between keyboard and file for input and between screen, printer, and file for output).

c. Dissimilarities and Standardized Measurements

In case of dissimilarity input and if large output was asked for, the dissimilarities between objects are given before the clustering results.

In case of input of measurement values which are to be standardized, the average and mean absolute deviation of each variable are printed. In case of large output this is followed by the transformed measurements and the dissimilarities. If the measurements are not standardized and large output was requested, only the dissimilarities are listed.

In Figure 3, parts *a*, *b*, and *c* of the output are given for the data of Table 1.

d. Numerical Output Concerning Each Clustering

The number of clusters (number of representative objects).

The average dissimilarity for the solution found in the first part of the algorithm (called BUILD).

The average dissimilarity for the solution found in the second part of the algorithm (called SWAP).

(These two values are defined as the average dissimilarity between each object and its most similar representative object; details are discussed in Section 4.1.)

For each cluster the following information is printed:

- its medoid (with the label)
- its number of objects (size)

```
                    ********************************
                    *                              *
                    *  PARTITIONING AROUND MEDOIDS *
                    *                              *
                    ********************************

TITLE  : Data of Table 1

DATA SPECIFICATIONS AND CHOSEN OPTIONS
--------------------------------------
 THERE ARE   10 OBJECTS
 LABELS OF OBJECTS ARE NOT READ
 INPUT OF MEASUREMENTS
 LARGE OUTPUT IS WANTED
 GRAPHICAL OUTPUT IS WANTED (SILHOUETTES)
 CLUSTERINGS ARE CARRIED OUT IN    2 TO    2 CLUSTERS

 THERE ARE    5 VARIABLES IN THE DATA SET,
  AND    2 OF THEM WILL BE USED IN THE ANALYSIS
 THE LABELS OF THESE VARIABLES ARE :
         x-coordina    (POSITION :  2)
         y-coordina    (POSITION :  3)
 THE MEASUREMENTS WILL NOT BE STANDARDIZED
 EUCLIDEAN DISTANCE WILL BE USED
 THERE ARE NO MISSING VALUES
 THE INPUT FORMAT FOR THE MEASUREMENTS IS
 (5F5.1)

 THE DATA WILL BE SAVED ON FILE  : C:TEST.DAT

DISSIMILARITY MATRIX
--------------------

001
002          5.00
003          4.47      1.00
004          4.00      3.00      2.00
005          9.00      5.83      5.39      5.00
006         24.00     20.22     20.10     20.00     15.00
007         24.08     20.62     20.40     20.10     15.13      2.00
008         24.19     20.88     20.62     20.22     15.30      3.00      1.00
009         24.33     21.19     20.88     20.40     15.52      4.00      2.00      1.00
010         28.16     24.74     24.52     24.19     19.24      5.00      4.12      4.00
             4.12
```

Figure 3 PAM output, parts *a*, *b*, and *c*.

- the labels of its members (if no labels were read in the input, numbers are generated by the program)
- the coordinates of the medoid: This only occurs when there are measurements (when standardization is requested, only standardized coordinates are given)

The clustering vector: The jth element of this vector is the number of the cluster to which object j belongs (the clusters are numbered in such a way that when reading the clustering vector from left to right, cluster 1 is encountered before cluster 2, cluster 2 before cluster 3, and so on; therefore cluster 1 is defined as the cluster containing the first object).

The clustering characteristics:

- whether a cluster is a singleton
- whether a cluster is isolated: Two types of isolated clusters are considered in the program, L-clusters and L^*-clusters. They have

the following definitions (see Gordon, 1981):

C is an L-cluster if for each object i belonging to C:

$$\max_{j \in C} d(i, j) < \min_{h \notin C} d(i, h)$$

C is an L^*-cluster if:

$$\max_{i, j \in C} d(i, j) < \min_{l \in C,\, h \notin C} d(l, h) \tag{5}$$

It is clear that L^*-clusters are also L-clusters. It should be observed that the property of being isolated depends both on the internal structure of a cluster and its position relative to other clusters. When dividing the data into two clusters, it can happen that one is isolated and the other is not.

- the diameter and separation of each cluster: the left-hand side of Eq. (5) is called the diameter of cluster C and the right-hand side its separation.
- in each cluster, the average dissimilarity of the objects to the medoid and the maximum dissimilarity of any object to the medoid.

As an illustration, the numerical output of the example of Section 1 concerning the clustering into two clusters ($k = 2$) is given in Figure 4.

e. Graphical Output Concerning Each Clustering

If this was requested in the interactive part of the program, a graphical representation of each clustering is provided (except for the boundary cases $k = 1$ and $k = n$) displaying the silhouettes introduced by Rousseeuw (1987). Each cluster is represented by one silhouette, showing which objects lie well within the cluster and which objects merely hold an intermediate position. The entire clustering is displayed by plotting all silhouettes into a single diagram, allowing the user to compare the quality of the clusters. Such silhouettes are especially useful when the dissimilarities are on a ratio scale (as in the case of Euclidean distances) and when one is seeking compact and widely separated clusters.

Silhouettes are constructed in the following way: For each object i the value $s(i)$ is defined and then these numbers are combined into a plot. Take any object i of the data set. In order to define $s(i)$, first denote by A the cluster to which it has been assigned and then calculate

$$a(i) = \text{average dissimilarity of } i \text{ to all other objects of } A$$

```
*********************************************
*                                           *
*  NUMBER OF REPRESENTATIVE OBJECTS     2   *
*                                           *
*********************************************

RESULT OF BUILD
  AVERAGE DISSIMILARITY =          3.422

FINAL RESULTS
  AVERAGE DISSIMILARITY =          2.186

  CLUSTERS
  NUMBER  MEDOID   SIZE      OBJECTS
    1       003      5     001 002 003 004 005

    2       008      5     006 007 008 009 010

COORDINATES OF MEDOIDS
**********************

003          5.00      2.00
008         25.00      7.00

CLUSTERING VECTOR
*****************

             1  1  1  1  1  2  2  2  2  2

CLUSTERING CHARACTERISTICS
**************************

 CLUSTER     1 IS ISOLATED,
        WITH DIAMETER  =        9.00 AND SEPARATION =        15.00
        THEREFORE IT IS AN L*-CLUSTER.

 CLUSTER     2 IS ISOLATED,
        WITH DIAMETER  =        5.00 AND SEPARATION =        15.00
        THEREFORE IT IS AN L*-CLUSTER.

 THE NUMBER OF ISOLATED CLUSTERS =     2

 DIAMETER OF EACH CLUSTER
      9.00      5.00

 SEPARATION OF EACH CLUSTER
     15.00     15.00

 AVERAGE DISSIMILARITY TO EACH MEDOID
      2.57      1.80

 MAXIMUM DISSIMILARITY TO EACH MEDOID
      5.39      4.00
```

Figure 4 Numerical output for the clustering into two clusters of the example of Table 1.

This can only be done when A contains other objects apart from i, so at this point we assume that A is not a singleton.

Now consider any cluster C different from A and define

$$d(i, C) = \text{average dissimilarity of } i \text{ to all objects of } C$$

After computing $d(i, C)$ for all clusters $C \neq A$, we select the smallest of those:

$$b(i) = \min_{C \neq A} d(i, C)$$

The cluster B for which this minimum is attained [that is, $d(i, B) = b(i)$] is called the *neighbor* of object i. This is like the second-best choice for object

i: If cluster A is discarded, cluster B is closest to i. Therefore, it is very useful to know the neighbor of each object in the data set. Note that the construction of $b(i)$ depends on the availability of clusters differing from A, which explains why silhouettes are not defined for $k = 1$.

The number $s(i)$ is obtained by combining $a(i)$ and $b(i)$ as follows:

$$\begin{aligned} s(i) &= 1 - \frac{a(i)}{b(i)} && \text{if } a(i) < b(i) \\ &= 0 && \text{if } a(i) = b(i) \\ &= \frac{b(i)}{a(i)} - 1 && \text{if } a(i) > b(i) \end{aligned} \tag{6}$$

It is possible to write this in one formula:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

When cluster A contains only a single object, it is unclear how $a(i)$ should be defined and then we simply set $s(i) = 0$. This choice is of course arbitrary, but a value of zero appears to be most neutral. Indeed, from the preceding definition we easily see that

$$-1 \leq s(i) \leq 1$$

for each object i.

Note that $s(i)$ remains invariant when all the original dissimilarities are multiplied by a positive constant, but that an additive constant is not allowed. This explains why we explicitly assume that the dissimilarities are on a ratio scale, which means that a dissimilarity of 6 may be considered twice as large as a dissimilarity of 3. For instance, Euclidean distances are on a ratio scale.

To strengthen our intuition about the meaning of $s(i)$, let us look at a few extreme situations. When $s(i)$ is at its largest (that is, close to 1), this implies that the "within" dissimilarity $a(i)$ is much smaller than the smallest "between" dissimilarity $b(i)$. Therefore, we can say that i is "well classified", as there appears to be little doubt that i has been assigned to an appropriate cluster: The second-best choice (B) is not nearly as close as the actual choice (A).

A different situation occurs when $s(i)$ is about zero. Then $a(i)$ and $b(i)$ are approximately equal and hence it is not clear at all whether i should have been assigned to A or to B. Object i lies equally far away from both, so it can be considered as an "intermediate" case.

The worst situation takes place when $s(i)$ is close to -1. Then $a(i)$ is much larger than $b(i)$, so i lies on the average much closer to B than to A. Therefore it would have seemed much more natural to assign object i to cluster B, so we can almost conclude that object i has been "misclassified".

To conclude, $s(i)$ measures how well object i matches the clustering at hand (that is, how well it has been classified). In the special case where there are only two clusters ($k = 2$), we note that moving object i from one cluster to the other will convert $s(i)$ to $-s(i)$.

Having computed the quantities $s(i)$ from the dissimilarities, we can now construct the graphical display. The silhouette of cluster A is a plot of the $s(i)$, ranked in decreasing order, for all objects i in A. On a line printer, we represent $s(i)$ by a row of asterisks, the length of which is proportional to $s(i)$. Therefore, the silhouette shows which objects lie well within their cluster and which ones are merely somewhere in between clusters. A wide silhouette indicates large $s(i)$ values and hence a pronounced cluster. The other dimension of a silhouette is its height, which simply equals the number of objects in A.

In order to obtain an overview, the silhouettes of the different clusters are printed below each other. In this way the entire clustering can be displayed by means of a single plot, which enables us to distinguish clear-cut clusters from weak ones. Above and below the plot there are scales going from 0.00 to 1.00 with steps of size 0.04 (to be read vertically). In the output of PAM, the following information is given for each object i:

The number of the cluster to which it belongs (under the header CLU).

The number of the neighboring cluster (under NEIG).

The value $s(i)$.

The three-character label of object i.

A line, the length of which is proportional to $s(i)$ (if this value is positive).

Also the following summary values are added:

The average of the $s(i)$ for all objects i in a cluster, which is called the *average silhouette width* of that cluster.

The average of the $s(i)$ for $i = 1, 2, \ldots, n$ which is called the *average silhouette width for the entire data set*.

