Multivariate Data Analysis with Fortran, C and Java Code

Fionn Murtagh

Queen's University Belfast, and
Astronomical Observatory Strasbourg

and contributions from

Gutti Jogesh Babu, Jonathan G. Campbell, André Heck, Eric Feigelson,
Chris Fraley, Soma Mukherjee, and Adrian Raftery

Foreword

Chapters 1 to 5 started life as lecture notes of a course given by F. Murtagh in the Joint Research Centre, Ispra, Italy, in 1984. The layout of the chapters, as “Mathematical Description”, “Examples and Bibliography”, and “Software and Sample Implementation”, is something we owe to Dr Alessandro Colombo who proposed a format along these lines. The course notes, in book format, were published as a JRC Ispra internal report.

There followed in 1987 the book *Multivariate Data Analysis*, written by F. Murtagh and A. Heck, and published by Kluwer Academic Publishers, Dordrecht, in hardback and softback. This book covered chapters 1 to 6 of the present text.

In the current version, Chapter 4 has been extended considerably. and the figures have been redone throughout. In Chapters 2 and 3, new software implementations are included as Java applications. Chapter 7 and the Appendix are new. Chapter 8 has seen various additions.

It is our objective to include Java code for the other chapters, and to remove the Fortran programs (which will remain available on the web). The current Java application codes will be enhanced. An index will be added. Some harmonization of the notation used in Appendix A is called for.

Case study 2 is based on the work described in S. Mukherjee, E.D. Feigelson, G.J. Babu, F. Murtagh, C. Fraley and A. Raftery, “Three types of gamma ray bursts”, The Astrophysical Journal, 508, 314–327, 1998.

We foresee also the possibility to add more material relating to input data coding, especially in the context of correspondence analysis. This will lead naturally to fuzzy data analysis and possibility theory, which are especially valuable for the handling of imprecise data. We will exemplify this with examples from articles on such decision support methods in the context of observatory support operations.

Among other topics which could be considered are the clustering of massive data sets (see F. Murtagh, “Clustering in massive data sets”, in J. Abello, P.M. Pardalos and M.G.C. Reisende, Eds., *Handbook of Massive Data Sets*, Kluwer, 2000, forthcoming), to include “on-line” algorithms used in the Sloan Digital Sky Survey, for instance; and noise filtering methods, in particular those based on multiresolution methods. See J.L. Starck, F. Murtagh and A. Bijaoui, *Image and Data Analysis: The Multiscale Approach*, Cambridge University Press, 1998. See also J.L. Starck, F. Murtagh, P. Querre and F. Bonnarel, “Entropy and astronomical data analysis”, recently (late 2000) submitted to Astronomy and Astrophysics.

Contents

List of Figures

List of Tables

Chapter 1

Data Coding and Initial Treatment

1.1 The Problem

Data Analysis has different connotations for different researchers. It is however definitely part of the chain of reduction and exploitation of data, located somewhere between, let us say, the taking of observations and the eventual, published article.

The versatility and range of possible applications of multivariate data analysis make it important to be understood by astronomical researchers. Its applicability–boundaries — or equally, the requirements for input data for such algorithms — should also be appreciated.

In the following chapters, the techniques to be studied are

Principal Components Analysis: which focusses on inter–object correlations, reduces their (parameter–space) dimensionality, and allows planar graphic representations of the data;

Cluster Analysis: which is the application of automatic grouping procedures;

Discriminant Analysis: which classifies items into pre–defined groups.

A number of other, related methods will also be studied.

The data array to be analysed crosses objects (e.g. ultraviolet spectra, spiral galaxies or stellar chemical abundances) with variables or parameters. The former are usually taken as the rows of the array, and the latter as the columns. The choice of parameters to use for the study of a particular set of objects is of vital importance, but concrete guidelines cannot unfortunately be given in advance. The results of the analysis will depend in large measure on the choice of parameters (and may indeed be used to judge the relevance of the set of parameters chosen).

The following remarks should be borne in mind, however, in choosing parameters to suitably characterise the objects under investigation.

- As far as possible the parameters should be *homogeneous*: in multivariate data analysis, we do not wish to find differences in the objects being explained purely by inhomogeneities in the parameters used.

- The parameters should allow for a *comprehensive* description of the objects; i.e. express as many aspects of the objects as possible.

- They, and the objects, should be chosen so that there is no *bias* in what is being studied (Pfleiderer, 1983). There should also be sufficient objects to constitute a reasonable sample (as a very general rule, a few dozen).

- As far as possible, *missing data* should be avoided.

Later in this chapter, we will describe the selection of parameters in two short case–studies.

Unlike classical, inferential statistics, there is rarely need for distributional assumptions in the methods to be studied. Multivariate methods have *description* of the data as their aim, — and hence the drawing of conclusions. Less precision in the possible conclusions of multivariate methods is balanced by the greater range of situations where they can be applied. Nonetheless, some knowledge of classical statistics is a *sine qua non* for successful analysis of data. Wall (1979) or Wetherill (1972), among many other texts, provide introductory material in this area.

The object–parameter dependencies can take many forms. Most suitable from the point of view of many of the algorithms to be discussed in subsequent chapters is *quantitative* data, i.e. real valued data, positive or negative, defined relative to some zero point. Another form of data is *qualitative* or *categorical*, i.e. the object–parameter relation falls into one of a number of categories; in its most simple form, this is a yes/no dependency indicating the presence or absence of the parameter. The coding of a categorical variable or parameter may take the general form of values "a", "b", etc. for the different categories, or "1", "2", etc. (where in the latter case, the values have "qualitative" significance only). A final form of data to be mentioned here is *ordinal* data where a rank characterises the object on the parameter.

1.2 Mathematical Description

1.2.1 Introduction

In the great majority of multivariate data analysis methods, the notion of distance (or similarity) is central. In clustering, objects are clustered on the basis of their mutual similarity, for instance, and in Principal Components Analysis (PCA) the points are considered as vectors in a metric space (i.e. a space with a distance defined).

A very large number of coefficients for measuring similarity or distance have at one time or another been proposed. We will not attempt an inventory in the following sections but will instead deal with commonly used coefficients.

If the data to be analysed is of conveniently small size, then a visual scan of pairwise distances can reveal interesting features. It could be asserted that descriptive procedures such as PCA and clustering provide means of exploring such distances when it becomes impractical to do it "manually".

Some of the problems which arise in deciding on a suitable distance are as follows. If the data to be analysed is all of one type, a suitable distance can be chosen without undue difficulty. If the values on different coordinates are quite different — as is more often the case — some *scaling* of the data will be required before using a distance. Equally, we may view the use of a distance on scaled data as the definition of another, new distance. More problemsome is the case of data which is of mixed type (e.g. quantitative, categorical, ordinal values). Here, it may be possible to define a distance which will allow all coordinates to be simultaneously considered. It may be recommendable, though, to redefine certain coordinate variables (e.g. to consider ordinal values as quantitative or real variables). As a general rule, it is usually best to attempt to keep "like with like". A final problem area relates to missing coordinate values: as far as possible care should be taken in the initial data collection phase to ensure that all values are present.

1.2.2 Distances

Proximity between any pair of items will be defined by *distance*, *dissimilarity* or *similarity*. *Distance* is simply a more restrictive *dissimilarity*, — it satisfies certain axioms listed below. Both *distances* and *dissimilarities* measure identical items by a zero value, and by increasingly large (positive) values as the proximity of the items decreases. *Similarities* are mathematical functions which treat pairs of items from the other perspective: large values indicate large proximity, while small (positive) or zero values indicate little or no proximity. The mathematician is happiest when dealing with *distances*: an established body of theory is immediately available, and many of the methods to be studied in subsequent chapters work on distances.

The most commonly used distance for quantitative (or continuous) data is the *Euclidean* distance. If

$$\mathbf{a} = \{a_j : j = 1, 2, ..., m\}$$

and

$$\mathbf{b} = \{b_j : j = 1, 2, ..., m\}$$

are two real-valued vectors then the unweighted squared Euclidean distance is given by

$$d^2(a, b) = \sum_j (a_j - b_j)^2 = (\mathbf{a} - \mathbf{b})'(\mathbf{a} - \mathbf{b})$$

where **a** and **b** are taken as column vectors, and $'$ denotes transpose,

$$= ||\mathbf{a}||^2 + ||\mathbf{b}||^2 - 2\mathbf{a}'\mathbf{b}$$

where $||.||$ is the norm, or distance from the origin.

The Euclidean, along with all distances, satisfies the following properties:

Symmetry: $d(a, b) = d(b, a)$.

Positive semi-definiteness: $d(a, b) > 0, \text{ if } a \neq b; d(a, b) = 0, \text{ if } a = b$.

Triangular inequality: $d(a, b) \leq d(a, c) + d(c, b)$.

The triangular inequality in the Euclidean plane is simply the assertion that going from a to b cannot be shortened by going first to another point c; equality corresponds to point c being located on the line connecting a and b. Some further aspects of the Euclidean distance are discussed in Chapter 2.

The *Hamming* distance is an alternative to the Euclidean distance:

$$d(a,b) = \sum_j \mid a_j - b_j \mid$$

where $\mid . \mid$ is absolute value.

When *binary* data is being studied (i.e. categorical data with presence/absence values for each object, on each parameter) mutual possession of a property contributes 0 to this distance, mutual non-possession also contributes 0, and presence/absence contributes 1.

The Hamming and Euclidean distances are both members of a general class of distances termed the *Minkowski* metrics:

$$d_p(a,b) = \sqrt[p]{\sum_j \mid a_j - b_j \mid^p} \qquad p \geq 1.$$

When $p = 2$ we have the Euclidean distance; $p = 1$ gives the Hamming distance; and $p = \infty$ reduces to

$$d_\infty(a,b) = max_j \mid a_j - b_j \mid$$

which is the "maximum coordinate" or *Chebyshev* distance. These three are the most commonly used Minkowski distances. The corresponding norms, i.e.

$$||\mathbf{a}||_p = d_p(a,0)$$

where 0 is the origin, are referred to as the L_1 , L_2 , and L_∞ norms.

Since these distances are symmetric, it is not necessary to store all n interpair distances for n objects: $n(n-1)/2$ suffice.

With a suitable coding, the Euclidean, Hamming or other distances may be used in a wide variety of circumstances. Consider in Figure 1.1 part of a set of records from a Quasar catalogue. Part of two records, x and y, are shown. In order to carry out a comparison of such records, we can as a preliminary recode each variable, as shown. The Hamming and Euclidean distances are both then

$$d(x,y) = 0+0+0+0+1+1+1 = 3$$

in this case. The breakdown of Seyfert spectrum type in this example is easier to accomplish than the breakdown of positional coordinates (it would be necessary to astutely break up the angle values into useful, but ad-hoc, categories). User choice is required in defining such a disjunctive form of coding. In the case of quantitative variables this coding may be especially useful when one is in the presence of widely varying values: specifying a set of categories may make the distances less vulnerable to undesirable fluctuations.

The possible preponderance of certain variables in, for example, a Euclidean distance leads to the need for a *scaling* of the variables, i.e. for their *centring* (zero mean), *normalization* (unit variance), or *standardization* (zero mean and unit standard deviation). If a_{ij} is the j^{th} coordinate of vector $\mathbf{a}_i$ (i.e. the ij^{th}

Record x:	S1, 18.2, X
Record y:	S1, 6.7, —

	Seyfert type spectrum				Integrated magnitude		X–ray data?
	S1	S2	S3	—	≤ 10	> 10	Yes
x	1	0	0	0	0	1	1
y	1	0	0	0	1	0	0

Figure 1.1: Two records (x and y) with three variables (Seyfert type, magnitude, X-ray emission) showing disjunctive coding.

table entry), $\bar{a}_j$ is the mean value on coordinate (variable) j, and σ_j is the standard deviation of variable j, then we standardize a_{ij} by transforming it to

$$(a_{ij} - \bar{a}_j)/\sigma_j$$

where

$$\bar{a}_j = \sum_{i=1}^{n} a_{ij}/n$$

$$\sigma_j^2 = \sum_{i=1}^{n} (a_{ij} - \bar{a}_j)^2/n$$

and n is the number of rows in the given table.

Standardization, defined in this way, is widely used, but nothing prevents some alternative scaling being decided on: for instance we could divide each value in the data matrix by the row mean, which has the subsequent effect of giving a zero distance to row vectors each of whose elements are a constant times the other. We may regard the resultant distance as a weighted Euclidean distance of the following form:

$$d^2(a,b) = \sum_j (w_1 a_j - w_2 b_j)^2.$$

Missing values constitute a current research problem: the simplest option is to delete rows or columns of the input data table in order to avoid these. There is no clear answer as to how to estimate missing values. However a dissimilarity, similar in form to a weighted Euclidean distance, may always be used for quantitative data:

$$d^2(a,b) = \sum_{j=1}^{m'} \frac{m'}{m}(a_j - b_j)^2$$

where $m' < m$ pairs of coordinates are simultaneously present for both **a** and **b**.

1.2.3 Similarities and Dissimilarities

A *dissimilarity* may satisfy some of the properties of a distance but often the triangular inequality is found to be the most difficult to satisfy.

Similarities, just as distances or dissimilarities, are functions mapping pairs of items (most often, vectors in a multidimensional space) into positive, real values.

In the case of binary data (i.e. data representing by 0 and 1 whether or not an object possesses a property), many similarity coefficients have been proposed. A similarity, s_{ij}, may be converted into a dissimilarity, δ_{ij}, when s_{max} is the maximum similarity present, in the following way:

$$\delta_{ij} = s_{max} - s_{ij}.$$

Similarity coefficients are often defined so that their resultant value will be between 0 and 1. For example, the Jaccard coefficient is defined for binary vectors **a** and **b**, and where N represents the counting operator, as

$$s(a,b) = \frac{N_j(a_j = b_j = 1)}{N_j(a_j = 1) + N_j(b_j = 1) - N_j(a_j = b_j = 1)}.$$

The numerator here reads: the number of times property j is simultaneously present for objects a and b. This can be written in vector notation as

$$s(a,b) = \frac{\mathbf{a'b}}{\mathbf{a'a} + \mathbf{b'b} - \mathbf{a'b}}.$$

As an example, the Jaccard similarity coefficient of vectors (10001001111) and (10101010111) is $5/(6+7-5) = 5/8$.

In the case of mixed quantitative–qualitative data, a suitable similarity coefficient may be difficult to define. The Gower coefficient considers all binary, other qualitative, and quantitative/ordinal coordinates in turn: for the binary, the Jaccard coefficient is determined; for other qualitative, a 1 indicates the same coordinate value, and a 0 indicates different coordinate values; and for the j^{th} quantitative or ordinal coordinate of objects **a** and **b**, we determine

$$1 - \frac{\mid a_j - b_j \mid}{max_j \mid a_j - b_j \mid}.$$

The denominator is the maximum spread. It is seen that all contributions to the Gower similarity coefficient are between 0 and 1. A similarity coefficient with values between 0 and 1 can be obtained by

$$s = \frac{n_1 s_1 + n_2 s_2 + n_3 s_3}{n_1 + n_2 + n_3}$$

where n_1 coordinates are binary and s_1 is the Jaccard similarity obtained; n_2 coordinates are otherwise qualitative and s_2 is their contribution to the similarity discussed above; and finally n_3 coordinates are quantitative or ordinal.

For further discussion of similarities and distances, Anderberg (1973) or Everitt (1980) may be consulted. For their relation to metric properties, Gower and Legendre (1986) can be recommended.

We will conclude this section by pointing to a very different approach to defining dissimilarity, which constitutes an interesting research direction. In the case of spectral matching, without knowledge of calibration, a best fit between any pair of spectra must first be obtained and on the basis of this a dissimilarity defined. A technique from the Operations Research field, known as *dynamic programming*, may be used for such optimal matching. Dynamic programming offers a flexible approach for optimally matching two ordered sets of values. Further reading is to be found in Kruskal (1983), Sankoff and Kruskal (1983) and Hall and Dowling (1980).

1.3 Examples and Bibliography

1.3.1 Selection of Parameters

The annotated bibliographies of subsequent chapters may be referred to for studies in a wide range of areas, and where details of data coding may be found. Here, we will restrict ourselves to a few examples of the way in which parameter selection is carried out.

1.3.2 Example 1: Star–Galaxy Separation

In the case of star–galaxy classification, following the scanning of digitised images, Kurtz (1983) lists the following parameters which have been used:

1. mean surface brightness;
2. maximum intensity, area;
3. maximum intensity, intensity gradient;
4. normalized density gradient;
5. areal profile;
6. radial profile;
7. maximum intensity, 2^{nd} and 4^{th} order moments, ellipticity;
8. the fit of galaxy and star models;
9. contrast versus smoothness ratio;
10. the fit of a Gaussian model;
11. moment invariants;
12. standard deviation of brightness;
13. 2^{nd} order moment;
14. inverse effective squared radius;
15. maximum intensity, intensity weighted radius;
16. 2^{nd} and 3^{rd} order moments, number of local maxima, maximum intensity.

References for all of these may be found in the cited work. Clearly there is room for differing views on parameters to be chosen for what is essentially the same problem! It is of course the case also that aspects such as the following will help to orientate us towards a particular set of parameters in a particular case: the quality of the data; the computational ease of measuring certain parameters;

the relevance and importance of the parameters measured relative to the data analysis output (e.g. the classification, or the planar graphics); similarly, the importance of the parameters relative to theoretical models under investigation; and in the case of very large sets of data, storage considerations may preclude extravagance in the number of parameters used.

1.3.3 Example 2: Galaxy Morphology Classification

The generating of a galaxy equivalent of the Hertzsprung–Russell diagram for stars, or the determining of a range of Hubble–type classes, using quantitative data on the galaxies, is a slowly burgeoning research topic. The inherent difficulty of characterising spirals (especially when not face–on) has meant that most work focusses on ellipticity in the galaxies under study. This points to an inherent bias in the potential multivariate statistical procedures. The inherent noisiness of the images (especially for faint objects) has additionally meant that the parameters measured ought to be made as robust as is computationally feasible. In the following, it will not be attempted to address problems of galaxy photometry per se (see Davoust and Pence, 1982; Pence and Davoust, 1985), but rather to draw some conclusions from recent (and ongoing) work which has the above–mentioned objectives in view.

From the point of view of multivariate statistical algorithms, a reasonably homogeneous set of parameters is required. Given this fact, and the available literature on quantitative galaxy morphological classification, two approaches to parameter selection appear to be strongly represented, albeit intermixed:

1. The luminosity profile along the major axis of the object is determined at discrete intervals. This may be done by the fitting of elliptical contours, followed by the integrating of light in elliptical annuli (Lefèvre *et al.*, 1986). A similar approach is used for the comprehensive database, currently being built up for the European Southern Observatory galaxy survey (by A. Lauberts). Noisiness and faintness require attention to robustness in measurement: the radial profile may be determined taking into account the assumption of a face–on optically–thin axisymmetric galaxy, and may be further adjusted to yield values for circles of given radius (Watanabe *et al.*, 1982). Alternatively, isophotal contours may determine the discrete radial values for which the profile is determined (Thonnat, 1985).

2. Specific morphology–related parameters may be derived instead of the profile. The integrated magnitude within the limiting surface brightness of 25 or 26 mag. arcsec^{-2} in the visual is popular (Takase *et al.*, 1984; Lefèvre *et al.*, 1986). The logarithmic diameter (D_{26}) is also supported by Okamura (1985). It may be interesting to fit to galaxies under consideration model bulges and disks using, respectively, $r^{\frac{1}{4}}$ or exponential laws (Thonnat, 1985), in order to define further parameters. Some catering for the asymmetry of spirals may be carried out by decomposing the object into octants; furthermore the taking of a Fourier transform of the intensity may indicate aspects of the spiral structure (Takase *et al.*, 1984).

In the absence of specific requirements for the multivariate analysis, the following remarks can be made.

- The range of parameters to be used should be linked, if feasible, to the further use to which they might be put (i.e. to underlying theoretical aspects of interest).
- It appears that many parameters can be derived from a carefully–constructed luminosity profile, rather than it being possible to derive a profile from any given set of parameters. The utility of profiles would appear to be thereby favoured in the long term in this field.
- The presence of both types of data in the database is of course not a hindrance to analysis: however it is more useful if the analysis is carried out on both types of data separately.

Parameter data may be analysed by clustering algorithms, by Principal Components Analysis or by methods for Discriminant Analysis (all of which will be studied in the following chapters). Profile data may be sampled at suitable intervals and thus analysed also by the foregoing procedures. It may be more convenient in practice to create dissimilarities between profiles, and analyse these dissimilarities: this may be done using clustering algorithms with dissimilarity input.

1.3.4 Example 3: Interactive Visualization

Frequently the analyst must interact with the data. This means that one type of display is made, followed by a different visualization of some subset of the data. The term "exploratory data analysis" is most closely associated with the name of Tukey (Princeton). Interactive statistics is another term used, and this activity may be supported by computer software. A prime example is the S language (or software environment) originating in ATT Bell Labs, and enhanced as the S-Plus package by MathSoft Inc. (formerly StatSci Inc.). Figures 1.2, 1.3 and 1.4 were produced using S-Plus. Statistical graphics is used to mean much the same thing.

To illustrate the types of visualizations which will be studied in greater depth elsewhere in this book, we will take a small data set used in Capaciolli et al. (1991). The authors collected data on 14 different galactic globular clusters, all of which had been collected in earlier CCD (digital detector) photometry studies. They studied a number of salient associations between variables. We will do this based on global simultaneous views of the data, and briefly note some aspects of these analyses.

A set of all pairwise plots is shown in Figure 1.2. We could label the points. Not having done so means that correlations and clusters are essentially what we will instead look for. Consider the plots of variables `x` and `x_0`: they are very highly correlated, such that one or other of these variables is in fact redundant. Consider `x` and `Z_g`: the relationship is less spectacular, but nonetheless very correlated. This fact is discussed in Capaccioli et al. (1991). Outliers, to be interpreted as anomalous observations or perhaps as separate classes, are to be seen in quite a few of the plot panels.

Figure 1.3 shows a principal components analysis of the variables. This is an optimal planar representation of the data (subject, of course, to what we mean by optimality: in Chapter 2 we will define this). The variable `[Fe/H]` is quite different from the others. This variable, metallicity, was discussed in

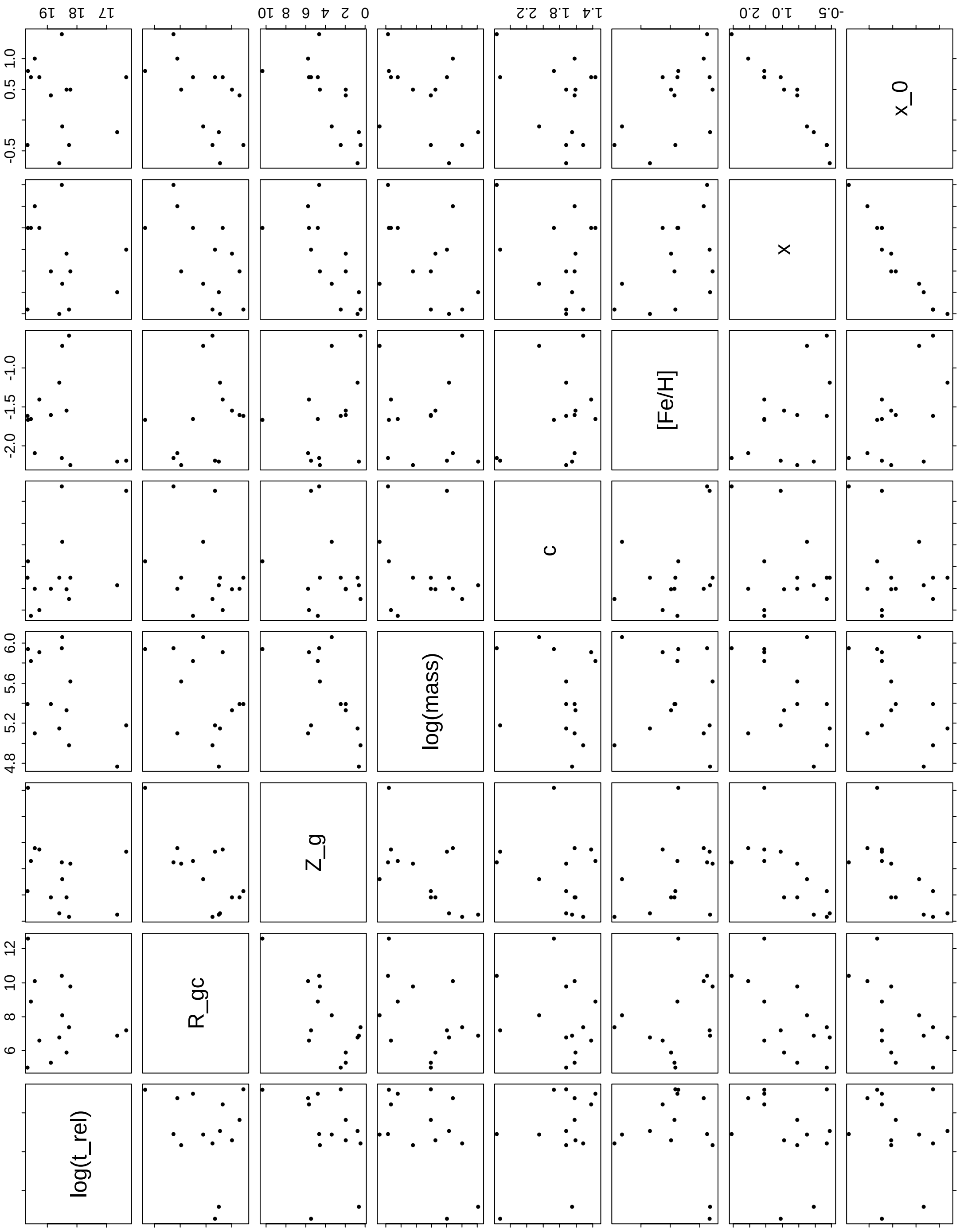


Figure 1.2: All pairwise plots between 8 variables.

Object	t_rlx	Rgc	Zg	log(M/	c	[Fe/H]	x	x0
	years	Kpc	Kpc	M.)				
M15	1.03e+8	10.4	4.5	5.95	2.54	-2.15	2.5	1.4
M68	2.59e+8	10.1	5.6	5.1	1.6	-2.09	2.0	1.0
M13	2.91e+8	8.9	4.6	5.82	1.35	-1.65	1.5	0.7
M3	3.22e+8	12.6	10.2	5.94	1.85	-1.66	1.5	0.8
M5	2.21e+8	6.6	5.5	5.91	1.4	-1.4	1.5	0.7
M4	1.12e+8	6.8	0.6	5.15	1.7	-1.28	-0.5	-0.7
47 Tuc	1.02e+8	8.1	3.2	6.06	2.03	-0.71	0.2	-0.1
M30	1.18e+7	7.2	5.3	5.18	2.5	-2.19	1.0	0.7
NGC 6397	1.59e+7	6.9	0.5	4.77	1.63	-2.2	0.0	-0.2
M92	7.79e+7	9.8	4.4	5.62	1.7	-2.24	0.5	0.5
M12	3.26e+8	5.0	2.3	5.39	1.7	-1.61	-0.4	-0.4
NGC 6752	8.86e+7	5.9	1.8	5.33	1.59	-1.54	0.9	0.5
M10	1.50e+8	5.3	1.8	5.39	1.6	-1.6	0.5	0.4
M71	8.14e+7	7.4	0.3	4.98	1.5	-0.58	-0.4	-0.4

Table 1.1: Data: 14 globular clusters, 8 variables.

Capaccioli et al. (1991) and used as a basis for subdividing the globular clusters. We could also view the latter in the principal plane and find, in that way, what observations are most positively associated with the variable `[Fe/H]`.

Another way to visualize the observations, if clustering is really what we are interested in, is to directly cluster them. A hierarchical clustering provides lots of classification-related information. Figure 1.4 shows such a classification tree, or dendrogram. Two large clusters are evident, comprising the 6 globular clusters to the left, and the 8 globular clusters to the right. Note how the branches could be reversed. However what belongs in any given branch will not change, subject to the particular clustering criterion being used. In Figure 1.4, a criterion seeking maximal cluster compactness (defined by within cluster variances) is employed.

These methods are relatively powerful. They allow us to answer questions related to internal associations and correlations in our data. They provide answers to degrees of redundancy and of "anomalicity". They provide visualizations to help us with communication of our conclusions to our clients or colleagues. They are tools (algorithmic, software) which are easy to use, and which let the data speak for themselves.

1.3.5 General References

1. M.R. Anderberg, *Cluster Analysis for Applications*, Academic Press, New York, 1973.

2. R.A. Becker, J.M. Chambers and A.R. Wilks, *The New S Language*, Chapman and Hall, 1988.

3. M. Capaccioli, S. Ortolani and G. Piotto, "Empirical correlations between globular cluster parameters and mass function morphology", *Astronomy and Astrophysics*, 244, 298–302, 1991.

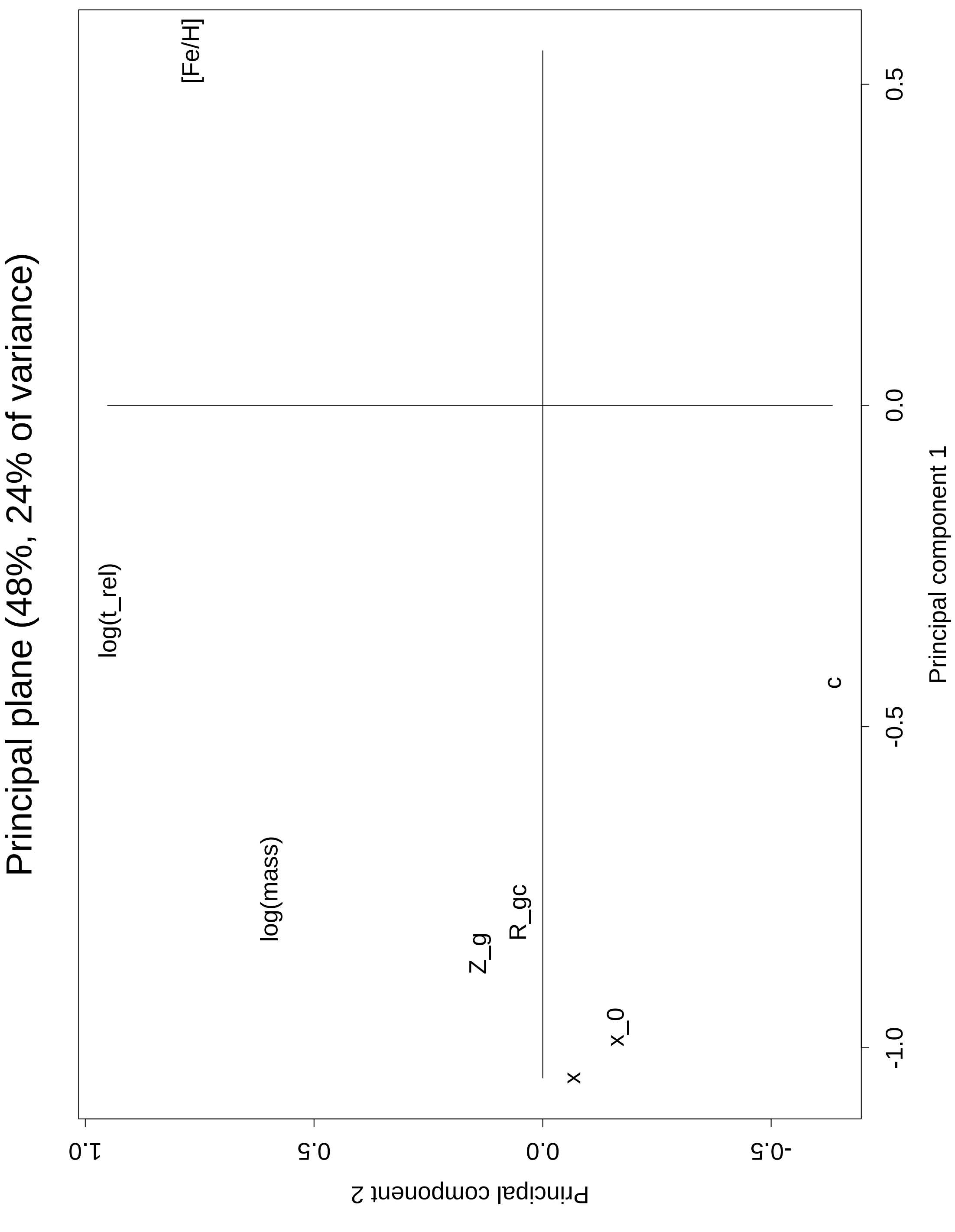


Figure 1.3: Principal component analysis (principal plane) of the 8 variables.

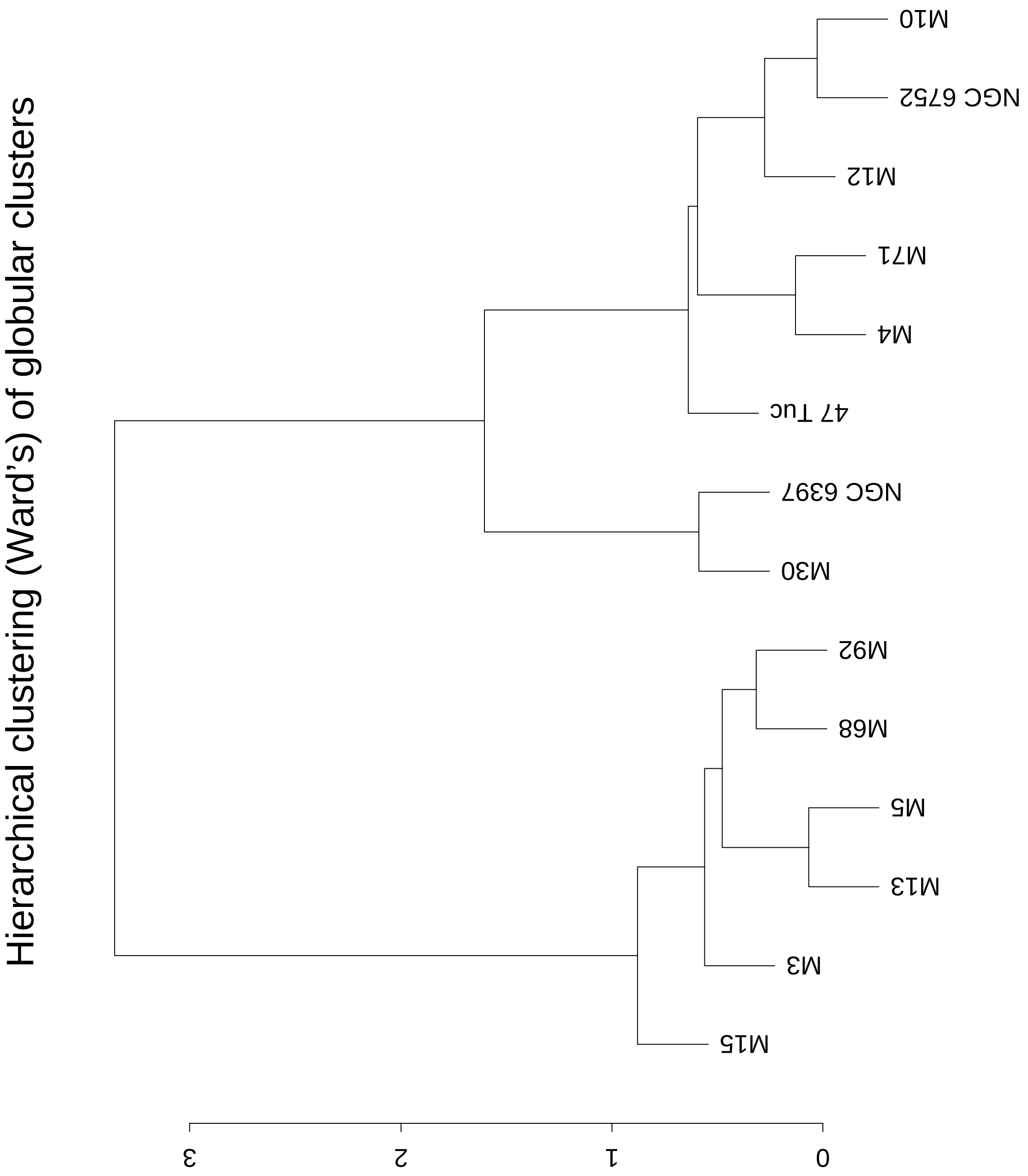


Figure 1.4: Hierarchical clustering of the 14 globular clusters.

4. J.M. Chambers and T. Hastie, Eds., *Statistical Models in S*, Chapman and Hall, 1993.

5. J.M. Chambers, *Programming with Data: A Guide to the S Language*, MathSoft, Seattle, 1998.

6. E. Davoust and W.D. Pence, "Detailed bibliography on the surface photometry of galaxies", *Astronomy and Astrophysics Supplement Series*, **49**, 631–661, 1982.

7. B. Everitt, *Cluster Analysis*, Heinemann Educational Books, London, 1980 (2nd ed.).

8. J.C. Gower and P. Legendre, "Metric and Euclidean properties of dissimilarity coefficients" *Journal of Classification* **3**, 5–48, 1986.

9. P.A.V. Hall and G.R. Dowling, "Approximate string matching", *Computing Surveys*, **12**, 381–402, 1980.

10. C. Jaschek, "Data growth in astronomy", *Quarterly Journal of the Royal Astronomical Society*, **19**, 269–276, 1978.

11. J.B. Kruskal, "An overview of sequence comparison: time warps, string edits, and macromolecules", *SIAM Review*, **25**, 201–237, 1983.

12. M.J. Kurtz, "Classification methods: an introductory survey", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 47–58, 1983.

13. O. Lefèvre, A. Bijaoui, G. Mathez, J.P. Picat and G. Lelièvre, "Electronographic BV photometry of three distant clusters of galaxies", *Astronomy and Astrophysics*, **154**, 92–99, 1986.

14. J.V. Narlikar, "Statistical techniques in astronomy", *Sankhya: The Indian Journal of Statistics, Series B, Part 2*, **44**, 125–134, 1982.

15. S. Okamura, "Global structure of Virgo cluster galaxies", in eds. O.G. Richter and B. Binggeli, *ESO Workshop on The Virgo Cluster of Galaxies*, ESO Conference and Workshop Proceedings No. 20, 201–215, 1985.

16. W.D. Pence and E. Davoust, "Supplement to the detailed bibliography on the surface photometry of galaxies", *Astronomy and Astrophysics Supplement Series*, **60**, 517–526, 1985.

17. J. Pfleiderer, "Data set description and bias", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 3–11, 1983.

18. D. Sankoff and J.B. Kruskal (Eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, Addison–Wesley, New York, 1983.

19. B. Takase, K. Kodaira and S. Okamura, *An Atlas of Selected Galaxies*, University of Tokyo Press, Tokyo, 1984.

20. M. Thonnat, "Automatic morphological description of galaxies and classification by an expert system", INRIA Rapport de Recherche, Centre Sophia Antipolis, No. 387, 1985.

21. J.V. Wall, "Practical statistics for astronomers. I. Definitions, the normal distribution, detection of signal", *Quarterly Journal of the Royal Astronomical Society*, **20**, 130–152, 1979.

22. M. Watanabe, K. Kodaira and S. Okamura, "Digital surface photometry of galaxies toward a quantitative classification. I. 20 galaxies in the Virgo cluster", *Astrophysics Journal Supplement Series*, **50**, 1–22, 1982.

23. G.B. Wetherill, *Elementary Statistical Methods*, Chapman and Hall, London, 1972.

Chapter 2

Principal Components Analysis

2.1 The Problem

We have seen in Chapter 1 how the $n \times m$ data array which is to be analysed may be viewed immediately as a set of n row–vectors, or alternatively as a set of m column–vectors. PCA seeks the best, followed by successively less good, summarizations of this data. Cluster Analysis, as will be seen in Chapter 3, seeks groupings of the objects or attributes. By focussing attention on particular groupings, Cluster Analysis can furnish a more economic presentation of the data. PCA (and other techniques, as will be seen in a later chapter) has this same objective but a very different summarization of the data is aimed at.

In Figure 2.1a, three points are located in $\mathbb{R}^2$. We can investigate this data by means of the coordinates on the axes, taken separately. We might note, for instance, that on axis 1 the points are fairly regularly laid out (with coordinates 1, 2 and 3), whereas on axis 2 it appears that the points with projections 4 and 5 are somewhat separated from the point with projection 2. In higher–dimensional spaces we are limited to being easily able to visualize one–dimensional and two–dimensional representations (axes and planes), although at the limit we can construct a three–dimensional representation.

Given, for example, the array of 4 objects by 5 attributes,

$$\begin{pmatrix} 7 & 3 & 4 & 1 & 6 \\ 3 & 4 & 7 & 2 & 0 \\ 1 & 7 & 3 & -1 & 4 \\ 2 & 0 & -6 & 4 & 1 \end{pmatrix}$$

the projections of the 4 objects onto the plane constituted by axes 1 and 3 is simply

$$\begin{pmatrix} 7 & 4 \\ 3 & 7 \\ 1 & 3 \\ 2 & -6 \end{pmatrix}.$$

Thus far, the projection of points onto axes or planes is a trivial operation.

$$\text{Array for Fig. 2.1} = \begin{pmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 5 \end{pmatrix}$$

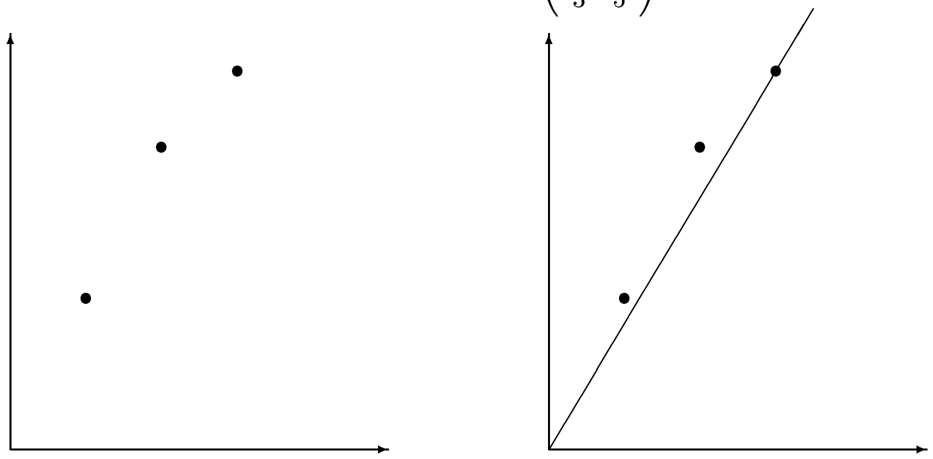


Figure 2.1: Points and their projections onto axes.

PCA, however, first obtains *better* axes. Consider Figure 2.1b where a new axis has been drawn as nearly as possible through all points. It is clear that if this axis went precisely through all points, then a second axis would be redundant in defining the locations of the points; i.e. the cloud of three points would be seen to be one–dimensional.

PCA seeks the axis which the cloud of points are closest to (usually the Euclidean distance defines *closeness*). This criterion will be seen below to be identical to another criterion: that the projections of points on the axis sought for be as elongated as possible. This second criterion is that the *variance* of the projections be as great as possible.

If, in general, the points under examination are m–dimensional, it will be very rare in practice to find that they approximately lie on a one–dimensional surface (i.e. a line). A second best–fitting axis, orthogonal to the first already found, will together constitute a best–fitting plane. Then a third best–fitting axis, orthogonal to the two already obtained, will together constitute a best–fitting three–dimensional subspace.

Let us take a few simple examples in two–dimensional space (Figure 2.2). Consider the case where the points are *centred* (i.e. the origin is located at the centre of gravity): this will usually be the case if the data are initially transformed to bring it about (see the previous Chapter and Section 2.2.5 below). We will seek the best–fitting axis, and then the next best–fitting axis. Figure 2.2a consists of just two points, which if centred must lie on a one–dimensional axis. In Figure 2.2b, the points are arranged at the vertices of a triangle. The vertical axis, here, accounts for the greatest variance, and the symmetry of the problem necessitates the positioning of this axis as shown. In the examples of Figure 2.2, the positive and negative orientations of the axes are arbitrary since they are not integral to our objective in PCA.

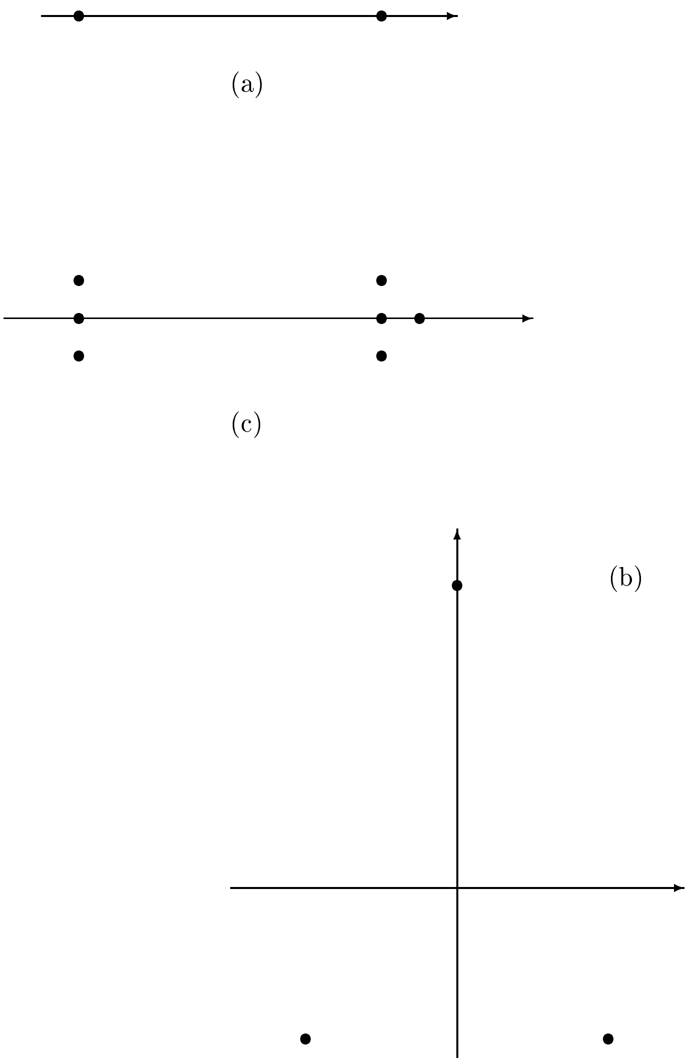


Figure 2.2: Some examples of PCA of centred clouds of points.

Central to the results of a PCA are the coordinates of the points (i.e. their projections) on the derived axes. These axes are listed in decreasing order of importance, or best–fit. Planar representations can also be output: the projections of points on the plane formed by the first and second new axes; then the plane formed by the first and third new axes; and so on, in accordance with user–request.

It is not always easy to remedy the difficulty of being unable to visualize high–dimensional spaces. Care must be taken when examining projections, since these may give a misleading view of interrelations among the points.

2.2 Mathematical Description

2.2.1 Introduction

The mathematical description of PCA which follows is important because other techniques (e.g. Discriminant Analysis) may be viewed as variants on it. It is one of the most straightforward geometric techniques, and is widely employed. These facts make PCA the best geometric technique to start with, and the most important to understand.

Section 2.2.2 will briefly deal with the basic notions of distance and projection to be used in subsequent sections. The Euclidean distance, in particular, has previously been studied in Chapter 1. Although some background knowledge of linear algebra is useful for the following sections, it is not absolutely necessary.

Section 2.2.3 will present the basic PCA technique. It answers the following questions:

- how is PCA formulated as a geometric problem ?
- having defined certain geometric criteria, how is PCA formulated as an optimisation problem ?
- finally, how is PCA related to the eigen–decomposition of a matrix ?

Although a range of solution techniques for the eigenproblem are widely implemented in packages and subroutine libraries, an intuitively simple iterative solution technique is described later in Section 2.2.6.

Section 2.2.4 relates the use of PCA on the n (row) points in space $\mathbb{R}^m$ and the PCA of the m (column) points in space $\mathbb{R}^n$. Secondly, arising out of this mathematical relationship, the use of PCA as a data reduction technique is described. This section, then, answers the following questions:

- how is the PCA of an $n \times m$ matrix related to the PCA of the transposed $m \times n$ matrix ?
- how may the new axes derived — the principal components — be said to be linear combinations of the original axes ?
- how may PCA be understood as a series expansion ?
- in what sense does PCA provide a lower–dimensional approximation to the original data ?

In practice the variables on which the set of objects are characterised may often have differing means (consider the case of n students and m examination papers: the latter may be marked out of different totals, and furthermore it is to be expected that different examiners will set different standards of rigour). To circumvent this and similar difficulties, PCA is usually carried out on a transformed data matrix. Section 2.2.5 describes this transformation.

As an aid to the memory in the mathematics of the following sections, it is often convenient to note the dimensions of the vectors and matrices in use; it is vital of course that consistency in dimensions be maintained (e.g. if $\mathbf{u}$ is a vector in $\mathbb{R}^m$ of dimensions $m \times 1$, then premultiplication by the $n \times m$ matrix X will yield a vector $X\mathbf{u}$ of dimensions $n \times 1$).

2.2.2 Preliminaries — Scalar Product and Distance

This section defines a general Euclidean space. The usual Euclidean space is defined by means of its scalar product

$$\mathbf{x}'\mathbf{y} = \mathbf{y}'\mathbf{x}$$

(where $'$ denotes transpose). From the definition of scalar product, we may derive firstly the norm (here it is squared):

$$||\mathbf{x}||^2 = \mathbf{x}'\mathbf{x}$$

which is easily verified to be > 0 if $\mathbf{x} \neq \mathbf{0}$ and $= 0$ otherwise; and secondly distance (squared):

$$||\mathbf{x} - \mathbf{y}||^2 = (\mathbf{x} - \mathbf{y})'(\mathbf{x} - \mathbf{y})$$

which is usually denoted by $d^2(x, y)$.

A general Euclidean space allows for non–orthogonal axes using the following definitions:

Scalar product: $\mathbf{x}'M\mathbf{y} = \mathbf{y}'M\mathbf{x}$ (M must be symmetric).

Norm (squared): $||\mathbf{x}||_M^2 = \mathbf{x}'M\mathbf{x}$ (M must be positive definite so that the norm is positive, or zero if $\mathbf{x} = \mathbf{0}$).

Distance (squared): $||\mathbf{x} - \mathbf{y}||_M^2 = (\mathbf{x} - \mathbf{y})'M(\mathbf{x} - \mathbf{y})$.

M–**orthogonality:** $\mathbf{x}$ is M–orthogonal to $\mathbf{y}$ if $\mathbf{x}'M\mathbf{y} = \mathbf{0}$.

Unless otherwise specified, the usual Euclidean space associated with the identity matrix is understood (i.e. M has ones on the main diagonal and zeros elsewhere).

The projection of vector $\mathbf{x}$ onto axis $\mathbf{u}$ is

$$\mathbf{y} = \frac{\mathbf{x}'M\mathbf{u}}{||\mathbf{u}||_M} \; \mathbf{u}$$

i.e. the coordinate of the projection on the axis is $\mathbf{x}'M\mathbf{u}/||\mathbf{u}||_M$. This becomes $\mathbf{x}'M\mathbf{u}$ when the vector $\mathbf{u}$ is of unit length.

The cosine of the angle between vectors $\mathbf{x}$ and $\mathbf{y}$ in the usual Euclidean space is $\mathbf{x}'\mathbf{y}/||\mathbf{x}|| ||\mathbf{y}||$. (That is to say, we make use of the triangle whose vertices are the origin, the projection of $\mathbf{x}$ onto $\mathbf{y}$, and vector $\mathbf{x}$. The cosine of the angle between $\mathbf{x}$ and $\mathbf{y}$ is then the coordinate of the projection of $\mathbf{x}$ onto $\mathbf{y}$, divided by the — hypotenuse — length of $\mathbf{x}$). The correlation coefficient between two vectors is then simply the cosine of the angle between them, when the vectors have first been centred (i.e. $\mathbf{x} - \mathbf{g}$ and $\mathbf{y} - \mathbf{g}$ are used, where $\mathbf{g}$ is the overall centre of gravity).

The notions of distance and of projection will be central in the description of PCA (to be looked at next) and in the description of other geometric data analytic techniques studied in subsequent chapters.

2.2.3 The Basic Method

Consider a set of n objects measured on each of m attributes or variables. The $n \times m$ matrix of values will be denoted by $X = \{x_{ij}\}$ where i is a member of the set of objects and j a member of the attribute set. The objects may be regarded as row vectors in $\mathbb{R}^m$ and the attributes as column vectors in $\mathbb{R}^n$.

In $\mathbb{R}^m$, the space of objects, PCA searches for the best–fitting set of orthogonal axes to replace the initially–given set of m axes in this space. An analogous procedure is simultaneously carried out for the dual space, $\mathbb{R}^n$. First, the axis which best fits the objects/points in $\mathbb{R}^m$ is determined. If $\mathbf{u}$ is this vector, and is of unit length, then the product $X\mathbf{u}$ of $n \times m$ matrix by $m \times 1$ vector gives the projections of the n objects onto this axis.

The criterion of goodness of fit of this axis to the cloud of points will be defined as the squared deviation of the points from the axis. Minimizing the sum of distances between points and axis is equivalent to maximizing the sum of squared projections onto the axis (see Figure 2.3), i.e. to maximizing the variance (or *spread*) of the points when projected onto this axis.

The sum of squared projections of points on the new axis, for all points, is

$$(X\mathbf{u})'(X\mathbf{u}).$$

Such a quadratic form would increase indefinitely if $\mathbf{u}$ were arbitrarily large, so $\mathbf{u}$ is chosen — arbitrarily but reasonably — to be of unit length, i.e. $\mathbf{u}'\mathbf{u} = 1$. We seek a maximum of the quadratic form $\mathbf{u}'S\mathbf{u}$ (where $S = X'X$) subject to the constraint that $\mathbf{u}'\mathbf{u} = 1$. This is done by setting the derivative of the Lagrangian equal to zero. Differentiation of

$$\mathbf{u}'S\mathbf{u} - \lambda(\mathbf{u}'\mathbf{u} - 1)$$

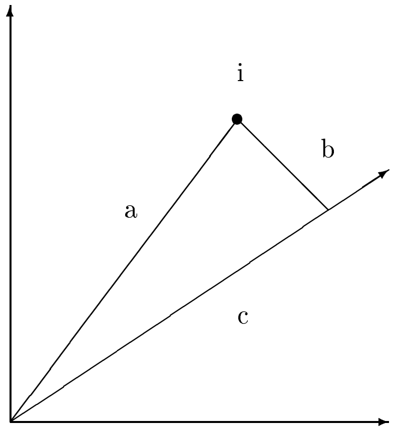
where λ is a Lagrange multiplier gives

$$2S\mathbf{u} - 2\lambda\mathbf{u}.$$

The optimal value of $\mathbf{u}$ (let us call it $\mathbf{u}_1$) is the solution of

$$S\mathbf{u} = \lambda\mathbf{u}.$$

The solution of this equation is well–known: $\mathbf{u}$ is the eigenvector associated with the eigenvalue λ of matrix S. Therefore the eigenvector of $X'X$, $\mathbf{u}_1$, is the axis sought, and the corresponding largest eigenvalue, λ_1, is a figure of merit for the axis, — it indicates the amount of variance explained by the axis (see next section).



For each point i,

b = distance of i from new axis,

c = projection of vector i onto new axis,

a = distance of point from origin.

By Pythagoras, $a^2 = b^2 + c^2$*; since a is constant, the choice of new axis which minimizes b simultaneously maximizes c (both of which are summed over all points, i).*

Figure 2.3: Projection onto an axis.

The second axis is to be orthogonal to the first, i.e. $\mathbf{u}'\mathbf{u}_1 = 0$, and satisfies the equation

$$\mathbf{u}'X'X\mathbf{u} - \lambda_2(\mathbf{u}'\mathbf{u} - 1) - \mu_2(\mathbf{u}'\mathbf{u}_1)$$

where λ_2 and μ_2 are Lagrange multipliers. Differentiating gives

$$2S\mathbf{u} - 2\lambda_2\mathbf{u} - \mu_2\mathbf{u}_1 .$$

This term is set equal to zero. Multiplying across by $\mathbf{u}_1'$ implies that μ_2 must equal 0. Therefore the optimal value of $\mathbf{u}$, $\mathbf{u}_2$, arises as another solution of $S\mathbf{u} = \lambda\mathbf{u}$. Thus λ_2 and $\mathbf{u}_2$ are the second largest eigenvalue and associated eigenvector of S.

The eigenvectors of $S = X'X$, arranged in decreasing order of corresponding eigenvalues, give the line of best fit to the cloud of points, the plane of best fit, the three–dimensional hyperplane of best fit, and so on for higher–dimensional subspaces of best fit. $X'X$ is referred to as the *sums of squares and cross products* matrix.

It has been assumed that the eigenvalues decrease in value: equal eigenvalues are possible, and indicate that equally privileged directions of elongation have been found. In practice, the set of equal eigenvalues may be arbitrarily ordered in any convenient fashion. Zero eigenvalues indicate that the space is actually of dimensionality less than expected (the points might, for instance, lie on a plane in three–dimensional space). The relevance of zero–valued eigenvalues is returned to in Section 2.3.1 below.

2.2.4 Dual Spaces and Data Reduction

In the dual space of attributes, $\mathbb{R}^n$, a PCA may equally well be carried out. For the line of best fit, $\mathbf{v}$, the following is maximized:

$$(X'\mathbf{v})'(X'\mathbf{v})$$

subject to

$$\mathbf{v}'\mathbf{v} = \mathbf{1}.$$

In $\mathbb{R}^m$ we arrived at

$$X'X\mathbf{u}_1 = \lambda_1\mathbf{u}_1.$$

By similar reasoning, in $\mathbb{R}^n$, we have

$$XX'\mathbf{v}_1 = \mu_1\mathbf{v}_1.$$

Premultiplying the first of these relationships by X yields

$$(XX')(X\mathbf{u}_1) = \lambda_1(X\mathbf{u}_1)$$

and so $\lambda_1 = \mu_1$ (because we have now arrived at two eigenvalue equations which are identical in form). We must be a little more attentive to detail before drawing a conclusion on the relationship between the eigenvectors in the two spaces: in order that these be of unit length, it may be verified that

$$\mathbf{v}_1 = \frac{1}{\sqrt{\lambda_1}}X\mathbf{u}_1$$

satisfies the foregoing equations. (The eigenvalue is necessarily positive, since if zero there are no associated eigenvectors.) Similarly,

$$\mathbf{v}_k = \frac{1}{\sqrt{\lambda_k}}X\mathbf{u}_k$$

and

$$\mathbf{u}_k = \frac{1}{\sqrt{\lambda_k}}X'\mathbf{v}_k$$

for the k^{th} largest eigenvalues and eigenvectors (or principal axes). Thus successive eigenvalues in both spaces are the same, and there is a simple linear transformation which maps the optimal axes in one space into those in the other.

The variance of the projections on a given axis in $\mathbb{R}^m$ is given by $(X\mathbf{u})'(X\mathbf{u})$, which by the eigenvector equation, is seen to equal λ.

In some software packages, the eigenvectors are rescaled so that $\sqrt{\lambda}\,\mathbf{u}$ and $\sqrt{\lambda}\,\mathbf{v}$ are used instead of $\mathbf{u}$ and $\mathbf{v}$. In this case, the *factor* $\sqrt{\lambda}\,\mathbf{u}$ gives the new, rescaled projections of the points in the space $\mathbb{R}^n$ (i.e. $\sqrt{\lambda}\,\mathbf{u} = X'\mathbf{v}$).

The coordinates of the new axes can be written in terms of the old coordinate system. Since

$$\mathbf{u} = \frac{1}{\sqrt{\lambda}} X' \mathbf{v}$$

each coordinate of the new vector $\mathbf{u}$ is defined as a linear combination of the initially–given vectors:

$$u_j = \sum_{i=1}^{n} \frac{1}{\sqrt{\lambda}} v_i x_{ij} = \sum_{i=1}^{n} c_i x_{ij}$$

(where $i \leq j \leq m$ and x_{ij} is the $(i,j)^{th}$ element of matrix X). Thus the j^{th} coordinate of the new vector is a *synthetic* value formed from the j^{th} coordinates of the given vectors (i.e. x_{ij} for all $1 \leq i \leq n$).

Since PCA in $\mathbb{R}^n$ and in $\mathbb{R}^m$ lead respectively to the finding of n and of m eigenvalues, and since in addition it has been seen that these eigenvalues are identical, it follows that the number of *non–zero eigenvalues* obtained in either space is less than or equal to $\min(n, m)$.

It has been seen that the eigenvectors associated with the p largest eigenvalues yield the best–fitting p–dimensional subspace of $\mathbb{R}^m$. A measure of the approximation is the percentage of variance explained by the subspace

$$\sum_{k \leq p} \lambda_k / \sum_{k=1}^{n} \lambda_k$$

expressed as a percentage.

An alternative view of PCA as an approximation to the given points is as follows. Taking the relationship between the unitary vectors in $\mathbb{R}^n$ and $\mathbb{R}^m$, $X\mathbf{u}_k = \sqrt{\lambda_k}\,\mathbf{v}_k$, postmultiplying by $\mathbf{u}'_k$, and summing gives

$$X \sum_{k=1}^{n} \mathbf{u}_k \mathbf{u}'_k = \sum_{k=1}^{n} \sqrt{\lambda_k}\,\mathbf{v}_k \mathbf{u}'_k.$$

The summation on the left hand side gives the identity matrix (this follows from the orthogonality of eigenvectors, so that the off–diagonal elements are zero; and the fact that the eigenvectors are of unit norm, so that the diagonal elements equal one), and so

$$X = \sum_{k=1}^{n} \sqrt{\lambda_k}\,\mathbf{v}_k \mathbf{u}'_k.$$

This series expansion of the given matrix, X, is termed the Karhunen– Loève expansion. The p best eigenvectors therefore approximate the original matrix by

$$\mathcal{X} = \sum_{k=1}^{p} \sqrt{\lambda_k}\,\mathbf{v}_k \mathbf{u}'_k$$

and if p is very much less than n, and $\mathcal{X}$ approximately equal to X, an appreciable economy in description has been obtained.

2.2.5 Practical Aspects

Since the variables or attributes under analysis are often very different (some will "shout louder" than others), it is usual to standardize each variable in the following way. If r_{ij} are the original measurements, then the matrix X of (i, j)–value

$$x_{ij} = \frac{r_{ij} - \overline{r}_j}{s_j \sqrt{n}}$$

where

$$\overline{r}_j = \frac{1}{n} \sum_{i=1}^{n} r_{ij}$$

and

$$s_j^2 = \frac{1}{n} \sum_{i=1}^{n} (r_{ij} - \overline{r}_j)^2$$

is submitted to the PCA (cf. Chapter 1, where standardization was discussed: the multiplicative constant, $1/\sqrt{n}$, in the definition of x_{ij} above is used so that we may conveniently define correlations). The matrix to be diagonalized, $X'X$, is then of $(j, k)^{th}$ term:

$$\rho_{jk} = \sum_{i=1}^{n} x_{ij} x_{ik} = \frac{1}{n} \sum_{i=1}^{n} (r_{ij} - \overline{r}_j)(r_{ik} - \overline{r}_k)/s_j s_k$$

which is the correlation coefficient between variables j and k.

Using the definitions of x_{ij} and s_j above, the distance between variables j and k is

$$d^2(j, k) = \sum_{i=1}^{n} (x_{ij} - x_{ik})^2 = \sum_{i=1}^{n} x_{ij}^2 + \sum_{i=1}^{n} x_{ik}^2 - 2 \sum_{i=1}^{n} x_{ij} x_{ik}$$

and, substituting, the first two terms both yield 1, giving

$$d^2(j, k) = 2(1 - \rho_{jk}).$$

Thus the distance between variables is directly proportional to the correlation between them.

The distance between row vectors is

$$d^2(i, h) = \sum_j (x_{ij} - x_{hj})^2 = \sum_j (\frac{r_{ij} - r_{hj}}{\sqrt{n} s_j})^2 = (\mathbf{r}_i - \mathbf{r}_h)' M (\mathbf{r}_i - \mathbf{r}_h)$$

where $\mathbf{r}_i$ and $\mathbf{r}_h$ are column vectors (of dimensions $m \times 1$) and M is the $m \times m$ diagonal matrix of j^{th} element $1/ns_j^2$. Therefore d is a Euclidean distance associated with matrix M. Note that the row points are now centred but the column points are not: therefore the latter may well appear in one quadrant on output listings.

Analysis of the matrix of $(j,k)^{th}$ term ρ_{jk} as defined above is PCA on a *correlation* matrix. Inherent in this, as has been seen, is that the row vectors are centred and reduced.

If, instead, centring was acceptable but not the rescaling of the variance, we would be analysing the matrix of $(j,k)^{th}$ term

$$c_{jk} = \frac{1}{n}\sum_{i=1}^{n}(r_{ij} - \overline{r}_j)(r_{ik} - \overline{r}_k).$$

In this case we have PCA of the *variance-covariance* matrix.

The following should be noted.

- We may speak about carrying out a PCA on the correlation matrix, on a variance–covariance matrix, or on the "sums of squares and cross–products" matrix. These relate to intermediate matrices which are most often determined by the PCA program. The user should however note the effect of transformations on his/her data (standardization, centring) which are inherent in these different options.
- Rarely is it recommendable to carry out a PCA on the "sums of squares and cross–products" matrix; instead some transformation of the original data is usually necessary. In the absence of justification for treating the data otherwise, the most recommendable strategy is to use the option of PCA on a correlation matrix.
- All the quantities defined above (standard deviation, variances and covariances, etc.) have been in *population* rather than in *sample* terms. That is to say, the data under examination is taken as all we can immediately study, rather than being representative of a greater underlying distribution of points. Not all packages and program libraries share this viewpoint, and hence discrepancies may be detected between results obtained in practice from different sources.

2.2.6 Iterative Solution of Eigenvalue Equations

A simple iterative scheme for solving the eigenvector equation $A\mathbf{u} = \lambda\mathbf{u}$ is as follows.

Choose some trial vector, $\mathbf{t}_0$: e.g. $(1,1,\ldots,1)$. Then define $\mathbf{t}_1, \mathbf{t}_2, \ldots$:

$$\begin{array}{ll} A\mathbf{t}_0 = \mathbf{x}_0 & \mathbf{t}_1 = \mathbf{x}_0/\sqrt{\mathbf{x}_0'\mathbf{x}_0} \\ A\mathbf{t}_1 = \mathbf{x}_1 & \mathbf{t}_2 = \mathbf{x}_1/\sqrt{\mathbf{x}_1'\mathbf{x}_1} \\ A\mathbf{t}_2 = \mathbf{x}_2 & \mathbf{t}_3 = \ldots \end{array}$$

We halt when there is sufficient convergence: $|\mathbf{t}_n - \mathbf{t}_{n+1}| \leq \epsilon$, for some small, real ϵ ; or when the number of iterations exceeds some maximum (e.g. 15). In the latter case, another choice of $\mathbf{t}_0$ will be tried.

If there is convergence, $\mathbf{t}_n = \mathbf{t}_{n+1}$, we have the following:

$$A\mathbf{t}_n = \mathbf{x}_n$$

$$\mathbf{t}_{n+1} = \mathbf{x}_n/\sqrt{\mathbf{x}_n'\mathbf{x}_n}.$$

Substituting for $\mathbf{x}_n$ in the first of these two equations gives

$$A\mathbf{t}_n = \sqrt{\mathbf{x}'_n\mathbf{x}_n} \;\; \mathbf{t}_{n+1}.$$

Hence, since $\mathbf{t}_n = \mathbf{t}_{n+1}$, $\mathbf{t}_n$ is none other than the eigenvector sought; and the associated eigenvalue is $\sqrt{\mathbf{x}'_n\mathbf{x}_n}$.

The second eigenvector and associated eigenvalue may be found by carrying out a similar iterative algorithm on a matrix where the effects of $\mathbf{u}_1$ and λ_1 have been *partialled out*: $A_{(2)} = A - \lambda_1\mathbf{u}_1\mathbf{u}'_1$. Let us prove that $A_{(2)}$ removes the effects due to the first eigenvector and eigenvalue. We have $A\mathbf{u} = \lambda\mathbf{u}$. Therefore $A\mathbf{u}\mathbf{u}' = \lambda\mathbf{u}\mathbf{u}'$; or equivalently, $A\mathbf{u}_k\mathbf{u}'_k = \lambda_k\mathbf{u}_k\mathbf{u}'_k$ for each eigenvalue. Summing over k gives:

$$A\sum_k \mathbf{u}_k\mathbf{u}'_k = \sum_k \lambda_k\mathbf{u}_k\mathbf{u}'_k.$$

The summed term on the left hand side equals the identity matrix (this has already been seen in Section 2.2.4 above). Therefore we have

$$A = \lambda_1\mathbf{u}_1\mathbf{u}'_1 + \lambda_2\mathbf{u}_2\mathbf{u}'_2 + \ldots$$

From this *spectral decomposition* of matrix A, we may successively remove the effects of the eigenvectors and eigenvalues as they are obtained.

Many other algorithms are available for solving eigen–equations: Chambers (1977) may be consulted for algorithms which are more likely to be implemented in the major commercial packages or subroutine libraries; and Smith *et al.* (1976) contains many Fortran implementations.

2.3 Examples and Bibliography

2.3.1 General Remarks

Among the objectives of PCA are the following:

1. dimensionality reduction;
2. the determining of linear combinations of variables;
3. feature selection: the choosing of the most useful variables;
4. visualization of multidimensional data;
5. identification of underlying variables;
6. identification of groups of objects or of outliers.

The tasks required of the analyst to carry these out are as follows:

1. In the case of a table of dimensions $n \times m$, each of the n rows or objects can be regarded as an m–dimensional vector. Finding a set of $m' < m$ principal axes allows the objects to be adequately characterised on a smaller number of (artificial) variables. This is advantageous as a prelude to further analysis as the $m - m'$ dimensions may often be ignored as constituting noise; and, secondly, for storage economy (sufficient information from the

initial table is now represented in a table with $m' < m$ columns). Reduction of dimensionality is practicable if the first m' new axes account for approximately 75 % or more of the variance. There is no set threshold, — the analyst must judge. The cumulative percentage of variance explained by the principal axes is consulted in order to make this choice.

2. If the eigenvalue is zero, the variance of projections on the associated eigenvector is zero. Hence the eigenvector is reduced to a point. If this point is additionally the origin (i.e. the data are centred), then we have $X\mathbf{u} = \lambda\mathbf{u} = \mathbf{0}$. I.e. $\sum_j u_j \mathbf{x}_j = 0$, where $\mathbf{x}_j$ is the j^{th} column vector of X. This allows linear combinations between the variables to be found. In fact, we can go a good deal further: by analysing second–order variables, defined from the given variables, quadratic dependencies can be staightforwardly sought. This means, for example, that in analysing three variables, y_1, y_2, and y_3, we would also input the variables y_1^2, y_2^2, y_3^2, y_1y_2, y_1y_3, and y_2y_3. If the linear combination

$$y_1 = c_1 y_2^2 + c_2 y_1 y_2$$

exists, then we would find it. Similarly we could feed in the logarithms or other functions of variables.

3. In feature selection we want to simplify the task of characterising each object by a set of attributes. Linear combinations among attributes must be found; highly correlated attributes (i.e. closely located attributes in the new space: cf. section 2.2.5) allow some attributes to be removed from consideration; and the proximity of attributes to the new axes indicate the more relevant and important attributes.

4. In order to provide a convenient representation of multidimensional data, planar plots are necessary. An important consideration is the adequacy of the planar representation: the percentage variance explained by the pair of axes defining the plane must be looked at here.

5. PCA is often motivated by the search for latent variables. Often it is relatively easy to label the highest or second highest components, but it becomes increasingly difficult as less relevant axes are examined. The objects with the highest loadings or projections on the axes (i.e. those which are placed towards the extremities of the axes) are usually worth examining: the axis may be characterisable as a spectrum running from a small number of objects with high positive loadings to those with high negative loadings.

6. A visual inspection of a planar plot indicates which objects are grouped together, thus indicating that they belong to the same family or result from the same process. Anomalous objects can also be detected, and in some cases it might be of interest to redo the analysis with these excluded because of the perturbation they introduce.

2.3.2 Artificial Data

Data of the following characteristics was generated in order to look at the use of PCA for ascertaining quadratic dependencies. Thirty objects were used, and in total 5 variables.

$y_{1j} = -1.4, -1.3, \ldots, 1.5$

$y_{2j} = 2.0 - y_{1j}^2$

$y_{3j} = y_{1j}^2$

$y_{4j} = y_{2j}^2$

$y_{5j} = y_{1j} y_{2j}$

The output obtained follows. We could plot the row–points or column–points in the principal plane, using the coordinates found. The fifth eigenvalue is zero; hence the variance of the associated principal component is zero. Since we know in addition that the eigenvectors are centred, we therefore have the equation:

$$0.7071\mathbf{y}_2 + 0.7071\mathbf{y}_3 = 0.0$$

Note again that variables $\mathbf{y}_2$ and $\mathbf{y}_3$ have been redefined so that each value is centred (see section 2.2.5 above regarding PCA of a covariance matrix).

```
COVARIANCE MATRIX FOLLOWS.

  22.4750
  -2.2475   13.6498
   2.2475  -13.6498   13.6498
  -2.9262   28.0250  -28.0250   62.2917
  14.5189    0.5619   -0.5619    0.7316   17.3709


EIGENVALUES FOLLOW.

Eigenvalues       As Percentages    Cumul. Percentages
-----------       --------------    ------------------
  88.3852            68.2842              68.2842
  34.5579            26.6985              94.9828
   5.2437             4.0512              99.0339
   1.2505             0.9661             100.0000
   0.0000             0.0000             100.0000


EIGENVECTORS FOLLOW.

 VBLE.   EV-1    EV-2    EV-3    EV-4    EV-5
------  ------  ------  ------  ------  ------
   1   -0.0630  0.7617  0.6242 -0.1620  0.0000
   2    0.3857  0.0067 -0.1198 -0.5803  0.7071
   3   -0.3857 -0.0067  0.1198  0.5803  0.7071
   4    0.8357  0.0499  0.1593  0.5232  0.0000
   5    0.0018  0.6460 -0.7458  0.1627  0.0000


PROJECTIONS OF ROW-POINTS FOLLOW.

OBJECT  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5
------  ------  ------  ------  ------  ------
```

```
    1   -2.5222 -1.2489 -0.9036  0.5779  0.0000
    2   -2.2418 -1.3886 -0.6320  0.2413  0.0000
    3   -1.8740 -1.4720 -0.3942  0.0050  0.0000
    4   -1.4437 -1.5046 -0.1905 -0.1478  0.0000
    5   -0.9741 -1.4915 -0.0209 -0.2324  0.0000
    6   -0.4862 -1.4379  0.1153 -0.2630  0.0000
    7    0.0009 -1.3488  0.2187 -0.2525  0.0000
    8    0.4702 -1.2292  0.2906 -0.2125  0.0000
    9    0.9065 -1.0837  0.3327 -0.1535  0.0000
   10    1.2969 -0.9171  0.3469 -0.0845  0.0000
   11    1.6303 -0.7338  0.3355 -0.0136  0.0000
   12    1.8977 -0.5383  0.3014  0.0528  0.0000
   13    2.0920 -0.3349  0.2477  0.1093  0.0000
   14    2.2083 -0.1278  0.1779  0.1516  0.0000
   15    2.2434  0.0791  0.0958  0.1771  0.0000
   16    2.1964  0.2817  0.0059  0.1840  0.0000
   17    2.0683  0.4762 -0.0873  0.1720  0.0000
   18    1.8620  0.6590 -0.1787  0.1420  0.0000
   19    1.5826  0.8264 -0.2629  0.0962  0.0000
   20    1.2371  0.9751 -0.3341  0.0381  0.0000
   21    0.8345  1.1016 -0.3860 -0.0278  0.0000
   22    0.3858  1.2027 -0.4121 -0.0955  0.0000
   23   -0.0959  1.2755 -0.4055 -0.1578  0.0000
   24   -0.5957  1.3168 -0.3587 -0.2062  0.0000
   25   -1.0964  1.3238 -0.2641 -0.2312  0.0000
   26   -1.5792  1.2938 -0.1135 -0.2216  0.0000
   27   -2.0227  1.2242  0.1015 -0.1653  0.0000
   28   -2.4042  1.1124  0.3898 -0.0489  0.0000
   29   -2.6984  0.9561  0.7607  0.1424  0.0000
   30   -2.8783  0.7530  1.2238  0.4245  0.0000


 PROJECTIONS OF COLUMN-POINTS FOLLOW.

  VBLE.  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5
 ------  ------  ------  ------  ------  ------
    1   -0.6739  2.7008  0.3289 -0.0203  0.0000
    2    4.1259  0.0236 -0.0632 -0.0726  0.0000
    3   -4.1259 -0.0236  0.0632  0.0726  0.0000
    4    8.9388  0.1768  0.0840  0.0654  0.0000
    5    0.0196  2.2906 -0.3930  0.0203  0.0000
```

2.3.3 Examples from Astronomy

PCA has been a fairly widely used technique in astronomy. The following list does not aim to be comprehensive, but indicates instead the types of problems to which PCA can be applied. It is also hoped that it may provide a convenient entry–point to literature on a topic of interest. References below are concerned with stellar parallaxes; a large number are concerned with the study of galaxies; and a large number relate also to spectral reduction.

1. A. Bijaoui, "Application astronomique de la compression de l'information", *Astronomy and Astrophysics*, **30**, 199–202, 1974.

2. A. Bijaoui, SAI Library, Algorithms for Image Processing, Nice Observatory, Nice, 1985.

   (A large range of subroutines for image processing, including the Karhunen–Loève expansion.)

3. P. Brosche, “The manifold of galaxies: Galaxies with known dynamical properties”, *Astronomy and Astrophysics*, **23**, 259–268, 1973.

4. P. Brosche and F.T. Lentes, “The manifold of globular clusters”, *Astronomy and Astrophysics*, **139**, 474–476, 1984.

5. V. Bujarrabal, J. Guibert and C. Balkowski, “Multidimensional statistical analysis of normal galaxies”, *Astronomy and Astrophysics*, **104**, 1–9, 1981.

6. R. Buser, “A systematic investigation of multicolor photometric systems. I. The UBV, RGU and *uvby* systems.”, *Astronomy and Astrophysics*, **62**, 411–424, 1978.

7. C.A. Christian and K.A. Janes, “Multivariate analysis of spectrophotometry”. *Publications of the Astronomical Society of the Pacific*, **89**, 415–423, 1977.

8. C.A. Christian, “Identification of field stars contaminating the colour–magnitude diagram of the open cluster Be 21”, *The Astrophysical Journal Supplement Series*, **49**, 555–592, 1982.

9. T.J. Deeming, “Stellar spectral classification. I. Application of component analysis”, *Monthly Notices of the Royal Astronomical Society*, **127**, 493–516, 1964.

   (An often referenced work.)

10. T.J. Deeming, “The analysis of linear correlation in astronomy”, *Vistas in Astronomy*, **10**, 125, 1968.

    (For regression also.)

11. G. Efstathiou and S.M. Fall, “Multivariate analysis of elliptical galaxies”, *Monthly Notices of the Royal Astronomical Society*, **206**, 453–464, 1984.

12. S.M. Faber, “Variations in spectral–energy distributions and absorption–line strengths among elliptical galaxies”, *The Astrophysical Journal*, **179**, 731–754, 1973.