This last number, which we will denote by $\bar{s}(k)$, can be used for the selection of a "best" value of k, by choosing that k for which $\bar{s}(k)$ is as

```
                                   ****************
                                   *              *
                                   * SILHOUETTES  *
                                   *              *
                                   ****************

                     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                     . . . . . . . . . . . . . . . . . . . . . . . . . .
                     0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                     0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0
CLU  NEIG  S(I)    I +++++++++++++++++++++++++++++++++++++++++++++++++++
                     +                                                 +
 1    2   .85     003+******************************************       +
 1    2   .83     004+*****************************************        +
 1    2   .83     002+*****************************************        +
 1    2   .77     001+**************************************           +
 1    2   .61     005+******************************                   +
                     +                                                 +
 2    1   .89     008+********************************************     +
 2    1   .89     007+********************************************     +
 2    1   .86     009+*******************************************      +
 2    1   .82     006+*****************************************        +
 2    1   .82     010+*****************************************        +
                     +                                                 +
                     +++++++++++++++++++++++++++++++++++++++++++++++++++
                     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                     . . . . . . . . . . . . . . . . . . . . . . . . . .
                     0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                     0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0

     CLUSTER     1 HAS AVERAGE SILHOUETTE WIDTH   .78
     CLUSTER     2 HAS AVERAGE SILHOUETTE WIDTH   .86

     FOR THE ENTIRE DATA SET, THE AVERAGE SILHOUETTE WIDTH IS   .82
```

Figure 5 Graphical output for the clustering into two clusters of the data in Table 1.

high as possible. The choice of k is one of the most difficult problems of cluster analysis, for which no unique solution exists; for a recent survey of alternative criteria see Milligan and Cooper (1985).

The silhouettes for $k = 2$ of the example of Figure 1 are given in Figure 5. Both clusters appear to be quite pronounced because their silhouettes are relatively wide. We see that the average silhouette width for the entire data set is $\bar{s}(k) = 0.82$ when $k = 2$. After computing $\bar{s}(k)$ for all possible k we find that this value 0.82 is the highest, so we conclude that $k = 2$ is an appropriate number of clusters for this data set. It is possible to repeat this in all applications by computing what we call the *silhouette coefficient*:

$$\mathrm{SC} = \max_k \bar{s}(k) \tag{7}$$

where the maximum is taken over all k for which the silhouettes can be constructed, which means $k = 2, 3, \ldots, n - 1$. On the one hand, this gives us a selected value of k. On the other, the SC is a useful measure of the amount of clustering structure that has been discovered by the classification algorithm. The silhouette coefficient is a dimensionless quantity which is at most equal to 1 and which does not change when all the original dissimilarities are multiplied by a constant factor. Experience with the SC has led us

Table 4 Subjective Interpretation of the Silhouette Coefficient (SC), Defined as the Maximal Average Silhouette Width for the Entire Data Set

SC	Proposed Interpretation
0.71–1.00	A strong structure has been found
0.51–0.70	A reasonable structure has been found
0.26–0.50	The structure is weak and could be artificial; please try additional methods on this data set
≤ 0.25	No substantial structure has been found

to a rather subjective interpretation, which is summarized in Table 4. Indeed, an SC close to 1 points to a very clear structure and a low SC indicates that one might better apply an alternative method of data analysis.

Because the silhouettes and the SC are not restricted to a particular partitioning algorithm, Table 4 will also be used in Chapters 3 and 4. Readers wanting to gather some more experience with silhouettes in order to obtain a better feeling for their interpretation are referred to the examples in Section 3 of this chapter.

2.3 Missing Values

In PAM provision is made for the case that there are missing values in the input of measurements. As this situation does not interest all users and as it has a strong impact on both input and output, it is treated in a special subsection.

If the user answers yes to the question concerning the occurrence of missing data the following questions must be answered:

```
IS THERE A UNIQUE VALUE WHICH IS TO BE INTERPRETED
AS A MISSING MEASUREMENT VALUE FOR ANY VARIABLE ?
PLEASE ANSWER YES OR NO : no
```

If missing values can occur but there is no single code representing a missing measurement, the following questions are asked:

```
SHOULD MISSING VALUES BE FORESEEN FOR THE VARIABLE
TEMPERATUR ? PLEASE ANSWER YES OR NO : no
```

```
SHOULD MISSING VALUES BE FORESEEN FOR THE VARIABLE
WEIGHT ? PLEASE ANSWER YES OR NO : yes
ENTER THE VALUE OF THE VARIABLE WHICH HAS TO BE INTER-
PRETED AS THE MISSING VALUE CODE : 9.999

SHOULD MISSING VALUES BE FORESEEN FOR THE VARIABLE
HEIGHT ? PLEASE ANSWER YES OR NO : yes
ENTER THE VALUE OF THE VARIABLE WHICH HAS TO BE INTER-
PRETED AS THE MISSING VALUE CODE : 99.99
```

This means that missing values are only foreseen for the second and third variables. The actual measurements are introduced in the following dialogue:

```
THE 3 MEASUREMENTS FOR OBJECT 001 :
12.3  8.328  38.76
THE 3 MEASUREMENTS FOR OBJECT 002 :
-5.4  9.999  18.12
THE 3 MEASUREMENTS FOR OBJECT 003 :
10.7  9.999  41.71
THE 3 MEASUREMENTS FOR OBJECT 004 :
-4.6  2.981  20.83
THE 3 MEASUREMENTS FOR OBJECT 005 :
-4.8  3.156  99.99
THE 3 MEASUREMENTS FOR OBJECT 006 :
11.0  7.826  40.54
```

The following adaptations have been incorporated into the program to take missing data into account. If the data are standardized, the average m_f and mean absolute deviation s_f are calculated using only present values. When calculating the distances $d(i, j)$, only those variables are considered in the sum for which the measurements for both objects are present. Subsequently the sum of terms is multiplied by p and divided by the number of such variables (in the case of Euclidean distances this is done before taking the square root). If the measurements are standardized, the transformed values corresponding to missing data are given the value 99.99.

Another consequence of missing values is that in the output of the program, right after the data specifications, the number of missing values for each variable and their total number are listed. In Figure 6 the output is given corresponding to these data.

```
                              ********************************
                              *                              *
                              *  PARTITIONING AROUND MEDOIDS *
                              *                              *
                              ********************************

TITLE   : DATA WITH MISSING VALUES

DATA SPECIFICATIONS AND CHOSEN OPTIONS
--------------------------------------
 THERE ARE     6 OBJECTS
 LABELS OF OBJECTS ARE NOT READ
 INPUT OF MEASUREMENTS
 LARGE OUTPUT IS WANTED
 GRAPHICAL OUTPUT IS WANTED (SILHOUETTES)
 CLUSTERINGS ARE CARRIED OUT IN    2 TO    2 CLUSTERS

 THERE ARE    3 VARIABLES IN THE DATA SET,
  AND    3 OF THEM WILL BE USED IN THE ANALYSIS
 THE LABELS OF THESE VARIABLES ARE :
         TEMPERATUR    (POSITION :  1)
         WEIGHT        (POSITION :  2)
         HEIGHT        (POSITION :  3)
 THE MEASUREMENTS WILL BE STANDARDIZED
 MANHATTAN DISTANCE WILL BE USED
 MISSING VALUES CAN OCCUR
 THE MEASUREMENTS WILL BE READ IN FREE FORMAT

 THE DATA WILL BE SAVED ON FILE : b:miss.dat

 VARIABLE WEIGHT     CONTAINS    2 MISSING VALUES
 VARIABLE HEIGHT     CONTAINS    1 MISSING VALUES

 THE TOTAL NUMBER OF MISSING VALUES IS       3

VARIABLE TEMPERATUR HAS AVERAGE      3.200     MEAN DEVIATION       8.133
VARIABLE WEIGHT     HAS AVERAGE      5.573     MEAN DEVIATION       2.504
VARIABLE HEIGHT     HAS AVERAGE     31.992     MEAN DEVIATION      10.014

STANDARDIZED MEASUREMENTS
-------------------------
 ( 99.99 DENOTES A MISSING VALUE)

001          1.12      1.10       .68
002         -1.06     99.99     -1.39
003           .92     99.99       .97
004          -.96     -1.03     -1.11
005          -.98      -.97     99.99
006           .96       .90       .85

DISSIMILARITY MATRIX
--------------------

001
002          6.36
003           .74      6.50
004          6.00       .55      5.95
005          6.25       .22      5.72       .14
006           .54      6.38       .23      5.82      5.71

**********************************************
*                                            *
*  NUMBER OF REPRESENTATIVE OBJECTS      2   *
*                                            *
**********************************************

RESULT OF BUILD
  AVERAGE DISSIMILARITY =          .189

FINAL RESULTS
  AVERAGE DISSIMILARITY =          .189

  CLUSTERS
  NUMBER   MEDOID    SIZE       OBJECTS

    1       006        3      001  003  006

    2       005        3      002  004  005
```

Figure 6 Output from the missing data example.

```
COORDINATES OF MEDOIDS (USING STANDARDIZED MEASUREMENTS)
********************************************************

006         .96       .90      .85
005       -.98      -.97     99.99

CLUSTERING VECTOR
*****************

            1  2  1  2  2  1

CLUSTERING CHARACTERISTICS
**************************

 CLUSTER     1 IS ISOLATED,
        WITH DIAMETER  =         .74 AND SEPARATION =        5.71
        THEREFORE IT IS AN L*-CLUSTER.

 CLUSTER     2 IS ISOLATED,
        WITH DIAMETER  =         .55 AND SEPARATION =        5.71
        THEREFORE IT IS AN L*-CLUSTER.

 THE NUMBER OF ISOLATED CLUSTERS =    2

 DIAMETER OF EACH CLUSTER
       .74       .55

 SEPARATION OF EACH CLUSTER
      5.71      5.71

 AVERAGE DISSIMILARITY TO EACH MEDOID
       .26       .12

 MAXIMUM DISSIMILARITY TO EACH MEDOID
       .54       .22

                                  ****************
                                  *              *
                                  *  SILHOUETTES *
                                  *              *
                                  ****************

                     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                     . . . . . . . . . . . . . . . . . . . . . . . . . .
                     0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                     0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0
CLU  NEIG  S(I)    I +++++++++++++++++++++++++++++++++++++++++++++++++++
                     +                                                 +
 1    2    .94   006+***********************************************   +
 1    2    .92   003+**********************************************    +
 1    2    .90   001+*********************************************     +
                     +                                                 +
 2    1    .97   005+*************************************************  +
 2    1    .94   004+***********************************************   +
 2    1    .94   002+***********************************************   +
                     +                                                 +
                     +++++++++++++++++++++++++++++++++++++++++++++++++++
                     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                     . . . . . . . . . . . . . . . . . . . . . . . . . .
                     0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                     0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0

      CLUSTER    1 HAS AVERAGE SILHOUETTE WIDTH   .92
      CLUSTER    2 HAS AVERAGE SILHOUETTE WIDTH   .95

      FOR THE ENTIRE DATA SET, THE AVERAGE SILHOUETTE WIDTH IS   .93

The output is on file : b:miss.res
```

Figure 6 (*Continued*)

3 EXAMPLES

In the previous sections, the algorithm was illustrated with an example consisting of two-dimensional measurements. In this section we start with an example in which the basic data consist of a dissimilarity matrix. (The same example will be used in Chapters 4, 5, and 6.)

The data were obtained by distributing a questionnaire in a political science class, asking the students to provide subjective dissimilarity coefficients between 12 countries. This is similar to an experiment of Wish (1970), with the students' own country (Belgium) included. The countries selected were (in alphabetical order) Belgium, Brazil, China, Cuba, Egypt, France, India, Israel, USA, USSR, Yugoslavia, and Zaire. The final dissimilarity coefficients were obtained by taking the averages of the coefficients given by the students.

In Table 5 the data are listed and Figure 7 contains the first part of the output, including the dissimilarity matrix because the option of large output is selected.