13. M. Fofi, C. Maceroni, M. Maravalle and P. Paolicchi, “Statistics of binary stars. I. Multivariate analysis of spectroscopic binaries”, *Astronomy and Astrophysics*, **124**, 313–321, 1983.

    (PCA is used, together with a non–hierarchical clustering technique.)

14. M. Fracassini, L.E. Pasinetti, E. Antonello and G. Raffaelli, “Multivariate analysis of some ultrashort period Cepheids (USPC)”, *Astronomy and Astrophysics*, **99**, 397–399, 1981.

15. M. Fracassini, G. Manzotti, L.E. Pasinetti, G. Raffaelli, E. Antonello and L. Pastori, “Application of multivariate analysis to the parameters of astrophysical objects”, in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 21–25, 1983.

16. P. Galeotti, “A statistical analysis of metallicity in spiral galaxies”, *Astrophysics and Space Science*, **75**, 511–519, 1981.

17. A. Heck, “An application of multivariate statistical analysis to a photometric catalogue”, *Astronomy and Astrophysics*, **47**, 129–135, 1976.

    (PCA is used, along with regression and discriminant analysis.)

18. A. Heck, D. Egret, Ph. Nobelis and J.C. Turlot, “Statistical confirmation of the UV spectral classification system based on IUE low–dispersion spectra”, *Astrophysics and Space Science*, **120**, 223–237, 1986.

    (Many other articles by these authors, which also make use of PCA, are referenced in the above.)

19. S.J. Kerridge and A.R. Upgren, “The application of multivariate analysis to parallax solutions. II. Magnitudes and colours of comparison stars”, *The Astronomical Journal*, **78**, 632–638, 1973.

    (See also Upgren and Kerridge, 1971, referenced below.)

20. J. Koorneef, “On the anomaly of the far UV extinction in the 30 Doradus region”, *Astronomy and Astrophysics*, **64**, 179–193, 1978.

    (PCA is used for deriving a photometric index from 5–channel photometric data.)

21. M.J. Kurtz, “Automatic spectral classification”, PhD Thesis, Dartmouth College, New Hampshire, 1982.

22. F.T. Lentes, “The manifold of spheroidal galaxies”, *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 73–76, 1983.

23. D. Massa and C.F. Lillie, “Vector space methods of photometric analysis: applications to O stars and interstellar reddening”, *The Astrophysical Journal*, **221**, 833–850, 1978.

24. D. Massa, “Vector space methods of photometric analysis. III. The two components of ultraviolet reddening”, *The Astronomical Journal*, **85**, 1651–1662, 1980.

25. B. Nicolet, “Geneva photometric boxes. I. A topological approach of photometry and tests.”, *Astronomy and Astrophysics*, **97**, 85–93, 1981.

    (PCA is used on colour indices.)

26. S. Okamura, K. Kodaira and M. Watanabe, “Digital surface photometry of galaxies toward a quantitative classification. III. A mean concentration index as a parameter representing the luminosity distribution”, *The Astrophysical Journal*, **280**, 7–14, 1984.

27. S. Okamura, “Global structure of Virgo cluster galaxies”, in O.–G. Richter and B. Binggeli (eds.), Proceedings of ESO Workshop on The Virgo Cluster of Galaxies, ESO Conference and Workshop Proceedings No. 20, 201–215, 1985.

28. D. Pelat, “A study of H I absorption using Karhunen–Loève series”, *Astronomy and Astrophysics*, **40**, 285–290, 1975.

29. A. W. Strong, "Data analysis in gamma–ray astronomy: multivariate likelihood method for correlation studies", *Astronomy and Astrophysics*, **150**, 273–275, 1985.

    (The method presented is not linked to PCA, but in dealing with the eigenreduction of a correlation matrix it is clearly very closely related.)

30. B. Takase, K. Kodaira and S. Okamura, *An Atlas of Selected Galaxies*, University of Tokyo Press, VNU Science Press, 1984.

31. D.J. Tholen, "Asteroid taxonomy from cluster analysis of photometry", PhD Thesis, University of Arizona, 1984.

32. A.R. Upgren and S.J. Kerridge, "The application of multivariate analysis to parallax solutions. I. Choice of reference frames", *The Astronomical Journal*, **76**, 655–664, 1971.

    (See also Kerridge and Upgren, 1973, referenced above.)

33. J.P. Vader, "Multivariate analysis of elliptical galaxies in different environments", *The Astrophysical Journal*, **306**, 390–400, 1986.

    (The Virgo and Coma clusters are studied.)

34. C.A. Whitney, "Principal components analysis of spectral data. I. Methodology for spectral classification", *Astronomy and Astrophysics Supplement Series*, **51**, 443–461, 1983.

35. B.C. Whitmore, "An objective classification system for spiral galaxies. I. The two dominant dimensions", *The Astrophysical Journal*, **278**, 61–80, 1984.

2.3.4 General References

1. T.W. Anderson, *An Introduction to Multivariate Statistical Analysis*, Wiley, New York, 1984 (2nd ed.).

   (For inferential aspects relating to PCA.)

2. J.M. Chambers, *Computational Methods for Data Analysis*, Wiley, New York, 1977.

3. C. Chatfield and A.J. Collins, *Introduction to Multivariate Analysis*, Chapman and Hall, 1980.

   (A good introductory textbook.)

4. R. Gnanadesikan, *Methods for Statistical Data Analysis of Multivariate Observations*, Wiley, New York, 1977.

   (For details of PCA, clustering and discrimination.)

5. M. Kendall, *Multivariate Analysis*, Griffin, London, 1980 (2nd ed.).

   (Dated in relation to computing techniques, but exceptionally clear and concise in its treatment of many practical problems.)

6. L. Lebart, A. Morineau and K.M. Warwick, *Multivariate Descriptive Statistical Analysis*, Wiley, New York, 1984.

   (An excellent geometric treatment of PCA.)

7. F.H.C. Marriott, *The Interpretation of Multiple Observations*, Academic Press, New York, 1974.

   (A short, readable textbook.)

8. B.T. Smith *et al.*, *Matrix Eigensystem Routines — EISPACK Guide*, Lecture Notes in Computer Science 6, Springer Verlag, Berlin and New York, 1976.

2.4 Software and Sample Implementation

2.4.1 Fortran Code, Sample Input and Output

The following is a listing of the Principal Components Analysis program. Note that input and output is handled by subroutines. It may, in certain cases, be desirable to alter the output formats. Alternatively, there is a “switch” which allows no output to be produced and instead a driving routine can use information passed back in the arrays and vectors, as indicated.

No extra subroutines are required to run the program, except for a driving routine.

The following subroutines are called from the main PCA routine.

1. CORCOL determines correlations.
2. COVCOL determines covariances.
3. SCPCOL determines sums of squares and cross–products.
4. TRED2 reduces a symmetric matrix to tridiagonal form.
5. TQL2 derives eigenvalues and eigenvectors of a tridiagonal matrix.
6. OUTMAT outputs a matrix.
7. OUTHMT outputs a diagonal half–matrix.
8. OUTEVL outputs eigenvalues.
9. OUTEVC outputs eigenvectors.
10. OUTPRX outputs projections of row–points.
11. OUTPRY outputs projections of column–points.
12. PROJX determines projections of row–points.
13. PROJY determines projections of column–points.

2.4.2 Fortran Progam Listing

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Carry out a PRINCIPAL COMPONENTS ANALYSIS
C              (KARHUNEN-LOEVE EXPANSION).
C
C  To call: CALL PCA(N,M,DATA,METHOD,IPRINT,A1,W1,W2,A2,IERR)
C           where
C
C
C  N, M  : integer dimensions of ...
C  DATA  : input data.
C          On output, DATA contains in first 7 columns the
C          projections of the row-points on the first 7
C          principal components.
C  METHOD: analysis option.
C          = 1: on sums of squares & cross products matrix.
```

```
C          = 2: on covariance matrix.
C          = 3: on correlation matrix.
C  IPRINT: print options.
C          = 0: no printed output- arrays/vectors, only, contain
C               items calculated.
C          = 1: eigenvalues, only, output.
C          = 2: printed output, in addition, of correlation (or
C               other) matrix, eigenvalues and eigenvectors.
C          = 3: full printing of items calculated.
C  A1    : correlation, covariance or sums of squares &
C          cross-products matrix, dimensions M * M.
C          On output, A1 contains in the first 7 columns the
C          projections of the column-points on the first 7
C          principal components.
C  W1,W2 : real vectors of dimension M (see called routines for
C          use).
C          On output, W1 contains the cumulative percentage
C          variances associated with the principal components.
C  A2    : real array of dimensions M * M (see routines for use).
C  IERR  : error indicator (normally zero).
C
C
C  Inputs here are N, M, DATA, METHOD, IPRINT (and IERR).
C  Output information is contained in DATA, A1, and W1.
C  All printed outputs are carried out in easily recognizable sub-
C  routines called from the first subroutine following.
C
C  If IERR > 0, then its value indicates the eigenvalue for which
C  no convergence was obtained.
C
C-----------------------------------------------------------------
        SUBROUTINE PCA(N,M,DATA,METHOD,IPRINT,A,W,FV1,Z,IERR)
        REAL    DATA(N,M), A(M,M), W(M), FV1(M), Z(M,M)
C
        IF (METHOD.EQ.1) GOTO 100
        IF (METHOD.EQ.2) GOTO 400
C       If method.eq.3 or otherwise ...
        GOTO 700
C
C          Form sums of squares and cross-products matrix.
C
  100   CONTINUE
        CALL SCPCOL(N,M,DATA,A)
C
        IF (IPRINT.GT.1) CALL OUTHMT(METHOD,M,A)
C
C          Now do the PCA.
C
        GOTO 1000
C
C          Form covariance matrix.
C
  400   CONTINUE
        CALL COVCOL(N,M,DATA,W,A)
C
        IF (IPRINT.GT.1) CALL OUTHMT(METHOD,M,A)

C
C          Now do the PCA.
C
        GOTO 1000
C
```

```
C          Construct correlation matrix.
C
  700   CONTINUE
        CALL CORCOL(N,M,DATA,W,FV1,A)
C
        IF (IPRINT.GT.1) CALL OUTHMT(METHOD,M,A)

C
C          Now do the PCA.
C
        GOTO 1000
C
C          Carry out eigenreduction.
C
 1000   M2 = M
        CALL TRED2(M,M2,A,W,FV1,Z)
        CALL TQL2(M,M2,W,FV1,Z,IERR)
        IF (IERR.NE.0) GOTO 9000
C
C          Output eigenvalues and eigenvectors.
C
        IF (IPRINT.GT.0) CALL OUTEVL(N,M,W)
        IF (IPRINT.GT.1) CALL OUTEVC(N,M,Z)
C
C          Determine projections and output them.
C
        CALL PROJX(N,M,DATA,Z,FV1)
        IF (IPRINT.EQ.3) CALL OUTPRX(N,M,DATA)
        CALL PROJY(M,W,A,Z,FV1)
        IF (IPRINT.EQ.3) CALL OUTPRY(M,A)
C
 9000   RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Determine correlations of columns.
C  First determine the means of columns, storing in WORK1.
C
C--------------------------------------------------------
        SUBROUTINE CORCOL(N,M,DATA,WORK1,WORK2,OUT)
        DIMENSION       DATA(N,M), OUT(M,M), WORK1(M), WORK2(M)
        DATA            EPS/1.E-10/
C
        DO 30 J = 1, M
           WORK1(J) = 0.0
           DO 20 I = 1, N
              WORK1(J) = WORK1(J) + DATA(I,J)
   20      CONTINUE
           WORK1(J) = WORK1(J)/FLOAT(N)
   30   CONTINUE
C
C          Next det. the std. devns. of cols., storing in WORK2.
C
        DO 50 J = 1, M
           WORK2(J) = 0.0
           DO 40 I = 1, N
              WORK2(J) = WORK2(J) + (DATA(I,J)
     X                  -WORK1(J))*(DATA(I,J)-WORK1(J))
   40      CONTINUE
           WORK2(J) = WORK2(J)/FLOAT(N)
           WORK2(J) = SQRT(WORK2(J))
           IF (WORK2(J).LE.EPS) WORK2(J) = 1.0
   50   CONTINUE
C
C          Now centre and reduce the column points.
C
        DO 70 I = 1, N
           DO 60 J = 1, M
              DATA(I,J) = (DATA(I,J)
     X                    -WORK1(J))/(SQRT(FLOAT(N))*WORK2(J))
   60      CONTINUE
   70   CONTINUE
C
C          Finally calc. the cross product of the data matrix.
C
        DO 100 J1 = 1, M-1
           OUT(J1,J1) = 1.0
           DO 90 J2 = J1+1, M
              OUT(J1,J2) = 0.0
              DO 80 I = 1, N
                 OUT(J1,J2) = OUT(J1,J2) + DATA(I,J1)*DATA(I,J2)
   80         CONTINUE
              OUT(J2,J1) = OUT(J1,J2)
   90      CONTINUE
  100   CONTINUE
        OUT(M,M) = 1.0
C
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Determine covariances of columns.
C  First determine the means of columns, storing in WORK.
C
C------------------------------------------------------
        SUBROUTINE COVCOL(N,M,DATA,WORK,OUT)
        DIMENSION       DATA(N,M), OUT(M,M), WORK(M)
C
        DO 30 J = 1, M
           WORK(J) = 0.0
           DO 20 I = 1, N
              WORK(J) = WORK(J) + DATA(I,J)
   20      CONTINUE
           WORK(J) = WORK(J)/FLOAT(N)
   30   CONTINUE
C
C          Now centre the column points.
C
        DO 50 I = 1, N
           DO 40 J = 1, M
              DATA(I,J) = DATA(I,J)-WORK(J)
   40      CONTINUE
   50   CONTINUE
C
C          Finally calculate the cross product matrix of the
C          redefined data matrix.
C
        DO 80 J1 = 1, M
           DO 70 J2 = J1, M
              OUT(J1,J2) = 0.0
              DO 60 I = 1, N
                 OUT(J1,J2) = OUT(J1,J2) + DATA(I,J1)*DATA(I,J2)
   60         CONTINUE
              OUT(J2,J1) = OUT(J1,J2)
   70      CONTINUE
   80   CONTINUE
C
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Determine sums of squares and cross-products of columns.
C
C---------------------------------------------------------
        SUBROUTINE SCPCOL(N,M,DATA,OUT)
        DIMENSION       DATA(N,M), OUT(M,M)
C
        DO 30 J1 = 1, M
           DO 20 J2 = J1, M
              OUT(J1,J2) = 0.0
              DO 10 I = 1, N
                 OUT(J1,J2) = OUT(J1,J2) + DATA(I,J1)*DATA(I,J2)
   10         CONTINUE
              OUT(J2,J1) = OUT(J1,J2)
   20      CONTINUE
   30   CONTINUE
C
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C Reduce a real, symmetric matrix to a symmetric, tridiagonal
C matrix.
C
C To call:    CALL TRED2(NM,N,A,D,E,Z)    where
C
C NM = row dimension of A and Z;
C N = order of matrix A (will always be <= NM);
C A = symmetric matrix of order N to be reduced to tridiag. form;
C D = vector of dim. N containing, on output, diagonal elts. of
C     tridiagonal matrix.
C E = working vector of dim. at least N-1 to contain subdiagonal
C     elements.
C Z = matrix of dims. NM by N containing, on output, orthogonal
C     transformation matrix producing the reduction.
C
C Normally a call to TQL2 will follow the call to TRED2 in order to
C produce all eigenvectors and eigenvalues of matrix A.
C
C Algorithm used: Martin et al., Num. Math. 11, 181-195, 1968.
C
C Reference: Smith et al., Matrix Eigensystem Routines - EISPACK
C Guide, Lecture Notes in Computer Science 6, Springer-Verlag,
C 1976, pp. 489-494.
C
C----------------------------------------------------------------
        SUBROUTINE TRED2(NM,N,A,D,E,Z)
        REAL A(NM,N),D(N),E(N),Z(NM,N)
C
        DO 100 I = 1, N
           DO 100 J = 1, I
              Z(I,J) = A(I,J)
  100   CONTINUE
        IF (N.EQ.1) GOTO 320
        DO 300 II = 2, N
           I = N + 2 - II
           L = I - 1
           H = 0.0
           SCALE = 0.0
           IF (L.LT.2) GOTO 130
           DO 120 K = 1, L
              SCALE = SCALE + ABS(Z(I,K))
  120      CONTINUE
           IF (SCALE.NE.0.0) GOTO 140
  130      E(I) = Z(I,L)
           GOTO 290
  140      DO 150 K = 1, L
              Z(I,K) = Z(I,K)/SCALE
              H = H + Z(I,K)*Z(I,K)
  150      CONTINUE
C
           F = Z(I,L)
           G = -SIGN(SQRT(H),F)
           E(I) = SCALE * G
           H = H - F * G
           Z(I,L) = F - G
           F = 0.0
C
           DO 240 J = 1, L
              Z(J,I) = Z(I,J)/H
              G = 0.0
C             Form element of A*U.
              DO 180 K = 1, J
                 G = G + Z(J,K)*Z(I,K)
  180         CONTINUE
              JP1 = J + 1
              IF (L.LT.JP1) GOTO 220
              DO 200 K = JP1, L
                 G = G + Z(K,J)*Z(I,K)
  200         CONTINUE
C             Form element of P where P = I - U U' / H .
  220         E(J) = G/H
              F = F + E(J) * Z(I,J)
  240      CONTINUE
           HH = F/(H + H)
C          Form reduced A.
           DO 260 J = 1, L
              F = Z(I,J)
              G = E(J) - HH * F
              E(J) = G
              DO 250 K = 1, J
                 Z(J,K) = Z(J,K) - F*E(K) - G*Z(I,K)
  250         CONTINUE
  260      CONTINUE
  290      D(I) = H
  300   CONTINUE
  320   D(1) = 0.0
        E(1) = 0.0
C       Accumulation of transformation matrices.
        DO 500 I = 1, N
           L = I - 1
           IF (D(I).EQ.0.0) GOTO 380
           DO 360 J = 1, L
              G = 0.0
              DO 340 K = 1, L
                 G = G + Z(I,K) * Z(K,J)
  340         CONTINUE
              DO 350 K = 1, L
```

```
                 Z(K,J) = Z(K,J) - G * Z(K,I)
  350         CONTINUE
  360      CONTINUE
  380      D(I) = Z(I,I)
           Z(I,I) = 1.0
           IF (L.LT.1) GOTO 500
           DO 400 J = 1, L
              Z(I,J) = 0.0
              Z(J,I) = 0.0
  400      CONTINUE
  500   CONTINUE
C
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C Determine eigenvalues and eigenvectors of a symmetric,
C tridiagonal matrix.
C
C To call:    CALL TQL2(NM,N,D,E,Z,IERR)    where
C
C NM = row dimension of Z;
C N = order of matrix Z;
C D = vector of dim. N containing, on output, eigenvalues;
C E = working vector of dim. at least N-1;
C Z = matrix of dims. NM by N containing, on output, eigenvectors;
C IERR = error, normally 0, but 1 if no convergence.
C
C Normally the call to TQL2 will be preceded by a call to TRED2 in
C order to set up the tridiagonal matrix.
C
C Algorithm used: QL method of Bowdler et al., Num. Math. 11,
C 293-306, 1968.
C
C Reference: Smith et al., Matrix Eigensystem Routines - EISPACK
C Guide, Lecture Notes in Computer Science 6, Springer-Verlag,
C 1976, pp. 468-474.
C
C--------------------------------------------------------------
        SUBROUTINE TQL2(NM,N,D,E,Z,IERR)
        REAL    D(N), E(N), Z(NM,N)
        DATA    EPS/1.E-12/
C
        IERR = 0
        IF (N.EQ.1) GOTO 1001
        DO 100 I = 2, N
           E(I-1) = E(I)
  100   CONTINUE
        F = 0.0
        B = 0.0
        E(N) = 0.0
C
        DO 240 L = 1, N
           J = 0
           H = EPS * (ABS(D(L)) + ABS(E(L)))
           IF (B.LT.H) B = H
C          Look for small sub-diagonal element.
           DO 110 M = L, N
              IF (ABS(E(M)).LE.B) GOTO 120
C             E(N) is always 0, so there is no exit through
C             the bottom of the loop.
  110      CONTINUE
  120      IF (M.EQ.L) GOTO 220
  130      IF (J.EQ.30) GOTO 1000
           J = J + 1
C          Form shift.
           L1 = L + 1
           G = D(L)
           P = (D(L1)-G)/(2.0*E(L))
           R = SQRT(P*P+1.0)
           D(L) = E(L)/(P+SIGN(R,P))
           H = G-D(L)
C
           DO 140 I = L1, N
              D(I) = D(I) - H
  140      CONTINUE
C
           F = F + H
C          QL transformation.
           P = D(M)
           C = 1.0
           S = 0.0
           MML = M - L
C
           DO 200 II = 1, MML
              I = M - II
              G = C * E(I)
              H = C * P
              IF (ABS(P).LT.ABS(E(I))) GOTO 150
              C = E(I)/P
              R = SQRT(C*C+1.0)
              E(I+1) = S * P * R
              S = C/R
              C = 1.0/R
              GOTO 160
  150         C = P/E(I)
              R = SQRT(C*C+1.0)
              E(I+1) = S * E(I) * R
              S = 1.0/R
              C = C * S
  160         P = C * D(I) - S * G
              D(I+1) = H + S * (C * G + S * D(I))
C             Form vector.
              DO 180 K = 1, N
                 H = Z(K,I+1)
                 Z(K,I+1) = S * Z(K,I) + C * H
                 Z(K,I) = C * Z(K,I) - S * H
  180         CONTINUE
  200      CONTINUE
           E(L) = S * P
           D(L) = C * P
           IF (ABS(E(L)).GT.B) GOTO 130
  220      D(L) = D(L) + F
  240   CONTINUE
C
C       Order eigenvectors and eigenvalues.
        DO 300 II = 2, N
           I = II - 1
           K = I
           P = D(I)
           DO 260 J = II, N
              IF (D(J).GE.P) GOTO 260
              K = J
              P = D(J)
  260      CONTINUE
           IF (K.EQ.I) GOTO 300
           D(K) = D(I)
           D(I) = P
           DO 280 J = 1, N
              P = Z(J,I)
              Z(J,I) = Z(J,K)
              Z(J,K) = P
  280      CONTINUE
  300   CONTINUE
C
        GOTO 1001
C       Set error - no convergence after 30 iterns.
 1000   IERR = L
 1001   RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output array.
C
C---------------------------------------------------------
        SUBROUTINE OUTMAT(N,M,ARRAY)
        DIMENSION ARRAY(N,M)
C
        DO 100 K1 = 1, N
           WRITE (6,1000) (ARRAY(K1,K2),K2=1,M)
  100   CONTINUE
C
 1000   FORMAT(10(2X,F8.4))
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output half of (symmetric) array.
C
C----------------------------------------------------------
        SUBROUTINE OUTHMT(ITYPE,NDIM,ARRAY)
        DIMENSION ARRAY(NDIM,NDIM)
C
        IF (ITYPE.EQ.1) WRITE (6,1000)
        IF (ITYPE.EQ.2) WRITE (6,2000)
        IF (ITYPE.EQ.3) WRITE (6,3000)
C
        DO 100 K1 = 1, NDIM
           WRITE (6,4000) (ARRAY(K1,K2),K2=1,K1)
  100   CONTINUE
C
 1000   FORMAT
     X  (1H0,'SUMS OF SQUARES & CROSS-PRODUCTS MATRIX FOLLOWS.',/)
 2000   FORMAT(1H0,'COVARIANCE MATRIX FOLLOWS.',/)
 3000   FORMAT(1H0,'CORRELATION MATRIX FOLLOWS.',/)
 4000   FORMAT(8(2X,F8.4))
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output eigenvalues in order of decreasing value.
C
C-----------------------------------------------------------
        SUBROUTINE OUTEVL(N,NVALS,VALS)
        DIMENSION       VALS(NVALS)
C
```

```
        TOT = 0.0
        DO 100 K = 1, NVALS
           TOT = TOT + VALS(K)
  100   CONTINUE
C
        WRITE (6,1000)
        CUM = 0.0
        K = NVALS + 1
C
        M = NVALS
C       (We only want Min(nrows,ncols) eigenvalues output:)
        M = MIN0(N,NVALS)
C
        WRITE (6,1010)
        WRITE (6,1020)
  200   CONTINUE
        K = K - 1
        CUM = CUM + VALS(K)
        VPC = VALS(K) * 100.0 / TOT
        VCPC = CUM * 100.0 / TOT
        WRITE (6,1030) VALS(K),VPC,VCPC
        VALS(K) = VCPC
        IF (K.GT.NVALS-M+1) GOTO 200
C
        RETURN
 1000   FORMAT(' EIGENVALUES FOLLOW.')
 1010   FORMAT
     X(' Eigenvalues        As Percentages    Cumul. Percentages')
 1020   FORMAT
     X(' -----------        --------------    ------------------')
 1030   FORMAT(F13.4,7X,F10.4,10X,F10.4)
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C         Output FIRST SEVEN eigenvectors associated with
C         eigenvalues in descending order.
C
C------------------------------------------------------------
        SUBROUTINE OUTEVC(N,NDIM,VECS)
        DIMENSION       VECS(NDIM,NDIM)
C
        NUM = MIN0(N,NDIM,7)
C
        WRITE (6,1000)
        WRITE (6,1010)
        WRITE (6,1020)
        DO 100 K1 = 1, NDIM
        WRITE (6,1030) K1,(VECS(K1,NDIM-K2+1),K2=1,NUM)
  100   CONTINUE
C
        RETURN
 1000   FORMAT(1H0,'EIGENVECTORS FOLLOW.',/)
 1010   FORMAT
     X  ('  VBLE.   EV-1    EV-2    EV-3    EV-4    EV-5    EV-6
     X   EV-7')
 1020   FORMAT
     X  (' ------  ------  ------  ------  ------  ------  ------
     X------')
 1030   FORMAT(I5,2X,7F8.4)
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
```

```
C  Output projections of row-points on first 7 principal components.
C
C-------------------------------------------------------------
        SUBROUTINE OUTPRX(N,M,PRJN)
        REAL    PRJN(N,M)
C
        NUM = MIN0(M,7)
        WRITE (6,1000)
        WRITE (6,1010)
        WRITE (6,1020)
        DO 100 K = 1, N
           WRITE (6,1030) K,(PRJN(K,J),J=1,NUM)
  100   CONTINUE
C
 1000   FORMAT(1H0,'PROJECTIONS OF ROW-POINTS FOLLOW.',/)
 1010   FORMAT
     X  (' OBJECT  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5  PROJ-6
     X  PROJ-7')
 1020   FORMAT
     X  (' ------  ------  ------  ------  ------  ------  ------
     X  ------')
 1030   FORMAT(I5,2X,7F8.4)
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output projections of columns on first 7 principal components.
C
C-------------------------------------------------------------
        SUBROUTINE OUTPRY(M,PRJNS)
        REAL    PRJNS(M,M)
C
        NUM = MIN0(M,7)
        WRITE (6,1000)
        WRITE (6,1010)
        WRITE (6,1020)
        DO 100 K = 1, M
           WRITE (6,1030) K,(PRJNS(K,J),J=1,NUM)
  100   CONTINUE
C
 1000   FORMAT(1H0,'PROJECTIONS OF COLUMN-POINTS FOLLOW.',/)
 1010   FORMAT
     X  ('  VBLE.  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5  PROJ-6
     X  PROJ-7')
 1020   FORMAT
     X  (' ------  ------  ------  ------  ------  ------  ------
     X  ------')
 1030   FORMAT(I5,2X,7F8.4)
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Form projections of row-points on first 7 principal components.
C
C--------------------------------------------------------------
        SUBROUTINE PROJX(N,M,DATA,EVEC,VEC)
        REAL    DATA(N,M), EVEC(M,M), VEC(M)
C
        NUM = MIN0(M,7)
        DO 300 K = 1, N
           DO 50 L = 1, M
              VEC(L) = DATA(K,L)
```

```
   50      CONTINUE
           DO 200 I = 1, NUM
              DATA(K,I) = 0.0
              DO 100 J = 1, M
                 DATA(K,I) = DATA(K,I) + VEC(J) *
     X                                   EVEC(J,M-I+1)
  100         CONTINUE
  200      CONTINUE
  300   CONTINUE
C
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Determine projections of column-points on 7 prin. components.
C
C--------------------------------------------------------------
        SUBROUTINE PROJY(M,EVALS,A,Z,VEC)
        REAL    EVALS(M), A(M,M), Z(M,M), VEC(M)
C
        NUM = MIN0(M,7)
        DO 300 J1 = 1, M
           DO 50 L = 1, M
              VEC(L) = A(J1,L)
   50      CONTINUE
           DO 200 J2 = 1, NUM
              A(J1,J2) = 0.0
              DO 100 J3 = 1, M
                 A(J1,J2) = A(J1,J2) + VEC(J3)*Z(J3,M-J2+1)
  100         CONTINUE
              IF (EVALS(M-J2+1).GT.0.00005) A(J1,J2) =
     X                       A(J1,J2)/SQRT(EVALS(M-J2+1))
              IF (EVALS(M-J2+1).LE.0.00005) A(J1,J2) = 0.0
  200      CONTINUE
  300   CONTINUE
C
        RETURN
        END
```

2.4.3 Input Data

The following is the input data set used (Adorf, 1986). It represents a set of representative spectral intensity values versus wavelength for 18 main sequence stars. The reference spectra are of type O, B3, B5, B8, A0, A5, F0, F5, G0, G5, K0, K3, K5, K7, M0, M2, M4, and M5. More intensity values were initially present for each star, but in order to arrive at features of relevance the values at the beginning, end and middle of the wavelength range 350–530 *nm* were taken. The subsets of the original spectra, thus defined, encompassed the essential characteristics of downward sloping spectra being associated with O and B stars, and generally upward sloping spectra associated with K and M stars. The data used here have in total 16 intensity values, for each star.

```
INPUT DATA SET: STARS

 Seq.no.   Col.1           Col.2           Col.3           Col.4

      1   3.00000         3.00000         3.00000         3.00000
      2   3.50000         3.50000         4.00000         4.00000
      3   4.00000         4.00000         4.50000         4.50000
      4   5.00000         5.00000         5.00000         5.50000
      5   6.00000         6.00000         6.00000         6.00000
      6   11.0000         11.0000         11.0000         11.0000
      7   20.0000         20.0000         20.0000         20.0000
      8   30.0000         30.0000         30.0000         30.0000
      9   30.0000         33.4000         36.8000         40.0000
     10   42.0000         44.0000         46.0000         48.0000
     11   60.0000         61.7000         63.5000         65.5000
     12   70.0000         70.1000         70.2000         70.3000
     13   78.0000         77.6000         77.2000         76.8000
     14   98.9000         97.8000         96.7000         95.5000
     15   160.000         157.000         155.000         152.000
     16   272.000         266.000         260.000         254.000
     17   382.000         372.000         362.000         352.000
     18   770.000         740.000         710.000         680.000

 Seq.no.   Col.5           Col.6           Col.7           Col.8

      1   3.00000         3.00000         35.0000         45.0000
      2   4.50000         4.50000         46.0000         59.0000
      3   5.00000         5.00000         48.0000         60.0000
      4   5.50000         5.50000         46.0000         63.0000
      5   6.50000         6.50000         51.0000         69.0000
      6   11.0000         11.0000         64.0000         75.0000
      7   20.0000         20.0000         76.0000         86.0000
      8   30.1000         30.2000         84.0000         96.0000
      9   43.0000         45.6000         100.000         106.000
     10   50.0000         51.0000         109.000         111.000
     11   67.3000         69.2000         122.000         124.000
     12   70.4000         70.5000         137.000         132.000
     13   76.4000         76.0000         167.000         159.000
     14   94.3000         93.2000         183.000         172.000
     15   149.000         147.000         186.000         175.000
     16   248.000         242.000         192.000         182.000
```

```
    17    343.000         333.000         205.000         192.000
    18    650.000         618.000         226.000         207.000

 Seq.no.   Col.9           Col.10          Col.11          Col.12

     1    53.0000         55.0000         58.0000         113.000
     2    63.0000         58.0000         58.0000         125.000
     3    68.0000         65.0000         65.0000         123.000
     4    70.0000         64.0000         63.0000         116.000
     5    77.0000         70.0000         71.0000         120.000
     6    81.0000         79.0000         79.0000         112.000
     7    93.0000         92.0000         91.0000         104.000
     8    98.0000         99.0000         96.0000         101.000
     9    106.000         108.000         101.000         99.0000
    10    110.000         110.000         103.000         95.5000
    11    124.000         121.000         103.000         93.2000
    12    134.000         128.000         101.000         91.7000
    13    152.000         144.000         103.000         89.8000
    14    162.000         152.000         102.000         87.5000
    15    165.000         156.000         120.000         87.0000
    16    170.000         159.000         131.000         88.0000
    17    178.000         166.000         138.000         86.2000
    18    195.000         180.000         160.000         82.9000

 Seq.no.   Col.13          Col.14          Col.15          Col.16

     1    113.000         86.0000         67.0000         90.0000
     2    126.000         110.000         78.0000         97.0000
     3    123.000         117.000         87.0000         108.000
     4    119.000         115.000         97.0000         112.000
     5    122.000         122.000         96.0000         123.000
     6    114.000         113.000         98.0000         115.000
     7    104.500         107.000         97.5000         104.000
     8    102.000         99.0000         94.0000         99.0000
     9    98.0000         99.0000         95.0000         95.0000
    10    95.5000         95.0000         92.5000         92.0000
    11    92.5000         92.2000         90.0000         90.8000
    12    90.2000         88.8000         87.3000         85.8000
    13    87.7000         85.7000         83.7000         81.8000
    14    85.3000         83.3000         81.3000         79.3000
    15    84.9000         82.8000         80.8000         79.0000
    16    85.8000         83.7000         81.6000         79.6000
    17    84.0000         82.0000         79.8000         77.5000
    18    80.2000         77.7000         75.2000         72.7000
```

2.4.4 Sample Output

The following is the output which is produced. The "linearity" of the data may be noted. The eigenvalues (percentages of variance explained by axes) are followed by the definition of the eigenvectors (principal components) in the parameter–space. Then the projections of the objects (rows) and of the parameters (columns) on the new principal components in the respective spaces are listed.

```
 EIGENVALUES FOLLOW.

 Eigenvalues        As Percentages    Cumul. Percentages
 -----------        --------------    ------------------
      12.7566          79.7285             79.7285
       1.8521          11.5756             91.3041
       1.1516           7.1975             98.5016
       0.1762           1.1009             99.6026
       0.0356           0.2226             99.8251
       0.0225           0.1406             99.9658
       0.0035           0.0219             99.9877
       0.0007           0.0044             99.9921
       0.0005           0.0033             99.9955
       0.0004           0.0027             99.9982
       0.0002           0.0015             99.9997
       0.0001           0.0003            100.0000
       0.0000           0.0000            100.0000
       0.0000           0.0000            100.0000
       0.0000           0.0000            100.0000
       0.0000           0.0000            100.0000
```

```
EIGENVECTORS FOLLOW.

 VBLE.   EV-1    EV-2    EV-3    EV-4    EV-5    EV-6    EV-7
------  ------  ------  ------  ------  ------  ------  ------
   1    0.2525 -0.3156  0.0281  0.0621  0.0475  0.1261 -0.0736
   2    0.2534 -0.3105  0.0294  0.0623  0.0362  0.1088 -0.0493
   3    0.2544 -0.3049  0.0305  0.0609  0.0215  0.0877 -0.0269
   4    0.2555 -0.2990  0.0321  0.0613  0.0087  0.0700  0.0050
   5    0.2565 -0.2927  0.0336  0.0594 -0.0065  0.0441  0.0374
   6    0.2578 -0.2847  0.0350  0.0558 -0.0189  0.0144  0.0809
   7    0.2678  0.1684  0.0717 -0.3855  0.0350  0.0712  0.1288
   8    0.2665  0.1671  0.1065 -0.4025  0.0352  0.0189  0.4010
   9    0.2663  0.1668  0.1267 -0.3752  0.1033  0.0802 -0.0830
  10    0.2649  0.2000  0.1322 -0.2432  0.0217 -0.0583 -0.2050
  11    0.2673  0.0493  0.2252  0.1838 -0.3041 -0.8467  0.0563
  12   -0.2457 -0.3228 -0.1133 -0.3296 -0.2382 -0.1378  0.1235
  13   -0.2489 -0.3182 -0.0784 -0.2706 -0.1544 -0.0606  0.5661
  14   -0.2365 -0.2575  0.3337 -0.4175 -0.3199 -0.0074 -0.5839
  15   -0.1354  0.0883  0.7990  0.2634 -0.2190  0.3249  0.2778
  16   -0.2407 -0.2209  0.3530 -0.0914  0.8115 -0.3116  0.0123
```

```
PROJECTIONS OF ROW-POINTS FOLLOW.

OBJECT  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5  PROJ-6  PROJ-7
------  ------  ------  ------  ------  ------  ------  ------
   1   -0.6334 -0.1040 -0.7981  0.1659  0.0722 -0.0174  0.0010
   2   -0.8520 -0.3140 -0.3998 -0.1283 -0.1015  0.0073  0.0096
   3   -0.8994 -0.3214 -0.0674 -0.1203 -0.0423 -0.0136 -0.0257
   4   -0.8988 -0.2452  0.1791  0.0087  0.0086  0.0907  0.0242
   5   -0.9446 -0.3344  0.2805 -0.1295  0.0877 -0.0433 -0.0091
   6   -0.7488 -0.1569  0.2835  0.0051  0.0547 -0.0080  0.0150
   7   -0.4970  0.0263  0.2529  0.0898 -0.0213 -0.0022 -0.0215
   8   -0.3373  0.1036  0.1385  0.1198 -0.0116 -0.0237  0.0167
   9   -0.2170  0.1691  0.1792  0.1023 -0.0577 -0.0124  0.0002
  10   -0.1077  0.2188  0.1092  0.1223 -0.0405 -0.0122  0.0004
  11    0.0436  0.2570  0.0727  0.0736  0.0053 -0.0017 -0.0114
  12    0.1606  0.3222 -0.0115  0.0371  0.0000  0.0322 -0.0155
  13    0.3541  0.4244 -0.0727 -0.0862  0.0142  0.0312 -0.0009
  14    0.4994  0.4539 -0.1191 -0.1304  0.0388  0.0573 -0.0051
  15    0.6883  0.3365 -0.0761 -0.0903  0.0066 -0.0375 -0.0005
  16    0.9433  0.0842  0.0061 -0.0685 -0.0123 -0.0485  0.0190
  17    1.2569 -0.1027  0.0096 -0.0507 -0.0062 -0.0298  0.0150
  18    2.1896 -0.8174  0.0333  0.0796  0.0053  0.0314 -0.0117




PROJECTIONS OF COLUMN-POINTS FOLLOW.

 VBLE.  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5  PROJ-6  PROJ-7
------  ------  ------  ------  ------  ------  ------  ------
    1    0.3607 -0.0612  0.0033  0.0011  0.0002  0.0003  0.0000
    2    0.3621 -0.0602  0.0034  0.0011  0.0001  0.0002  0.0000
    3    0.3635 -0.0591  0.0035  0.0011  0.0001  0.0002  0.0000
    4    0.3650 -0.0579  0.0037  0.0011  0.0000  0.0002  0.0000
    5    0.3664 -0.0567  0.0039  0.0010  0.0000  0.0001  0.0000
    6    0.3683 -0.0552  0.0041  0.0010 -0.0001  0.0000  0.0000
    7    0.3826  0.0326  0.0083 -0.0068  0.0001  0.0002  0.0000
    8    0.3807  0.0324  0.0124 -0.0071  0.0001  0.0000  0.0001
    9    0.3804  0.0323  0.0147 -0.0066  0.0004  0.0002  0.0000
   10    0.3784  0.0388  0.0153 -0.0043  0.0001 -0.0001 -0.0001
   11    0.3819  0.0096  0.0261  0.0032 -0.0011 -0.0019  0.0000
   12   -0.3511 -0.0626 -0.0131 -0.0058 -0.0008 -0.0003  0.0000
   13   -0.3556 -0.0617 -0.0091 -0.0048 -0.0006 -0.0001  0.0002
   14   -0.3378 -0.0499  0.0387 -0.0074 -0.0011  0.0000 -0.0002
   15   -0.1934  0.0171  0.0927  0.0046 -0.0008  0.0007  0.0001
   16   -0.3439 -0.0428  0.0410 -0.0016  0.0029 -0.0007  0.0000
```

2.4.5 Java Application

For some core functionality, we use mathematical class libraries available from Visual Numerics. Therefore this must be on your system before you use the following program. Of course the Java Development Kit, JDK, must also be available. We used version 1.1.6 of the JDK.

```
import VisualNumerics.math.*;
import java.text.*;

public class PCA
{
    // This class carries out a PCA with standardization of the
    // inputs (i.e. PCA on correlations), with hardwiring of the
    // input data, and with output to standard output of results.

    // The VisualNumerics.math.* classes, in particular DoubleMatrix
    // and DoubleSVD, are used.

    // Reference: for Visual Numerics classes, http://www.vni.com
    // For everything else, Chapter 8 of Richard Davies, Introductory
    // Java for Scientists and Engineers, Addison-Wesley, 1999.
    // Coded by F. Murtagh, July 1999.

    // Methods used in this class:
    // RowProj, for determining row projections
    // ColProj, for determining column projections
    // printMatrix, for printing a matrix
    // printVect, for printing a vector
    // DoubleVector.sum, sum of vectors
    // DoubleVector.multiply, multiply vector by scalar
    // DoubleMatrix.multiply, matrix multiplication
    // DoubleMatrix.transpose, matrix transpose
    // system.out.print
    // system.out.println


    //-------------------------------------------------------------------
    // Method for standardizing the input data
    public static double[][] Standardize(int nrow, int ncol, double[][] A)
    {
    double[] colmeans = new double[ncol];
    double[] colstdevs = new double[ncol];
    // Adat will contain the standardized data and will be returned
    double[][] Adat = new double[nrow][ncol];
    double[] tempcol = new double[nrow];

    // Determine means and standard deviations of variables/columns
    for (int j=0; j<ncol; j++)
        {
         for (int i=0; i<nrow; i++)
             {
                tempcol[i] = A[i][j];
             }
         colmeans[j] = Statistics.average(tempcol);
 // Sample stddev = sqrt((\sum_i(x_i - \bar{x})^2)/(n-1))
         colstdevs[j] = Statistics.standardDeviation(tempcol);
        }
        System.out.println("Variable means:");
        printVect(colmeans);
        System.out.println("Variable standard deviations:");
        printVect(colstdevs);

// Now ceter to zero mean, and reduce to unit standard deviation
    for (int j=0; j<ncol; j++)
        {
         for (int i=0; i<nrow; i++)
             {
               Adat[i][j] = (A[i][j] - colmeans[j])/colstdevs[j];
             }
        }
    return Adat;
    }
    //-------------------------------------------------------------------

    //-------------------------------------------------------------------
    // Method for determining row projections
    // From SVD decomposition, here we want product: A U
    public static double[][] RowProj(double[][] evecs, double[][] dat)
    {
    double[][] rproj = DoubleMatrix.multiply(dat, evecs);
    return rproj;
    }
    //-------------------------------------------------------------------
    //-------------------------------------------------------------------
    // Method for determining column projections
    // From SVD decomposition, here we want: Corr U with colwise div by
    // sqrt(lambda)
    public static double[][] ColProj(int n1, int n2,
                                     double[][] evecs,
                                     double[][] cdat,
                                     double[] evals)
    {
    double[][] cproj = DoubleMatrix.multiply(cdat,evecs);

    // Rescale by eigenvalues
    for (int j1=0; j1<n2; j1++)
{
        for (int j2=0; j2<n2; j2++)
            {
               cproj[j1][j2] = cproj[j1][j2]/Math.sqrt(evals[j2]);
            }
}

    return cproj;

    }
    //-------------------------------------------------------------------

    // Little method for helping in output formating
    public static String getSpaces(int n) {

    StringBuffer sb = new StringBuffer(n);
    for (int i = 0; i < n; i++) sb.append(' ');
    return sb.toString();
    }


//---------------------------------------------------------------------
    // Utility for printing a matrix
    public static void printMatrix(int n1, int n2, double[][] m)
    {
// Some definitions for handling output formating
      NumberFormat myFormat = NumberFormat.getNumberInstance();
      FieldPosition fp = new FieldPosition(NumberFormat.INTEGER_FIELD);
      myFormat.setMaximumIntegerDigits(4);
      myFormat.setMaximumFractionDigits(4);
      myFormat.setMinimumFractionDigits(4);
      for (int i=0; i<n1; i++)
  {
      // Print each row, elements separated by spaces
              for (int j=0; j<n2; j++)
  // Following unfortunately doesn't format at all
  //                  System.out.print(m[i][j] + "  ");
                  {
                  String valString = myFormat.format(
                       m[i][j], new StringBuffer(), fp).toString();
                  valString = getSpaces(4 - fp.getEndIndex()) + valString;
                  System.out.print(valString);
                  }
              // Start a new line at the end of a row
              System.out.println();
          }
      // Leave a gap after the entire matrix
      System.out.println();
    }
    //--------------------------------------------------------------------

    //--------------------------------------------------------------------
    // Utility for printing a vector
    public static void printVect(double[] m)
    {
// Some definitions for handling output formating
      NumberFormat myFormat = NumberFormat.getNumberInstance();
      FieldPosition fp = new FieldPosition(NumberFormat.INTEGER_FIELD);
      myFormat.setMaximumIntegerDigits(4);
      myFormat.setMaximumFractionDigits(4);
      myFormat.setMinimumFractionDigits(4);
      int len = m.length;
      for (int i=0; i<len; i++)
  {
      // Following would be nice, but doesn't format adequately
      //                  System.out.print(m[i] + "  ");
             String valString = myFormat.format(
                   m[i], new StringBuffer(), fp).toString();
             valString = getSpaces(3 - fp.getEndIndex()) + valString;
                   System.out.print(valString);
          }
          // Start a new line at the row end
          System.out.println();
      // Leave a gap after the entire vector
      System.out.println();
    }
    //--------------------------------------------------------------------


    //--------------------------------------------------------------------
    // The main method contains the body of the program
    public static void main(String[] argv)
    {

        // Define dimensions of the matrix we'll use
        int nrow = 18;
        int ncol = 16;
// Define the matrix that we are going to use
        double[][] A = {
   {   3.0,   3.0,   3.0,   3.0,   3.0,   3.0,  35.0,  45.0,
      53.0,  55.0,  58.0, 113.0, 113.0,  86.0,  67.0,  90.0},
   {   3.5,   3.5,   4.0,   4.0,   4.5,   4.5,  46.0,  59.0,
      63.0,  58.0,  58.0, 125.0, 126.0, 110.0,  78.0,  97.0},
   {   4.0,   4.0,   4.5,   4.5,   5.0,   5.0,  48.0,  60.0,
      68.0,  65.0,  65.0, 123.0, 123.0, 117.0,  87.0, 108.0},
   {   5.0,   5.0,   5.0,   5.5,   5.5,   5.5,  46.0,  63.0,
      70.0,  64.0,  63.0, 116.0, 119.0, 115.0,  97.0, 112.0},
   {   6.0,   6.0,   6.0,   6.0,   6.5,   6.5,  51.0,  69.0,
      77.0,  70.0,  71.0, 120.0, 122.0, 122.0,  96.0, 123.0},
   {  11.0,  11.0,  11.0,  11.0,  11.0,  11.0,  64.0,  75.0,
      81.0,  79.0,  79.0, 112.0, 114.0, 113.0,  98.0, 115.0},
   {  20.0,  20.0,  20.0,  20.0,  20.0,  20.0,  76.0,  86.0,
      93.0,  92.0,  91.0, 104.0, 104.5, 107.0,  97.5, 104.0},
   {  30.0,  30.0,  30.0,  30.0,  30.1,  30.2,  84.0,  96.0,
      98.0,  99.0,  96.0, 101.0, 102.0,  99.0,  94.0,  99.0},
   {  30.0,  33.4,  36.8,  40.0,  43.0,  45.6, 100.0, 106.0,
     106.0, 108.0, 101.0,  99.0,  98.0,  99.0,  95.0,  95.0},
   {  42.0,  44.0,  46.0,  48.0,  50.0,  51.0, 109.0, 111.0,
     110.0, 110.0, 103.0,  95.5,  95.5,  95.0,  92.5,  92.0},
   {  60.0,  61.7,  63.5,  65.5,  67.3,  69.2, 122.0, 124.0,
     124.0, 121.0, 103.0,  93.2,  92.5,  92.2,  90.0,  90.8},
   {  70.0,  70.1,  70.2,  70.3,  70.4,  70.5, 137.0, 132.0,
     134.0, 128.0, 101.0,  91.7,  90.2,  88.8,  87.3,  85.8},
   {  78.0,  77.6,  77.2,  76.8,  76.4,  76.0, 167.0, 159.0,
     152.0, 144.0, 103.0,  89.8,  87.7,  85.7,  83.7,  81.8},
   {  98.9,  97.8,  96.7,  95.5,  94.3,  93.2, 183.0, 172.0,
     162.0, 152.0, 102.0,  87.5,  85.3,  83.3,  81.3,  79.3},
   { 160.0, 157.0, 155.0, 152.0, 149.0, 147.0, 186.0, 175.0,
     165.0, 156.0, 120.0,  87.0,  84.9,  82.8,  80.8,  79.0},
   { 272.0, 266.0, 260.0, 254.0, 248.0, 242.0, 192.0, 182.0,
     170.0, 159.0, 131.0,  88.0,  85.8,  83.7,  81.6,  79.6},
   { 382.0, 372.0, 362.0, 352.0, 343.0, 333.0, 205.0, 192.0,
     178.0, 166.0, 138.0,  86.2,  84.0,  82.0,  79.8,  77.5},
   { 770.0, 740.0, 710.0, 680.0, 650.0, 618.0, 226.0, 207.0,
     195.0, 180.0, 160.0,  82.9,  80.2,  77.7,  75.2,  72.7}
        };

        // Print it out
        System.out.println("A is our input matrix:");
        printMatrix(nrow, ncol, A);

        // Standardize the input matrix
        double[][] Adat = Standardize(nrow, ncol, A);
        // Print it out
        System.out.println("Standardized matrix for analysis:");
        printMatrix(nrow, ncol, Adat);

        // Determine correlations
        double[][] Corr = DoubleMatrix.multiply(
                          DoubleMatrix.transpose(Adat), Adat);

        // Perform the SVD
        DoubleSVD svdres = new DoubleSVD(Corr);

        System.out.println("Eigenvalues:");
        printVect(svdres.S());

        // Sum of eigenvalues, then percentage variances
        // We use methods sum and multiply from class DoubleVector from JNL
        double tot = DoubleVector.sum(svdres.S());
        tot = 1.0/tot;
        double[] percentvar = DoubleVector.multiply(tot, svdres.S());
        percentvar = DoubleVector.multiply(100.0, percentvar);

        System.out.println("Percentage variances:");
        printVect(percentvar);
        System.out.println();

        System.out.println("Eigenvectors:");
        System.out.println("Number of variables (rows) x Dimensionality (cols)");
        System.out.println("Columns = definitions of new axes in terms of old variables");
        System.out.println("Col 1 = new axis 1, etc.");
        printMatrix(ncol, ncol, svdres.U());

// Right singular vectors are the same in this case
// System.out.println("Right eigenvectors, column principal components:");
// printMatrix(ncol, ncol, svdres.V());

        System.out.println("Row projections:");
        System.out.println("Rows: obj1, obj2, ..., objn");
        System.out.println("Columns: proj1, proj2, ...,  projm");
        double[][] rproj = RowProj(svdres.U(), Adat);
        printMatrix(nrow, ncol, rproj);

        System.out.println("Column projections:");
        System.out.println("Rows on output: (Old)Vbe1, (Old)Vbe2,..., (Old)Vbem");
        System.out.println("Columns on output: proj1, proj2, ..., projm");
        double[][] cproj = ColProj(nrow, ncol, svdres.V(), Corr, svdres.S());
        printMatrix(ncol, ncol, cproj);

// Some remarks on comparison of results with MDA, F Murtagh and
// A Heck, Multivariate Data Analysis, Kluwer, 1987, Chapter 2.

// Due to eigenvalues being different, we have projections
        // which are rescaled compared to MDA.
//
        // In MDA, std. dev. is calculated as
// sqrt{ \sum_i (x_i - \bar{x})^2 }    (Cf. p. 37)
// In our program, std. dev. is
        // sqrt(\frac{\sum_i (x_i - \bar{x})^2}{n-1}),
// sample standard deviation.
//
// Implications:
// Eigenvectors are *exactly* the same.
// Standardized array is within a constant (but n-related) factor.
// Row projections are therefore also to within this factor.
// For the spectral dataset given in MDA, here these values are
// all about 4.1 times greater (i.e. sqrt(n-1) = sqrt(17)).
// Eigenvalues are different in value, though percentages are ident.
// Eigenvalues for spect.dat:
// 216.86, 31.48, 19.57, etc. here.
// 12.75, 1.85, 1.15, etc. in MDA case.
// Percentages are identical as mentioned

    }
    //------------------------------------------------------------------
}
```

Byte code is created for this program, `PCA.java`, by the command: `javac PCA.java`. The program is then run as follows: `java PCA`.

Chapter 3

Cluster Analysis

3.1 The Problem

Automatic classification algorithms are used in widely different fields in order to provide a description or a reduction of data. A clustering algorithm is used to determine the inherent or natural groupings in the data, or provide a convenient summarization of the data into groups. Although the term "classification" is often applied both to this area and to Discriminant Analysis, this chapter will be solely concerned with unsupervised clustering, with no prior knowledge on the part of the analyst regarding group memberships.

As is the case with Principal Components Analysis, and with most other multivariate techniques, the objects to be classified have numerical measurements on a set of variables or attributes. Hence, the analysis is carried out on the rows of an array or matrix. If we have not a matrix of numerical values, to begin with, then it may be necessary to skilfully construct such a matrix. Chapter 1, in particular, can be referred to here for difficulties arising when missing data or qualitative attributes are present. The objects, or rows of the matrix, can be viewed as vectors in a multidimensional space (the dimensionality of this space being the number of variables or columns). A geometric framework of this type is not the only one which can be used to formulate clustering algorithms; it is the preferred one in this chapter because of its generality and flexibility.

Needless to say, a clustering analysis can just as easily be implemented on the columns of a data matrix. There is usually no direct relationship between the clustering of the rows and clustering of the columns, as was the case for the dual spaces in Principal Components Analysis. It may also be remarked that suitable alternative forms of storage of a rectangular array of values are not inconsistent with viewing the problem in geometric (and in matrix) terms: in the case of large sparse matrices, for instance, suitable storage schemes require consideration in practice.

Motivation for clustering may be categorized under the following headings, the first two of which will be of principal interest in this chapter:

1. Analysis of data: here, the given data is to be analyzed in order to reveal its fundamental features. The significant interrelationships present in the data are sought. This is the multivariate statistical use of clustering, and

the validity problem (i.e. the validation of clusters of data items produced by an algorithm) is widely seen as a major current difficulty.

2. User convenience: a synoptic classification is to be obtained which will present a useful decomposition of the data. This may be a first step towards subsequent analysis where the emphasis is on heuristics for summarizing information. Appraisal of clustering algorithms used for this purpose can include algorithmic efficiency and ease of use.

3. Storage and retrieval: we are here concerned with improving access speeds by providing better routing strategies to stored information. The effectiveness of the clustering is measured by time and space efficiency, and by external criteria (related, for example, to the amount of relevant material retrieved).

4. Machine vision: the distinguishing of point patterns, or the processing of digital image data, is often assessed visually, and computational efficiency is highly important also.

Applications of clustering embrace many diverse fields (establishing taxonomies in biology and ecology; efficient retrieval algorithms in computer and information science; grouping of test subjects and of the test items in psychology and educational research; and so on). The range of algorithms which have been proposed (for the most part since the early 1960s with the advent of computing power on a wide scale) has been correspondingly large.

3.2 Mathematical Description

3.2.1 Introduction

Most published work in Cluster Analysis involves the use of either of two classes of clustering algorithm: hierarchical or non–hierarchical (often partitioning) algorithms. Hierarchical algorithms in particular have been dominant in the literature and so this chapter concentrates on these methods (Sections 3.2.2, 3.2.3). Each of the many clustering methods — and of the many hierarchical methods — which have been proposed over the last two decades have possibly advantageous properties. Many textbooks catalogue these methods, which makes the task facing the practitioner of choosing the right method an onerous one. We feel that it is more helpful instead to study a method which is recommendable for general purpose applications. This we do with the minimum variance method.

Sections 3.2.4 and 3.2.5 will focus on Ward's minimum variance method. For most applications this method can be usefully employed for the summarization of data. Section 3.2.4 will attempt to justify this statement: it will informally describe the minimum variance agglomerative method, and will look at properties of this method which are of practical importance. Mathematical properties of the minimum variance method are then detailed in Section 3.2.5.

Finally, non–hierarchical routines have also been widely implemented. Section 3.2.6 looks at the minimal spanning tree (closely related to the single linkage hierarchical method). Finally, Section 3.2.7 briefly describes partitioning methods.

3.2.2 Hierarchical Methods

The single linkage hierarchical clustering approach outputs a set of clusters (to use graph theoretic terminology, a set of maximal connected subgraphs) at each level — or for each threshold value which produces a new partition. The following algorithm, in its general structure, is relevant for a wide range of hierarchical clustering methods which vary only in the update formula used in Step 2. These methods may, for example, define a criterion of compactness in Step 2 to be used instead of the connectivity criterion used here. Such hierarchical methods will be studied in Section 3.2.3, but the single linkage method with which we begin is one of the oldest and most widely used methods (its usage is usually traced to the early 1950s). An example is shown in Figure 3.1 — note that the dissimilarity coefficient is assumed to be symmetric, and so the clustering algorithm is implemented on half the dissimilarity matrix.

Single linkage hierarchical clustering

Input An $n(n-1)/2$ set of dissimilarities.

Step 1 Determine the smallest dissimilarity, d_{ik}.

Step 2 Agglomerate objects i and k: i.e. replace them with a new object, $i \cup k$; update dissimilarities such that, for all objects $j \neq i, k$:

$$d_{i \cup k, j} = \min \{d_{ij}, d_{kj}\}.$$

Delete dissimilarities d_{ij} and d_{kj}, for all j, as these are no longer used.

Step 3 While at least two objects remain, return to Step 1.

Equal dissimilarities may be treated in an arbitrary order. There are precisely $n-1$ agglomerations in Step 2 (allowing for arbitrary choices in Step 1 if there are identical dissimilarities). It may be convenient to index the clusters found in Step 2 by $n+1, n+2, \ldots, 2n-1$, or an alternative practice is to index cluster $i \cup k$ by the lower of the indices of i and k.

The title *single linkage* arises since, in Step 2, the interconnecting dissimilarity between two clusters ($i \cup k$ and j) or components is defined as the least interconnecting dissimilarity between a member of one and a member of the other. Other hierarchical clustering methods are characterized by other functions of the interconnecting linkage dissimilarities.

Since there are $n-1$ agglomerations, and hence iterations, and since Step 2 requires $< n$ operations, the algorithm for the single linkage hierarchic clustering given above is of time complexity $O(n^2)$.

Compared to other hierarchic clustering techniques, the single linkage method can give rise to a notable disadvantage for summarizing interrelationships. This is known as *chaining*. An example is to consider four subject–areas, which it will be supposed are characterized by certain attributes: computer science, statistics, probability, and measure theory. It is quite conceivable that "computer science" is connected to "statistics" at some threshold value, "statistics" to "probability", and "probability" to "measure theory", thereby giving rise to

```
        1  2  3  4  5                 1   2U4   3   5
    --+--------------           ----+----------------
    1 | 0  4  9  5  8            1  | 0    4    9   8
    2 | 4  0  6  3  6           2U4 | 4    0    6   5
    3 | 9  6  0  6  3            3  | 9    6    0   3
    4 | 5  3  6  0  5            5  | 8    5    3   0
    5 | 8  6  3  5  0


 Agglomerate 2 and 4 at         Agglomerate 3 and 5 at
    dissimilarity 3                dissimilarity 3
```

```
         1   2U4   3U5                1U2U4    3U5
  ----+----------------        ------+-------------
    1 |  0    4     8          1U2U4 |  0       5
  2U4 |  4    0     5           3U5  |  5       0
  3U5 |  8    5     0

Agglomerate 1 and 2U4 at     Finally agglomerate 1U2U4
   dissimilarity 4           and 3U5 at dissimilarity 5
```

```
Resulting dendrogram
                            |
                            |
                      +----------+           ... 4 ... 5
                      |          |
                   +------+      |           ... 3 ... 4
                   |      |      |
                   |      |    +---+         ... 2 ... 3
                   |      |    |   |
                   |    +---+  |   |         ... 1 ... 3
                   |    |   |  |   |
                   |    |   |  |   |         ... 0 ... 0