Figure 7 also contains the clustering into a single cluster (selection of one representative object). An interesting feature is that Belgium is found as medoid for the entire data set. A possible explanation is that the Belgian students tend to perceive smaller differences between their own country and each of the others, than among foreign countries. Further results given are

Table 5 Dissimilarities between 12 Countries, Obtained by Averaging the Results of a Survey among Political Science Students

Country	Dissimilarities to Other Countries										
	BEL	BRA	CHI	CUB	EGY	FRA	IND	ISR	USA	USS	YUG
BRA	5.58										
CHI	7.00	6.50									
CUB	7.08	7.00	3.83								
EGY	4.83	5.08	8.17	5.83							
FRA	2.17	5.75	6.67	6.92	4.92						
IND	6.42	5.00	5.58	6.00	4.67	6.42					
ISR	3.42	5.50	6.42	6.42	5.00	3.92	6.17				
USA	2.50	4.92	6.25	7.33	4.50	2.25	6.33	2.75			
USS	6.08	6.67	4.25	2.67	6.00	6.17	6.17	6.92	6.17		
YUG	5.25	6.83	4.50	3.75	5.75	5.42	6.08	5.83	6.67	3.67	
ZAI	4.75	3.00	6.08	6.67	5.00	5.58	4.83	6.17	5.67	6.50	6.92

```
                         ********************************
                         *                              *
                         *  PARTITIONING AROUND MEDOIDS *
                         *                              *
                         ********************************

TITLE  : Dissimilarities between 12 countries

DATA SPECIFICATIONS AND CHOSEN OPTIONS
--------------------------------------
 THERE ARE   12 OBJECTS
 LABELS OF OBJECTS ARE READ
 INPUT OF DISSIMILARITIES
 LARGE OUTPUT IS WANTED
 GRAPHICAL OUTPUT IS WANTED (SILHOUETTES)
 CLUSTERINGS ARE CARRIED OUT IN    1 TO    3 CLUSTERS
 THE DISSIMILARITIES WILL BE READ IN FREE FORMAT

 YOUR DATA RESIDE ON FILE          : COUNTRY.DAT


DISSIMILARITY MATRIX
--------------------

BEL
BRA         5.58
CHI         7.00     6.50
CUB         7.08     7.00     3.83
EGY         4.83     5.08     8.17     5.83
FRA         2.17     5.75     6.67     6.92     4.92
IND         6.42     5.00     5.58     6.00     4.67     6.42
ISR         3.42     5.50     6.42     6.42     5.00     3.92     6.17
USA         2.50     4.92     6.25     7.33     4.50     2.25     6.33     2.75
USS         6.08     6.67     4.25     2.67     6.00     6.17     6.17     6.92
            6.17
YUG         5.25     6.83     4.50     3.75     5.75     5.42     6.08     5.83
            6.67     3.67
ZAI         4.75     3.00     6.08     6.67     5.00     5.58     4.83     6.17
            5.67     6.50     6.92

*******************************************
*                                         *
*  NUMBER OF REPRESENTATIVE OBJECTS    1  *
*                                         *
*******************************************

FINAL RESULTS
  AVERAGE DISSIMILARITY =        4.590

  CLUSTERS
  NUMBER   MEDOID    SIZE      OBJECTS

    1       BEL       12      BEL BRA CHI CUB EGY FRA IND ISR USA USS
                              YUG ZAI

 DIAMETER OF EACH CLUSTER
      8.17

 AVERAGE DISSIMILARITY TO EACH MEDOID
      4.59

 MAXIMUM DISSIMILARITY TO EACH MEDOID
      7.08
```

Figure 7 First part of the output for the 12 countries example, including the clustering for $k = 1$.

the diameter of the cluster and the average and maximum dissimilarity to the medoid. The respective values of these cluster characteristics are 8.17, 4.59, and 7.08. Two of these values (diameter and maximum dissimilarity) can be found in Table 5 in which they have been circled. As mentioned in Section 2, silhouettes do not exist in the situation of a single cluster.

In Figure 8 the output is given for $k = 3$. The clustering obtained corresponds rather well to three groups of countries: Western, developing and Communist. It is remarkable that cluster 3, consisting of Cuba, China, USSR, and Yugoslavia, is the only isolated cluster. These countries are perceived as being very similar (within the limited set of countries we considered). It is also interesting to note that the other values given (diameter and separation, average dissimilarity, and maximum dissimilarity) are very close to each other for the three clusters.

Figure 8 also contains the silhouettes of these clusters. The first silhouette is higher than the others because the first cluster contains more objects than the remaining two. Our first impression is that the silhouettes are not very wide, which indicates that the clustering structure is no better than reasonable. In the first cluster, USA possesses the largest $s(i)$, which means that it was classified with the least amount of doubt. On the other hand, Egypt only attains $s(i) = 0.02$, which means that it lies on the boundary of the first cluster. Because its neighbor is cluster 2, this means that it has approximately the same dissimilarity to clusters 1 and 2 [moving Egypt to cluster 2 would yield $s(i) = -0.02$]. The second cluster contains the remaining developing countries and possesses a rather narrow silhouette,

```
*********************************************
*                                           *
*  NUMBER OF REPRESENTATIVE OBJECTS      3  *
*                                           *
*********************************************

RESULT OF BUILD
  AVERAGE DISSIMILARITY =          2.583

FINAL RESULTS
  AVERAGE DISSIMILARITY =          2.507

  CLUSTERS
  NUMBER   MEDOID   SIZE      OBJECTS

    1       USA      5      BEL EGY FRA ISR USA

    2       ZAI      3      BRA IND ZAI

    3       CUB      4      CHI CUB USS YUG

CLUSTERING VECTOR
*****************

           1  2  3  3  1  1  2  1  1  3  3  2

CLUSTERING CHARACTERISTICS
**************************

 CLUSTER    3 IS ISOLATED,
        WITH DIAMETER  =       4.50 AND SEPARATION =         5.25
        THEREFORE IT IS AN L*-CLUSTER.

 THE NUMBER OF ISOLATED CLUSTERS =    1

 DIAMETER OF EACH CLUSTER
      5.00      5.00      4.50

 SEPARATION OF EACH CLUSTER
      4.67      4.67      5.25

 AVERAGE DISSIMILARITY TO EACH MEDOID
      2.40      2.61      2.56

 MAXIMUM DISSIMILARITY TO EACH MEDOID
      4.50      4.83      3.83
```

Figure 8 Output of the 12 countries example for $k = 3$.

```
                                       *****************
                                       *               *
                                       *  SILHOUETTES  *
                                       *               *
                                       *****************

                      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                      . . . . . . . . . . . . . . . . . . . . . . . . . .
                      0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                      0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0
CLU   NEIG   S(I)   I +++++++++++++++++++++++++++++++++++++++++++++++++++
                      +                                                 +
 1     2     .47   USA+***********************                          +
 1     2     .44   FRA+**********************                           +
 1     2     .42   BEL+*********************                            +
 1     2     .37   ISR+******************                               +
 1     2     .02   EGY+*                                                +
                      +                                                 +
 2     1     .28   ZAI+**************                                   +
 2     1     .25   BRA+************                                     +
 2     3     .17   IND+********                                         +
                      +                                                 +
 3     2     .48   CUB+************************                         +
 3     1     .44   USS+*********************                            +
 3     1     .31   YUG+***************                                  +
 3     2     .31   CHI+***************                                  +
                      +                                                 +
                      +++++++++++++++++++++++++++++++++++++++++++++++++++

                      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                      . . . . . . . . . . . . . . . . . . . . . . . . . .
                      0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                      0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0

         CLUSTER     1 HAS AVERAGE SILHOUETTE WIDTH    .34
         CLUSTER     2 HAS AVERAGE SILHOUETTE WIDTH    .24
         CLUSTER     3 HAS AVERAGE SILHOUETTE WIDTH    .38

         FOR THE ENTIRE DATA SET, THE AVERAGE SILHOUETTE WIDTH IS    .33

The output is on file : COUNTRY.RES
```

Figure 8 (*Continued*)

which means that cluster 2 is not very clearly separated from the other clusters. It appears that Zaire and Brazil are more inclined toward the Western countries because their neighbor is cluster 1, whereas India [which has a smaller value of $s(i)$] seems to be closer to cluster 3. In the third cluster there is also a lack of unanimity: USSR and Yugoslavia seem to have more resemblance to the Western countries, whereas Cuba and China appear to be closer to the developing ones.

Note that silhouettes only depend on the actual partition of the objects and not on the clustering algorithm that was used to obtain it. As a consequence, silhouettes could be used to improve the results of cluster analysis [for instance, by moving an object with negative $s(i)$ to its neighbor] or to compare the output of different clustering algorithms applied to the same data.

However, we think that the main usefulness of silhouettes lies in the interpretation and validation of cluster analysis results. Often silhouettes can prevent us from drawing the wrong conclusions. For instance, suppose the data set consists of some dense clusters that are far away from each other, but that we have set k too low. In this case, most clustering algorithms will combine some natural clusters in order to reduce the total number of groups to the specified value of k. Fortunately, the silhouette plot will often expose such artificial fusions. Indeed, joining different clusters will lead to large "within" dissimilarities and hence a large $a(i)$, resulting in small $s(i)$ values for the objects in such a conglomerate and yielding a narrow silhouette.

On the other hand, suppose that we have set k too high. Then some natural clusters have to be divided in an artificial way in order to conform to the specified number of groups. However, these artificial fragments will typically also show up through their narrow silhouettes. Indeed, the objects in such a fragment are on the average very close to the remaining part(s) of their natural cluster, and hence the "between" dissimilarities $b(i)$ will become very small, which also results in small $s(i)$ values.

This heuristic reasoning implies that the silhouettes should look best for a "natural" value of k. Therefore, we want the silhouettes to be as wide (or as dark) as possible. For each cluster, we have defined the *average silhouette width* as the average of the $s(i)$ for all objects i belonging to that cluster. This allows us to distinguish clear-cut from weak clusters in the same plot: Clusters with a larger average silhouette width are more pronounced. In Figure 8, we see that the average silhouette width of the second cluster is only 0.24, whereas the first and third clusters attain higher values.

We can also consider the *overall average silhouette width* for the entire plot, which is simply the average of the $s(i)$ for all objects i in the whole data set. In Figure 8 this yields 0.33. In general, each value of k will yield another overall average silhouette width $\bar{s}(k)$. One way to choose k appropriately is to select that value of k for which $\bar{s}(k)$ is as large as possible. For the 12 countries data, it turns out that the best choice in this respect is $k = 3$, hence the so-called silhouette coefficient (SC) equals 0.33. According to Table 4, this is interpreted as a weak structure, but indeed the features of this grouping will be confirmed when applying the clustering algorithms of Chapters 4, 5, and 6.

Let us now look at some rather extreme examples to obtain a better feeling for the meaning of silhouettes. First suppose we have eight objects that are divided over some very tight clusters, far away from each other. For instance, assume that five objects coincide with one geometrical point and the remaining three coincide with another geometrical point at a large distance from the first. This is an extremely sharp clustering structure, which should be recovered by any reasonable clustering algorithm when $k = 2$. The resulting silhouette plot in Figure 9 is as wide and as dark as possible. All $s(i)$ equal 1.00, so the overall average silhouette width attains its maximal value 1.00 (therefore, $k = 2$ is the best choice and the SC of

```
                              *****************
                              *               *
                              *  SILHOUETTES  *
                              *               *
                              *****************

                     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                     . . . . . . . . . . . . . . . . . . . . . . . . .
                     0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                     0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0
CLU   NEIG   S(I)   I +++++++++++++++++++++++++++++++++++++++++++++++++++
                      +                                                 +
 1     2    1.00   001+*************************************************+
 1     2    1.00   002+*************************************************+
 1     2    1.00   003+*************************************************+
 1     2    1.00   004+*************************************************+
 1     2    1.00   005+*************************************************+
                      +                                                 +
 2     1    1.00   006+*************************************************+
 2     1    1.00   007+*************************************************+
 2     1    1.00   008+*************************************************+
                      +                                                 +
                      +++++++++++++++++++++++++++++++++++++++++++++++++++
                     0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                     . . . . . . . . . . . . . . . . . . . . . . . . .
                     0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                     0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0

          CLUSTER    1 HAS AVERAGE SILHOUETTE WIDTH  1.00
          CLUSTER    2 HAS AVERAGE SILHOUETTE WIDTH  1.00

          FOR THE ENTIRE DATA SET, THE AVERAGE SILHOUETTE WIDTH IS   1.00
```

Figure 9 Silhouettes of an example in which eight objects are divided over two very tight clusters, for $k = 2$.

Table 4 equals 1.00). In general, a very large overall average silhouette width can be taken to mean that the algorithm has discovered a very strong clustering structure.

However, one must be careful with the interpretation of the last statement. For instance, a situation where the data set contains one far outlier is also an example of a strong clustering structure. Indeed, when the outlier is far enough, the other data look like a tight cluster by comparison. We may expect our clustering algorithm to provide the following picture for $k = 2$: one cluster containing just the outlier and the other cluster consisting of the bulk of the data. For instance, suppose seven objects have zero dissimilarities to each other and all have large dissimilarities to the eighth object, which is an outlier. The resulting silhouette plot is given in Figure 10, clearly separating object 8 from the rest. Because cluster 2 contains only a single object, its $s(i)$ was put equal to 0 by convention. In order to mark such singleton clusters in a more distinctive way, John Tukey (personal communication) suggested printing the number 1 in the rightmost column of the plot. The overall average silhouette width equals 0.88, which is very high (in general, one obtains $1 - 1/n$ where n is the total number of objects). Therefore, one should never merely accept a high overall average silhouette width at its face value, but also look at the graphical output itself

```
                                   ****************
                                   *              *
                                   *  SILHOUETTES *
                                   *              *
                                   ****************

                         0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                         . . . . . . . . . . . . . . . . . . . . . . . . . .
                         0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                         0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0
CLU  NEIG  S(I)       I +++++++++++++++++++++++++++++++++++++++++++++++++++
                         +                                                 +
 1    2   1.00      001+**************************************************+
 1    2   1.00      002+**************************************************+
 1    2   1.00      003+**************************************************+
 1    2   1.00      004+**************************************************+
 1    2   1.00      005+**************************************************+
 1    2   1.00      006+**************************************************+
 1    2   1.00      007+**************************************************+
                         +                                                 +
 2    1    .00      008+                                                 1+
                         +                                                 +
                        +++++++++++++++++++++++++++++++++++++++++++++++++++
                         0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                         . . . . . . . . . . . . . . . . . . . . . . . . . .
                         0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                         0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0

      CLUSTER      1 HAS AVERAGE SILHOUETTE WIDTH  1.00
      CLUSTER      2 HAS AVERAGE SILHOUETTE WIDTH   .00

      FOR THE ENTIRE DATA SET, THE AVERAGE SILHOUETTE WIDTH IS   .88
```

Figure 10 Silhouettes of a data set containing a distant outlier, for $k = 2$.

to find out what caused the large value. Depending on the subject matter and the task at hand, one might want to put the outlier aside for further investigation and run the clustering algorithm again on the remaining data.

When the clustering algorithm does not succeed in finding any "natural" clustering, the overall average silhouette width tends to become very low. An extreme case is when all dissimilarities between pairs of objects equal the same positive constant, so all off-diagonal entries of the dissimilarity matrix are identical. (In the case of Euclidean distances, this happens with the n vertices of a regular simplex in $n - 1$ dimensions.) In such a situation no clustering is more natural than any other, so there is a total absence of clustering structure. Whatever the value of k and whatever clustering algorithm is used, all $s(i)$ will be 0 [as well as the overall average silhouette width $\bar{s}(k)$ and the silhouette coefficient SC] and the silhouette plot (see Figure 11) stays completely empty. In actual applications one sometimes does encounter cases where even the largest value of $\bar{s}(k)$ is very small, pointing to a lack of clustering structure.

Figure 12 shows a plot of the well-known Ruspini data (1970). This data set consists of 75 points and was originally used by Ruspini in order to illustrate fuzzy clustering techniques. The actual coordinates, given in Table 6, were communicated to us by G. Libert. The points make up four groups, A, B, C, and D, as indicated in the plot. Because this example is two-

```
                                 *****************
                                 *               *
                                 *  SILHOUETTES  *
                                 *               *
                                 *****************

                        0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                        . . . . . . . . . . . . . . . . . . . . . . . . . .
                        0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                        0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0
CLU  NEIG  S(I)     I   +++++++++++++++++++++++++++++++++++++++++++++++++++++
                        +                                                   +
 1    2    .00    001+                                                      +
 1    2    .00    002+                                                      +
 1    2    .00    003+                                                      +
 1    2    .00    004+                                                      +
 1    2    .00    005+                                                      +
 1    2    .00    006+                                                      +
 1    2    .00    007+                                                      +
                        +                                                   +
 2    1    .00    008+                                                     1+
                        +                                                   +
                        +++++++++++++++++++++++++++++++++++++++++++++++++++++
                        0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
                        . . . . . . . . . . . . . . . . . . . . . . . . . .
                        0 0 0 1 1 2 2 2 3 3 4 4 4 5 5 6 6 6 7 7 8 8 8 9 9 0
                        0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0 4 8 2 6 0

      CLUSTER    1 HAS AVERAGE SILHOUETTE WIDTH    .00
      CLUSTER    2 HAS AVERAGE SILHOUETTE WIDTH    .00

      FOR THE ENTIRE DATA SET, THE AVERAGE SILHOUETTE WIDTH IS    .00
```

Figure 11 Silhouettes of a data set without any clustering structure, for $k = 2$.

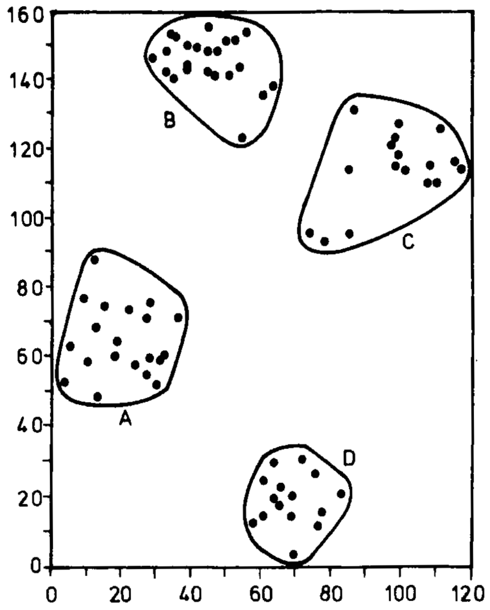


Figure 12 Plot of the two-dimensional data set of Ruspini. The loops are merely drawn to indicate four groups of points.

Table 6 Coordinates of the Ruspini Data

x	y	x	y	x	y
4	53	41	150	98	124
5	63	38	145	99	119
10	59	38	143	99	128
9	77	32	143	101	115
13	49	34	141	108	111
13	69	44	156	110	111
12	88	44	149	108	116
15	75	44	143	111	126
18	61	46	142	115	117
19	65	47	149	117	115
22	74	49	152	70	4
27	72	50	142	77	12
28	76	53	144	83	21
24	58	52	152	61	15
27	55	55	155	69	15
28	60	54	124	78	16
30	52	60	136	66	18
31	60	63	139	58	13
32	61	86	132	64	20
36	72	85	115	69	21
28	147	85	96	66	23
32	149	78	94	61	25
35	153	74	96	76	27
33	154	97	122	72	31
38	151	98	116	64	30

dimensional, we can now compare the silhouettes to the structure that we perceive by the naked eye. Figure 13 contains the silhouette plots of the k-medoid partitions with $k = 2, \ldots, 6$ (making use of Euclidean distance).

In the case of $k = 2$, one cluster is formed as the union of A with D, whereas the second combines B and C. As we saw before, artificial fusions are penalized by narrow silhouettes. For $k = 3$ we see that clusters B and C are found, but A and D still stick together. The corresponding silhouette plot shows clearly that both B and C are more pronounced than the union of A with D. For $k = 4$ the "right" solution is found, which leads to four silhouettes of about the same good quality. When $k = 5$ is imposed, the algorithm splits C into two parts. The second part contains the three lowest points of C (as viewed in Figure 12), that is, the three points of C with smallest y coordinates. This trio has a rather prominent silhouette and

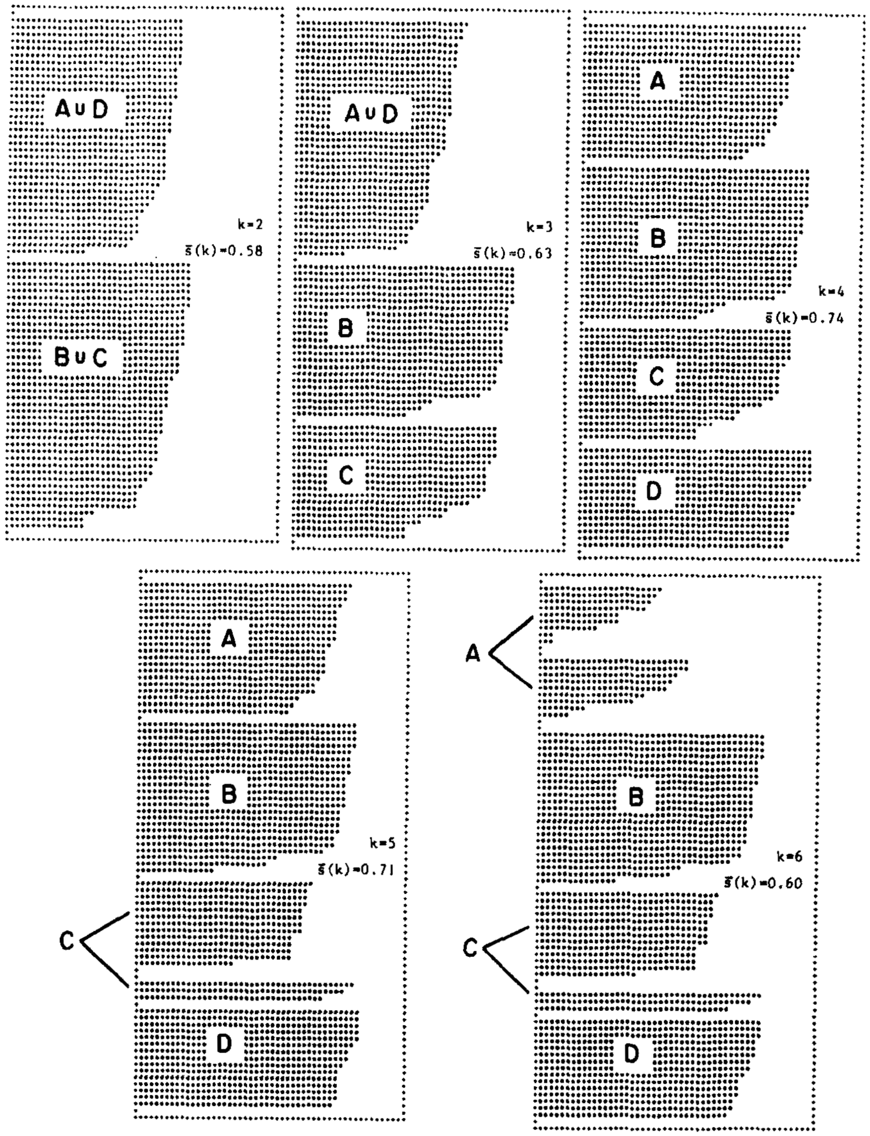


Figure 13 Silhouette plots of the Ruspini data, for *k* ranging from 2 to 6.

indeed some people consider it as a genuine cluster (see Delattre and Hansen, 1980). However, the silhouette of the major part of C becomes somewhat less wide because this cluster is not so well separated from the three-point one [indeed, one object has an $s(i)$ value of about zero because it lies rather close to the three-point cluster and therefore holds an intermediate position]. The last case ($k = 6$) leads to a more dramatic effect: Cluster A is being split up in an artificial way and consequently both parts obtain narrow silhouettes. For $k = 7, \dots, 74$ the results are still worse.