                                            Ranks    Criterion
                                             or       values
                                            levels   (linkage
                                                      weights)
```

Figure 3.1: Construction of a dendrogram by the single linkage method.

the fact that “computer science” and “measure theory” find themselves, undesirably, in the same cluster. This is due to the intermediaries “statistics” and “probability”.

We will turn attention now to the general role played by a dendrogram, constructed by any criterion (Figure 3.2 illustrates differing possible representations).

About 75% of all published work on clustering has employed hierarchical algorithms (according to Blashfield and Aldenderfer, 1978): this figure for published work might or might not hold for practical clustering usage, but it is nonetheless revealing. Interpretation of the information contained in a dendrogram will often be of one or more of the following kinds:

– set inclusion relationships,

– partition of the object–sets, and

– significant clusters.

We will briefly examine what each of these entail.

Much early work on hierarchic clustering was in the field of biological taxonomy, from the 1950s and more so from the 1960s onwards. The central reference in this area, the first edition of which dates from the early 1960s, is Sneath and Sokal (1973). One major interpretation of hierarchies has been the evolution relationships between the organisms under study. It is hoped, in this context, that a dendrogram provides a sufficiently accurate model of underlying evolutionary progression. As an example, consider the hierarchical classification of galaxies based on certain features. It could be attempted to characterize clusters at higher levels of the tree as being major spiral and elliptical groups, to which other subclasses are related.

The most common interpretation made of hierarchic clustering is to derive a partition: a line is drawn horizontally through the hierarchy, to yield a set of classes. These clusters are precisely the connected components in the case of the single linkage method. A line drawn just above rank 3 (or criterion value 4) on the dendrogram in Figure 3.1 yields classes $\{1, 2, 4\}$ and $\{3, 5\}$. Generally the choice of where “to draw the line” is arrived at on the basis of large changes in the criterion value. However the changes in criterion value increase (usually) towards the final set of agglomerations, which renders the choice of best partition on this basis difficult. Since every line drawn through the dendrogram defines a partition, it may be expedient to choose a partition with convenient features (number of classes, number of objects per class).

A final type of interpretation, less common than the foregoing, is to dispense with the requirement that the classes chosen constitute a partition, and instead detect maximal (i.e. disjoint) clusters of interest at varying levels of the hierarchy. Such an approach is used by Rapoport and Fillenbaum (1972) in a clustering of colours based on semantic attributes.

In summary, a dendrogram provides a résumé of many of the proximity and classificatory relationships in a body of data. It is a convenient representation which answers such questions as: “How many groups are in this data?”, “What are the salient interrelationships present?”. But it should be stressed that differing answers can feasibly be provided by a dendrogram for most of these questions, just as different human observers would also arrive at different conclusions.

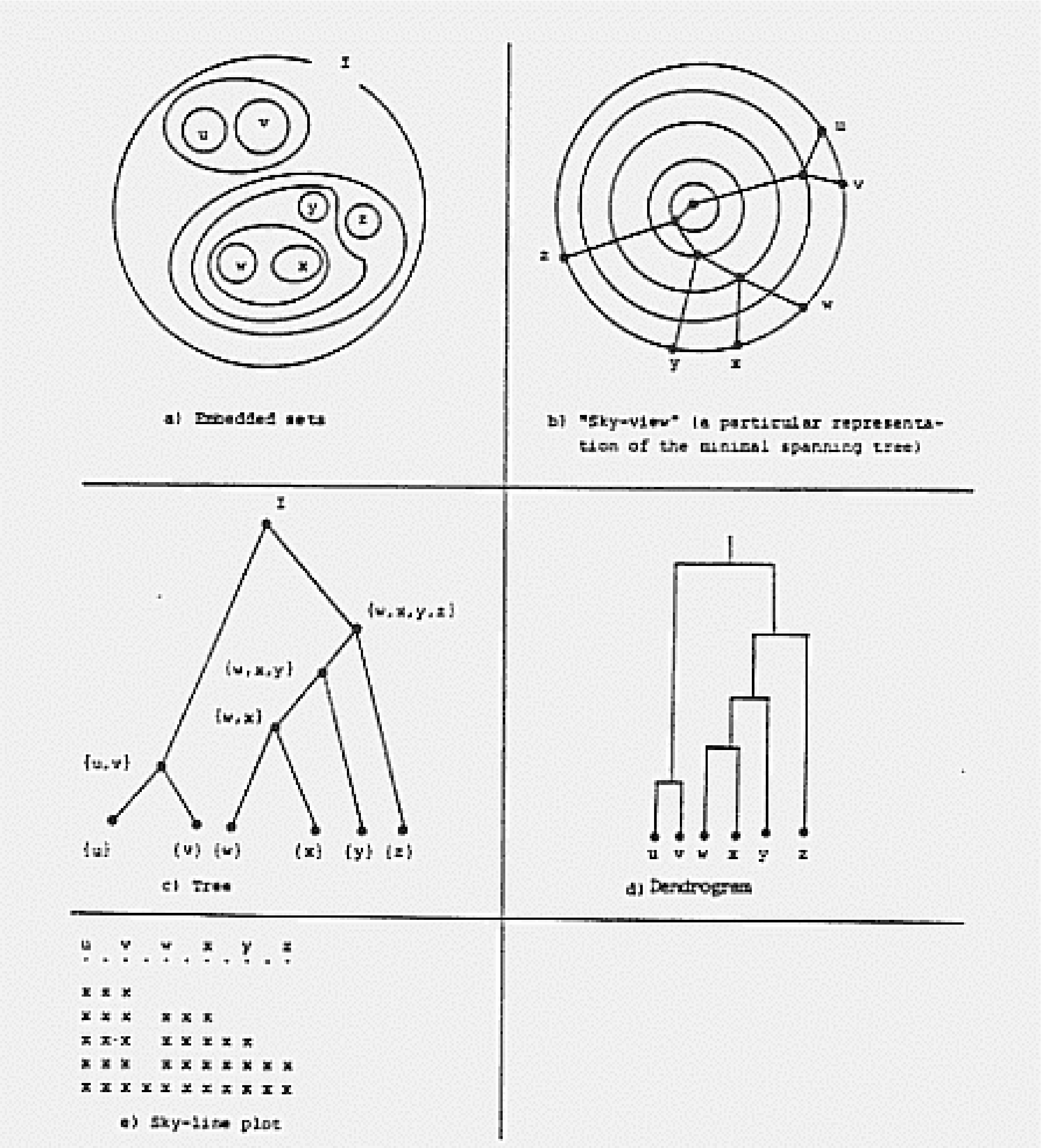


Figure 3.2: Differing representations of a hierarchic clustering on 6 objects.

3.2.3 Agglomerative Algorithms

In the last section, a general agglomerative algorithm was discussed. A wide range of these algorithms have been proposed at one time or another. Hierarchic agglomerative algorithms may be conveniently broken down into two groups of methods. The first group is that of linkage methods — the single, complete, weighted and unweighted average linkage methods. These are methods for which a graph representation can be used. Figure 3.3 shows the close relationship between the single linkage hierarchy and the minimal spanning tree; we see that the single linkage dendrogram can be constructed by first sorting the dissimilarities into ascending order; and that in some cases, e.g. in going from d_{bc} to d_{ac}, there is no change in the dendrogram and information which is irrelevant to the minimal spanning tree is ignored. Sneath and Sokal (1973) may be consulted for many other graph representations of the stages in the construction of hierarchic clusterings.

The second group of hierarchic clustering methods are methods which allow the cluster centres to be specified (as an average or a weighted average of the member vectors of the cluster). These methods include the centroid, median and minimum variance methods.

The latter may be specified either in terms of dissimilarities, alone, or alternatively in terms of cluster centre coordinates and dissimilarities. A very convenient formulation, in dissimilarity terms, which embraces all the hierarchical methods mentioned so far, is the *Lance–Williams dissimilarity update formula*. If points (objects) i and j are agglomerated into cluster $i \cup j$, then we must simply specify the new dissimilarity between the cluster and all other points (objects or clusters). The formula is:

$$d(i \cup j, k) = \alpha_i d(i,k) + \alpha_j d(j,k) + \beta d(i,j) + \gamma \mid d(i,k) - d(j,k) \mid$$

where α_i, α_j, β, and γ define the agglomerative criterion. Values of these are listed in the second column of Table 3.1. In the case of the single link method, using $\alpha_i = \alpha_j = \frac{1}{2}$, $\beta = 0$, and $\gamma = -\frac{1}{2}$ gives us

$$d(i \cup j, k) = \frac{1}{2} d(i,k) + \frac{1}{2} d(j,k) - \frac{1}{2} \mid d(i,k) - d(j,k) \mid$$

which, it may be verified by taking a few simple examples of three points, i, j, and k, can be rewritten as

$$d(i \cup j, k) = \min \, \{d(i,k), d(j,k)\}.$$

This was exactly the update formula used in the agglomerative algorithm given in the previous Section. Using other update formulas, as given in Column 2 of Table 3.1, allows the other agglomerative methods to be implemented in a very similar way to the implementation of the single link method.

In the case of the methods which use cluster centres, we have the centre coordinates (in Column 3 of Table 3.1) and dissimilarities as defined between cluster centres (Column 4 of Table 3.1). The Euclidean distance must be used, initially, for equivalence between the two approaches. In the case of the *median method*, for instance, we have the following (cf. Table 3.1).

Let $\mathbf{a}$ and $\mathbf{b}$ be two points (i.e. m–dimensional vectors: these are objects or cluster centres) which have been agglomerated, and let $\mathbf{c}$ be another point.

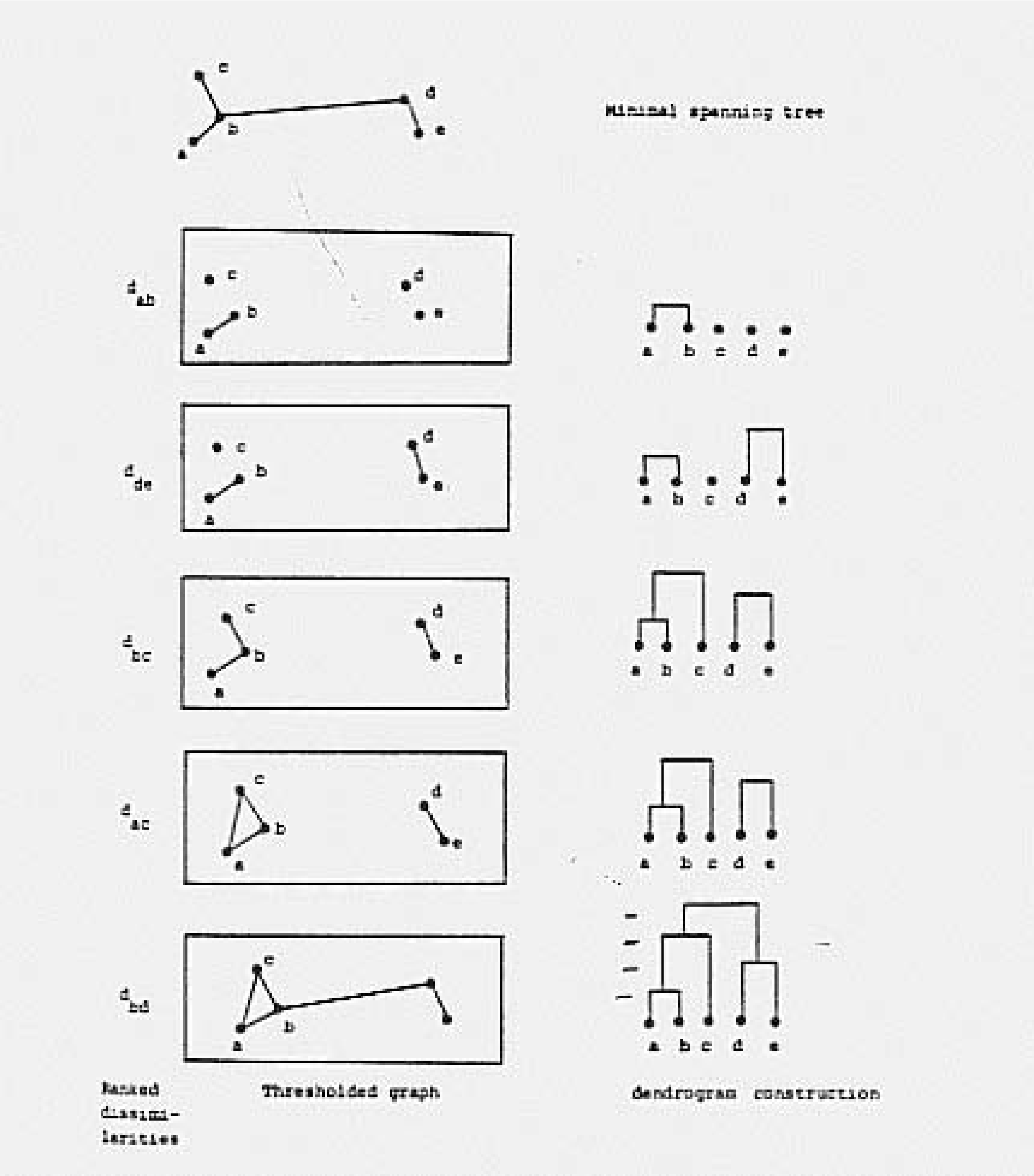


Figure 3.3: Another approach to constructing a single linkage dendrogram.

Hierarchical clustering methods (and aliases).	Lance and Williams dissimilarity update formula.	Coordinates of centre of cluster, which agglomerates clusters i and j.	Dissimilarity between cluster centres g_i and g_j.
Single link (nearest neighbour).	$\alpha_i = 0.5$ $\beta = 0$ $\gamma = -0.5$ (More simply: $min\{d_{ik}, d_{jk}\}$)		
Complete link (diameter).	$\alpha_i = 0.5$ $\beta = 0$ $\gamma = 0.5$ (More simply: $max\{d_{ik}, d_{jk}\}$)		
Group average (average link, UPGMA).	$\alpha_i = \frac{\|i\|}{\|i\|+\|j\|}$ $\beta = 0$ $\gamma = 0$		
McQuitty's method (WPGMA).	$\alpha_i = 0.5$ $\beta = 0$ $\gamma = 0$		
Median method (Gower's, WPGMC).	$\alpha_i = 0.5$ $\beta = -0.25$ $\gamma = 0$	$\mathbf{g} = \frac{\mathbf{g}_i + \mathbf{g}_j}{2}$	$\|\|\mathbf{g}_i - \mathbf{g}_j\|\|^2$
Centroid (UPGMC).	$\alpha_i = \frac{\|i\|}{\|i\|+\|j\|}$ $\beta = -\frac{\|i\|\|j\|}{(\|i\|+\|j\|)^2}$ $\gamma = 0$	$\mathbf{g} = \frac{\|i\|\mathbf{g}_i + \|j\|\mathbf{g}_j}{\|i\|+\|j\|}$	$\|\|\mathbf{g}_i - \mathbf{g}_j\|\|^2$
Ward's method (minimum variance, error sum of squares.	$\alpha_i = \frac{\|i\|+\|k\|}{\|i\|+\|j\|+\|k\|}$ $\beta = -\frac{\|k\|}{\|i\|+\|j\|+\|k\|}$ $\gamma = 0$	$\mathbf{g} = \frac{\|i\|\mathbf{g}_i + \|j\|\mathbf{g}_j}{\|i\|+\|j\|}$	$\frac{\|i\|\|j\|}{\|i\|+\|j\|}\|\|\mathbf{g}_i - \mathbf{g}_j\|\|^2$

Notes: $\mid i \mid$ is the number of objects in cluster i; $\mathbf{g}_i$ is a vector in m-space (m is the set of attributes), — either an intial point or a cluster centre; $\|.\|$ is the norm in the Euclidean metric; the names UPGMA, etc. are due to Sneath and Sokal (1973); finally, the Lance and Williams recurrence formula is:

$$d_{i \cup j,k} = \alpha_i d_{ik} + \alpha_j d_{jk} + \beta d_{ij} + \gamma \mid d_{ik} - d_{jk} \mid .$$

Table 3.1: Specifications of seven hierarchical clustering methods.

From the Lance–Williams dissimilarity update formula, using squared Euclidean distances, we have:

$$\begin{aligned} d^2(a \cup b, c) &= \tfrac{d^2(a,c)}{2} + \tfrac{d^2(b,c)}{2} - \tfrac{d^2(a,b)}{4} \\ &= \tfrac{\|\mathbf{a}-\mathbf{c}\|^2}{2} + \tfrac{\|\mathbf{b}-\mathbf{c}\|^2}{2} - \tfrac{\|\mathbf{a}-\mathbf{b}\|^2}{4}. \end{aligned} \tag{3.1}$$

The new cluster centre is $(\mathbf{a}+\mathbf{b})/2$, so that its distance to point $\mathbf{c}$ is

$$\|\mathbf{c} - \frac{\mathbf{a}+\mathbf{b}}{2}\|^2. \tag{3.2}$$

That these two expressions are identical is readily verified. The correspondence between these two perspectives on the one agglomerative criterion is similarly proved for the centroid and minimum variance methods.

The single linkage algorithm discussed in the last Section, duly modified for the use of the Lance–Williams dissimilarity update formula, is applicable for all agglomerative strategies. The update formula listed in Table 3.1 is used in Step 2 of the algorithm.

For cluster centre methods, and with suitable alterations for graph methods, the following algorithm is an alternative to the general dissimilarity based algorithm (the latter may be described as a “stored dissimilarities approach”).

Stored data approach

Step 1 Examine all interpoint dissimilarities, and form cluster from two closest points.

Step 2 Replace two points clustered by representative point (centre of gravity) or by cluster fragment.

Step 3 Return to Step 1, treating clusters as well as remaining objects, until all objects are in one cluster.

In Steps 1 and 2, “point” refers either to objects or clusters, both of which are defined as vectors in the case of cluster centre methods. This algorithm is justified by storage considerations, since we have $O(n)$ storage required for n initial objects and $O(n)$ storage for the $n-1$ (at most) clusters. In the case of linkage methods, the term “fragment” in Step 2 refers (in the terminology of graph theory) to a connected component in the case of the single link method and to a clique or complete subgraph in the case of the complete link method. The overall complexity of the above algorithm is $O(n^3)$: the repeated calculation of dissimilarities in Step 1, coupled with $O(n)$ iterations through Steps 1, 2 and 3. Note however that this does not take into consideration the extra processing required in a linkage method, where “closest” in Step 1 is defined with respect to graph fragments.

Recently some other very efficient improvements on this algorithm have been proposed (for a survey, see Murtagh, 1985). In particular there is the *Nearest Neighbour (NN) chain* algorithm. Here is a short description of this approach.

A *NN*–chain consists of an arbitrary point (a in Figure 3.4); followed by its *NN* (b in Figure 3.4); followed by the *NN* from among the remaining points (c,

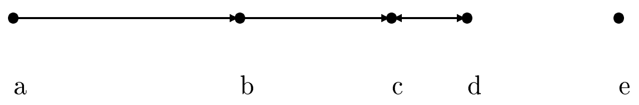


Figure 3.4: Five points, showing *NN*s and *RNN*s.

d, and e in Figure 3.4) of this second point; and so on until we necessarily have some pair of points which can be termed reciprocal or mutual *NN*s. (Such a pair of *RNN*s may be the first two points in the chain; and we have assumed that no two dissimilarities are equal.)

In constructing a *NN*–chain, irrespective of the starting point, we may agglomerate a pair of *RNN*s as soon as they are found. What guarantees that we can arrive at the same hierarchy as if we used the "stored dissimilarities" or "stored data" algorithms described earlier in this section? Essentially this is the same condition as that under which no inversions or reversals are produced by the clustering method. Figure 3.5 gives an example of this, where d is agglomerated at a lower criterion value (i.e. dissimilarity) than was the case at the previous agglomeration.

This is formulated as:

$$\text{Inversion impossible if: } d(i,j) < d(i,k) \text{ or } d(j,k) \Rightarrow d(i,j) < d(i \cup j, k)$$

Using the Lance–Williams dissimilarity update formula, it can be shown that the minimum variance method does not give rise to inversions; neither do the linkage methods; but the median and centroid methods cannot be guaranteed not to have inversions.

To return to Figure 3.4, if we are dealing with a clustering criterion which precludes inversions, then c and d can justifiably be agglomerated, since no other point (for example, b or e) could have been agglomerated to either of these.

The processing required, following an agglomeration, is to update the *NN*s of points such as b in Figure 3.4 (and on account of such points, this algorithm was initially dubbed *algorithme des célibataires* when first proposed!). The following is a summary of the algorithm:

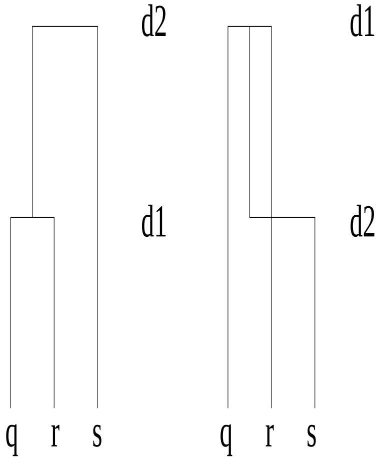


Figure 3.5: Alternative representations of a hierarchy with an inversion.

***NN*–chain algorithm**

Step 1 Select a point arbitrarily.

Step 2 Grow the *NN*–chain from this point until a pair of *RNN*s are obtained.

Step 3 Agglomerate these points (replacing with a cluster point, or updating the dissimilarity matrix).

Step 4 From the point which preceded the *RNN*s (or from any other arbitrary point if the first two points chosen in Steps 1 and 2 constituted a pair of *RNN*s), return to Step 2 until only one point remains.

3.2.4 Minimum Variance Method in Perspective

The next Section will review mathematical properties of the minimum variance method; in this Section we will informally motivate our preference for this method. First, let us briefly review the overall perspective.

Agglomerative clustering methods have been motivated by graph theory (leading to linkage–based methods) or by geometry (leading to cluster centre methods). This is true for the more commonly used methods studied here. In cluster centre methods, the cluster centre may be used for subsequent agglomerations. Alternatively, inter–cluster dissimilarities may be used throughout, and therefore these methods may be implemented using the Lance–Williams dissimilarity update formula (see Table 3.1) in an identical manner to the linkage–based methods.

In order to specify an agglomerative criterion simultaneously in terms of cluster mean vectors, and in terms of dissimilarity, it was also necessary to adopt a particular dissimilarity (i.e. the Euclidean distance). Restricting the choice of dissimilarity to this distance is not usually inconvenient in practice, and a Euclidean space offers a well–known and powerful standpoint for analysis.

The variance or spread of a set of points (i.e. the sum of squared distances from the centre) has been the point of departure for specifying many clustering algorithms. Many of these algorithms, — iterative, optimization algorithms as well as the hierarchical, agglomerative algorithms — are briefly described and appraised in Wishart (1969). The use of variance in a clustering criterion links the resulting clustering to other data–analytic techniques which involve a decomposition of variance. Principal Components Analysis, for example, which has been studied in Chapter 2 seeks the principal directions of elongation of the multidimensional points, i.e. the axes on which the projections of the points have maximal variance. Using a clustering of the points with minimal variance within clusters as the cluster criterion is, perhaps, the most suitable criterion for two different but complimentary analyses of the same set of points. The reality of clusters of projected points resulting from the Principal Components Analysis may be assessed using the Cluster Analysis results; and the interpretation of the axes of the former technique may be used to facilitate interpretation of the clustering results.

The search for clusters of maximum homogeneity leads to the minimum variance criterion. Since no coordinate axis is privileged by the Euclidean distance, the resulting clusters will be approximately hyperspherical. Such ball–shaped clusters will therefore be very unsuitable for examining straggly patterns of points. However, in the absence of information about such patterns in the data, homogeneous clusters will provide the most useful condensation of the data.

The following properties make the minimum variance agglomerative strategy particularly suitable for synoptic clustering:

1. As discussed in the Section to follow, the two properties of cluster homogeneity and cluster separability are incorporated in the cluster criterion. For summarizing data, it is unlikely that more suitable criteria could be devised.

2. As in the case of other geometric strategies, the minimum variance method defines a cluster centre of gravity. This mean set of cluster members' coordinate values is the most useful summary of the cluster. It may also be used for the fast selection and retrieval of data, by matching on these cluster representative vectors rather than on each individual object vector.

3. A top–down hierarchy traversal algorithm may also be implemented for information retrieval. Using a query vector, the left or right subtree is selected at each node for continuation of the traversal (it is best to ensure that each node has precisely two successor nodes in the construction of the hierarchy). Such an algorithm will work best if all top–down traversals through the hierarchy are of approximately equal length. This will be the case if and only if the hierarchy is as "symmetric" or "balanced" as possible (see Figure 3.6). Such a balanced hierarchy is usually of greatest interest for interpretative purposes also: a partition, derived from a hierarchy, and consisting of a large number of small classes, and one or a few large classes, is less likely to be of practical use.

   For such reasons, a "symmetric" hierarchy is desirable. It has been shown, using a number of different measures of hierarchic symmetry, that the minimum variance (closely followed by the complete link) methods generally give the most symmetric hierarchies (see Murtagh, 1984).

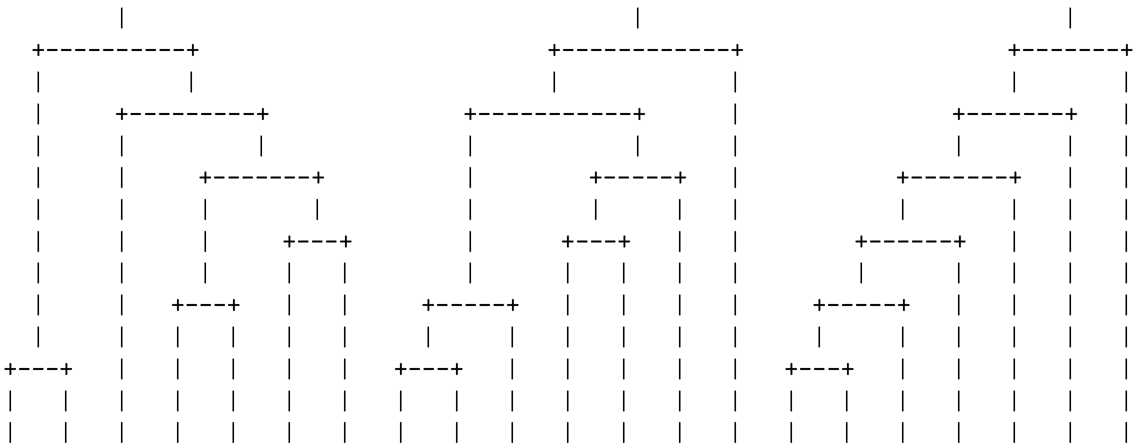


Figure 3.6: Three binary hierarchies: balanced, unbalanced and intermediate.

4. Unlike other geometric agglomerative methods — in particular the centroid and the median methods (see definitions, Table 3.1, above) — the sequence of agglomerations in the minimum variance method is guaranteed not to allow inversions in the cluster criterion value. Inversions or reversals (Figure 3.5) are inconvenient, and can make interpretation of the hierarchy difficult.

5. Finally, computational performance has until recently favoured linkage based agglomerative criteria, and in particular the single linkage method. The computational advances described above for the minimum variance method (principally the *NN*–chain algorithm) make it increasingly attractive for practical applications involving large amounts of data.

3.2.5 Minimum Variance Method: Mathematical Properties

The minimum variance method produces clusters which satisfy compactness and isolation criteria. These criteria are incorporated into the dissimilarity, noted in Table 3.1, as will now be shown.

In Ward's method, we seek to agglomerate two clusters, c_1 and c_2, into cluster c such that the within–class variance of the partition thereby obtained is minimum. Alternatively, the between–class variance of the partition obtained is to be maximized. Let P and Q be the partitions prior to, and subsequent to, the agglomeration; let p_1, p_2, ... be classes of the partitions:

$$\begin{aligned} P &= \{p_1, p_2, \ldots, p_k, c_1, c_2\} \\ Q &= \{p_1, p_2, \ldots, p_k, c\}. \end{aligned}$$

Finally, let i denote any individual or object, and I the set of such objects.

In the following, classes (i.e. p or c) and individuals (i.e. i) will be considered as vectors or as sets: the context, and the block typing of vectors, will be sufficient to make clear which is the case.

Total variance of the cloud of objects in m–dimensional space is decomposed into the sum of within–class variance and between–class variance. This is Huyghen's theorem in classical mechanics. Let V denote *variance*. The total variance of the cloud of objects is

$$V(I) = \frac{1}{n} \sum_{i \in I} (\mathbf{i} - \mathbf{g})^2$$

where $\mathbf{g}$ is the grand mean of the n objects: $\mathbf{g} = \frac{1}{n} \sum_{i \in I} \mathbf{i}$. The between–class variance is

$$V(P) = \sum_{p \in P} \frac{\mid p \mid}{n} (\mathbf{p} - \mathbf{g})^2$$

where $\mid p \mid$ is the cardinality of (i.e. number of members in) class p. (Note that $\mathbf{p}$ — in block type–face — is used to denote the centre of gravity — a vector — and p the set whose centre of gravity this is). Finally, the within–class variance is

$$\frac{1}{n} \sum_{p \in P} \sum_{i \in p} (\mathbf{i} - \mathbf{p})^2 .$$

For two partitions, before and after an agglomeration, we have respectively:

$$V(I) = V(P) + \sum_{p \in P} V(p)$$

$$V(I) = V(Q) + \sum_{p \in Q} V(p).$$

Hence,

$$\begin{aligned} &V(P) + V(p_1) + \ldots + V(p_k) + V(c_1) + V(c_2) \\ &\quad = V(Q) + V(p_1) + \ldots + V(p_k) + V(c). \end{aligned}$$

Therefore:

$$V(Q) = V(P) + V(c_1) + V(c_2) - V(c).$$

In agglomerating two classes of P, the variance of the resulting partition (i.e. $V(Q)$) will necessarily decrease: therefore in seeking to minimize this decrease, we simultaneously achieve a partition with maximum between–class variance. The criterion to be optimized can then be shown to be:

$$\begin{aligned} V(P) - V(Q) &= V(c) - V(c_1) - V(c_2) \\ &= \tfrac{|c_1| \ |c_2|}{|c_1| + |c_2|} ||\mathbf{c_1} - \mathbf{c_2}||^2 \ , \end{aligned}$$

which is the dissimilarity given in Table 3.1. This is a dissimilarity which may be determined for any pair of classes of partition P; and the agglomerands are those classes, c_1 and c_2, for which it is minimum.

It may be noted that if c_1 and c_2 are singleton classes, then $V(\{c_1, c_2\}) = \frac{1}{2} ||\mathbf{c}_1 - \mathbf{c}_2||^2$ (i.e. the variance of a pair of objects is equal to half their Euclidean distance).

3.2.6 Minimal Spanning Tree

Aspects of the minimal spanning tree (MST) are covered in most texts on graph theory, and on many other areas besides. In graph theory — a subdiscipline of discrete mathematics — *vertices* (i.e. points) and *edges* (i.e. lines joining the points) are commonly dealt with. A graph is defined as a set of such vertices and edges. A MST contains (*spans*) all vertices, and has minimal totalled edge length (or *weights*). We will restrict ourselves to a brief description of this important method. The following algorithm formalises Figure 3.3 in constructing a MST by a "greedy" or nearest neighbour approach.

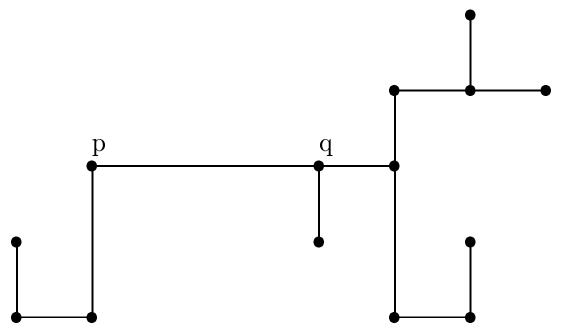


Figure 3.7: Minimal spanning tree of a point pattern (non-unique).

Minimal Spanning Tree algorithm

Step 1 Select an arbitrary point and connect it to the least dissimilar neighbour. These two points constitute a subgraph of the MST.

Step 2 Connect the current subgraph to the least dissimilar neighbour of any of the members of the subgraph.

Step 3 Loop on Step 2, until all points are in the one subgraph: this, then, is the MST.

Step 2 agglomerates subsets of objects using the criterion of connectivity. For proof that this algorithm does indeed produce a MST, see for example Tucker (1980). The close relationship between the MST and the single linkage hierarchical clustering method is illustrated in Figure 3.3.

A *component* in a graph is a subgraph consisting of a set of vertices with at least one edge connecting each vertex to some other vertex in the component. Hence a component is simply one possible definition of a cluster, — in fact, a component is closely related to clusters which can be derived from a single linkage hierarchy (cf. Figure 3.3).

Breaking up the MST, and thereby automatically obtaining components, is a problem addressed by Zahn (1971). He defined an edge to be *inconsistent* if it is of length much greater than the lengths of other nearby edges (see Figure 3.7).

Inconsistent edges may be picked out by looking at a histogram of edge lengths in the MST, and marking as deletable a set percentage of greatest–length edges. Alternatively, inconsistent edges may be obtained by defining a threshold of inconsistency: if it is supposed, for instance, that edge lengths are normally distributed, a threshold of two standard deviations above the mean length of all edges which are within two edges (say, for some robustness) of the given vertex may be a useful indicator of inconsistency. Rarely is it worthwhile to attempt to test a supposition such as normality of edge lengths, and instead a rule such as this would be judged only on the results given in practice. Zahn applied these approaches to point pattern recognition, — obtaining what he termed "Gestalt patterns" among sets of planar points (see Fig. 3.8); picking out bubble chamber particle tracks, indicated by curved sequences of points; and detecting density gradients, where differing clusters of points have different densities associated with them and hence are distinguishable to the human eye. The MST provides a useful starting point for undertaking such pattern recognition problems.

The MST is also often suitable for outlier detection. Since outlying data items will be of greater than average distance from their neighbours in the MST, they may be detected by drawing a histogram of edge lengths. Unusually large lengths will indicate the abnormal points (or data items) sought. Rohlf (1975) gave a statistical gap test, under the assumption that the edge lengths in the MST were normally distributed.

3.2.7 Partitioning Methods

We will conclude this Chapter with a short look at other non–hierarchical clustering methods.

A large number of assignment algorithms have been proposed. The single–pass approach usually achieves computational efficiency at the expense of precision, and there are many iterative approaches for improving on crudely–derived partitions.

As an example of a single–pass algorithm, the following one is given in Salton and McGill (1983). The general principle followed is: make one pass through the data, assigning each object to the first cluster which is close enough, and making a new cluster for objects that are not close enough to any existing cluster.

Single–pass overlapping cluster algorithm

Input n objects, threshold t, dissimilarity on objects.

Step 1 Read object 1, and insert object 1 in membership list of cluster 1. Let representative of cluster 1 be given by object 1. Set i to 2.

Step 2 Read i^{th} object. If *diss*(i^{th} object, cluster j) $\leq t$, for any cluster j, then include the i^{th} object in the membership list of cluster j, and update the cluster representative vector to take account of this new member. If *diss*(i^{th} object, cluster j) $> t$, for all clusters j, then create a new cluster, placing the i^{th} object in its membership list, and letting the representative of this cluster be defined by the i^{th} object.

Step 3 Set i to $i+1$. If $i \leq n$, go to Step 2.

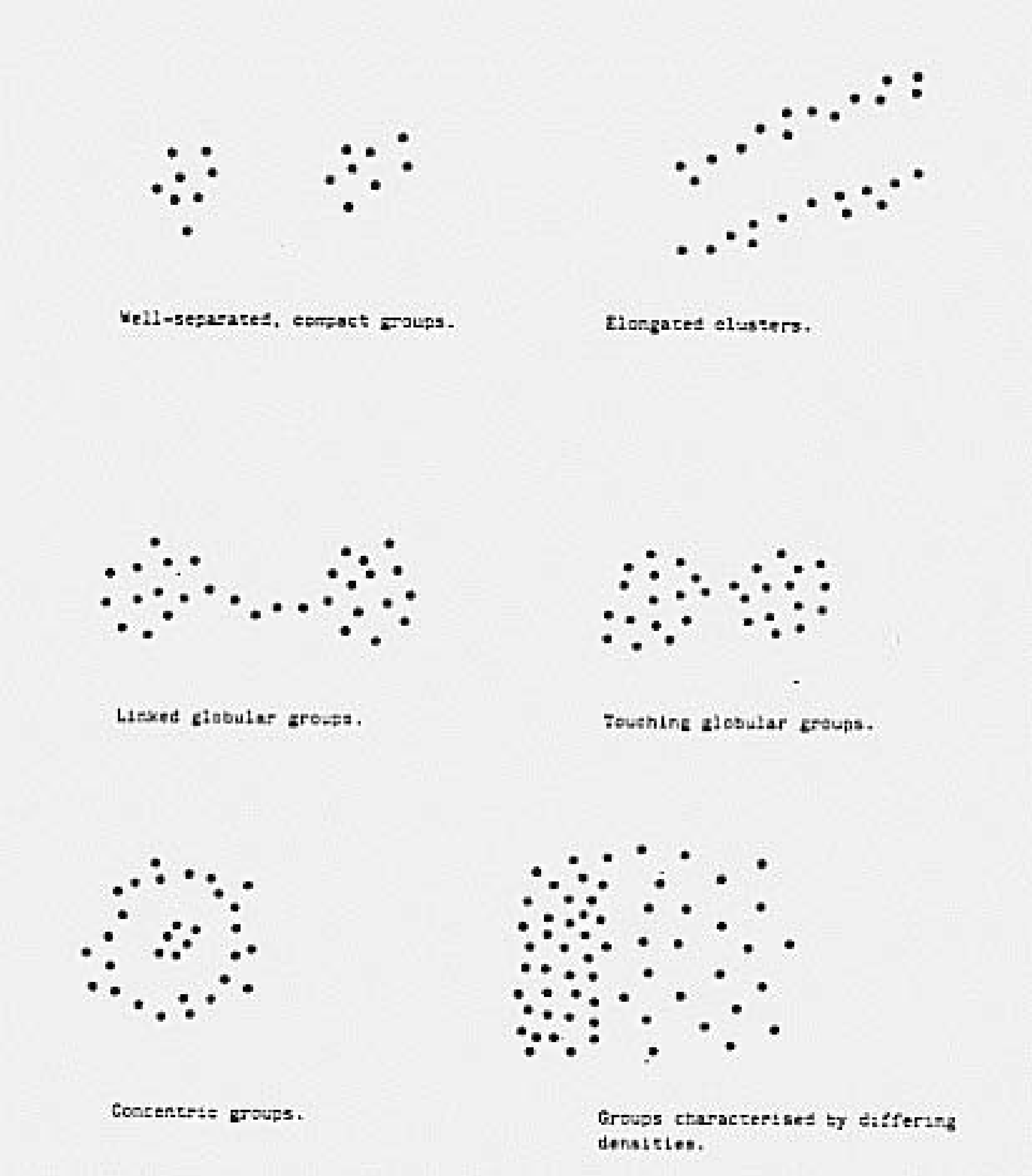

Well-separated, compact groups.
Elongated clusters.
Linked globular groups.
Touching globular groups.
Concentric groups.
Groups characterised by differing densities.

Figure 3.8: Differing point patterns.

The cluster representative vector used is usually the mean vector of the cluster's members; in the case of binary data, this representative might then be thresholded to have 0 and 1 coordinate values only. In Step 2, it is clear that overlapping clusters are possible in the above algorithm. In the worst case, if threshold t is chosen too low, all n objects will constitute clusters and the number of comparisons to be carried out will be $O(n^2)$. The dependence of the algorithm on the given sequence of objects is an additional disadvantage of this algorithm. However, its advantages are that it is conceptually very simple, and for a suitable choice of threshold will probably not require large processing time. In practice, it can be run for a number of different values of t.

As a non-hierarchic strategy, it is hardly surprising that the variance criterion has always been popular (for some of the same reasons as were seen in Section 3.2.4 for the hierarchical approach based on this criterion). We may, for instance, minimize the within–class variance

$$V_{opt} = min_P \sum_{p \in P} \sum_{i \in p} ||\mathbf{i} - \mathbf{p}||^2$$

where the partition P consists of classes p of centre $\mathbf{p}$, and we desire the minimum of this criterion over all possible partitions, P. To avoid a nontrivial outcome (e.g. each class being singleton, giving zero totalled within class variance), the number of classes (k) must be set.

A hierarchical clustering, using the minimum variance criterion, then provides a solution, — not necessarily optimal — at level $n - k$ when n objects are being processed. An alternative approach uses iterative refinement, as follows.

Iterative optimization algorithm for the variance criterion

Step 1 Arbitrarily define a set of k cluster centres.

Step 2 Assign each object to the cluster to which it is closest (using the Euclidean distance, $d^2(i,p) = ||\mathbf{i} - \mathbf{p}||^2$).

Step 3 Redefine cluster centres on the basis of the current cluster memberships.

Step 4 If the totalled within class variances is better than at the previous iteration, then return to Step 2.

We have omitted in Step 4 a test for convergence (the number of iterations should not exceed, e.g., 25). Cycling is also possible between solution states. This algorithm could be employed on the results (at level $n - k$) of a hierarchic clustering in order to improve the partition found. It is however a suboptimal algorithm, — the minimal distance strategy used by this algorithm is clearly a *sufficient* but not a *necessary* condition for an optimal partition.

The initial cluster centres may be chosen arbitrarily (for instance, by averaging a small number of object–vectors); or they may be chosen from prior knowledge of the data. In the latter case we may for example "manually" choose a small set of stars and galaxies, determine their (parameter–space) centres of gravity, and use these as the basis for the classification of objects derived from a digitized image.

Yet another approach to optimizing the same minimum variance criterion is the exchange method.

Exchange method for the minimum variance criterion

Step 1 Arbitrarily choose an initial partition.

Step 2 For each $i \in p$, see if the criterion is bettered by relocating i in another class q. If this is the case, we choose class q such that the criterion V is least; if it is not the case, we proceed to the next i.

Step 3 If the maximum possible number of iterations has not been reached, and if at least one relocation took place in Step 2, return again to Step 2.

Two remarks may be made: we will normally require that an object not be removed from a singleton class; and the change in variance brought about by relocating object i from class p to class q can be shown to be

$$\frac{|p|}{|p|-1}||\mathbf{i}-\mathbf{p}||^2 - \frac{|q|}{|q|-1}||\mathbf{i}-\mathbf{q}||^2$$

Hence, if this expression is positive, i ought to be relocated from class p (to class q if another, better, class is not found).

Späth (1985) offers a lucid and thorough treatment of algorithms of the sort described.

In terminating, it is necessary to make some suggestion as to when these partitioning algorithms should be used in preference to hierarchical algorithms. We have seen that the number of classes must be specified, as also the requirement that each class be non-empty. A difficulty with iterative algorithms, in general, is the requirement for parameters to be set in advance (Anderberg, 1973, describes a version of the ISODATA iterative clustering method which requires 7 pre–set parameters). As a broad generalization, it may thus be asserted that iterative algorithms ought to be considered when the problem is clearly defined in terms of numbers and other characteristics of clusters; but hierarchical routines often offer a more general–purpose and user–friendly option.

3.3 Examples and Bibliography

3.3.1 Examples from Astronomy

Principal Components Analysis has often been used for determining a classification, and these references are not included in this Section (see Section 2.3.3).

The problems covered in the following include: star-galaxy separation, using digitized image data; spectral classification, — the prediction of spectral type from photometry; taxonomy construction (for asteroids, stars, and stellar light curves); galaxies; gamma and X–ray astronomy; a clustering approach not widely used elsewhere is employed for studies relating to the Moon, to asteroids and to cosmic sources; and work relating to interferogram analysis is represented.

1. J.D. Barrow, S.P. Bhavsar and D.H. Sonoda, "Minimal spanning trees, filaments and galaxy clustering", *Monthly Notices of the Royal Astronomical Society*, **216**, 17–35, 1985.

   (This article follows the seminal approach of Zahn — see reference among the general clustering works — in using the MST for finding visually evident groupings.)

2. R. Bianchi, A. Coradini and M. Fulchignoni, "The statistical approach to the study of planetary surfaces", *The Moon and the Planets*, **22**, 293–304, 1980.

   (This article contains a general discussion which compares the so–called G–mode clustering method to other multivariate statistical methods. References 7 and 8 below also use this method.)

3. R. Bianchi, J.C. Butler, A. Coradini and A.I. Gavrishin, "A classification of lunar rock and glass samples using the G–mode central method", *The Moon and the Planets*, **22**, 305–322, 1980.

4. A. Bijaoui, "Méthodes mathématiques pour la classification stellaire", in *Classification Stellaire, Compte Rendu de l'Ecole de Goutelas*, ed. D. Ballereau, Observatoire de Meudon, Meudon, 1979, pp. 1–54.

   (This presents a survey of clustering methods.)

5. R. Buccheri, P. Coffaro, G. Colomba, V. Di Gesù and S. Salemi, "Search of significant features in a direct non–parametric pattern recognition method. Application to the classification of multiwire spark chamber pictures", in (eds.) C. de Jager and H. Nieuwenhuijzen, *Image Processing Techniques in Astronomy*, D. Reidel, Dordrecht, pp. 397–402, 1975.

   (A technique is developed for classifying γ–ray data.)

6. S.A. Butchins, "Automatic image classification", *Astronomy and Astrophysics*, **109**, 360–365, 1982.

   (A method for determining Gaussian clusters, due to Wolf, is used for star/galaxy separation in photometry.)

7. A. Coradini, M. Fulchignoni and A.I. Gavrishin, "Classification of lunar rocks and glasses by a new statistical technique", *The Moon*, **16**, 175–190, 1976.

   (The above, along with the references of Bianchi and others, make use of a clustering technique called the G–mode method. The above contains a short mathematical description of the technique proposed.)

8. A. Carusi and E. Massaro, "Statistics and mapping of asteroid concentrations in the proper elements' space", *Astronomy and Astrophysics Supplement Series*, **34**, 81–90, 1978.

(This article also uses the so–called G–mode method, employed by Bianchi, Coradini, and others.)

9. C.R. Cowley and R. Henry, "Numerical taxonomy of Ap and Am stars", *The Astrophysical Journal*, **233**, 633–643, 1979.

(40 stars are used, characterised on the strength with which particular atomic spectra — the second spectra of yttrium, the lanthanides, and the iron group — are represented in the spectrum. Stars with very similar spectra end up correctly grouped; and anomolous objects are detected. Clustering using lanthanides, compared to clustering using iron group data, gives different results for A_p stars. This is not the case for A_m stars, which thus appear to be less heterogeneous. The need for physical explanations are thus suggested.)

10. C.R. Cowley, "Cluster analysis of rare earths in stellar spectra", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 1983, pp. 153–156.

(About twice the number of stars, as used in the previous reference, are used here. A greater role is seen for chemical explanations of stellar abundances and/or spectroscopic patterns over nuclear hypotheses.)

11. J.K. Davies, N. Eaton, S.F. Green, R.S. McCheyne and A.J. Meadows, "The classification of asteroids", *Vistas in Astronomy*, **26**, 243–251, 1982.

(Physical properties of 82 asteroids are used. The dendrogram obtained is compared with other classification schemes based on spectral characteristics or colour–colour diagrams. The clustering approach used is justified also in being able to pinpoint objects of particular interest for further observation; and in allowing new forms of data — e.g. broadband infrared photometry — to be quickly incorporated into the overall approach of classification–construction.)

12. G.A. De Biase, V. di Gesù and B. Sacco, "Detection of diffuse clusters in noise background", *Pattern Recognition Letters* **4**, 39–44, 1986.

13. P.A. Devijver, "Cluster analysis by mixture identification", in V. Di Gesù, L. Scarsi, P. Crane, J.H. Friedman and S. Levialdi (eds.), *Data Analysis in Astronomy*, Plenum Press, New York, 1984, pp. 29–44.

(A useful review article, with many references. A perspective similar to perspectives adopted by many discriminant analysis methods is used.)

14. V. Di Gesù and B. Sacco, "Some statistical properties of the minimum spanning forest", *Pattern Recognition*, **16**, 525–531, 1983.

(In this and the following works, the minimal spanning tree or fuzzy set theory — which, is clear from the article titles — are applied to point pattern distinguishing problems involving gamma and X–ray data. For a rejoinder to the foregoing reference, see R.C. Dubes and R.L. Hoffman, "Remarks on some statistical properties of the minimum spanning forest", *Pattern Recognition*, **19**, 49–53, 1986. A reply to this article is forthcoming, from the authors of the original paper.)

15. V. Di Gesù, B. Sacco and G. Tobia, "A clustering method applied to the analysis of sky maps in gamma–ray astronomy", *Memorie della Società Astronomica Italiana*, 517–528, 1980.

16. V. Di Gesù and M.C. Maccarone, "A method to classify celestial shapes based on the possibility theory", in G. Sedmak (ed.), ASTRONET 1983 (Convegno Nazionale Astronet, Brescia, Published under the auspices of the Italian Astronomical Society), 355–363, 1983.

17. V. Di Gesù and M.C. Maccarone, "Method to classify spread shapes based on possibility theory", Proceedings of the 7th International Conference on Pattern Recognition, Vol. 2, IEEE Computer Society, 1984, pp. 869–871.

18. V. Di Gesù and M.C. Maccarone, "Features selection and possibility theory", *Pattern Recognition*, *19*, 63–72, 1986.

19. J.V. Feitzinger and E. Braunsfurth, "The spatial distribution of young objects in the Large Magellanic Cloud — a problem of pattern recognition", in eds. S. van den Bergh and K.S. de Boer, *Structure and Evolution of the Magellanic Clouds*, IAU, 93–94, 1984.

    (In an extended abstract, the use of linkages between objects is described.)

20. I.E. Frank, B.A. Bates and D.E. Brownlee, "Multivariate statistics to analyze extraterrestrial particles from the ocean floor", in V. Di Gesù, L. Scarsi, P. Crane, J.H. Friedman and S. Levialdi (eds.), *Data Analysis in Astronomy*, Plenum Press, New York, 1984.

21. A. Fresneau, "Clustering properties of stars outside the galactic disc", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 1983, pp. 17–20.

    (Techniques from the spatial processes area of statistics are used to assess clustering tendencies of stars.)

22. A. Heck, A. Albert, D. Defays and G. Mersch, "Detection of errors in spectral classification by cluster analysis", *Astronomy and Astrophysics*, **61**, 563–566, 1977.

23. A. Heck, D. Egret, Ph. Nobelis and J.C. Turlot, "Statistical confirmation of the UV spectral classification system based on IUE low–dispersion stellar spectra", *Astrophysics and Space Science*, **120**, 223–237, 1986.

    (Among other results, it is found that UV standard stars are located in the neighbourhood of the centres of gravity of groups found, thereby helping to verify the algorithm implemented. A number of other papers, by the same authors, analysing IUE spectra are referenced in this paper. Apart from the use of a large range of clustering methods, these papers also introduce a novel weighting procedure — termed the "variable Procrustean bed" (see Chapter 6) — which adjusts for the symmetry/asymmetry of the spectrum. Therefore, a useful study of certain approaches to the coding of data is to be found in these papers.)

24. J.P. Huchra and M.J. Geller, "Groups of galaxies. I. Nearby groups", *The Astrophysical Journal*, **257**, 423–437, 1982.

(The single linkage hierarchical method, or the minimal spanning tree, have been rediscovered many times — see, for instance, Graham and Hell, 1985, referenced in the general clustering section. In this study, a close variant is used for detecting groups of galaxies using three variables, — two positional variables and redshift.)

25. J.F. Jarvis and J.A. Tyson, "FOCAS: faint object classification and analysis system", *The Astronomical Journal*, **86**, 476–495, 1981.

(An iterative minimal distance partitioning method is employed in the FOCAS system to arrive at star/galaxy/other classes.)

26. G. Jasniewicz, "The Böhm–Vitense gap in the Geneva photometric system", *Astronomy and Astrophysics*, *141*, 116–126, 1984.

(The minimal spanning tree is used on colour–colour diagrams.)

27. A. Kruszewski, "Object searching and analyzing commands", in *MIDAS — Munich Image Data Analysis System*, European Southern Observatory Operating Manual No. 1, Chapter 11, 1985.

(The *Inventory* routine in MIDAS has a non–hierarchical iterative optimization algorithm. It can immediately work on up to 20 parameters, determined for each object in a scanned image.)

28. M.J. Kurtz, "Classification methods: an introductory survey", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 1983, pp. 47–58.

(Kurtz lists a large number of parameters — and functions of these parameters — which have been used to differentiate stars from galaxies.)

29. J. Materne, "The structure of nearby clusters of galaxies. Hierarchical clustering and an application to the Leo region", *Astronomy and Astrophysics*, **63**, 401–409, 1978.

(Ward's minimum variance hierarchic method is used, following discussion of the properties of other hierarchic methods.)

30. M.O. Mennessier, "A cluster analysis of visual and near–infrared light curves of long period variable stars", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 1983, pp. 81–84.

(Light curves — the variation of luminosity with time in a wavelength range — are analysed. Standardization is applied, and then three hierarchical methods. "Stable clusters" are sought from among all of these. The study is continued in the following.)

31. M.O. Mennessier, "A classification of miras from their visual and near–infrared light curves: an attempt to correlate them with their evolution", *Astronomy and Astrophysics*, **144**, 463–470, 1985.

32. MIDAS (Munich Image Data Analysis System), European Southern Observatory, Garching–bei–München (Version 4.1, January 1986). Chapter 13: Multivariate Statistical Methods.

(This premier astronomical data reduction package contains a large number of multivariate algorithms.)

33. M. Moles, A. del Olmo and J. Perea, "Taxonomical analysis of superclusters. I. The Hercules and Perseus superclusters", *Monthly Notices of the Royal Astronomical Society*, **213**, 365–380, 1985.

    (A non–hierarchical descending method, used previously by Paturel, is employed.)

34. F. Murtagh, "Clustering techniques and their applications", *Data Analysis and Astronomy* (Proceedings of International Workshop on Data Analysis and Astronomy, Erice, Italy, April 1986) Plenum Press, New York, 1986 (in press).

35. F. Murtagh and A. Lauberts, "A curve matching problem in astronomy", *Pattern Recognition Letters*, 1986 (in press).

    (A dissimilarity is defined between galaxy luminosity profiles, in order to arrive at a spiral–elliptical grouping.)

36. G. Paturel, "Etude de la région de l'amas Virgo par taxonomie", *Astronomy and Astrophysics*, **71**, 106–114, 1979.

    (A descending non–hierarchical method is used.)

37. J. Perea, M. Moles and A. del Olmo, "Taxonomical analysis of the Cancer cluster of galaxies", *Monthly Notices of the Royal Astronomical Society*, **222**, 49–53, 1986.

    (A non–hierarchical descending method is used.)

38. D.J. Tholen, "Asteroid taxonomy from cluster analysis of photometry", PhD Thesis, University of Arizona, 1984.

    (Between 400 and 600 asteroids using good–quality multi–colour photometric data are analysed.)

39. F. Giovannelli, A. Coradini, J.P. Lasota and M.L. Polimene, "Classification of cosmic sources: a statistical approach", *Astronomy and Astrophysics*, **95**, 138–142, 1981.

40. B. Pirenne, D. Ponz and H. Dekker, "Automatic analysis of interferograms", *The Messenger*, No. 42, 2–3, 1985.

    (The minimal spanning tree is used to distinguish fringes; there is little description of the MST approach in the above article, but further articles are in preparation and the software — and accompanying documentation — are available in the European Southern Observatory's MIDAS image processing system.)

41. A. Zandonella, " Object classification: some methods of interest in astronomical image analysis", in *Image Processing in Astronomy*, eds. G. Sedmak, N. Capaccioli and R.J. Allen, Osservatorio Astronomico di Trieste, Trieste, 304–318, 1979.

    (This presents a survey of clustering methods.)

3.3.2 General References

1. M.R. Anderberg, *Cluster Analysis for Applications*, Academic Press, New York, 1973.

   (A little dated, but still very much referenced; good especially for similarities and dissimilarities.)

2. J.P. Benzécri et coll., *L'Analyse des Données. I. La Taxinomie*, Dunod, Paris, 1979 (3rd ed.).

   (Very influential in the French speaking world; extensive treatment, and impressive formalism.)

3. R.K. Blashfield and M.S. Aldenderfer, "The literature on cluster analysis", *Multivariate Behavioral Research*, **13**, 271–295, 1978.

4. H.H. Bock, *Automatische Klassifikation*, Vandenhoek und Rupprecht, Göttingen, 1974.

   (Encyclopaedic.)

5. CLUSTAN, Clustan Ltd., 16 Kingsburgh Road, Edinburgh EH12 6DZ, Scotland.

   (The only exclusively clustering package available.)

6. B. Everitt, *Cluster Analysis*, Heinemann Educational Books, London, 1980 (2nd ed.).

   (A readable, introductory text.)

7. A.D. Gordon, *Classification*, Chapman and Hall, London, 1981.

   (Another recommendable introductory text.)

8. R.L. Graham and P. Hell, "On the history of the minimum spanning tree problem", *Annals of the History of Computing*, **7**, 43–57, 1985.

   (An interesting historical study.)

9. J.A. Hartigan, *Clustering Algorithms*, Wiley, New York, 1975.

   (Often referenced, this book could still be said to be innovative in its treatment of clustering problems; it contains a wealth of sample data sets.)

10. M. Jambu and M.O. Lebeaux, *Cluster Analysis and Data Analysis*, North–Holland, Amsterdam, 1983.

    (Some of the algorithms discussed have been overtaken by, for instance, the "nearest neighbour chain" or "reciprocal nearest neighbour" algorithms.)

11. L. Lebart, A. Morineau and K.M. Warwick, *Multivariate Descriptive Statistical Analysis*, Wiley, New York, 1984.

    (A useful book, centred on Multiple Correspondence Analysis.)

12. R.C.T. Lee, "Clustering analysis and its applications", in J.T. Tou (ed.) *Advances in Information Systems Science, Vol. 8*, Plenum Press, New York, 1981, pp. 169–292.

    (Practically book–length, it is especially useful for the links between clustering and problems in computing and in Operations Research.)

13. F. Murtagh, "Structure of hierarchic clusterings: implications for information retrieval and for multivariate data analysis", *Information Processing and Management*, **20**, 611–617, 1984.

14. F. Murtagh, *Multidimensional Clustering Algorithms*, COMPSTAT Lectures Volume 4, Physica–Verlag, Wien, 1985.

    (Algorithmic details of a range of clustering methods.)

15. A. Rapoport and S. Fillenbaum, "An experimental study of semantic structures", in eds. A.K. Romney, R.N. Shepard and S.B. Nerlove, *Multidimensional Scaling; Theory and Applications in the Behavioral Sciences. Vol. 2, Applications*, Seminar Press, New York, 93–131, 1972.

16. F.J. Rohlf, "Generalization of the gap test for the detection of multivariate outliers", *Biometrics*, **31**, 93–101, 1975.

    (One application of the Minimal Spanning Tree.)

17. G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*, McGraw–Hill, New York, 1983.

    (A central reference in the information retrieval area.)

18. P.H.A. Sneath and R.R. Sokal, *Numerical Taxonomy*, Freeman, San Francisco, 1973.

    (Very important for biological applications, it also has some impressive collections of graph representations of clustering results.)

19. H. Späth, *Cluster Dissection and Analysis: Theory, Fortran Programs, Examples*, Ellis Horwood, Chichester, 1985.

    (Recommendable reference for non–hierarchic, partitioning methods.)

20. A. Tucker, *Applied Combinatorics*, Wiley, New York, 1980.

    (For graph theory and combinatorics.)

21. D. Wishart, "Mode analysis: a generalization of nearest neighbour which reduces chaining effects", in ed. A.J. Cole, *Numerical Taxonomy*, Academic Press, New York, 282–311, 1969.

    (Discusses various variance–based clustering criteria which, interestingly, are justified by the difficulties experienced by more mainstream algorithms in clustering data of the type found in the H–R diagram.)

22. C.T. Zahn, "Graph–theoretical methods for detecting and describing Gestalt clusters", *IEEE Transactions on Computers*, **C–20**, 68–86, 1971.

    (Central reference for the use of the Minimal Spanning Tree for processing point patterns.)

3.4 Software and Sample Implementation

3.4.1 Fortran Code, Sample Input and Output

In the first section, following, hierarchical clustering is dealt with. The principal subroutines are HC, which carries out a dissimilarity–based hierarchical clustering; and HCON2, which carries out a hierarchic clustering based on the original data in $O(n^2)$ time. The use of HCASS and HCDEN may follow either: these subroutines determine class assignments and allow a dendrogram to be drawn.

The second section, which follows, deals with partitioning. Following Späth (1985), the essential subroutines are MINDST and EXCH for the minimal distance and exchange methods, respectively.

3.4.2 Program Listing: Hierarchical Clustering

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  HIERARCHICAL CLUSTERING using (user-specified) criterion.
C
C  Parameters:
C
C  DATA(N,M)         input data matrix,
C  DISS(LEN)         dissimilarities in lower half diagonal
C                    storage; LEN = N.N-1/2,
C  IOPT              clustering criterion to be used,
C  IA, IB, CRIT      history of agglomerations; dimensions
C                    N, first N-1 locations only used,
C  MEMBR, NN, DISNN  vectors of length N, used to store
C                    cluster cardinalities, current nearest
C                    neighbour, and the dissimilarity assoc.
C                    with the latter.
C  FLAG              boolean indicator of agglomerable obj./
C                    clusters.
C
C-------------------------------------------------------
      SUBROUTINE HC(N,M,LEN,IOPT,DATA,IA,IB,CRIT,MEMBR,NN,DISNN,
     X                FLAG,DISS)
      REAL DATA(N,M),MEMBR(N),DISS(LEN)
      INTEGER IA(N),IB(N)
      REAL CRIT(N)
      DIMENSION NN(N),DISNN(N)
      LOGICAL FLAG(N)
      REAL INF
      DATA INF/1.E+20/
C
C  Initializations
C
      DO 10 I=1,N
         MEMBR(I)=1.
         FLAG(I)=.TRUE.
 10   CONTINUE
      NCL=N
C
C  Construct dissimilarity matrix
C
      DO 40 I=1,N-1
         DO 30 J=I+1,N
```

```
            IND=IOFFS(N,I,J)
            DISS(IND)=0.
            DO 20 K=1,M
               DISS(IND)=DISS(IND)+(DATA(I,K)-DATA(J,K))**2
 20         CONTINUE
            IF (IOPT.EQ.1) DISS(IND)=DISS(IND)/2.
C           (Above is done for the case of the min. var. method
C            where merging criteria are defined in terms of
C            variances rather than distances.)
 30       CONTINUE
 40   CONTINUE
C
C  Carry out an agglomeration - first create list of NNs
C
      DO 60 I=1,N-1
         DMIN=INF
         DO 50 J=I+1,N
            IND=IOFFS(N,I,J)
            IF (DISS(IND).GE.DMIN) GOTO 50
               DMIN=DISS(IND)
               JM=J
 50      CONTINUE
         NN(I)=JM
         DISNN(I)=DMIN
 60   CONTINUE
C
 70   CONTINUE
C     Next, determine least diss. using list of NNs
      DMIN=INF
      DO 80 I=1,N-1
         IF (.NOT.FLAG(I)) GOTO 80
         IF (DISNN(I).GE.DMIN) GOTO 80
            DMIN=DISNN(I)
            IM=I
            JM=NN(I)
 80   CONTINUE
      NCL=NCL-1
C
C  This allows an agglomeration to be carried out.
C
      I2=MIN0(IM,JM)
      J2=MAX0(IM,JM)
      IA(N-NCL)=I2
      IB(N-NCL)=J2
      CRIT(N-NCL)=DMIN
C
C  Update dissimilarities from new cluster.
C
      FLAG(J2)=.FALSE.
      DMIN=INF
      DO 170 K=1,N
         IF (.NOT.FLAG(K)) GOTO 160
         IF (K.EQ.I2) GOTO 160
         X=MEMBR(I2)+MEMBR(J2)+MEMBR(K)
         IF (I2.LT.K) IND1=IOFFS(N,I2,K)
         IF (I2.GE.K) IND1=IOFFS(N,K,I2)
         IF (J2.LT.K) IND2=IOFFS(N,J2,K)
         IF (J2.GE.K) IND2=IOFFS(N,K,J2)
         IND3=IOFFS(N,I2,J2)
         XX=DISS(IND3)
C
C  WARD'S MINIMUM VARIANCE METHOD - IOPT=1.
```

```
C
         IF (IOPT.NE.1) GOTO 90
            DISS(IND1)=(MEMBR(I2)+MEMBR(K))*DISS(IND1)+
     X                 (MEMBR(J2)+MEMBR(K))*DISS(IND2)-
     X                 MEMBR(K)*XX
            DISS(IND1)=DISS(IND1)/X
 90      CONTINUE
C
C  SINGLE LINK METHOD - IOPT=2.
C
         IF (IOPT.NE.2) GOTO 100
            DISS(IND1)=MIN(DISS(IND1),DISS(IND2))
 100     CONTINUE
C
C  COMPLETE LINK METHOD - IOPT=3.
C
         IF (IOPT.NE.3) GOTO 110
            DISS(IND1)=MAX(DISS(IND1),DISS(IND2))
 110     CONTINUE
C
C  AVERAGE LINK (OR GROUP AVERAGE) METHOD - IOPT=4.
C
         IF (IOPT.NE.4) GOTO 120
            DISS(IND1)=(MEMBR(I2)*DISS(IND1)+MEMBR(J2)*DISS(IND2))/
     X                 (MEMBR(I2)+MEMBR(J2))
 120     CONTINUE
C
C  MCQUITTY'S METHOD - IOPT=5.
C
         IF (IOPT.NE.5) GOTO 130
            DISS(IND1)=0.5*DISS(IND1)+0.5*DISS(IND2)
 130     CONTINUE
C
C  MEDIAN (GOWER'S) METHOD - IOPT=6.
C
         IF (IOPT.NE.6) GOTO 140
            DISS(IND1)=0.5*DISS(IND1)+0.5*DISS(IND2)-0.25*XX
 140     CONTINUE
C
C  CENTROID METHOD - IOPT=7.
C
         IF (IOPT.NE.7) GOTO 150
            DISS(IND1)=(MEMBR(I2)*DISS(IND1)+MEMBR(J2)*DISS(IND2)-
     X          MEMBR(I2)*MEMBR(J2)*XX/(MEMBR(I2)+MEMBR(J2)))/
     X          (MEMBR(I2)+MEMBR(J2))
 150     CONTINUE
C
         IF (I2.GT.K) GOTO 160
         IF (DISS(IND1).GE.DMIN) GOTO 160
            DMIN=DISS(IND1)
            JJ=K
 160     CONTINUE
 170  CONTINUE
      MEMBR(I2)=MEMBR(I2)+MEMBR(J2)
      DISNN(I2)=DMIN
      NN(I2)=JJ
C
C  Update list of NNs insofar as this is required.
C
      DO 200 I=1,N-1
         IF (.NOT.FLAG(I)) GOTO 200
         IF (NN(I).EQ.I2) GOTO 180
         IF (NN(I).EQ.J2) GOTO 180
         GOTO 200
 180     CONTINUE
C        (Redetermine NN of I:)
         DMIN=INF
         DO 190 J=I+1,N
            IND=IOFFS(N,I,J)
            IF (.NOT.FLAG(J)) GOTO 190
            IF (I.EQ.J) GOTO 190
            IF (DISS(IND).GE.DMIN) GOTO 190
               DMIN=DISS(IND)
               JJ=J
 190     CONTINUE
         NN(I)=JJ
         DISNN(I)=DMIN
 200  CONTINUE
C
C  Repeat previous steps until N-1 agglomerations carried out.
C
      IF (NCL.GT.1) GOTO 70
C
C
      RETURN
      END
C
C
      FUNCTION IOFFS(N,I,J)
C  Map row I and column J of upper half diagonal symmetric matrix
C  onto vector.
      IOFFS=J+(I-1)*N-(I*(I+1))/2
      RETURN
      END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  HIERARCHICAL CLUSTERING using Minimum Variance Criterion,
C  using the O(N**2) time Nearest Neighbour Chain algorithm.
C
C  Parameters:
C
C  DATA(N,M) :      input data,
C  IA(N), IB(N), CRIT(N) : sequence of agglomerands and
C                   values returned (only locations 1 to N-1
C                   are of interest),
C  MEMBR(N), DISS(N), ICHAIN(N) : used in the routines to store
C                   cluster cardinalities, nearest neighbour
C                   dissimilarities, and the NN-chain.
C  FLAG(N) :        (boolean) used to indicate agglomerable
C                   objects and clusters.
C
C  Reference: Murtagh, Multidimensional Clustering Algorithms,
C             Physica-Verlag, 1985.
C
C-------------------------------------------------------------
      SUBROUTINE HCON2(N,M,DATA,IA,IB,CRIT,MEMBR,DISS,ICHAIN,FLAG)
      REAL    MEMBR(N), DATA(N,M), DISS(N), CRIT(N)
      INTEGER ICHAIN(N), IA(N), IB(N)
      REAL INF
      LOGICAL FLAG(N)
      DATA INF/1.E+25/
C     EQUIVALENCE (ICHAIN(1),IA(1)),(DISS(1),CRIT(1))
C
      DO 150 I=1,N
```

```
         MEMBR(I)=1
         FLAG(I)=.TRUE.
  150 CONTINUE
      NCL=N
      I1=1
C
C  Start the NN-chain:
C
  200 LEN=N
      ICHAIN(LEN)=I1
      DISS(LEN)=INF
C
C  Determine NN of object I1
C
  300 FLAG(I1)=.FALSE.
C
C  Turn off FLAG so that 0 diss. of I1 with self not obtained.
C
      D=DISS(LEN)
      IF (LEN.LT.N) I2=ICHAIN(LEN+1)
C
C  For identical diss.'s, above ensures that RNN will be found.
C
      CALL DETNN(DATA,FLAG,MEMBR,N,M,I1,I2,D)
      FLAG(I1)=.TRUE.
C
C  If LEN = 1 place obj. I2 as second obj. in NN-chain.
C
      IF (LEN.LT.N) GOTO 350
      LEN=LEN-1
      IF (LEN.LT.N-NCL) GOTO 700
      ICHAIN(LEN)=I2
      DISS(LEN)=D
      GOTO 500
C
C  If LEN < N distinguish between having RNN & continuing NN-chain.
C
  350 CONTINUE
      IF (I2.NE.ICHAIN(LEN+1)) GOTO 400
C
C  Have RNN.
C
      NCL=NCL-1
      CALL AGGLOM(I1,I2,D,DATA,MEMBR,FLAG,IA,IB,CRIT,NCL,N,M)
      LEN=LEN+2
      GOTO 500
  400 CONTINUE
C
C  Grow extra link on NN-chain.
C
      IDUM=ICHAIN(LEN+1)
      FLAG(IDUM)=.FALSE.
      LEN=LEN-1
      IF (LEN.LE.N-NCL) GOTO 700
      ICHAIN(LEN)=I2
      DISS(LEN)=D
      GOTO 500
C
C  Select obj. for continuing to grow (or restarting) NN-chain.
C
  500 CONTINUE
      IF (NCL.EQ.1) GOTO 600
      IF (LEN.EQ.N+1) GOTO 550
      I1=ICHAIN(LEN)
      FLAG(I1)=.TRUE.
      IDUM=ICHAIN(LEN+1)
      IF (LEN.LT.N) FLAG(IDUM)=.TRUE.
C
C  Reestablish agglomerability of objects in NN-chain.
C
      GOTO 300
  550 CALL NEXT(FLAG,I1,N)
      GOTO 200
C
  600 CONTINUE
      RETURN
  700 WRITE(6,750)
  750 FORMAT
     X(' ERROR IN NN-CHAIN ROUTINE - INSUFFICIENT CHAIN SPACE'/)
      STOP
      END
C-------------------------------------------------------------
      SUBROUTINE DETNN(DATA,FLAG,MEM,N,M,I1,I2,D)
C
C  Determine a nearest neighbour.
C
      REAL DATA(N,M),MEM(N)
      LOGICAL FLAG(N)
C
      DO 200 I=1,N
         IF (.NOT.FLAG(I)) GOTO 200
         DISS=0.
         DO 100 J=1,M
  100    DISS=DISS+(DATA(I1,J)-DATA(I,J))*(DATA(I1,J)-DATA(I,J))
         DISS=DISS*MEM(I)*MEM(I1)/(MEM(I1)+MEM(I))
         IF (DISS.GE.D) GOTO 200
            D=DISS
            I2=I
  200 CONTINUE
C
      RETURN
      END
C-----------------------------------------------------------
      SUBROUTINE AGGLOM(I1,I2,D,DATA,MEM,FLAG,IA,IB,CRIT,NCL,N,M)
C
C  Carry out an agglomeration.
C
      REAL MEM(N),DATA(N,M),CRIT(N)
      INTEGER IA(N),IB(N)
      LOGICAL FLAG(N)
      INTEGER O1,O2,LB,UB
C
C
      O1=MIN0(I1,I2)
      O2=MAX0(I1,I2)
      DO 100 J=1,M
         DATA(O1,J)=( MEM(O1)*DATA(O1,J)+MEM(O2)*DATA(O2,J)  )
     X            / (MEM(O1)+MEM(O2))
         DATA(O2,J)=DATA(O1,J)
  100 CONTINUE
      NAGGL=N-NCL
      MEM(O1)=MEM(O1)+MEM(O2)
      FLAG(O2)=.FALSE.
C
```

```
C  Keep sorted list of criterion values: find first where
C  new criterion value fits.
C
      I=NAGGL-1
  120 IF (D.GE.CRIT(I)) GOTO 140
      I=I-1
      IF (I.GE.1) GOTO 120
C
C  Arriving here must mean that D > all crit. values found so far.
C
      I=0
  140 CONTINUE
C
C  Now, shift rightwards from I+1 to AGGL-1 to make room for
C  new criterion value.
C
      LB=I+1
      UB=NAGGL-1
      IF (LB.GT.UB) GOTO 180
      J=UB
  160 J1=J+1
      IA(J1)=IA(J)
      IB(J1)=IB(J)
      CRIT(J1)=CRIT(J)
      J=J-1
      IF (J.GE.LB) GOTO 160
  180 CONTINUE
      IA(LB)=O1
      IB(LB)=O2
      CRIT(LB)=D
C
      RETURN
      END
C---------------------------------------------------------
      SUBROUTINE NEXT(FLAG,I1,N)
C
C  Determine next agglomerable object/cluster.
C
      LOGICAL FLAG(N)
C
      NXT=I1+1
      IF (NXT.GT.N) GOTO 150
      DO 100 I=NXT,N
         IF (FLAG(I)) GOTO 500
  100 CONTINUE
  150 DO 200 I=1,I1
         IF (FLAG(I)) GOTO 500
  200 CONTINUE
C
      STOP
C
  500 I1=I
C
      RETURN
      END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Given a HIERARCHIC CLUSTERING, described as a sequence of
C  agglomerations, derive the assignments into clusters for the
C  top LEV-1 levels of the hierarchy.
C  Prepare also the required data for representing the
C  dendrogram of this top part of the hierarchy.
```

```
C
C  Parameters:
C
C  IA, IB, CRIT: vectors of dimension N defining the agglomer-
C                 ations.
C  LEV:          number of clusters in largest partition.
C  HVALS:        vector of dim. LEV, used internally only.
C  ICLASS:       array of cluster assignments; dim. N by LEV.
C  IORDER, CRITVL, HEIGHT: vectors describing the dendrogram,
C                all of dim. LEV.
C
C  N should be greater than LEV.
C
C-------------------------------------------------------------
      SUBROUTINE HCASS(N,IA,IB,CRIT,LEV,ICLASS,HVALS,IORDER,
     X        CRITVL,HEIGHT)
      INTEGER IA(N),IB(N),ICLASS(N,LEV),HVALS(LEV),IORDER(LEV),
     X        HEIGHT(LEV)
      REAL CRIT(N),CRITVL(LEV)
C
C  Pick out the clusters which the N objects belong to,
C  at levels N-2, N-3, ... N-LEV+1 of the hierarchy.
C  The clusters are identified by the lowest seq. no. of
C  their members.
C  There are 2, 3, ... LEV clusters, respectively, for the
C  above levels of the hierarchy.
C
      HVALS(1)=1
      HVALS(2)=IB(N-1)
      LOC=3
      DO 59 I=N-2,N-LEV,-1
         DO 52 J=1,LOC-1
            IF (IA(I).EQ.HVALS(J)) GOTO 54
  52     CONTINUE
         HVALS(LOC)=IA(I)
         LOC=LOC+1
  54     CONTINUE
         DO 56 J=1,LOC-1
            IF (IB(I).EQ.HVALS(J)) GOTO 58
  56     CONTINUE
         HVALS(LOC)=IB(I)
         LOC=LOC+1
  58     CONTINUE
  59  CONTINUE
C
      DO 400 LEVEL=N-LEV,N-2
         DO 200 I=1,N
            ICL=I
            DO 100 ILEV=1,LEVEL
  100       IF (IB(ILEV).EQ.ICL) ICL=IA(ILEV)
            NCL=N-LEVEL
            ICLASS(I,NCL-1)=ICL
  200    CONTINUE
  400  CONTINUE
C
      DO 120 I=1,N
      DO 120 J=1,LEV-1
      DO 110 K=2,LEV
      IF (ICLASS(I,J).NE.HVALS(K)) GOTO 110
         ICLASS(I,J)=K
         GOTO 120
  110 CONTINUE
  120 CONTINUE
C
      WRITE (6,450)
  450 FORMAT(4X,' SEQ NOS 2CL 3CL 4CL 5CL 6CL 7CL 8CL 9CL')
      WRITE (6,470)
  470 FORMAT(4X,' ------- --- --- --- --- --- --- --- --- ----')
      DO 500 I=1,N
      WRITE (6,600) I,(ICLASS(I,J),J=1,8)
  600 FORMAT(I11,8I4)
  500 CONTINUE
C
C  Determine an ordering of the LEV clusters (at level LEV-1)
C  for later representation of the dendrogram.
C  These are stored in IORDER.
C  Determine the associated ordering of the criterion values
C  for the vertical lines in the dendrogram.
C  The ordinal values of these criterion values may be used in
C  preference, and these are stored in HEIGHT.
C  Finally, note that the LEV clusters are renamed so that they
C  have seq. nos. 1 to LEV.
C
      IORDER(1)=IA(N-1)
      IORDER(2)=IB(N-1)
      CRITVL(1)=0.0
      CRITVL(2)=CRIT(N-1)
      HEIGHT(1)=LEV
      HEIGHT(2)=LEV-1
      LOC=2
      DO 700 I=N-2,N-LEV+1,-1
         DO 650 J=1,LOC
            IF (IA(I).EQ.IORDER(J)) THEN
C              Shift rightwards and insert IB(I) beside IORDER(J):
               DO 630 K=LOC+1,J+1,-1
                  IORDER(K)=IORDER(K-1)
                  CRITVL(K)=CRITVL(K-1)
                  HEIGHT(K)=HEIGHT(K-1)
  630          CONTINUE
               IORDER(J+1)=IB(I)
               CRITVL(J+1)=CRIT(I)
               HEIGHT(J+1)=I-(N-LEV)
               LOC=LOC+1
            ENDIF
  650   CONTINUE
  700 CONTINUE
      DO 705 I=1,LEV
         DO 703 J=1,LEV
            IF (HVALS(I).EQ.IORDER(J)) THEN
               IORDER(J)=I
               GOTO 705
            ENDIF
  703    CONTINUE
  705 CONTINUE
C
      RETURN
      END
C++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Construct a DENDROGRAM of the top 8 levels of
C  a HIERARCHIC CLUSTERING.
C
C  Parameters:
C
```

```
C  IORDER, HEIGHT, CRITVL: vectors of length LEV
C          defining the dendrogram.
C          These are: the ordering of objects
C          along the bottom of the dendrogram
C          (IORDER); the height of the vertical
C          above each object, in ordinal values
C          (HEIGHT); and in real values (CRITVL).
C
C  NOTE: these vectors MUST have been set up with
C        LEV = 9 in the prior call to routine
C        HCASS.
C
C-------------------------------------------------
      SUBROUTINE HCDEN(LEV,IORDER,HEIGHT,CRITVL)
      CHARACTER*80 LINE
      INTEGER IORDER(LEV),HEIGHT(LEV)
      REAL CRITVL(LEV)
      INTEGER OUT(27,27)
      INTEGER UP,ACROSS,BLANK
      DATA UP,ACROSS,BLANK/'|','-',' '/
C
C
      DO 10 I=1,27
        DO 10 J=1,27
          OUT(I,J)=BLANK
 10   CONTINUE
C
C
      DO 50 I=3,27,3
         I2=I/3
C
         J2=28-3*HEIGHT(I2)
         J = 27
 20      CONTINUE
            OUT(J,I)=UP
    J = J-1
 IF (J.GE.J2) GOTO 20
C
         K = I
 30      CONTINUE
            I3=INT((K+2)/3)
            IF ( (28-HEIGHT(I3)*3).LT.J2) GOTO 40
            OUT(J2,K)=ACROSS
            K = K-1
         IF (K.GE.3) GOTO 30
 40      CONTINUE
C
 50   CONTINUE
C
C
      IC=3
      DO 90 I=1,27
      IF (I.NE.IC+1) GOTO 80
                   IDUM=IC/3
                   IDUM=9-IDUM
                   DO 60 L=1,9
                      IF (HEIGHT(L).EQ.IDUM) GOTO 70
 60                CONTINUE
 70                IDUM=L
                   WRITE(6,200) CRITVL(IDUM),(OUT(I,J),J=1,27)
                   IC=IC+3
                   GOTO 90
 80                CONTINUE
                   LINE = ' '
                   WRITE(6,210) (OUT(I,J),J=1,27)
 90   CONTINUE
      WRITE(6,250)
      WRITE(6,220)(IORDER(J),J=1,9)
      WRITE(6,250)
      WRITE(6,230)
      WRITE(6,240)
  200 FORMAT(1H ,8X,F12.2,4X,27A1)
  210 FORMAT(1H ,24X,27A1)
  220 FORMAT(1H ,24X,9I3)
  230 FORMAT(1H ,13X,'CRITERION        CLUSTERS 1 TO 9')
  240 FORMAT(1H ,13X,'VALUES.      (TOP 8 LEVELS OF HIERARCHY).')
  250 FORMAT(/)
C
C
      RETURN
      END
```

3.4.3 Program Listing: Partitioning

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Generate a random partition.
C
C  Parameters:
C
C  MEMGP(N)         Group memberships,
C  NG               number of groups,
C  ISEED            seed for random number generator.
C
C  Note: random number generator is machine-dependent.
C
C------------------------------------------------------------
        SUBROUTINE RANDP(N,NG,MEMGP,ISEED)
        DIMENSION MEMGP(N)
C
        DO 100 I = 1, N
           MEMGP(I) = 1
  100   CONTINUE
C
        IF (NG.LE.1.OR.N.LE.1) GOTO 500
        X = 1.0/FLOAT(NG)
        DO 400 I = 1, N
           VAL = RAN(ISEED)
           BNDRY = X
           ICL = 1
  200      IF (ICL.EQ.NG) GOTO 300
           IF (VAL.LT.BNDRY) GOTO 300
           BNDRY = BNDRY + X
           ICL = ICL + 1
           GOTO 200
  300      MEMGP(I) = ICL
  400   CONTINUE
C
  500   CONTINUE
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Optimise the variances of a set of groups, by assigning
C  the objects in groups such that they are minimally distant
C  from group centres.
C
C  Parameters:
C
C  N, M, NG         Numbers of rows, columns, groups,
C  A(N,M)           initial data,
C  MEMGP(N)         group memberships,
C  NGP0             minimum acceptable group cardinality,
C  NUMGP(NG)        cardinalities of groups,
C  GPCEN(NG,M)      group centres,
C  COMP(NG)         compactness values for the groups,
C  CTOT             sum of these compactnesses,
C  IERR             error indicator (should be zero).
C
C  IERR = 1: invalid group number (<1 or >NG), - is number of
C  groups correctly specified?  IERR = 2: a group has < minimum
C  allowed number of members, - reduce the number of groups and
C  try again.
C
```

```
C  Reference: Spaeth, 1985.
C
C--------------------------------------------------------------
        SUBROUTINE MINDST(A,N,M,MEMGP,NGP0,NUMGP,GPCEN,NG,
     X            COMP,CTOT,ITER,IERR)
        DIMENSION       A(N,M), MEMGP(N), NUMGP(NG), GPCEN(NG,M),
     X                  COMP(NG)
C
        BIG = 1.0E+30
        ONE = 0.999
        CMAX = BIG
        ITER = 0
  100   ITER = ITER + 1
        IF (ITER.GT.15) GOTO 500
        CALL GMEANS(A,N,M,MEMGP,NGP0,NUMGP,GPCEN,NG,IERR)
        CALL COMPCT(A,N,M,NG,MEMGP,GPCEN,COMP,CTOT)
        IF (IERR.NE.0) GOTO 500
        IF (NG.LE.1) GOTO 500
        IF (CTOT.GE.CMAX) GOTO 500
        CMAX = CTOT*ONE
        DO 400 I = 1, N
           X = BIG
           DO 300 K = 1, NG
              Y = 0.0
              DO 200 J = 1, M
                 DIFF = GPCEN(K,J) - A(I,J)
                 Y = Y + DIFF*DIFF
  200         CONTINUE
              IF (Y.GE.X) GOTO 300
              X = Y
              ICL = K
  300      CONTINUE
           MEMGP(I) = ICL
  400   CONTINUE
        GOTO 100
C
  500   RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Optimise the variances of a set of groups, by exchanging
C  the objects between groups such that they are minimally
C  distant from group centres.
C
C  Parameters:
C
C  N, M, NG         Numbers of rows, columns, groups,
C  A(N,M)           initial data,
C  MEMGP(N)         group memberships,
C  NGP0             minimum acceptable group cardinality,
C  NUMGP(NG)        cardinalities of groups,
C  GPCEN(NG,M)      group centres,
C  COMP(NG)         compactness values for the groups,
C  CTOT             sum of these compactnesses,
C  IERR             error indicator (should be zero).
C
C  IERR = 1: invalid group number (<1 or >NG), - is number of
C  groups correctly specified?  IERR = 2: a group has < minimum
C  allowed number of members, - reduce the number of groups and
C  try again.
C
C  Reference: Spaeth, 1985.
```

```
C
C---------------------------------------------------------------
        SUBROUTINE EXCH(A,N,M,MEMGP,NGP0,NUMGP,GPCEN,NG,
     X            COMP,CTOT,ITER,IERR)
        DIMENSION       A(N,M), MEMGP(N), NUMGP(NG), GPCEN(NG,M),
     X                  COMP(NG)
C
        BIG = 1.0E+30
        ONE = 0.999
        CALL GMEANS(A,N,M,MEMGP,NGP0,NUMGP,GPCEN,NG,IERR)
        CALL COMPCT(A,N,M,NG,MEMGP,GPCEN,COMP,CTOT)
        IF (IERR.NE.0) GOTO 800
        IF (NG.LE.1) GOTO 800
        ITER = 0
        I = 0
        IS = 0
  100   IS = IS + 1
        IF (IS.GT.N) GOTO 800
  200   I = I + 1
        IF (I.LE.N) GOTO 300
        ITER = ITER + 1
        IF (ITER.GT.15) GOTO 800
        I = 1
  300   ICL = MEMGP(I)
        NUM = NUMGP(ICL)
        IF (NUM.LE.NGP0) GOTO 100
        V = NUM
        EQ = BIG
        DO 600 K = 1, NG
           X = 0.0
           DO 400 J = 1, M
              DIFF = GPCEN(K,J) - A(I,J)
              X = X + DIFF*DIFF
  400      CONTINUE
           IF (K.NE.ICL) GOTO 500
           FRAC1 = V/(V-1.0)
           EP = X*FRAC1
           GOTO 600
  500      FRAC2 = NUMGP(K)
           FRAC = FRAC2/(FRAC2+1.0)
           EK = FRAC*X
           IF (EK.GE.EQ) GOTO 600
           EQ = EK
           IQ = K
           W = FRAC2
  600   CONTINUE
        IF (EQ.GE.EP*ONE) GOTO 100
        IS = 0
        COMP(ICL) = COMP(ICL) - EP
        COMP(IQ) = COMP(IQ) + EQ
        CTOT = CTOT - EP + EQ
        FRAC1 = 1.0/(V-1.0)
        FRAC2 = 1.0/(W+1.0)
        DO 700 J = 1, M
           VAL = A(I,J)
           GPCEN(ICL,J) = (V*GPCEN(ICL,J)-VAL)*FRAC1
           GPCEN(IQ,J) = (W*GPCEN(IQ,J)+VAL)*FRAC2
  700   CONTINUE
        MEMGP(I) = IQ
        NUMGP(ICL) = NUM - 1
        NUMGP(IQ) = NUMGP(IQ) + 1
        GOTO 200
```

```
C
  800   CONTINUE
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Standardize to zero mean and unit standard deviation.
C
C  Parameters:
C
C  N, M, NG         Numbers of rows, columns, groups,
C  A(N,M)           initial data, replaced by standardized values.
C
C------------------------------------------------------------
        SUBROUTINE STND(A,N,M)
        DIMENSION  A(N,M)
C
        DO 500 J = 1, M
           X = 0.0
           DO 100 I = 1, N
              X = X + A(I,J)
  100      CONTINUE
           XBAR = X/FLOAT(N)
           X = 0.0
           DO 200 I = 1, N
              DIFF = A(I,J) - XBAR
              X = X + DIFF*DIFF
  200      CONTINUE
           IF (X.LE.0.0) X = 1.0
           X = 1.0/SQRT(X)
           DO 300 I = 1, N
              A(I,J) = X*(A(I,J)-XBAR)
  300      CONTINUE
  500   CONTINUE
C
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Determine means of the groups.
C
C  Parameters:
C
C  N, M, NG         Numbers of rows, columns, groups,
C  A(N,M)           initial data,
C  MEMGP(N)         group memberships,
C  NGP0             minimum acceptable group cardinality,
C  NUMGP(NG)        cardinalities of groups,
C  GPCEN(NG,M)      group centres,
C  IERR             error indicator (should be zero).
C
C-------------------------------------------------------------
        SUBROUTINE GMEANS(A,N,M,MEMGP,NGP0,NUMGP,GPCEN,NG,IERR)
        DIMENSION       A(N,M), MEMGP(N), NUMGP(NG), GPCEN(NG,M)
C
        DO 200 K = 1, NG
           NUMGP(K) = 0
           DO 100 J = 1, M
              GPCEN(K,J) = 0.0
  100      CONTINUE
  200   CONTINUE
C
```

```
        DO 500 I = 1, N
           ICL = MEMGP(I)
           IF (ICL.GE.1.AND.ICL.LE.NG) GOTO 300
              IERR = 1
              RETURN
  300      CONTINUE
           NUMGP(ICL) = NUMGP(ICL) + 1
           DO 400 J = 1, M
              GPCEN(ICL,J) = GPCEN(ICL,J) + A(I,J)
  400      CONTINUE
  500   CONTINUE
C
        DO 800 K = 1, NG
           NUM = NUMGP(K)
           IF (NUM.GE.NGP0) GOTO 600
              IERR = 2
              RETURN
  600      CONTINUE
           X = 1.0/FLOAT(NUM)
           DO 700 J = 1, M
              GPCEN(K,J) = GPCEN(K,J)*X
  700      CONTINUE
  800   CONTINUE
C
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Determine compactness of the groups (i.e. their variances).
C
C  Parameters:
C
C  N, M, NG         Numbers of rows, columns, groups,
C  A(N,M)           initial data,
C  MEMGP(N)         group memberships,
C  GPCEN(NG,M)      group centres,
C  COMP(NG)         variances of groups (output),
C  CTOT             sum of these variances.
C
C--------------------------------------------------------------
        SUBROUTINE COMPCT(A,N,M,NG,MEMGP,GPCEN,COMP,CTOT)
        DIMENSION       A(N,M), MEMGP(N), GPCEN(NG,M), COMP(NG)
C
        CTOT = 0.0
        DO 100 K = 1, NG
           COMP(K) = 0.0
  100   CONTINUE
C
        DO 300 I = 1, N
           ICL = MEMGP(I)
           X = 0.0
           DO 200 J = 1, M
              DIFF = GPCEN(ICL,J) - A(I,J)
              X = X + DIFF*DIFF
  200      CONTINUE
           COMP(ICL) = COMP(ICL) + X
           CTOT = CTOT + X
  300   CONTINUE
C
        RETURN
        END
```

3.4.4 Input Data

The input data is identical to that described in the Principal Components Analysis chapter.

3.4.5 Sample Output

For the minimum variance criterion, we find the following upper dendrogram result. Note that class numbers are purely sequence numbers.

```
SEQ NOS 2CL 3CL 4CL 5CL 6CL 7CL 8CL 9CL
------- --- --- --- --- --- --- --- ---
      1   1   1   1   1   1   1   1   1
      2   1   1   1   1   1   1   1   1
      3   1   1   1   1   1   1   1   1
      4   1   1   1   1   1   1   1   1
      5   1   1   1   1   1   1   1   1
      6   1   1   1   1   1   1   1   1
      7   1   1   1   1   1   7   7   7
      8   1   1   1   1   1   7   7   7
      9   1   1   1   1   1   7   7   9
     10   1   1   1   1   1   7   7   9
     11   1   1   4   4   4   4   4   4
     12   1   1   4   4   4   4   4   4
     13   1   1   4   4   4   4   8   8
     14   1   1   4   4   4   4   8   8
     15   1   1   4   5   5   5   5   5
     16   2   2   2   2   2   2   2   2
     17   2   2   2   2   6   6   6   6
     18   2   3   3   3   3   3   3   3
```

```
                -
                |
                |
2432849.25      |------------------
                |                 |
                |                 |
 608270.06      |                 |------
                |                 |     |
                |                 |     |
 196383.22      |---------        |     |
                |        |        |     |
                |        |        |     |
  30684.13      |        |------  |     |
                |        |     |  |     |
                |        |     |  |     |
  30549.01      |        |     |  |---  |
                |        |     |  |  |  |
                |        |     |  |  |  |
  30261.67      |---     |     |  |  |  |
                |  |     |     |  |  |  |
                |  |     |     |  |  |  |
   7236.88      |  |     |---  |  |  |  |
                |  |     |  |  |  |  |  |
                |  |     |  |  |  |  |  |
   3428.06      |  |---  |  |  |  |  |  |
                |  |  |  |  |  |  |  |  |
                |  |  |  |  |  |  |  |  |

                1  7  9  4  8  5  2  6  3

   CRITERION        CLUSTERS 1 TO 9
   VALUES.      (TOP 8 LEVELS OF HIERARCHY).
```

The following dendrogram output is obtained for the single link method.

```
SEQ NOS 2CL 3CL 4CL 5CL 6CL 7CL 8CL 9CL
------- --- --- --- --- --- --- --- --- ----
     1   1   1   1   1   1   1   1   1
     2   1   1   1   1   1   1   1   1
     3   1   1   1   1   1   1   1   1
     4   1   1   1   1   1   1   1   1
     5   1   1   1   1   1   1   1   1
     6   1   1   1   1   1   1   1   1
     7   1   1   1   1   1   1   1   9
     8   1   1   1   1   1   1   1   9
     9   1   1   1   1   1   1   1   9
    10   1   1   1   1   1   1   1   9
    11   1   1   1   1   1   7   7   7
    12   1   1   1   1   1   7   7   7
    13   1   1   1   1   1   7   8   8
    14   1   1   1   1   6   6   6   6
    15   1   1   1   5   5   5   5   5
```

```
16   1   3   3   3   3   3   3   3
17   1   3   4   4   4   4   4   4
18   2   2   2   2   2   2   2   2
                  -
                  |
                  |
   691853.06      |------------------------
                  |                       |
                  |                       |
    64923.62      |------------------     |
                  |                 |     |
                  |                 |     |
    61098.02      |                 |---  |
                  |                 |  |  |
                  |                 |  |  |
    20083.52      |---------------  |  |  |
                  |              |  |  |  |
                  |              |  |  |  |
     2809.86      |------------  |  |  |  |
                  |           |  |  |  |  |
                  |           |  |  |  |  |
     2565.14      |------     |  |  |  |  |
                  |     |     |  |  |  |  |
                  |     |     |  |  |  |  |
     2539.18      |     |---  |  |  |  |  |
                  |     |  |  |  |  |  |  |
                  |     |  |  |  |  |  |  |
     1519.50      |---  |  |  |  |  |  |  |
                  |  |  |  |  |  |  |  |  |
                  |  |  |  |  |  |  |  |  |

                  1  9  7  8  6  5  3  4  2

     CRITERION        CLUSTERS 1 TO 9
     VALUES.      (TOP 8 LEVELS OF HIERARCHY).
```

For the minimum distance partitioning method, with two classes requested, a solution was found with sum of variances equal to 914415.4375, after 5 iterations. This solution assigned the first 15 objects to class 1 and the final 3 objects to class 2.

For the exchange method, and again with two classes requested, the sum of variances was 906848.4375 after 2 iterations. For this solution, only the final two objects comprised class 2.

For three classes, and the exchange method, the sum of variances was 297641.7813 after 1 iteration. The classes were: object 18; objects 15, 16, 17; and all remaining objects.

3.4.6 Java Application

For some core functionality, we use mathematical class libraries available from Visual Numerics. Therefore this must be on your system before you use the following program. Of course the Java Development Kit, JDK, must also be available. We used version 1.1.6 of the JDK.

```
import VisualNumerics.math.*;
import java.text.*;

/*

Carry out pairwise agglomerations.  For n items, therefore there
are n-1 agglomerations.  Represent the cluster labels is an nxn
cluster label matrix.  Column no. n will be the singleton labels,
1 to n.  Column no. n-1 will have n-1 unique values (or label sequence
numbers).  Column no. n-2 will have n-2 unique values.  Column no. 1
will have the value 1 only, implying that all n items are in one
cluster.

ClustMat is our agglomeration "engine".  It looks after labeling
only, and is independent of any agglomerative clustering criterion.