In conclusion, the silhouette plots tell us that a partition into $k = 4$ clusters is probably most natural. Indeed, the overall average silhouette width $\bar{s}(k)$ is largest for this value (even if one tries all values of k ranging from 2 to 74). The second best $\bar{s}(k)$ is attained for $k = 5$ and the silhouette plot shows us the advantages and disadvantages of the corresponding clustering.

*4 MORE ON THE ALGORITHM AND THE PROGRAM

4.1 Description of the Algorithm

In the description of the k-medoid method given in Section 1, we minimize the *average* dissimilarity of objects to their closest representative object. However, in the program itself we prefer to minimize the *sum* of these dissimilarities, which is mathematically equivalent but gives rise to more accurate calculations. Therefore, from now on we shall only talk about the minimization of this sum.

The algorithm we are using in PAM consists of two phases. In a first phase, called BUILD, an initial clustering is obtained by the successive selection of representative objects until k objects have been found. The first object is the one for which the sum of the dissimilarities to all other objects is as small as possible. This object is the most centrally located in the set of objects. Subsequently, at each step another object is selected. This object is the one which decreases the objective function as much as possible. To find this object, the following steps are carried out:

1. Consider an object i which has not yet been selected.
2. Consider a nonselected object j and calculate the difference between its dissimilarity D_j with the most similar previously selected object, and its dissimilarity $d(j, i)$ with object i.
3. If this difference is positive, object j will contribute to the decision to select object i. Therefore we calculate

$$C_{ji} = \max\left(D_j - d(j, i), 0\right)$$

4. Calculate the total gain obtained by selecting object i:

$$\sum_j C_{ji}$$

5. Choose the not yet selected object i which

$$\underset{i}{\text{maximizes}} \sum_j C_{ji}$$

This process is continued until k objects have been found. In the second phase of the algorithm (called SWAP), it is attempted to improve the set of representative objects and therefore also to improve the clustering yielded by this set. This is done by considering all pairs of objects (i, h) for which *object i has been selected and object h has not*. It is determined what effect is obtained on the value of the clustering when a swap is carried out, i.e., when object i is no longer selected as a representative object but object h is. Here we should recall that in this subsection, the value of a clustering determined by k representative objects is defined as the sum of dissimilarities between each object and the most similar representative object.

To calculate the effect of a swap between i and h on the value of the clustering, the following calculations are carried out (steps 1 and 2):

1. Consider a nonselected object j and calculate its contribution C_{jih} to the swap:
   - **a.** If j is more distant from both i and h than from one of the other representative objects, C_{jih} is zero.
   - **b.** If j is not further from i than from any other selected representative object ($d(j, i) = D_j$), two situations must be considered:
   - **b1.** j is closer to h than to the second closest representative object

$$d(j, h) < E_j$$

where E_j is the dissimilarity between j and the second most similar representative object. In this case the contribution of object j to the swap between objects i and h is

$$C_{jih} = d(j, h) - d(j, i)$$

   - **b2.** j is at least as distant from h than from the second closest representative object

$$d(j, h) \geq E_j$$

In this case the contribution of object j to the swap is

$$C_{jih} = E_j - D_j$$

It should be observed that in situation b1 the contribution C_{jih} can be either positive or negative depending on the relative position of objects j, h, and i. Only if object j is closer to i than to h is the contribution positive, which indicates that the swap is not favorable from the point of view of object j. On the other hand, in situation b2 the contribution is always positive because it cannot be advantageous to replace i by an object h further away from j than from the second closest representative object.

c. j is more distant from object i than from at least one of the other representative objects but closer to h than to any representative object. In this case the contribution of j to the swap is

$$C_{jih} = d(j, h) - D_j$$

2. Calculate the total result of a swap by adding the contributions C_{jih}:

$$T_{ih} = \sum_j C_{jih}$$

In the next steps it is decided whether to carry out a swap.

3. Select the pair (i, h) which

$$\underset{i,h}{\text{minimizes}}\, T_{ih}$$

4. If the minimum T_{ih} is negative, the swap is carried out and the algorithm returns to step 1. If the minimum T_{ih} is positive or 0, the value of the objective cannot be decreased by carrying out a swap and the algorithm stops.

Note that as all potential swaps are considered, the results of the algorithm do not depend on the order of the objects in the input file (except in case some of the distances between objects are tied).

4.2 Structure of the Program

The program PAM is written in Fortran and consists of approximately 1100 statement lines. Details about portability are given in the Appendix.

The program consists of a main unit, one function, and eight subroutines. In this section the purpose of each of these units is outlined.

The MAIN unit consists of the following parts:

- dimensions of the arrays
- setting the maximum values of the number of objects, the total number of variables, and the number of variables which can be used in a single analysis
- defining the unit for data input (called LUA) and the two units for output (LUB for the output of results and LUC for storing data entered by keyboard); in the present version they are given the values 1, 2, and 3
- call of the subroutine ENTR, which contains the interactive part of the program
- examination of objects and variables for missing values
- calls of the subroutines STAND (standardization) and DYSTA (computation of dissimilarities); these subroutines are associated with the input of measurements. After their input or calculation, the dissimilarities are stored in an array called DYS. In order to save memory space only the lower triangular part of the dissimilarity matrix is stored (an example is given in Figure 14). We chose the *lower* half matrix because this makes it possible to add new objects by simply adding some lines to the input file.
- parts *a*, *b*, and *c* of the output (see Section 2.2)
- a DO loop (DO 140 KK = KBEG,KEND) in which the subroutines BSWAP, CSTAT, and DARK are called; KK is the variable in which the number of clusters is stored (the subroutine DARK is called if graphical output was requested by the user, except if KK equals one or *n*)

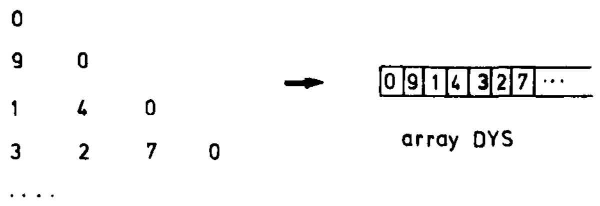


Figure 14 Illustration of the way dissimilarities are stored in a one-dimensional array. The first entry of the array is always 0 for convenience.

Subroutine ENTR: This subroutine controls the interactive use of the program and the input and output of data. It consists of the following parts:

- input of data specifications and chosen options
- if required by the user, input of labels of objects and information concerning missing values
- output of special messages
- output of a table with a recapitulation of data specifications and chosen options
- input of dissimilarities or measurement values

Subroutine QYN: Allows input of the answer to a yes–no question concerning options of the program (only the first letter of the answer is read and both upper and lowercase answers are allowed).

Function MEET: When a dissimilarity between objects L and J is needed, function MEET(L, J) gives the index of DYS where this dissimilarity is stored:

$$d(L, J) = \mathrm{DYS}(\mathrm{MEET}(L, J))$$

Subroutine NWLAB: This subroutine, which is accessed from the subroutine ENTR, is only called if no labels are provided. In it labels of objects are constructed:

$$001 \quad 002 \quad 003 \quad \cdots$$

They are stored in an integer matrix LAB(L, J) ($L = 1, 2, 3$ and $J = 1, 2, \ldots, n$) using the characters $0, 1, 2, \ldots, 9$ stored in a character type variable (NUM).

Subroutine STAND: In this subroutine, which is only called if standardization is requested, new standardized measurements are computed.

Subroutine DYSTA: Computes Euclidean or Manhattan distances (these are the dissimilarities that will be used in the clustering).

Subroutine BSWAP: Performs the clustering. It consists of two parts in which the BUILD and SWAP techniques are used.

Subroutine CSTAT: Gives the numerical output concerning each partition (except for the average dissimilarities, which are already computed in the subroutine BSWAP).

Subroutine DARK: Gives the graphical output for each partition.

Let us now investigate how the program can be adapted to special situations. Right after the dimension statements, four variables (MAXNN, MAXTT, MAXPP, and MAXHH) are given values, which make it possible to use the program for clustering data sets of different sizes. MAXNN and MAXTT are the maximum numbers of objects and variables in the data set, whereas MAXPP is the maximum number of variables that are actually used in the computations. These values can be freely chosen by the user taking into account their effect on the storage requirements of the program. For example when clustering 50 objects measured on 10 variables (MAXNN = 50, MAXTT = 10, MAXPP = 10) the EXE file occupies 108,644 bytes, whereas for 100 objects and 20 variables (MAXNN = 100, MAXTT = 20, MAXPP = 20) 136,308 bytes are necessary. Often trial and error seems to be the easiest way of determining the maximum size of problems which can be solved on a particular computer.

The fourth variable defined after the dimensions is MAXHH, which gives the size of the array DYS and must be set equal to MAXNN(MAXNN − 1)/2 + 1.

The four variables MAXNN, MAXTT, MAXPP, and MAXHH are used as dimensions of arrays (vectors and matrices) in the subroutines. In the main program the dimensions of the arrays in the first five dimension statements must be set equal to the values of these four parameters. In Section 2 the following values were used:

$$\begin{aligned}\text{MAXNN} &= 100\\ \text{MAXTT} &= 80\\ \text{MAXPP} &= 20\\ \text{MAXHH} &= 4951\end{aligned}$$

If the program is used with input of dissimilarities, it is recommended to give a value of at least 1 to MAXTT and MAXPP.

Another important aspect of using the program on different computers is the time necessary to solve given problems. As an indication of the speed of the program, the times (in minutes) on an IBM/XT with an 8087 accelera-

Table 7 Computation Times (in minutes) on an IBM-XT with 8087 Accelerator for a Set of Randomly Generated Problems of Increasing Sizes

Objects	Variables	Clusters	Time
20	2	5	0.30
40	2	5	1.17
60	2	5	1.87
80	2	5	3.20
100	2	5	6.17

tor for a set of randomly generated problems of increasing sizes are given in Table 7.

Specific information concerning compilation of the program is given in the Appendix.

*5 RELATED METHODS AND REFERENCES

5.1 The k-Medoid Method and Optimal Plant Location

When constructing partitions with a fixed number k of clusters, it is often assumed that there exists a function which measures the quality of different clusterings of the same data set. This is also the case for the clustering method used in the program PAM. This method is based on a location model (the k-median model) with the following general formulation: Given a finite number of users, whose demands for a given service are known and must be satisfied, and given a finite set of possible locations among which k must be chosen for the location of service centers, select the locations in such a way as to minimize the total distance (or equivalently the average distance) travelled by the users. In the formulation used in clustering, the sets of users and of possible locations coincide and both correspond to the set of objects to be clustered. The location of a center is interpreted as the selection of an object as a representative object (or centrotype, median, or medoid) of a cluster. The distance travelled by a user corresponds to the dissimilarity between an object and the representative object of the cluster to which it belongs. The idea to use this model for cluster analysis was introduced by Vinod (1969) and later also discussed by Rao (1971), Church (1978), and Mulvey and Cowder (1979).

In the mathematical formulation of the k-medoid model the following notations are used:

The set of objects is denoted by X:

$$X = \{x_1, x_2, \dots, x_n\}$$

The dissimilarity between objects x_i and x_j (also called objects i and j) is denoted by $d(i, j)$.

A solution of the model is determined by two types of decisions:

The selection of objects as representative objects in clusters: y_i is defined as a 0-1 variable, equal to 1 if and only if object i ($i = 1, 2, \dots, n$) is selected as a representative object.

The assignment of each object j to one of the selected representative objects: z_{ij} is a 0-1 variable, equal to 1 if and only if object j is assigned to the cluster of which i is the representative object (and also the medoid).

The corresponding optimization model, which was first proposed by Vinod (1969), can then be written as:

$$\text{minimize} \sum_{i=1}^{n} \sum_{j=1}^{n} d(i,j) z_{ij} \tag{8}$$

subject to

$$\sum_{i=1}^{n} z_{ij} = 1, \qquad j = 1,2,\dots,n \tag{9}$$

$$z_{ij} \le y_i, \qquad i,j = 1,2,\dots,n \tag{10}$$

$$\sum_{i=1}^{n} y_i = k, \qquad k = \text{number of clusters} \tag{11}$$

$$y_i, z_{ij} \in \{0,1\}, \qquad i,j = 1,2,\dots,n \tag{12}$$

Constraints (9) express that each object j must be assigned to a single representative object. They imply [together with constraints (12)] that for a given j, one of the z_{ij} is equal to 1 and all others are 0. Constraints (10) ensure that an object j can only be assigned to an object i if this last object has been selected as a representative object. Indeed, if this is not the case, then y_i is 0 and the constraint [together with constraints (12)] implies that all z_{ij} are 0. If i is a representative object, then all the z_{ij} (for this i) can be either 0 or 1. Equation (11) expresses that exactly k objects are to be chosen as representative objects. As the clusters are formed by assigning each object to the most similar representative object, there will be exactly k nonempty clusters. (In case of ties the object is assigned to the representative object which was entered first.) Equation (9) implies that the dissimilarity between an object j and its representative object is given by

$$\sum_{i=1}^{n} d(i,j) z_{ij}$$

As all objects must be assigned, the total dissimilarity is given by

$$\sum_{j=1}^{n} \sum_{i=1}^{n} d(i,j) z_{ij} \tag{13}$$

which is the function to be minimized in the model.

The algorithm discussed in Section 4 yields a good but not necessarily optimal solution to model (8) to (12). A branch and bound method which always yields an optimal solution was discussed in Massart et al. (1983). Unfortunately this approach is only computationally feasible for relatively small problems with up to 50 or 60 objects. Another method was used by Klastorin (1985) in a comparison of several clustering algorithms. It is based on an algorithm proposed by Erlenkotter (1978) for the simple plant location problem.

Usually one does not know the number of clusters present in a data set. Because most partitioning methods provide a fixed number k of clusters, one must apply them for several values of k in order to find the most meaningful clustering. A possible way of selecting a value of k is by means of the silhouette coefficient (see Section 2). Another way is to search for sets of objects that persistently stay together in the clusterings obtained for several values of k. This approach has been investigated by Massart et al. (1983) and Plastria (1986). Another method, based on the bootstrap technique, was proposed by Moreau and Jain (1987). For one-dimensional observations, Müller and Sawitzki (1987) chose k by estimating the number of modes.

5.2 Other Methods Based on the Selection of Representative Objects

The objective of the k-medoid method is to minimize the sum of dissimilarities between the objects and their representative objects. This objective is suitable if a total value is liable to give a correct description of the structure being investigated.

One of the possible alternative aims is to try to minimize the largest distance from any object to the representative object of its cluster. The maximum dissimilarity between an object and its representative object is given by

$$\max_{j=1,\ldots,n} \sum_{i=1}^{n} d(i,j) z_{ij}$$

and the model of minimizing the maximum dissimilarity can be written as

$$\text{minimize} \max_{j=1,\ldots,n} \sum_{i=1}^{n} d(i,j) z_{ij}$$

subject to constraints (9) to (12). This is called the *k-center model* and, like the k-medoid model, it finds its equivalent in location theory and also in graph theory. Algorithms for the k-center model have been proposed by

Hakimi (1965). A characteristic of the clusterings obtained with the k-center model is that they are on average looser, but on the other hand they will never contain very loose clusters.

Another model using representative objects is the *covering model.* In this model the number of clusters is not fixed, but each object must be within a given distance D of its representative object. The objective is to minimize the number of representative objects necessary to achieve this aim. As in the k-medoid method, once the representative objects have been selected the clustering is obtained by assigning each object to its most similar selected representative object. The model can be stated as

$$\text{minimize} \sum_{i=1}^{n} y_i \tag{14}$$

subject to

$$\sum_{i=1}^{n} z_{ij} = 1, \qquad j = 1,2,\ldots,n \tag{15}$$

$$z_{ij} \leq y_i, \qquad i, j = 1,2,\ldots,n \tag{16}$$

$$\sum_{i=1}^{n} d(i,j) z_{ij} \leq D, \qquad j = 1,2,\ldots,n \tag{17}$$

$$y_i, z_{ij} \in \{0,1\}, \qquad i, j = 1,2,\ldots,n \tag{18}$$

Constraints (17) express that each object j must lie within a maximum dissimilarity D of its representative object.

There are two reasons why the k-medoid, k-center, and covering models have been formulated as 0-1 linear programs such as (8) to (12) and (14) to (18). The first is that such a formulation can lead to their optimal solution using methods such as branch and bound. Unfortunately, this approach is only feasible for small problems. Another reason is that such a formulation makes it possible to quite easily impose a particular structure on the clustering being sought. For example, adding a simple constraint can limit the number of objects in the clusters or can prevent some of the objects from being selected as representative objects. Integer programming techniques can then be used to find optimal or good solutions. For a survey of integer programming see, for example, Garfinkel and Nemhauser (1972).

5.3 Methods Based on the Construction of Central Points

In the methods considered in the previous section each cluster is characterized by a centrally located object called the representative object. In case the objects are defined by measurement values, there exists an alternative way of characterizing a cluster, namely by its *centroid*. This approach has been used extensively in the literature as a basis for clustering algorithms.

The centroid of a cluster v is defined as a point in p-dimensional space found by averaging the measurement values along each dimension (variable). For instance, its fth coordinate is

$$\bar{x}_f(v) = \frac{1}{n_v} \sum_{i \in C_v} x_{if}$$

where C_v represents the set of indices of cluster v (which contains n_v objects). Therefore, the centroid of cluster v is given by

$$\bar{x}(v) = \left(\bar{x}_1(v), \bar{x}_2(v), \dots, \bar{x}_p(v)\right)$$

We note two important facts: The centroid does not have to be one of the objects in the original data set, and it is not defined when the data are dissimilarities not based on interval-scaled measurement values.

A possible measure for the tightness of a cluster is the error sum of squares, defined as the sum of squares of (Euclidean) distances between the objects of a cluster and its centroid:

$$\mathrm{ESS}(C_v) = \sum_{i \in C_v} \sum_{f=1}^{p} \left(x_{if} - \bar{x}_f(v)\right)^2 \tag{19}$$

The error sum of squares of the whole clustering is

$$\mathrm{ESS} = \sum_{v=1}^{k} \mathrm{ESS}(C_v) \tag{20}$$

Therefore a possible approach to the clustering problem is to look for a partition into k nonempty subsets that minimize the error sum of squares. Methods that try to achieve this aim are called *variance minimization techniques*. (Observe that a similar objective could be attained in a variant of the k-medoid method, by entering the *squares* of the dissimilarities as input to PAM.)

Many variance minimization techniques have been proposed, but we will limit ourselves to two widely used ones. Let us start with a relatively simple method suggested by Forgy (1965). This method consists of the following steps:

1. An initial partition of the objects into k nonempty subsets is randomly generated. Then go to step 2. The method can also start with a

set of central points, called seed points, in which case one proceeds with step 3.

2. Compute seed points as the centroids of the clusters of the current partition.
3. Assign each object to the cluster with the nearest seed point. The seed points remain fixed for an entire pass through the set of objects. If this is the first pass through the objects, go to step 2. In subsequent passes the clustering is compared to the previous clustering and if no change in the assignment of objects has occurred, stop. If there has been a change, go to step 2.

An efficient program for the method can ensure that an object is only moved if it is strictly closer to a different seed point and therefore the error sum of squares must decrease. As the number of possible different clusterings is finite, this ensures that the method converges. However, it cannot be foreseen how many repetitions of steps 2 and 3 will be required to reach a (locally) optimal solution. According to Anderberg (1973), five to ten iterations (passes through steps 2 and 3) are sufficient in most practical situations. A shortcoming of this method, and of many other variance minimization techniques, is that during step 3 it is possible that all objects of a cluster are assigned to other clusters, and at the same time no other objects are assigned to the centroid of this cluster. In this way the algorithm sometimes ends up with less than k clusters. (Note that this problem cannot occur for the k-medoid method.)

In a variant of this method, proposed by Jancey (1966), the new seed points of the clusters are found (in step 2) by reflection of the old seed points with respect to the centroids. In this way the steps taken (by the seed points) are amplified and a locally optimal solution is usually found in fewer iterations.

The *k-means method* of MacQueen (1967) is probably the most widely applied nonhierarchical clustering technique. It is very similar to Forgy's method, the only difference being that each time an object changes clusters the centroids of both its old and new cluster are recalculated. This can be done quite easily using an update equation for the centroid coordinates. If object i is moved from cluster v to cluster w, the coordinates of the new centroids are given by

$$\bar{x}_f(v') = \frac{1}{n_v - 1}\left(n_v \bar{x}_f(v) - x_{if}\right)$$

and

$$\bar{x}_f(w') = \frac{1}{n_w + 1}\left(n_w \bar{x}_f(w) + x_{if}\right)$$

where v' and w' represent the new clusters.

As in Forgy's method, an object is only moved if it is nearer to the new centroid. Therefore the sum of squares of distances to the centroids must decrease. Furthermore, in general, the centroids of the newly formed clusters will not coincide with the old ones. As the centroid is the point which minimizes the sum of squares of distances, the total sum of squares will decrease by an even larger quantity. An interesting feature of MacQueen's method is that the number of clusters cannot change. The reason for this is that if only two objects are left in a cluster and one of these is removed, the centroid of the new cluster (which is now a singleton) coincides with its object. This object then cannot be assigned to a different cluster. Finally, a drawback of this method (as well as those of Forgy and Jancey) is that the results (sometimes strongly) depend on the order of the objects in the input file, unlike the algorithm in PAM.

Many more variance minimization algorithms have been published, such as the method of Friedman and Rubin (1967) which has frequently been applied. However, it is not our aim to give a complete enumeration of these algorithms.

One of the problems with variance minimization methods is the effect of *outliers* on the value of the function to be minimized (formula 20). This has been our principal motivation for selecting the k-medoid method for PAM. Another reason was the possibility of obtaining representative objects in the clusters, which is very appealing and useful in a wide range of applications. It should be added that a method was proposed by Cooper (1963) which makes use of the criterion of the k-medoid method [Eq. (8)], but in which the representative objects are replaced by general p-dimensional points. This method and related ones are discussed by Massart and Kaufman (1983) and by Späth (1980). In another variant, proposed by Diday (1971, 1974), more general central regions of clusters are considered, which can consist of one or more objects or p-dimensional points; they are the so-called kernels.