Other utility methods:

Dissim                ... calculate dissimilarity matrix
getNNs                ... get nearest neighbors and associated
                          nearest neighbor dissimilarities
getSpaces             ... helping in output formating
printMatrix           ... print matrix of doubles
printMatrixI          ... print matrix of integers
printVect             ... print vector of doubles
printVectI            ... print vector of integers

main does the following:

1. Generates data.  (To be changed later to arbitrary input data streams)
2. Calculate pairwise dissimilarities, and determines nearest neighbors
   and corresponding dissimilarities.
3. Determines the closest nearest neighbors.
4. Carries out an agglomeration in ClustMat.
5. Updates the  pairwise dissimilarity matrix, and then, on the basis
   of this, the nearest neighbors, and the nearest neighbor
   dissimilarities.
6. Repeats while no. of clusters is greater than 2.

Note how 0 and 999.0 values are used in, resp., the nearest neighors and
nearest neighbor dissimilarities, to indicate when items are processed and
no longer exist as singletons.

Step 5 here determines the agglomerative clustering criterion.  We are
currently using the single link method only.

F. Murtagh, f.murtagh@qub.ac.uk, Nov. 1999

*/


public class HCL
{
    public static final double MAXVAL = 1.0e8;

    // Calculate dissimilarity n x n array
    public static double[][] Dissim(int nrow, int ncol, double[][] A)
    {
    // Adiss will contain the dissimilarity data to be returned
    double[][] Adiss = new double[nrow][nrow];
    for (int i1 = 0; i1 < nrow; i1++)
```

```
        {
           for (int i2 = 0; i2 < nrow; i2++)
               {
                  Adiss[i1][i2] = 0.0;
                  for (int j = 0; j < ncol; j++)
                      {
                          Adiss[i1][i2] = Adiss[i1][i2] +
                          (A[i1][j] - A[i2][j])*(A[i1][j] - A[i2][j]);
      }
                  Adiss[i1][i2] = Math.sqrt(Adiss[i1][i2]);
       }
}
        return Adiss;
    }


    // Get NNs and NN dissimilarities
    public static void getNNs(int nrow, double[][] diss,
           int[] nn, double[]nndiss)
    {
    // nn and nndiss will contain data to be returned

    int minobs = -1;
    for (int i1 = 0; i1 < nrow; i1++)
        {
           minobs = -1;
           double mindist = 999.0;
           for (int i2 = 0; i2 < nrow; i2++)
               {
   if ((diss[i1][i2] < mindist) && (i1 != i2)) {
                      mindist = diss[i1][i2];
                      minobs = i2;
                   }
       }
            nn[i1] = minobs + 1;
            nndiss[i1] = mindist;
}
    // Return type void => no return stmt
    }

    // Update cluster structure matrix
    public static void ClustMat(int nrow, int[][] clusters,
           int clust1, int clust2, int ncl)
    {
        // If either clust* is not initialized, then we must init. clusters
if ((clust1 == 0) || (clust2 == 0)) {
           for (int j = 0; j < nrow; j++) {
               for (int i = 0; i < nrow; i++) {
                   clusters[i][j] = 0;
       }
   }
           System.out.println("check on ncl:");
           System.out.println(ncl);
   // Adjust for 0-sequencing in extreme right-hand term.
           for (int i = 0; i < nrow; i++) clusters[i][ncl-1] = i + 1;
           return;
}

        // For some agglomeration, we are told that label clust1 and
        // label clust2, among all labels in col. ncl-1 (ncl ranges over
        // 0 thru nrow-1, are to be agglomerated
      int ncl1 = ncl - 1;
      for (int i = 0; i < nrow; i++) {
          clusters[i][ncl1] = clusters[i][ncl];
          if (clusters[i][ncl1] == clust2) clusters[i][ncl1] = clust1;
      }
      // Return type void => no return stmt
  }

  // Little method for helping in output formating
  public static String getSpaces(int n) {
  StringBuffer sb = new StringBuffer(n);
  for (int i = 0; i < n; i++) sb.append(' ');
  return sb.toString();
  }

  // Utility for printing a matrix
  public static void printMatrix(int n1, int n2, double[][] m)
  {
      // Some definitions for handling output formating
    NumberFormat myFormat = NumberFormat.getNumberInstance();
    FieldPosition fp = new FieldPosition(NumberFormat.INTEGER_FIELD);
    myFormat.setMaximumIntegerDigits(5);
    myFormat.setMaximumFractionDigits(4);
    myFormat.setMinimumFractionDigits(4);
    for (int i=0; i<n1; i++)
        {
            // Print each row, elements separated by spaces
            for (int j=0; j<n2; j++)
                // Following unfortunately doesn't format at all
                //                  System.out.print(m[i][j] + "  ");
                {
                String valString = myFormat.format(
                     m[i][j], new StringBuffer(), fp).toString();
                valString = getSpaces(5 - fp.getEndIndex()) + valString;
                System.out.print(valString);
                }
            // Start a new line at the end of a row
            System.out.println();
        }
    // Leave a gap after the entire matrix
    System.out.println();
  }

  // Utility for printing an integer matrix (no operater overloading!)
  public static void printMatrixI(int n1, int n2, int[][] m)
  {
      // Some definitions for handling output formating
    NumberFormat myFormat = NumberFormat.getNumberInstance();
    FieldPosition fp = new FieldPosition(NumberFormat.INTEGER_FIELD);
    myFormat.setMaximumIntegerDigits(5);
    for (int i=0; i<n1; i++)
        {
            // Print each row, elements separated by spaces
            for (int j=0; j<n2; j++)
                // Following unfortunately doesn't format at all
                //                  System.out.print(m[i][j] + "  ");
                {
                String valString = myFormat.format(
                     m[i][j], new StringBuffer(), fp).toString();
// 4 character locations per integer number
                valString = getSpaces(4 - fp.getEndIndex()) + valString;
                System.out.print(valString);
```

```
              }
          // Start a new line at the end of a row
          System.out.println();
      }
  // Leave a gap after the entire matrix
  System.out.println();
}

// Utility for printing a vector
public static void printVect(double[] m)
{
    // Some definitions for handling output formating
  NumberFormat myFormat = NumberFormat.getNumberInstance();
  FieldPosition fp = new FieldPosition(NumberFormat.INTEGER_FIELD);
  myFormat.setMaximumIntegerDigits(5);
  myFormat.setMaximumFractionDigits(4);
  myFormat.setMinimumFractionDigits(4);
  int len = m.length;
  for (int i=0; i<len; i++)
      {
          // Following would be nice, but doesn't format adequately
          //                  System.out.print(m[i] + "  ");
         String valString = myFormat.format(
               m[i], new StringBuffer(), fp).toString();
         valString = getSpaces(4 - fp.getEndIndex()) + valString;
               System.out.print(valString);
      }
      // Start a new line at the row end
      System.out.println();
  // Leave a gap after the entire vector
  System.out.println();
}

// Utility for printing an integer vector (no operator overloading!)
public static void printVectI(int[] m)
{
    // Some definitions for handling output formating
  NumberFormat myFormat = NumberFormat.getNumberInstance();
  FieldPosition fp = new FieldPosition(NumberFormat.INTEGER_FIELD);
  myFormat.setMaximumIntegerDigits(4);
  int len = m.length;
  for (int i=0; i<len; i++)
      {
          // Following would be nice, but doesn't format adequately
          //                  System.out.print(m[i] + "  ");
         String valString = myFormat.format(
               m[i], new StringBuffer(), fp).toString();
         valString = getSpaces(3 - fp.getEndIndex()) + valString;
               System.out.print(valString);
      }
      // Start a new line at the row end
      System.out.println();
  // Leave a gap after the entire vector
  System.out.println();
}

// The main method contains the body of the program
public static void main(String[] argv)
{
    // Define dimensions of the matrix we'll use
    int nrow = 18;
    int ncol = 16;
```

```
        int ncl  = nrow;   // Initial number of clusters
        int[][] clusters = new int[nrow][nrow];  // cluster label matrix
        int[] nn = new int[nrow];                // cluster nearest neigh's
        double[] nndiss = new double[nrow];      // nearest neigh diss's
        int minobs = -1;

        // Define the matrix that we are going to use
        double[][] A = {
   {   3.0,   3.0,   3.0,   3.0,   3.0,   3.0,  35.0,  45.0,
      53.0,  55.0,  58.0, 113.0, 113.0,  86.0,  67.0,  90.0},
   {   3.5,   3.5,   4.0,   4.0,   4.5,   4.5,  46.0,  59.0,
      63.0,  58.0,  58.0, 125.0, 126.0, 110.0,  78.0,  97.0},
   {   4.0,   4.0,   4.5,   4.5,   5.0,   5.0,  48.0,  60.0,
      68.0,  65.0,  65.0, 123.0, 123.0, 117.0,  87.0, 108.0},
   {   5.0,   5.0,   5.0,   5.5,   5.5,   5.5,  46.0,  63.0,
      70.0,  64.0,  63.0, 116.0, 119.0, 115.0,  97.0, 112.0},
   {   6.0,   6.0,   6.0,   6.0,   6.5,   6.5,  51.0,  69.0,
      77.0,  70.0,  71.0, 120.0, 122.0, 122.0,  96.0, 123.0},
   {  11.0,  11.0,  11.0,  11.0,  11.0,  11.0,  64.0,  75.0,
      81.0,  79.0,  79.0, 112.0, 114.0, 113.0,  98.0, 115.0},
   {  20.0,  20.0,  20.0,  20.0,  20.0,  20.0,  76.0,  86.0,
      93.0,  92.0,  91.0, 104.0, 104.5, 107.0,  97.5, 104.0},
   {  30.0,  30.0,  30.0,  30.0,  30.1,  30.2,  84.0,  96.0,
      98.0,  99.0,  96.0, 101.0, 102.0,  99.0,  94.0,  99.0},
   {  30.0,  33.4,  36.8,  40.0,  43.0,  45.6, 100.0, 106.0,
     106.0, 108.0, 101.0,  99.0,  98.0,  99.0,  95.0,  95.0},
   {  42.0,  44.0,  46.0,  48.0,  50.0,  51.0, 109.0, 111.0,
     110.0, 110.0, 103.0,  95.5,  95.5,  95.0,  92.5,  92.0},
   {  60.0,  61.7,  63.5,  65.5,  67.3,  69.2, 122.0, 124.0,
     124.0, 121.0, 103.0,  93.2,  92.5,  92.2,  90.0,  90.8},
   {  70.0,  70.1,  70.2,  70.3,  70.4,  70.5, 137.0, 132.0,
     134.0, 128.0, 101.0,  91.7,  90.2,  88.8,  87.3,  85.8},
   {  78.0,  77.6,  77.2,  76.8,  76.4,  76.0, 167.0, 159.0,
     152.0, 144.0, 103.0,  89.8,  87.7,  85.7,  83.7,  81.8},
   {  98.9,  97.8,  96.7,  95.5,  94.3,  93.2, 183.0, 172.0,
     162.0, 152.0, 102.0,  87.5,  85.3,  83.3,  81.3,  79.3},
   { 160.0, 157.0, 155.0, 152.0, 149.0, 147.0, 186.0, 175.0,
     165.0, 156.0, 120.0,  87.0,  84.9,  82.8,  80.8,  79.0},
   { 272.0, 266.0, 260.0, 254.0, 248.0, 242.0, 192.0, 182.0,
     170.0, 159.0, 131.0,  88.0,  85.8,  83.7,  81.6,  79.6},
   { 382.0, 372.0, 362.0, 352.0, 343.0, 333.0, 205.0, 192.0,
     178.0, 166.0, 138.0,  86.2,  84.0,  82.0,  79.8,  77.5},
   { 770.0, 740.0, 710.0, 680.0, 650.0, 618.0, 226.0, 207.0,
     195.0, 180.0, 160.0,  82.9,  80.2,  77.7,  75.2,  72.7}
        };

        // Print it out
        System.out.println("A is our input matrix:");
        printMatrix(nrow, ncol, A);

        // Get dissimilarities
        double[][] diss = Dissim(nrow, ncol, A);
        // Print it out
        System.out.println("Dissimilarity matrix for analysis:");
        printMatrix(nrow, nrow, diss);

// Get nearest neighbors and nearest neighbor dissimilarities
        getNNs(nrow, diss, nn, nndiss);
System.out.println("Nearest neighbors:");
printVectI(nn);
        System.out.println("Nearest neighbors dissimilarities:");
        printVect(nndiss);
        // Get closest neighbors, using nndiss, followed by nn
        int clust1 = 0;
        int clust2 = 0;
        int cl1    = 0;
        int cl2    = 0;
// Call to initialize
        ncl = nrow;
        ClustMat(nrow, clusters, clust1, clust2, ncl);
        System.out.println("No. of clusters:  " + ncl +
                   "    Cluster label matrix:");
        printMatrixI(nrow, nrow, clusters);


        do
    {

        minobs = -1;
        double mindist = MAXVAL;
    for (int i = 0; i < ncl; i++) {
                if (nndiss[i] < mindist) {
                   mindist = nndiss[i];
                   minobs = i;
}
    }
    // minobs is one cluster label, the other is nn[minobs]
    // Adjust for 0-sequencing
       if (minobs < nn[minobs]) {
               clust1 = minobs + 1;
               clust2 = nn[minobs];
            }
       if (minobs > nn[minobs]) {
               clust2 = minobs + 1;
               clust1 = nn[minobs];
            }
    // Now we have clust1 < clust2, and we'll agglomerate
    System.out.println("clust #1: "  + clust1 +
                               ";  clust #2: "  + clust2 +
                               "; # clusters left: " + ncl);

    // Now we will carry out an agglomeration
            ncl = ncl - 1;
            ClustMat(nrow, clusters, clust1, clust2, ncl);
    System.out.println("#clusters left: " + ncl +
                               "; cluster label matrix: ");

            printMatrixI(nrow, nrow, clusters);

    // Update the following: diss, nndiss
            // Strategy:
            // nn[clust2] ceases to exist; similarly nndiss[clust2]
            // ... for all occurrences of clust2
            // nn[clust1] must be updated, as must nndiss[clust1]
            // Only other change is for any nn[i] such that nn[i] =
            // ... clust1 or clust2; this must be updated.

            // 1er volet de notre strategie, update diss

            cl1 = clust1 - 1;
            cl2 = clust2 - 1;
            for (int i = 0; i < nrow; i++) {
// Slink, to begin with
```

```
                diss[cl1][i] = Math.min(diss[cl1][i], diss[cl2][i]);
                diss[i][cl1] = diss[cl1][i];
    }
    // Cluster label clust2 is knocked out
            for (int i = 0; i < nrow; i++) {
// Unweighted average distance, to begin with
                diss[cl2][i] = 9999.0;
                diss[i][cl2] = diss[cl2][i];
    }
    /*
            System.out.println("Updated diss. matrix for analysis:");
            printMatrix(nrow, nrow, diss);
    */



// Get nearest neighbors and nearest neighbor dissimilarities
        getNNs(nrow, diss, nn, nndiss);
System.out.println("Nearest neighbors of items 1, 2, ... n:");
printVectI(nn);
        System.out.println("Corresponding nearest neighbors dissimilarities:");
        printVect(nndiss);

    }
            while (ncl > 1);

    }
}
```

Byte code is created for this program, HCL.java, by the command: javac HCL.java. The program is then run as follows: java HCL.

Output for the single link method (all that is currently implemented) is as follows.

```
A is our input matrix:
    3.0000    3.0000    3.0000    3.0000    3.0000    3.0000   35.0000
   45.0000   53.0000   55.0000   58.0000  113.0000  113.0000   86.0000
   67.0000   90.0000
    3.5000    3.5000    4.0000    4.0000    4.5000    4.5000   46.0000
   59.0000   63.0000   58.0000   58.0000  125.0000  126.0000  110.0000
   78.0000   97.0000
    4.0000    4.0000    4.5000    4.5000    5.0000    5.0000   48.0000
   60.0000   68.0000   65.0000   65.0000  123.0000  123.0000  117.0000
   87.0000  108.0000
    5.0000    5.0000    5.0000    5.5000    5.5000    5.5000   46.0000
   63.0000   70.0000   64.0000   63.0000  116.0000  119.0000  115.0000
   97.0000  112.0000
    6.0000    6.0000    6.0000    6.0000    6.5000    6.5000   51.0000
   69.0000   77.0000   70.0000   71.0000  120.0000  122.0000  122.0000
   96.0000  123.0000
   11.0000   11.0000   11.0000   11.0000   11.0000   11.0000   64.0000
   75.0000   81.0000   79.0000   79.0000  112.0000  114.0000  113.0000
   98.0000  115.0000
   20.0000   20.0000   20.0000   20.0000   20.0000   20.0000   76.0000
   86.0000   93.0000   92.0000   91.0000  104.0000  104.5000  107.0000
   97.5000  104.0000
   30.0000   30.0000   30.0000   30.0000   30.1000   30.2000   84.0000
   96.0000   98.0000   99.0000   96.0000  101.0000  102.0000   99.0000
   94.0000   99.0000
   30.0000   33.4000   36.8000   40.0000   43.0000   45.6000  100.0000
  106.0000  106.0000  108.0000  101.0000   99.0000   98.0000   99.0000
   95.0000   95.0000
   42.0000   44.0000   46.0000   48.0000   50.0000   51.0000  109.0000
  111.0000  110.0000  110.0000  103.0000   95.5000   95.5000   95.0000
   92.5000   92.0000
   60.0000   61.7000   63.5000   65.5000   67.3000   69.2000  122.0000
  124.0000  124.0000  121.0000  103.0000   93.2000   92.5000   92.2000
   90.0000   90.8000
   70.0000   70.1000   70.2000   70.3000   70.4000   70.5000  137.0000
  132.0000  134.0000  128.0000  101.0000   91.7000   90.2000   88.8000
   87.3000   85.8000
   78.0000   77.6000   77.2000   76.8000   76.4000   76.0000  167.0000
  159.0000  152.0000  144.0000  103.0000   89.8000   87.7000   85.7000
   83.7000   81.8000
   98.9000   97.8000   96.7000   95.5000   94.3000   93.2000  183.0000
  172.0000  162.0000  152.0000  102.0000   87.5000   85.3000   83.3000
   81.3000   79.3000
  160.0000  157.0000  155.0000  152.0000  149.0000  147.0000  186.0000
  175.0000  165.0000  156.0000  120.0000   87.0000   84.9000   82.8000
   80.8000   79.0000
  272.0000  266.0000  260.0000  254.0000  248.0000  242.0000  192.0000
  182.0000  170.0000  159.0000  131.0000   88.0000   85.8000   83.7000
   81.6000   79.6000
  382.0000  372.0000  362.0000  352.0000  343.0000  333.0000  205.0000
  192.0000  178.0000  166.0000  138.0000   86.2000   84.0000   82.0000
   79.8000   77.5000
  770.0000  740.0000  710.0000  680.0000  650.0000  618.0000  226.0000
  207.0000  195.0000  180.0000  160.0000   82.9000   80.2000   77.7000
   75.2000   72.7000

Dissimilarity matrix for analysis:
    0.0000   38.6264   51.6478   56.0424   72.2392   79.0127  104.2809
  126.6541  156.1773  176.2605  222.2397  245.4764  290.7754  339.1081
  451.4413  680.1981  917.4608 1731.4171
   38.6264    0.0000   19.8368   29.6100   42.9826   55.9196   87.4786
  112.9971  142.6063  164.0937  210.6672  234.0655  278.4312  327.2640
  441.6547  672.4206  910.7760 1726.5418
   51.6478   19.8368    0.0000   14.5172   24.2899   41.0427   76.5637
  104.4296  134.5484  156.9626  204.1969  228.0536  272.5305  321.8018
  436.8434  668.5701  907.4601 1724.0412
   56.0424   29.6100   14.5172    0.0000   20.2793   36.3834   72.8989
  101.3385  131.7851  154.3033  201.5972  225.6435  270.3477  319.6283
  434.7181  666.4868  905.4252 1722.0917
   72.2392   42.9826   24.2899   20.2793    0.0000   27.9911   65.9773
   94.9613  124.8137  147.7236  194.7878  218.8425  262.7639  312.2335
  428.0790  660.9620  900.4935 1718.1477
   79.0127   55.9196   41.0427   36.3834   27.9911    0.0000   38.9808
   68.7710   99.1340  122.3182  170.6074  194.5941  238.5002  288.5993
  406.4476  642.1338  882.9752 1702.7696
  104.2809   87.4786   76.5637   72.8989   65.9773   38.9808    0.0000
   31.3935   62.1744   85.5468  134.8165  158.7037  202.6244  253.2628
  372.9513  611.8258  854.1655 1676.4621
  126.6541  112.9971  104.4296  101.3385   94.9613   68.7710   31.3935
    0.0000   33.5316   55.2485  104.4500  128.4999  173.1644  223.4924
  342.9308  583.0440  826.0606 1649.6174
  156.1773  142.6063  134.5484  131.7851  124.8137   99.1340   62.1744
   33.5316    0.0000   25.7470   75.2944   99.7116  143.7613  194.3000
  315.5136  558.2442  802.3565 1627.9910
  176.2605  164.0937  156.9626  154.3033  147.7236  122.3182   85.5468
   55.2485   25.7470    0.0000   50.6472   74.9667  120.4122  170.1757
  290.6911  534.1972  778.6850 1604.9461
  222.2397  210.6672  204.1969  201.5972  194.7878  170.6074  134.8165
  104.4500   75.2944   50.6472    0.0000   27.2613   76.0380  122.1726
  240.7587  486.0941  731.3964 1559.1582
  245.4764  234.0655  228.0536  225.6435  218.8425  194.5941  158.7037
  128.4999   99.7116   74.9667   27.2613    0.0000   50.3903   96.1948
  219.2679  467.6714  713.7365 1542.5976
  290.7754  278.4312  272.5305  270.3477  262.7639  238.5002  202.6244
  173.1644  143.7613  120.4122   76.0380   50.3903    0.0000   53.0081
  190.5448  444.3601  691.4116 1522.2374
  339.1081  327.2640  321.8018  319.6283  312.2335  288.5993  253.2628
  223.4924  194.3000  170.1757  122.1726   96.1948   53.0081    0.0000
  141.7163  396.1628  643.0442 1474.1984
  451.4413  441.6547  436.8434  434.7181  428.0790  406.4476  372.9513
  342.9308  315.5136  290.6911  240.7587  219.2679  190.5448  141.7163
    0.0000  254.8011  501.8350 1333.2201
  680.1981  672.4206  668.5701  666.4868  660.9620  642.1338  611.8258
  583.0440  558.2442  534.1972  486.0941  467.6714  444.3601  396.1628
  254.8011    0.0000  247.1801 1078.6593
  917.4608  910.7760  907.4601  905.4252  900.4935  882.9752  854.1655
  826.0606  802.3565  778.6850  731.3964  713.7365  691.4116  643.0442
  501.8350  247.1801    0.0000  831.7770
 1731.4171 1726.5418 1724.0412 1722.0917 1718.1477 1702.7696 1676.4621
 1649.6174 1627.9910 1604.9461 1559.1582 1542.5976 1522.2374 1474.1984
 1333.2201 1078.6593  831.7770    0.0000

Nearest neighbors:
  2  3  4  3  4  5  8  7 10  9 12 11 12 13 14 17 16 17

Nearest neighbors dissimilarities:
  38.6264  19.8368  14.5172  14.5172  20.2793  27.9911  31.3935  31.3935
  25.7470  25.7470  27.2613  27.2613  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770
```

```
check on ncl:
18
No. of clusters:  18    Cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   3
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   4
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   5
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   6
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   7
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   8
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  10
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  11
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  12
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  13
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  14
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  15
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  16
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  18

clust #1: 3;  clust #2: 4; # clusters left: 18
#clusters left: 17; cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   1
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   3   3
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   3   4
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   5   5
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   6   6
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   7   7
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   8   8
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   9   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  10  10
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  11  11
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  12  12
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  13  13
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  14  14
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  15  15
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  16  16
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17  17
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  18  18

Nearest neighbors of items 1, 2, ... n:
  2  3  2  0  3  5  8  7 10  9 12 11 12 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  19.8368  19.8368 999.0000  20.2793  27.9911  31.3935  31.3935
  25.7470  25.7470  27.2613  27.2613  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770
```

```
clust #1: 2;  clust #2: 3; # clusters left: 17
#clusters left: 16; cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   1   1
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   3   3
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   3   4
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   5   5   5
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   6   6   6
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   7   7   7
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   8   8   8
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   9   9   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  10  10  10
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  11  11  11
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  12  12  12
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  13  13  13
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  14  14  14
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  15  15  15
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  16  16  16
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17  17  17
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  18  18  18

Nearest neighbors of items 1, 2, ... n:
  2  5  0  0  2  5  8  7 10  9 12 11 12 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  20.2793 999.0000 999.0000  20.2793  27.9911  31.3935  31.3935
  25.7470  25.7470  27.2613  27.2613  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 2;  clust #2: 5; # clusters left: 16
#clusters left: 15; cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   1   1   1
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   3   3
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   3   4
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   5   5   5
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   6   6   6   6
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   7   7   7   7
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   8   8   8   8
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   9   9   9   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  10  10  10  10
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  11  11  11  11
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  12  12  12  12
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  13  13  13  13
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  14  14  14  14
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  15  15  15  15
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  16  16  16  16
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  17  17  17  17
   0   0   0   0   0   0   0   0   0   0   0   0   0   0  18  18  18  18
```

```
Nearest neighbors of items 1, 2, ... n:
  2  6  0  0  0  2  8  7 10  9 12 11 12 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  27.9911 999.0000 999.0000 999.0000  27.9911  31.3935  31.3935
  25.7470  25.7470  27.2613  27.2613  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 9;  clust #2: 10; # clusters left: 15
#clusters left: 14; cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   0   0   0   1   1   1   1   1
   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2
   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   3   3
   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   3   4
   0   0   0   0   0   0   0   0   0   0   0   0   0   2   2   5   5   5
   0   0   0   0   0   0   0   0   0   0   0   0   0   6   6   6   6   6
   0   0   0   0   0   0   0   0   0   0   0   0   0   7   7   7   7   7
   0   0   0   0   0   0   0   0   0   0   0   0   0   8   8   8   8   8
   0   0   0   0   0   0   0   0   0   0   0   0   0   9   9   9   9   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   9  10  10  10  10
   0   0   0   0   0   0   0   0   0   0   0   0   0  11  11  11  11  11
   0   0   0   0   0   0   0   0   0   0   0   0   0  12  12  12  12  12
   0   0   0   0   0   0   0   0   0   0   0   0   0  13  13  13  13  13
   0   0   0   0   0   0   0   0   0   0   0   0   0  14  14  14  14  14
   0   0   0   0   0   0   0   0   0   0   0   0   0  15  15  15  15  15
   0   0   0   0   0   0   0   0   0   0   0   0   0  16  16  16  16  16
   0   0   0   0   0   0   0   0   0   0   0   0   0  17  17  17  17  17
   0   0   0   0   0   0   0   0   0   0   0   0   0  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
  2  6  0  0  0  2  8  7  8  0 12 11 12 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  27.9911 999.0000 999.0000 999.0000  27.9911  31.3935  31.3935
  33.5316 999.0000  27.2613  27.2613  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 11;  clust #2: 12; # clusters left: 14
#clusters left: 13; cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   0   0   1   1   1   1   1   1
   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2
   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   3   3
   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   3   4
   0   0   0   0   0   0   0   0   0   0   0   0   2   2   2   5   5   5
   0   0   0   0   0   0   0   0   0   0   0   0   6   6   6   6   6   6
   0   0   0   0   0   0   0   0   0   0   0   0   7   7   7   7   7   7
   0   0   0   0   0   0   0   0   0   0   0   0   8   8   8   8   8   8
   0   0   0   0   0   0   0   0   0   0   0   0   9   9   9   9   9   9
   0   0   0   0   0   0   0   0   0   0   0   0   9   9  10  10  10  10
   0   0   0   0   0   0   0   0   0   0   0   0  11  11  11  11  11  11
   0   0   0   0   0   0   0   0   0   0   0   0  11  12  12  12  12  12
   0   0   0   0   0   0   0   0   0   0   0   0  13  13  13  13  13  13
   0   0   0   0   0   0   0   0   0   0   0   0  14  14  14  14  14  14
   0   0   0   0   0   0   0   0   0   0   0   0  15  15  15  15  15  15
   0   0   0   0   0   0   0   0   0   0   0   0  16  16  16  16  16  16
   0   0   0   0   0   0   0   0   0   0   0   0  17  17  17  17  17  17
   0   0   0   0   0   0   0   0   0   0   0   0  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
  2  6  0  0  0  2  8  7  8  0 13  0 11 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  27.9911 999.0000 999.0000 999.0000  27.9911  31.3935  31.3935
  33.5316 999.0000  50.3903 999.0000  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 2;  clust #2: 6; # clusters left: 13
#clusters left: 12; cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   0   1   1   1   1   1   1   1
   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   2
   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   3   3
   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   3   4
   0   0   0   0   0   0   0   0   0   0   0   2   2   2   2   5   5   5
   0   0   0   0   0   0   0   0   0   0   0   2   6   6   6   6   6   6
   0   0   0   0   0   0   0   0   0   0   0   7   7   7   7   7   7   7
   0   0   0   0   0   0   0   0   0   0   0   8   8   8   8   8   8   8
   0   0   0   0   0   0   0   0   0   0   0   9   9   9   9   9   9   9
   0   0   0   0   0   0   0   0   0   0   0   9   9   9  10  10  10  10
   0   0   0   0   0   0   0   0   0   0   0  11  11  11  11  11  11  11
   0   0   0   0   0   0   0   0   0   0   0  11  11  12  12  12  12  12
   0   0   0   0   0   0   0   0   0   0   0  13  13  13  13  13  13  13
   0   0   0   0   0   0   0   0   0   0   0  14  14  14  14  14  14  14
   0   0   0   0   0   0   0   0   0   0   0  15  15  15  15  15  15  15
   0   0   0   0   0   0   0   0   0   0   0  16  16  16  16  16  16  16
   0   0   0   0   0   0   0   0   0   0   0  17  17  17  17  17  17  17
   0   0   0   0   0   0   0   0   0   0   0  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
  2  1  0  0  0  0  8  7  8  0 13  0 11 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  38.6264 999.0000 999.0000 999.0000 999.0000  31.3935  31.3935
  33.5316 999.0000  50.3903 999.0000  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 7;  clust #2: 8; # clusters left: 12
#clusters left: 11; cluster label matrix:
   0   0   0   0   0   0   0   0   0   0   1   1   1   1   1   1   1   1
   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   2   2
   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   3   3
   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   3   4
   0   0   0   0   0   0   0   0   0   0   2   2   2   2   2   5   5   5
   0   0   0   0   0   0   0   0   0   0   2   2   6   6   6   6   6   6
   0   0   0   0   0   0   0   0   0   0   7   7   7   7   7   7   7   7
   0   0   0   0   0   0   0   0   0   0   7   8   8   8   8   8   8   8
   0   0   0   0   0   0   0   0   0   0   9   9   9   9   9   9   9   9
   0   0   0   0   0   0   0   0   0   0   9   9   9   9  10  10  10  10
   0   0   0   0   0   0   0   0   0   0  11  11  11  11  11  11  11  11
   0   0   0   0   0   0   0   0   0   0  11  11  11  12  12  12  12  12
   0   0   0   0   0   0   0   0   0   0  13  13  13  13  13  13  13  13
   0   0   0   0   0   0   0   0   0   0  14  14  14  14  14  14  14  14
   0   0   0   0   0   0   0   0   0   0  15  15  15  15  15  15  15  15
   0   0   0   0   0   0   0   0   0   0  16  16  16  16  16  16  16  16
   0   0   0   0   0   0   0   0   0   0  17  17  17  17  17  17  17  17
   0   0   0   0   0   0   0   0   0   0  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
  2  1  0  0  0  0  9  0  7  0 13  0 11 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  38.6264 999.0000 999.0000 999.0000 999.0000  33.5316 999.0000
  33.5316 999.0000  50.3903 999.0000  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 7;  clust #2: 9; # clusters left: 11
#clusters left: 10; cluster label matrix:
   0   0   0   0   0   0   0   0   0   1   1   1   1   1   1   1   1   1
   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   2   2   2
   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   2   3   3
   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   2   3   4
   0   0   0   0   0   0   0   0   0   2   2   2   2   2   2   5   5   5
   0   0   0   0   0   0   0   0   0   2   2   2   6   6   6   6   6   6
   0   0   0   0   0   0   0   0   0   7   7   7   7   7   7   7   7   7
   0   0   0   0   0   0   0   0   0   7   7   8   8   8   8   8   8   8
   0   0   0   0   0   0   0   0   0   7   9   9   9   9   9   9   9   9
   0   0   0   0   0   0   0   0   0   7   9   9   9   9  10  10  10  10
   0   0   0   0   0   0   0   0   0  11  11  11  11  11  11  11  11  11
   0   0   0   0   0   0   0   0   0  11  11  11  11  12  12  12  12  12
   0   0   0   0   0   0   0   0   0  13  13  13  13  13  13  13  13  13
   0   0   0   0   0   0   0   0   0  14  14  14  14  14  14  14  14  14
   0   0   0   0   0   0   0   0   0  15  15  15  15  15  15  15  15  15
   0   0   0   0   0   0   0   0   0  16  16  16  16  16  16  16  16  16
   0   0   0   0   0   0   0   0   0  17  17  17  17  17  17  17  17  17
   0   0   0   0   0   0   0   0   0  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
  2  1  0  0  0  0  2  0  0  0 13  0 11 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.6264  38.6264 999.0000 999.0000 999.0000 999.0000  38.9808 999.0000
 999.0000 999.0000  50.3903 999.0000  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 1;  clust #2: 2; # clusters left: 10
#clusters left: 9; cluster label matrix:
   0   0   0   0   0   0   0   0   1   1   1   1   1   1   1   1   1   1
   0   0   0   0   0   0   0   0   1   2   2   2   2   2   2   2   2   2
   0   0   0   0   0   0   0   0   1   2   2   2   2   2   2   2   3   3
   0   0   0   0   0   0   0   0   1   2   2   2   2   2   2   2   3   4
   0   0   0   0   0   0   0   0   1   2   2   2   2   2   2   5   5   5
   0   0   0   0   0   0   0   0   1   2   2   2   6   6   6   6   6   6
   0   0   0   0   0   0   0   0   7   7   7   7   7   7   7   7   7   7
   0   0   0   0   0   0   0   0   7   7   7   8   8   8   8   8   8   8
   0   0   0   0   0   0   0   0   7   7   9   9   9   9   9   9   9   9
   0   0   0   0   0   0   0   0   7   7   9   9   9   9  10  10  10  10
   0   0   0   0   0   0   0   0  11  11  11  11  11  11  11  11  11  11
   0   0   0   0   0   0   0   0  11  11  11  11  11  12  12  12  12  12
   0   0   0   0   0   0   0   0  13  13  13  13  13  13  13  13  13  13
   0   0   0   0   0   0   0   0  14  14  14  14  14  14  14  14  14  14
   0   0   0   0   0   0   0   0  15  15  15  15  15  15  15  15  15  15
   0   0   0   0   0   0   0   0  16  16  16  16  16  16  16  16  16  16
   0   0   0   0   0   0   0   0  17  17  17  17  17  17  17  17  17  17
   0   0   0   0   0   0   0   0  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
  7  0  0  0  0  0  1  0  0  0 13  0 11 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  38.9808 999.0000 999.0000 999.0000 999.0000 999.0000  38.9808 999.0000
 999.0000 999.0000  50.3903 999.0000  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 1;  clust #2: 7; # clusters left: 9
#clusters left: 8; cluster label matrix:
   0   0   0   0   0   0   0   1   1   1   1   1   1   1   1   1   1   1
   0   0   0   0   0   0   0   1   1   2   2   2   2   2   2   2   2   2
   0   0   0   0   0   0   0   1   1   2   2   2   2   2   2   2   3   3
   0   0   0   0   0   0   0   1   1   2   2   2   2   2   2   2   3   4
   0   0   0   0   0   0   0   1   1   2   2   2   2   2   2   5   5   5
   0   0   0   0   0   0   0   1   1   2   2   2   6   6   6   6   6   6
   0   0   0   0   0   0   0   1   7   7   7   7   7   7   7   7   7   7
   0   0   0   0   0   0   0   1   7   7   7   8   8   8   8   8   8   8
   0   0   0   0   0   0   0   1   7   7   9   9   9   9   9   9   9   9
   0   0   0   0   0   0   0   1   7   7   9   9   9   9  10  10  10  10
   0   0   0   0   0   0   0  11  11  11  11  11  11  11  11  11  11  11
   0   0   0   0   0   0   0  11  11  11  11  11  11  12  12  12  12  12
   0   0   0   0   0   0   0  13  13  13  13  13  13  13  13  13  13  13
   0   0   0   0   0   0   0  14  14  14  14  14  14  14  14  14  14  14
   0   0   0   0   0   0   0  15  15  15  15  15  15  15  15  15  15  15
   0   0   0   0   0   0   0  16  16  16  16  16  16  16  16  16  16  16
   0   0   0   0   0   0   0  17  17  17  17  17  17  17  17  17  17  17
   0   0   0   0   0   0   0  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
 11  0  0  0  0  0  0  0  0  0 13  0 11 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  50.6472 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000  50.3903 999.0000  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 1;  clust #2: 11; # clusters left: 8
#clusters left: 7; cluster label matrix:
   0   0   0   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1
   0   0   0   0   0   0   1   1   1   2   2   2   2   2   2   2   2   2
   0   0   0   0   0   0   1   1   1   2   2   2   2   2   2   2   3   3
   0   0   0   0   0   0   1   1   1   2   2   2   2   2   2   2   3   4
   0   0   0   0   0   0   1   1   1   2   2   2   2   2   2   5   5   5
   0   0   0   0   0   0   1   1   1   2   2   2   6   6   6   6   6   6
   0   0   0   0   0   0   1   1   7   7   7   7   7   7   7   7   7   7
   0   0   0   0   0   0   1   1   7   7   7   8   8   8   8   8   8   8
   0   0   0   0   0   0   1   1   7   7   9   9   9   9   9   9   9   9
   0   0   0   0   0   0   1   1   7   7   9   9   9   9  10  10  10  10
   0   0   0   0   0   0   1  11  11  11  11  11  11  11  11  11  11  11
   0   0   0   0   0   0   1  11  11  11  11  11  11  12  12  12  12  12
   0   0   0   0   0   0  13  13  13  13  13  13  13  13  13  13  13  13
   0   0   0   0   0   0  14  14  14  14  14  14  14  14  14  14  14  14
   0   0   0   0   0   0  15  15  15  15  15  15  15  15  15  15  15  15
   0   0   0   0   0   0  16  16  16  16  16  16  16  16  16  16  16  16
   0   0   0   0   0   0  17  17  17  17  17  17  17  17  17  17  17  17
   0   0   0   0   0   0  18  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
 13  0  0  0  0  0  0  0  0  0  0  0  1 13 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  50.3903 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000 999.0000 999.0000  50.3903  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 1;  clust #2: 13; # clusters left: 7
#clusters left: 6; cluster label matrix:
   0   0   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1   1
   0   0   0   0   0   1   1   1   1   2   2   2   2   2   2   2   2   2
   0   0   0   0   0   1   1   1   1   2   2   2   2   2   2   2   3   3
   0   0   0   0   0   1   1   1   1   2   2   2   2   2   2   2   3   4
   0   0   0   0   0   1   1   1   1   2   2   2   2   2   2   5   5   5
   0   0   0   0   0   1   1   1   1   2   2   2   6   6   6   6   6   6
   0   0   0   0   0   1   1   1   7   7   7   7   7   7   7   7   7   7
   0   0   0   0   0   1   1   1   7   7   7   8   8   8   8   8   8   8
   0   0   0   0   0   1   1   1   7   7   9   9   9   9   9   9   9   9
   0   0   0   0   0   1   1   1   7   7   9   9   9   9  10  10  10  10
   0   0   0   0   0   1   1  11  11  11  11  11  11  11  11  11  11  11
   0   0   0   0   0   1   1  11  11  11  11  11  11  12  12  12  12  12
   0   0   0   0   0   1  13  13  13  13  13  13  13  13  13  13  13  13
   0   0   0   0   0  14  14  14  14  14  14  14  14  14  14  14  14  14
   0   0   0   0   0  15  15  15  15  15  15  15  15  15  15  15  15  15
   0   0   0   0   0  16  16  16  16  16  16  16  16  16  16  16  16  16
   0   0   0   0   0  17  17  17  17  17  17  17  17  17  17  17  17  17
   0   0   0   0   0  18  18  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
 14  0  0  0  0  0  0  0  0  0  0  0  0  1 14 17 16 17

Corresponding nearest neighbors dissimilarities:
  53.0081 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000 999.0000 999.0000 999.0000  53.0081 141.7163 247.1801
 247.1801 831.7770

clust #1: 1;  clust #2: 14; # clusters left: 6
#clusters left: 5; cluster label matrix:
   0   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   0   0   0   0   1   1   1   1   1   2   2   2   2   2   2   2   2   2
   0   0   0   0   1   1   1   1   1   2   2   2   2   2   2   2   3   3
   0   0   0   0   1   1   1   1   1   2   2   2   2   2   2   2   3   4
   0   0   0   0   1   1   1   1   1   2   2   2   2   2   2   5   5   5
   0   0   0   0   1   1   1   1   1   2   2   2   6   6   6   6   6   6
   0   0   0   0   1   1   1   1   7   7   7   7   7   7   7   7   7   7
   0   0   0   0   1   1   1   1   7   7   7   8   8   8   8   8   8   8
   0   0   0   0   1   1   1   1   7   7   9   9   9   9   9   9   9   9
   0   0   0   0   1   1   1   1   7   7   9   9   9   9  10  10  10  10
   0   0   0   0   1   1   1  11  11  11  11  11  11  11  11  11  11  11
   0   0   0   0   1   1   1  11  11  11  11  11  11  12  12  12  12  12
   0   0   0   0   1   1  13  13  13  13  13  13  13  13  13  13  13  13
   0   0   0   0   1  14  14  14  14  14  14  14  14  14  14  14  14  14
   0   0   0   0  15  15  15  15  15  15  15  15  15  15  15  15  15  15
   0   0   0   0  16  16  16  16  16  16  16  16  16  16  16  16  16  16
   0   0   0   0  17  17  17  17  17  17  17  17  17  17  17  17  17  17
   0   0   0   0  18  18  18  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
 15  0  0  0  0  0  0  0  0  0  0  0  0  0  1 17 16 17

Corresponding nearest neighbors dissimilarities:
 141.7163 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 141.7163 247.1801
 247.1801 831.7770

clust #1: 1;  clust #2: 15; # clusters left: 5
```

```
#clusters left: 4; cluster label matrix:
   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   0   0   0   1   1   1   1   1   1   2   2   2   2   2   2   2   2   2
   0   0   0   1   1   1   1   1   1   2   2   2   2   2   2   2   3   3
   0   0   0   1   1   1   1   1   1   2   2   2   2   2   2   2   3   4
   0   0   0   1   1   1   1   1   1   2   2   2   2   2   2   5   5   5
   0   0   0   1   1   1   1   1   1   2   2   2   6   6   6   6   6   6
   0   0   0   1   1   1   1   1   7   7   7   7   7   7   7   7   7   7
   0   0   0   1   1   1   1   1   7   7   7   8   8   8   8   8   8   8
   0   0   0   1   1   1   1   1   7   7   9   9   9   9   9   9   9   9
   0   0   0   1   1   1   1   1   7   7   9   9   9   9  10  10  10  10
   0   0   0   1   1   1   1  11  11  11  11  11  11  11  11  11  11  11
   0   0   0   1   1   1   1  11  11  11  11  11  11  12  12  12  12  12
   0   0   0   1   1   1  13  13  13  13  13  13  13  13  13  13  13  13
   0   0   0   1   1  14  14  14  14  14  14  14  14  14  14  14  14  14
   0   0   0   1  15  15  15  15  15  15  15  15  15  15  15  15  15  15
   0   0   0  16  16  16  16  16  16  16  16  16  16  16  16  16  16  16
   0   0   0  17  17  17  17  17  17  17  17  17  17  17  17  17  17  17
   0   0   0  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
 16  0  0  0  0  0  0  0  0  0  0  0  0  0  0 17 16 17

Corresponding nearest neighbors dissimilarities:
 254.8011 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 247.1801
 247.1801 831.7770

clust #1: 1;  clust #2: 16; # clusters left: 4
#clusters left: 3; cluster label matrix:
   0   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   0   0   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2   2
   0   0   1   1   1   1   1   1   1   2   2   2   2   2   2   2   3   3
   0   0   1   1   1   1   1   1   1   2   2   2   2   2   2   2   3   4
   0   0   1   1   1   1   1   1   1   2   2   2   2   2   2   5   5   5
   0   0   1   1   1   1   1   1   1   2   2   2   6   6   6   6   6   6
   0   0   1   1   1   1   1   1   7   7   7   7   7   7   7   7   7   7
   0   0   1   1   1   1   1   1   7   7   7   8   8   8   8   8   8   8
   0   0   1   1   1   1   1   1   7   7   9   9   9   9   9   9   9   9
   0   0   1   1   1   1   1   1   7   7   9   9   9   9  10  10  10  10
   0   0   1   1   1   1   1  11  11  11  11  11  11  11  11  11  11  11
   0   0   1   1   1   1   1  11  11  11  11  11  11  12  12  12  12  12
   0   0   1   1   1   1  13  13  13  13  13  13  13  13  13  13  13  13
   0   0   1   1   1  14  14  14  14  14  14  14  14  14  14  14  14  14
   0   0   1   1  15  15  15  15  15  15  15  15  15  15  15  15  15  15
   0   0   1  16  16  16  16  16  16  16  16  16  16  16  16  16  16  16
   0   0  17  17  17  17  17  17  17  17  17  17  17  17  17  17  17  17
   0   0  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
 17  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1 17

Corresponding nearest neighbors dissimilarities:
 247.1801 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 247.1801 831.7770

clust #1: 1;  clust #2: 17; # clusters left: 3
#clusters left: 2; cluster label matrix:
   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   0   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2   2
   0   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   3   3
   0   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   3   4
   0   1   1   1   1   1   1   1   1   2   2   2   2   2   2   5   5   5
   0   1   1   1   1   1   1   1   1   2   2   2   6   6   6   6   6   6
   0   1   1   1   1   1   1   1   7   7   7   7   7   7   7   7   7   7
   0   1   1   1   1   1   1   1   7   7   7   8   8   8   8   8   8   8
   0   1   1   1   1   1   1   1   7   7   9   9   9   9   9   9   9   9
   0   1   1   1   1   1   1   1   7   7   9   9   9   9  10  10  10  10
   0   1   1   1   1   1   1  11  11  11  11  11  11  11  11  11  11  11
   0   1   1   1   1   1   1  11  11  11  11  11  11  12  12  12  12  12
   0   1   1   1   1   1  13  13  13  13  13  13  13  13  13  13  13  13
   0   1   1   1   1  14  14  14  14  14  14  14  14  14  14  14  14  14
   0   1   1   1  15  15  15  15  15  15  15  15  15  15  15  15  15  15
   0   1   1  16  16  16  16  16  16  16  16  16  16  16  16  16  16  16
   0   1  17  17  17  17  17  17  17  17  17  17  17  17  17  17  17  17
   0  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
 18  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1

Corresponding nearest neighbors dissimilarities:
 831.7770 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 831.7770

clust #1: 1;  clust #2: 18; # clusters left: 2
#clusters left: 1; cluster label matrix:
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2   2
   1   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   3   3
   1   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   3   4
   1   1   1   1   1   1   1   1   1   2   2   2   2   2   2   5   5   5
   1   1   1   1   1   1   1   1   1   2   2   2   6   6   6   6   6   6
   1   1   1   1   1   1   1   1   7   7   7   7   7   7   7   7   7   7
   1   1   1   1   1   1   1   1   7   7   7   8   8   8   8   8   8   8
   1   1   1   1   1   1   1   1   7   7   9   9   9   9   9   9   9   9
   1   1   1   1   1   1   1   1   7   7   9   9   9   9  10  10  10  10
   1   1   1   1   1   1   1  11  11  11  11  11  11  11  11  11  11  11
   1   1   1   1   1   1   1  11  11  11  11  11  11  12  12  12  12  12
   1   1   1   1   1   1  13  13  13  13  13  13  13  13  13  13  13  13
   1   1   1   1   1  14  14  14  14  14  14  14  14  14  14  14  14  14
   1   1   1   1  15  15  15  15  15  15  15  15  15  15  15  15  15  15
   1   1   1  16  16  16  16  16  16  16  16  16  16  16  16  16  16  16
   1   1  17  17  17  17  17  17  17  17  17  17  17  17  17  17  17  17
   1  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18  18

Nearest neighbors of items 1, 2, ... n:
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0

Corresponding nearest neighbors dissimilarities:
 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000 999.0000
 999.0000 999.0000
```

Chapter 4

Discriminant Analysis

4.1 The Problem

Discriminant Analysis may be used for two objectives: either we want to *assess* the adequacy of classification, given the group memberships of the objects under study; or we wish to *assign* objects to one of a number of (known) groups of objects. Discriminant Analysis may thus have a descriptive or a predictive objective.

In both cases, some group assignments must be known before carrying out the Discriminant Analysis. Such group assignments, or labelling, may be arrived at in any way. Hence Discriminant Analysis can be employed as a useful complement to Cluster Analysis (in order to judge the results of the latter) or Principal Components Analysis. Alternatively, in star–galaxy separation, for instance, using digitised images, the analyst may define group (stars, galaxies) membership visually for a conveniently small *training set* or *design set*.

The basic ideas used in Discriminant Analysis are quite simple. For assessment of a classification, we attempt to optimally locate a curve — often a straight line — between the classes. For assignment, we basically look for the class which is closest to the new object that we wish to classify. As with other multivariate techniques, a variety of Discriminant Analysis methods are available, based on the precise meaning to be given to such terms as "optimal" and "closeness".

A brief "appetizer" of the range of Discriminant Analysis techniques that are widely used is as follows. It might be assumed for instance that the groups to be distinguished are of Gaussian distribution; or a very flexible separation surface might be used, such as would be entailed when assigning a new sample to the cluster which the k closest neighbours of the new sample belonged to (for some small constant k). Linear discrimination (or, in higher dimensional spaces, hyperplane separation) is both mathematically tractable, and may fulfil our objective of optimally separating classes. If the clusters are intertwined (see Figure 4.1b), linear separation will perform badly; on the other hand in Figure 4.1a, there are only two points misclassified. Hyperplane separation performs well when the classes are reasonably separated, and assumes no distributional or other properties on the part of the data.

In the following, we will adopt two distinct frames of reference for the de-

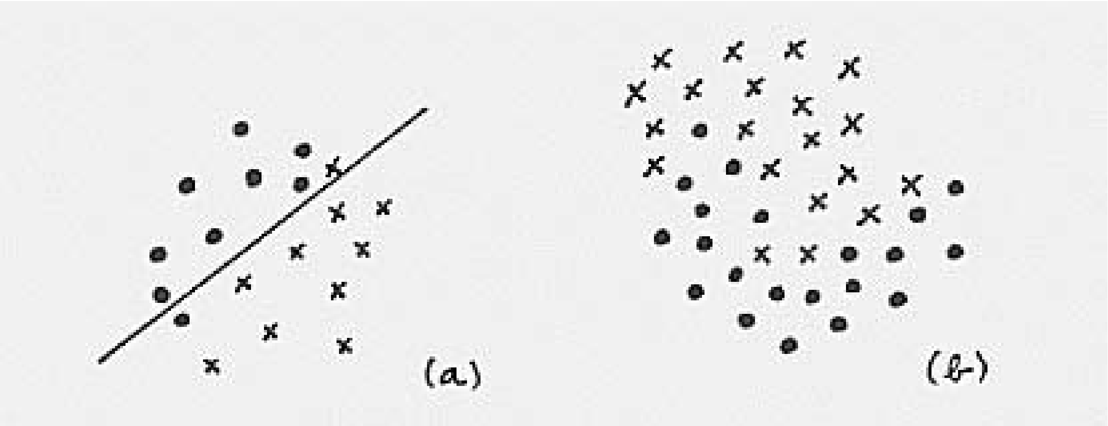


Figure 4.1: Two sets of two groups. Linear separation performs adequately in (a) but non-linear separation is more appropriate in (b).

scription of discriminant methods. On the one hand, we will look at a geometrical framework, which nicely complements the type of perspective employed in PCA. On the other hand, we will overview discriminant methods which make use of a probabilistic perspective. Interestingly, one well known method (Linear Discriminant Analysis) may be specified using *either* mathematical framework.

The fact that Discriminant Analysis is often embraced under the term "classification" means that care should be taken in distinguishing between Cluster and Discriminant Analyses. The former is *unsupervised classification*: no prior knowledge of the group memberships is usually required. Discriminant Analysis, when used for confirmatory or assessment purposes, is *supervised classification* and has been referred to as "learning with a teacher". Because of possible confusion in what are usually quite distinct problems and ranges of methods, we have generally eschewed the use of the term "classification".

A good deal of this chapter gathers together summaries of the major linear (i.e. matrix) transformations applicable to statistical pattern recognition – mostly – and estimation. One important objective touched on is to provide a foundation for research into the application of such techniques for (a) automatic recognition of two-dimensional patterns (human face images, morphologies of galaxies), and, (b) estimation from multidimensional signals, i.e. a generalisation of multivariate regression.

4.2 Mathematical Description

4.2.1 Statistical Pattern Recognition

In pattern recognition studies, the term pattern is usually synonomous with "collection of measurements or measured attributes", i.e. a tuple or vector. Thus, a general "pattern" or multidimensional "observed signal" may be represented by a p-dimensional vector x. Pattern recognition is the study of the decision mechanism: compute the class to which x belongs. A quite general solution is to transform x to some scalar, y, so that the decision can be expressed as a threshold – or multiple thresholds for multiple classes. In the theory of statistcal pattern recognition, the previous processes – transformation, followed by thresholding – correspond to the general separation: feature extraction from the raw input data, followed by classification based on the features. We are essentially dealing with what has been termed discriminant analysis in this perspective.

In signal estimation we start with the same measurement vector, x, from this we are required to estimate (derive) some unmeasurable quantity, y. Again, linear transformation of x to the scalar attribute y is a general solution, as, for example, in regression analysis.

Another fairly universal requirement in scientific data analysis, exploration, visualisation, etc. is to map from p-dimensions to q dimensions, where $q << p$, based on some optimality criterion, e.g. maximum information retention in the reduced dimensionality space – – data compression, maximization of discrimination between classes – feature extraction, as mentioned above.

For each of the above problems, it is possible (at least theoretically) to obtain solutions based on linear transformations of the form $y = Ax$, where x is a $p \times 1$ vector, y a $q \times 1$ vector, and A, a $q \times p$ matrix. Most of the transformations are based on statistical criteria: least-square-error, maximum likelihood.

If the data are in image format, i.e. in the form of an N row $\times M$ column image, F, the above form is not directly applicable. Nevertheless, such an image easily can be vectorized – rearranged as a single $NM \times 1$ vector – so the transformations are still valid.

In the previous case, and where dimensionality of the input vectors is large, the size of p ($= NM$ for an image) may make the matrix A difficult to handle, and worse still, estimate. For example for a 512×512 image, F, $p = NM = 262144$, the correlation matrix of F is $262144 \times 262144 =$ (approx.) 69×109 elements. However, there are methods for transformations which operate on greatly reduced subspaces of the input space.

Linear transformations are attractive because of their analytical tractability, for example in most cases it is possible to obtain formulas linking statistics of input vectors to those in the transformed space.

In the next section, each transformation is described according to its optimization criterion and applicable problems; where it will help understanding of the applicability of the transformation, some basic theory is given; implementation details such as mathematical formulas, algorithms or reference to specific software is given. We then describe adaptations of vector transformations and special treatment required for image patterns.

Appendix A contains a review of relevent results from linear algebra.

4.2.2 Linear Transformations: Introduction

Figure 4.2 summarizes a general statistical pattern recognition system.

The observation vector x is the input, the final output is the class or desision, w; in general the input can come from one of c classes, $w_i, i = 1, 2, \ldots c$.

The feature extraction abstraction is an important one; the purpose of the feature extractor is to produce (usually a minimal) set of measurements (a feature vector) that allow (a) the classes to be distinguished, and, (b) remove, as far as possible, noise and other defects introduced by the measurement system.

The purpose of the classifier is to identify, on the basis of the components of an input feature vector (x), what class x belongs to. In a simple implementation,

```
            +-----------------+          +---------------+
            |                 |          |               |
Observation | Feature         | Feature  |  Classifier   |Decision
----------->| Extraction      |--------->|               |-------->
x (Vector)  |                 | Vector y |               |Vector w
            |                 |          |               |
            +-----------------+          +---------------+
```

Figure 4.2: Pattern recognition system.

this might involve computing the “similarity measure” between x and a number of stored “perfect” vectors, and choosing the class with maximum similarity.

More formally, the feature vectors form a *feature space.* The job of the classifier is to partition this space into (disjoint) regions, each region corresponding to a class; normally, the classification task will be eased if there are as few features as possible, and, they are chosen so as to best separate the classes.

There are two major categories of feature extraction: (a) feature selection – based on possibly ad hoc information about the data, and (b) feature extraction using linear tranformation. This chapter is primarily concerned with (b) – input vector x is transformed to a feature vector y where the components of y satisfy some optimality criteria, e.g. dimensionality reduction – with retention of information, class separability.

The simplest possible transformation is given by:

$$y = a'x$$

i.e. x is transformed to a single scalar. Classification is then a simple matter of thresholding. If, without loss of generality, we restrict to the two-class case:

$$\begin{array}{cccll} \text{y} & > & T & \Longrightarrow & \text{class 1} \\ & < & T & \Longrightarrow & \text{class 2} \\ & = & T & \Longrightarrow & \text{class chosen arbitrarily} \end{array}$$

4.2.3 Discriminants

The expressions just seen for y at the end of the previous section

form what is called a linear discriminant. A diagrammatic form is shown in Figure 4.3.

The readers familiar with neural networks will notice the similarity between this figure and a single neuron using a hard-limiting activation function.

A general discriminant function, g, is a function of the form:

$$y = g(x)$$

$$\begin{array}{cccll} \text{y} & > & T & \Longrightarrow & \text{class 1} \\ & < & T & \Longrightarrow & \text{class 2} \\ & = & T & \Longrightarrow & \text{class chosen arbitrarily} \end{array}$$

```
                                       T
        x0                             |
         \                             |
          \a1                          v
           \                      +----+----+
            \                     | 1|   +--|
             +--+--+              |  |   |  |  output 1 => class 1
x1    a2     |     |    y         | 0|   |  |         0 => class 2
------------+    +---------------->+--+---+--+-------->
      .      |     |              |  |   T  |
      .     /+--+--+              +---------+
           /
          /ap-1       y = sum of         y>T?  output = 1 if y>T
         /            ai.xi for i=1                     = 0 otherwise
       xp-1           to i=p
```

Figure 4.3: Linear discriminant.

```
      |       1        2 2 2 2
y1    |      1 1 1       2 2 2 2 2
      |    1 1 1 1 1 1   2 2 2 2 2
      |    1 1 1 1 1 1 1  2 2 2 2 2
      |    1 1 1 1 1 1          2 2
      |      1 1 1 1        class w2
      |          1
      |  class w1
      +--------------+---+--------------------------->
                      t1    t2                            y0
```

Figure 4.4: Feature space, two-class, two dimensions.

Linear Discriminant as Projection

It is often useful to think geometrically about pattern recognition and signal estimation problems. Feature vectors y reside in a p-dimensional feature space. Figure 4.4 shows feature vectors (points, data) for a two-class, two-dimensional feature space.

We can now examine the utility of a simple linear discriminant, i.e. $y = a'x$.

Projecting the data onto the y_0 axis corresponds to a discriminant vector $a = (a_1, a_2)' = (1.0, 0.0)'$, i.e. y_0 is given a weight of 1.0, y_1 is given zero weight. This projection would result in a little overlap of the classes (between points t_1, and t_2 – see Figure 4.4).

Projecting data onto axis y_2, discriminant vector = (0.0,1.0), would result in much overlap. However, projecting the vectors onto the line shown in Figure 4.5 would be close to optimal – no class overlap.

```
                             *
      |       1        2 2 * 2
y1    |      1 1 1       * 2 2 2 2
      |    1 1 1 1 1 1*  2 2 2 2 2
      |    1 1 1 1 1* 1 1  2 2 2 2 2
      |    1 1 1 * 1 1          2 2
      |      1*1 1 1         class w2
      |    *     1
      | *
      +-----------------------------------------------
                                                          y0


      * * * projection line
```

Figure 4.5: Linear discriminant as projection.

Projecting the data onto different axes is equivalent to rotating the axes, i.e. in the case above, we would rotate to a new set of axes y_0, y_1, where y_0 is the axis shown as `* * *`; obviously, in the case above, we would ignore the second, y_1, dimension since it would contribute nothing to the discrimination of the classes.

Fisher Linear Discriminant

The Fisher discriminant, defined for two-class problems, is one of the best known of all pattern recognition algorithms. It projects the data onto a single axis, defined by the Fisher discriminant vector a:

$$y = a'x$$

The Fisher discriminant simultaneously optimises two measures or criteria:

- maximum between-class separation, expressed as separation of the class means m_1, m_2,
- minimum within-class scatter, expressed as the within class variances, v_1, v_2

The Fisher criterion combines these two optimands as:

$$J = (m_1 - m_2)/(v_1 + v_2)$$

where the transformed means and variances are: $m_j = a'm_j, v_j = aS_ja', S_j =$ covariance for class j, $m_j =$ mean vector for class j.

The discriminant is computed using:

$$a = W^{-1}(m_1 - m_2)$$

where W is the pooled (overall, class-independent) covariance matrix, $W = P_1S_1 + P_2S_2$. P_1, P_2 are the prior probabilities (see Appendix A).

Procedure

1. Estimate class means m_j and covariance matrices S_j, and prior probabilities, P_j.
2. Compute pooled covariance matrix, W (see equation above).
3. Invert matrix W (using some standard matrix inversion subprogram).
4. Compute the discriminant vector, a (see equation above).
5. Apply the discriminant using equation $y = a'x$.

4.2.4 Karhunen-Loève Transform

Also called Principal Components Analysis, Factor Analysis, Hotelling Transform (see Chapter 2 for a different perspective on the same method).

The Karhunen-Loève (KL) transform rotates the axes such that the covariance matrix is diagonalized:

$$y = Ux$$

where, see Appendix A, U is the eigenvector matrix of S the covariance matrix (over all classes), i.e.

$$U'SU = L$$

where

$$L = \begin{pmatrix} \lambda_1 & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \lambda_p \end{pmatrix}$$

L is a diagonal matrix containing the variances in the transformed space, and

$$U = \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ u_i \\ \dots \\ u_p \end{pmatrix}$$

is the matrix formed by the *eigenvectors*, u_i, of S.

There is an eigenvector, u_i, corresponding to each eigenvalue λ_i. If we order the eigenvectors, u_i, according to decreasing size of the corresponding eigenvalue, we arrive at a transform in which the variance in dimension y_0 is largest, y_1, the next largest, etc.

If we equate variance to information, then the KL transform gives a method of mapping to a lower dimensionality space, m, say, $m < p$,

with maximum retention of information; thus its theoretical appeal for data compression. Let U_2 be the $m \times p$ matrix containing the first m rows of U, i.e. the m eigenvectors (projections) corresponding to the m largest eigenvalues. We can rewrite equation $y = Ux$ as

```
      |                                * first
y1    |                      x x  x * x  eigenvector.
      |                specification
```

Figure 4.6: KL transform.

$$y_2 = U_2 x$$

which corresponds to the coding part of the compression. We have reduced the data from p to m numbers. We can decode using,

$$x_2 = U_2' y_2$$

Now, x_2 will be a maximally faithful (in a least square error sense) recreation of the original vector x, i.e. it minimizes the expected square error between the original vector, and the decoded vector; the minimum square error (MSE) criterion is expressed as:

$$J = E\{(x - x_2)'(x - x_2)\}$$

Figure 4.6 gives a geometrical representation of a KL transform. We are given a cluster of data approximately elliptically shaped, with the major axis of the ellipse subtending an angle of about 45 degrees to the horizontal axis. The minor axis is perpendicular to that. The line denoted with the asterisks corresponds to the first eigenvector, i.e. the eigenvector corresponding to maximum variance.

Frequently the eigenvector/eigenvalue equation is expressed as:

$$R_{uk} = \lambda_k u_k$$

where, as above, λ_k is the kth eigenvalue, and u_k is the associated kth eigenvector.

Procedure

1. Estimate the overall covariance matrix S.

2. Compute the set of eigenvalues, Λ, and the set of eigenvectors, U, of S, using some standard subprogram.

3. Order the rows of U according to the corresponding eigenvalues, see above.