To conclude the discussion of variance minimization, we will consider a method that combines the search for a tight clustering (with small error sum of squares) with the determination of an "optimal" number of clusters. Almost none of the partitioning methods tackles the problem of finding an adequate (natural) number of clusters. Ball and Hall (1965) proposed a method called ISODATA the main features of which are:

The method starts with a clustering into a given number k of clusters, according to the method of Forgy.

In a second step, outliers and very small clusters are eliminated; they are disregarded for the remainder of the method.

Then perform either a lumping (fusion) or splitting of one of the clusters. This is done according to the following rules:

- perform a lumping if the current number of clusters is more than $2k$
- perform a splitting if the current number of clusters is less than $k/2$
- otherwise alternate between lumping and splitting
- stop if the same clustering is obtained twice.

Return to the first step with the newly obtained number of clusters (which replaces k), unless the user-specified maximum number of iterations is reached.

During lumping, clusters are merged with centroids nearer to each other than the lumping threshold, while during splitting clusters are divided if the average sum of squares of dissimilarities is greater than the splitting threshold. If no clusters meet the desired criteria, the algorithm returns to the first step. ISODATA requires a considerable amount of input from the user, which implies that the user already has some knowledge of the structure of the data set. According to Anderberg (1973), who discusses several versions of ISODATA and gives many references, the method is too elaborate to be used without periodic human intervention.

The methods of Ball and Hall (1965), Forgy (1965), Jancey (1966), and MacQueen (1967) are among the numerous methods implemented in the CLUSTAN package (Wishart, 1978).

The literature also contains some interesting special cases of variance minimization. Several algorithms have been proposed for the one-dimensional case in which a much smaller number of partitions has to be considered. The number of partitions that must be considered when partitioning a set of objects is a type 2 Stirling number. For n objects and k clusters it is of the order of $k^n/k!$, for example, 50 objects can be clustered into five clusters in approximately 7.4×10^{32} different ways. A complete enumeration of all partitions is clearly impossible except for very small clustering problems. Fisher (1958) has shown that for one-dimensional data the optimal clusters for the error sum of squares criterion must be contiguous; this of course reduces drastically the number of partitions to be considered. Fisher proposed an algorithm for this case. He also extended the method to multidimensional data for which there exists a known complete ordering of the objects. This method is especially useful for the analysis of time series. Work along the same lines can be found in the paper by Vinod (1969). Another interesting special case is the clustering into two subsets ($k = 2$), which can serve as a basis for a divisive hierarchical method (see Chapter 6). An extensive discussion of these situations as well as many other nonhierarchical methods can be found in Bock (1974).

5.4 Some Other Nonhierarchical Methods

In two or three dimensions, an observer can distinguish clusters mainly on the basis of an impression of the density of objects in different areas. In particular, a cluster is often defined as a region in which the density of objects is locally higher than in other regions. Many clustering algorithms have been proposed which use the density of objects to construct clusters. Two general approaches have been used for a definition of this concept. In a first type of method, the density near an object is defined as the number of objects within a sphere (around this object) with a fixed radius R. Sometimes the objects are weighted by their dissimilarity to the central object being considered. In another approach, the density is defined as the inverse of the dissimilarity to the Tth nearest object or the inverse of the average dissimilarity to the T nearest objects. In both approaches there is an arbitrary element. In the first it is the radius R of the sphere and in the second it is the number T of objects considered. A too large value of R or of T will lead to a too small number of clusters. The most frequently applied density seeking method is Wishart's mode analysis (Wishart, 1969a) which attempts to isolate and remove outliers and to detect dense regions of space in which clusters of objects can be found. Based on this approach Wishart proposes two algorithms, one hierarchical and the other nonhierarchical. Another approach, used by Wolfe (1970) and Coomans and Massart (1981), is based on the search for k multivariate distributions and the determination of the most probable population for each object. An extensive discussion of density seeking methods can be found in the book by Chandon and Pinson (1981).

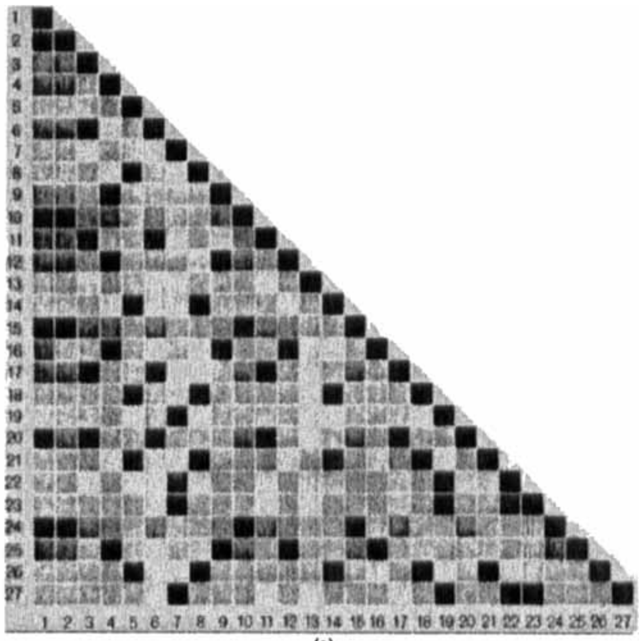
The vast majority of clustering methods are designed to classify a set of objects characterized by measurements of one or more variables. The same methods can be used to cluster a set of *variables* using their values on a given set of objects. In this situation one uses other types of dissimilarity measures, such as those based on correlation coefficients, but once the dissimilarities have been calculated the same algorithms can be used. However, the separate classification of objects (or variables) is sometimes inadequate for finding relationships that can exist between groups of objects and groups of variables. For example, it is quite probable that the clustering of the objects would be different for different subsets of variables; this picture might become blurred if all variables are considered together. In order to discover relationships between subsets of objects and subsets of variables, methods are required that cluster objects and variables simultaneously. These are called two-way or block clustering methods. Hartigan (1972, 1975) and Good (1965) have proposed hierarchical methods for block clustering, while Kaufman and Plastria (1981) have introduced a partitioning algorithm. The subject of block clustering is discussed in some detail by Chandon and Pinson (1981).

5.5 Why Did We Choose the k-Medoid Method?

As mentioned in Section 5.3, the objective chosen in the k-medoid method is appealing because it is more robust than the error sum of squares employed in most methods. (The maximal dissimilarity used in the k-center method is even less robust.) Furthermore it allows a good characterization of all clusters that are not too elongated and makes it possible to isolate outliers in most situations.

As will be seen in later chapters, one of the themes of this book is the preference given to methods based on *average dissimilarities* instead of sums of squares of dissimilarities. Indeed, the k-medoid method minimizes the average dissimilarity of the objects to the representative objects they are assigned to. Our preference for average distances is also apparent in the choice of standardization of the variables, in which the mean absolute deviation is used and not the more classical standard deviation (see Section 2 of Chapter 1). In many branches of univariate and multivariate statistics it has been known for a long time that methods based on the minimization of sums (or averages) of dissimilarities or absolute residuals (the so-called L_1 methods) are much more robust than methods based on sums of squares (which are called L_2 methods). The computational simplicity of many of the latter methods does not make up for the fact that they are extremely sensitive to the effect of one or more outliers. [The effect of error perturbation on clustering algorithms was already examined by Milligan (1980), but his study did not contain the k-medoid method.]

Also note that the clustering found by PAM does not depend on the order in which the objects are presented (except when equivalent solutions exist, which very rarely occurs in practice). This is not the case for many other algorithms described in this section. Also, if we ask for k clusters we do obtain exactly k clusters, and not less.


(a)

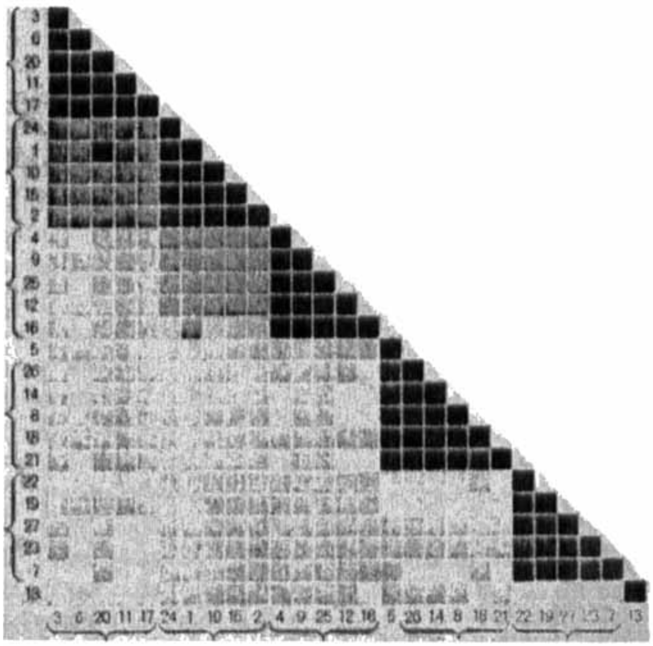

(b)

Figure 15 Shaded dissimilarity matrix of 27 objects: (a) In random order. (b) Ranked according to clusters (from Sokal, 1966).

The L_1 method described here is invariant with respect to translations and orthogonal transformations of the data points, but not with respect to affine transformations which change the interobject distances. However, we wanted to construct a method able to deal with general dissimilarity data, to which the notion of affine invariance does not apply. In fact, the affine invariant clustering methods in the literature make use of geometric notions (like centroids or tolerance ellipsoids) which do not exist for dissimilarity data. On the other hand, the medoids always exist, even when the data are only a collection of dissimilarities (see also Kaufman and Rousseeuw, 1987).

5.6 Graphical Displays

Whereas for hierarchical clustering (Chapters 5, 6, and 7) various versions of the dendrogram have been proposed and widely used to represent clustering results in a graphical way, for nonhierarchical methods relatively little has been done in the way of graphics. The main reason is that the tree-like structure yielded by hierarchical algorithms is suitable for an appealing graphical representation.

To display a partition, one of the approaches consists of a graphical representation of the matrix of dissimilarities between objects. The general idea consists of two phases. In the first phase the dissimilarities are replaced by symbols that give an impression of their magnitude. For example, black squares might represent small dissimilarities, white squares large dissimilarities, and shades of grey intermediate values. In a second phase, permutations of the objects are carried out (which change the rows and columns of the matrix) in such a way that the smallest dissimilarities are found close to the diagonal. Sokal (1966, p. 110) gave an interesting example, which is reproduced in Figure 15. A display like this can be generated automatically and drawn with a line printer. Recently, Gale, Halperin, and Costanzo (1984) proposed a more refined version based on so-called unclassed cloropleth mapping.

Wainer (1983) gives a survey of methods for displaying multivariate data, in which each object is represented by an icon (such as a polygon), made up of parts that vary in size or shape with the measured attributes. Also some techniques are mentioned for allowing tables to communicate the data structure in a more efficient way, such as rounding, reordering, and blocking. Applying the latter to the dissimilarities between 12 countries (of Table 5) yields Figure 16, given to us by Howard Wainer (personal communication).