```
                +-----------------+      y class vector
                |                 +--->     class 0
     Observ-    | Feature         +--->     class 1
     ---------> | Extraction /    +--->     class j
     -ation x   | Classification  +--->
     Vector     |                 +--->     class c-1
                +-----------------+
```

Figure 4.7: Multiclass least square error linear discriminant.

4. Apply the transform using equation $y = Ux$. The components of y are ordered according to "information" content. Thus we can retain the first, m, say, of them as the most "significant" features.

The KL transform U is computed without any reference to class occupancy, – the class labels are ignored when S is computed. Therefore, is contrast with, for example, the Fisher discriminant, KL has no particular discriminating capability.

4.2.5 Least Square Error Linear Discriminant

The Fisher discriminant works only for two classes although it is possible to tackle multi-class problems using pairwise discriminants for each of the $\begin{pmatrix} c \\ 2 \end{pmatrix}$ dichotomies, and furthermore we will discuss the fully multiclass Fisher discriminant later in this chapter. Multiple or canonical discriminant analysis are terms which are also used for the multiple class case.

An alternative (but actually similar to the Fisher discriminant) approach is to express the pattern recognition problem as in Figure 4.7, which shows a mapping from the $p \times 1$ observation vector x directly to a $c \times 1$ class vector, y.

The class vector is a binary vector, with only one bit set at any time, in which a bit j set ($= 1$) denotes class j; i.e.

$$y = (1, 0, 0, ...0)' \text{ denotes class } 0$$

$$y = (0, 1, 0, ...0)' \text{ denotes class } 1$$

etc.

The problem can be set up as one of multiple linear regression, in which the independent variables are the components of x, and the dependent variables – the c class "bits" – are the components of y.

For, initially, just one component of y, y_j, the regression problem can be expressed compactly as:

$$y_{ji} = x_i' b + e_i$$

where $b = (b_0, b_1, \ldots, b_{p-1})'$ is a $p \times 1$ vector of coefficients for class j, $x_i = (1, x_{i1}, \ldots x_{ip-1})'$ is the pattern vector, and $e_i =$ error, $y_{ji} = 1$ if the pattern x belongs to class j, $= 0$ otherwise.

Formulation in terms of the augmented vector x, which contains the bias element 1 is important; without it we would effectively be fitting a straight line through the origin – the bias (b_0) corresponds to a non-zero intercept of the y-axis; compared to using a separate bias element, the analysis is greatly simplified.

The complete set of n observation equations can be expressed as:

$$y = Xb + e$$

where $e = (e_1, e_2, \dots e_i, \dots e_n)'$, and $y = (y_1, y_2, \dots y_i, \dots y_n)'$, the $n \times 1$ vector of observations of the class variable (bit). X is the $n \times p$ matrix formed by n rows of p pattern components.

The least square error fitting is given by:

$$\textit{Least square error fitting equation: } b' = (X'X)^{-1}X'y$$

Note: the jkth element of the $p \times p$ matrix $X'X$ is $\sum_i x_{ji}x_{ik}$, and the jth row of the $p \times 1$ vector $X'y$ is $\sum_i x_{ij}y_i$. Thus, $X'X$ differs from the autocorrelation matrix of x only by a multiplicative factor, n, so that the major requirement for the least square error fitting equation above to provide a valid result is that the autocorrelation matrix of x is non-singular.

We can express the complete problem, where the vector y has c components by replacing the vector y in the least square error fitting equation with the matrix Y, the $n \times c$ matrix formed by n rows of c observations.

Thus, the least square error fitting equation extends to the complete least square error linear discriminant:

$$B' = (X'X)^{-1}X'Y$$

$X'Y$ is now a $p \times c$ matrix, and B' is a $p \times c$ matrix of parameters, i.e. one column of p parameters for each dependent variable.

Applying the discriminant/transformation is simply a matter of premultiplying the (augmented) vector x by B':

$$y' = B'x$$

4.2.6 Special Cases for Image Patterns

The transforms given in the previous sections are, in general, directly applicable to image patterns. By an "image" pattern we mean that the components of the pattern are derived from the pixels of a two-dimensional image. Let f be an image pattern, where the general pixel at row r, and column c is $f[r,c]$ – or f_{rc}; there are N rows, $r = 0 \dots N-1$ and M columns, $c = 0 \dots M-1$.

$$f = \begin{pmatrix} f_{00} & f_{01} & \cdots & \cdots & \cdots & f_{0,M-1} \\ f_{10} & f_{11} & \cdots & \cdots & \cdots & f_{1,M-1} \\ \cdots & \cdots & \cdots & \cdots & f_{rc} & \cdots \\ f_{N-1,0} & f_{N-1,1} & \cdots & \cdots & \cdots & f_{N-1,M-1} \end{pmatrix}$$

Trivially, f can be represented as an $NM \times 1$ vector as follows:

$$x = (f_{00}, f_{01}, \dots f_{0,M-1}, f10 \dots f_{rc} \dots f_{N-1,0} \cdots f_{N-1,M-1})$$

Usually, the major problem with extending our results to images is in the difficulty of estimating statistics. For example, if the images f are $N \times M$, the vectors x are $NM \times 1$; therefore the autocorrelation and covariance matrices for x (and f) are $NM \times NM$. If the images are modestly sized at 128×128 ($N = 128, M = 128$) we have an autocorrelation matrix of 16384×16384 values. Such a matrix is simply not estimatible, nor handleable.

Eigenimages

The treatment of the Karhunen-Loève transform (eigenvector expansion) above uses the autocorrelation matrix R as the basis of the eigenvectors. Frequently, the covariance matrix S is used instead. In this section on eigenfaces (or eigengalaxies etc.), we will use the covariance matrix. Daubos (1997) studied eigengalaxies in the same way.

Naively, the Karhunen-Loève transform ($y = Ux$) can be easily extended to apply to image patterns, by expressing the sample images as vectors. However, the problem mentioned in the previous section, of a huge dimensionality covariance matrix, is immediately obvious; the dimensionality of x is $p = NM$; assume for the remainder that $N = M$, i.e. we have a square image.

If we express the image patterns f_i as vectors x_i using the vectorization approach above, let there be n of them in the sample, $x_i, i = 1 \ldots n$.

If we center our data, i.e., $x' = x - m$ (the pattern vector is reduced to zero mean – note that the prime here indicates just another variable and not transpose), , and define the matrix X as $X = [x'_1 x'_2 \ldots x'_i \ldots x'_n]$, a $p \times n$ matrix where $p = N^2$, the sample covariance matrix can be expressed as

$$S = \frac{1}{n} X X'$$

which is $N^2 \times N^2$. However, see Appendix A, the rank of S is only n, and therefore there are only n non-zero eigenvalues when S is diagonalized. Based on this we have a method of finding these n eigenvalues/eigenvectors based on a reduced dimensionality $n \times n$ matrix, as we will now show.

If we express the eigenvalue/eigenvector equation as $R_{uk} = \lambda_k u_k$ (see above), then for the $n \times n$ matrix $T = X'X$,

$$T_{vk} = \lambda_k v_k$$

i.e.

$$X' X v_k = \lambda_k v_k$$

where, as before, λ_k is the kth eigenvalue, and v_k is the associated kth eigenvector; note that the v_k are $n \times 1$ vectors.

Premultiply each side by X,

$$X X' X u_k = \lambda_k X v_k$$

Now this too is an eigenvalue/eigenvector equation for the matrix XX' which $= nS$. Therefore the $p \times 1$ vectors $X v_k = u_k$ are the eigenvectors of nS.

Note the dimensionalities: u_k is $p \times 1$, $X v_k$ is $(p \times n) \times (n \times 1)$.

To clarify what is happening we can show the above expression fully expanded:

$$\begin{pmatrix} u_{0k} \\ u_{1k} \\ . \\ u_{p-1,k} \end{pmatrix} = \begin{pmatrix} x_{01} & x_{02} & ... & x_{0i} & ... & x_{0n} \\ x_{11} & x_{12} & ... & x_{1i} & ... & x_{1n} \\ ... & ... & ... & ... & ... & ... \\ x_{p-1,1} & x_{p-1,2} & ... & x_{p-1,i} & ... & x_{p-1,n} \end{pmatrix} \begin{pmatrix} v_{k1} \\ v_{k2} \\ . \\ u_{kn} \end{pmatrix}$$

i.e., the kth eigenvector of S (an eigenimage) is formed by a linear combination of all the n training images; the coefficient/weight for image i being the ith component of v_k.

Procedure

Assume we have n training images $f_i, i = 1 \times n$, and they are of size $N \times N$; in fact, it is more convenient to deal with vectors x_i and $x'_i = x_i - m$ where m is the average over the n training vectors (i.e. here, images).

1. Compute the average image m:

$$m = \frac{1}{n} \sum_{i=1}^{n} x_i$$

2. Form the $n \times n$ matrix T ($= X'X$) where the rcth component of T, $T_{rc} = x'_r x_c$, the dot product of the rth and cth training vectors/images.

3. Find the n eigenvectors of T, v_k.

4. Order the v_k according to decreasing eigenvalue.

5. Use the equation $u_k = Xv_k$ to compute the n' ($< n$) most significant eigenimages, u_k, $k = 1 \ldots n'$.

Conclusion on Special Cases

The eigenimage can be effectively used to characterize basic components in our image data. Examples of its use include eigenfaces (i.e. eigenimages of faces), for the purposes of simplifying human face recognition. Eigenimages may be a useful dataset to store on a memory chip on a surveillance card (credit card, access card), with the relatively less information-rich discrepancy between it and any individual's face being stored in addition. Another example is to define eigengalaxies, to address the problem of sorting and organizing images of different galaxy morphologies.

4.2.7 Multiple Discriminant Analysis

Multiple Discriminant Analysis (MDA) is also termed Discriminant Factor Analysis and Canonical Discriminant Analysis. It adopts a similar perspective to PCA: the rows of the data matrix to be examined constitute points in a multidimensional space, as also do the group mean vectors. Discriminating axes are determined in this space, in such a way that optimal separation of the predefined groups is attained. As with PCA, the problem becomes mathematically the eigenreduction of a real, symmetric matrix.

Consider the set of objects, $i \in I$; they are characterised by a finite set of parameters, $j \in J$. The vectors associated with the objects are given as the row vectors of the matrix $X = \{x_{ij}\}$. The grand mean of all vectors is given by

$$g_j = \frac{1}{n} \sum_{i \in I} x_{ij}$$

(where n is the cardinality of I). Let y_j be the j^{th} coordinate of the mean of group y; i.e.

$$y_j = \frac{1}{n_y} \sum_{i \in y} x_{ij}$$

(where n_y is the cardinality of group y). Finally, we consider the case of mutually disjoint groups, y, whose union gives I. Let Y be the set of these groups, and let n_Y be the number of groups considered. Evidently, $n_Y \leq n$.

We now define the following three variance–covariance matrices. T (of jk^{th} term, t_{jk}) is the total covariance matrix; W is the within classes covariance matrix; and B is the between classes covariance matrix:

$\mathbf{T}\ : t_{jk} = \frac{1}{n} \sum_{i \in I} (x_{ij} - g_j)(x_{ik} - g_k)$

$\mathbf{W}\ : w_{jk} = \frac{1}{n} \sum_{y \in Y} \sum_{i \in y} (x_{ij} - y_j)(x_{ik} - y_k)$

$\mathbf{B}\ : b_{jk} = \sum_{y \in Y} \frac{n_y}{n} (y_j - g_j)(y_k - g_k).$

The three matrices, T, W and B, are of dimensions $m \times m$ where m is the number of attributes (i.e. the vectors considered, their grand mean, and the group means are located in $\mathbb{R}^m$).

Generalizing Huyghen's Theorem in classical mechanics, we have that $T = W + B$. This is proved as follows. We have, for the $(j,k)^{th}$ terms of these matrices:

$$\frac{1}{n} \sum_{i \in I} (x_{ij} - g_j)(x_{ik} - g_k)$$

$$= \frac{1}{n} \sum_{y \in Y} \sum_{i \in y} (x_{ij} - y_j)(x_{ik} - y_k) + \sum_{y \in Y} \frac{n_y}{n} (y_j - g_j)(y_k - g_k).$$

Rewriting the first term on the right hand side of the equation as

$$\frac{1}{n} \sum_{y \in Y} \sum_{i \in y} ((x_{ij} - g_j) - (y_j - g_j))((x_{ik} - g_k) - (y_k - g_k))$$

and expanding gives the required result.

The sum of squared projections of the points in $\mathbb{R}^m$ along any given axis $\mathbf{u}$ is given by $\mathbf{u}'T\mathbf{u}$ (cf. the analogous situation in Principal Components Analysis). For the class means along this axis we have $\mathbf{u}'B\mathbf{u}$. Finally, for the within class deviations along this axis, we have $\mathbf{u}'W\mathbf{u}$. Since $T = W + B$, we have that

$$\mathbf{u}'T\mathbf{u} = \mathbf{u}'B\mathbf{u} + \mathbf{u}'W\mathbf{u}.$$

The optimal discrimination of the given groups is carried out as follows. We choose axis $\mathbf{u}$ to maximize the spread of class means, while restraining the compactness of the classes, i.e.

$$max \frac{\mathbf{u}' B \mathbf{u}}{\mathbf{u}' W \mathbf{u}}.$$

This maximization problem is the same as

$$min \frac{\mathbf{u}' W \mathbf{u}}{\mathbf{u}' B \mathbf{u}} = min \frac{\mathbf{u}' W \mathbf{u}}{\mathbf{u}' B \mathbf{u}} + 1 = min \frac{\mathbf{u}' W \mathbf{u} + \mathbf{u}' B \mathbf{u}}{\mathbf{u}' B \mathbf{u}}$$

$$= min \frac{\mathbf{u}' T \mathbf{u}}{\mathbf{u}' B \mathbf{u}} = max \frac{\mathbf{u}' B \mathbf{u}}{\mathbf{u}' T \mathbf{u}}.$$

As in PCA (refer to Chapter 2), we use λ as a Lagrangian multiplier, and differentiate the expression $\mathbf{u}' B \mathbf{u} - \lambda(\mathbf{u}' T \mathbf{u})$ with respect to $\mathbf{u}$. This yields $\mathbf{u}$ as the eigenvector of $T^{-1}B$ associated with the largest eigenvalue, λ. Eigenvectors associated with successively large eigenvalues define discriminating factors or axes which are orthogonal to those previously obtained. We may therefore say that MDA is the PCA of a set of centred vectors (the group means) in the T^{-1}-metric.

A difficulty has not been mentioned in the foregoing: the matrix product, $T^{-1}B$ is not necessarily symmetric, and so presents a problem for diagonalization. This difficulty is circumvented as follows. We have that $B\mathbf{u} = \lambda T \mathbf{u}$. Writing B as the product of its square roots CC' (which we can always do because of the fact that B is necessarily positive definite and symmetric) gives: $CC'\mathbf{u} = \lambda T \mathbf{u}$. Next, define a new vector $\mathbf{a}$ as follows: $\mathbf{u} = T^{-1}C\mathbf{a}$. This gives:

$$CC'T^{-1}C\mathbf{a} = \lambda T T^{-1} C\mathbf{a}$$

$$\Rightarrow C(C'T^{-1}C)\mathbf{a} = \lambda C\mathbf{a}$$

$$\Rightarrow (C'T^{-1}C)\mathbf{a} = \lambda \mathbf{a}.$$

We now have an eigenvalue equation, which has a matrix which is necessarily real and symmetric. This is solved for $\mathbf{a}$, and substituted back to yield $\mathbf{u}$.

Since the largest eigenvalue is

$$\frac{\mathbf{u}' B \mathbf{u}}{\mathbf{u}' T \mathbf{u}} = \frac{\mathbf{u}' T \mathbf{u} - \mathbf{u}' W \mathbf{u}}{\mathbf{u}' T \mathbf{u}},$$

it is seen that the right side here, and hence all eigenvalues, are necessarily positive and less than 1.

The eigenvalues represent the discriminating power of the associated eigenvectors. Unlike in PCA, the percentage variance explained by a factor has no sense in MDA, since the sum of eigenvalues has no meaning.

The n_Y classes lie in a space of dimension at most $n_Y - 1$. This will be the number of discriminant axes or factors obtainable in the most common practical case when $n > m > n_Y$.

Linear of Fisher Discriminant Analysis

In this section, we will examine the 2-group case of MDA, and focus on the assignment problem. Discriminant Analysis may be used for assigning a new object, as well as for confirming the separation between given groups. The distance, in this new T^{-1}-metric, between some new vector $\mathbf{a}$ and the barycentre (or centre of gravity) $\mathbf{y}$ of class y is defined by the *Mahalanobis* or *generalized distance*:

$$d(a, y) = (\mathbf{a} - \mathbf{y})' T^{-1} (\mathbf{a} - \mathbf{y}).$$

Vector $\mathbf{a}$ is assigned to the class y such that $d(a, y)$ is minimal over all groups. In the two-group case, we have that $\mathbf{a}$ is assigned to group y_1 if

$$d(a, y_1) < d(a, y_2).$$

Equality in the above may be resolved in accordance with user choice. Writing out explicitly the Euclidean distances associated with the matrix T^{-1}, and following some simplifications, we find that vector $\mathbf{a}$ is assigned to group y_1 if

$$(\mathbf{y}_1 - \mathbf{y}_2)' T^{-1} \mathbf{a} > \frac{1}{2} (\mathbf{y}_1 - \mathbf{y}_2)' T^{-1} (\mathbf{y}_1 + \mathbf{y}_2)$$

and to group y_2 if

$$(\mathbf{y}_1 - \mathbf{y}_2)' T^{-1} \mathbf{a} < \frac{1}{2} (\mathbf{y}_1 - \mathbf{y}_2)' T^{-1} (\mathbf{y}_1 + \mathbf{y}_2).$$

The left hand side is the T^{-1}–projection of $\mathbf{a}$ onto $\mathbf{y}_1 - \mathbf{y}_2$ (i.e. the vector connecting $\mathbf{y}_2$ to $\mathbf{y}_1$; and the right hand side is the T^{-1}–projection of $(\mathbf{y}_1 + \mathbf{y}_2)/2$ onto $\mathbf{y}_1 - \mathbf{y}_2$ (see Figure 4.8). This concords well with an intuitive idea of the objective in assignment: a new sample is assigned to a group if it lies closer to it than does the mid–way point between the two groups.

This allocation rule may be rewritten as

$$(\mathbf{y}_1 - \mathbf{y}_2)' T^{-1} (\mathbf{a} - \frac{\mathbf{y}_1 + \mathbf{y}_2}{2}) > 0 \Longrightarrow \mathbf{a} \rightarrow y_1$$

$$(\mathbf{y}_1 - \mathbf{y}_2)' T^{-1} (\mathbf{a} - \frac{\mathbf{y}_1 + \mathbf{y}_2}{2}) < 0 \Longrightarrow \mathbf{a} \rightarrow y_2.$$

The left hand side here is known as *Fisher's linear discriminant function.*

4.2.8 Bayesian Discrimination: Quadratic Case

The assignment aspect of discrimination is at issue in this section. As a general rule, it is clearly best if we attempt to take as much information as possible about the problem into account. Let us look at how successively more problem–related characteristics can be considered, but simultaneously we will have to pinpoint difficulties in the implementation of our successive solutions. Overcoming these difficulties will lead to other approaches for the assignment problem, as will be seen.

Consider a vector of measured parameters, $\mathbf{x}$, relating to attributes of galaxies. Next consider that a sample of galaxies which is being studied consists of 75% spirals and 25% ellipticals. That is

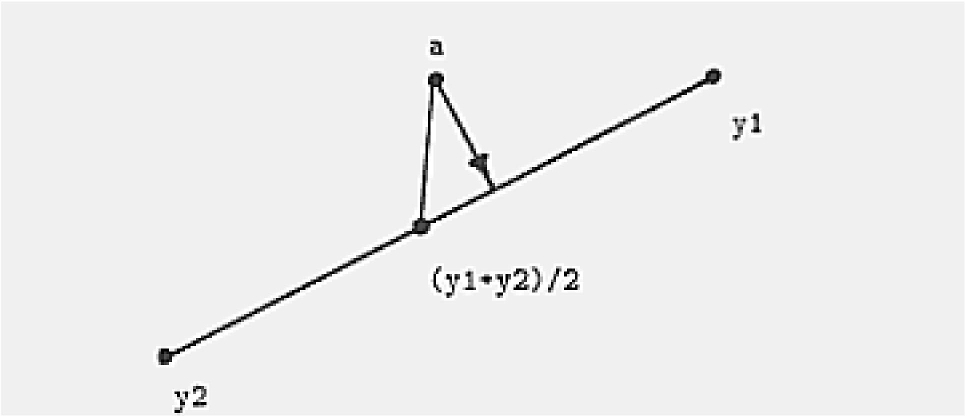


Figure 4.8: The assignment of a new sample $\mathbf{a}$ to one of two groups of centres $\mathbf{y}_1$ and $\mathbf{y}_2$.

$$P(spiral) = 0.75,$$
$$P(elliptical) = 0.25.$$

where $P(.)$ denotes probability. In the absence of any other information, we would therefore assign any unknown galaxy to the class of spirals. In the long run, we would be correct in 75% of cases, but we have obviously derived a very crude assignment rule.

Consider now that we are given also the conditional probabilities: for a particular set of parameter values, $\mathbf{x}_0$, we have

$$P(spiral \mid \mathbf{x}_0) = 0.3$$
$$P(elliptical \mid \mathbf{x}_0) = 0.7.$$

In this case, we are led to choose the class of ellipticals for our unknown galaxy, for which we have measured the parameter values $\mathbf{x}_0$. The conditional probabilities used above are referred to as *prior* or *a priori* probabilities.

This leads to Bayes' rule for the assignment of an unknown object to group c rather than to any other group, y:

$$P(c \mid \mathbf{x}_0) > P(y \mid \mathbf{x}_0) \quad for\ all\ y \neq c. \tag{4.1}$$

Tied optimal assignment values may be arbitrarily resolved.

A difficulty arises with Bayes' rule as defined above: although we could attempt to determine $P(c \mid \mathbf{x})$ for all possible values of $\mathbf{x}$ (or, perhaps, for a discrete set of such values), this is cumbersome. In fact, it is usually simpler to derive values for $P(\mathbf{x}_0 \mid c)$, i.e. the probability of having a given set of measurements, $\mathbf{x}_0$, given that we are dealing with a given class, c. Such probabilities are referred to as *posterior* or *a posteriori* probabilities. Bayes' theorem relates priors and posteriors. We have:

$$P(c \mid \mathbf{x}_0) = \frac{P(\mathbf{x}_0 \mid c)P(c)}{\sum_{all\ y} P(\mathbf{x}_0 \mid y)P(y)}. \tag{4.2}$$

All terms on the right hand side can be sampled: $P(c)$ is determined straightforwardly; $P(\mathbf{x}_0 \mid c)$ may be sampled by looking at each parameter in turn among the vector $\mathbf{x}_0$, and deriving estimates for the members of class c.

Substituting equation (4.2) into both sides of equation (4.1), and cancelling the common denominator, gives an assignment rule of the following form: *choose class c over all classes y, if*

$$P(\mathbf{x}_0 \mid c)\ P(c) > P(\mathbf{x}_0 \mid y)\ P(y) \quad \mathit{for\ all}\ y \neq c. \tag{4.3}$$

This form of Bayes' rule is better than the previous one. But again a difficulty arises: a great deal of sampling is required to estimate the terms of expression (4.3). Hence it is convenient to make distributional assumptions about the data. As always the Gaussian or normal distribution (for historical rectitude, known in the French literature as the distribution of Gauss–Laplace) figures prominently.

The multivariate normal density function (defining a multidimensional bell–shaped curve) is taken to better represent the distribution of $\mathbf{x}$ than the single point as heretofore. This is defined as

$$(2\pi)^{-\frac{n}{2}} \mid V \mid^{-\frac{1}{2}} \ \mathit{exp}\ (-\frac{1}{2}(\mathbf{x}-\mathbf{g})'V^{-1}(\mathbf{x}-\mathbf{g}))$$

where V is the variance–covariance matrix. It is of dimensions $m \times m$, if m is the dimensionality of the space. If equal to the identity matrix, it would indicate that $\mathbf{x}$ is distributed in a perfectly symmetric fashion with no privileged direction of elongation. $\mid V \mid$ is the determinant of the matrix V.

Assuming that each group, c, is a Gaussian, we have

$$P(\mathbf{x} \mid c) = (2\pi)^{-\frac{n}{2}} \mid V_c \mid^{-\frac{1}{2}} \ \mathit{exp}\ (-\frac{1}{2}(\mathbf{x}-\mathbf{g}_c)'V_c^{-1}(\mathbf{x}-\mathbf{g}_c))$$

where $\mathbf{g}_c$ is the centre of class c, and V_c is its variance–covariance matrix.

Substituting this into equation (4.3), taking natural logs of both sides of the inequality, and cancelling common terms on both sides, gives the following assignment rule: *assign* $\mathbf{x}$ *to class c if*

$$\begin{gathered} \mathit{ln} \mid V_c \mid + (\mathbf{x}-\mathbf{g}_c)'V_c^{-1}(\mathbf{x}-\mathbf{g}_c) - \mathit{ln}\ P(c) \\ < \mathit{ln} \mid V_y \mid + (\mathbf{x}-\mathbf{g}_y)'V_y^{-1}(\mathbf{x}-\mathbf{g}_y) - \mathit{ln}\ P(y) \quad \mathit{for\ all}\ y \neq c. \end{gathered}$$

This expression is simplified by defining a "discriminant score" as

$$\delta_c(\mathbf{x}) = \mathit{ln} \mid V_c \mid + (\mathbf{x}-\mathbf{g_c})'V_c^{-1}(\mathbf{x}-\mathbf{g_c}).$$

The assignment rule then becomes: *assign* $\mathbf{x}$ *to class c if*

$$\delta_c(\mathbf{x}) - \mathit{ln}\ P(c) < \delta_y(\mathbf{x}) - \mathit{ln}\ P(y) \quad \mathit{for\ all}\ y \neq c.$$

The dividing curve between any two classes immediately follows from this. It is defined by

$$\delta_c(\mathbf{x}) - \mathit{ln}\ P(c) = \delta_y(\mathbf{x}) - \mathit{ln}\ P(y) \quad \mathit{for\ all}\ y \neq c$$

The shape of a curve defined by this equation is quadratic. Hence this general form of Bayesian discrimination is also referred to as *quadratic discrimination*.

4.2.9 Maximum Likelihood Discrimination

In a practical context we must estimate the mean vectors ($\mathbf{g}_y$) and the variance–covariance matrices (V_y) from the data which may be taken to constitute a sample from an underlying population. These are termed *plug in estimators*, since they are sample estimates of the unknown parameters.

We have used a multivariate normal density function for $P(\mathbf{x} \mid y)$. If all n objects $\mathbf{x}_i$ have been independently sampled, then their joint distribution is

$$\mathcal{L} = \Pi_{i=1}^{n} P(\mathbf{x}_i \mid y).$$

Considering $\mathcal{L}$ as a function of the unknown parameters $\mathbf{g}$ and V, it is termed a *likelihood function*. The *principle of maximum likelihood* then states that we should choose the unknown parameters such that $\mathcal{L}$ is maximized. The classical approach for optimizing $\mathcal{L}$ is to differentiate it with respect to $\mathbf{g}$ and then with respect to V, and to set the results equal to zero. Doing this for the multivariate normal expression used previously allows us to derive estimates for the mean and covariances as follows.

$$\hat{\mathbf{g}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$$

$$\hat{V} = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{g})(\mathbf{x}_i - \mathbf{g})'.$$

These are used to provide *maximum likelihood estimates* for the Bayesian classifier.

In a more general setting, we could wish to consider a multivariate normal mixture of the following form:

$$P(\mathbf{x} \mid y) = \sum_{k} w_k f_k(\mathbf{x} \mid \mathbf{g}_k, V_k)$$

where k ranges over the set of mixture members, w is a weighting factor, and the function f depends on the mean and the covariance structure of the mixture members. For such density functions, an iterative rather than an analytic approach is used, although boundedness and other convergence properties are problemsome (see Hand, 1981).

4.2.10 Bayesian Equal Covariances Case

The groups we study will not ordinarily have the same covariance structure. However it may be possible to assume that this is the case, and here we study what this implies in Bayesian discrimination.

The discriminant score, defined in section 4.2.3, when expanded is

$$\delta_c(\mathbf{x}) = ln \mid V_c \mid + \mathbf{x}' V_c^{-1} \mathbf{x} - \mathbf{g}_c' V_c^{-1} \mathbf{x} - \mathbf{x}' V_c^{-1} \mathbf{g}_c + \mathbf{g}_c' V_c^{-1} \mathbf{g}_c.$$

The first two terms on the right hand side can be ignored since they will feature on both sides of the assignment rule (by virtue of our assumption of equal covariances); and the third and fourth terms are equal. If we write

$$\phi_c(\mathbf{x}) = 2\mathbf{g}_c' V_c^{-1} \mathbf{x} - \mathbf{g}_c' V_c^{-1} \mathbf{g}_c$$

then the assignment rule is: *assign* **x** *to class c if*

$$\phi_c(\mathbf{x}) + ln\ P(c) > \phi_y(\mathbf{x}) + ln\ P(y) \quad for\ all\ y \neq c.$$

However, ϕ can be further simplified. Its second term is a constant for a given group, a_0; and its first term can be regarded as a vector of constant coefficients (for a given group), **a**. Hence ϕ may be written as

$$\phi_c(\mathbf{x}) = a_{c0} + \sum_{j=1}^{m} a_{cj}x_j.$$

Assuming $P(c) = P(y)$, for all y, the assignment rule in the case of equal covariances thus involves a linear decision surface. We have a result which is particularly pleasing from the mathematical point of view: Bayesian discrimination in the equal covariances case, when the group cardinalities are equal, gives *exactly* the same decision rule (i.e. a linear decision surface) as linear discriminant analysis discussed from a geometric standpoint.

4.2.11 Non–Parametric Discrimination

Non–parametric (distribution–free) methods dispense with the need for assumptions regarding the probability density function. They have become very popular especially in the image processing area.

Given a vector of parameter values, $\mathbf{x}_0$, the probability that any unknown point will fall in a local neighbourhood of $\mathbf{x}_0$ may be defined in terms of the relative volume of this neighbourhood. If n' points fall in this region, out of a set of n points in total, and if v is the volume taken up by the region, then the probability that any unknown point falls in the local neighbourhood of $\mathbf{x}_0$ is n'/nv. An approach to classification arising out of this is as follows.

In the *k-NN* (k nearest neighbours) approach, we specify that the volume is to be defined by the k NNs of the unclassified point. Consider n_c of these k NNs to be members of class c, and n_y to be members of class y (with $n_c + n_y = k$). The conditional probabilities of membership in classes c and y are then

$$P(\mathbf{x}_0 \mid c) = \frac{n_c}{nv}$$
$$P(\mathbf{x}_0 \mid y) = \frac{n_y}{nv}.$$

Hence the decision rule is: *assign to group c if*

$$\frac{n_c}{nv} > \frac{n_y}{nv}$$
$$i.e. \quad n_c > n_y.$$

The determining of NNs of course requires the definition of distance: the Euclidean distance is usually used.

An interesting theoretical property of the NN–rule relates it to the Bayesian misclassification rate. The latter is defined as

$$1 - max_y P(y \mid \mathbf{x}_0)$$

or, using notation introduced previously,

$$1 - P(c \mid \mathbf{x}_0). \tag{4.4}$$

This is the probability that $\mathbf{x}_0$ will be misclassified, given that it should be classified into group c.

In the 1-NN approach, the misclassification rate is the product of: the conditional probability of class y given the measurement vector $\mathbf{x}$, and one minus the conditional probability of class y given the NN of $\mathbf{x}$ as the measurement vector:

$$\sum_{all\ y} P(y \mid \mathbf{x})(1 - P(y \mid NN(\mathbf{x})). \tag{4.5}$$

This is the probability that we assign to class y given that the NN is not in this class. It may be shown that the misclassification rate in the 1–NN approach (expression 4.5) is not larger than twice the Bayesian misclassification rate (expression 4.4). Hand (1981) may be referred to for the proof.

4.3 Examples and Bibliography

4.3.1 Practical Remarks

We can evaluate *error rates* by means of a training sample (to construct the discrimination surface) and a test sample. An optimistic error rate is obtained by reclassifying the design set: this is known as the *apparent error rate*. If an independent test sample is used for classifying, we arrive at the *true error rate*.

The *leaving one out* method attempts to use as much of the data as possible: for every subset of $n-1$ objects from the given n objects, a classifier is designed, and the object omitted is assigned. This leads to the overhead of n discriminant analyses, and to n tests from which an error rate can be derived. Another approach to appraising the results of a discriminant analysis is to determine a *confusion matrix* which is a contingency table (a table of frequencies of co–occurrence) crossing the known groups with the obtained groups.

We may improve our discrimination by implementing a *reject option*: if for instance we find $P(\mathbf{x} \mid c) > P(\mathbf{x} \mid y)$ for all groups $y \neq c$, we may additionally require that $P(\mathbf{x} \mid c)$ be greater than some threshold for assignment of $\mathbf{x}$ to c. Such an approach will of course help to improve the error rate.

There is no best discrimination method. A few remarks concerning the advantages and disadvantages of the methods studied are as follows.

- Analytical simplicity or computational reasons may lead to initial consideration of linear discriminant analysis or the NN–rule.

- Linear discrimination is the most widely used in practice. Often the 2-group method is used repeatedly for the analysis of pairs of multigroup data (yielding $k(k-1)/2$ decision surfaces for k groups).

- To estimate the parameters required in quadratic discrimination requires more computation and data than in the case of linear discrimination. If there is not a great difference in the group covariance matrices, then the latter will perform as well as quadratic discrimination.

- The k–NN rule is simply defined and implemented, especially if there is insufficient data to adequately define sample means and covariance matrices.

- MDA is most appropriately used for *feature selection*. As in the case of PCA, we may want to focus on the variables used in order to investigate the differences between groups; to create synthetic variables which improve the grouping ability of the data; to arrive at a similar objective by discarding irrelevant variables; or to determine the most parsimonious variables for graphical representational purposes.

4.3.2 Examples from Astronomy

1. H.–M. Adorf, "Classification of low–resolution stellar spectra via template matching — a simulation study", *Data Analysis and Astronomy*, (Proceedings of International Workshop on Data Analysis and Astronomy, Erice, Italy, April 1986) Plenum Press, New York, 1986 (in press).

2. E. Antonello and G. Raffaelli, "An application of discriminant analysis to variable and nonvariable stars", *Publications of the Astronomical Society of the Pacific*, **95**, 82–85, 1983.

   (Multiple Discriminant Analysis is used.)

3. T. Daubos, "Représentation paramétrique d'images d'objets individuels", Rapport de stage de DEA, Strasbourg Observatory, 54 pp., 1997.

   (A study of galaxy morphologies, using eigengalaxies.)

4. M. Fracassini, L.E. Pasinetti and G. Raffaelli, "Discriminant analysis on pulsar groups in the diagram P versus P", in *Proceedings of a Course and Workshop on Plasma Astrophysics*, European Space Agency Special Publication 207, 315–317, 1984.

5. M. Fracassini, P. Maggi, L.E. Pasinetti and G. Raffaelli, "Pair of pulsars in the diagram P versus P", *Proceedings of the Joint Varenna–Abastumani International School and Workshop on Plasma Astrophysics*, Sukhami, European Space Agency Special Publication 251, 441–445, 1986.

6. A. Heck, "An application of multivariate statistical analysis to a photometric catalogue", *Astronomy and Astrophysics*, **47**, 129–135, 1976.

   (Multiple Discriminant Analysis and a stepwise procedure are applied.)

7. J.F. Jarvis and J.A. Tyson, "FOCAS — Faint object classification and analysis system", *SPIE Instrumentation in Astronomy III*, **172**, 422–428, 1979.

   (See also other references of Tyson/Jarvis and Jarvis/Tyson.)

8. J.F. Jarvis and J.A. Tyson, "Performance verification of an automated image cataloging system", *SPIE Applications of Digital Image Processing to Astronomy*, **264**, 222–229, 1980.

9. J.F. Jarvis and J.A. Tyson, "FOCAS — Faint object classification and analysis system", *The Astronomical Journal*, **86**, 1981, 476–495.

   (A hyperplane separation surface is determined in a space defined by 6 parameters used to characterise the objects. This is a 2–stage procedure where the first stage is that of training, and the second stage uses a partitioning clustering method.)

10. M.J. Kurtz, "Progress in automation techniques for MK classification", in ed. R.F. Garrison, *The MK Process and Stellar Classification*, David Dunlap Observatory, University of Toronto, 1984, pp. 136–152.

    (Essentially a k–NN approach is used for assigning spectra to known stellar spectral classes.)

11. H.T. MacGillivray, R. Martin, N.M. Pratt, V.C. Reddish, H. Seddon, L.W.G. Alexander, G.S. Walker, P.R. Williams, "A method for the automatic separation of the images of galaxies and stars from measurements made with the COSMOS machine", *Monthly Notices of the Royal Astronomical Society*, **176**, 265–274, 1976.

    (Different parameters are appraised for star/galaxy separation. Kurtz — see reference in Chapter 3 (Cluster Analysis) — lists other parameters which have been used for the same objective.)

12. M.L. Malagnini, "A classification algorithm for star–galaxy counts", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 1983, pp. 69–72.

    (A linear classifier is used and is further employed in the following reference.)

13. M.L. Malagnini, F. Pasian, M. Pucillo and P. Santin, "FODS: a system for faint object detection and classification in astronomy", *Astronomy and Astrophysics*, **144**, 1985, 49–56.

14. "Recommendations for Guide Star Selection System", GSSS Group documentation, Space Telescope Science Institute, Baltimore, 1984.

    (A Bayesian approach, using the IMSL subroutine library — see below — is employed in the GSSS system.)

15. W.J. Sebok, "Optimal classification of images into stars or galaxies — a Bayesian approach", *The Astronomical Journal*, **84**, 1979, 1526–1536.

    (The design of a classifier, using galaxy models, is studied in depth and validated on Schmidt plate data.)

16. J.A. Tyson and J.F. Jarvis, "Evolution of galaxies: automated faint object counts to 24th magnitude", *The Astrophyiscal Journal*, **230**, 1979, L153–L156.

    (A continuation of the work of Jarvis and Tyson, 1979, above.)

17. F. Valdes, "Resolution classifier", *SPIE Instrumentation in Astronomy IV*, **331**, 1982, 465–471.

(A Bayesian classifier is used, which differs from that used by Sebok, referenced above. The choice is thoroughly justified. A comparison is also made with the hyperplane fitting method used in the FOCAS system — see the references of Jarvis and Tyson. It is concluded that the results obtained within the model chosen are better than a hyperplane based approach in parameter space; but that the latter is computationally more efficient.)

4.3.3 General References

In the following, some software packages are included. The accompanying documentation often constitutes a quick and convenient way to get information on analysis methods.

1. S. Aeberhard, D. Coomans, and O. de Vel. Comparative Analysis of Statistical Pattern Recognition Methods in High Dimensional Settings. *Pattern Recognition*, Vol. 27, No. 8, 1994.

2. A.K. Agrawala (ed.). *Machine Recognition of Patterns.* IEEE Press. (Collection of key papers and a tutorial), 1974.

3. J.V. Beck, and K.J. Arnold. *Parameter Estimation in Engineering and Science.* John Wiley and Sons, 1977.

4. S.–T. Bow, *Pattern Recognition*, Marcel Dekker, New York, 1984.

   (A textbook detailing a range of Discriminant Analysis methods, together with clustering and other topics.)

5. J.G. Campbell. 1978. Linear Transformations in Pattern Recognition. Plessey Co. Havant U.K. unpublished memorandum.

6. J.G. Campbell and A.A. Hashim. Fuzzy Sets, Pattern Recognition, Linear estimation, and Neural Networks - A Unification of the Theory with Relevance to Remote Sensing. in A.P. Cracknell, and R.A. Vaughan (eds.) *Proc. 18th Annual Conf. of the Remote Sensing Society*, The Remote Sensing Society, 1992.

7.

8. C. Chatfield and A.J. Collins, *Introduction to Multivariate Analysis*, Chapman and Hall, London, 1980.

   (A good introductory textbook.)

9. E. Diday, J. Lemaire, J. Pouget and F. Testu, *Eléments d'Analyse de Données*, Dunod, Paris, 1982.

   (Describes a large range of methods.)

10. R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

    (Excellent treatment of many image processing problems.)

11. R.A. Fisher, “The use of multiple measurements in taxonomic problems”, *The Annals of Eugenics*, **7**, 179–188, 1936.

    (Still an often referenced paper; contains the famous Iris data.)

12. D.H. Foley, and J.W. Sammon. An Optimal Set of Discriminant Vectors. *IEEE Trans. Comp.* March 1975

13. K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, New York, 1972.

14. D.J. Hand, *Discrimination and Classification*, Wiley, New York, 1981.

    (A comprehensive description of a wide range of methods; very recommendable.)

15. International Mathematical and Statistical Library (IMSL), Manual sections on ODFISH, ODNORM, etc.

    (A useful range of algorithms is available in this widely used subroutine library.)

16. M. James, *Classification Algorithms*, Collins, London, 1985.

    (A readable introduction.)

17. M.G. Kendall, *Multivariate Analysis*, Griffin, London, 1980 (2nd ed.).

    (Dated in relation to computing techniques, but exceptionally clear and concise in its treatment of many practical problems.)

18. P.A. Lachenbruch, *Discriminant Analysis*, Hafner Press, New York, 1975.

19. J.L. Melsa and D.L. Cohn, *Decision and Estimation Theory*, McGraw–Hill, New York, 1978.

    (A readable decision theoretic perspective.)

20. W.K. Pratt. *Digital Image Processing.* 2nd. ed. Wiley- Interscience, 1991.

21. J.M. Romeder, *Méthodes et Programmes d’Analyse Discriminante*, Dunod, Paris, 1973.

    (A survey of commonly–used techniques.)

22. Statistical Analysis System (SAS), SAS Institute Inc., Box 8000, Cary, NC 27511–8000, USA; Manual chapters on STEPDISC, NEIGHBOUR, etc.

    (A range of relevant algorithms is available in this, — one of the premier statistical packages.)

23. C.W. Therrien. *Decision Estimation and Classification.* John Wiley and Sons, 1989.

24. M. Turk and A. Pentland. Eigenfaces for Recognition. *J. Cognitive Neuroscience.* Vol. 3, No. 1, 1991.

4.4 Software and Sample Implementation

4.4.1 Program Listing: Linear Discriminant Analysis

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Carry out a LINEAR DISCRIMINANT ANALYSIS, assigning ungrouped
C  items to the closest group centre, using the Mahalanobis
C  distance.
C
C
C  To call:   CALL LDA(N,M,DATA,GP,IPRINT,MEAN,MGP,TOTAL,DIFFVC,
C                                 W1,W2,IERR)     where
C
C
C  N, M  : integer dimensions of ...
C  DATA  : input data (real).
C          On output the first column of DATA contains projections
C          of all N items on the line connecting the two group
C          means. Zero is the boundary point.
C  GP    : Integer vector of length N giving group assignments.  An
C          unassigned item has group 0.  Otherwise groups 1 and 2
C          will be looked for, and other values here are not
C          acceptable.
C  IPRINT: integer; print options (= 3: full; otherwise none).
C  MEAN  : real vector of length M (number of attributes or
C          variables).
C  MGP   : real array of dimensions 2 by M.
C  TOTAL : real array of dims. M by M; on output contains inverse
C          of total variance/covariance matrix.
C  DIFFVC: real vector of length M.
C  W1, W2: real vectors of length M.
C
C
C  Inputs here are N, M, DATA, GP, IPRINT (and IERR).
C  The principle output information is contained in DATA.
C  IERR = 1 means that more than two groups have been specified;
C  IERR = 2 means that the total variance-covariance matrx was
C           singular.
C
C  Note: we require N > M > 2, to prevent the seeking of the
C        inverse of a singular matrix.
C
C-------------------------------------------------------------------
        SUBROUTINE LDA(N,M,DATA,GP,IPRINT,MEAN,MGP,TOTAL,DIFFVC,
     X                                         W1,W2,IERR)
        REAL    DATA(N,M), TOTAL(M,M), MEAN(M), MGP(2,M)
        REAL    DIFFVC(M), W1(M), W2(M)
        INTEGER GP(N), NOG(2), G
C
C          Form global mean.
C
        IERR = 0
        NEFF = 0
        DO 50 I = 1, N
           IF (GP(I).NE.0) NEFF = NEFF + 1
           IF (GP(I).LE.2) GOTO 40
              IERR = 1
              GOTO 9000
   40   CONTINUE
   50   CONTINUE
C
        DO 200 J = 1, M
           MEAN(J) = 0.0
           DO 100 I = 1, N
```

```
              IF (GP(I).NE.0) MEAN(J) = MEAN(J) + DATA(I,J)
  100      CONTINUE
           MEAN(J) = MEAN(J)/FLOAT(NEFF)
  200   CONTINUE
C
C          Form (total) variance-covariance matrix.
C
        DO 500 J1 = 1, M
           DO 400 J2 = 1, M
              TOTAL(J1,J2) = 0.0
              DO 300 I = 1, N
                 IF (GP(I).NE.0) TOTAL(J1,J2) = TOTAL(J1,J2) +
     X            (DATA(I,J1)-MEAN(J1))*(DATA(I,J2)-MEAN(J2))
  300         CONTINUE
              TOTAL(J1,J2) = TOTAL(J1,J2)/FLOAT(NEFF)
  400      CONTINUE
  500   CONTINUE
C
        IMAT = 1
        IF (IPRINT.EQ.3) CALL OUTMAT(IMAT,M,TOTAL)
C
C          Form group means.
C
        DO 700 J = 1, M
           DO 600 K = 1, 2
              MGP(K,J) = 0.0
  600      CONTINUE
  700   CONTINUE
C
        DO 900 I = 1, N
           G = GP(I)
           IF (G.EQ.0) GOTO 900
           NOG(G) = NOG(G) + 1
           DO 800 J = 1, M
              MGP(G,J) = MGP(G,J) + DATA(I,J)
  800      CONTINUE
  900   CONTINUE
C
        DO 1100 K = 1, 2
           DO 1000 J = 1, M
              MGP(K,J) = MGP(K,J)/NOG(K)
 1000      CONTINUE
 1100   CONTINUE
C
C          Invert variance-covariance matrix.
C
        CALL MATINV(M,TOTAL,D,W1,W2)
        IF (D.GT.0.00001) GOTO 1150
           IERR = 2
           GOTO 9000
 1150   CONTINUE
C
C          Form difference vector of group mean vectors.
C
        DO 1200 J = 1, M
           DIFFVC(J) = MGP(1,J) - MGP(2,J)
           MEAN(J) = (MGP(1,J) + MGP(2,J))/2.
 1200   CONTINUE
C
C          Determine projections and output them.
C
        CALL PROJX(N,M,DATA,MEAN,W1,TOTAL,DIFFVC)
        IF (IPRINT.EQ.3) CALL OUTPRX(N,M,DATA)
C
C
C
 9000   CONTINUE
        RETURN
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output a matrix.
C
C------------------------------------------------------------
        SUBROUTINE OUTMAT(IMAT,M,ARRAY)
C
C          Output array.
C
        DIMENSION ARRAY(M,M)
C
        IF (IMAT.EQ.1)WRITE (6,900)
        DO 100 K1 = 1, M
           WRITE (6,1000) (ARRAY(K1,K2),K2=1,M)
  100   CONTINUE
C
  900   FORMAT(' VARIANCE/COVARIANCE MATRIX FOLLOWS.',/)
 1000   FORMAT(10(2X,F8.4))
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Invert a symmetric matrix and calculate its determinant.
C
C
C  To call:      CALL MATINV(M,ARRAY,DET,W1,W2)   where
C
C
C  M        : dimension of ...
C  ARRAY    : input matrix which is replaced by its inverse.
C  NORDER   : degree of matrix (order of determinant)
C  DET      : determinant of input matrix.
C  W1, W2   : work vectors of dimension M.
C
C
C  Reference: Philip B Bevington, "Data Reduction and Error
C             Analysis for the Physical Sciences", McGraw-Hill,
C             New York, 1969, pp. 300-303.
C
C---------------------------------------------------------------
        SUBROUTINE MATINV(M,ARRAY,DET,IK,JK)
        REAL    ARRAY(M,M), IK(M), JK(M)
C
   10   DET = 1.0
   11   DO 100 K = 1, M
C       Find largest element ARRAY(I,J) in rest of matrix.
        AMAX = 0.0
   21      DO 30 I = K, M
              DO 30 J = K, M
   23            IF (ABS(AMAX)-ABS(ARRAY(I,J))) 24,24,30
   24            AMAX = ARRAY(I,J)
                 IK(K) = I
                 JK(K) = J
   30      CONTINUE
C          Interchange rows and columns to put AMAX in ARRAY(K,K).
```

```
   31      IF (AMAX) 41,32,41
   32      DET = 0.0
           GOTO 140
   41      I = IK(K)
           IF (I-K) 21,51,43
   43      DO 50 J = 1, M
              SAVE = ARRAY(K,J)
              ARRAY(K,J) = ARRAY(I,J)
   50      ARRAY(I,J) = -SAVE
   51      J = JK(K)
           IF (J-K) 21,61,53
   53      DO 60 I = 1, M
              SAVE = ARRAY(I,K)
              ARRAY(I,K) = ARRAY(I,J)
   60      ARRAY(I,J) = -SAVE
C          Accumulate elements of inverse matrix.
   61      DO 70 I = 1, M
              IF (I-K) 63,70,63
   63         ARRAY(I,K) = -ARRAY(I,K)/AMAX
   70      CONTINUE
   71      DO 80 I = 1, M
              DO 80 J = 1, M
                 IF (I-K) 74,80,74
   74            IF (J-K) 75,80,75
   75            ARRAY(I,J) = ARRAY(I,J) + ARRAY(I,K)*ARRAY(K,J)
   80      CONTINUE
   81      DO 90 J = 1, M
              IF (J-K) 83,90,83
   83         ARRAY(K,J) = ARRAY(K,J)/AMAX
   90      CONTINUE
           ARRAY(K,K) = 1.0/AMAX
  100   DET = DET * AMAX
C       Restore ordering of matrix.
  101   DO 130 L = 1, M
           K = M - L + 1
           J = IK(K)
           IF (J-K) 111,111,105
  105      DO 110 I = 1, M
              SAVE = ARRAY(I,K)
              ARRAY(I,K) = -ARRAY(I,J)
  110      ARRAY(I,J) = SAVE
  111      I = JK(K)
           IF (I-K) 130,130,113
  113      DO 120 J = 1, M
              SAVE = ARRAY(K,J)
              ARRAY(K,J) = -ARRAY(I,J)
  120      ARRAY(I,J) = SAVE
  130   CONTINUE
  140   RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output projections of row points.
C
C-------------------------------------------------------------
        SUBROUTINE OUTPRX(N,M,PRJN)
        REAL    PRJN(N,M)
C
C
        NUM = 1
        WRITE (6,1000)
        WRITE (6,1010)
        WRITE (6,1020)
        DO 100 K = 1, N
           WRITE (6,1030) K,(PRJN(K,J),J=1,NUM)
  100   CONTINUE
C
 1000   FORMAT(1H0,'PROJECTIONS OF ROW-POINTS FOLLOW.',/)
 1010   FORMAT(' OBJECT   PROJN')
 1020   FORMAT(' ------  ------')
 1030   FORMAT(I5,2X,F8.4)
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Form projections of row-points on factors.
C
C---------------------------------------------------------------
        SUBROUTINE PROJX(N,M,DATA,MEAN,VEC,TOTINV,DIFF)
        REAL    DATA(N,M), MEAN(M), VEC(M), TOTINV(M,M), DIFF(M)
C
        NUM = 1
        DO 300 K = 1, N
           DO 50 L = 1, M
              VEC(L) = DATA(K,L)
   50      CONTINUE
           DO 200 I = 1, NUM
              DATA(K,I) = 0.0
              DO 100 J1 = 1, M
                 DO 75 J2 = 1, M
                    DATA(K,I) = DATA(K,I) + (VEC(J1)-MEAN(J1))*
     X                          TOTINV(J1,J2)*DIFF(J2)
   75            CONTINUE
  100         CONTINUE
  200      CONTINUE
  300   CONTINUE
C
        RETURN
        END
```

4.4.2 Program Listing: Multiple Discriminant Analysis

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Carry out a MULTIPLE DISCRIMINANT ANALYSIS
C              (DISCRIMINANT FACTOR ANALYSIS,
C               CANONICAL DISCRININANT ANALYSIS).
C
C
C  To call:   CALL MDA(N,M,NG,DATA,GP,IPRINT,NOG,MEAN,MGP,TOTAL,
C                      BETWEE,BETW2,CPROJ,W1,W2,IERR)     where
C
C
C  N, M  : integer dimensions of ...
C  DATA  : input data (real).
C          On output the first NG-1 columns of DATA contain projns.
C          of the N items on the discriminant factors.
C  NG    : (integer) number of groups.
C  GP    : Integer vector of length N giving group assignments.
C          Must be specified correctly - no 0s or values > NG.
C  IPRINT: integer; print options (= 3: full; otherwise none).
C  NOG   : integer vect. of len. NG (to contain gp. cardinalities).
C  MEAN  : real vector of length M (number of attributes or vbes).
C  MGP   : real array of dimensions 2 by M.
C  TOTAL : real array of dims. M by M; on output contains inverse
C          of total variance/covariance matrix.
C  BETWEE: real array of dimensions NG by NG.
C  BETW2 : real array of dimensions NG by NG.
C  CPROJ : real array of dimensions M by NG; on output contains the
C          coefficients of the discriminant factors in terms of the
C          original variables.
C  W1, W2: real vectors of length M.
C  IERR  : initially 0; = 1 if there is no convergence in the TQL2
C          eigenroutine; = 2 if the total variance-covariance to be
C          inverted is singular, - in this case, check that there
C          are no columns with identical values, that N > M > NG,
C          etc.
C
C  Inputs here are N, M, NG, DATA, GP, IPRINT (and IERR).
C  The principle output information is contained in DATA and CPROJ;
C  and W1(NG-1), W1(NG-2) ... contain the eigenvalues in decreasing
C  order.
C
C  Notes: we require that N > M > NG (otherwise, an error is likely
C         in the matrix inversion routine due to singularity).
C         NG-1 eigenvalues eigenvectors are output.
C
C------------------------------------------------------------------
        SUBROUTINE MDA(N,M,NG,DATA,GP,IPRINT,NOG,MEAN,MGP,TOTAL,
     X                 BETWEE,BETW2,CPROJ,W1,W2,IERR)
        REAL    DATA(N,M), TOTAL(M,M), MEAN(M), MGP(NG,M)
        REAL    W1(M), W2(M), BETW2(NG,NG), CPROJ(M,NG)
        REAL    BETWEE(NG,NG)
        INTEGER GP(N), NOG(NG), G
C
C          Form global mean.
C
        DO 200 J = 1, M
           MEAN(J) = 0.0
           DO 100 I = 1, N
              MEAN(J) = MEAN(J) + DATA(I,J)
  100      CONTINUE
           MEAN(J) = MEAN(J)/FLOAT(N)
  200   CONTINUE
C
C          Form (total) variance-covariance matrix.
C
        DO 500 J1 = 1, M
           DO 400 J2 = 1, M
              TOTAL(J1,J2) = 0.0
              DO 300 I = 1, N
                 TOTAL(J1,J2) = TOTAL(J1,J2) +
     X           (DATA(I,J1)-MEAN(J1))*(DATA(I,J2)-MEAN(J2))
  300         CONTINUE
              TOTAL(J1,J2) = TOTAL(J1,J2)/FLOAT(N)
  400      CONTINUE
  500   CONTINUE
C
        IMAT = 1
C       CALL OUTMAT(IMAT,M,TOTAL)
C
C          Form group means.
C
        DO 700 J = 1, M
           DO 600 K = 1, NG
              MGP(K,J) = 0.0
  600      CONTINUE
  700   CONTINUE
C
        DO 900 I = 1, N
           G = GP(I)
           IF (G.EQ.0) GOTO 9000
           NOG(G) = NOG(G) + 1
           DO 800 J = 1, M
              MGP(G,J) = MGP(G,J) + DATA(I,J)
  800      CONTINUE
  900   CONTINUE
C
        DO 1100 K = 1, NG
           DO 1000 J = 1, M
              MGP(K,J) = MGP(K,J)/NOG(K)
 1000      CONTINUE
 1100   CONTINUE
C
C          Invert variance-covariance matrix.
C
        CALL MATINV(M,TOTAL,D,W1,W2)
        IF (D.GT.0.000001) GOTO 1150
           IERR = 2
           GOTO 9000
 1150   CONTINUE
        IMAT = 2
C       CALL OUTMAT(IMAT,M,TOTAL)
C
C          Form the symmetric variant of the BETWEE-groups
C          variance-covariance matrix for diagonalization.
C
        DO 1200 K1 = 1, NG
         DO 1200 K2 = 1, NG
         BETWEE(K1,K2) = 0.0
          DO 1200 J1 = 1, M
           DO 1200 J2 = 1, M
            D1 = MGP(K1,J1) - MEAN(J1)
            D2 = FLOAT(NOG(K1))/FLOAT(N)
            D2 = SQRT(D2)
            D3 = MGP(K2,J2) - MEAN(J2)
            D4 = FLOAT(NOG(K2))/FLOAT(N)
            D4 = SQRT(D4)
            BETWEE(K1,K2) = BETWEE(K1,K2) +
     X                       (D1*D2)*TOTAL(J1,J2)*(D3*D4)
 1200   CONTINUE
C
        IMAT = 4
C       CALL OUTMAT(IMAT,M,TOTAL)
C
C          Carry out eigenreduction.
C
        NG2 = NG
        CALL TRED2(NG,NG2,BETWEE,W1,W2,BETW2)
        CALL TQL2(NG,NG2,W1,W2,BETW2,IERR)
        IF (IERR.NE.0) GOTO 9000
C
C          Output eigenvalues and eigenvectors.
C
        IF (IPRINT.GT.1) CALL OUTEVL(N,M,NG,W1)
        IF (IPRINT.GT.1) CALL OUTEVC(N,M,NG,BETW2,NG-1)
C
C          Convert eigenvectors in NG-space to those in M-space.
C
        DO 1300 J = 1, M
           DO 1300 K = 1, NG
              CPROJ(J,K) = 0.0
              DO 1300 J2 = 1, M
                 DO 1300 K2 = 1, NG
                    D1 = MGP(K2,J2) - MEAN(J2)
                    D2 = FLOAT(NOG(K2))/FLOAT(N)
                    D1 = D1*SQRT(D2)
                    CPROJ(J,K)=CPROJ(J,K)+
     X              TOTAL(J,J2)*D1*BETW2(K2,NG-K+1)
 1300   CONTINUE
        IF (IPRINT.GT.1) CALL OUTEVC(N,NG,M,CPROJ,NG-1)
C
C          Determine projections and output them.
C
        CALL PROJX(N,M,NG,DATA,MEAN,CPROJ,W2,TOTAL)
        IF (IPRINT.EQ.3) CALL OUTPRX(N,M,NG,DATA)
C
C
C
 9000   CONTINUE
        RETURN
        END
C-----------------------------------------------------------
        SUBROUTINE OUTMAT(IMAT,M,ARRAY)
        DIMENSION ARRAY(M,M)
C
        WRITE (6,900) IMAT
        DO 100 K1 = 1, M
           WRITE (6,1000) (ARRAY(K1,K2),K2=1,M)
  100   CONTINUE
C
  900   FORMAT(' IMAT =',I6)
 1000   FORMAT(10(2X,F8.4))
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
```

```
C  Invert a symmetric matrix and calculate its determinant.
C
C
C  To call:      CALL MATINV(M,ARRAY,DET,W1,W2)   where
C
C
C  M       : dimension of ...
C  ARRAY   : input matrix which is replaced by its inverse.
C  NORDER  : degree of matrix (order of determinant)
C  DET     : determinant of input matrix.
C  W1, W2  : work vectors of dimension M.
C
C
C  Reference: Philip B Bevington, "Data Reduction and Error
C             Analysis for the Physical Sciences", McGraw-Hill,
C             New York, 1969, pp. 300-303.
C
C-------------------------------------------------------------
       SUBROUTINE MATINV(M,ARRAY,DET,IK,JK)
       REAL    ARRAY(M,M), IK(M), JK(M)
C
   10  DET = 1.0
   11  DO 100 K = 1, M
C      Find largest element ARRAY(I,J) in rest of matrix.
       AMAX = 0.0
   21     DO 30 I = K, M
             DO 30 J = K, M
   23           IF (ABS(AMAX)-ABS(ARRAY(I,J))) 24,24,30
   24           AMAX = ARRAY(I,J)
                IK(K) = I
                JK(K) = J
   30     CONTINUE
C         Interchange rows and columns to put AMAX in ARRAY(K,K).
   31     IF (AMAX) 41,32,41
   32     DET = 0.0
          GOTO 140
   41     I = IK(K)
          IF (I-K) 21,51,43
   43     DO 50 J = 1, M
             SAVE = ARRAY(K,J)
             ARRAY(K,J) = ARRAY(I,J)
   50     ARRAY(I,J) = -SAVE
   51     J = JK(K)
          IF (J-K) 21,61,53
   53     DO 60 I = 1, M
             SAVE = ARRAY(I,K)
             ARRAY(I,K) = ARRAY(I,J)
   60     ARRAY(I,J) = -SAVE
C         Accumulate elements of inverse matrix.
   61     DO 70 I = 1, M
             IF (I-K) 63,70,63
   63        ARRAY(I,K) = -ARRAY(I,K)/AMAX
   70     CONTINUE
   71     DO 80 I = 1, M
             DO 80 J = 1, M
                IF (I-K) 74,80,74
   74           IF (J-K) 75,80,75
   75           ARRAY(I,J) = ARRAY(I,J) + ARRAY(I,K)*ARRAY(K,J)
   80     CONTINUE
   81     DO 90 J = 1, M
             IF (J-K) 83,90,83
   83        ARRAY(K,J) = ARRAY(K,J)/AMAX
   90      CONTINUE
           ARRAY(K,K) = 1.0/AMAX
  100   DET = DET * AMAX
C       Restore ordering of matrix.
  101   DO 130 L = 1, M
           K = M - L + 1
           J = IK(K)
           IF (J-K) 111,111,105
  105      DO 110 I = 1, M
              SAVE = ARRAY(I,K)
              ARRAY(I,K) = -ARRAY(I,J)
  110      ARRAY(I,J) = SAVE
  111      I = JK(K)
           IF (I-K) 130,130,113
  113      DO 120 J = 1, M
              SAVE = ARRAY(K,J)
              ARRAY(K,J) = -ARRAY(I,J)
  120      ARRAY(I,J) = SAVE
  130   CONTINUE
  140   RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C Reduce a real, symmetric matrix to a symmetric, tridiagonal
C matrix.
C
C To call:    CALL TRED2(NM,N,A,D,E,Z)    where
C
C NM = row dimension of A and Z;
C N = order of matrix A (will always be <= NM);
C A = symmetric matrix of order N to be reduced to tridiagonal
C     form;
C D = vector of dim. N containing, on output, diagonal elements of
C     tridiagonal matrix;
C E = working vector of dim. at least N-1 to contain subdiagonal
C     elements;
C Z = matrix of dims. NM by N containing, on output, orthogonal
C     transformation matrix producting the reduction.
C
C Normally a call to TQL2 will follow the call to TRED2 in order
C to produce all eigenvectors and eigenvalues of matrix A.
C
C Algorithm used: Martin et al., Num. Math. 11, 181-195, 1968.
C
C Reference: Smith et al., Matrix Eigensystem Routines - EISPACK
C Guide, Lecture Notes in Computer Science 6, Springer-Verlag,
C 1976, pp. 489-494.
C
C-----------------------------------------------------------------
        SUBROUTINE TRED2(NM,N,A,D,E,Z)
C
        REAL A(NM,N),D(N),E(N),Z(NM,N)
C
        DO 100 I = 1, N
           DO 100 J = 1, I
              Z(I,J) = A(I,J)
  100   CONTINUE
        IF (N.EQ.1) GOTO 320
        DO 300 II = 2, N
           I = N + 2 - II
           L = I - 1
           H = 0.0
           SCALE = 0.0
           IF (L.LT.2) GOTO 130
           DO 120 K = 1, L
              SCALE = SCALE + ABS(Z(I,K))
  120      CONTINUE
           IF (SCALE.NE.0.0) GOTO 140
  130      E(I) = Z(I,L)
           GOTO 290
  140      DO 150 K = 1, L
              Z(I,K) = Z(I,K)/SCALE
              H = H + Z(I,K)*Z(I,K)
  150      CONTINUE
C
           F = Z(I,L)
           G = -SIGN(SQRT(H),F)
           E(I) = SCALE * G
           H = H - F * G
           Z(I,L) = F - G
           F = 0.0
C
           DO 240 J = 1, L
              Z(J,I) = Z(I,J)/H
              G = 0.0
C             Form element of A*U.
              DO 180 K = 1, J
                 G = G + Z(J,K)*Z(I,K)
  180         CONTINUE
              JP1 = J + 1
              IF (L.LT.JP1) GOTO 220
              DO 200 K = JP1, L
                 G = G + Z(K,J)*Z(I,K)
  200         CONTINUE
C             Form element of P where P = I - U U' / H .
  220         E(J) = G/H
              F = F + E(J) * Z(I,J)
  240      CONTINUE
           HH = F/(H + H)
C          Form reduced A.
           DO 260 J = 1, L
              F = Z(I,J)
              G = E(J) - HH * F
              E(J) = G
              DO 250 K = 1, J
                 Z(J,K) = Z(J,K) - F*E(K) - G*Z(I,K)
  250         CONTINUE
  260      CONTINUE
  290      D(I) = H
  300   CONTINUE
  320   D(1) = 0.0
        E(1) = 0.0
C       Accumulation of transformation matrices.
        DO 500 I = 1, N
           L = I - 1
           IF (D(I).EQ.0.0) GOTO 380
           DO 360 J = 1, L
              G = 0.0
              DO 340 K = 1, L
                 G = G + Z(I,K) * Z(K,J)
  340         CONTINUE
              DO 350 K = 1, L
                 Z(K,J) = Z(K,J) - G * Z(K,I)
  350         CONTINUE
  360      CONTINUE
  380      D(I) = Z(I,I)
           Z(I,I) = 1.0
           IF (L.LT.1) GOTO 500
           DO 400 J = 1, L
              Z(I,J) = 0.0
              Z(J,I) = 0.0
  400      CONTINUE
  500   CONTINUE
C
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C Determine eigenvalues and eigenvectors of a symmetric,
C tridiagonal matrix.
C
C To call:    CALL TQL2(NM,N,D,E,Z,IERR)    where
C
C NM = row dimension of Z;
C N = order of matrix Z;
C D = vector of dim. N containing, on output, eigenvalues;
C E = working vector of dim. at least N-1;
C Z = matrix of dims. NM by N containing, on output, eigenvectors;
C IERR = error, normally 0, but 1 if no convergence.
C
C Normally the call to TQL2 will be preceded by a call to TRED2 in
C order to set up the tridiagonal matrix.
C
C Algorithm used: QL method of Bowdler et al., Num. Math. 11,
C 293-306, 1968.
C
C Reference: Smith et al., Matrix Eigensystem Routines - EISPACK
C Guide, Lecture Notes in Computer Science 6, Springer-Verlag,
C 1976, pp. 468-474.
C
C-----------------------------------------------------------------
        SUBROUTINE TQL2(NM,N,D,E,Z,IERR)
C
        REAL    D(N), E(N), Z(NM,N)
        DATA    EPS/1.E-12/
C
        IERR = 0
        IF (N.EQ.1) GOTO 1001
        DO 100 I = 2, N
           E(I-1) = E(I)
  100   CONTINUE
        F = 0.0
        B = 0.0
        E(N) = 0.0
C
        DO 240 L = 1, N
           J = 0
           H = EPS * (ABS(D(L)) + ABS(E(L)))
           IF (B.LT.H) B = H
C          Look for small sub-diagonal element.
           DO 110 M = L, N
              IF (ABS(E(M)).LE.B) GOTO 120
C             E(N) is always 0, so there is no exit through the
C             bottom of the loop.
  110      CONTINUE
  120      IF (M.EQ.L) GOTO 220
  130       IF (J.EQ.30) GOTO 1000
            J = J + 1
C           Form shift.
            L1 = L + 1
            G = D(L)
            P = (D(L1)-G)/(2.0*E(L))
            R = SQRT(P*P+1.0)
            D(L) = E(L)/(P+SIGN(R,P))
            H = G-D(L)
C
            DO 140 I = L1, N
               D(I) = D(I) - H
  140       CONTINUE
C
            F = F + H
C           QL transformation.
            P = D(M)
            C = 1.0
            S = 0.0
            MML = M - L
C
            DO 200 II = 1, MML
               I = M - II
               G = C * E(I)
               H = C * P
               IF (ABS(P).LT.ABS(E(I))) GOTO 150
               C = E(I)/P
               R = SQRT(C*C+1.0)
               E(I+1) = S * P * R
               S = C/R
               C = 1.0/R
               GOTO 160
  150          C = P/E(I)
               R = SQRT(C*C+1.0)
               E(I+1) = S * E(I) * R
               S = 1.0/R
               C = C * S
  160          P = C * D(I) - S * G
               D(I+1) = H + S * (C * G + S * D(I))
C              Form vector.
               DO 180 K = 1, N
                  H = Z(K,I+1)
                  Z(K,I+1) = S * Z(K,I) + C * H
                  Z(K,I) = C * Z(K,I) - S * H
  180          CONTINUE
  200       CONTINUE
            E(L) = S * P
            D(L) = C * P
            IF (ABS(E(L)).GT.B) GOTO 130
  220       D(L) = D(L) + F
  240    CONTINUE
C
C        Order eigenvectors and eigenvalues.
         DO 300 II = 2, N
            I = II - 1
            K = I
            P = D(I)
            DO 260 J = II, N
               IF (D(J).GE.P) GOTO 260
               K = J
               P = D(J)
  260       CONTINUE
```

```
           IF (K.EQ.I) GOTO 300
           D(K) = D(I)
           D(I) = P
           DO 280 J = 1, N
              P = Z(J,I)
              Z(J,I) = Z(J,K)
              Z(J,K) = P
  280      CONTINUE
  300   CONTINUE
C
        GOTO 1001
C       Set error - no convergence after 30 iterations.
 1000   IERR = 1
 1001   RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output eigenvalues in order of decreasing value.
C
C-------------------------------------------------------------
        SUBROUTINE OUTEVL(N,M,NG,VALS)
        DIMENSION       VALS(NG)
C
        TOT = 0.0
        DO 100 K = 2, NG
           TOT = TOT + VALS(K)
  100   CONTINUE
C
        WRITE (6,1000)
        CUM = 0.0
        K = NG + 1
        WRITE (6,1010)
        WRITE (6,1020)
  200   CONTINUE
        K = K - 1
        CUM = CUM + VALS(K)
        VPC = VALS(K) * 100.0 / TOT
        VCPC = CUM * 100.0 / TOT
        WRITE (6,1030) VALS(K),VPC,VCPC
        IF (K.GT.2) GOTO 200
C
        RETURN
 1000 FORMAT(1H0,'EIGENVALUES FOLLOW.',/)
 1010 FORMAT
     X(' Eigenvalues       As Percentages    Cumul. Percentages')
 1020 FORMAT
     X(' -----------       --------------    ------------------')
 1030 FORMAT(F10.4,9X,F10.4,10X,F10.4)
        END
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output FIRST SEVEN eigenvectors associated with eigenvalues in
C  decreasing order.
C
C----------------------------------------------------------------
        SUBROUTINE OUTEVC(N1,N2,N3,VECS,N4)
        DIMENSION       VECS(N3,N3)
C
        NUM = MIN0(N4,7)
        WRITE (6,1000)
        WRITE (6,1010)
        WRITE (6,1020)
```

```
       DO 100 K1 = 1, N3
       WRITE (6,1030) K1,(VECS(K1,K2),K2=1,NUM)
  100  CONTINUE
C
       RETURN
 1000  FORMAT(1H0,'EIGENVECTORS FOLLOW.',/)
 1010  FORMAT
     X  ('  VBLE.   EV-1    EV-2    EV-3    EV-4    EV-5    EV-6
     X   EV-7')
 1020  FORMAT
     X  (' ------  ------  ------  ------  ------  ------  ------
     X------')
 1030  FORMAT(I5,2X,7F8.4)
       END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output projections on discriminant factors.
C
C---------------------------------------------------------------
       SUBROUTINE OUTPRX(N,M,NG,PRJN)
       REAL    PRJN(N,M)
C
       NUM = MIN0(N,M,NG,7)
       WRITE (6,1000)
       WRITE (6,1010)
       WRITE (6,1020)
       DO 100 K = 1, N
          WRITE (6,1030) K,(PRJN(K,J),J=1,NUM-1)
  100  CONTINUE
C
 1000  FORMAT(1H0,'PROJECTIONS OF ROW-POINTS FOLLOW.',/)
 1010  FORMAT
     X  (' OBJECT  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5  PROJ-6
     X  PROJ-7')
 1020  FORMAT
     X  (' ------  ------  ------  ------  ------  ------  ------
     X  ------')
 1030  FORMAT(I5,2X,7F8.4)
       RETURN
       END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Output projections of column points on up to first 7
C  discriminant axes.
C
C---------------------------------------------------------------
       SUBROUTINE OUTPRY(N,M,NG,PRJNS)
       REAL    PRJNS(NG,NG)
C
       NUM = MIN0(N,M,MG,7)
       WRITE (6,1000)
       WRITE (6,1010)
       WRITE (6,1020)
       DO 100 K = 1, M
          WRITE (6,1030) K,(PRJNS(K,J),J=1,NUM)
  100  CONTINUE
C
 1000  FORMAT(1H0,'PROJECTIONS OF COLUMN-POINTS FOLLOW.',/)
 1010  FORMAT
     X  ('  VBLE.  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5  PROJ-6
     X  PROJ-7')
 1020  FORMAT
```

```
     X  (' ------  ------  ------  ------  ------  ------  ------
     X  ------')
 1030   FORMAT(I5,2X,7F8.4)
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Form projections of row-points on (up to) first 7 factors.
C
C---------------------------------------------------------------
        SUBROUTINE PROJX(N,M,NG,DATA,MEAN,EVEC,VEC,TOTINV)
        REAL    DATA(N,M), EVEC(M,M), VEC(M), TOTINV(M,M), MEAN(M)
C
        NUM = MIN0(N,M,NG,7)
        DO 300 K = 1, N
           DO 50 L = 1, M
              VEC(L) = DATA(K,L)
   50      CONTINUE
           DO 200 I = 1, NUM
              DATA(K,I) = 0.0
              DO 100 J1 = 1, M
C                DO 75 J2 = 1, M
                    DATA(K,I) = DATA(K,I) + (VEC(J1) - MEAN(J1))*
     X                         EVEC(J1,I)
   75            CONTINUE
  100         CONTINUE
  200      CONTINUE
  300   CONTINUE
C
        RETURN
        END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Determine projections of column points on (up to) 7 factors.
C
C---------------------------------------------------------------
        SUBROUTINE PROJY(N,M,NG,EVALS,A,Z,VEC)
        REAL    EVALS(M), A(M,M), Z(M,M), VEC(M)
C
        NUM = MIN0(N,M,NG,7)
        DO 300 J1 = 1, M
           DO 50 L = 1, M
              VEC(L) = A(J1,L)
   50      CONTINUE
           DO 200 J2 = 1, NUM
              A(J1,J2) = 0.0
              DO 100 J3 = 1, M
                 A(J1,J2) = A(J1,J2) + VEC(J3)*Z(J3,M-J2+1)
  100         CONTINUE
              IF (EVALS(M-J2+1).GT.0.0) A(J1,J2) =
     X           A(J1,J2)/SQRT(EVALS(M-J2+1))
              IF (EVALS(M-J2+1).EQ.0.0) A(J1,J2) = 0.0
  200      CONTINUE
  300   CONTINUE
C
        RETURN
        END
```

4.4.3 Program Listing: K–NNs Discriminant Analysis

```
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
C  Carry out a K-NN DISCRIMINANT ANALYSIS,
C  with two groups defined in a training set, and with
C  assignment of members of a test set.
C
C  Parameters:
C
C  TRAIN(N,M)     training set, where first N1 rows relate to the
C                 first group, and the next N2 rows to the second
C                 group.  Must have N1 + N2 = N.
C  TEST(N3,M)     test set;
C  K              number of nearest neighbours to consider;
C  KLIST(K), DK(K), KPOP(K)   are used for storing the K NNs,
C                 their distances to the object under consider-
C                 ation, and the group to which the NNs belong.
C
C-----------------------------------------------------------------
        SUBROUTINE KNN(TRAIN,N1,N2,N,TEST,N3,M,K,KLIST,DK,KPOP)
        DIMENSION TRAIN(N,M),TEST(N3,M)
        DIMENSION KLIST(K),DK(K),KPOP(K)
C
        WRITE(6,22)
        WRITE(6,33)
C
        DO 90 I = 1, N3
        DO 10 IX = 1, K
        KLIST(IX) = 0
        DK(IX) = 1.E+15
        KPOP(IX) = 0
   10   CONTINUE
        IND = 0
                DO 38 I2 = 1, N1
                CALL DIST(I2,TRAIN,I,TEST,N,N3,M,D)
                IF (IND.LE.0) GOTO 35
                   DO 30 IK = 1, K
                        IF (D.GE.DK(IK)) GOTO 28
                           IF (IK.GE.K) GOTO 25
                              DO 20 IK2 = K, IK+1, -1
                                 DK(IK2) = DK(IK2-1)
                                 KLIST(IK2) = KLIST(IK2-1)
                                 KPOP(IK2) = KPOP(IK2-1)
   20                         CONTINUE
   25                         CONTINUE
                           DK(IK) = D
                           KLIST(IK) = I2
                           KPOP(IK) = 1
                           GOTO 36
   28               CONTINUE
   30               CONTINUE
                    GOTO 36
   35           CONTINUE
                    IND = IND + 1
                    DK(IND) = D
                    KLIST(IND) = I2
                    KPOP(IND) = 1
   36           CONTINUE
   38           CONTINUE
C
C
C
```

```
                DO 68 I2 = N1+1, N
                CALL DIST(I2,TRAIN,I,TEST,N,N3,M,D)
                IF (IND.LE.0) GOTO 65
                   DO 60 IK = 1, K
                        IF (D.GE.DK(IK)) GOTO 58
                           IF (IK.GE.K) GOTO 55
                              DO 50 IK2 = K, IK+1, -1
                                 DK(IK2) = DK(IK2-1)
                                 KLIST(IK2) = KLIST(IK2-1)
                                 KPOP(IK2) = KPOP(IK2-1)
   50                         CONTINUE
   55                         CONTINUE
                           DK(IK) = D
                           KLIST(IK) = I2
                           KPOP(IK) = 2
                           GOTO 66
   58               CONTINUE
   60               CONTINUE
                    GOTO 66
   65           CONTINUE
                    IND = IND + 1
                    DK(IND) = D
                    KLIST(IND) = I2
                    KPOP(IND) = 2
   66           CONTINUE
   68           CONTINUE
C
        NUM1 = 0
        NUM2 = 0
        DO 80 IX = 1, K
           IF (KPOP(IX).EQ.1) NUM1 = NUM1 + 1
           IF (KPOP(IX).EQ.2) NUM2 = NUM2 + 1
   80   CONTINUE
C       (Error check:)
        IF ((NUM1+NUM2).EQ.K) GOTO 85
           WRITE (6,600)
           STOP
   85   CONTINUE
        IF (NUM1.GT.NUM2) WRITE (6,500) I,FLOAT(NUM1)*100./FLOAT(K)
        IF (NUM2.GT.NUM1) WRITE (6,525) I,FLOAT(NUM2)*100./FLOAT(K)
        IF (NUM1.EQ.NUM2) WRITE (6,550) I,FLOAT(NUM1)*100./FLOAT(K)
   90   CONTINUE
C
        RETURN
   22   FORMAT(' Object  -->  group with probability')
   33   FORMAT(/)
  500   FORMAT(I6,6X,'    1   ',F8.2,'%')
  525   FORMAT(I6,6X,'    2   ',F8.2,'%')
  550   FORMAT(I6,6X,' 1 or 2 ',F8.2,'%   (equiprobable).')
  600   FORMAT(' The total of assignments to gp. 1 and to gp. 2',
     X          'does not equal K; check pgm. listing; aborting.')
        END
C-------------------------------------------------------------------------
        SUBROUTINE DIST(I,ARR1,J,ARR2,NR1,NR2,NC,D)
        DIMENSION ARR1(NR1,NC),ARR2(NR2,NC)
C
        D = 0.0
        DO 10 K = 1, NC
        D = D + (ARR1(I,K)-ARR2(J,K))**2
   10   CONTINUE
C
        RETURN
```

```
END
```

4.4.4 Input Data

The input data was based on that used for Principal Components Analysis. PCA indicated very correlated data: for this reason the discriminating potential of variables 1, 2 and 18 only were investigated. The class assignments used were as follows. For LDA, the first seven objects (rows) constituted group 1, and the remaining objects constituted group 2. For MDA, three groups were used: objects 1 to 4, objects 5 to 14, and objects 15 to 18. For KNN, the first 10 objects were taken as group 1 and the remainder as group 2. The training set and the test set in the case of this last method were the same.

4.4.5 Sample Output: Linear Discriminant Analysis

```
PROJECTIONS OF ROW-POINTS FOLLOW.


 OBJECT   PROJN
 ------  ------
    1    2.1423
    2    1.6503
    3    2.2293
    4    1.3423
    5    3.5538
    6    1.7404
    7   -0.2216
    8   -0.5764
    9   -1.4843
   10   -1.6291
   11   -1.4359
   12   -1.9213
   13   -2.0746
   14   -2.1384
   15   -2.0838
   16   -2.0498
   17   -2.0950
   18   -2.0546
```

4.4.6 Sample Output: Multiple Discriminant Analysis

```
 EIGENVALUES FOLLOW.


 Eigenvalues        As Percentages    Cumul. Percentages
 -----------        --------------    ------------------
    0.6985             57.0614             57.0614
    0.5256           42.9386           100.0000




EIGENVECTORS (in the group space) FOLLOW.



  VBLE.   EV-1    EV-2    EV-3    EV-4    EV-5    EV-6    EV-7
 ------  ------  ------  ------  ------  ------  ------ ------
    1   -0.4714  0.6515
    2   -0.7454 -0.6546
    3   -0.4714  0.3835




EIGENVECTORS (in the parameter space) FOLLOW.



  VBLE.   EV-1    EV-2    EV-3    EV-4    EV-5    EV-6    EV-7
 ------  ------  ------  ------  ------  ------  ------ ------
    1   -0.0034  0.0015
    2   -0.0474 -0.1074
    3    0.0358  0.0487




 PROJECTIONS OF ROW-POINTS FOLLOW.



 OBJECT  PROJ-1  PROJ-2  PROJ-3  PROJ-4  PROJ-5  PROJ-6  PROJ-7
 ------  ------  ------  ------  ------  ------  ------  ------
    1    1.1862  1.7901
    2    0.9135  0.9508
    3    0.8789  0.5212
    4    0.5445 -0.3562
    5    0.9826  0.2888
    6    0.5844 -0.3084
    7    0.1839 -0.7774
    8    0.1373 -0.6302
    9   -0.0534 -0.9326
   10   -0.0826 -0.7924
   11   -0.0674 -0.5555
   12   -0.1520 -0.4943
   13   -0.1513 -0.2907
   14   -0.1972 -0.1236
   15   -0.3892  0.0068
   16   -0.7815  0.1176
   17   -1.1404  0.3730
   18   -2.3961  1.2130
```

4.4.7 Sample Output: K–NNs Discriminant Analysis

```
Object  -->  group with probability

    1          1     100.00%
    2          1     100.00%
    3          1     100.00%
    4          1     100.00%
    5          1     100.00%
    6          1     100.00%
    7          1     100.00%
    8          1     100.00%
    9          1     100.00%
   10          1      66.67%
   11          2      66.67%
   12          2     100.00%
   13          2     100.00%
   14          2     100.00%
   15          2     100.00%
   16          2     100.00%
   17          2     100.00%
   18          2     100.00%
```

Chapter 5

Other Methods

5.1 The Problems

Principal Components Analysis (PCA), among whose objectives are dimensionality reduction and the display of data, assumes points in the usual Euclidean space as input. For other types of input data, alternative methods exist. Such other methods may have somewhat different objectives, which may be more relevant for the given type of input.

Correspondence Analysis is particularly suitable for arrays of frequencies or for data in complete disjunctive form (cf. Chapter 1). It may be described as a PCA in a different metric (the χ^2 metric replaces the usual Euclidean metric). Mathematically, it differs from PCA also in that points in multidimensional space are considered to have a mass (or weight) associated with them, at their given locations. The percentage *inertia* explained by axes takes the place of the percentage variance of PCA, — and in the former case the values can be so small that such a figure of merit assumes less importance than in the case of PCA. Correspondence Analysis is a technique in which it is a good deal more difficult to interpret results, but it considerably expands the scope of a PCA–type analysis in its ability to handle a wide range of data.

Principal Coordinates Analysis is very similar to PCA. The problem here is that rather than the usual objects × variables array, we are given an objects × objects distance matrix. A minimal amount of alteration to the approach adopted in PCA allows this type of input data to be handled.

In Canonical Correlation Analysis, the third technique to be looked at in this Chapter, we are given a set of objects (rows) crossed by two distinct sets of variables (columns). The objective is to examine the relationship between the two sets of characterisations of the same object–population. A pair of best–fitting axes are derived in the spaces of the two sets of variables, such that in addition these two axes are optimally correlated. Successive axes are subsequently obtained. Canonical Correlation Analysis is difficult to use in practice when the two sets of variables are not highly related.

Regression Analysis, the final topic of this Chapter, is widely dealt with elsewhere in the physical and statistical literature (see in particular Lutz, 1983). For completeness in the range of important multivariate methods studied, we introduce it and discuss a few points of relevance to astronomy. Relevant bibliographic references in astronomy in this Chapter primarily relate to regression.

5.2 Correspondence Analysis

5.2.1 Introduction

Correspondence Analysis (CA) is not unlike PCA in its underlying geometrical bases, and the description to follow will adopt a similar perspective to that used in Chapter 2. While PCA is particularly suitable for quantitative data, CA is recommendable for the following types of input data, which will subsequently be looked at more closely: frequencies, contingency tables, probabilities, categorical data, and mixed qualitative/categorical data.

In the case of *frequencies* (i.e. the ij^{th} table entry indicates the frequency of occurrence of attribute j for object i) the row and column "profiles" are of interest. That is to say, the relative magnitudes are of importance. Use of a weighted Euclidean distance, termed the χ^2 distance, gives a zero distance for example to the following 5–coordinate vectors which have identical *profiles* of values: (2,7,0,3,1) and (8,28,0,12,4). Probability type values can be constructed here by dividing each value in the vectors by the sum of the respective vector values.

A particular type of frequency of occurrence data is the *contingency table*, — a table crossing (usually, two) sets of characteristics of the population under study. As an example, an $n \times m$ contingency table might give frequencies of the existence of n different metals in stars of m different ages. CA allows the study of the two sets of variables which constitute the rows and columns of the contingency table. In its usual variant, PCA would privilege either the rows or the columns by standardizing (cf. Section 2.2.5): if, however, we are dealing with a contingency table, both rows and columns are equally interesting. The "standardizing" inherent in CA (a consequence of the χ^2 distance) treats rows and columns in an identical manner. One byproduct is that the row and column projections in the new space may both be plotted on the same output graphic presentations (— the lack of an analogous direct relationship between row projections and column projections in PCA precludes doing this in the latter technique).

Categorical data may be coded by the "scoring" of 1 (presence) or 0 (absence) for each of the possible categories. Such coding leads to *complete disjunctive coding*, as seen in Chapter 1. It will be discussed below how CA of an array of such complete disjunctive data is referred to as Multiple Correspondence Analysis (MCA), and how such a coding of categorical data is, in fact, closely related to contingency table type data.

Dealing with a complex astronomical catalogue may well give rise in practice to a mixture of quantitative (real valued) and qualitative data. One possibility for the analysis of such data is to "discretize" the quantitative values, and treat them thereafter as categorical. In this way a set of variables — many more than the initially given set of variables — which is homogeneous, is analysed.

CA is described initially below with reference to frequency or probability type data as input. We will then look at how the same method is also used for complete disjunctive data.

5.2.2 Properties of Correspondence Analysis

From the initial frequencies data matrix, a set of probability data, x_{ij}, is defined by dividing each value by the grand total of all elements in the matrix. In CA, each row (or column) point is considered to have an associated weight. The weight of the i^{th} row point is given by $x_i = \sum_j x_{ij}$ and the weight of the j^{th} column point is given by $x_j = \sum_i x_{ij}$. We consider the row points to have coordinates x_{ij}/x_i, thus allowing points of the same *profile* to be identical (i.e. superimposed). The following weighted Euclidean distance, the χ^2 distance, is then used between row points:

$$d^2(i,k) = \sum_j \frac{1}{x_j}(\frac{x_{ij}}{x_i} - \frac{x_{kj}}{x_k})^2$$

and an analogous distance is used between column points.

Table 5.1 summarizes the situation in the dual spaces. Item 4 indicates that the Euclidean distance between points of coordinates x_{ij}/x_i and x_{ij}/x_j is not the ordinary Euclidean distance but is instead with respect to the specified weights. Item 5 indicates that the inertia rather than the variance will be studied, i.e. that the masses of the points are incorporated into the criterion to be optimised.

The mean row point is given by the weighted average of all row points:

$$\sum_i x_i \frac{x_{ij}}{x_i} = x_j$$

for $j = 1, 2, \ldots, m$. Similarly the mean column profile has i^{th} coordinate x_i.

5.2.3 The Basic Method

As in the case of PCA, we first consider the projections of the n profiles in $\mathbb{R}^m$ onto an axis, $\mathbf{u}$. This is given by

$$\sum_j \frac{x_{ij}}{x_i} \frac{1}{x_j} u_j$$

for all i (note how the scalar product, used here, is closely related to the definition of distance — item 4 in Table 5.1). Let the above, for convenience, be denoted by w_i.

The weighted sum of projections uses weights x_i (i.e. the row masses), since the inertia of projections is to be maximized. Hence the quantity to be maximized is

$$\sum_i x_i {w_i}^2$$

subject to the vector $\mathbf{u}$ being of unit length (this, as in PCA, is required since otherwise vector $\mathbf{u}$ could be taken as unboundedly large):

$$\sum_j \frac{1}{x_j} {u_j}^2 = 1.$$

It may then be verified using Lagrangian multipliers that optimal $\mathbf{u}$ is an eigenvector of the matrix of dimensions $m \times m$ whose $(j,k)^{th}$ term is

$$\sum_i \frac{x_{ij} x_{ik}}{x_i x_k}$$

Space $\mathbb{R}^m$:

1. n row points, each of m coordinates.
2. The j^{th} coordinate is x_{ij}/x_i.
3. The mass of point i is x_i.
4. The χ^2 distance between i and k is:
   $d^2(i,k) = \sum_j \frac{1}{x_j}(\frac{x_{ij}}{x_i} - \frac{x_{kj}}{x_k})^2$.
   Hence this is a Euclidean distance with respect
   to the weighting $1/x_j$ (for all j).
5. The criterion to be optimised: the weighted sum
   of squares of projections, where the weighting
   is given by x_i (for all i).

Space $\mathbb{R}^n$:

1. m column points, each of n coordinates.
2. The i^{th} coordinate is x_{ij}/x_j.
3. The mass of point j is x_j.
4. The χ^2 distance between column points g and j is:
   $d^2(g,j) = \sum_i \frac{1}{x_i}(\frac{x_{ig}}{x_g} - \frac{x_{ij}}{x_j})^2$.
   Hence this is a Euclidean distance with respect
   to the weighting $1/x_i$ (for all i).
5. The criterion to be optimised: the weighted sum
   of squares of projections, where the weighting
   is given by x_j (for all j).

Table 5.1: Properties of spaces $\mathbb{R}^m$ and $\mathbb{R}^n$ in Correspondence Analysis.

where $1 \leq j, k \leq m$. (Note that this matrix is not symmetric, and that a related symmetric matrix must be constructed for eigenreduction: we will not detail this here.) The associated eigenvalue, λ, indicates the importance of the best fitting axis, or eigenvalue: it may be expressed as the *percentage of inertia explained* relative to subsequent, less good fitting, axes.

The results of a CA are centred (x_j and x_i are the j^{th} and i^{th} coordinates — the average profiles — of the origin of the output graphic representations). The first eigenvalue resulting from CA is a trivial one, of value 1; the associated eigenvector is a vector of 1s (Lebart *et al.*, 1984; Volle, 1981).

5.2.4 Axes and Factors

In the previous section it has been seen that projections of points onto axis $\mathbf{u}$ were with respect to the $1/x_i$ weighted Euclidean metric. This makes interpreting projections very difficult from a human/visual point of view, and so it is more natural to present results in such a way that projections can be simply appreciated. Therefore *factors* are defined, such that the projections of row vectors onto factor ϕ associated with axis $\mathbf{u}$ are given by

$$\sum_j \frac{x_{ij}}{x_i} \phi_j$$

for all i. Taking

$$\phi_j = \frac{1}{x_j} u_j$$

ensures this and projections onto ϕ are with respect to the ordinary (unweighted) Euclidean distance.

An analogous set of relationships hold in $\mathbb{R}^n$ where the best fitting axis, $\mathbf{v}$, is searched for. A simple mathematical relationship holds between $\mathbf{u}$ and $\mathbf{v}$, and between ϕ and ψ (the latter being the factor associated with eigenvector $\mathbf{v}$):

$$\sqrt{\lambda}\psi_i = \sum_j \frac{x_{ij}}{x_i} \phi_j$$

$$\sqrt{\lambda}\phi_j = \sum_i \frac{x_{ij}}{x_j} \psi_i.$$

These are termed *transition formulas*. Axes $\mathbf{u}$ and $\mathbf{v}$, and factors ϕ and ψ, are associated with eigenvalue λ and best fitting higher–dimensional subspaces are associated with decreasing values of λ, determined in the diagonalization.

The transition formulas allow *supplementary rows* or columns to be projected into either space. If ξ_j is the j^{th} element of a supplementary row, with mass ξ, then a factor loading is simply obtained subsequent to the CA:

$$\psi_i = \frac{1}{\sqrt{\lambda}} \sum_j \frac{\xi_j}{\xi} \phi_j.$$

A similar formula holds for supplementary columns. Such supplementary elements are therefore "passive" and are incorporated into the CA results subsequent to the CA being carried out.

5.2.5 Multiple Correspondence Analysis

When the input data is in *complete disjunctive form*, CA is termed Multiple CA (MCA). Complete disjunctive form is a form of coding where the response categories, or modalities, of an attribute have one and only one non–zero response (see Figure 5.1a). Ordinarily CA is used for the analysis of contingency tables: such a table may be derived from a table in complete disjunctive form by taking the matrix product between its transpose and itself. The symmetric table obtained in this way is referred to as a *Burt table*. CA of either table gives similar results, only the eigenvalues differing (see Lebart *et al.*, 1984; or Volle, 1981).

A few features of the analysis of tables in complete disjunctive form will be mentioned.

- The modalities (or response categories) of each attribute in MCA have their centre of gravity at the origin.
- The number of nonzero eigenvalues found is less than or equal to the total number of modalities less the total number of attributes.
- Due to this large dimensionality of the space being analyzed, it is not surprising that eigenvalues tend to be very small in MCA. It is not unusual to find that the first few factors can be usefully interpreted and yet account for only a few percent of the total inertia.

The principal steps in interpreting the output of MCA, as in CA, are similar to the interpreting of PCA output.

- The Burt table is scanned for significantly high frequencies of co–occurrence.
- The axes are interpreted in order of decreasing importance using the modalities which contribute most, in terms of inertia, to the axes (i.e. mass times projected distance squared). The projection coordinates serve to indicate how far the modality can be assessed relative to the axis.
- The planar graphic representations (projections of row and column points in the plane formed by factors 1 and 2, and by other pairs of factors) are examined.
- The interrelationships between modalities, relative to the axes, are examined, and substantive conclusions are drawn.

It may be noted that in the variant of Correspondence Analysis looked at in this section, the row–points are of constant weight. This allows, quite generally, user intervention in the weighting of rows relative to columns. In our experience, we have often obtained very similar results for a Principal Components Analysis with the usual standardization to zero mean and unit standard deviation, on the one hand; and on the other, a Correspondence Analysis with twice the number of columns as the matrix analyzed by PCA such that for each column j we also have a column j' with value $x_{ij'} = \max_k x_{ik} - x_{ij}$. This is referred to as *doubling* the data.

Some typical output configurations can arise, the most well known being the “horseshoe” shaped curve associated with pronounced linearity in the data.

Type			Age			Properties				
T1	T2	T3	A1	A2	A3	P1	P2	P3	P4	P5
1	0	0	0	1	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	1
1	0	0	0	0	1	0	0	1	0	0
1	0	0	0	1	0	0	0	1	0	0
1	0	0	1	0	0	1	0	0	0	0
0	0	1	0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0	0	0	1

(a) Table in complete disjunctive form.

	T1	T2	T3	A1	A2	A3	P1	P2	P3	P4	P5
T1	4	0	0	1	2	1	1	0	2	0	1
T2	0	1	0	0	0	1	0	0	0	0	1
T3	0	0	2	1	1	0	1	0	0	0	1
A1	1	0	1	2	0	0	1	0	0	0	1
A2	2	0	1	0	3	0	1	0	1	0	1
A3	1	1	0	0	0	2	0	0	1	0	1
P1	1	0	1	1	1	0	2	0	0	0	0
P2	0	0	0	0	0	0	0	0	0	0	0
P3	2	0	0	0	1	1	0	0	2	0	0
P4	0	0	0	0	0	0	0	0	0	0	0
P5	1	1	1	1	1	1	0	0	0	0	3

(b) Burt table.

Notes:

- Attributes: Type, Age, Properties.
- Modalities: T1, T2, ..., P5.
- Row sums of table in complete disjunctive form are constant, and equal the number of attributes.
- Each attribute × attribute submatrix of the Burt table (e.g. Ages × Ages) is necessarily diagonal, with column totals of the table in complete disjunctive form making up the diagonal values.

Figure 5.1: Table in complete disjunctive form and associated Burt table.

1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	0	0	1	1	1	1	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	1	1	0	0	0	0	0	0
4	0	0	0	0	0	0	1	1	1	1	0	0	0	0
5	0	0	0	0	0	0	0	0	1	1	1	1	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1

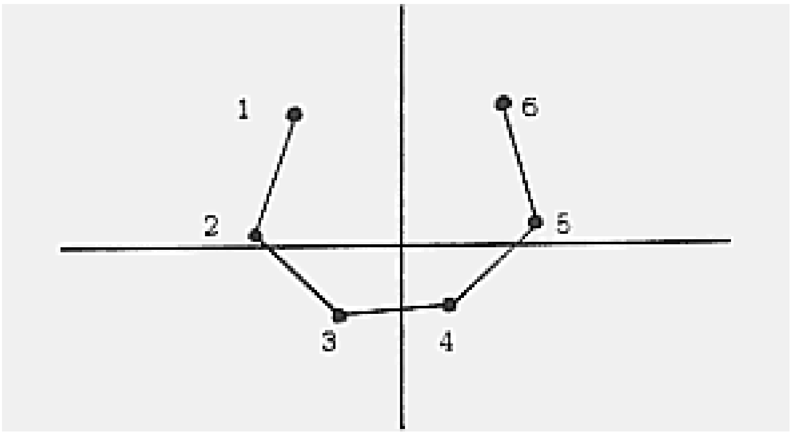


Figure 5.2: Horseshoe pattern in principal plane of Correspondence Analysis.

Figure 5.3 gives an example of the type of doubled data for which this pattern arises. It may be explained by the constraints imposed on the pairwise distances resulting from the input data. The one–dimensional ordering inherent in the input data is referred to as a *seriation.*

5.3 Principal Coordinates Analysis

5.3.1 Description

Principal Coordinates Analysis has also been referred to as *Classical Multidimensional Scaling* and *metric scaling*, and has been associated with the names of Torgerson and Gower. It takes distances as input and produces coordinate values as output. In this, it has been described as producing something from nothing.

It may be useful to review a possible situation where distance input is readily available whereas the raw data are not. Consider a catalogue or database information which contains attributes which are a mixture of quantitative and qualitative (categorical) types. Correspondence Analysis offers one approach to the analysis of such data, by recoding all data in a qualitative form, and using the complete disjunctive form of coding seen above. An alternative is to use a distance for mixed data derived from the Gower coefficient described in Chapter 1, and then to use Principal Coordinates Analysis on such data.

We now describe the principles on which Principal Coordinates Analysis is based, and how it is implemented. Consider the initial data matrix, X, of dimensions $n \times m$, and the “sums of squares and cross products” matrix of the rows:

$$A = XX'$$
$$a_{ik} = \sum_j x_{ij} x_{kj}.$$

If d_{ik} is the Euclidean distance between objects i and k (using row–vectors i and k of matrix X) we have that:

$$\begin{aligned} d_{ik}^2 &= \sum_j (x_{ij} - x_{kj})^2 \\ &= \sum_j x_{ij}^2 + \sum_j x_{kj}^2 - 2 \sum_j x_{ij} x_{kj} \\ &= a_{ii} + a_{kk} - 2a_{ik}. \end{aligned} \tag{5.1}$$

In Principal Coordinates Analysis, we are given the distances and we want to obtain X. We will assume that the columns of this matrix are centred, i.e.

$$\sum_i x_{ij} = 0.$$

It will now be shown that matrix A can be constructed from the distances using the following formula:

$$a_{ik} = -\frac{1}{2}(d_{ik}^2 - d_i^2 - d_k^2 - d^2) \tag{5.2}$$

where

$$d_i^2 = \frac{1}{n} \sum_k d_{ik}^2$$
$$d_k^2 = \frac{1}{n} \sum_i d_{ik}^2$$
$$d^2 = \frac{1}{n^2} \sum_i \sum_k d_{ik}^2.$$

This result may be proved by substituting for the distance terms (using equation 5.1), and knowing that by virtue of the centring of the row vectors of matrix X, we have

$$\sum_i a_{ik} = 0$$

(since $a_{ik} = \sum_j x_{ij} x_{kj}$; and consequently in the term $\sum_i \sum_j x_{ij} x_{kj}$ we can separate out $\sum_i x_{ij}$ which equals zero). Similarly (by virtue of the symmetry of A) we use the fact that

$$\sum_k a_{ik} = 0.$$

Having thus been given distances, we have constructed matrix $A = XX'$. We now wish to reconstruct matrix X; or, since this matrix has in fact never existed, we require some matrix X which satisfies $XX' = A$.

If matrix A is positive, symmetric and semidefinite, it will have rank $p \leq n$. We may derive p non–zero eigenvalues, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p > 0$, with corresponding eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_p$. Consider the scaled eigenvectors, defined as $\mathbf{f}_i = \sqrt{\lambda_i}\mathbf{u}_i$. Then the matrix $X = (\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_p)$ is a possible coordinate matrix. This is proved as follows. We have, in performing the eigen–decomposition of A:

$$A\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

and by requirement

$$XX'\mathbf{u}_i = \lambda_i \mathbf{u}_i.$$

In the left hand side, $X \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{f}_p \end{pmatrix} \mathbf{u}_i = X \begin{pmatrix} \sqrt{\lambda_1}\mathbf{u}_1 \\ \sqrt{\lambda_2}\mathbf{u}_2 \\ \cdot \\ \cdot \\ \cdot \\ \sqrt{\lambda_p}\mathbf{u}_p \end{pmatrix} \mathbf{u}_i = X \begin{pmatrix} 0 \\ 0 \\ \cdot \\ \sqrt{\lambda_i} \\ \cdot \\ 0 \end{pmatrix}$ since eigenvectors are mutually orthogonal. Continuing:

$$(\sqrt{\lambda_1}\mathbf{u}_1, \sqrt{\lambda_2}\mathbf{u}_2, \ldots, \sqrt{\lambda_p}\mathbf{u}_p) \begin{pmatrix} 0 \\ 0 \\ \cdot \\ \sqrt{\lambda_i} \\ \cdot \\ 0 \end{pmatrix} = \lambda_i \mathbf{u}_i.$$

Thus we have succeeded in constructing a matrix X, having been initially given a set of distances. The net result is very similar to PCA, — we have a series of orthogonal axes which indicate inherent dimensionality and which may be used for plotting the objects studied. We have not used any set of variables or attributes in Principal Coordinates Analysis, but on carrying out this technique we have a set of projections on principal axes which may be used as attributes.

Principal Coordinates Analysis may be programmed using a PCA program with relatively few changes. Rather than reading in the data and constructing a sums of squares and cross products (SSCP) matrix, we read in a matrix of distances and use Equation 5.2 above to form the SSCP matrix.

In practice, we might be given dissimilarities rather than distances. Then, matrix A will be symmetric and have zero values on the diagonal but will not be positive semidefinite. In this case negative eigenvalues are obtained. These are inconvenient but may often be ignored if the approximate Euclidean representation (given by the eigenvectors corresponding to positive eigenvalues) is satisfactory.

5.3.2 Multidimensional Scaling

Principal Coordinates Analysis has been referred to as *metric multidimensional scaling*. From the early 1960s onwards, *non–metric multidimensional scaling* was developed and became widely used especially in the areas of psychology and marketing. Given dissimilarities, δ_{ij}, it estimates (using iterative optimization) Euclidean distances, d_{ij}, such that rank orders are preserved as far as possible:

$$\delta_{ij} \leq \delta_{kl} \Rightarrow d_{ij} \leq d_{kl}.$$

A figure of merit for the Euclidean configuration in a space of some given dimension is the "stress", a definition of which is

$$\sum_{i>j}(d_{ij} - \delta_{ij})^2 / \sum_{i>j} d_{ij}^2.$$

A space of low dimension is sought, coupled with a low value of the stress criterion. Non–metric multidimensional scaling is less restrictive in its assumptions about input data compared to Principal Coordinates Analysis, but it may require greater computation time. Further reading is to be found in Gordon (1981) and Kruskal and Wish (1978).

5.4 Canonical Correlation Analysis

In Canonical Correlation (or Variate) Analysis, we are given n objects crossed by two sets of p and q variables, respectively. The $n \times p$ matrix, X, and the $n \times q$ matrix, Y, together give the raw data matrix Z of dimensions $n \times (p + q)$. As examples of problems suitable for study using Canonical Correlation Analysis, consider n objects with attributes defined by two different types of instrument, by two different sets of observers, using two different photometric systems, and so on.

Two sets of linear combinations of row i's coordinates are defined, relative to X and relative to Y:

$$u_i = \sum_{j=1}^{p} a_j x_{ij} \qquad \mathbf{u} = X\mathbf{a}$$

$$v_i = \sum_{j=1}^{q} b_j y_{ij} \qquad \mathbf{v} = Y\mathbf{b}$$

where $\mathbf{a}$ and $\mathbf{b}$ are vectors of constants of dimensions, respectively, $p \times 1$ and $q \times 1$.

Canonical Correlation Analysis seeks two sets of linear combinations, $\mathbf{u}$ and $\mathbf{v}$, such that these are as highly correlated as possible. As in PCA (Chapter 2), the restriction is imposed that $\mathbf{u}$ and $\mathbf{v}$ have unit variance; and we proceed to an optimization, using Lagrangian multipliers, which results in an eigenvalue equation. Successive pairs of correlated linear combinations are determined.

The eigenvalues are referred to as *canonical correlation coefficients*. If the first two coefficients are close to 1, there is a strong resemblance between the two sets of variables, as measured on the first two canonical variates. For a planar representation of the variables, we may select the canonical variates of either group of variables, and project the other group onto this plane. As in PCA we will seek to interpret the axes (points which are projected towards the extremities may offer greatest help for doing this); and to look for clusters of variables, if possible leading to indications of close association between the variables of the two sets used.

Canonical Correlation Analysis is not as widely used as might be expected from the generality of the problems which it can address. This is because results are difficult to interpret, especially when canonical correlations, associating the two sets of variables, are not high.

5.5 Regression Analysis

In linear regression we are given a set of m values, $\{x_j, y_j : j = 1, \ldots m\}$ and we require a fit of the form $y = a + bx$, the *regression line of y on x*. We assume that x is fixed or controlled, i.e. that its exact value is known. The variable y is *stochastic*. The problem is to predict y, given the values of x. The philosophy is somewhat different here compared to principal components analysis and may be easily visualized as shown in Figure 5.3. Think of x as a time-related quantity: to predict y, we are not allowed to use future values of x.

It is assumed that the j^{th} y value, y_j, is distributed about some mean $\bar{y}_j$ with standard deviation σ_j. Thus, for a given x_j, we have a certain scatter in corresponding y_j values, and the eventual linear fit will be based on $\bar{y}_j$. The distributional form of the y_j values is assumed to be Gaussian. Hence the probability of a particular value of y_j is given by

$$\frac{1}{\sqrt{2\pi}\,\sigma_j}\exp(-\frac{1}{2}(\frac{y_j-\bar{y}_j}{\sigma_j})^2).$$

The simultaneous probability of a given set of m values is given by the product of this expression, over all j:

$$\prod_{j=1}^{m}(\frac{1}{\sqrt{2\pi}\,\sigma_j}\exp(-\frac{1}{2}\sum_{j=1}^{m}(\frac{y_j-\bar{y}_j}{\sigma_j})^2).$$

In order to determine $\bar{y}_j$ we employ the method of maximum likelihood: maximizing the above term implies minimizing the exponent. Hence we minimize

$$\sum_{j=1}^{m}(\frac{y_j-\bar{y}_j}{\sigma_j})^2$$

which leads to the least squares estimate of the $\bar{y}_j$ values. (Linear regression is often introduced more directly as this least squares optimizing, with weights defined by $w_j = 1/\sigma_j^2$.) Using $\bar{y}_j = a + bx_j$, the expression to be minimized becomes

$$\sum_{j=1}^{m}(\frac{y_j-a-bx_j}{\sigma_j})^2.$$

To solve for a and b, the partial derivatives of this term with respect to a and to b are determined, and the resulting equations solved. The expressions found in this way for a (the *intercept*) and b (the *slope* or *regression coefficient*) are given in Bevington (1969), together with approximate error bounds.

In the foregoing, a *regression of y on x* has been looked at. By supposition, there is variation in y but not in x. Some variation in x is acceptable in practice, but if both variables are considered as stochastic, this linear regression is not suitable. In the case of stochastic x and error–free y, it is of course straightforward to regress x on y.

When both x and y are stochastic, determining the best fitting straight line is studied by York (1966; see also Lybanon, 1984). In this case, analytic expressions for a and b are not feasible. Instead, given some initial estimate of the slope, b_0 (obtained, for instance, by a regression analysis ignoring the error weights on x and y), iterative optimization is used to improve the value of b.

Linear regression generalizes to *multiple regression* when a fit of the form

$$y = a_0 + a_1x_1 + a_2x_2 + \ldots + a_nx_n$$

is sought, for n independent variables. The coefficients, a, are called *partial regression coefficients*.

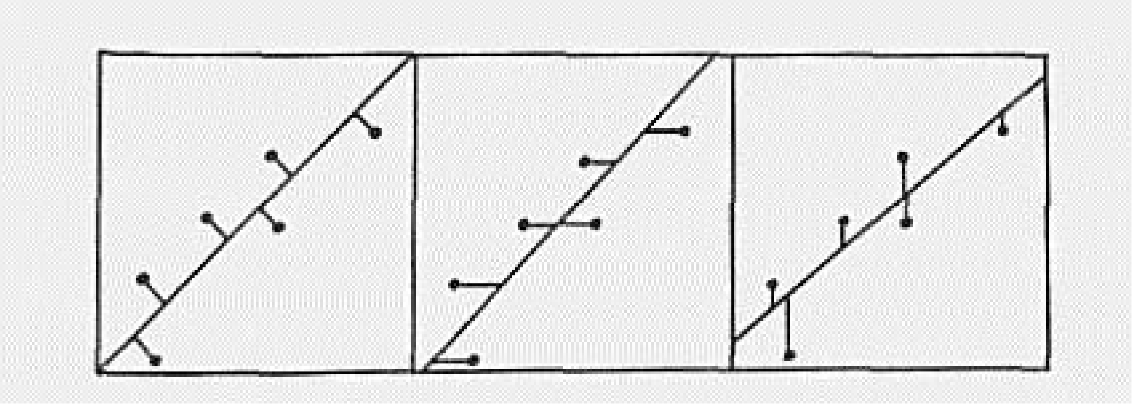


Figure 5.3: Projections: left PCA; centre regression of y on x (usual case of regression); right regression of x on y.

5.6 Examples and Bibliography

5.6.1 Regression in Astronomy

1. E. Antonello and M. Fracassini, "Pulsars and interstellar medium: multiple regression analysis of related parameters", *Astrophysics and Space Science*, **108**, 187–193, 1985.

2. R.L. Branham Jr., "Alternatives to least–squares", *The Astronomical Journal*, **87**, 928–937, 1982.

3. R. Buser, "A systematic investigation of multicolor photometric systems. II. The transformations between the UBV and RGU systems.", *Astronomy and Astrophysics*, **62**, 425–430, 1978.

4. C.R. Cowley and G.C.L. Aikman, "Stellar abundances from line statistics", *The Astrophysical Journal*, **242**, 684–698, 1980.

5. M. Crézé, "Influence of the accuracy of stellar distances on the estimations of kinematical parameters from radial velocities", *Astronomy and Astrophysics*, **9**, 405–409, 1970.

6. M. Crézé, "Estimation of the parameters of galactic rotation and solar motion with respect to Population I Cepheids", *Astronomy and Astrophysics*, **9**, 410–419, 1970.

7. T.J. Deeming, "The analysis of linear correlation in astronomy", *Vistas in Astronomy*, **10**, 125, 1968.

8. H. Eichhorn, "Least–squares adjustment with probabilistic constraints", *Monthly Notices of the Royal Astronomical Society*, **182**, 355–360, 1978.

9. H. Eichhorn and M. Standish, Jr., "Remarks on nonstandard least–squares problems", *The Astronomical Journal*, **86**, 156–159, 1981.

10. M. Fracassini, L.E. Pasinetti and E. Antonello, "Pulsars and interstellar medium", *Proceedings of a Course and Workshop on Plasma Astrophysics*, European Space Agency Special Publication 207, 319–321, 1984.

11. J.R. Gott III and E.L. Turner, "An extension of the galaxy covariance function to small scales", *The Astrophysical Journal*, **232**, L79–L81, 1979.

12. A. Heck, "Predictions: also an astronomical tool", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 1983, pp. 135–143.

    (A survey article, with many references. Other articles in this conference proceedings also use regression and fitting techniques.)

13. A. Heck and G. Mersch, "Prediction of spectral classification from photometric observations — application to the $uvby\beta$ photometry and the MK spectral classification. I. Prediction assuming a luminosity class", *Astronomy and Astrophysics*, **83**, 287–296, 1980.

    (Stepwise multiple regression and isotonic regression are used.)

14. W.H. Jefferys, "On the method of least squares", *The Astronomical Journal*, **85**, 177–181, 1980.

15. W.H. Jefferys, "On the method of least squares. II.", *The Astronomical Journal*, **86**, 149–155, 1981.

16. J.R. Kuhn, "Recovering spectral information from unevenly sampled data: two machine–efficient solutions", *The Astronomical Journal*, **87**, 196–202, 1982.

17. T.E.Lutz, "Estimation — comments on least squares and other topics", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 179–185, 1983.

18. M.O. Mennessier, "Corrections de précession, apex et rotation galactique estimées à partir de mouvements propres fondamentaux par une méthode de maximum vraisemblance", *Astronomy and Astrophysics*, **17**, 220–225, 1972.

19. M.O. Mennessier, "On statistical estimates from proper motions. III.", *Astronomy and Astrophysics*, **11**, 111–122, 1972.

20. G. Mersch and A. Heck, "Prediction of spectral classification from photometric observations — application to the $uvby\beta$ photometry and the MK spectral classification. II. General case", *Astronomy and Astrophysics*, **85**, 93–100, 1980.

21. J.F. Nicoll and I.E. Segal, "Correction of a criticism of the phenomenological quadratic redshift–distance law", *The Astrophysical Journal*, **258**, 457–466, 1982.

22. J.F. Nicoll and I.E. Segal, "Null influence of possible local extragalactic perturbations on tests of redshift–distance laws", *Astronomy and Astrophysics*, **115**, 398–403, 1982.

23. D.M. Peterson, “Methods in data reduction. I. Another look at least squares”, *Publications of the Astronomical Society of the Pacific*, **91**, 546–552, 1979.

24. I.E. Segal, “Distance and model dependence of observational galaxy cluster concepts”, *Astronomy and Astrophysics*, **123**, 151–158, 1983.

25. I.E. Segal and J.F. Nicoll, “Uniformity of quasars in the chronometric cosmology”, *Astronomy and Astrophysics*, **144**, L23–L26, 1985.

5.6.2 Regression in General

1. P.R. Bevington, *Data Reduction and Error Analysis for the Physical Sciences*, McGraw–Hill, New York, 1969.

   (A recommendable text for regression and fitting, with many examples.)

2. N.R. Draper and H. Smith, *Applied Regression Analysis*, Wiley, New York, 1981 (2nd ed.).

3. B.S. Everitt and G. Dunn, *Advanced Methods of Data Exploration and Modelling*, Heinemann Educational Books, London, 1983.

   (A discursive overview of topics such as linear models and analysis of variance; PCA and clustering are also covered.)

4. M. Lybanon, “A better least–squares method when both variables have uncertainties”, *American Journal of Physics* **52**, 22–26, 1984.

5. D.C. Montgomery and E.A. Peek, *Introduction to Linear Regression Analysis*, Wiley, New York, 1982.

6. G.A.F. Seber, *Linear Regression Analysis*, Wiley, New York, 1977.

7. G.B. Wetherill, *Elementary Statistical Methods*, Chapman and Hall, London, 1967.

   (An elementary introduction, with many examples.)

8. D. York, “Least squares fitting of a straight line”, *Canadian Journal of Physics* **44**, 1079–1086, 1966.

   (Deals with the case of stochastic independent and dependent variables.)

5.6.3 Other Techniques

Regression excepted, little to date has been accomplished in astronomy using the other, varied, techniques discussed in this Chapter. In this, there clearly is scope for change! Some general references follow.

1. J.P. Benzécri, *L’Analyse des Données. II. L’Analyse des Correspondances*, Dunod, Paris, 1979 (3rd ed.).

   (The classical tome on Correspondence Analysis.)

2. W.W. Cooley and P.R. Lohnes, *Multivariate Data Analysis*, Wiley, New York, 1971.

   (See for Canonical Correlation Analysis, together with other techniques.)

3. A.D. Gordon, *Classification*, Chapman and Hall, London, 1981.

   (For some reading on multidimensional scaling, together with other topics.)

4. M. Greenacre, *Theory and Applications of Correspondence Analysis*, Academic Press, New York, 1984.

   (As the title suggests, a detailed study of Correspondence Analysis.)

5. J.B. Kruskal and M. Wish, *Multidimensional Scaling*, Sage, Beverly Hills, 1978.

6. L. Lebart, A. Morineau and K.M. Warwick, *Multivariate Statistical Analysis*, Wiley, New York, 1984.

   (This is especially recommendable for Multiple Correspondence Analysis.)

7. E. Malinvaud and J.C. Deville, "Data analysis in official socio–economic statistics", *Journal of the Royal Statistical Society Series A* **146**, 335–361, 1983.

   (The use of multivariate statistics on large data collections is discussed.)

8. S.S. Schifman, M.L. Reynolds and F.L. Young, *Introduction to Multidimensional Scaling*, Academic Press, New York, 1981.

9. W.S. Torgerson, *Theory and Methods of Scaling*, Wiley, New York, 1958.

   (Although old, this book is very readable on the subject of non–metric multidimensional scaling.)

10. M. Volle, *Analyse des Données*, Economica, Paris, 1981.

    (Correspondence Analysis is dealt with in a practical fashion.)

Chapter 6

Case Study 1: IUE Low Dispersion Spectra

6.1 Presentation

The aim of this case study is to illustrate the use of various methods introduced in previous chapters on a specific, real–life example. The emphasis here will be more on the statistical aspects of the problem than on the astrophysical details, which the interested reader can find in the specialized papers referenced and more particularly in Heck *et al.* (1984b, 1986a).

6.2 The IUE Satellite and its Data

The International Ultraviolet Explorer (IUE) satellite was launched on 26 January 1978 to collect spectra in the ultraviolet wavelength range (UV) for all types of celestial objects. To date, IUE can be qualified as the most successful and the most productive astronomical satellite. It is also the first "space telescope" to be exploited in the same way as a "mission" ground observatory, with visiting astronomers participating in real time in the decision loop for programming the observational sequences and collecting data during the observing shifts allocated to their approved programmes. At the time of writing, the satellite is still active 24 hours per day in geosynchronous orbit at about 40,000 km from the Earth. Its construction, launch and utilisation resulted from a joint venture by three space agencies: NASA (USA), SERC (UK) and ESA (Europe). It is exploited from two ground stations: one for 16 hours per day at NASA Goddard Space Flight Center (Greenbelt, Maryland, USA), and the other for the remaining 8 hours per day at the ESA VILSPA Satellite Tracking Station (Villafranca del Castillo, near Madrid, Spain).

IUE scientific instrumentation essentially consists of a Ritchey–Chrétien 45 cm telescope with two spectrographs: one working in the $1150 - 2000$ Å range, and the other in the $1900 - 3200$ Å range. Each spectrograph can record spectra in a low–resolution mode (≈ 7 Å) and in a high–resolution mode (≈ 0.1 Å). For more details on IUE, see Boggess *et al.* (1978a, 1978b), and see also the IUE Memorial Book (Kondo *et al.*, 1986) containing among others a chapter on UV

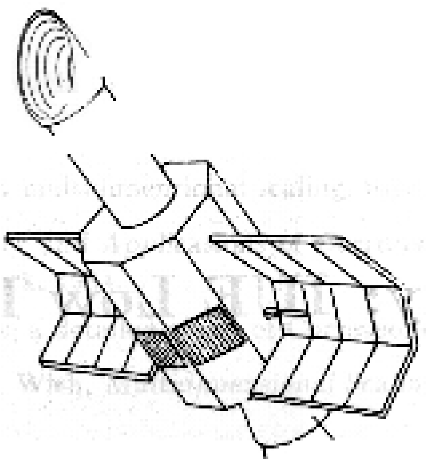


Figure 6.1: The International Ultraviolet Explorer (IUE).

spectral classification (Heck, 1986).

Only low–dispersion spectra will be considered in the following, and by spectrum we shall understand 410 flux values at 5 Å steps (just below the actual resolution) covering the whole IUE wavelength range ($1150-3200$ Å), obtained by merging the outputs of both spectrographs.

6.3 The Astrophysical Context

From earlier work on data collected by the S2/68 experiment on board the TD1 satellite (called "TD1" in the following), it had been shown that stars which were classified as spectrally normal in the visible range ($\approx 3500-4800$ Å) did not necessarily behave normally in the ultraviolet range and vice versa (see Cucchiaro *et al.*, 1978 and the references quoted therein). Consequently, MK spectral classifications which are defined in the visible range cannot simply be extrapolated to the UV.

IUE covers a larger wavelength range than TD1 ($1150-3200$ Å as against $1250-2550$ Å) and it has, even at low dispersion, a resolution about five times better than TD1 (7 Å as against 36 Å). Moreover IUE has observed a much broader range of stellar types than TD1 and has also reached significantly fainter magnitudes. Therefore a UV stellar classification programme was initiated in order to define, from IUE low–dispersion spectra, smooth UV spectral sequences. The latter were to describe stellar behaviour in the UV while staying as far as possible in accordance with the MK scheme in the visible.

The first volume of an *IUE Low–Dispersion Spectra Reference Atlas* (called the *Atlas* in the following) has already been produced (Heck *et al.*, 1984a), together with reference sequences and standard stars, and a second volume devoted to peculiar groups is in preparation (see Heck *et al.*, 1986b). The considerable classification work undertaken follows a classical morphological approach (Jaschek and Jaschek, 1984) and it essentially confirms that there is no one–to–one correspondence between the UV and visible ranges.

Let us recall here that the IUE spectral classification consists of a symbol expressing membership in a luminosity class (e.g. *s+* defining supergiant, bright), followed by spectral–type symbols linked to the effective temperature of the star (e.g. *B5*).

Stellar spectral classifications are more than just taxonomical exercises aimed at labelling and categorizing stars by comparison with standards. They are used for describing fundamental physical parameters in the outer atmospheres of the stars, for discriminating peculiar objects, and for other subsidiary applications like distance determinations, interstellar extinction and population synthesis studies.

It is important to bear in mind that the classification systems are built independently of stellar physics in the sense that they are defined completely from spectral features, selected in standard stars, in a given wavelength range (see, e.g., Jaschek, 1979 and Morgan, 1984). If the schemes are based on a sufficiently large number of objects, then these classification schemes cannot be other than intimately linked with stellar physics. Such classification schemes will not necessarily relate to the same stellar layers, if they refer to different wavelength ranges. Consequently, the discrepancies found between the MK system and the UV framework are not surprising.

This also implies that the only way to confirm the correctness of the UV classification framework introduced in the *Atlas* is to remain in the same wavelength range. A statistical approach would moreover be independent of any *a priori* bias arising from existing schemes, either in the visible or in the ultraviolet ranges. An additional advantage of statistical methodology lies in the fact that it is able to work at will in a multidimensional parameter space, while classical morphological classifications rarely go beyond two dimensions.

The aim of this study is thus to apply multidimensional statistical algorithms to variables expressing as objectively as possible the information contained in the continuum and the spectral features of low–dispersion IUE stellar spectra. Several objectives may be pursued, but the most important will be the arrangement of the corresponding stars into groups, whose homogeneity, in terms of the classification symbolism introduced in the *Atlas*, should reflect the accuracy of this IUE UV classification scheme.

6.4 Selection of the Sample

Although much of the following is applicable to all types of spectra, consideration is restricted here to stars that are normal in the UV. Thus, for this application, we retained 264 IUE low–dispersion spectra which were technically good in the sense that images of poor quality (i.e. strongly underexposed or affected by saturation, microphonic noise or other defects) were discarded.

TD1 data had the advantage of resulting from a survey and of representing a magnitude–limited unbiased sample of bright stars, whereas IUE is pointed only at preselected targets from accepted proposals. It thereby provides a biased sample, mainly in favour of early (or hotter) spectral types (typical UV emitters). This is illustrated in Table 6.1, which gives the distribution of the sample stars for their UV spectral types (effective temperature decreasing from hotter *O* to cooler *K* stars) and luminosity classes (luminosity decreasing from bright supergiants, *s+*, to dwarfs, *d*).

Since we intend to compare flux values of stars with (sometimes very) dif-

	O	B	A	F	G	K
s+	3	16				
s	4	4	9	5	2	
s−	2	9				
g+	4	6	2			
g	9	23	4	5		
d+	5	13	3			
d	20	60	31	17	5	2

Table 6.1: UV spectral distribution of the stars sampled.

ferent apparent magnitudes, it is necessary to normalize the flux scales. This was done by dividing each spectrum by the average flux value over the whole interval, which was equivalent to normalizing the integrated flux over the whole IUE range.

6.5 Definition of the Variables

Some of the algorithms will be applied below to the original 264 × 410 original flux values, but the 410 flux values will also be condensed into a smaller number of variables expressing as exhaustively as possible the information contained in the spectra. Clearly, the smaller the number of variables, the less computation time is required in the case of some of the multivariate data analysis algorithms.

Essentially two types of information can be extracted from a spectrum: on the one hand, the general shape of the continuum which will be described by an asymmetry coefficient; and on the other hand, the various data relative to the individual spectral lines. In total, 60 line intensity values were used.

6.5.1 The Continuum Asymmetry Coefficient

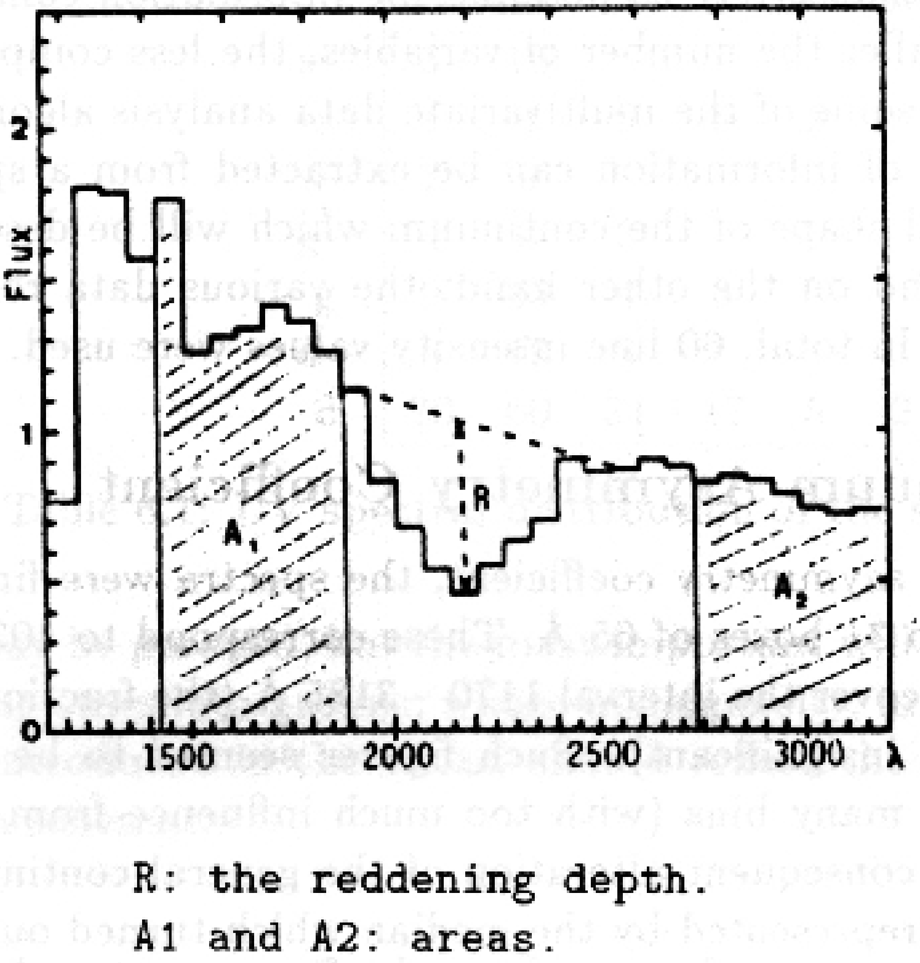
To define the continuum asymmetry coefficient, the spectra were first smoothed by binning the fluxes into 31 boxes of 65 Å. These correspond to 403 intervals of the initial 5 Å steps, and cover the interval 1170 − 3185 Å (the fractions of spectra dropped at each end are insignificant). Such figures seemed to be an excellent compromise between too many bins (with too much influence from the spectral lines) and too few (with consequent alteration of the general continuum shape). In each bin, the flux was represented by the median which turned out to be more suitable than the mean, the latter being too much affected by possible lines in the bin.

The following asymmetry coefficient was then calculated:

$$S = (A_1 - A_2)/(A_1 + A_2)$$

where A_1 and A_2 are the areas illustrated in Figure 6.2 and correspond respectively to the ranges 1430 − 1885 Å and 2730 − 3185 Å. The first interval was selected as the largest feasible interval between possible parasitic effects arising in the spectra from:

- on the shortwavelength side, either a $L\alpha$ geocoronal emission or strong stellar absorption;



R: the reddening depth.
A1 and A2: areas.

Figure 6.2: Illustration of terms in the asymmetry coefficient S (see text).

- on the longwavelength side, a 2200 Å interstellar absorption bump.

There were seven 65 Å bins between these two features. Area A_2 was taken with the same length at the longwavelength end of the spectrum, and as far as possible from the 2200 Å bump. The positions and sizes of A_1 and A_2 could be defined differently, but the choices made here were fully satisfactory for our purposes.

It is easy to see that S varies from 1 (the extreme case of hot stars — all the flux at the shortwavelength end) to -1 (the extreme case of cool stars — all the flux at the longwavelength end). Both flat and balanced spectra give $S = 0$.

The asymmetry coefficient appeared immediately to be a powerful discriminating parameter for normal stars. However, the reddening turned out to be more troublesome than expected and it had to be taken into account, even outside the zone that we avoided. Actually, a simple glance at Seaton's (1979) interstellar–extinction curve (his Figure 1) makes it clear that the absorption is as strong around 1350 Å as around 2200 Å. This means that a hot–star spectrum (clearly asymmetric with S close to 1 when unreddened) becomes fairly symmetric (with S close to 0) when reddened (see Figure 6.3).

6.5.2 The Reddening Effect

In order to correct the asymmetry coefficient for the reddening effect, we introduced a new parameter R (see Figure 6.2), which we called the reddening depth and defined as:

$$R = (F_{1890} + F_{2530})/2 - F_{2210}$$

(where F_λ represents the flux at wavelength λ) giving a rough measurement of the amplitude of the 2200 Å bump.

We then dereddened a number of spectra from our sample (chosen at various degrees of reddening) by using Seaton's (1979) absorption law and compared the pairs (reddened–dereddened) in the normalized representation. As demonstrated by Figure 6.3, the reddening increases area A_2 compared to area A_1, which stays roughly at the same level in most cases.

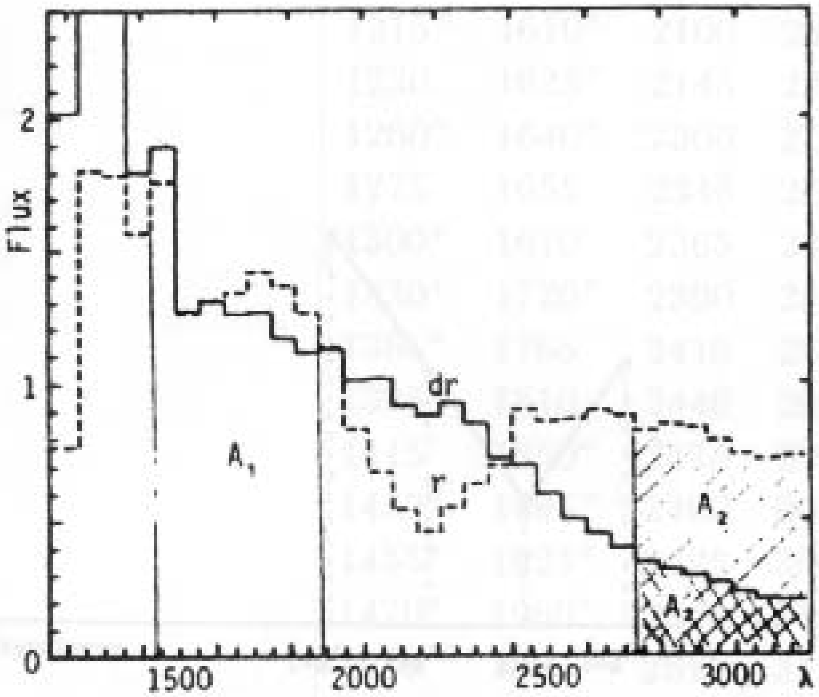
A rather simple functional relationship could be empirically established between the reddening depth R and the difference in area A_2 due to the reddening:

$$\ln(A_2 - A'_2) = 3.0 \ln R - 0.5$$

where A_2 corresponds to the reddened spectrum and A'_2 to the unreddened one. For $R = 0$, we have of course $A_2 = A'_2$. Quite naturally, the coefficients of this relationship could be refined with a bigger sample.

Thus, when a positive reddening is detected, area A_2 has to be decreased by $(A_2 - A'_2)$ before calculating the asymmetry coefficient S.

It should be emphasized here that, although only one reddening law has been used for illustrating and for determining a rough empirical relationship, this does not imply that we consider it of universal application in the UV. It is however sufficient within the overall accuracy of this case study, whose results are mainly qualitative.



r denotes the reddened curve,
dr denotes the de-reddened curve.
A2 is actually smaller in the dereddened spectrum.

r denotes the reddened curve,
dr denotes the de-reddened curve.
A2 is actually smaller in the dereddened spectrum.

Figure 6.3: Illustration of the reddening effect on a normalized spectrum.

6.6 Spectral Features

6.6.1 Generalities

We decided to concentrate on spectral features which are classically used for spectral classification, i.e. emission and/or absorption lines which can play a rôle with different degrees of sophistication: firstly, by their presence or absence only; secondly, by their comparative intensity; and finally, by their intensity weighted in an appropriate way.

In any case, an objective approach should begin by a systematic search for these lines. As our sample consists only of normal stars, we consider in the following only absorption lines but, *mutatis mutandis*, similar considerations could also apply to emission features.

6.6.2 Objective Detection of the Spectral Lines

We looked for the most frequent minima in the spectra of our sample. Various morphological tests were applied to the values of each original flux vector by moving sets of 3 to 7 points. Then a histogram was constructed to indicate the frequency of the minima detected over the whole spectral range and over the whole sample. The test which appeared to be the most adequate and the most efficient (in terms of the best compromise between line detection and insensitivity to noise) is reproduced in Figure 6.4. In other terms, we considered as a potential line wavelength, that corresponding to the flux component n such that:

$$F_{n-2} > F_{n-1} > F_n < F_{n+1}$$

or such that

$$F_{n-1} > F_n < F_{n+1} < F_{n+2}$$

where F_i represents the flux at the i^{th} component. Such an asymmetry in the tests permits the detection of lines in the wings of larger nearby ones.

We decided to select the most frequent lines because we were dealing with a sample of normal stars. With peculiar stars, the criterion would be different since one given line appearing only in a few stars could be a discriminator for that particular group. The approximate wavelength (5 Å precision) of the 60 most frequent lines are given in Table 6.2. They include all the lines used in the morphological classification given in the *Atlas*.

Additional lines could have been used, but 60 turned out to be a good figure for this study. The line selection was in principle done above the noise level corresponding to a purely random process. However, the minima corresponding to the IUE réseau marks were ignored, as were those resulting from, on the one hand, the junction of the "long-" and "shortwavelength" parts of the spectra and, on the other hand, from the camera sensitivity drops towards the extremities of their respective spectral ranges.

6.6.3 Line Intensities

Line intensities were calculated (from normalized flux values) as the differences between the minima and associated continua smoothed at the median values over 13 components centred on each line.

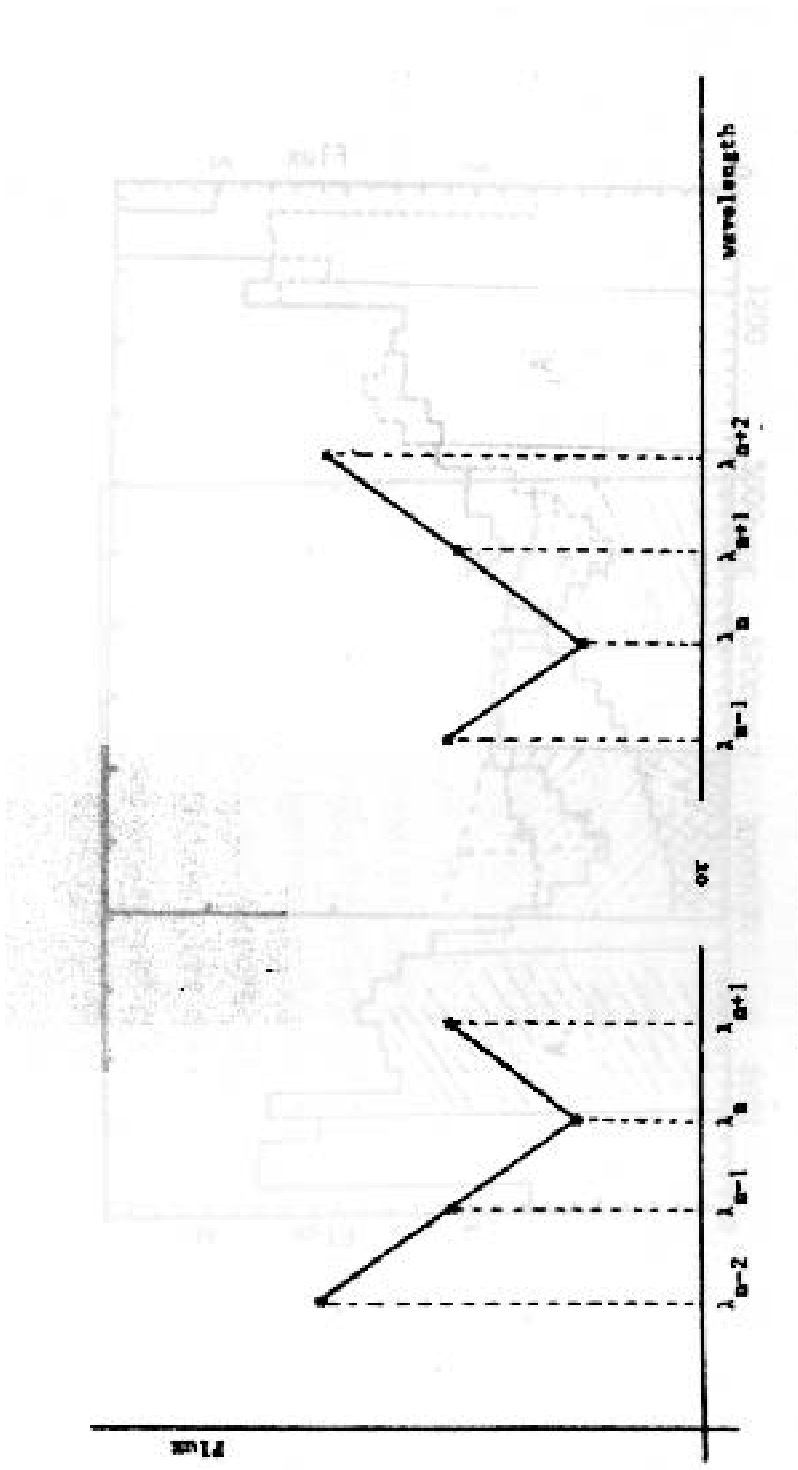


Figure 6.4: Test used for objectively detecting potential lines in the spectra.

1175*	1565	2065	2665*
1215*	1610*	2100	2695
1230	1625*	2145	2720
1260*	1640*	2300	2750*
1275	1655	2345	2800*
1300*	1670	2365	2835
1330*	1720*	2390	2855*
1365*	1765	2410	2875
1395*	1810	2440	2930
1415	1850*	2475	2990
1430*	1895*	2490	3025
1455*	1925*	2525	3065
1470*	1960*	2540	3105
1535	1995	2610	3120
1550*	2040	2635	3160

Asterisks indicate lines used for classification
in the IUE Normal Stars Atlas (Heck et al., 1984a).

Table 6.2: Bin wavelengths corresponding to the 60 most frequent lines in the spectral sample at hand.

Allowing for the usual difficulties of drawing a spectral continuum, we believe this rough systematic procedure does not introduce additional imprecision in the whole approach and does not modify substantially the essentially qualitative results of the study.

6.6.4 Weighting Line Intensities

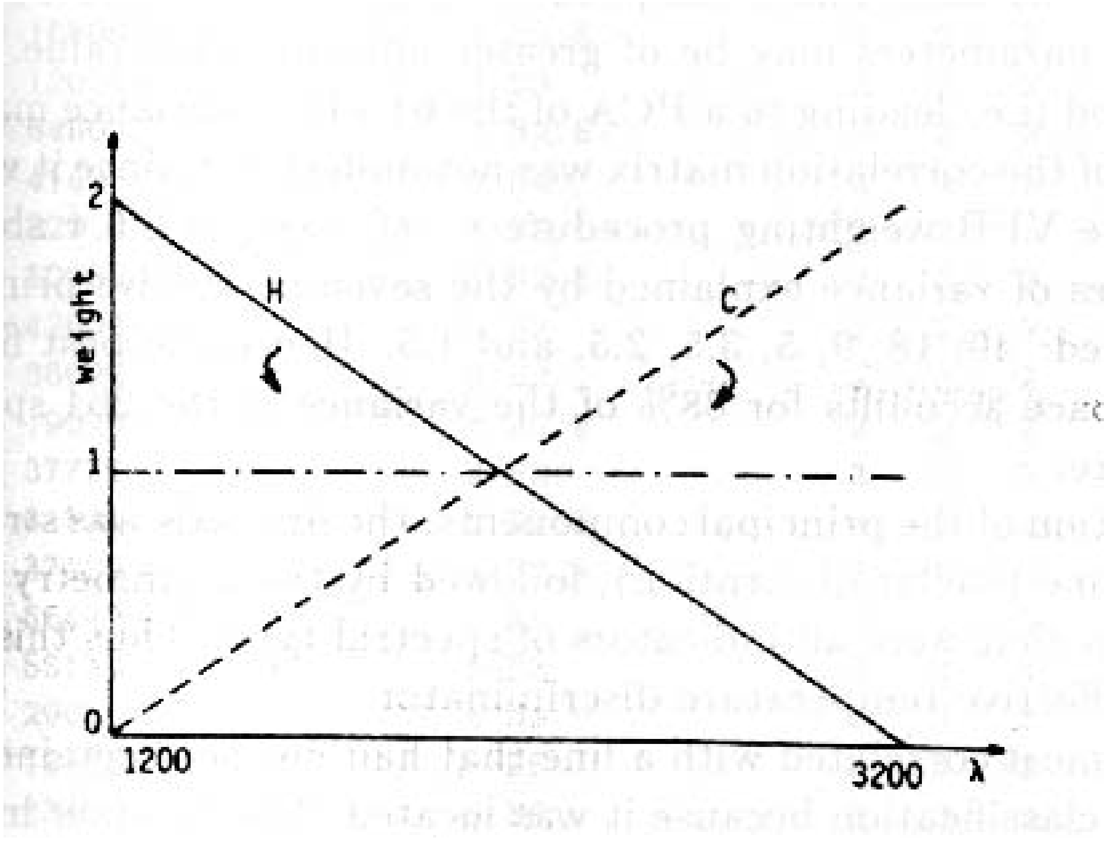
In order to enhance the importance of lines in the shortwavelength range for hot stars (where the signal is most significant), and similarly of lines in the longwavelength range for cool stars, the line intensities were weighted with the "variable Procrustean bed" (VPB) technique (introduced in Heck *et al.*, 1984b) through the formula

$$D_i' = D_i(1 + S - 2S(L_i - 1155)/2045)$$

where D_i is the unweighted depth of the i^{th} line, S is the asymmetry coefficient, and L_i is the wavelength (Å) of the i^{th} line ($i = 1$ to 60).

As illustrated in Figure 6.5, this corresponds, for an extreme hot star (curve H), to multiplying the shortwavelength–side line intensities by 2 and the longwavelength–side ones by 0. Those in between have a weight progressively decreasing from 2 to 0 as the wavelength increases. The situation is reversed for an extreme cool star (curve C), and all intermediate cases are weighted intermediately (in the sense of the arrows) in a similar way. The neutral case (all line intensities weighted by 1) corresponds to $S = 0$.

Here again, in a statistical parody of the deplorable mania of the Greek mythological robber Procrustes, the line intensities were forced to comply with the asymmetry coefficient (Procrustes' bed) which is variable with the star at



```
The abscissa represents increasing wavelength.
Curve H corresponds to the extreme case of hot stars (S = 1),
  and curve C to the extreme case of cool stars (S = -1).
The horizontal line corresponds to S = 0.
```

Figure 6.5: Weighting of the line intensities by the asymmetry coefficient through the Variable Procrustean Bed technique.

hand. In other words, while a statistical variable generally receives the same weight for all individuals of a sample, in this application the weight is a function of the individuals themselves.

Finally it may be noted that the number of variables used could be even more reduced by looking at correlations between those defined, and rejecting redundant variables from consideration.

6.7 Multivariate Analyses

We are given a 264 × 403 table, consisting of 264 spectra each measured on 403 flux values (slightly reduced from the original 410 flux values, by dropping insignificant extremities of the spectra: cf. section 6.5.1 above). We also have a

264×61 table, the parameters being the asymmetry coefficient and 60 weighted line intensity values. As a general rule, as analyses are carried out, it may be interesting or even necessary to discard anomalous spectra or confusing parameters. It may similarly be of interest to redo the initial coding of the data, to make use of data transformations, and so on.

6.7.1 Principal Components Analysis

A Principal Components Analysis (PCA) condenses the dimensionality of a parameter space. It may be used as a data "cleaning" stage, prior to a clustering. In the case of the 264×403 table of flux values, the 264 spectra were *centred*: i.e. the PCA was carried out on a 403×403 covariance matrix. The percentage variance explained by the first axis was 77.5%, by the first plane 89%, and by the best fitting 3–dimensional subspace 94%. The marked linearity of the data was identified as being temperature–related, but also had some luminosity effects. Greater negativity on the first axis was essentially associated with cooler stars and, for a given temperature, more luminous stars.

The PCA of the 264×61 table offers computational advantages, and because of the carefully selected parameters may be of greater interpretative value. The spectra were again centred (i.e. leading to a PCA of the 61×61 covariance matrix; an alternative of a PCA of the correlation matrix was not undertaken since it would destroy the effects of the VPB weighting procedure — cf. section 6.6.4 above). The following percentages of variance explained by the seven successive principal components were obtained: 49, 18, 9, 5, 3.5, 2.5, and 1.5. Hence the best fitting seven–dimensional subspace accounts for 88% of the variance of the 264 spectra in the 61 parameter space.

Regarding interpretation of the principal components, the first axis was strongly correlated with the $L\alpha$ line (stellar absorption), followed by the asymmetry coefficient, and by other lines that were all indicators of spectral type. Thus this first axis was essentially an effective temperature discriminator.

The second axis was most correlated with a line that had not been retained for the *Atlas* morphological classification because it was located close to other important lines. The three following ones discriminate mainly the effective temperature. The fifth line was not selected for the *Atlas* either.

The third and fourth axes were essentially discriminators of effective temperature among the cool types, while the fifth axis discriminated luminosity among hot stars.

6.7.2 Cluster Analysis

Among the cluster analyses carried out was a clustering on the seven principal components resulting from the PCA of the 264×61 matrix. Hierarchical clustering was used because, not knowing the inherent number of clusters, a range of groups could be studied (the 30, 40, and 50 group solutions, in particular, were examined). The minimum variance criterion was used for the desired objective of determining synoptic groups.

The results are shown in Table 6.3. The first column is the star identifier, the second is the IUE spectral classification (this symbolism is used in the *Atlas*) and the third, fourth and fifth columns are respectively the 30, 40 and 50 group solutions.