Another approach is to represent a partition using a diagram in which the groups are displayed as circles. Lines between the groups denote their

	B	E	F	Is	US	Bzl	In	Z	Ch	Cu	U
Belgium											
Egypt	5										
France	2	5									
Israel	3	5	4								
USA	3	5	2	3							
Brazil	6	5	6	6	5						
India	6	5	6	6	6	5					
Zaire	5	5	6	6	6	3	5				
China	7	8	7	6	6	7	6	6			
Cuba	7	6	7	6	7	7	6	7	4		
USSR	6	6	6	7	6	7	6	7	4	3	
Yugoslavia	5	6	5	6	7	7	6	7	5	4	4

Figure 16 Dissimilarities between 12 countries, with reordered objects and rounded entries, blocked according to clustering structure.

global dissimilarity. In one of these techniques, proposed by Carmichael and Sneath (1969) and called the method of *taxometric maps*, the diameter of a circle representing a cluster is proportional to the diameter of the cluster. Clusters with only one object are represented by points. It is attempted to place the clusters in the map in such a way that the distances between them are proportional to their actual distance (as defined by the clustering algorithm). For the pairs of clusters for which this is possible, a straight full line is drawn. When the distance on the map exceeds the actual distance, the straight line is divided into a full part of correct length and dashes for the remainder, and when it is too small a V-shaped line of correct length is drawn. An example is given in Figure 17. Taxometric maps are complementary to silhouettes because they depict the relationship between clusters but not the behavior of the individual objects within those clusters.

With the silhouettes defined by Rousseeuw (1987) it is possible to obtain more information on the objects themselves. The height of the silhouette of a cluster is proportional to the number of objects, while its width gives a picture of its tightness with respect to the other clusters. Furthermore,

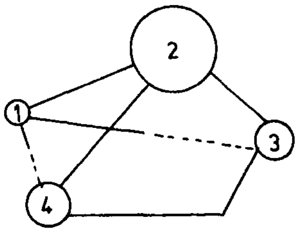


Figure 17 An example of a taxometric map.

information is immediately perceived concerning the individual objects. For example, a small (or even negative) value $s(i)$ denotes an object located at the outskirt of the cluster, whereas a value close to 1 shows the object to be centrally located. [In this way the $s(i)$ value can be compared with a correlation coefficient.] By looking at the neighbors one can find the objects located between two clusters. Additional information is given by the average silhouette width $\bar{s}(k)$ over the whole data set. This value can be used as a criterion to select the optimal number of clusters, but is by no means the only combination of $s(i)$ that could be computed for this purpose: One might also use transformations, medians, and so on. Like most validity coefficients, $\bar{s}$ could be used as an objective function for the clustering itself (that is, one might want to find a clustering that maximizes $\bar{s}$). Another idea would be to define $s(i)$ simply by $1 - a(i)/b(i)$, so $s(i)$ could become much smaller than -1, thereby penalizing misclassified points to a larger extent. Several other variants of the definition of $s(i)$ are possible. For instance, one could replace $a(i)$ and $b(i)$ of Eq. (6) by medians of dissimilarities (instead of average dissimilarities) in order to obtain more robustness. Also, when the clustering method is based on the selection of representative objects or the construction of central points, one could use the dissimilarities to these privileged entities in order to avoid the (lengthy) computation of average dissimilarities. However, we prefer the present definition because it only depends on the partition itself and not on the method used to construct it.

Silhouettes might be displayed in different ways, for instance, by using more sophisticated plotting devices instead of a line printer. One could also plot the negative $s(i)$ [we have not done so in PAM because very negative $s(i)$ rarely occur, and we wanted to use the width of the plot to its fullest advantage].

Recently, Edmonston (1986) proposed a graphical display that is very similar to the silhouette plot. It is called a clustergram and also contains a separate panel for each cluster. Each object is again represented by one printed line, in which two quantities are combined: the object's distance to the nearest object of another cluster (which is a kind of "neighbor") and the distance to the object's own cluster centroid.

The *distance graph* was proposed and used by Chen, Gnanadesikan, and Kettenring (1974) with further applications and developments by Cohen et al. (1977), Gnanadesikan, Kettenring, and Landwehr (1977, 1982), and Chambers and Kleiner (1982). For each cluster centroid, they plot the distances of every entity from that centroid (the symbol plotted is the cluster to which the entity was assigned). Figure 18 is an example of a distance graph. Although such a display gives an indication of the internal cohesiveness of a cluster, it does not allow the study of individual objects

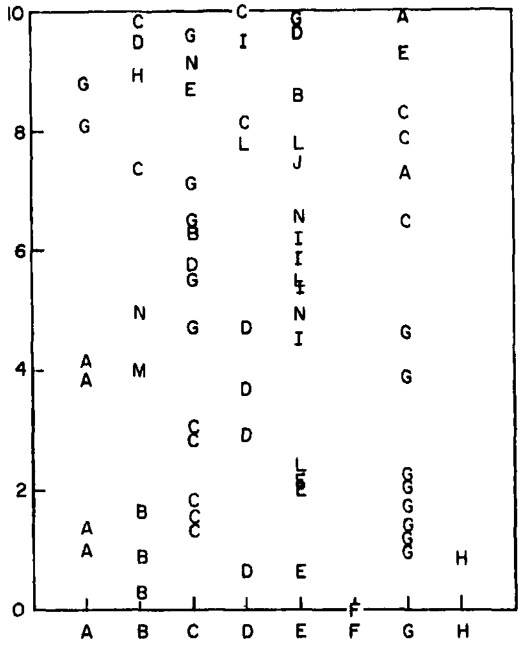


Figure 18 A distance graph of eight clusters (from Cohen et al., 1977). The centroids are listed on the horizontal axis. Distances of objects to these centroids are given on the vertical axis. The plotting symbol (A to H) identifies the cluster to which the object belongs.

because they remain anonymous in each list. Other variants display the (average) distances between the whole clusters, without further regard to the objects themselves. For a recent sociological application, see Andes (1985).

Another development is the introduction of *cluster validity profiles* (Bailey and Dubes, 1982). Although their name might suggest otherwise, the silhouettes proposed in this book are not related to these profiles. The latter displays are based exclusively on the ranks of the $n(n - 1)/2$ entries of the triangular dissimilarity matrix. They originate from a method to investigate the validity of a certain clustering by means of formal testing against some null hypothesis. The adopted null hypothesis is due to random graph theory and states that the matrix of ranks of dissimilarities is chosen randomly from a uniform distribution on all $[n(n - 1)/2]!$ possible rank matrices, supposing that there are no ties. A probability profile for a cluster shows, for each rank, the probability that an index of clustering is at least as good as the observed index (under the null hypothesis already specified). The resulting plots are drawn in function of dissimilarity ranks, whereas silhouettes are drawn in function of the objects of the cluster. Profiles differ from silhouettes in an essential way because they are used with ordinal dissimilarities whereas silhouettes are constructed from dissimilarities on a ratio scale. Also their roots are different: Profiles are based on hypothesis testing in a mathematical framework, whereas silhouettes are developed as data analytical tools which are useful mainly because of their simplicity and intuitive appeal.

Finally Dunn and Landwehr (1976, 1980) proposed graphical representations for the changes in cluster characteristics when two clusterings are carried out on the same data set (for example, at two different time periods or with two sets of variables). Another problem discussed by these authors is the representation of relationships between clusters and one or several exogeneous variables (that is, variables not employed in the clustering).

The topic of graphical representation of clustering results has so far received too little attention in the literature. Some books on clustering do not even mention the subject. The most comprehensive discussion can be found in a book by Everitt (1978) on graphical techniques for multivariate data. Some general methods for the graphical representation of multivariate data, like Andrews' plots (1972), biplots (Gabriel, 1971), Chernoff's faces (1973), and Wegman's parallel coordinates (1985), frequently allow identification of clusters. Our own program CLUSPLOT, displaying the size, shape, and relative positions of the clusters, is briefly discussed in Section 4 of the Appendix.

EXERCISES AND PROBLEMS

1. Edwards and Cavalli-Sforza (1965) considered the following dissimilarity matrix between six families of bacteria:

$$\begin{array}{c|cccccc} & A & B & C & D & E & F \\ \hline A & 0 & & & & & \\ B & 5 & 0 & & & & \\ C & 11 & 10 & 0 & & & \\ D & 11 & 6 & 6 & 0 & & \\ E & 14 & 13 & 17 & 13 & 0 & \\ F & 14 & 15 & 21 & 15 & 6 & 0 \end{array}.$$

Partition these data into two clusters with PAM and check that both are L^*-clusters. What is the "best" number of clusters according to the silhouettes? (Try all values of k.)

2. Show that for one-dimensional data the choice of a distance function (Euclidean or Manhattan) and of the standardization option has no effect on the clustering found by PAM.

3. Show that if a k-medoid clustering contains a doubleton (a cluster consisting of two objects), either object can be chosen as the medoid of that cluster.

4. The k-medoid method allocates objects to the nearest medoid. Give a geometrical interpretation of this fact when the data are two-dimensional and $k = 2$, by drawing the line which forms the boundary between the regions determined by both medoids. Do the same for two-dimensional data when $k = 3$.

5. Consider the countries data of Table 5. Multiply all dissimilarities by a constant factor (e.g., 5). Then use the program PAM to partition the countries into three clusters and compare the results with those described in Section 3. Repeat the exercise after adding a constant (e.g., 10) to each of the original dissimilarities.

6. Consider the following data set consisting of 28 two-dimensional points:

(0,1)	(0,2)	(0,3)	(1,0)	(1,1)	(1,2)	(1,3)
(1,4)	(1,8)	(1,9)	(2,1)	(2,2)	(2,4)	(2,7)
(2,9)	(3,3)	(3,5)	(3,7)	(3,8)	(4,4)	(4,6)
(4,7)	(5,4)	(5,5)	(5,6)	(6,3)	(6,4)	(6,5)

(a) Make a plot of these points.

(b) Cluster this data set into two clusters by means of PAM, with the options "no standardization" and "Euclidean distance."

(c) Repeat the clustering using only the first variable and then using only the second variable. Show that both variables are necessary to retrieve the cluster structure.

7. Cluster the squares of the integers between 0 and 50 (i.e., $0, 1, 4, 9, 16, 25, \ldots, 2500$) into 2 to 10 clusters by means of PAM.

8. Consider the dissimilarity matrix (4) in Section 2.

(a) Cluster these data by means of PAM for $k = 2$.

(b) If you have access to a program for multidimensional scaling (MDS), construct a two-dimensional representation of the five objects characterized by (4). Indicate the clustering obtained by PAM on this plot and compare with the information obtained from the silhouettes.

9. Consider the artificial data set of Figure 1 of Chapter 4 (the actual data are listed in Section 2.1 of that chapter). Cluster these objects with PAM for $k = 3$, using the options "no standardization" and "Euclidean distance." What happens with the intermediate objects 6 and 13?

10. Atypical objects usually become singleton clusters. If too few clusters are requested, these objects are (wrongly) grouped with some other objects. A particular type of atypical object are the so-called bridges. These are objects located in between two or more clusters. Suppose too few clusters have been requested and a bridging object i has been assigned to one of the clusters. Can this be recognized from the values of $a(i)$ and $b(i)$ used to construct the silhouette for this object?

11. Show that for a one-dimensional data set the k-medoid clustering consists of contiguous clusters.

12. Show that when two clusters are required, the algorithm used in the program PAM always yields the exact minimum of the objective function.

13. (Research.) Construct a k-medoid algorithm for the special case of one-dimensional data, making use of the fact that all clusters must be contiguous. It is advisable to begin by sorting the observations in increasing order, because this allows one to exclude many possible clusters. Another simplification in one dimension is that there is no longer a need to store all distances between objects in central memory because the distances can easily be computed along the way. The program will therefore need little storage space.