```
HD 8890                 s F8              1   1   1
HD 79447                d+B3              2   2   2
HD 36824                d B2.5            2   2   2
HD 192685               d B2.5            2   2   2
HD 32630                d B3              2   2   2
HD 37674                d B3              2   2   2
HD 168905               d B2.5            2   2   2
HD 120315               d B3              2   2   2
HD 64802                d B2.5            2   2   2
HD 4142                 d B4              2   2   2
HD 829                  d+B3              2   2   2
HD 100600               d B2.5            2   2   2
HD 42690                d B2              2   2   2
HD 36629                d B2              2   2   2
HD 190993               d B2.5            2   2   2
HD 37776                d B2              2   2   2
HD 61831                d B2.5            2   2   2
HD 37129                d+B2.5            2   2   2
HD 86440                s-B5              3   3   3
HD 53138                s+B3              3   3   3
HD 2905                 s+B1              3   3   3
HD 164353               s-B5              3   3   3
HD 23408                g B6              3   3   3
HD 83183                g+B6              3   3   3
HD 51309                g+B3              3   3   3
HD 46769                d B6              3   3   3
CPD -72 1184            g B0              3   3  49
HD 34078                d+O9.5            3   3  49
HD 152233               g O5              4   4   4
HD 93403                g O5              4   4   4
HD 210839               s O5              4   4   4
HD 152248               s O7              4   4   4
HD 46223                d O4              4   4   4
HD 46150                d O5              4   4   4
HD 47129                g+O7              4   4   4
HD 162978               g+O7              4   4  45
HD 48099                d O7              4   4  45
HD 165052               d O7              4   4  45
HD 213558               d A1              5   5   5
HD 87737                s A0              5   5   5
HD 41695                d A1              5   5   5
HD 156208               g A1              5   5   5
HD 86986                d A1              5   5   5
HD 29646                d A1              5   5   5
HD 20346                g:A2              5   5   5
HD 80081                d A2              5   5   5
HD 166205               d A2              5   5   5
HD 104035               s A2              5   5   5
HD 177724               d A0              5   5   5
HD 48250                d A2              5   5   5
HD 111775               d+A0              5   5   5
HD 58142                d+A1              5   5   5
HD 60778                d A2              5   5   5
HD 149212               g B9              5   5   5
HD 137422               g+A2              5   5   5
HD 62832                d A1              5   5   5
HD 11031                d A2              5   5   5
HD 167838              s+B5               5   5   5
HD 103287              d A0               5   5   5
HD 95418               d A1               5   5   5
HD 27962               d A2               5   5   5
HD 9132                d A1               5   5   5
HD 12301               s-B8               5   5   5
HD 166937              s+B8               5   5   5
HD 53367               g B0               5   5  43
HD 198478              s+B3               5   5  43
HD 77581               s+B0.5             5   5  43
HD 190603              s B2               5   5  43
HD 41117               s+B2               5   5  43
HD 199216              s-B2               5   5  43
HD 148379              s+B2               5   5  43
HD 206165              s-B2               5   5  43
HD 163181              s B1               5   5  50
HD 29335               d B6               6   6   6
HD 147394              d B5               6   6   6
HD 210424              d B7               6   6   6
HD 23480               d B6               6   6   6
HD 25340               d B5               6   6   6
HD 22928               g B5               6   6   6
HD 197392              g B8               6   6   6
HD 21071               d B7               6   6   6
HD 90994               d B6               6   6   6
HD 162374              d B6               6   6   6
HD 199081                    (example)    6   6   6
HD 31512               d B6               6   6   6
HD 37903               d B1.5             6   6  41
HD 37367               d+B2.5             6   6  41
HD 27396               d B4               6   6  41
HD 23060               d B3               6   6  41
HD 4727                d+B4               6   6  41
HD 34759               d B4               6   6  41
HD 83754               d B5               6   6  41
HD 52942               d B1.5             6   6  41
HD 142983              g B4               6   6  47
HD 50846               g+B5               6   6  47
HD 47755               d B3               6   6  47
HD 60753               d B3               6   6  47
HD 40136               d F2               7   7   7
HD 128167              d F3               7   7   7
HD 20902               s F5               7   7   7
HD 173667              d F6               7   7   7
HD 61421               d F5               7   7   7
HD 77370               d F4               7   7   7
HD 113139              d F2               7   7   7
HD 99028               g F3               7   7   7
HD 152667              s+B0.5             8   8   8
HD 47240               s+B1               8   8   8
HD 24398               s-B1               8   8   8
BD -9 4395             s B1.5             8   8   8
HD 47432               s+O9               8  36  36
HD 122879              s+B0               8  36  36
HD 37043               d+O9               9   9   9
HD 47839               d O8               9   9   9
HD 13268               g O8              10  10  10
HD 188209              s-O9.5            10  10  10
HD 30614               s+O9              10  10  10
BD +60 497             d O7              10  10  10
HD 48329               s G8              11  11  11
HD 16901               s G0              11  39  39
```

HD 93028	d+O9.5	12	12	12
HD 40111	g B1	12	12	12
HD 14633	d O9	12	12	12
HD 37061	d+B0.5	12	12	12
HD 218376	g B1	12	12	12
HD 3360	d+B1.5	13	13	13
HD 200120	d B0.5	13	13	13
HD 37023	d B0	13	13	13
HD 52918	d+B1	13	13	13
HD 63922	d+B0	13	13	13
HD 75821	g B0	13	13	13
HD 37744	d B1.5	13	13	44
HD 37020	d B0	13	13	44
HD 37042	d B0.5	13	13	44
HD 31726	d B2	13	13	44
HD 183143	s+B7	14	14	14
HD 147084	g+A5	14	14	14
HD 21291	s B9	14	14	14
HD 5448	d A4	14	14	14
HD 122408	d:A3	14	14	14
HD 122408	d:A3	14	14	14
HD 24432	s A0	14	14	14
HD 79439	d A5	14	14	14
HD 199478	s+B8	14	14	14
HD 17138	d A3	14	14	14
HD 21389	s A0	14	14	14
HD 6619	d A5	14	14	42
HD 40932	d A4	14	14	42
HD 116842	d A5	14	14	42
HD 197345	s A2	14	14	42
HD 76644	d A7	14	14	42
HD 216701	d A5	14	14	42
HD 189849	g A5	14	14	42
HD 87696	d A7	14	14	42
HD 11636	d A5	14	14	42
HD 216956	d A3	14	14	42
HD 76576	d:A5	14	14	42
HD 74180	s F2	14	14	42
HD 159561	g A5	14	14	42
HD 22049	d K	14	14	48
HD 193682	g:O4	15	15	15
HD 215835	d O5	15	15	15
HD 192281	d O4	15	15	15
HD 168076	d O4	15	15	15
HD 15570	s O4	15	15	15
HD 15629	d O5	15	15	15
HD 170153	d F6	16	16	16
HD 10700	d G	16	35	35
HD 85504	d+A0	17	17	17
HD 38206	d A0	17	17	17
HD 23850	g B8	17	17	17
HD 29365	d B8	17	17	17
HD 61429	d+B7	17	17	17
HD 108767	d B9	17	17	17
HD 58350	s+B5	17	17	17
HD 23753	g B8	17	17	17
HD 107832	d+B8	17	17	17
HD 10250	d B9	17	17	17
HD 23630	g B7	17	17	17
HD 148605	d B2	17	17	17
HD 201908	d B8	17	17	17
HD 23432	d B8	17	17	17

HD 86360	d+B9	17	17	17
HD 223778	d K3	18	18	18
HD 36512	d B0	19	19	19
HD 214680	d O9	19	19	19
HD 34816	d B0.5	19	19	19
HD 55857	d B0.5	19	19	19
HD 38666	d O9	19	19	19
HD 57682	d O9.5	19	19	19
HD 74273	d B1.5	19	34	34
HD 144470	d B1	19	34	34
HD 212571	d B1	19	34	34
HD 46056	d O8	20	20	20
HD 46149	d+O8	20	20	20
HD 52266	g O9	20	20	20
HD 46202	d+O9	20	20	20
HD 53974	g B0.5	20	20	20
HD 209481	g O9	20	20	20
HD 58946	d F2	21	21	21
HD 59612	s A5	21	21	21
HD 36673	s F0	21	21	21
HD 182640	d:F2	21	21	21
HD 27290	d F1	21	21	21
HD 90589	d F3	21	21	21
HD 65456	d A5	21	21	21
HD 27176	d F0	21	21	21
HD 147547	g F0	21	21	21
HD 161471	s F3	21	21	21
HD 12311	g F0	21	21	21
HD 206901	d F4	21	31	31
HD 89025	g F2	21	31	31
HD 202444	d:F3	21	31	31
HD 78362	g F3	21	31	31
HD 127739	d F3	21	31	31
HD 210221	s A3	21	33	33
HD 90772	s A5	21	33	33
HD 148743	s A7	21	33	33
HD 128620	d:G	22	22	22
HD 6582	d G	22	32	32
HDE 326330	g B1:	23	23	23
HD 64760	s-B0.5	23	23	23
HD 91316	s-B1	23	23	23
HD 123008	s+O9	23	23	23
HD 152249	s-O9	23	37	37
HD 152247	g+O9	23	37	37
HD 54439	d:B1.5	24	24	24
HD 52721	d B2:	24	24	24
HD 200310	d B1	24	24	24
HD 154445	d B1	24	24	24
HD 54306	d B1	24	24	24
HD 147933	d B1.5	24	24	24
HD 207330	g B2.5	24	38	38
HD 92741	g+B1	24	38	38
HD 149881	g:B0.5	24	38	38
HD 150898	g+B0:	24	38	38
HD 165024	g+B1	24	38	38
HD 51283	g B2	24	38	38
HD 219188	g B0.5	24	38	38
HD 173502	g B1	24	38	38
HD 167756	s-B0.5	24	38	38
HD 164794	d O3	25	25	25
HD 93205	d O3	25	25	25
HD 93204	d:O3	25	25	25

HD 93250	d 03	25	25	46
CPD-59 2600	d:0	25	25	46
HDE 303308	d 03	25	25	46
HD 93130	g 05	25	25	46
BD +60 594	g:08	26	26	26
BD +63 1964	g B0	26	26	26
HD 40893	g 09.5	26	26	26
CPD -41 7711	d B1	26	40	40
HD 164402	s-B0	27	27	27
HD 57061	g+09.5	27	27	27
HD 167264	s+B0	27	27	27
HD 10307	d G1.5	28	28	28
HD 102870	d F9	28	28	28
HD 20630	d G5	28	28	28
BD +60 2522	s 07	29	29	29
HD 109387	d+B7	30	30	30
HD 23302	g B6	30	30	30
HD 87901	d B7	30	30	30
HD 183914	d B8	30	30	30
HD 47054	d B8	30	30	30
HD 23324	d B8	30	30	30

Table 6.3: Star identifiers, IUE spectral classification, and 30–, 40– and 50–cluster solutions.

Note that the group sequence numbers used in the final three columns of Table 6.3 have no inherent meaning. Sorting of star identifiers was carried out to make the grouping clear, but in other respects has no inherent significance. The hierarchical structure between the 30, 40 and 50 cluster solutions (produced by the one algorithm) can be seen. It is inherent in the clustering algorithm used that the 50–group solution (the rightmost column of Table 6.3) is both a finer grouping and gives a more homogenous set of clusters than the 40 and 30 group solutions. The number of clusters looked at (i.e. 30, 40 and 50) were selected as reasonable figures, given the number of stars per group and the spectral classification (shown in Column 2 of Table 6.3, and now to be used for comparison).

Relative to spectral classification (Column 2), as the number of groups increases from 30 to 50, bigger groups are split into smaller more homogenous groups. This homogeneity is limited, firstly, by the relatively small size of the sample at hand (i.e. 264 stars) compared to the number of possible spectral classifications; and, secondly, by the underlying continuous physical variables (viz. effective temperature and intrinsic luminosity).

The overall approach developed here could also be used to predict UV spectral classification by assimilation. Stars with unknown UV spectral classification could be introduced into the general sample and the procedure applied to the whole population. The predicted classification could be taken as the spectral *mode* of the group to which the star has been assigned by the algorithm. The precision obtained would then be of the order of the *ranges* in spectral type and luminosity class of these groups.

As an example, star HD 199081 (on page 186) is in group number 6. The range here is $d - g$ (dwarf – giant) in luminosity, and B5–B7 in spectral type. The mode is *d B6*. Hence in the absence of further information, we would assign this star to spectral class *d B6*, with a tolerance indicated by the above–mentioned interval.

6.7.3 Multiple Discriminant Analysis

Cluster Analysis allowed good discrimination between spectral types, but was somewhat less effective in discriminating between luminosity classes alone. Hence, Multiple Discriminant Analysis was used to assess the discrimination between these classes. The following luminosity classes were used: *s* (supergiant), *g* (giant) and *d* (dwarf) (the star discussed at the end of section 6.7.2, Cluster Analysis, being assigned to *d*).

It was found that the three discriminant factors obtained allowed pairwise discrimination between these classes. Figure 6.6 shows a plot of *d* versus *s* in the plane defined by discriminant factors 1 and 2; while Figure 6.7 shows the plot obtained of discriminant factors 1 and 3 where the discrimination between *g* and *s* can be seen. Unclear cases lie close to separating lines, and unknown cases may be decided upon by investigating the relative position in the discriminant factor space.

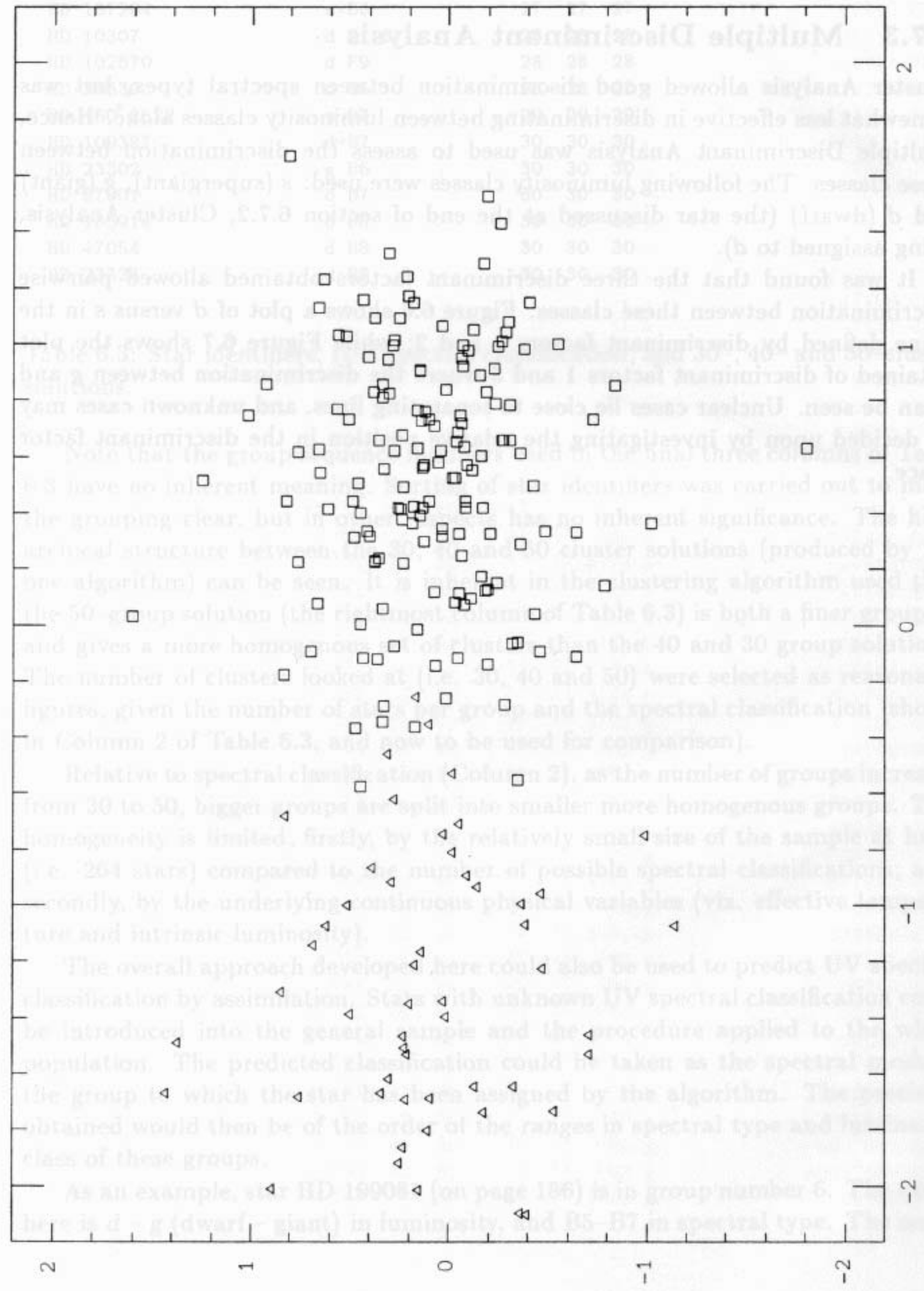


Figure 6.6: Dwarf and supergiant stars in the plane of discriminant factors 1 and 2.

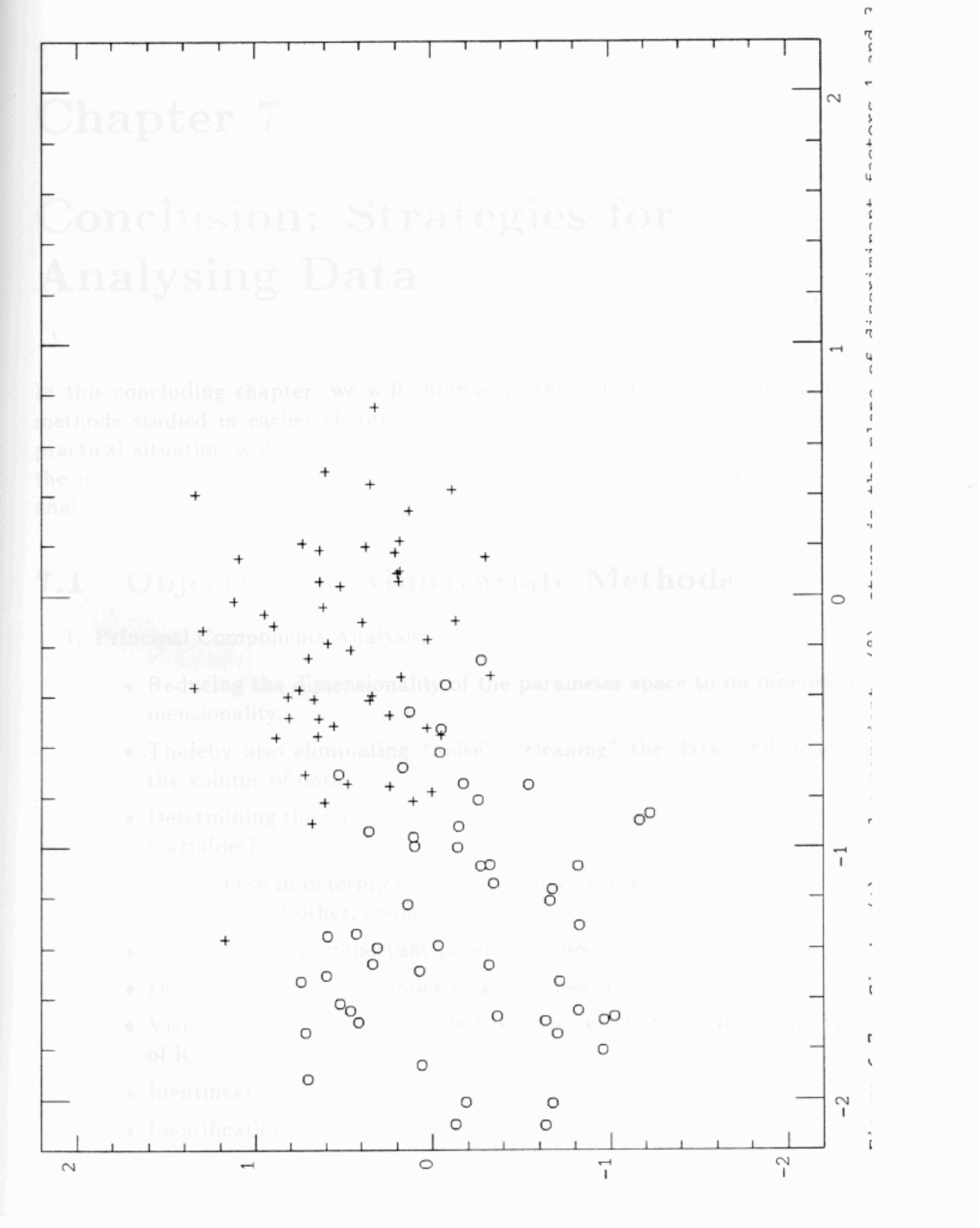


Figure 6.7: Giant and supergiant stars in the plane of discriminant factors 1 and 3.

Chapter 7

Case Study 2: Classification of GRBs

7.1 Presentation

As very few gamma-ray burst (GRB) sources have astronomical counterparts at other wavebands, empirical studies of GRBs have been largely restricted to the analysis of their gamma ray properties: bulk properties such as fluence and spectral hardness, and evolution of these properties within a burst event (Fishman & Meegan 1995). While bursts exhibit a vast range of complex temporal behaviors, their bulk properties appear simpler and amenable to straight-forward statistical analyses. Studies fall into two categories: examination whether GRB bulk properties comprise a homogeneous population or are divided into distinct classes; and search for relationships between bulk properties. Both types of study may lead to astrophysical insight, just as the distinction between main sequence stars and red giants and the measurement of a luminosity-mass relation along the main sequence assisted the development of stellar astrophysics early in the century.

The most widely accepted taxonomy of GRBs is the division between short-hard and long-soft bursts proposed by Dezelay et al. (1992) and Kouveliotou et al. (1993, henceforth K93). K93 noticed a bimodality in the burst duration variable T_{90} (time within which 90% of the flux arrived), suggesting the presence of two distinct types of bursts separated at $T_{90} \simeq 2$ sec. The short bursts have systematically harder gamma-ray spectra than longer bursts. The two groups seemed indistinguishable in most other bulk properties, although the larger group of long-soft bursts may have a subclass with a different fluence distribution (i.e., different $< V/V_m >$; Katz & Canel 1996) and the groups may have different Galactic latitude distributions (Belli 1997). Other researchers point to small groups of bursts with distinctive properties such as the soft-gamma repeaters (Norris et al. 1991), two possible classes with differing short-timescale variability (Lamb, Graziani & Smith 1993), fast-rise exponential-decay bursts (Bhat et al. 1994), and two types of bursts with different ratios of total fluence and >300 keV fluence (Pendleton et al. 1997).

A variety of relationships between burst properties have also been reported. Norris et al. (1995) find an anti-correlation between T_{90} (calculated after wavelet thresholding) and peak intensity, consistent with a cosmological time dilation. However, a positive correlation between T_{90} and total fluence is also seen which does not agree with the simplest cosmological interpretation (Lee & Petrosian 1997). Additional reported relationships include: T_{90} correlated with peak heights (Lestrade 1994), peak energy correlated with peak flux (Mallozzi et al. 1995), and peak duration anticorrelated with gamma-ray energy (Fenimore et al. 1995).

Most of these studies suffer from a failure to treat all of the bulk property variables in an unbiased and quantitative way. Astronomers typically examine univariate or bivariate distributions, sometimes constructing composite variables (such as hardness ratios) with pre-determined relationships to include one or two additional variables. But it is quite possible that the complex astrophysics producing GRBs will not manifest itself in simple bivariate plots, just as the division between short-hard and long-soft bursts is not evident in spectral variables alone (Pendleton et al. 1994). GRB catalogs, like most multiwavelength astronomical catalogs, are multivariate databases and should be treated with multivariate statistical methods that can objectively and effectively uncover structure involving many variables (Feigelson & Babu 1997). Two previous studies take a fully multivariate approach to understanding GRB bulk properties. Baumgart (1994) constructs a neural network taxonomy of 99 GRBs from the PVO satellite using 26 variables representing both bulk burst properties and detailed temporal characteristics (e.g. number of peaks, fractal dimension, wavelet transform crossings) and finds two or three distinct GRB classes. Bagoly et al. (1997) perform principal components and factor analyses of nine bulk property variables using 625 GRBs from the BATSE 3B catalog. They find that that the relationships in the database are determined principally by only three variables: an appropriately weighted fluence, a weighted burst duration, and (to a lesser extent) flux in the highest energy bin.

We note, however, that it can be dangerous to look for correlations prior to classification (or establishing the homogeneity) of the population. While the anticorrelation between hardness ratio and burst duration seen in full samples (K93) may be the manifestation of a single astrophysical process, it may alternatively reflect differences between distinct processes. The latter possibility is suggested by a reported hardness-duration positive correlation within the long-soft class of bursts (Dezalay et al. 1996; Horack & Hakkila 1997). Most multivariate analyses thus begin with a study of homogeneity and classification, and then investigate the variance-covariance structure (i.e. correlations) within each class.

This paper describes a multivariate analysis of GRBs from the Third BATSE Catalog (Meegan et al. 1996). After defining the sample (§2), we start with a simple statistical description of the variables and their bivariate relationships for entire dataset (§3). We then seek distinct types of clusters in two ways. First, a standard nonparametric agglomerative hierarchical clustering analysis is performed (§4) which reveals three distinct. The statistical significance of the third cluster is validated, under Gaussian assumptions, with MANOVA tests. Second, a parametric maximum likelihood clustering procedure is adopted which reveals the same three groups and confirms them at high statistical significance (§5). The variance-covariance structure of each group is then examined (§6). Results are assimilated in the discussion (§7).

Throughout the paper, we discuss our mathematical techniques to help the reader understand the complexities of multivariate analysis. From the vast literature in this subject, we recommend the following monographs for interested readers: Johnson & Wichern (1992) and Jobson (1992) for overviews of applied multivariate analysis; Hartigan (1975), Jain & Dubes (1988) and Kaufman & Rousseeuw (1990) for multivariate clustering algorithms; Murtagh & Heck (1987) and, more briefly, Feigelson & Babu (1996) for multivariate methodology in astronomy.

7.2 The GRB Sample and Statistical Software

Our sample is drawn from the Third Catalog of the *Burst and Transient Source Experiment* (BATSE) on board the Compton Gamma Ray Observatory. This 3B catalog has 1122 GRBs detected by BATSE between 1991 April 19 and 1994 September 19. The catalog is presented and fully described by Meegan et al. (1996). Our database was extracted from the on-line database www.batse.msfc.nasa.gov/data/grb/catalog in May 1996, which provides many properties of each burst. There are roughly eleven variables of potential astrophysical interest: two measures of location in Galactic coordinates, l and b; two measures of burst durations, the times within which 50% (T_{50}) and 90% (T_{90}) of the flux arrives; three peak fluxes P_{64}, P_{256} and P_{1024} measured in 64 ms, 256 ms and 1024 ms bins respectively; and four time-integrated fluences $F_1 - F_4$ in the 0-50 keV, 50-100 keV, 100-300 keV and $>$ 300 keV spectral channels respectively. Researchers commonly consider three composite variables: the total fluence, $F_T = F_1 + F_2 + F_3 + F_4$, and two measures of spectral hardness derived from the ratios of channel fluences, $H_{32} = F_3/F_2$ and $H_{321} = F_3/(F_1 + F_2)$. Due to the limitations of available multivariate statistical techniques, we ignore other variables of potential relevance including the heteroscedastic measurement errors of each quantity and truncation values associated with BATSE triggering operations.

Of the 1122 listed bursts, 807 have data on all the variables described above. Ten bursts listed with zero fluences were eliminated. Our sample thus has 797 BATSE GRBs. For some analyses, we also used a subset of 644 bursts with 'debiased' durations, T_{90}^d. Here the durations are modified to account for the effect that brighter bursts will have signal above the noise for longer periods than fainter bursts with the same time profiles (J. Norris, private communication).

Statistical analyses in §§3, 4 and 6 were conducted within the Statistical Analysis System *SAS/STAT*[1], a very large and widespread commercial statistical software package (SAS Institute Inc. 1989). *SAS/STAT* procedures *CLUSTER*, *GLM*, *PRINCOMP* and *VARCLUS* were used. The analysis in §5 was performed with the *MCLUST* software (Banfield & Raftery 1993; Fraley 1998). A stand-alone FORTRAN version for Gaussian mixtures is available on-line in the *StatLib* archive at http://lib.stat.cmu.edu/general/mclust, and versions with interface to the *S* statistical package are available at http://lib.stat.cmu.edu/S for both Gaussian and Poisson mixtures. For multivariate data visualization, we used the XGobi (Swayne, Cook & Buja 1991) program, available from *StatLib* at http://lib.stat.cmu.edu/general/XGobi. Hypertext links to a variety of public domain software for multivariate analysis, classification and visualization is available at the Penn State *StatCodes* Web site, http://www.astro.psu.edu/statcodes.

[1] *SAS/STAT* is a registered trademark of the SAS Institute Inc.

7.3 Statistical Properties of the Entire Sample

We are faced with a multivariate database of 797 objects and 15 variables (11 variables from the catalog, 3 composite variables, and T_{90}^{d}). Two initial problems are frequently faced in analyses of multivariate databases. First, variables with incompatible units and ranges must be compared. Units can be removed by normalization (e.g. replacing F_1 by F_1/F_{tot}), standardization (e.g. replacing F_1 by F_1/σ_{F_1} where σ is the sample standard deviation), or taking logrithms. Second, the dimensionality of the problem should be reduced, as many of the variables are closely interrelated either by construction or by astronomical circumstance. Although there are no mathematical rules regulating reduction of dimensionality, it can usefully be guided by a correlation matrix showing bivariate relationships and a principal components analysis showing multivariate relationships that are mainly responsible for structured variance in the data. Scientific reasoning can also be used to eliminate consideration of variables. We conducted a preliminary examination of data representations, correlation matrices and bivariate plots, and principal components analyses to facilitate choice of variables. When no mathematical preference arose, we selected variables most commonly used by previous researchers to facilitate comparison of results.

Our choices were as follows. We use log variables, rather than normalized or standardized variables. We kept information on burst intensity and spectra through F_{tot} and hardness ratios rather than with the original fluxes F_1-F_4. We initially eliminated P_{64} and P_{1024} from consideration, and later eliminated P_{256} when we found it contributed mainly noise to the clustering process. We chose to remove the location variables (l, b), already established by other researchers to be random for the entire sample, but use them later to test for isotropy of subsamples. The debiased $T_{90,d}$ is used only in special tests. Our analysis was thus performed in six or fewer dimensions using log T_{50}, log T_{90}, log F_{tot}, log P_{256}, log H_{321} and log H_{32}.

Tables 1 and 2 give basic statistics for these six variables: means, standard deviations, and bivariate values of Pearson's linear correlation coefficient r. The latter statistic for two variables V_1, V_2 is defined to be

$$r = \frac{\sum V_{1i}V_{2i} - \frac{1}{n}\sum V_{1i}\sum V_{2i}}{\sqrt{[\sum V_{1i}^2 - \frac{1}{n}(\sum V_{1i})^2][\sum V_{2i}^2 - \frac{1}{n}(\sum V_{2i})^2]}}, \tag{7.1}$$

where the sums are over $i = 1, \ldots, N$. For $N = 797$ and assuming bivariate normal populations, any $|r| > 0.013$ implies that a correlation between the two variables exists at a two-tailed significance level $P < 0.001$ (Beyer 1968, pp. 389 and 283). But from an astrophysical perspective, we might consider any relationship with $|r| < 0.1$ to be of little interest. Figure 1 shows the bivariate scatter plots.

Table 2 gives an simple linearized view of the integrated database. The two measures of duration and the two measures of spectral hardness have correlation near unity, indicating they are nearly redundant. F_{tot} and P_{256} are quite dissimilar: F_{tot} shows a strong correlation with burst duration (as found by Lee & Petrosian 1997) and no relation to hardness, while P_{256} shows no relation to duration but is mildly correlated with hardness (Mallozzi et al. 1995). Burst duration is anticorrelated with hardness (K93; Fenimore et al. 1995). The cosmological anticorrelation between duration and peak flux reported by Norris et al. (1995) is statistically significant, but accounts for only a few percent of the variance between these variables. The correlation matrix based on the debiased T^d_{90} values yields very similar results.

However, the scatter plots show a more complex story. First, many plots show inhomogeneous distributions inconsistent with the unimodal multinormal (i.e. multivariate Gaussian) population assumed by Pearson's r. The distributions often seem bimodal with asymmetrical non-Gaussian shapes. One outlier burst is also seen in several projections. We therefore consider the hypothesis that the sample consists of two or more distinct classes, and proceed to find the 'clusters' using well-established methods.

7.4 Nonparametric Hierarchical Cluster Analysis

7.4.1 Methodological Background

Unsupervised agglomerative hierarchical clustering is a procedure based on the successive merging of proximate pairs of clusters of objects in multivariate parameter space. It produces a clustering tree or dendrogram starting with N clusters of 1 member and ending with one cluster of N members. Hierarchical clustering is a nonparametric operation assuming no prior knowledge of the distributions of objects in multidimensional parameter space. Unfortunately, there are many possible ways to proceed; mathematics provides little guidance among the choices and no probabilistic evaluation of the results without the imposition of additional assumptions. The scientist must make four decisions to fully define the clustering procedure:

1. Creating *unit-free variables* is essential for meaningful treatment of objects in multivariate space (§3). A favorite choice by statisticians is standardization, where each variable is normalized by the standard deviation of the sample. Astronomers more commonly make logrithmic transformations or construct ratios of variables sharing the same units. We follow the tradition of GRB researchers by measuring spectral hardness with ratios of fluences having the same units, and making logrithmic transformations of all variables.

2. The *metric* defines the meaning of proximity between two objects or clusters. Common choices are the simple Euclidean distance between unit-free variables and the squares of Euclidean distances. We chose the former option for most of the analysis.

3. Several *merging procedures* can be used. One might begin by merging the clusters with the nearest neighbors. This is called Single Linkage clustering and is most familiar in astronomy where it is frequently called the friends-of-friends algorithm. It tends to produce long stringy clusters, and is equivalent to a well-known divisive clustering procedure known as pruning the minimal spanning tree. Complete Linkage proceeds by maximizing the distance between clusters, and leads to evenly bifurcating dendrograms. For most of our analysis, we choose Average Linkage where the distance between two clusters is the average of the pair of observations.

This is a compromise between Single and Complete Linkage and tends to give compact clusters. Specifically, the distance between clusters K and L is given by (e.g. SAS Institute Inc. 1989, pp. 529ff; Johnson & Wichern 1992, pp. 584ff.)

$$D_{KL} = |\overline{\mathbf{x}}_K - \overline{\mathbf{x}}_L|^2 + \frac{W_K}{n_K} + \frac{W_L}{n_L}, \tag{7.2}$$

where the bar indicates an unweighted mean, $W = \sum_{i=1}^{n_k} |\mathbf{x_i} - \overline{\mathbf{x}}|^2$, and n_k is the number of members of the k-th cluster. Another popular choice is Ward's minimum variance criterion where the distance between the two clusters is the ANOVA (analysis of variance) sum of squares between two clusters added up over all variables (Ward 1963),

$$D_{KL} = |\overline{\mathbf{x}}_K - \overline{\mathbf{x}}_L|^2 / (\frac{1}{n_K} + \frac{1}{n_L}). \tag{7.3}$$

If the sample is an unbiased mixture of multinormal clusters, this method joins clusters to maximize the likelihood at each level of the hierarchy.

4. As the procedure gives a hierarchy from N to one cluster, the user must choose what *level of the dendrogram* to report as scientifically important clusters. This choice can be assisted by examination of two statistics. The squared correlation coefficient, R^2, states the fraction of the total variance accounted for by a selected level of clustering,

$$R^2 = 1 - \frac{\Sigma_{j=1}^{g} W_j}{\Sigma_{i=1}^{n} |\mathbf{x_i} - \overline{\mathbf{x}}|^2} \tag{7.4}$$

where the numerator is summed over the g clusters at the g-th level of hierarchy. The squared semi-partial correlation coefficient, R^2_{sp} measures the difference in the variance between the resulting cluster and the immediate parent clusters normalized by the total sample variance,

$$R^2_{sp} = \frac{W_M - W_K - W_L}{\Sigma_{i=1}^{N} |\mathbf{x_i} - \overline{\mathbf{x}}|^2}. \tag{7.5}$$

R^2 thus tells how much of the scatter is explained by a given level of clustering, and R^2_{sp} tells how much improvement is achieved between levels.

We emphasize again that there is no mathematically 'best' choice, although extensive experience with problems in many fields has led to a preference for certain combinations (e.g. standardized variables, squared Euclidean distances and Ward's minimum variance criterion). Once a clustering procedure and level is chosen, the stablity of result can be investigated with k-means partitioning and its statistical significance can be estimated if one is willing to make parametric assumptions such as multinormal shapes for each cluster. We conducted extensive experiments with different choices.

7.4.2 Results

The last several levels of the clustering tree for the 797 GRBs using the six unit-free variables shown in Table 1, Average Linkage and a Euclidean metric is shown in Figure 2a with details in Table 3a. The action taken at each level is indicated in column 2 of Table 3, which may refer to a level higher in the tree which (for brevity) is not shown here. Two types of mergers are seen: the incorporation of 'twigs' of one or a few GRBs into a large preexisting 'trunk' (levels 1, 3, 4, and 5); and the union of two substantial branches into a single larger trunk (levels 2 and 6). The first type has little effect on the variance of the sample with $R^2_{sp} \leq 1\%$. The single GRB brought into the main trunk in level at level 1 is the distant outlier seen in several panels of Figure 1. The level 2 merger of clusters with 190 and 606 members is clearly the most important structure, accounting for roughly 53% of the variance of the entire sample. This is the bifurcation of the sample into two classes easily seen in Figure 1 and noted by K93 and others. The principal finding that is not immediately obvious from Figure 1 is the structure indicated at level 6. The main trunk of 599 bursts (plus a few twigs to be merged later) is divided into groups of 93 and 506 bursts. This division accounts for 10% of the total variance of the sample, indicated in both the R^2 and R^2_{sp} values.

We found that the twigs in the tree structure disappear if the peak flux P_{256} variable is omitted and the analysis is made in 5-dimensional space (Figure 2b and Table 3b). Here the largest cluster of 593 members is formed by the union of clusters with 107 and 486 bursts, again accounting for 10% of the sample variance. It is possible that P_{256} is a nuisance variable irrelevant to the basic astrophysics of GRBs, producing noisy 'twigs' seen in Table 3a and Figure 2.

We tested many variants of hierarchical clustering. We replaced Average Linkage hypothesis with Complete Linkage, Single Linkage and Ward's minimum variance criterion. The Ward's criterion computation, for example, gave three clusters with 468, 145 and 184 bursts. We clustered using nonparametric density estimation based on the 100 nearest neighbors, and clustered using the principal components rather than the observed variables. Various methods were tried with both the observed T_{90} values and debiased T^d_{90} values, with little effect on the results. All methods showed two strong clusters and the outlier but, in some cases, the third cluster appeared only weakly.

To proceed further, we choose a single clustering structure for detailed study: the 5-dimensional Average Linkage analysis (Table 3b) with three clusters — Class I with 486 bursts, Class II with 203 bursts, Class III with 107 bursts — and outlier. The membership of these clusters is given in Table 4, and four projections of the clusters onto two-dimensional scatter plots are shown in Figure 3. These are frames from the 'grand tour' movie of the 5-dimensional dataset provided by the XGobi software where each cluster is 'brushed' with a different symbol. Note that, in general, there is no reason why classification structure should be most evident in projections parallel to the variable axes shown in Figure 1. It is more important that the clusters show cohesion in many projections of the dataset. The grand tour of the 797 GRBs shows that Classes I, III and the outlier are very distinct in most projections. Class II often lies between Classes I and III (e.g. Figures 3a and 3b), but in other projections is offset from the line between Classes I and III (e.g. Figures 3c and 3d). It also appears elongated along some projections, while the larger Classes I and III appear roughly hyperspherical.

This analyses described here provide considerable evidence for three major clusters and an outlier. But there is some some worry that Class II is simply a group of bursts with properties intermediate between the Classes I and III and should not be viewed as a distinct subpopulation. While nonparametric hierarchical clusters methods can not address this question, it can be investigated with parametric methods.

7.4.3 Validation of the classification

Mathematically well-founded methods for evaluating the statistical significance of a proposed multivariate classification scheme are available under the assumption that the population is a multinormal mixture; that is, the objects of each class are drawn from multivariate Gaussians. All relationships between the variables must thus be linear (as in Table 2), although the relationships may differ between clusters. There is no requirement of sphericity, so that clusters may have shapes akin to pancakes or cigars with arbitrary orientations in multidimensional space. These methods fall under the rubric of multivariate analysis of variance (MANOVA).

The model can be expressed as follows (e.g. Johnson & Winchern 1992, pp. 246ff). For a p-dimensional dataset of g clusters each with n_l members, the i-th GRB in the j-th cluster gives a p-dimensional vector

$$\mathbf{X}_{ij} = \mu + \tau_j + \epsilon_{ij} \tag{7.6}$$

where μ is the overall population mean, τ_j is the offset of the j-th cluster mean from μ, and ϵ_{ij} are independent normal variables with zero mean representing the scatter of individual points about the mean. We test the null hypothesis

$$H_0 : \tau_1 = \tau_2 = \ldots = \tau_g = 0 \tag{7.7}$$

that the cluster means are not offset from each other. We construct two matrices of sums of squares and cross-products as follows:

$$\begin{aligned} \mathbf{B} &= \sum_{l=1}^{g} n_l(\overline{\mathbf{x}}_l - \overline{\mathbf{x}})(\overline{\mathbf{x}}_l - \overline{\mathbf{x}})' \\ \mathbf{W} &= \sum_{l=1}^{g}\sum_{j=1}^{n_l}(\overline{\mathbf{x}}_{lj} - \overline{\mathbf{x}})(\overline{\mathbf{x}}_{lj} - \overline{\mathbf{x}})'. \end{aligned} \tag{7.8}$$

Three tests statistics have been proposed to test the null hypothesis (e.g. SAS Institute Inc. 1989, pp. 17ff)):

$$\begin{aligned} \text{Wilk's Lambda} \quad \Lambda^* &= \det(\mathbf{W})/\det(\mathbf{B}+\mathbf{W}), \\ \text{Pillai's trace} \quad \mathbf{V} &= \text{trace}(\mathbf{B}(\mathbf{B}+\mathbf{W})^{-1}, \text{and} \\ \text{Hotelling's trace} \quad \mathbf{U} &= \text{trace}(\mathbf{W^{-1}B}). \end{aligned} \tag{7.9}$$

The distributions of these statistics have been determined mathematically. For example, for large $N = \sum_{l=1}^{g} n_l$, $-(N-1-(p+g)/2)\ln\Lambda^*$ has approximately a chi-squared distribution with $p(g-1)$ degrees of freedom (Wilks 1932; Bartlett 1938). More generally, the distributions are related to the non-central F distribution. For the 2-sample case, Hotelling's trace is commonly known as the Mahalanobis D^2 statistic. One can thus accept or reject the null hypothesis that the clusters have the same mean location at a chosen level of statistical significance.

The results of our MANOVA calculations are summarized in Table 4. The first row tests the hypothesis that the Classes I, II and III have the same mean with $N = 796$ bursts, omitting the outlier burst. The F values are very high in all cases, indicating that the clusters are different with extremely high statistical significance ($P << 10^{-4}$). For comparison, the $P = 0.01$ and 10^{-4} confidence levels correspond to $F =$??. and ??. for 10 degrees of freedom. (Note that it is not meaningful to quote probilities like $P = 10^{-8}$ as the tails of the distribution are poorly determined unless the sample size is extremely large.) This is the first *quantitative* demonstration that at least two clusters exist among GRBs, which was *qualitatively* reported by Delazay et al. (1992) and K93. The other rows in Table 4 test the hypotheses that each proposed class has the same mean as each other class. Again, the classes are found to be distinct with very high significance (for 5 degrees of freedom, $F =$??. corresponds to $P = 10^{-4}$).

7.5 Model-Based Maximum Likelihood Cluster Analysis

7.5.1 Methodological Background

In the previous section, we conducted a hierarchical clustering analysis without making assumptions regarding the shapes of the clusters, but needed the parametric assumption of normality to estimate the statistical significance of the resulting clusters. It is reasonable to conduct the entire analysis, both clustering and validation, within a model-based framework. We report here an analysis of this type again assuming that the GRB population consists of a mixture of multivariate Gaussian classes. This model is developed by Scott & Symons (1971), Murtagh & Raftery (1984), and Banfield & Raftery (1993). Application of the Bayesian Informatoin Criterion for validation is discussed by Dasgupta & Raftery (1998), and use of the EM Algorithm for computation is described by Celeux & Govaert (1995) and Fraley (1998).

In the model considered here, the p-dimensional observations $\mathbf{x}_i$ are drawn from g multinormal groups charaterized by a vector of parameters θ_k for $k = 1, \ldots, g$. Given observations $\mathbf{x} = x_1, x_2, \ldots, x_N$, let $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_N)^T$ be a cluster assignment vector, where $\gamma_i = k$ when x_i comes from the k-th group. Our goals are: to determine the number of GRB types. g; to determine the cluster assignment γ of each burst; and estimate the mean μ_k location and covariance matrix (containing standard deviations) $\mathbf{\Sigma}_k$ for each cluster.

Following Fraley (1998), the population density is expressed as follows:

$$\begin{aligned} f_k(\mathbf{x}|\theta) &\sim MVN(\mu_k, \mathbf{\Sigma}_k) \quad k = 1, \ldots, g \\ f(\mathbf{x}|\theta) &= \sum_{k=1}^{g} \pi_k f_k(\mathbf{x}|\theta) \qquad (7.10) \\ \sum_{k=1}^{g} \pi_k &= 1. \end{aligned}$$

where MVN means multivariate normal and π are the mixing probabilities associated with $\gamma = (\gamma_1, \ldots, \gamma_N)^T$ and ^{T} is the vector transform. We estimate the parameters using the principle of maximum likelihood, where the likelihood is

$$L(\theta,\gamma)=\prod_{i=1}^{N} f(x_i|\theta)$$

$$L(\mu_1,\ldots,\mu_k;\mathbf{\Sigma}_1,\ldots,\mathbf{\Sigma}_k)=$$

$$\prod_{k=1}^{g}\prod_{i\in I_k}(2\pi)^{p/2}|\mathbf{\Sigma}_k|^{-1/2}\exp\{-\frac{1}{2}(\mathbf{x}_i-\mu_k)^T\mathbf{\Sigma}_k^{-1}(\mathbf{x}_i-\mu_k)\}, \tag{7.11}$$

where $I_k=\{i:\gamma_i=k\}$ is the set of indices corresponding to observations belonging to the k-th group. For MVN populations, the maximum likelihood estimator of the group means μ_k is known to be the group average $\hat{\mu}=\overline{\mathbf{x}}=\sum_{i\in I_k}\mathbf{x}_i/n_k$, where n_k is the number of observations in the k-th group. It can then be shown algebraically that the log-likelihood is

$$l(\mathbf{\Sigma}_1,\ldots,\mathbf{\Sigma}_g;\gamma|\mathbf{x};\hat{\mu}_1,\ldots,\hat{\mu}_g)=$$
$$-\frac{pNlog(2\pi)}{2}-\frac{1}{2}\sum_{k=1}^{g}\{tr(\mathbf{W}_k\mathbf{\Sigma}_k^{-1})+n_k\log|\mathbf{\Sigma}_k|\}, \tag{7.12}$$

where

$$\mathbf{W}_k=\sum_{i\in I_k}(\mathbf{x}_i-\overline{\mathbf{x}}_k)(\mathbf{x}_i-\overline{\mathbf{x}}_k)^T \tag{7.13}$$

is the sample cross-product matrix for the k-th group. ??Should $\mathbf{W}$ be renamed to avoid confusion with $\mathbf{W}$ in the previous section??

The maximum likelihood solution is computationally straightforward in some cases. If all of the groups are hyperspheres with the same standard deviations (i.e. $\mathbf{\Sigma}_k=\sigma^2\mathbf{I}$), then the solution is identical to Ward's (1963) minimum variance criterion in agglomerative clustering. Other well-known solutions apply for hyperspheres of different standard deviations or identical hyperellipsoids. But, in the case of GRBs, we have no astrophysical reason to believe these simplifications occur. We thus treat the general case that the $\mathbf{\Sigma}_k$ contain covariances and differ between clusters (e.g. one cluster may be a hypercigar with arbitrary orientation in p-space, and another may be a hyperpancake with another orientation). In this case, the maximum likelihood solution corresponds to the cluster assignments γ that minimize (Scott & Symons 1971)

$$\sum_{k=1}^{g} n_k log|\frac{\mathbf{W}_k}{n_k}|. \tag{7.14}$$

Computation of the maximum likelihood solution can be embedded within an agglomerative hierarchical clustering procedure as described in §3. But the computation becomes cumbersome for problems with large N and/or large p, and a number of more efficient approaches have been proposed. The method used here (Fraley 1998) and implemented in the *MCLUST* code involves parameterization of the $\mathbf{\Sigma}_k$ matrices in terms of their eigenvectors and eigenvalues (analogous to a principal components analysis), and iterative alternation between hierarchical clustering steps and relocation of the cluster using the EM Algorithm. The EM (Estimation-Maximization) Algorithm (Dempster, Laird &

Rubin 1977), one of the most successful methods in modern statistics, is a procedure for iteratively maximizing likelihoods in a wide variety of circumstances. In the present application, one estimates the conditional probability that observation $\mathbf{x}_i$ belongs to the k-th group, and relocates the cluster means that maximizes the likelihood at each step of the hierarchical clustering procedure. Although the computational procedure has some limitations (e.g. convergence of the EM iterations is not guaranteed; clusters can not be extremely small), it is generally efficient and effective for Gaussian clustering problems.

Models that do not maximize the likelihood can be quantitatively compared to the best solution using a well-respected tool known as the Bayes Information Criterion (BIC; ??Schwarz 1978??; Akaike 1979). This issue is important in the present application since we wish to determine the statistical significance of (say) three *vs.* two classes of GRBs. The likelihood ratio of two models, also known as the Bayes factor, is (Kass & Raftery 1995)

$$\text{Bayes factor} = \frac{p(\mathbf{x}|M_2)}{p(\mathbf{x}|M_1)}. \tag{7.15}$$

The Bayes factor can be approximated by the BIC,

$$BIC = 2(l_1 - l_2) - (m_1 - m_2) logN, \tag{7.16}$$

where l_1 is the likelihood and m_1 is the number of parameters for one mixture model, and similarly for l_2 and m_2. The BIC (and its closely related Akaike Information Criterion) measure the balance between the improvement in the likelihood and the number of model parameters needed to achieve that likelihood. While the absolute value of the AIC or BIC is not informative, differences between two models give reliable measures of their relative validity and are widely used in maximum likelihood studies. The use of the BIC in validating clustering models is discussed by Dasgupta & Raftery (1998).

7.5.2 Results and Validation

To reduce the dimensionality of the problem and the complexity of the calculation, we eliminated the highly redundant T_{50} and H_{32} variables (see Figure 1) and considered only the 3 variables T_{90}, F_{tot} and H_{321} for the sample of 797 BATSE GRBs. The *MCLUST* model-based clustering procedure described above was run for trials of $g = 2, 3, \ldots, 36?$ groups. The resulting values of the relative BIC values BIC(g)$-$BIC(1) are plotted in Figure 4. The maximum BIC is achieved for three classes. Most importantly, the BIC value for $g = 3$ is ?? $\simeq$ 70 ?? above that for $g = 2$. The corresponding probability that the $g = 2$ model is as successful as the $g = 3$ model is $P < 10^{-?}$ (??ref??). This result strongly confirms the analysis in §3 regarding the existence of three statistical significant clusters.

We have also calculated the BIC for $g = 1, \ldots, 9$ with various constraints on the covariance matrix Σ such as hypersphericity and uniformly shaped ellipsoids. Spherical clusters give poor fits. Uniform ellipsoids give good fits with 4 and 8 clusters. But in all cases, the maximum likelihood asuming 2 clusters is much lower than the likelihood of ≥ 3 clusters.

The cluster assignment vector γ for the $g = 3$ model with unconstrained Σ is given in Table 4. About 85% of the assignments are the same as those obtained from the nonparametric hierarchical clustering procedure in §4, so that we note only differences between the two clustering results with * and # markings. The close agreement between the cluster assignments in the two methods supports the assumption that the clusters can be viewed as multivariate Gaussians with considerable accuracy.

7.6 Cluster Properties

We can now examine the properties of GRBs within each cluster with reasonable confidence that the populations are distinct from each other but internally homogeneous. These properties become inputs to astrophysical theories seeking to explain GRB bulk properties. Table 6a lists the means and standard deviations of the principal variables for each cluster based on both the nonparametric and model-based clustering procedures. The two methods give very similar results. The three types are well-separated in the burst duration variables: Cluster I bursts have the longest durations around $10 - 20$ seconds, Cluster III bursts have the shortest durations below 1 second, and Cluster II. bursts have intermediate durations around 2 seconds. Cluster II bursts are also intermediate in their fluences, although their fluence distribution overlaps that of the fainter Class III. bursts. The hardness ratios of all three clusters overlap somewhat, but here Class II bursts have the softest spectra and Class I bursts have intermediate spectra. We can thus classify the types in the three principal dimensions Duration/Fluence/Spectrum (Table 6c): Class I is long/bright/intermediate, Class II is intermediate/intermediate/soft, and Class III is short/faint/hard.

A major constraint for the interpretation of GRBs has been the remarkable isotropy of their spatial distribution in the celestial sphere. It is possible that, while the bulk of GRBs are isotropic and have an inferred extragalactic origin, some class of GRBs have significant anisotropy which would reflect a Galactic origin (see Lamb 1995). We apply four statistical tests for isotropy discussed by Briggs (1993) and applied by Briggs et al. (1996) to various subsamples of the 3rd BATSE Catalog of GRBs. The statistics are: $<\cos\theta>$, where θ is the angle between a burst and the Galactic center; $<\sin^2 b - \frac{1}{3}>$, where b is the Galactic latitude; Rayleigh-Watson $\mathcal{W}$; and Bingham $\mathcal{B}$. $<\cos\theta>$ tests the dipole moment around the Galactic center, $<\sin^2 b - \frac{1}{3}>$ tests the quadrapole moment with respect to the Galactic plane, $\mathcal{W}$ tests the dipole moment around any point in the celestial sphere, and $\mathcal{B}$ tests the quadrapole moment around any plane or two poles. The expected values for the four statistics assuming random isotropic distribution on the sphere are 0, 0, 3 and 5 respectively. The asymptotic distributions of these statistics are known.

Table 5c shows the results of this analysis for Clusters I-III, kindly calculated for us by Michael Briggs. No deviations from isotropy are found. The $<\cos\theta>$ and $<\sin^2 - \frac{1}{3}$ values lie within one standard deviation of the expected value for a random distribution. The $\mathcal{W}$ and $\mathcal{B}$ values must be larger than the expected value to indicate anisotropy. The only such case, Class II with $\mathcal{B} = \imath\ni\in$, has a deviation with very low significance ($Prob < 0.2$). We thus do not confirm the Belli's (1997) report of significant differences in spatial distributions of burst Classes I and III, although we did not specifically test the Galactic latitude distribution.

In principle, the relative populations of the three classes may be an important constraint on astrophysical theory. We find that Class I contains more than half of the bursts with the remainder divided roughly evenly between Class II and Class III (Table 6c). But we do not believe our analysis gives a precise census for two reasons. First, the exact assignments of individual bursts to clusters depend strongly on the detailed assumptions of the clustering algorithms. For example, Class II is larger than Class III in the 5-dimensional nonparametric procedure but is smaller in the 3-dimensional model-based procedure. Second, the numbers of weaker bursts in Classes II and III are strongly dependent on the details of the BATSE instrument's burst triggering process which produces a complicated truncation bias.

We look for structure within each of the clusters by computing the correlation coefficients similar to those in §3 for the entire sample. Results are given in Table 7. Here we see a systematic difference between the two clustering methodologies: nonparametric Average Linkage clustering tends to give stronger correlations between the variables than the parametric multinormal clustering. ?? Can this be explained in an intuitive fashion ?? The Average Linkage clustering shows positive correlations between fluence and spectral hardness and between fluence and duration in Classes I and III, but anticorrelations in Class II. Only the fluence-spectral relations in Classes I and III are convincingly present in the model-based results. We consider this result to be a possible but not secure result of our study.

7.7 Discussion

We thus find, using clustering and validation methods with different mathematical underpinnings, that three classes of GRBs are present in our subset of the Third BATSE Catalog. Most of the structure can be found using three fundamental burst properties, Duration/Fluence/Spectrum. The class properties and relation to previous research can be briefly summarized as follows:

Class I These long/bright/intermediate bursts correspond to the well-known populous long/soft class of K93 and others. Within this group, we do not confirm a hardness-duration correlation reported by Dezalay et al. (1996) and Horack & Hakkila (1997), but rather find a possible fluence-hardness correlation (Table 7a and 7b).

Class II The discovery of this group with intermediate/intermediate/soft properties is the principal result of this study. The group is easily distinguished in the projections of Figure 3, but can also be discerned in some panels of Figure 1. For example, it lies between Classes 1 and 2 in the $T_{50} - H_{32}$, $T_{90} - F_{tot}$and $T_{90} - H_{321}$ scatter plots.Figure 5 shows the projection of the classes onto the univariate T_{90}. Class II accounts for many, but not all, of the bursts in the small peak around $2 < T_{90} < 5$ sec between the major short and long duration peaks. It is possible that our Class II is related to the class of no-high-energy (NHE) burst and peaks discussed by Pendleton et al. (1997). These bursts have unusually weak F_4 emission, soft $50 - 300$ keV spectra, and low F_{tot}. However, the NHE class does not appear to exhibit a clear duration segregation from other bursts as we find for Class II. Class II does not appear to be the third cluster found by Baumgart (1994, see his Table 3), but the high dimensionality of his analysis prevents a simple comparison with our low dimensionality study.

Class III This short/faint/hard group and corresponds to the short/hard burst type of K93 and others. Fluence-duration and fluence-hardness correlations may tentatively be present within the class. Note that while the mean location of this type is consistent in the two clustering schemes, the population varies considerably between clustering algorithms.

Outlier BATSE trigger event 2757, or burst 3B 940114, is the outlier in the nonparametric analysis of §4 and is clearly visible in many projections in Figures 1 and 3. (The model-based analysis of §5 can not locate clusters with very few members and assigned this event to Class II.) It has an exceedingly soft hardness ratio and short burst duration. But examination of the original BATSE database shows that the $F_1 - F_4$ fluxes are very weak with large measurement uncertainties. The published 3B catalog gives only an upper limit to its total fluence and no estimate of its hardness ratio. The unusual properties of this burst are thus very likely to be spurious.

The multivariate analysis described here is not comprehensive and may not have uncovered all of the structure in the Third BATSE Catalog of bulk GRB properties. Our reduction of dimensionality may have been too severe omitting, for example, the potentially important F_4 as a distinct variable (Pendleton et al. 1997; Bagoly et al. 1997). Many methodological options were not exercised. For example, it would be valuable to repeatedly apply the k-means partitioning algorithm to the database under the assumption that three clusters are present (see Murtagh 1992 for an astronomical application of this method), check for skewness or kurtosis in the clusters, and undertake an oblique decision tree analysis to give analytical formulation to hyperplanes separating the clusters. (see White 1997). Codes for these and many other multivariate techniques are pubically available through the Web metasite *StatCodes* at www.astro.psu.edu/statcodes.

However, the efforts described here are far more capable of finding and quantifying clustering in the database than most previous analyses (§1). Previous studies have been based on qualitative rather than quantitative procedures for identifying structures, and provide no statistical validation of their claims. It is thus not surprising that we uncovered structure missed by previous researchers. In particular, our confidence in the presence of a third cluster, Class II, is strong. Two completely independent mathematical procedures (§4 and §5) found very similar structure, each validated with high statistical confidence.

It is possible that the clustering reported here is indeed present in the database, but does not have an astrophysical origin. The complex triggering mechanism of the BATSE instrument mechanism and biases in bulk property values at low signal-to-noise ratio are two problems that probably affect the multivariate structure. However, we believe such instrumental biases will generally affect the number of bursts found in some regions of the multivariate hyperspace and may alter the location of clusters, but are unlikely to cause the appearance of clustering that is not present in the underlying population. Furthermore, the third cluster we uncovered lies, in most projections, between the two clusters of K93. Observational selection effects usually modify the edges of distributions rather than their interiors.

We conclude that the Third BATSE Catalog shows three statistically significant types of bursts (Duration/Fluence/Spectrum): long/bright/intermediate, intermediate/intermediate/soft, and short/faint/hard. These types are likely to be astrophysically real and not caused by observational selection. This finding should be considered an important input into astrophysical theories for GRBs. However, statistical anlaysis is unable to distinguish between burst types representing fundamentally different astrophysical processes, and distinct conditions within a single astrophysical model.

Our results can be confirmed and extended in two fashions. First, the analysis described here can be validated with several hundred more bursts collected by BATSE since the September 1994 cutoff in the database used here. Second, following Baumgart (1994), the dimensionality of the problem can be enlarged to include detailed characteristics of the burst temporal behaviors. Burst smoothness *vs.* peakiness, charactertic wavelet scales, spectral evolution, and other parameters can be included. With this enlarged database, one can perform both an unsupervised exploratory cluster analysis similar to that described here, and a MANOVA-type analysis that assumes the existence of the three groups to determine whether the clusters have distinctive temporal properties.

7.8 Bibliography

1. Akaike, H. 1979, Biometrika, 66, 237

2. Babu, G. J. & Feigelson, E. D. 1996, Astrostatistics (London:Chapman & Hall)

3. Bagoly, Z., Mészáros, A., Horváth, I., Balázs, L. G. & Mészáros, P. 1997, ApJ, submitted

4. Banfield, J. & Raftery, A. 1993, Biometrics, 49, 803

5. Bartlett, M. S. 1938, Proc. Camb. Phil. Soc., 34, 33

6. Baumgart, C. W. 1994, in Applications of Artificial Neural Networks V, edited by S. K. Rogers & D. W. Ruck, SPIE Proc. vol. 2243, (Bellingham WA:SPIE), 552

7. Belli, B. M. 1997, ApJ, 479, L31

8. Beyer, W. H. (editor) 1968, Handbook of Tables for Probability and Statistics, 2nd ed. (Boca Raton FL:CRC Press)

9. Bhat, P. H., Fishman, G. J., Meegan, C. A., Wilson, R. B., Kouveliotou, C., Paciesas, W. S., Pendleton, G. N. & Schaefer, B. E. 1994, ApJ, 426, 604

10. Briggs, M. S. 1993, ApJ, 407, 126

11. Briggs, M. S., Paciesas, W. S., Pendleton, G. N., Meegan, C. A., Fishman, G. J., Horack, J. M., Brock, M. N., Kouveliotou, C., Hartmann, D. H. & Hakkila, J. 1996, ApJ, 459, 40

12. Celeus, G. & Govaert, G. 1995, Pattern Recognition, 28, 781

13. Dasgupta, A. & Raftery, A. E. 1998 , J. Amer. Stat. Assn. in press

14. Dempster, A. P., Laird, N. M. & Rubin, D. B. 1977, J. Royal Stat. Soc. B., 31, 1

15. Dezalay, J.-P., Barat, C., Talon, R., Sunyaev, R., Terekhov, O. & Kuznetsov, A. 1992, in Gamma-Ray Bursts, edited by W. S. Paciesas & G. J. Fishman, AIP Conf. 265, (NY:AIP), 304

16. Dezalay, P.-P., Lestrade, J. P., Barat, C., Talon, R., Sunyaev, R., Terekhov, O. & Kuznetsov, A. 1996, ApJ, 471, L27

17. Feigelson, E. D. & Babu, G. J. 1997, in New Horizons from Multiwavelength Sky Surveys, edited by H. MacGillivray and B. Lasker, IAU 179, Dordrecht:Kluwer, in press

18. Fenimore, E. E., in't Zand, J. J., Norris, J. P., Bonnell, J. T. & Nemiroff, R. J. 1995, ApJ, 448, L101

19. Fishman, G. J. & Meegan, C. A. 1995, ARAA, 33, 415

20. Fraley, C. 1998, SIAM J. Sci. Computing, in press

21. Hartigan, J. A. 1975, Clustering Algorithms (NY:Wiley)

22. Horack, J. M. & Hakkila, J. 1997, ApJ, 479, 371

23. Jain, A. K. & Dubes, R. C. 1988, Algorithms for Clustering Data (Amsterdam:North-Holland)

24. Jobson, J. D. 1992, Applied Multivariate Data Analysis, 2 vols. (NY:Springer)

25. Johnson, R. A. & Wichern, D. W. 1992, Applied Multivariate Statistical Analysis, 3rd ed., (Englewood Cliffs NJ:Prentice Hall)

26. Kass, R. E. & Raftery, A. E. 1995, J. Amer. Stat. Assn., 90, 773

27. Katz, J. I. & Canel, L. M. 1996, ApJ, 471, 915

28. Kaufman, L. & Rousseeuw. P. J. 1990, Finding Groups in Data (NY:Wiley)

29. Kouveliotou, C., Meegan, C. A., Fishman, G. J., Bhat, N. P., Briggs, M. S.,

30. Koshut, T. M., Paciesas, W. S. & Pendleton, G. N. 1993, ApJ, 413, L101 (K93)

31. Lamb, D. Q. 1995, PASP, 107, 1152

32. Lamb, D. Q., Graziani, C. & Smith, I. A. 1993, ApJ, 413, L11

33. Lee, T. T. & Petrosian, V. 1997, ApJ, 474, L37

34. Lestrade, J. P. 1994, ApJ, 429, L5

35. Mallozzi, R. S., Paciesas, W. S., Pendleton, G. N., Briggs, M. S., Preece, R. D., Meegan, C. A. & Fishman, G. J. 1995, ApJ, 454, 597

36. Meegan, C. A., et al. 1996, ApJS, 106, 65

37. Murtagh, F. & Heck, A. 1987, Multivariate Data Analysis (Dordrecht:Reidel)

38. Murtagh, F. & Raftery, A. E. 1984, Pattern Recognition, 17, 479

39. Norris, J. P., Hertz, P., Wood, K. S. & Kouveliotou, C. 1991, ApJ, 366, 240

40. Norris, J. P., Bonnell, J. T., Nimiroff, R. J., Scargle, J. D., Kouveliotou, C., Paciesas, W. S., Meegan, C. A. & Fishman, G. J. 1995, ApJ, 439, 542

41. Pendleton, G. N., Paciesas, W. S., Briggs, M. S., Koshut, T. M., Fishman, G. J., Meegan, C. A., Wilson, R. B., Harmon, A. B. & Kouveliotou, C. 1994, ApJ, 431, 416

42. Pendleton, G. N., Paciesas, W. S., Briggs, M. S., Preece, R. D., Mallozzi, R. S., Meegan, C. A., Horack, J. M., Fishman, G. J., Band, D. L., Matteson, J. L., Skelton, R. T., Hakkila, J., Ford, L. A., Kouveliotou, C. & Koshut, T. M. 1997, ApJ, 489, 175

43. SAS Institute Inc. 1989, SAS/STAT User's Guide, Version 6, Fourth Edition, 2 volumes, (Cary NC:SAS Institute Inc.)

44. Schwartz, 1978, ??????

45. Scott, A. J. & Symons, M. J. 1971, Biometrics, 27, 387

46. Swayne, D. F.., Cook, D. & Buja, A. 1991, User's Manual for XGobi, a Dynamic Graphics Program for Data Analysis Implemented in the X Windows System, BellCore Tech. Memo.

47. Ward, J. H. 1963, J. Amer. Stat. Assn., 58, 236

48. Wilks, S. S. 1932, Biometrika, 24, 471

Table 7.1: Average GRB properties for the entire sample

Variable	Mean	S.D
log T_{50} (sec)	0.55	0.92
log T_{90} (sec)	0.96	0.92
log F_{tot} (erg cm^{-2})	-5.61	0.76
log P_{256} (photons s^{-1} cm^{-2})	0.16	0.45
log H_{321}	0.25	0.33
log H_{32}	0.48	0.30

Table 7.2: Correlation coefficients for the entire sample

	log T_{50}	log T_{90}	log F_{tot}	log P_{256}	log H_{321}	log H_{32}
log T_{50}	1.00					
log T_{90}	0.97	1.00				
log F_{tot}	0.63	0.66	1.00			
log P_{256}	-0.01	0.04	0.59	1.00		
log H_{321}	-0.36	-0.36	0.02	0.24	1.00	
log H_{32}	-0.35	-0.35	-0.00	0.19	0.96	1.00

Table 7.3: Average linkage hierarchical cluster analysis

Level	Merger	Members	R^2_{sp}	R^2
(a) Six dimensional analysis				
8	10 + 15	506	0.08	0.65
7	14 + 137	93	0.00	0.65
6	8 + 7	599	0.10	0.55
5	9 + 266	188	0.00	0.55
4	5 + 26	190	0.00	0.55
3	6 + 12	606	0.01	0.54
2	3 + 4	796	0.53	0.01
1	2 + 616	797	0.00	0.00
(b) Five dimensional analysis				
6	15 + 21	107	0.01	0.70
5	10 + 8	486	0.01	0.69
4	7 + 20	203	0.01	0.68
3	6 + 5	593	0.10	0.58
2	3 + 4	796	0.58	0.00
1	2 + 616	797	0.00	0.00

Table 7.4: Class membership of bursts based on 5-dimensional Average Linkage clsutering (3B trigger number)

(a) Class I (486 bursts)

107*	472	816	1159	1533	1657	1982	2138	2304	2431*	2703	2877	3003	3109	
109	473	820	1192	1540	1660	1989	2140	2306*	2432	2560	2706	2889	3005	
110	503	824	1196	1541	1661	1993	2143	2309	2435	2569	2709	2890	3011	
111	540*	825	1197	1546	1663	1997	2148	2310	2436	2570	2711	2891	3012*	
114	543*	829	1200	1551	1667	2018	2149	2311	2437*	2581	2725*	2894*	3015	
121	548	840	1213	1552	1676	2019	2151	2315	2438	2586	2727	2897	3017	
130	549	841	1218*	1558	1683*	2037	2156	2316	2440	2589	2728*	2898	3026*	
133	559	867*	1235	1559	1687	2041*	2181	2321	2441	2593	2736*	2900*	3029	
143	563	869	1244	1561	1700*	2044*	2187*	2324	2443	2600	2749*	2901*	3032	3
148	577	907	1279	1567	1704	2047	2188	2325	2446	2603	2751	2913	3035	
160	591	927*	1288	1574	1709*	2053	2189	2328	2447	2606	2753	2916	3039*	
171	594	938	1291	1578	1711	2061	2190	2329	2450	2608	2770	2919	3040	
204	606	946	1303	1579	1712	2067*	2191	2340	2451	2610	2774	2922	3042	3
211	630	973	1318	1580	1714	2069	2193	2344	2452	2611	2775	2924	3055	3
214	647	999*	1384	1586	1717	2070	2197	2345	2472	2619	2780	2925	3056	
219	658	1009	1385	1590	1730	2074	2202	2346	2476	2620	2790	2927	3057	3
222	659	1025*	1390	1601	1731	2077	2203*	2362	2477	2628	2793	2929	3067	
223	660	1036	1396	1604	1733	2079	2204	2367	2482	2634	2797	2931	3070	3
226	673	1039	1406	1606	1734*	2080	2211	2371	2484*	2636	2798	2932*	3071	
235	676	1042	1419	1609	1740	2081	2213	2373	2495	2640*	2799*	2947	3072*	
237	678	1046	1425	1611	1742	2083	2219*	2375	2496	2660	2812	2948	3074	
249	685*	1085	1432	1614	1806	2087*	2228	2380	2500	2662	2815	2950	3075	
257	686*	1086	1440	1623	1807	2090	2232	2383	2505	2663	2825	2953	3076	
288	692	1087	1446	1625	1815	2093	2233	2385*	2508	2664	2831	2958	3080	
332	704	1122	1447	1626	1819	2101	2230	2387	2510	2665	2843	2961	3084	
351	717*	1123	1449	1628	1830	2102*	2244	2391	2511	2671*	2852	2984	3085	
394	741*	1126	1452*	1642	1883	2106	2252	2392	2519	2681	2853*	2985	3091	
398	761	1141	1456	1646	1885	2110	2253	2394	2522	2688	2855	2992	3093	
404	764	1148	1458	1651	1886	2111	2267	2405	2528	2691	2856	2993	3100	
408	773	1150	1467	1652	1922	2112	2276	2419	2530	2695	2857	2994	3101	
451	795	1152	1468	1653	1924	2119*	2277	2428	2533	2696	2862	2996	3102	
467	803*	1156	1472	1655	1956	2122	2287	2429	2537	2697	2863	2998	3103	
469	815	1157	1515	1656	1967	2133	2298	2430	2541	2700	2864	3001	3105	

?? One burst is missing from this list ??

(b) Class II (203 bursts)

138	512	856	1154	1635	2003	2146	2291	2384	2523	2693	2846
185	537#	878	1211	1636#	2040	2155	2312	2395	2529	2701	2849
207	547	906	1223	1659	2043	2159	2317#	2434	2536	2715#	2851
218#	551	909	2389	1662	2049	2161	2320	2448	2564	2748	2860#
229	568	936#	1308	1665	2068	2163	2326	2449	2583	2755	2873
254	575	1051	1359	1680	2095	2167	2327	2454#	2585	2788	2879
289	603#	1073	1404#	1694	2099#	2201	2330	2463	2597	2795	2892
297	677	1076	1453	1719#	2103	2205	2332	2464	2599	2800	2896
298	729	1088	1461	1736	2115	2206	2352	2485	2614	2801	2910
432	788	1096	1463	1741	2117	2217#	2353	2487	2615	2810#	2918
444	799	1097	1481	1760	2125	2220	2357	2502	2623#	2814	2933
474	809	1102	1518	1791	2126	2265	2360	2504	2632#	2821	2952
480	830	1112	1553#	1851	2132	2268#	2365#	2512	2649	2823	2964
486	836	1128	1566	1953#	2142#	2273	2372	2513#	2679#	2828	2966
491	845#	1129	1588	1968	2145	2288	2377#	2514	2690	2834	2973

(c) Class III (107 bursts)

105	493	752	1120	1298	1492	1974	2207	2381	2458	2750	2880
108	501	753	1125	1306	1634	2035	2230	2382	2460	2760	2917
179	516	755	1145	1346	1637	2056	2254	2393	2515	2776	2944
228	526	834	1153	1382	1664	2105	2283	2401	2633	2830	2945
373	555	914	1167	1416	1679	2114	2347	2423	2641	2844	2951
401	680	942	1190	1435	1693	2129	2349	2424	2677	2848	2980
414	690	974	1204	1439	1701	2133	2358	2442	2680	2850	2986
465	734#	1114	1221	1443	1747	2152	2368	2453	2719	2861	2990

(d) Outlier

2757*

∗ Placed into Class II by Gaussian model-based clustering procedure

Placed into Class III by Gaussian model-based clustering procedure

?? Format of Table 4 will be fixed. Requires AASTeX deluxetable style file. ??

Table 7.5: Multivariate analysis of variance statistical tests

Clusters	Wilk's		Pillai's		Hotelling-Lawley's		Degrees of
	Λ∗	*F*	Trace	*F*	Trace	*F*	freedom
I, II, III	0.153	245.	0.934	139.	4.96	391.	10
I, II	0.159	719.	0.840	719.	5.27	719.	5
I, III	0.515	111.	0.485	111.	0.94	111.	5
II, III	0.301	141.	0.699	141.	2.32	141.	5

?? Some F values seem wrong. Text is missing probabilities. ??

Table 7.6: Cluster properties

Variable	Method[1]	Cluster		
		I	II	III
	(a) Means and standard deviations			
log T_{50}	NP	1.13 ± 0.44	0.31 ± 0.29	-0.80 ± 0.41
	MB	1.22 ± 0.39	0.29 ± 0.41	-0.91 ± 0.35
log T_{90}	NP	1.55 ± 0.40	0.60 ± 0.37	-0.42 ± 0.44
log F_{tot}	NP	-5.21 ± 0.59	-6.10 ± 0.37	-6.37 ± 0.57
	MB	-5.13 ± 0.58	-5.93 ± 0.47	-6.46 ± 0.54
log H_{321}	NP	0.19 ± 0.27	0.07 ± 0.43	0.51 ± 0.27
	MB	0.21 ± 0.26	0.16 ± 0.40	0.52 ± 0.28
log H_{32}	NP	0.43 ± 0.23	0.34 ± 0.41	0.70 ± 0.26
		(b) Isotropy		
$<\cos\theta>$	NP	0.015	-0.041	0.010
$<\sin^2 b - 1/3>$	NP	-0.012	-0.025	0.028
Rayleigh-Watson	NP	0.39	1.28	1.80
Bingham	NP	2.02	7.32	1.95
		(c) Summary		
Number	NP	486	203	107
	MB	426	170	201
Duration		long	intermediate	short
Fluence		bright	intermediate	faint
Spectrum		intermediate	soft	hard

[1] NP = nonparametric clustering analysis in 5 dimensions (§3);
MB = model-based clustering analysis in 3 dimensions (§4)

Table 7.7: Correlation coefficients within clusters

	log T_{50}	log T_{90}	log F_{tot}	log H_{321}	log H_{32}
(a) Class I – Nonparametric clustering					
$\lvert r\rvert > 0.??$ corresponds to $P < 0.001$ significance level					
log T_{50}	1.00				
log T_{90}	0.88	1.00			
log F_{tot}	0.10	0.22	1.00		
log H_{321}	-0.11	-0.08	0.39	1.00	
log H_{32}	-1.11?	-0.08	0.38	0.97	1.00
(b) Class I – Model-based clustering					
$\lvert r\rvert > 0.??$ corresponds to $P < 0.001$ significance level					
log T_{50}	1.00				
log T_{90}	N/A	N/A			
log F_{tot}	N/A	0.01	1.00		
log H_{321}	N/A	-0.01	0.06	1.00	
log H_{32}	N/A	N/A	N/A	N/A	N/A
(c) Class II – Nonparametric clustering					
$\lvert r\rvert > 0.??$ corresponds to $P < 0.001$ significance level					
log T_{50}	1.00				
log T_{90}	0.89	1.00			
log F_{tot}	-0.05	-0.02	1.00		
log H_{321}	-0.00	-0.08	-0.21	1.00	
log H_{32}	-0.00	-0.08	-0.26	0.96	1.00
(d) Class II – Model-based clustering					
$\lvert r\rvert > 0.??$ corresponds to $P < 0.001$ significance level					
log T_{50}	1.00				
log T_{90}	N/A	N/A			
log F_{tot}	N/A	-0.03	1.00		
log H_{321}	N/A	-0.05	0.02	1.00	
log H_{32}	N/A	N/A	N/A	N/A	N/A
(e) Class III – Nonparametric clustering					
$\lvert r\rvert > 0.??$ corresponds to $P < 0.001$ significance level					
log T_{50}	1.00				
log T_{90}	0.08	1.00			
log F_{tot}	0.26	0.32	1.00		
log H_{321}	-0.13	-0.16	0.40	1.00	
log H_{32}	-0.10	-0.12	0.44	0.90	1.00
(f) Class III – Model-based clustering					
$\lvert r\rvert > 0.??$ corresponds to $P < 0.001$ significance level					
log T_{50}	1.00				
log T_{90}	N/A	N/A			
log F_{tot}	N/A	0.03	1.00		
log H_{321}	N/A	-0.01	0.07	1.00	
log H_{32}	N/A	N/A	N/A	N/A	N/A

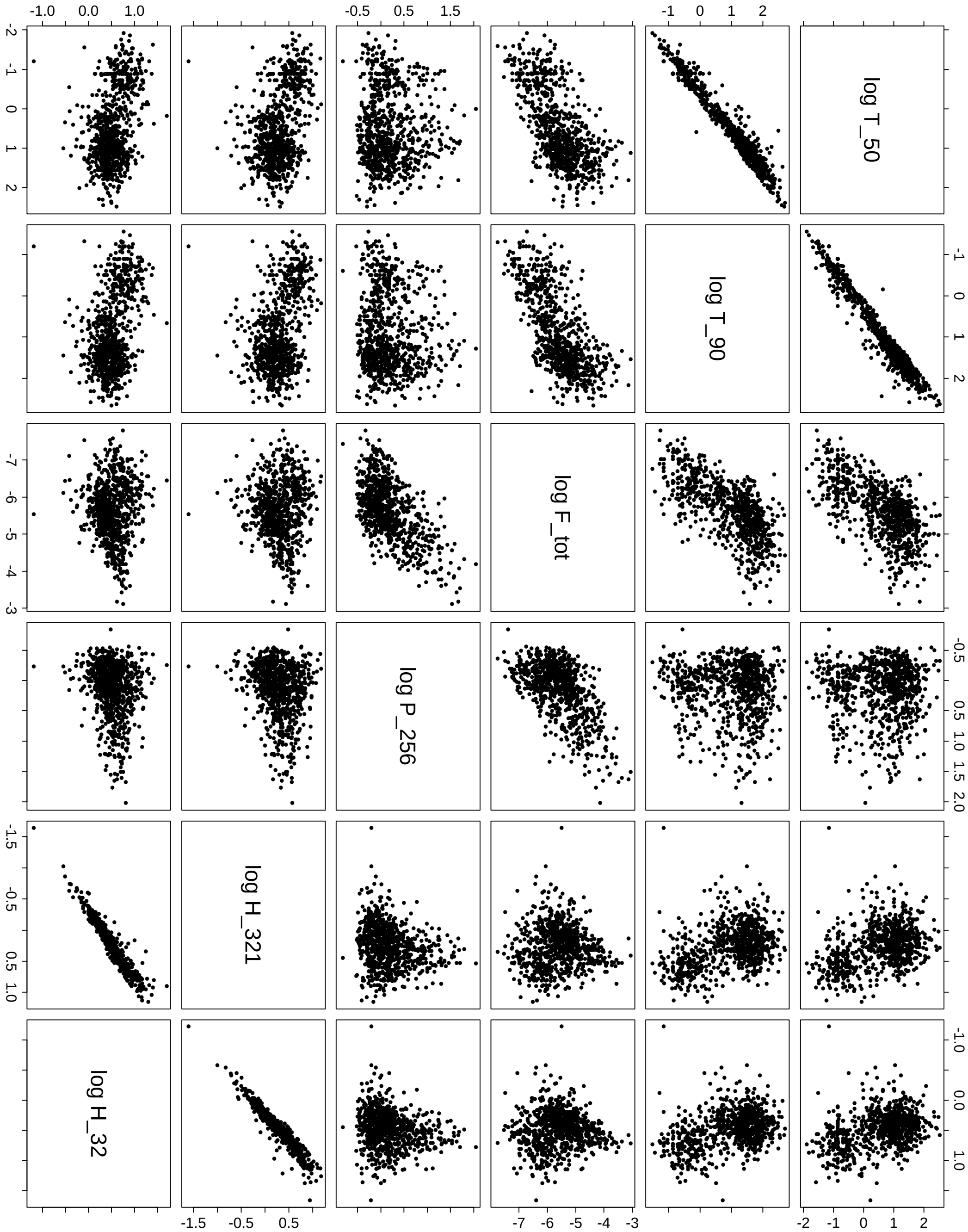


Figure 7.1: Mosaic of scatter plots of six bulk properties for the $N = 797$ GRBs from the Third BATSE Catalog.

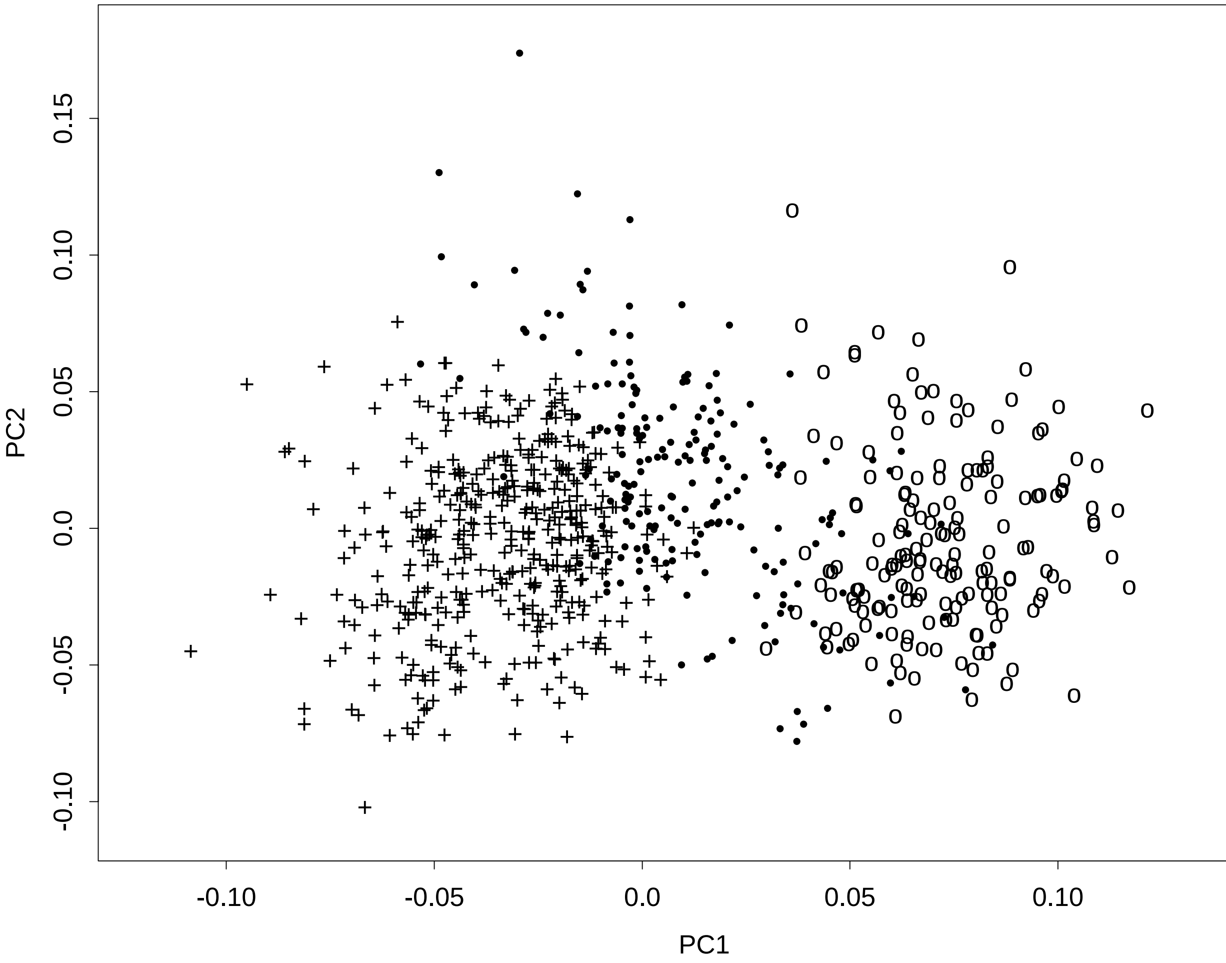


Figure 7.2: Three-cluster results on unconstrained model clustering, using variables 1, 3 and 4, in principal component space.

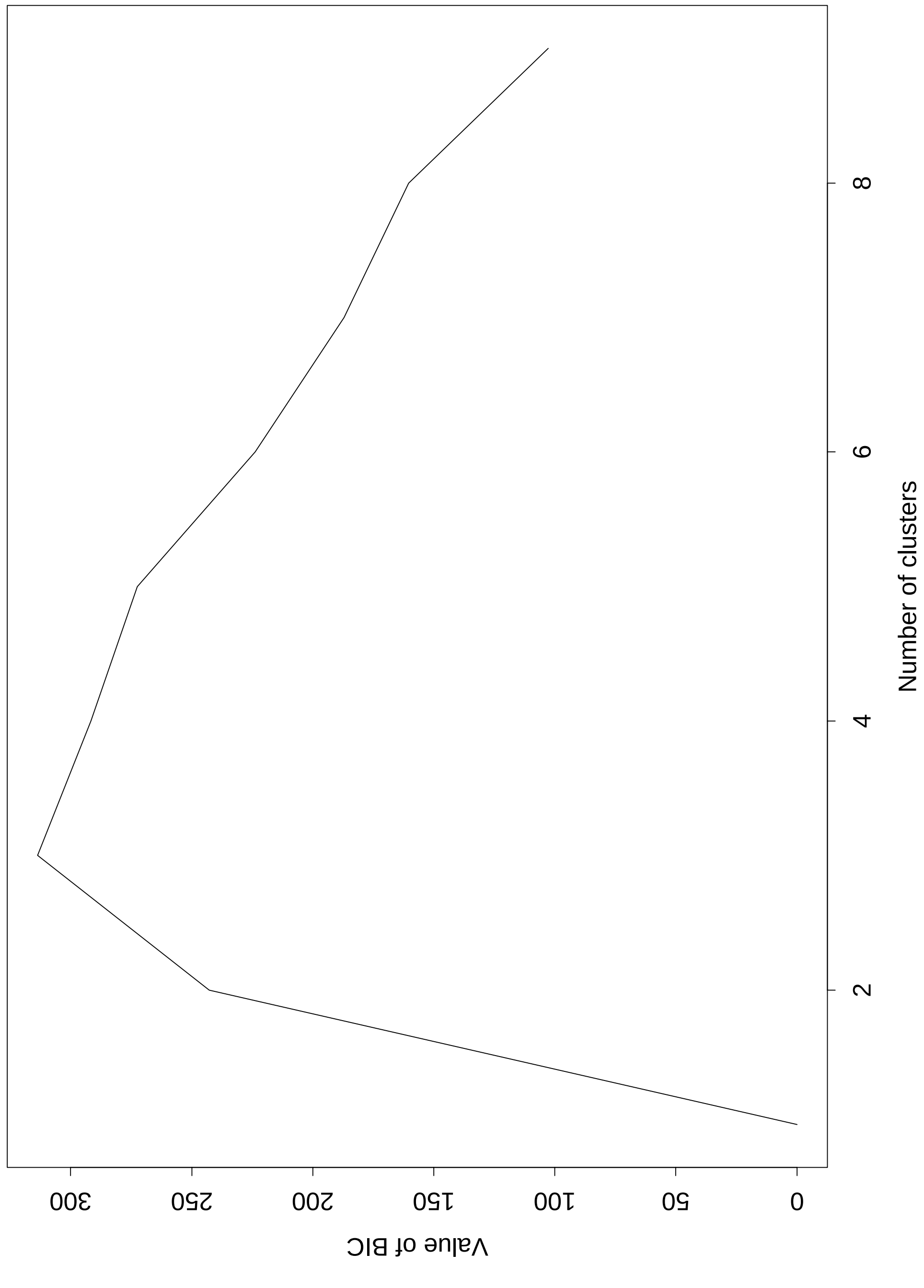


Figure 7.3: Bayesian Information Criterion value *vs*. assumed number of clusters for the model-based Gaussian mixture maximum likelihood clustering analysis ?? Figure may be redrawn with revised values for $g = 1, ..., 9$. ??

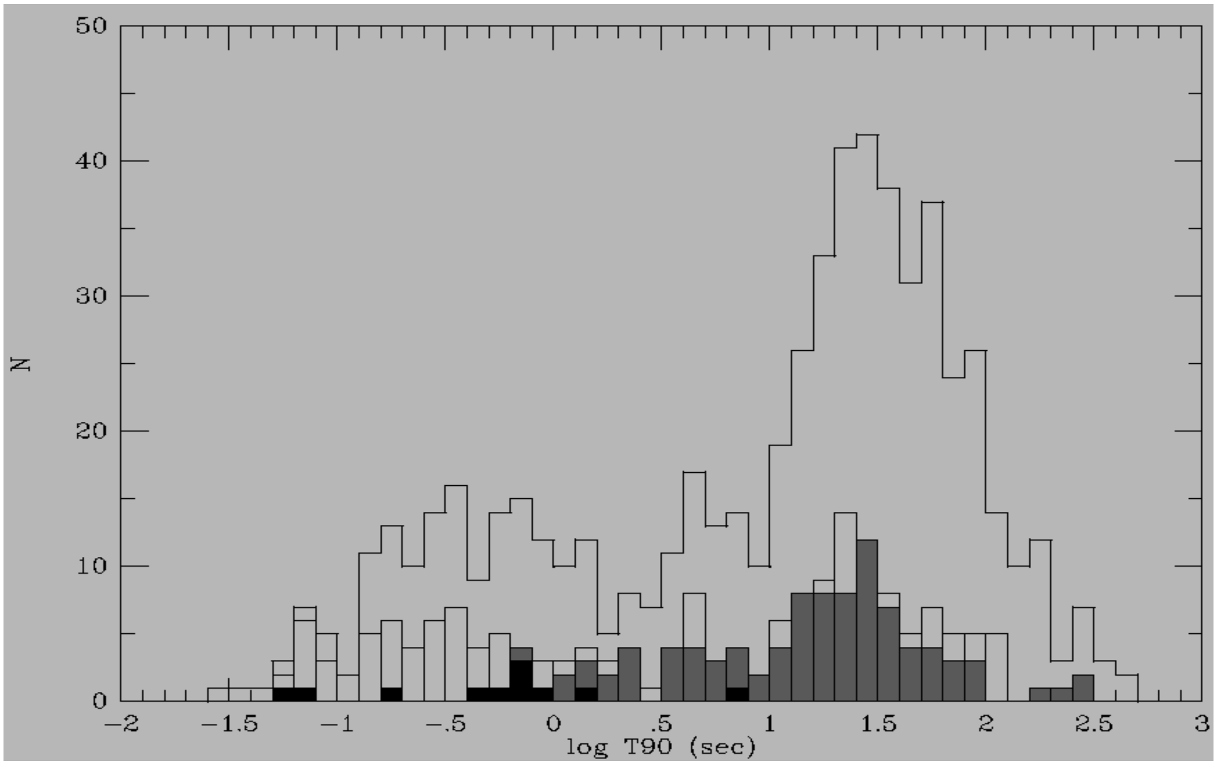


Figure 7.4: Burst types and T_{90} burst duration based on 5-dimensional Average Linkage clustering procedure. Class 1 = white, Class 2 = black, Class 3 = grey. The Class 4 burst is shown as an ×.

Chapter 8

Neural Networks

8.1 The Problem

The multilayer perceptron (MLP) is an example of a *supervised* method, in that it a training set of samples or items of known properties is used. The Kohonen self-organizing feature map is an example of an *unsupervised* method. In general terms, discriminant analysis seen in Chapter 4 is a family of supervised method, whereas cluster analysis explored in Chapter 3 is a family of unsupervised methods.

In dealing with the MLP, the single perceptron is first described, and subsequently the networking of perceptrons in a set of interconnected, multiple layers to form the MLP. The influential generalized delta rule, used in training the network, is introduced via the simpler case of the delta rule. We then look at what design considerations one should consider, for the design of one's network for a specific problem.

The self-organizing feature map method is then described.

A number of examples are given of both MLP and Kohonen map.

8.2 Mathematical Description

8.2.1 Perceptron Learning

Improvement of classification decision boundaries, or assessment of assignment of new cases, have both been implemented using neural networks since the 1950s. The perceptron algorithm is due to Rosenblatt ni the late 1950s. The perceptron, a simple computing engine which has been dubbed a "linear machine" for reasons which will become clear below, is best related to supervised classification. The idea of the perceptron has been influential, and generalization in the form of multilayer networks will be looked at later.

Let $\mathbf{x}$ be an input vector of binary values; o an output scalar; and bf w a vector of weights (or learning coefficients; initially containing arbitrary values). The perceptron calculates $o = \sum_j w_j x_j$. Let θ be some threshold.

If $o \geq \theta$, when we would have wished $o < \theta$ for the given input, then i is incorrectly categorized. We therefore seek to modify the weights and the threshold.

Set $\theta \longleftarrow \theta + 1$ to make it less likely that wrong categorization will take place again.

If $x_j = 0$ then no change is made to w_j. If $x_j = 1$ then $w_j \longleftarrow w_j - 1$ to lessen the influence of this weight.

If the output was found to be less than the threshold, when it should have been greater for the given input, then the reciprocal updating schedule is implemented. The updates to weights and thresholds may be denoted as follows::

$$o = \sum_j w_j x_j$$

$$\Delta\theta = -(t_p - o_p) = -\delta_p$$

(which the change in threshold for patter p)

$$\Delta w_i = (t_p - o_p)x_{pi} = \delta_p x_{pi}$$

(change in weights for pattern p).

If a set of weights exist, then the perceptron algorithm will find them. A counter-example is the exclusive-or, XOR, problem, represented in the following table:

Input vector	Desired output
0, 0	0
0, 1	1
1, 0	1
1, 1	0

Here it can be quickly verified that no way exists to choose values of w_1 and w_2 to allow discrimination between the first and fourth vectors (on the one hand) and the second and third (on the other hand).

Consider the following sets of four input vectors which we wish to automatically categorize in accordance with the given truth table values. Truth tables, per se, are not at issue here. These sets of vectors, and class assignments, will serve purely as examples. A condition for the applicability of the perceptron learning algorithm will become clear, on consideration of graphical representations of these problems. Draw the 2-valued points in the plane. Note the regions spanned by the F-related points versus the T-related points in each case.

```
  AND           OR            XOR
0  0  F       0  0  F       0  0  F
0  1  F       0  1  T       0  1  T
1  0  F       1  0  T       1  0  T
1  1  T       1  1  T       1  1  F
```

The line (hyperplane) separating T from F is defined by $< sum_j w_j x_j = \theta$. In the XOR case, linear separability does not hold. Perceptron learning fails for non-separable problems.

A multilayer perceptron which solves the XOR problem is shown in Fig. 8.1 Values of weights and thresholds which permit this are shown.

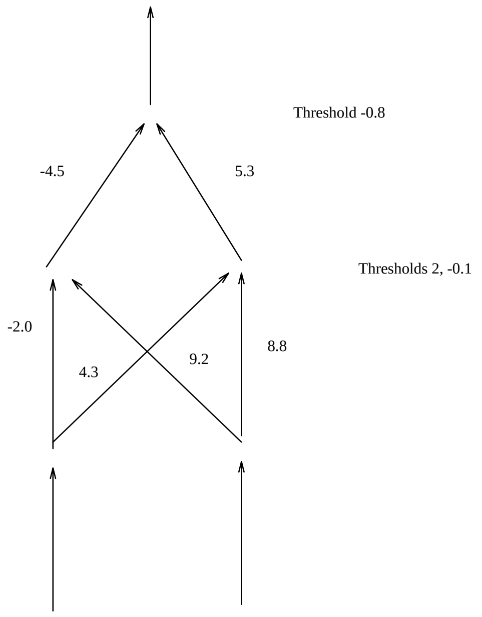


Figure 8.1: MLP solving the XOR problem, which cannot be solved with a single perceptron.

Network designs which are more sophisticated than the simple perceptron can be used to solve this problem. The network shown in Fig. 7.1 is a feedforward 3-layer network. This type of network is the most widely used. It has directed links from all nodes at a level to all nodes at the next level. Such a network architecture can be related to a linear algebra formulation (see remarks at the end of the next section). It has been found to provide a reasonably straightforward architecture which can also carry out learning, or supervised classification, tasks quite well. For further discussion of the perceptron, Gallant (1990) can be referred to for various interesting extensions and alternatives. Other architectures are clearly possible: directed links between all pairs of nodes in the network, as in the Hopfield model; directed links from later levels back to earlier levels as in recurrent networks; and so on.

The Delta Rule for Weight Updating

A generalization of a perceptron is a set of layers of perceptron, with connections between all neurons in one layer and all neurons in the following layer. This gives rise to what is termed a feedforward multilayer perceptron. See Fig. 7.2.

Initially we consider linear units only, i.e.

$$o_{pj} = \sum_i w_{ij} x_{pi}$$

The input at the ith neuron, x_{pi}, is occasioned by the pth pattern. The weight connecting the ith neuron in a given layer, to the jth neuron in the subsequent layer, is denoted w_{ij}. Consider an error term which we seek to minimize at each output neuron j; and let p by be the pattern which is currently

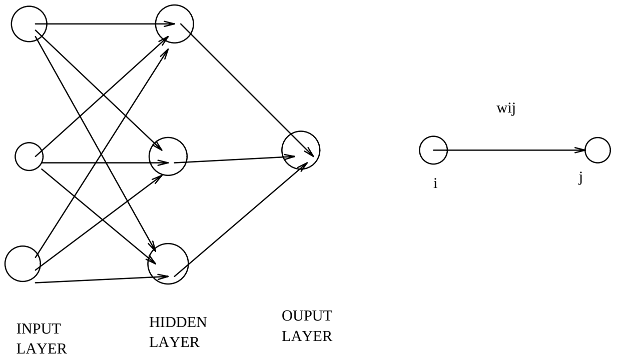


Figure 8.2: The MLP showing a 3-layer architecture, and one arc.

being presented to the net. Then

$$E_p = \frac{1}{2} \sum_j (t_{pj} - o_{pj})^2$$

where o is the output obtained when using the current set of weights. The multiplicative constant of a half is purely conventional, to make this error term look like an energy expression. The target output, t, is what we wish the network to replicate on being presented with the pth pattern. Consider

$$E = \sum_p E_p$$

We may write the expression for E_p as

$$\frac{1}{2} \sum_j \delta_{pj}^2$$

The rate of change of E_p with respect to w_{ij} is given by the chain rule:

$$\frac{\partial E_p}{\partial w_{ij}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial w_{ij}}$$

Now, since $E_p = \frac{1}{2} \sum_j (t_{pj} - o_{pj})^2$, the first term here is $-(t_{pj} - o_{pj})$. Given that linear units are being considered, i.e. $o_{pj} = \sum_i w_{ij} x_{pi}$, the second term equals x_{pi}. The gradient of E_p is thus

$$\frac{\partial E_p}{\partial w_{ij}} = -\delta_{pj} x_{pi}$$

If $\partial E / \partial w_{ij} = \sum_p \partial E_p / \partial w_{ij}$ and if updates do not take place after every training pattern, p, then we may consider the updating of weights as a classical gradient descent minimization of E. Each weight is updated according to

$$\Delta_p w_{ij} = \eta (t_{pj} - o_{pj}) x_{pi}$$

where η is a small constant. This corresponds to steepest descent optimization: we vary the weights in accordance with the downwards slope.

Although the algorithm presented for determining optimal weight values through training is valid, it is not difficult to see that we can view what has been done in linear algebra terms (see Appendix). Consider the matrix W_1 as defining the weights between all input layer neurons, i, and all hidden layer neurons, j. Next consider the matrix W_2 as defining the weights between all hidden layer neurons, j, and all output layer neurons, k. For simplicity, we consider the case of the three-layer network but results are immediately applicable to a network with more layers. Given a vector of inputs, x_p, the values at the hidden layer are given by $x_p W_1$. The values at the output layer are then $x_p W_1 W_2$. Note that we can "collapse" our network to one layer by seeking the weights matrix $W = W_1 W_2$. If t_p is the target vector, then we are seeking a solution of the equation $x_p W = t_p$. This is linear regression. The backpropagation algorithm described above would not be interesting for such a classical problem. Backpropagation assumes greater relevance when nonlinear transfer functions are used at neurons.

Generalized Delta Rule for Nonlinear Units

Nonlinear transformations are less tractable mathematically but may offer more sensitive modeling of real data. They provide a more faithful modeling of electronic gates or biological neurons. Now we will amend the delta rule introduced in the preceding section to take into consideration the case of a nonlinear transfer function at each neuron.

Consider the accumulation of weighted values of a neuron

$$\mathrm{net}_{pj} = \sum_i w_{ij} o_{pi}$$

where $o_i = x_i$ if unit i is an input one. This is passed through a differentiable and nondecreasing transformation, f,

$$o_{pj} = f_j(\mathrm{net}_{pj})$$

Normally this transfer function is a sigmoidal one. If it were a step function, this would violate our requirement for a differentiable function. We will see shortly that we will want to take the derivative of f, in other words find the slope which will be further use in making iterative improvements. A sigmoidal function is an elongated "S" function, which is not allowed to buckle backwards. One possibility is the function $y = (1+e^{-x})^{-1}$. Another choice is the hyperbolic tangent or tanh function: for $x > 20.0, y = +1.0$; for $y < -20.0, y = -1.0$; otherwise $y = (e^x - e^{-x})/(e^x + e^{-x})$. Both of these functions are invertible and continuously differentiable. Both have semilinear zones which allow good (linear) fidelity to input data. Both can make "soft" or fuzzy decisions. Finally, they are similar to the response curve of a biological neuron.

As before, the change in weights will be defined to be proportional to the energy (or error function) slope, and the chain rule yeilds:

$$\Delta_p w_{ij} \propto -\frac{\partial E_p}{\partial w_{ij}} = -\frac{\partial E_p}{\partial \mathrm{net}_{pj}} \frac{\partial \mathrm{net}_{pj}}{\partial w_{ij}}$$

From the definition of net_{pj}, the last term is o_{pi}. Let $\delta_{pj} = -\partial E_p / \partial \mathrm{net}_{pj}$. Hence

$$-\frac{\partial E_p}{\partial w_{ij}} = \delta_{pj} o_{pj}$$

or

$$\Delta_p w_{ij} = \eta \delta_{pj} o_{pi}$$

where η is the learning constant, usually a small fraction which prevents rebounding from side to side in ravines of the energy surface.

Note that for a linear output unit, by definition $o_{pj} = \mathrm{net}_{pj}$ and so we have $-\partial E_p / \partial o_{pj} = \delta_{pj}$ as was seen above when considering such units.

It must now be determined how to define δ_{pj}. We have

$$\delta_{pj} = -\frac{\partial E_p}{\partial \mathrm{net}_{pj}} = -\frac{\partial E_p}{\partial \mathrm{o}_{pj}} \frac{\partial o_{pj}}{\partial \mathrm{net}_{pj}}$$

and the last term is equal to $f_j'(\mathrm{net}_p j)$, i.e. the derivative of the transfer function. Two cases will be distinguished depending on whether the unit is an output one or a hidden layer one.

Case 1: Unit j is an output one, and it is found that

$$\delta_{pj} = (t_{pj} - o_{pj}) f_j'(\mathrm{net}_{pj})$$

Case 2: Unit j is a hidden layer one and it is found that

$$\delta_{pj} = f_j'(\mathrm{net}_{pj}) \sum_k \delta_{pk} w_{kj}$$

Hence the deltas at an internal node can be derived from the deltas at a subsequent (closer to output) node.

The overall algorithm is as follows: present pattern; feed forward, through successive layers; backpropagate – updating weight; repeat.

An alternative is to determine the changes in weights as a result of presenting all patterns: $\Delta w_{ij} = \sum_p \Delta_p w_{ij}$. This so-called "off-line" updating of weights is computationally more efficient but loses on the adaptivity of the overall approach.

Some further notes on the multilayer perceptron using the generalized delta rule follow.

A local optimum set of weights may be arrived at. There is no guarantee of arriving at a global optimum in weight space.

For the logistic activation function defined by

$$f_j(\mathrm{net}_{pj}) = 1/(1 + \exp^{-\mathrm{net}_{pj}})$$

where

$$\mathrm{net}_{pj} = \sum_i w_{ij} o_{pi} + \theta_j$$

and the last term is a bias (or threshold, we have the following result:

$$f_j'(\mathrm{net}_{pj}) = o_{pj}(1 - o_{pj})$$

For this function:

$$\delta_{pj} = (t_{pj} - o_{pj}) o_{pj} (1 - o_{pj})$$

for an output unit,

$$\delta_{pj} = o_{pj} (1 - o_{pj}) \sum_k \delta_{pk} w_{kj}$$

for a hidden layer unit.

In practice one must use approximations to hard-limiting values of 0, 1; e.g. 0.1, 0.9 can be used. Otherwise, infinite weight values would be needed in the above expression for $f_j(\text{net}_{pj})$.

It is found that symmetry breaking is necessary in order to get the backpropagation algorithm started. This involves randomizing the initial arbitrary weight values.

The presence of an additional momentum term, $\Delta w_{ij}^{(n+1)} = \eta \delta_{pj} o_{pi} + \alpha \Delta w_{ij}^{(n)}$ often helps the convergence properties of the steepest descent iterative optimization. This term takes the effect of previous steps into account. If the change in the previous step was large, the momentum tends to continue to keep the delta large.

The learning rate, η, should be small (0.7 or smaller). A larger learning rate implies quicker learning, but possible oscillations. The additional momentum term should have momentum rate α close to 1 (e.g. 0.9). On the one hand these parameters should be set such that convergence is attained; on the other hand they must allow some change to take place. It may be desired to allow α to be small initially, and to increase it in the course of convergence. In sum, the most appropriate choice of these parameters depends on the training data and the network architecture.

The MLP architecture using the generalized delta rule can be very slow for the following reasons. The compounding effect of sigmoids at successive levels can lead to a very flat energy surface. Hence movement towards fixed points is not easy. A second reason for slowness when using the generalized delta rule is that weights tend to try together to attain the same value. This is unhelpful – it would be better for weights to prioritize themselves when changing value. An approach known as cascade correlation, due to Scott Fahlman, addresses this by introducing weights one at a time.

The number of hidden layers in an MLP and the number of nodes in each layer can vary for a given problem. In general, more nodes offer greater sensitivity to the problem being solved, but also the risk of overfitting. The network architecture which should be used in any particular problem cannot be specified in advance. A 3-layer network with the hidden layer containing, say, less than $2m$ neurons, where m is the number of values in the input pattern, can be suggested.

Other optimization approaches and a range of examples are considered in Murtagh (1990/1991). Another short, comparative overview can be seen in Murtagh (1994).

8.2.2 The Kohonen Self-Organizing Feature Map

The Kohonen self-organizing feature map (SOFM) with a regular grid output representational or display space, involves determining vectors w_k, such that inputs x_i are parsimoniously summarized (clustering objective); and in addition the vectors w_k are positioned in representational space so that similar vectors are close (low-dimensional projection objective) in *representation space*: $k, k', k^{''} \in \{(r,s) \mid r = 1, \ldots, R; s = 1 \ldots, S\}$.

Clustering: Associate each x_i with some one w_k such that $k = argmin \parallel x_i - w_k \parallel$

Low-Dimensional projection: $\parallel w_k - w'_k \parallel < \parallel w_k - w^{''}_k \parallel \Longrightarrow \parallel k - k' \parallel \leq \parallel k - k^{''} \parallel$

By way of example, $R = S = 10$ and the output representation grid is a regular, square one. The metric chosen for norm $\parallel . \parallel$ is usually Euclidean, and this will be assumed here. Without loss of generality, we can consider the squared Euclidean distance whenever this is useful to us. Evidently $x_i \in \mathbb{R}^m$ and $w_k \in \mathbb{R}^m$ for some dimensionality, or cardinality of attribute-set, m.

Iterative algorithms for clustering are widely used, requiring an initial random choice of values for w_k to be updated on the basis of presentation of input vectors, x_i. At each such update, the low-dimensional projection objective is catered for by updating not just the so-called winner w_k, but also neighbors of w_k with respect to the representational space. The neighborhood is initially chosen to be quite large (e.g. a 4×4 zone) and as the epochs proceed, is reduced to 1×1 (i.e. no neighborhood). An epoch is the term used for a complete set of presentations, and consequent updates, of the N input vectors. The result obtained by the SOFM algorithm is sub-optimal, as also is the case usually for clustering algorithms of this sort (k-means, partitioning) and quite often for dimensionality-reduction methods (Sammon's mapping). A range of studies showing how well the SOFM method performs compared to these methods can be found in Murtagh and Hernández-Pajares (1995).

An example follows. We used a set of 45 astronomical spectra. These were of the complex AGN (active galactic nucleus) object, NGC 4151, and were taken with the small but very successful IUE (International Ultraviolet Explorer) satellite which is still active in 1996 after nearly two decades of operation. We chose a set of 45 spectra observed with the SWP spectral camera, with wavelengths from 1191.2 Å to approximately 1794.4 Å, with values at 512 interval steps. There were some minor discrepancies in the wavelength values, which we discounted: an alternative would have been to interpolate flux values (vertical axis, y) in order to have values at identical wavelength values (horizontal axis, x), but we did not do this since the infrequent discrepancies were fractional parts of the most common regular interval widths. Fig. 7.3 shows a sample of 20 of these spectra. Fig. 7.4 shows a Kohonen map determined from a set of 45 such spectra. Some nodes have a number of spectra associated with them, and this is indicated. Some nodes have no associated spectrum and these are indicated also.

8.3 Examples and Bibliography

8.3.1 Discrimination of Cosmic Ray Hits

Space-borne observatories are more likely to suffer from cosmic ray hits since the Earth's atmosphere cannot fully shield the detectors. The original Wide

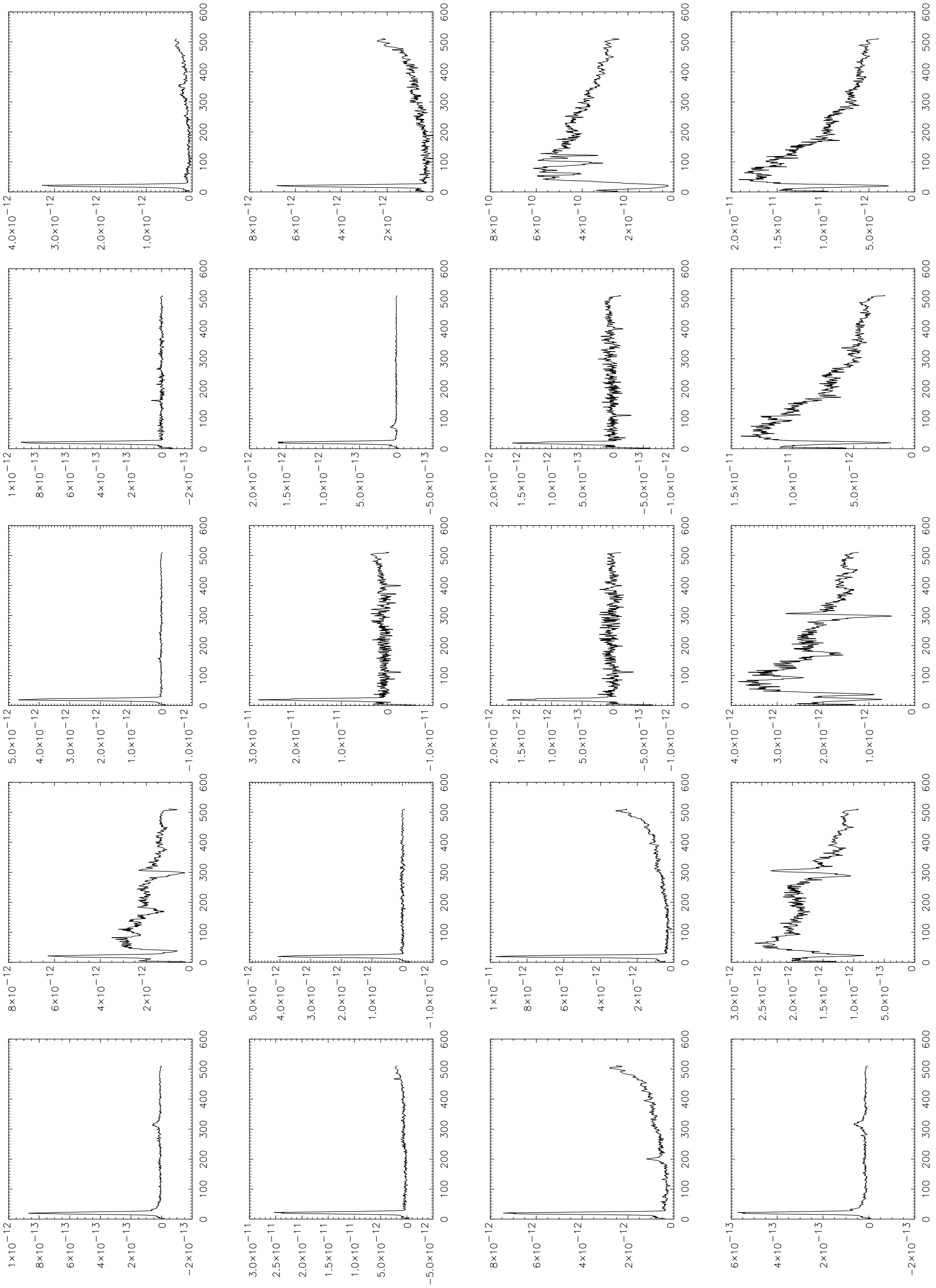


Figure 8.3: Sample of 20 spectra (from 45 used) with original flux measurements plotted on the y-axis.

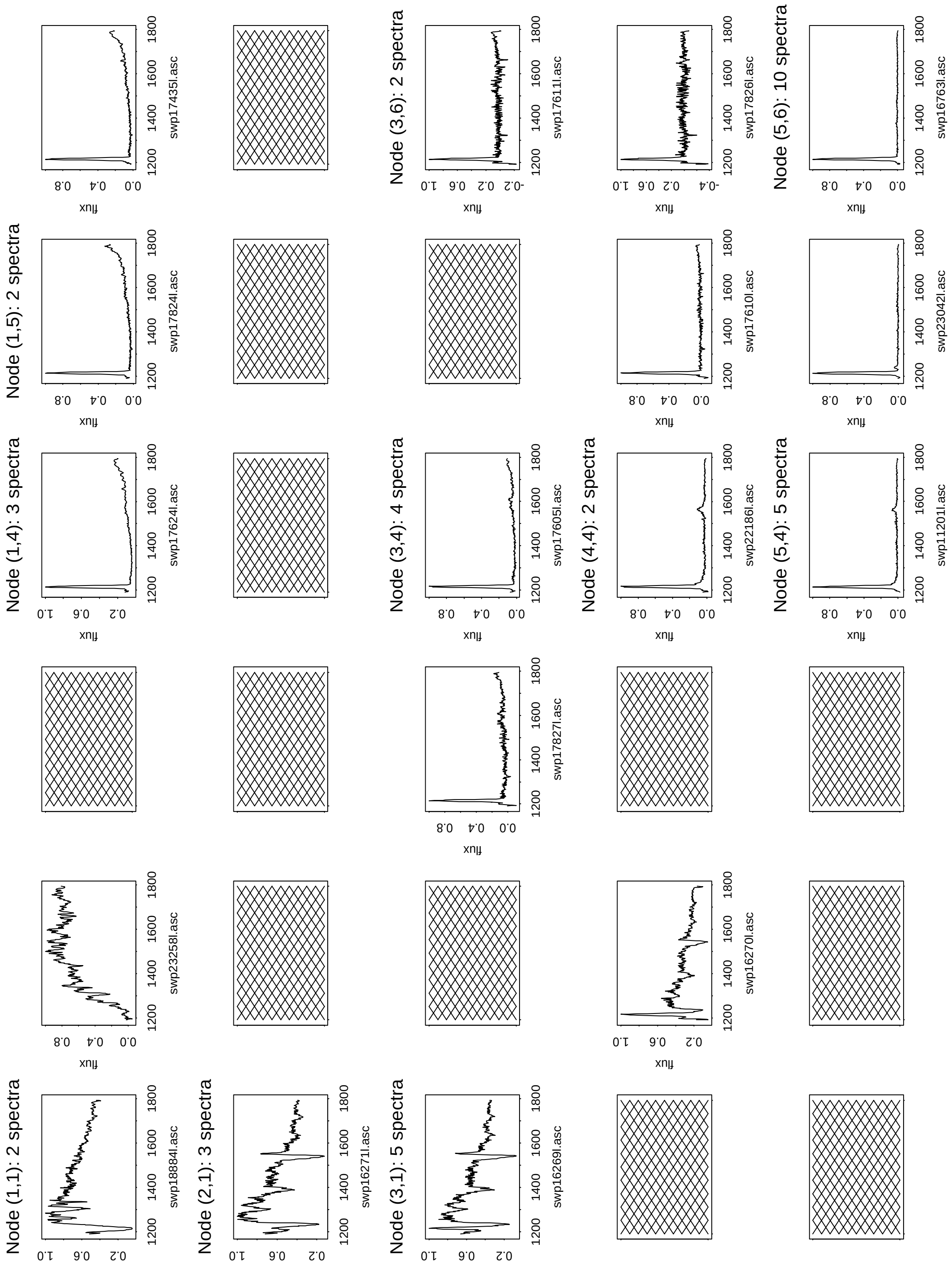


Figure 8.4: Kohonen self-organizing map of 45 spectra.

Field and Planetary Camera (WF/PC) detector of the Hubble Space Telescope consisted of eight Texas Instruments CCD chips, four for each of the wide-field camera and the planetary camera. Each CCD provided 800 × 800 pixel frames, allowing a 1600 × 1600 image to be constructed for the WFC or for the PC. Later versions of this detector differed somewhat in configuration. It was estimated that in a 45-minute WF/PC exposure, 10 to 25% of all pixels would be hit by a cosmic ray. At certain locations (in particular the South Atlantic Anomaly region off the coast of Argentina) the effect could be much larger. The extensive contamination due to such cosmic ray hits made it imperative to remove them during data reduction. The best way to do this was to take multiple perfectly aligned images, – "split mode" or repeated observations. The overhead of observing time that this requires motivates the work to be described here, which was based on a trainable classifier for object recognition from single frames (Murtagh and Adorf, 1991; Murtagh, 1992).

For our purposes, cosmic rays can be divided into two classes, – high energy primary, and low energy secondary particles. High energy cosmic rays pass straight through the detector. On average in the HST WF/PC case, they caused electrons to be deposited in two to three pixels.

A single event caused 80 to 100 electrons to be deposited per dimensions of $15 \times 15\ times 8 \mu\mathrm{m}^3$ in length, breadth, and thickness. One of the PC chips was noticeably less thick. Depending on the traversal angle, between 8 and 20 times (say) 80 electrons was deposited in a pixel by high energy cosmic ray hits. Diffusion along the path traversed by the cosmic ray also takes place, but was sufficiently small to be ignored. A conversion factor of 7.5 brough the electron counts to ADU (astronomical data units) used in the image data.

Secondary particles were produced by cosmic rays hitting the surrounding spacecraft components. In particular the detector housing and "shielding" gave rise to these low energy secondaries and the number of electrons deposited by them could be very high.

This is a typical pattern recognition problem and the question becomes how to set up a system. Two approaches seem feasible: freature- (or astronomical object-) based; and image- (or pixel-) based. For a feature-based classifier one would have to define a good set of features describing the image context in a region of interest surrounding the pixel in question. A trainable classifier would then be trained on these features together with the correct answers "CR" and "non-CR" to be provided externally. An image-based classifier works directly on the pixel data in the region of interest. Working directly on the data (the latter case) off-loads the construction of a good feature set from us to the classifier. Note though that it doesn't obviate it. Simply, the pixel values are used as features. This is the approach which we will describe here.

The method employed was to: (i) have cosmic ray hits on images flagged by a human expert, providing "ground truth"; (ii) use this information to train and test various classifiers; and (iii) apply the classifiers to other images.

Calibrated WFC images of the open cluster NGC 1850, with 10 seconds exposure time, and without any preprocessing, were used in the experiments. A colleagues (Jon Holtzman) marked the same 200 × 200 image on two different occasions, separated by serveral months. It transpired that the first time 85 cosmic ray pixels were noted, and 102 the second time, with 74 common to both. If the first marking is taken as ground truth, this corresponds to a redetection rate of 87%, whereas if the second marking is regarded as ground truth then

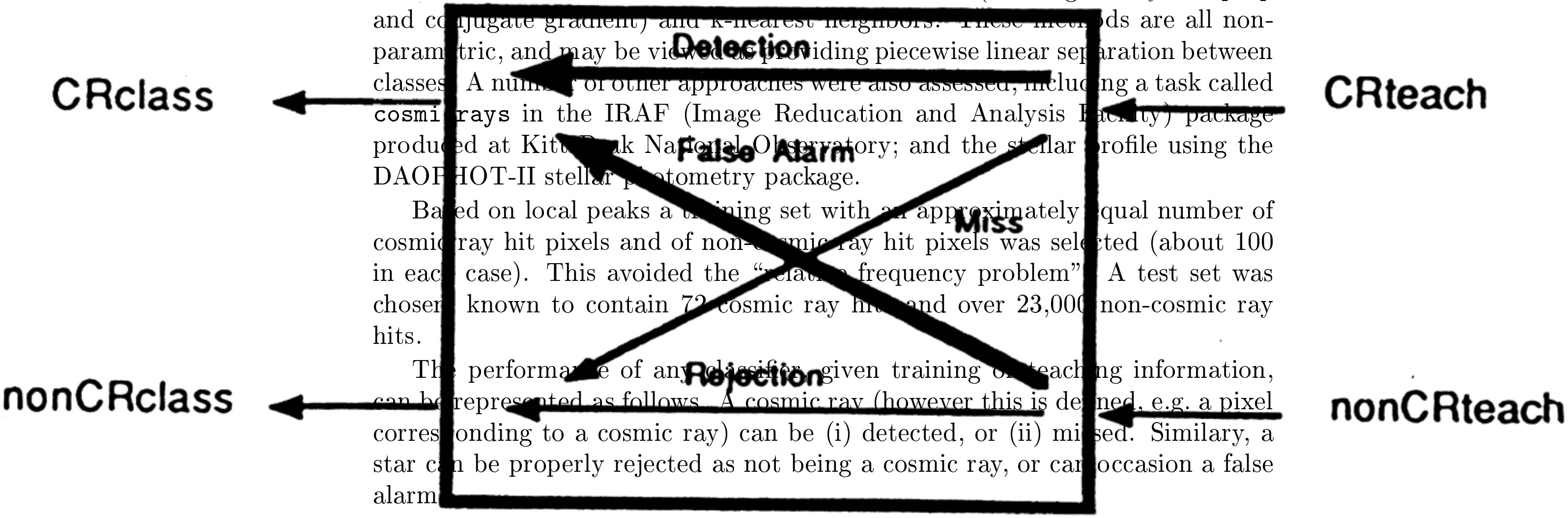


Figure 8.5: Performance of the classifier.

the redection rate drops to 73%.

We used two variants of a neural network method (training was by backprop and conjugate gradient) and k-nearest neighbors. These methods are all non-parametric, and may be viewed as providing piecewise linear separation between classes. A number of other approaches were also assessed, including a task called `cosmic rays` in the IRAF (Image Reducation and Analysis Facility) package produced at Kitt Peak National Observatory; and the stellar profile using the DAOPHOT-II stellar photometry package.

Based on local peaks a training set with an approximately equal number of cosmic ray hit pixels and of non-cosmic ray hit pixels was selected (about 100 in each case). This avoided the "relative frequency problem". A test set was chosen, known to contain 72 cosmic ray hits and over 23,000 non-cosmic ray hits.

The performance of any classifier, given training or teaching information, can be represented as follows. A cosmic ray (however this is defined, e.g. a pixel corresponding to a cosmic ray) can be (i) detected, or (ii) missed. Similary, a star can be properly rejected as not being a cosmic ray, or can occasion a false alarm.

The rate of detection, RD, is an estimate of the probability of detection, P(CR$_{class}$ | CR$_{teach}$). We wish to maximize RD. The rate of false alarms, RFA, is an estimate of the false alarm probability, P(CR$_{class}$ | non-CR$_{teach}$. We wish to minize RFA. The OC or operating characteristic diagram (also called ROC, receiver operating characteristic, diagram) is the plot of RD, RFA couples. Each point defines an operating point of a classifier (Melsa and Cohn, 1978). Within the basic cycle of train-test-apply, the results of training and testing a statistical classifier are represented conventionally using an operating characteristic diagram. Figure 7.5 shows the schema used for an OC diagram.

Figure 7.6 presents operating characteristics found for various methods. It reflects the trade-off inherent in maximizing RD while minimizing RFA. The OC diagram shown here could be used to implement a cosmic ray detection strategy. For instance, the Neyman-Pearson strategy would hold the RFA value fixed, at some upper acceptable limit, and find the best RD value corresponding to this; or, given that the priors convey much information, one could use a maximum a

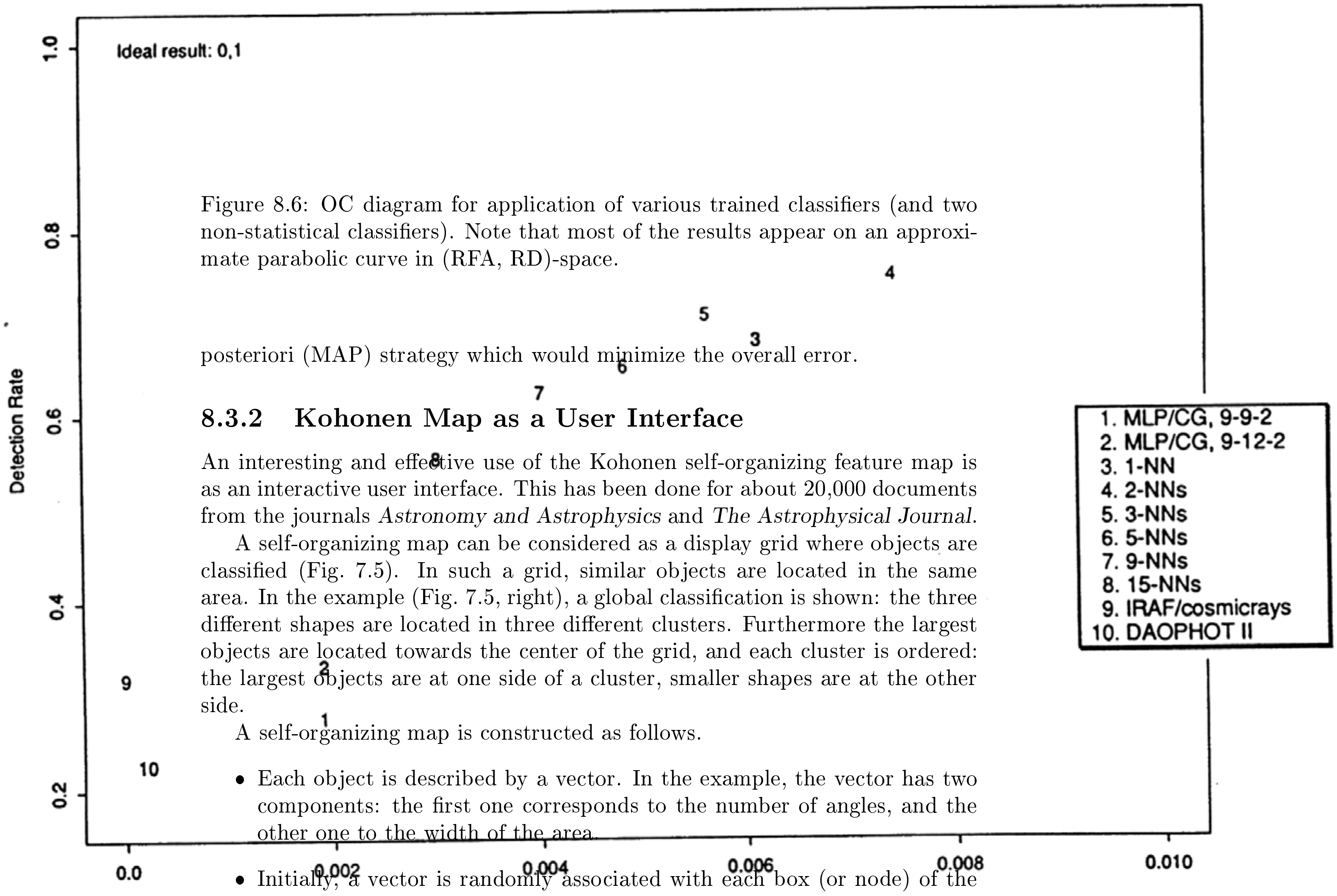


Figure 8.6: OC diagram for application of various trained classifiers (and two non-statistical classifiers). Note that most of the results appear on an approximate parabolic curve in (RFA, RD)-space.

posteriori (MAP) strategy which would minimize the overall error.

8.3.2 Kohonen Map as a User Interface

An interesting and effective use of the Kohonen self-organizing feature map is as an interactive user interface. This has been done for about 20,000 documents from the journals *Astronomy and Astrophysics* and *The Astrophysical Journal*.

A self-organizing map can be considered as a display grid where objects are classified (Fig. 7.5). In such a grid, similar objects are located in the same area. In the example (Fig. 7.5, right), a global classification is shown: the three different shapes are located in three different clusters. Furthermore the largest objects are located towards the center of the grid, and each cluster is ordered: the largest objects are at one side of a cluster, smaller shapes are at the other side.

A self-organizing map is constructed as follows.

- Each object is described by a vector. In the example, the vector has two components: the first one corresponds to the number of angles, and the other one to the width of the area.
- Initially, a vector is randomly associated with each box (or node) of the grid.
- Each document is located in a box whose descriptive vector is the most similar to the object's vector.
- During an iterative learning process, the components of the nodes describing vectors are modified. The learning process produces the classification.

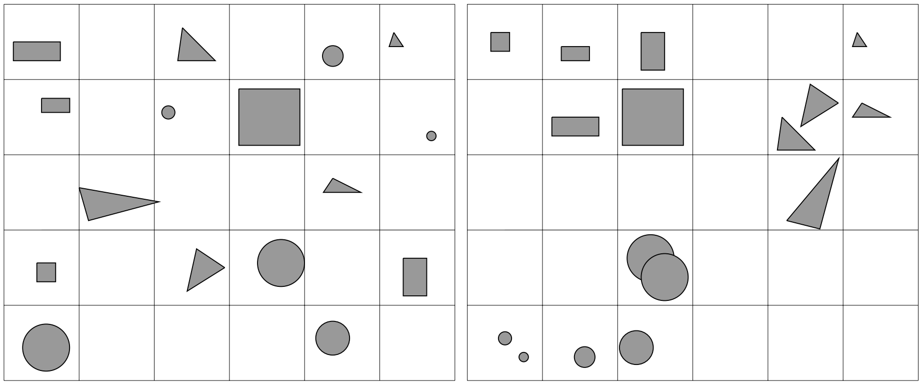


Figure 8.7: Object locations. Left: before learning. Right: after learning.

Typical use of the self-organizing map in order to classify bibliographical data was as follows. Our set of documents relates to the journal *Astronomy and Astrophysics* in the time-interval 1994 onwards. The descriptive vector is based on the journal keywords associated with each document. We eliminated rare keywords (found in less than 5 documents). Finally, we used about 6500 documents described by 269 keywords.

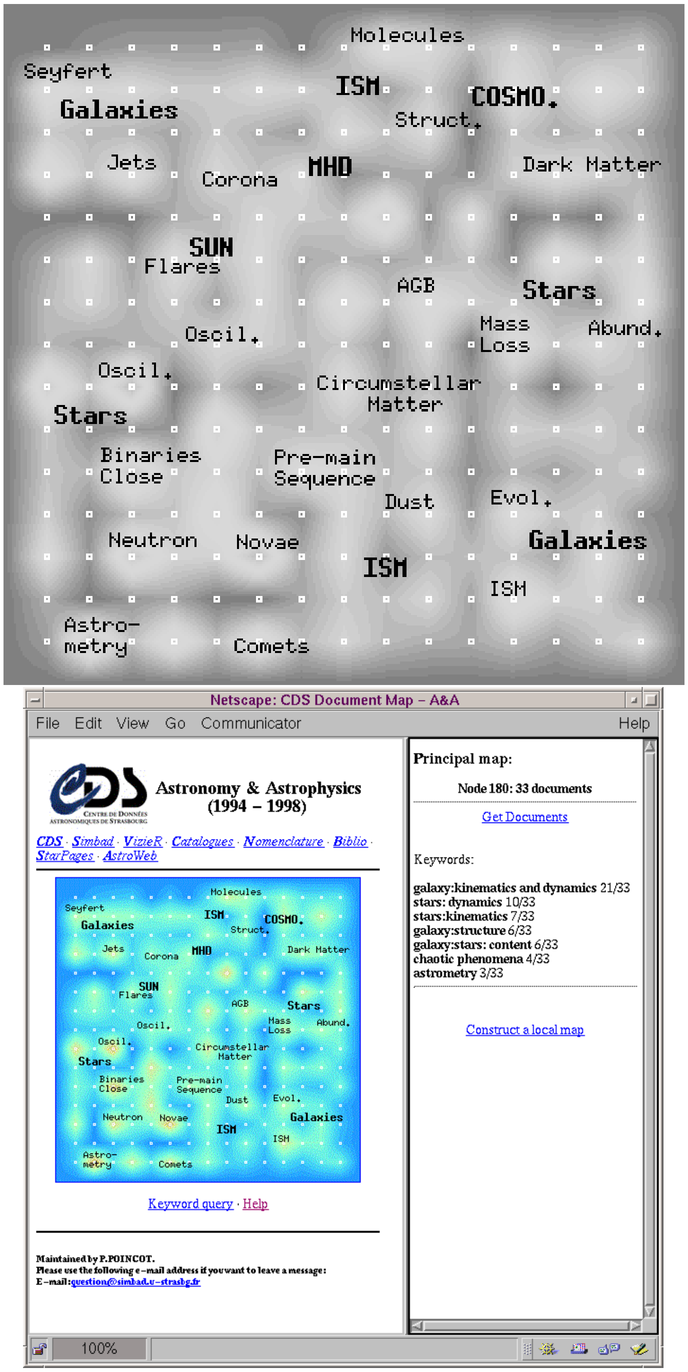
Adaptation of the map included "wrap-around" so that neighbors at one extremity would meet with another extremity of the map (reminiscent of the map portraying a flattened sphere); and when there was a large number of documents associated with a node, a subsidiary map was determined, leading to a hierarchy of maps. A 15 × 15 grid was used for the principal map, and a 5 × 5 grid for the detailed maps.

Next we come to the graphical user interface. We display the trained self-organizing map as a density map, which represents graphically the areas containing papers of similar content and the number of documents in these areas. The map is labeled to locate on it the themes dealt with (Fig. 7.6, top).

The user can select one node of the map (by clicking on the image) to obtain information about the articles located at this node (the number of documents and the keywords describing them appear on the right side of the interface) (Fig. 7.6, bottom). The user can also access the detailed map, and/or the article content (title, authors, abstract) and all the facilities provided by the CDS bibliographical service (including a link to ADS and to the online full paper in many cases).

The user interface allows one to select and display on the map only a part of the database. This is used with keyword queries (only the documents containing selected keywords are shown), or with an external list of documents (bibcode queries: a "bibcode" is a 19-character standard for a document reference.

Fig. 7.7 gives an overall view of the system. It is based on imagemap and CGI scripts. Hence our implementation is server-side and imagemap and CGI scripts. Hence our implementation is server-side and compatible with any browser supporting imagemaps.



Netscape: CDS Document Map – A&A

Astronomy & Astrophysics
(1994 – 1998)

CDS · Simbad · VizieR · Catalogues · Nomenclature · Biblio · StarPages · AstroWeb

Keyword query · Help

Maintained by P.POINCOT.
Please use the following e-mail address if you want to leave a message:
E-mail:question@simbad.u-strasbg.fr

Principal map:

Node 180: 33 documents

Get Documents

Keywords:

galaxy:kinematics and dynamics 21/33
stars: dynamics 10/33
stars:kinematics 7/33
galaxy:structure 6/33
galaxy:stars: content 6/33
chaotic phenomena 4/33
astrometry 3/33

Construct a local map

Figure 8.8: Top: the principal map. Bottom: the user interface. URL http://simbad.u-strasbg.fr/A+A/map.pl

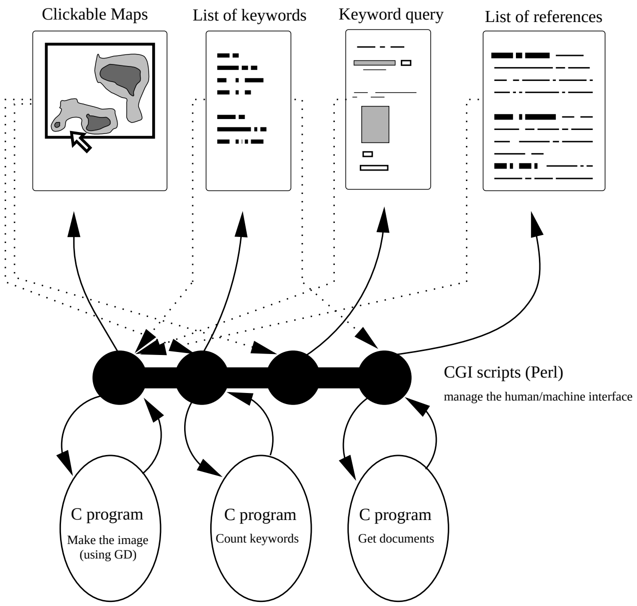


Figure 8.9: SOM-based system.

8.3.3 Bibliography

1. E.R. Davies, "Training sets and a priori probabilities with the nearest neighbour method of pattern recognition, *Pattern Recognition Letters*, **8**, 11–13, 1988.

2. S.I. Gallant, "Percepton-based learning algorithms", *IEEE Transactions on Neural Networks*, **1**, 179–191, 1990.

3. J.L. Melsa and D.L. Cohn, *Decision and Estimation Theory*, McGraw-Hill, New York, 1978.

4. F.Murtagh and H.-M. Adorf, "Detecting cosmic ray hits on HST WF/PC images", in P.J. Grosbol and R.H. Warmels, Eds., *3rd ESO/ST-ECF Data Analysis Workshop*, European Southern Observatory, Munich, 51–56, 1992.

5. F. Murtagh, "Cosmic ray discrimination on HST WF/PC images: object recognition-by-example", in D.M. Worrall, C. Biemesderfer and J. Barnes, Eds., *Astronomical Data Analysis Software and Systems I¿*, 265–273, 1992.

6. F. Murtagh, "Multilayer perceptrons for classification and regression", *Neurocomputing*, **2**, 183–197, 1990.

7. F. Murtagh, "Neural network and related massively parallel methods for statistics: a short overview", *International Statistical Review*, **62**, 275–288, 1994.

8. F. Murtagh and M. Hernández-Pajares, "The Kohonen self-organizing feature map method: an assessment", *Journal of Classification*, **12**, 165–190, 1995.

9. P. Poinçot,SOM maps, 1998. *Astronomy and Astrophysics* document map, http://simbad.u-strasbg.fr/A+A/map.pl. *Astrophysical Journal* document map, http://simbad.u-strasbg.fr/ApJ/map.pl. Catalog document map, accessible on page http://vizir.u-strasbg.fr/viz-bin/VizieR.

10. P. Poinçot, S. Lesteven and F. Murtagh, "A spatial user interface to the astronomical literature", *Astronomy and Astrophysics Supplement Series*, **130**, 183–191, 1998.

11. P. Poinçot, S. Lesteven and F. Murtagh, "Maps of information spaces: assessments from astronomy", *Journal of the American Society for Information Science*, submitted, 1999.

Chapter 9

Conclusion: Strategies for Analysing Data

In this chapter, we will summarize the options involved in applying methods studied in earlier chapters. Whether a method will perform well in a practical situation will depend both on the geometrical/mathematical nature of the method employed, and simultaneously on the quality and type of the data analysed. We will review the former first.

9.1 Objectives of Multivariate Methods

1. Principal Components Analysis:
   - Reducing the dimensionality of the parameter space to its inherent dimensionality.
   - Thereby also eliminating “noise”, “cleaning” the data, and lessening the volume of data.
   - Determining the most important linear combinations of the parameters (variables).
   - Using these to determine linear combinations, and even quadratic, logarithmic, and other, combinations.
   - Determining the important parameters present.
   - Determining “latent”, underlying variables present.
   - Visualizing the data by the selection of the most important planar views of it.
   - Identification of groups of objects or of variables.
   - Identification of outliers or anomalous objects/variables.
2. Cluster Analysis:
   - Determining groups of objects, using some criterion.
   - Without prior knowledge of what groups are present, obtaining a range of groupings and using this to assess the true group structure.

3. Discriminant Analysis:

   - Assessing the separability of known groups.
   - Assigning objects to one of a number of known groups.
   - Displaying the groups, optimally separating the known classes.

4. Correspondence Analysis:

   - Identifying simultaneously groups of objects and variables.
   - Determining "latent", underlying variables.
   - Determining the important variables present.
   - Identifying outliers or anomalous objects/variables.

5. Principal Coordinates Analysis:

   - Determining groups, and displaying.

6. Canonical Correlation Analysis:

   - Assessing relationships of two groups of variables relating to the same object population.

7. Regression Analysis:

   - Fitting of a straight line to a set of points and assessing the significance of anomalous points.

9.2 Types of Input Data

1. Principal Components Analysis implicitly uses Euclidean distances: real valued data should therefore be used.

2. Clustering uses either distances or dissimilarities. Either may be calculated from given data. Clustering routines may accept the latter as input, or may work on distance/dissimilarity input.

3. Most of the Discriminant Analysis methods looked at assume real valued data.

4. Correspondence Analysis allows a wide range of data types. It is suitable for data in the form of frequencies of occurence. It also is suitable for data in a form of coding which associates response categories with all possible data values (this latter is particularly interesting for diverse – qualitative/real and categorical/integer – types of data).

5. Principal Coordinates Analysis assumes distances as input. Other multidimensional scaling methods assume dissimilarities.

6. Canonical Correlation Analysis assumes real valued data.

7. Regression Analysis, as discussed in this book, assumes real valued data.

9.3 Strategies of Analysis

The complimentary use of multivariate methods often offer powerful ways of extracting information from the data. The following are some possibilities.

1. The use of clustering and of a display technique (Principal Components Analysis, Correspondence Analysis, etc.) can give complimentary views of the same data set, and can lead to more robust conclusions about the data. The latter, for example, look at groupings relative to projections onto axes, whilst the former looks at fully multidimensional grouping.

2. A Cluster Analysis can be carried out on the more informative first few axes of a Principal Components Analysis.

3. A Principal Components Analysis can be used on large clusters resulting from a Cluster Analysis in order to help interpret them.

4. Principal Components Analysis or Cluster Analysis may be used to define groups, and Discriminant Analysis may be used to subsequently assess them.

5. For complex data sets, a recoding can be carried out so that Correspondence Analysis can be used; or distances/dissimilarities can be defined so that clustering or multidimensional scaling can be employed.

6. Prior to analysis, the analyst should examine the correlation matrix in the case of Principal Components Analysis, or the Burt table in the case of Multiple Correspondence Analysis. All findings should be verified as far as possible with respect to the given, original data.

9.4 Online Software Resources

Having seen a range of analysis methods in theory and practice, the user may well wish to try to locate software to carry out some assessments quickly.

Major statistical packages not only contain an impressive range of methods, but – on many occasions – the user manuals contain an excellent succinct overview of the capabilities of different methods. We assume that such user manuals may be found in libraries, even if the software is not necessarily available on one's local system.

Commercial packages includes SAS (Statistical Analysis System), widely used in the context of business information systems. SPSS (Statistical Package for the Social Sciences) is a major package used in a variety of fields. S-Plus is a newer package, originating as the statistical language, S, in Bell Labs, and marketed with a wide range of additional features as S-Plus by MathSoft Inc. There is a wealth of software packages in the data mining area, often based on methods for divisely constructing classification and decision trees (see Michie et al., 199x, Hand, 1999). A package catering for cluster analysis is Clustan.

The most important general source of statistical programs (source code) on the web is Statlib, located at Carnegie Mellon University. It has a wealth of material including codes from *Applied Statistics*, newer Bayesian modeling and other programs, and a wide range of programs for use in the S-Plus environment.

An extensive resource list of programs and packages available for public access on the web, with description, is maintained by F. Murtagh and is available at the following address. This takes the form of statements made in newsgroup postings or email messages.

http://astro.u-strasbg.fr/~fmurtagh/mda-sw/online-sw.html

This list is currently about 25 pages in length.

Among newsgroups of importance are KDD-Nuggets, for data mining, CLASS-L for classification, the Connectionists list for neural networks, the Vision List for image processing and computer vision, among others.

9.5 Bibliography

1. D. Michie, D.J. Spiegelhalter and C.C. Taylor, *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, 199x.

2. Very recent book on statistical data analysis by D.J. Hand.

3. Very recent book on pattern recognition by A.R. Webb.

Appendix A

Essential Mathematics

A.1 Introduction

This Appendix firstly gathers together some basic definitions, symbols and terminology to do with random variables, random processes, and random fields; the topics are chosen according to their applicability to pattern recognition, signal processing, image processing and data compression. We present some fundamental theorems and definitions related to estimation, prediction, and general analysis of data that are generated by random processes.

Secondly, basic definitions of linear algebra are covered. This mathematical language, and tool-set, is invaluable for topics such as pattern recognition and neural networks, image segmentation, and many other areas besides.

A.2 Random Variables, Signals and Fields

We start by presenting relevant definitions and theorems, and progress to identifying the types of (theoretical) processes that will be appropriate models for our data. We identify properties of these processes that are relevant to pattern recognition, signal processing, image processing and data compression.

A.2.1 Basic Probability and Random Variables

Events and Probability

See Rosenfeld and Kak (1982), and Mortensen (1987). Let there be a set of *outcomes* to an experiment:

$$\{\omega_1, \omega_2, \omega_3, \ldots\} = \Omega$$

For each w_i there is a probability p_i:

$$p_i \geq 0 \tag{A.1}$$

$$\sum p_i = 1$$

The simple definition of probability over outcomes is satisfactory for simple applications, but for many applications we need to extend it to apply to subsets of Ω, called *events.*

Let there be subsets of Ω called events: a general event is a, the set of all a is A. We define a probability measure P on A; P is a number. P satisfies the following axioms:

$$P(a) \geq 0 \tag{A.2}$$

$$P(\Omega) = 1 \qquad \text{(certain event)}$$

$$a_i \cap a_j = \emptyset \quad \forall \;\; i, j$$

$$a_1, a_2, \ldots \text{ are } \textit{disjoint} \text{ members of } A \text{ and } \emptyset \text{ is the } \textit{empty set}$$

$$P(\cup_{k=1}^{\infty} a_k) = \sum_{k=1}^{\infty} P(a_k) \tag{A.3}$$

Put simply, if a_i and a_j are disjoint,

$$P(a_i \cup a_k) = P(a_i) + P(a_k)$$

A fourth axiom, which is really a corollary of these is sometimes included:

$$P(0) = 0 \qquad \text{(the impossible event)}$$

Some Comments on Events and the Probability Measure

This subsection discusses some limitations on events and probabilities which are theoretical and, in practice of little restriction. This subsection primarily introduces some further terminology related to the previous subsection, and may be skipped by the reader who prefers to continue with more central themes.

Some papers and texts, though of an applied nature, feel obliged to use the terminology of rigorous probability theory; the purpose of this note is to dispel some of the mystique of that terminology.

As in earlier and subsequent sections, Ω is the set of (elementary) outcomes; and A is the set of subsets of Ω to which probabilities can be allocated.

Theoretically, it is not possible to assign probabilities to *all* subsets of Ω, but only to a class of these subsets that form a type of *field* called a *Borel field.* This is of no practical consequence, but is an analytical neccessity arising (inter alia) from the impossibility of assigning probabilities in the case where there are infinitely many subsets of Ω, and still have these probabilities satisfy the axioms of probability (eqns. A.1, A.2 and A.3). In general, the powerset of Ω is infinite, and the Borel field A which defines the restricted collection of subsets which can be allocated a probability is not.

The term "probability *measure*" was used deliberately. Roughly speaking, measure is a method of associating a *number* with a subset. Measure is a general form of integration. For example, in many practical applications, the definition of probability involves integration over a domain, e.g. integral between x_1 and x_2 on the real line. The measurability of a function depends on its values, and its domain.

Example: A simple function, $f(x)$ that is 1 between 0 and 1, and 0 elsewhere: $f(x) = 1 : 0 \leq x \leq 1, f(x) = 0$: otherwise. Obviously,

$$\int_0^1 f(x)dx = 1$$

i.e. the area is 1.

What then if the function drops to 0 at 0.5, but only at 0.5? Clearly the integral (area) is still 1, and so on for many such zero values; i.e. the integral is still defined, even though the function is not "well behaved" according to our ordinary understanding. However, at some stage, when the number of zeros become infinite, the area must decrease: at this stage (still roughly speaking) the function becomes unmeasurable.

Incidentally, Borel measurability is a general criterion applied to functions. It is claimed that the class of multilayer neural networks with three layers, feedforward, and using a sigmoid activation function can be trained to represent any Borel measurable function (see Hecht-Nielsen 1990, p. 122 for a discussion).

Random Variable

If, to every outcome, ω, of an experiment, we assign a number, $X(\omega)$, X is called a *random variable* (r.v.). X is a function over the set $\Omega = \{\omega_1, \omega_2, \ldots\}$ of outcomes; if the range of X is the real numbers or some subset of them, X is a *continuous* r.v.; if the range of X is some integer set, then X is a *discrete* r.v.

Distribution Function

Also called cumulative discribution function.

1. Of a continuous r.v.

   $$FX(x) = P\{-\infty < X \leq x\}$$

   Note: $\{..\}$ is an event, so that, although X is a function over outcomes (Ω), FX is a function over events.

2. Of a discrete r.v.

   If X can assume only a finite number of possible values $x_1, x_2, \ldots, x_n$, the probability of the matter can be adequately described by defining a corresponding list of probabilities, $p_1, p_2, \ldots, p_n$. In this case FX is shaped like a staircase. And, there is little need for the probability density function (of a discrete r.v.) described below.

Probability Density Function

If FX (continuous) is differentiable,

$$fX(x) = d/dxFX(x)$$

is called the *probability density function* of the r.v. X. Note that the values of fX are *not* probabilities; the values of FX are. fX must be integrated to get useful probability values, e.g.

$$P\{x_1 \leq X \leq x_2\} = \int_{x_1}^{x_2} fX(x)dx$$

Expected Value of a Random Variable

1. Continuous

$$mX = E\{X\} = \int_{-\infty}^{+\infty} xfX(x)dx$$

2. Discrete

$$mX = EX = \sum_i x_i p_i$$

General interpretation of expected value: average over range of x, weighted by probability of individual values.

In practice, it is usual that neither $fX(x)$ nor p are known, and *estimates* must be used: e.g. for the mean of a discrete random variable:

$$mX = \sum_{j=1}^{N} x_j/N$$

where the x_j are representative examples of the r.v., and where N is large enough that the sample, $\{x_1, \ldots, x_N\}$, in turn is large enough that the frequency of occurrence of x values properly represents the probabilities.

Random Vector

If the value assigned to an outcome, ω, is vector valued,

$$X = [X_1, \ldots, X_n]$$

then X is called a *random vector*, i.e. $X \in R^n$.

Note: there is temptation to think of X as a sequence of random variables, i.e. each X_i generated by a separate outcome, ω. Strictly, this is incorrect: in a random vector each $X_i, i = 1 \ldots n$, is generated by the same outcome. However, there are no immediate detrimental consequences of making such a mistake. And, on reflection, it may well be possible to "manufacture" a composite experiment whose outcome is a set containing a number of ωs.

A.2.2 Random Processes

Definition

A remark first: the adjective "stochastic" is entirely equivalent to "random"; thus the frequently encountered term "stochastic process" is equivalent to "random process".

As already stated, a r.v. is a rule for assigning a number, X, to each outcome, ω. Correspondingly, a *random process* is a rule for assigning, to each outcome, a *function*, $X(s,\omega)$. s is called the parameter, or the index, of the random process. Sometimes $X(s,\omega)$ is written $X_s(\omega)$.

S is the set of admissable parameter values, s. In general, S is a subset of the real line, R, or of R^n. Commonly, S is the time axis, and X is a function of time: $X(t,\omega)$, or $X_t(\omega)$; this is why t replaces s in most of the literature.

If S is one-dimensional, and discrete (i.e. the parameters can take only a finite set of values), X is just a random vector, which is equivalent to a discrete parameter random process. Note: a discrete *state* random process refers to one in which the values of X are discrete.

Interpretations of a random process (one-dimensional) (see Papoulis, 1991, p. 285):

1. A family (an *ensemble* in the literature) of functions, $X(s,\omega)$.
2. A single function, $X(s)$, i.e. ω is fixed.
3. If s is fixed and ω is variable, then $X(s)$ is a random variable equal to the *state* of the process at 'time' s.
4. If s and ω are fixed, $X(s)$ is a (plain) number.

   Although we have used s in this section, we will bow to the preponderance of usage, and use t (for time) in later sections.

Random Fields

If S is two-dimensional or of greater dimensionality then X is called a *random field*. Thus, if $S = \{x,y : x_0 \leq x \leq x_1, y_0 \leq y \leq y_1\}$, i.e. the domain commonly associated with a (continuous) image function, we have a random field (as in e.g., Rosenfeld and Kak, p.38). Likewise for a discrete image, $S = \{(x_1,y_1),\ldots,(x_n,y_n)\}$.

First Order Statistics of Random Processes

Since $x(t)$ is a random variable for a given t, we can extend the definition of functions and expected values by including an extra parameter, s (or t).

1. Distribution function.

   For a given t,

$$FX(x,t) = P{-\infty} < X(t) \leq x$$

2. Expected Value (Mean).

   Continuous case.

$$mX(t) = E\{X(t)\} = \int_{\infty}^{+\infty} x(t)fX(x,t)dx$$

   The discrete case is defined analogously.

3. Probability density function, pdf,

$$fX(x,t) = \frac{\partial}{\partial x} FX(x,t)$$

(This is the partial derivative of $FX(x,t)$ with respect to x.)

Second Order Statistics of Random Processes

1. Distribution function.

   The second order distribution of the random process $x(t)$ is the joint distribution

$$FXX(x_1, x_2; t_1, t_2) = P\{x(t_1) \leq x_1, x(t_2) \leq x_2\}$$

2. Probability density function, pdf,

$$fXX(x,t) = \frac{\partial^2}{\partial x_1 \partial x_2} FXX(x_1, x_2; t_1, t_2)$$

And, of course, to fully describe the probability distribution of a random process it is neccessary to extend to nth-order:

$$FXt_1, \ldots, Xt_n(x_1, \ldots, x_n; t_1, \ldots, t_n) = P\{x(t_1) \leq x_1, \ldots, x(t_n) \leq x_n\}$$

$FXt_1, \ldots Xt_n$ in the foregoing equation completely determines the statistical properties of $x(t)$. The joint density function $fXt_1, \ldots Xt_n$ is defined by analogy with the above definition of pdf.

Autocorrelation

The autocorrelation of a random process x(t) is the expected value of the product of the two random variables x(t1), x(t2):

Continuous.

$$R_{XX}(t_1, t_2) = E\{x(t_1)x(t_2)\}$$

$$= \int_{\infty}^{+\infty} \int_{\infty}^{+\infty} x_1 x_2 f_{XX}(x_1, x_2; t_1, t_2) dx_1 dx_2$$

Discrete.

$$R_{XX}(t_i, t_j) = \sum_{j=1}^{n} \sum_{j=1}^{n} x_i x_j p_{ij}$$

From its definition,

$$R_{XX}(t_1, t_2) = R_{XX}(t_2, t_1)$$

i.e. R_{XX} is symmetric.

Autocovariance.

$$C_{XX}(t_1, t_2) = E\{(x(t_1) - mX(t_1)).(x(t_2) - mX(t_2))\}$$

$$= R_{XX}(t_1, t_2) - mX(t_1).mX(t_2)$$

Stationary Process

1. Strict-sense stationary (see Papoulis, 1991, p. 297).

   Let $t_1, \ldots t_n$ be a set of points in T, likewise $t_1 + t_0, \ldots t_n + t_0$; the corresponding r.v.s are characterised by nth-order joint pdfs:

   $$fXt_1, \ldots Xt_n(x(t_1), \ldots, x(t_n); t_1, t_2, \ldots t_n)$$

   , and

   $$fXt_1 + t_0, \ldots, Xt_n + t_0(x(t_1 + t_0), \ldots, x(t_n + t_0; t_1 + t_0, t_2 + t_0, \ldots t_n + t_0)$$

   When these two functions are equal $\forall \;\; t_0$, then the process is *strict-sense stationary*, i.e. *all* statistical properties are invariant to shifts in the origin; there is an analogous definition for random fields (e.g. images).

2. Wide-sense stationary.

   Also called second-order stationary.

   If the following two conditions are met, the process is called wide-sense stationary:

   $$mX(t) = mX$$

   i.e. m is constant for all t, and,

   $$R_{XX}(t_1, t_2) = R_{XX}(t_1 - t_2)$$

   i.e. the autocorrelation depends only on $(t_1 - t_2)$ (the displacement).

   In the discrete case, wide-sense stationarity has the following important consequence, when R_{XX} is expressed as a matrix:

   $$R_{XX} = \left(\begin{array}{cccccc} r_{00} & r_{01} & \cdots & \cdots & \cdots & r_{0,n-1} \\ r_{10} & r_{11} & \cdots & \cdots & \cdots & r_{1,n-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{n-1,0} & r_{n-1,1} & \cdots & \cdots & \cdots & r_{n-1,n-1} \end{array} \right)$$

   from which we get, using relations established previously:

   $$R_{XX} = \left(\begin{array}{cccccc} r_0 & r_1 & r_2 & \cdots & \cdots & r_{n-1} \\ r_1 & r_0 & r_1 & \cdots & \cdots & r_{n-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{n-1} & r_{n-2} & \cdots & \cdots & \cdots & r_0 \end{array} \right)$$

i.e. R_{XX} is a circulant matrix – each row is merely the previous row rotated by one position. Sets of linear simultaneous equations that are characterised by a circulant matrix (or more generally a Toeplitz matrix) can be solved using fast and recursive algorithms – such solution is required in prediction.

Another property of Toeplitz matrices is that they are diagonalised by the Discrete Fourier Transform – an important consequence of this fact is that the DFT is equivalent to the Karhunen-Loève Transform for such data (the KL transform is an important transform in lossy data compression).

Incidentally, discrete convolution can be expressed as multiplication by a circulant matrix – the delayed impulse response weights form the rows; therefore the matrix may be disgonalised by the DFT, and this is why convolution decomposes into multiplication in the Discrete Fourier domain.

To read further, see Jain (1989), Pratt (1991), and Press et al. (1992).

Ergodic Processes

The statistics defined in the previous sections on random processes are defined by taking the expectation, $E\{.\}$, over the *ensemble* of xs; now, we rarely have access to a sufficiently large ensemble of xs that we can estimate pdfs; indeed, practically, we rarely have more than *one sample*, x. If the process is *ergodic*, i.e. roughly speaking, we can replace expectations over the ensemble with *time averages* over one sample, we have a practical method by which to estimate statistics.

Of course, a process first has to be stationary, so that these time averages do not vary with time.

Thus, the mean of a (discrete) ergodic process:

$$mX_i = E\{x_i\}$$

can be estimated

$$\hat{m}X = 1/N \sum_{i=1}^{N} x_i$$

Another example is the estimate of autocorrelation:

$$\hat{R}_{XX}(k) = 1/N \sum_{i=1}^{N} x(i+k)x(i)$$

As with stationarity, ergodicity can be wide-sense (e.g. ergodic in the mean (first-order), ergodic in autocorrelation (second-order) etc., or strict-sense, in which the ergodicity is defined in terms of the probability functions.

Markov Process

A Markov process is a random process in which the probability of achieving any future state depends only on the present, and not on the past. As with stationarity and ergodicity, Markov processes can be defined as strict sense or wide sense.

The definition of a strict-sense Markov process can be given more formally, in terms of conditional pdfs as follows:

$$fXt_{m+1}, Xt_{m+2} \ldots, Xt_n \mid Xt_1, \ldots Xt_m(x_{m+1}, xt_{m+2} \ldots, x_n \mid x_1, \ldots, x_m)$$

$$= fXt_m, \ldots Xt_n \mid Xt_m(x_{m+1}, \ldots, x_n \mid x_m)$$

where t_m = present time, and $\mid$ denotes conditional (probability).

In a strict-sense Markov process, the probabilities of states at $t_{m+1} \ldots$ depend only on the state at t_m (present) and not on $t_{m-1}, t_{m-2}, \ldots$ and backwards.

Clearly this has strong implications for the usage of past states in predicting the future.

In an image context (see Rosenfeld and Kak p. 312), we have Markov random fields; in that context Markov means that the probability of the state (greylevel) of a pixel depends only on the states of its eight neighbours.

Gaussian Process

The random process, X_t, is called a Gaussian process, provided that for any finite collection of times $(t_1, t_2, \ldots, t_n)$ the vector of states $X = (xt_1, xt_2, \ldots, xt_n)$ has the joint pdf:

$$fX(x) = (\frac{1}{2}\pi\frac{n}{2})(\mid C_{XX} \mid \frac{1}{2})\exp\{-\frac{1}{2}(x-m)^T C_{XX}^{-1}(x-m)\}$$

where C_{XX} is the autocovariance, $m = E\{x_t\}$ and $\mid . \mid$ denotes determinant.

Gaussian processes are good models for many natural random processes. In addition, they are analytically convenient, Additionally their pdf is totally determined by first and second order statistics (mean and autocovariance, respectively).

A.2.3 Further Background Reading

1. Chung, K.L. 1968. A Course in Probability Theory. New York: Harcourt, Brace and World.

2. Feller, W. 1966. An Introduction to Probability Theory and its Applications. Vol II. New York: John Wiley and Sons.

3. Hecht-Nielsen R. 1990. Neurocomputing. Reading, Mass: Addison- Wesley.

4. A.K. Jain. 1989. Fundamentals of Digital Image Processing. Englewood Cliffs, NJ: Prentice-Hall Int.

5. Kosko, B. 1992. Neural Networks and Fuzzy Systems. Englewood Cliffs, NJ: Prentice-Hall Int., 1992

6. Mortensen R.E. 1987. Random Signals and Systems. New York: John Wiley and Sons.

7. Papoulis A. 1991. Probability, Random Variables and Stochastic Processes. 3rd ed. New York: McGraw-Hill.

8. W.K. Pratt. 1991. Digital Image Processing. New York: Wiley- Interscience.

9. W.H. Press, S.A. Teukolsky, W.T. Vetterling,and B.P. Flannery. 1992. Numerical Recipes in C. Cambridge, U.K. : Cambridge University Press.

10. Proakis J.G. 1989. Digital Communications. 2nd ed. New York: McGraw-Hill.

11. Rosenfeld A. and A.C. Kak. 1982. Digital Picture Processing. 2nd ed. London: Academic Press. (2 Volumes).

12. Thomasian, A.J. 1969. The Structure of Probability Theory with Applications. New York: McGraw-Hill.

13. Widrow, B., and Lehr, M.A. 1990. 30 Years of Adaptive Neural Networks. Proceedings of the IEEE. 78, No. 9.

A.3 Linear Algebra

A.3.1 Basic Definitions

Vectors and Matrices

Pattern or measurement vector before any processing:

$$x = (x_0, x_1, \dots x_i, \dots x_{p-1})^T$$

a $p \times 1$ column vector.

After transformation:

$$y = (y_0, y_1, \dots y_i, \dots y_{p-1})^T$$

a $q \times 1$ column vector.

A matrix transformation is defined by an equation of the form:

$$y = Ax$$

wit respective dimensionalities: $q \times 1, q \times p$, and $p \times 1$.

Multivariate Statistics

$$x = (x_0, x_1, \dots x_i, \dots x_{p-1})^T$$

a $p \times 1$ feature vector.

Mean vector:

$$m = (m_0, m_1, \dots m_i, \dots m_{p-1})^T$$

$$= E\{x\}$$

where $E\{.\}$ denotes expectation.

Normally we estimate expected values using the sample average, e.g. mean m is estimated using the average of x computed over a sample of n values, generically called x_i:

$$\hat{m} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Autocorrelation matrix:

$$R = E\{xx^T\}$$
$$= [r_{ij}]$$

where

$$r_{ij} = E\{x_i x_j\}$$

the expected value of the product of the ith and jth components of x.

Normally we will use the sample autocorrelation matrix, i.e. $\hat{R}$ estimated from a sample of n vectors, $x_i, i = 1 \ldots n$,

$$\hat{R} = \frac{1}{n}\sum_{i=1}^{n} x_i x_i$$

It is sometimes convenient to write the autocorrelation completely in matrix notation; let X be the $p \times n$ matrix formed by arranging the n x_i values as columns

$$X = [x_1 x_2 \ldots x_i \ldots x_n]$$

a $p \times n$ matrix, and so

$$\hat{R} = \frac{1}{n} X X^T$$

Covariance matrix:

$$S = E\{(x - m)(x - m)^T\}$$

$$= [s_{ij}]$$

where

$$s_{ij} = E\{(x_i - m_i)(x_j - m_j)\}$$

the expected value of the product of the ith and jth components of the deviation of x from its mean.

The diagonal elements s_{ii} of S are the variances of the element x_i.

It is easy to verify that:

$$S = R - mm^T$$

Also, there is a matrix representation, analogous to the autocorrelation above.

If $x' = (x - m)$, i.e. the pattern vector reduced to zero mean, and

$$X' = [x'_1 x'_2 \dots x'_i \dots x'_n]$$

of dimensions $p \times n$, then the sample covariance can be rewritten as

$$\hat{S} = \frac{1}{n} X' X'^T$$

Multivariate Gaussian Random Vectors:

Multivariate vectors generated by a multivariate random process follow the probability density function (pdf):

$$f_x(x) = (\frac{1}{2}\pi\frac{n}{2})(\mid S \mid \frac{1}{2}) \exp\{-\frac{1}{2}(x-m)^T S^{-1}(x-m)\}$$

where S is the covariance matrix, $m = E\{x\}$ and $\mid . \mid$ denotes determinant.

Gaussian processes are good models for many natural random processes. In addition, they are analytically convenient, since their pdf is totally determined by first and second order statistics (mean and covariance, respectively).

Statistics after Transformation:

If we transform into a feature space $y = Ax$, the mean, autocorrelation, and covariance statistics are transformed as follows:

Mean: $m' = Am$

Autocorrelation: $R' = ARA^T$

Covariance: $S' = ASA^T$

Prior Probabilities:

Many pattern recognition algorithms use (estimates of) the relative frequency of occurrence of the various classes in the population; these are called *prior*, or *a priori*, probabilities, because these probabilities are known before any measurement is made.

Prior probabilities are denoted P_1, P_2 etc. for P(class 1) etc.

Contrast *posterior*, or *a posteriori* probabilities, which are the probabilities of the classes after the measurements have been taken into account.

A.3.2 Linear Simultaneous Equations

Eqn. A.4 is a system of linear (simultaneous) equations.

$$\begin{aligned} y_1 &= 3x_1 + 1x_2 \\ y_2 &= 2x_1 + 4x_2 \end{aligned} \tag{A.4}$$

Practically, Eqn. A.4 could express the following:

Price of an apple = x_1, price of an orange = x_2 (both unknown). Person A buys 3 oranges, and 1 apple and the total bill is 5p (y_1). Person B buys 2 oranges, 4 apples and the total bill is 10p (y_2).

Question: What is x_1, the price of apples, and x_2, the price of oranges?

$$\begin{aligned} (1) \qquad 5 &= 1x_1 + 3x_2 \\ (2) \qquad 10 &= 4x_1 + 2x_2 \\ (1) \Longrightarrow x_2 &= -5 + 3x_1 \end{aligned}$$

Substitute into (2):

$$\begin{aligned} 10 &= 4x_1 + 2(-5 + 3x_1) \\ 10 &= 10x_1 - 10 \\ 20 &= 10x_1 \\ 2 &= x_1 \end{aligned}$$

Now, substitute $x_1 = 2$ into (1):

$$\begin{aligned} 5 &= 2 + 3x_2 \\ 3 &= 3x_2 \\ x_2 &= 1 \end{aligned}$$

The simultaneous equations A.4 can be written in matrix form as follows:

$$y = Ax$$

where y is a two-dimensional vector, $y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ x is a two-dimensional vector, $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ and A is a 2 row × 2 column matrix, $A = \begin{pmatrix} 1 & 3 \\ 4 & 2 \end{pmatrix}$.

Note: a *vector* is simply an array of numbers. A vector with n numbers is called an *n-dimensional* vector; such a vector represents a point in n-dimensional space. Don't try to visualise $n > 3$. Just think of the n numbers grouped together.

In two- or three-dimensions it is possible to visualise a vector as a line with an arrow-head – the arrow indicates the path between the origin $(0, 0)$ and the point (x, y) that the vector represents; again, for our purposes this view has limited use.

Generally, a system of m equations, in n variables,

$$\begin{aligned} y_1 &= a_{11}x_1 + a_{12}x_2 \ldots + a_{1n}x_n \\ y_2 &= a_{21}x_1 + a_{22}x_2 \ldots + a_{2n}x_n \\ &\ldots \\ y_r &= a_{r1}x_1 + a_{r2}x_2 \ldots + a_{rc}x_c \ldots + a_{rn}x_n \\ &\ldots \\ y_m &= a_{m1}x_1 + a_{m2}x_2 \ldots + a_{mn}x_n \end{aligned}$$

can be written in matrix form as,

$$y = Ax \tag{A.5}$$

where y is an m-dimensional vector

$$y = \begin{pmatrix} y_1 \\ y_2 \\ . \\ . \\ y_m \end{pmatrix}$$

x is an n-dimensional vector,

$$x = \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_m \end{pmatrix}$$

and A is an m-row × n-column matrix

$$A = \begin{pmatrix} a_{11} & a_{12} & & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ .. & .. & .. & .. \\ .. & a_{rc} & .. & .. \\ .. & .. & .. & .. \\ a_{m1} & a_{m2} & .. & a_{mn} \end{pmatrix}$$

That is, the matrix A is a rectangular array of numbers whose element in row r, column c is a_{rc}. The matrix A is said to be $m \times n$, i.e. m rows, n columns.

Vectors can be considered as specialisations of matrices, i.e. matrices with only one column. Thus x is $m \times 1$, and y is $n \times 1$.

Eqns. A.4 or A.5 can be interpreted as the definition of a function which takes n arguments $(x_1, x_2..x_n)$ and returns m variables $(y_1, y_2 \ldots y_m)$. Such a function is also called a *transformation*: it transforms n-tuples of real numbers to m-tuples of real numbers.

Eqn. A.5 is a *linear* transformation because there are no terms in x_r^2 or higher, only in x_r.

A.3.3 Basic Matrix Operations

Matrix Multiplication

We may multiply two matrices $A, m \times n$, and $B, q \times p$, as long as $n = q$. Such a multiplication produces an $m \times p$ result. Thus,

$$\underset{m \times p}{C} = \underset{m \times n}{A} \quad \underset{n \times p}{B} \tag{A.6}$$

Method: The element at the rth row and cth column of C is the product (*dot* or *inner* or *scalar* product) of the rth row vector of A with the cth column vector of B.

Pictorially:

```
            n                   p                p
     ----------------    ----------        -----------
     |---->         |    |        | |      |         |
     |  A           |    | B      | |  =   |  C      |
     |              |    |        | |      |         |
m    |              |    |        | | n    |         | m
     ----------------    |       V |       -----------
                         |         |
|         |
----------
```

Thus,

$$C = AB$$

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

$$B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

$$C = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

Example Consider Eqn. A.5, $y = Ax$. Thus the product of $A(m \times n)$ and $x(n \times 1)$

$$y_1 = a_{11}x_1 + a_{12}x_2 \ldots + a_{1n}x_n$$

(row1 col1 = product of 1st row of A with 1st column of x) etc.

The product is $(m \times n) \times (n \times 1)$ so the result is $(m \times 1)$, i.e. y.

Example Apply the transformation given by

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix}$$

to

$$x = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Scaling and Rotation

Consider scaling and rotation in computer graphics. Here the vectors are:

$$\begin{pmatrix} x \\ y \end{pmatrix}$$

and the output, transformed, vector is:

$$\begin{pmatrix} x' \\ y' \end{pmatrix}$$

The scaling transformation takes the form:

$$x' = xS_x$$
$$y' = yS_y$$

That is, x is expanded ($S_x > 1$) or contracted ($S_x < 1$) to give x'; ditto y-axis. S_x and S_y are called scaling factors. The matrix is

$$\begin{pmatrix} S_x & 0 \\ 0 & S_y \end{pmatrix}$$

The following transformation rotates (x, y) a clockwise angle B about the origin (0,0):

$$\begin{aligned} x' &= x\cos B + y\sin B \\ y' &= -x\sin B + y\cos B \end{aligned}$$

The matrix is

$$R(B) = \begin{pmatrix} \cos B & \sin B \\ -\sin B & \cos B \end{pmatrix} \tag{A.7}$$

Example (a) Rotate the point

$$\begin{pmatrix} 0.707 \\ 0.707 \end{pmatrix}$$

clockwise by 45 degrees.

(Note: $\sin 45 = 0.707 = 1/\sqrt{2} = \cos 45; 0.707 \times 0.707 = 0.5$.)

(b) Rotate the point

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

clockwise by 90 degrees.

Example What is the effect of applying the rotation matrix twice? That is, what is

$$R(B)R(B)\begin{pmatrix} x \\ y \end{pmatrix}$$

The following formulae may be useful:

$$\begin{aligned} \sin(a+b) &= \sin(a)\cos(b) + \cos(a)\sin(b) \\ \cos(a+b) &= \cos(a)\cos(b) - \sin(a)\sin(b) \end{aligned}$$

Example What is the effect of applying the negative rotation, $-B$, to a point that has already been rotated by $+B$. That is, what is

$$R(-B)R(B)\begin{pmatrix} x \\ y \end{pmatrix}$$

Multiplication by a Scalar

$$c\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \begin{pmatrix} ca_{11} & ca_{12} \\ ca_{21} & ca_{22} \end{pmatrix}$$

Addition of Matrices

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}+b_{11} & a_{12}+b_{12} \\ a_{21}+b_{21} & a_{22}+b_{22} \end{pmatrix}$$

The matrices must be the same size (dimensions).

Inverses of Matrices

Only for square matrices ($m = n$).

Consider Eqn. A.4:

$$\begin{aligned} y_1 &= 3x_1 + 1x_2 \\ y_2 &= 2x_1 + 4x_2 \end{aligned}$$

i.e. $y = Ax$ where

$$y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

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

Apply this to

$$x = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Get:

$$\begin{aligned} y_1 &= 3.1 + 1.2 = 5 \\ y_2 &= 2.1 + 4.2 = 10 \end{aligned}$$

What if you know $y = (5, 10)$ and want to retrieve $x = (x_1, x_2)$?

Answer: Apply the inverse transformation to y. That is, multiply y by the inverse of the matrix.

$$x = A^{-1}y$$

In the case of a 2 × 2 matrix

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

$$A^{-1} = \frac{1}{\mid A \mid} \begin{pmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{pmatrix} \tag{A.8}$$

where the determinant of the array, A, is $\mid A \mid = a_{11}a_{22} - a_{12}a_{21}$

If $\mid A \mid$ is zero, then A is not invertible, it is *singular*.

Thus for

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

we have $\mid A \mid = 3 \times 4 - 2 \times 1 = 10$ so

$$A^{-1} = (1/10) \begin{pmatrix} 4 & -1 \\ -2 & 3 \end{pmatrix} = \begin{pmatrix} 0.4 & -0.1 \\ -0.2 & 0.3 \end{pmatrix}$$

Therefore, apply A^{-1} to $\begin{pmatrix} 5 \\ 10 \end{pmatrix}$

We find: $A^{-1}y =$

$$\begin{pmatrix} 0.4 & -0.1 \\ -0.2 & 0.3 \end{pmatrix} . \begin{pmatrix} 5 \\ 10 \end{pmatrix} = \begin{pmatrix} 5 \times 0.4 + 10 \times -0.1 \\ 5 \times -0.2 + 10 \times 0.3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

which is what we started off with, i.e.

$$x = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Note: in solving the linear equation system above for the price of apples and oranges, we were actually doing something that is very similar to inverting the matrix

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

A.3.4 Particular Matrices

Diagonal Matrices

The scaling faactor matrix mentioned above, in dealing with rotation and scaling,

$$A = \begin{pmatrix} Sx & 0 \\ 0 & Sy \end{pmatrix}$$

is *diagonal*, i.e. the only non-zero elements are on the diagonal.

The inverse of a diagonal matrix

$$\begin{pmatrix} a_{11} & 0 \\ 0 & a_{22} \end{pmatrix}$$

is

$$\begin{pmatrix} 1/a_{11} & 0 \\ 0 & 1/a_{22} \end{pmatrix}$$

Transpose of a Matrix (A^t)

Only for square matrices. If

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

then

$$A^t = \begin{pmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{pmatrix}$$

i.e. replace column 1 with row 1 etc.

The transpose is often written as A^t or A^T or A'. It is pronounced “A-transpose”.

The Identity Matrix

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

i.e. produces no transformation effect. Thus, $IA = A$

Note: If $AB = I$ then $B = A^{-1}$.

Orthogonal Matrix

A matrix which satisfies the property:

$$AA^t = I$$

i.e. the transpose of the matrix is its inverse.

Another way of viewing this is:

For each row of the matrix $(a_{r1}a_{r2}....a_{rn})$, the dot product with itself is 1, and with all other rows 0. I.e.

$$\begin{aligned} \textstyle\sum_{c=1}^{n} a_{rc}a_{pc} &= 1 \text{ for } r = p \\ &= 0 \text{ otherwise} \end{aligned}$$

A.3.5 Complex Numbers

A complex number is simply a convenient way of representing the pair of numbers that represent the coordinates (x, y) of points in a plane,

$$z = x + jy$$

where $j = \sqrt{-1}$.

In many ways, a complex number is like a two-dimensional vector.

The *modulus* of the complex number (which may be interpreted as the distance between the origin and (x, y)) is given by:

$$\mid z \mid = \mid x + jy \mid = \sqrt{(}x \times x + y \times y)$$

i.e. using Pythagoras' Theorem

The angle, or *argument*, which may be interpreted as the angle between the line $(0, 0)$ to (x, y) and the x-axis, is given by:

$$\arg z = \arctan y/x$$

i.e. the angle whose tangent (opposite/adjacent) is y/x.

Addition of complex numbers is as follows: If

$$z = x + jy, \quad w = u + jv$$

then

$$z + w = x + u + j(y + v)$$

A graphical interpretation of addition of complex numbers is,

– draw a line from (0,0) to (x, y),

– using (x, y) as the origin, draw a line to (u, v),

– the point reached is $(x + u, y + v)$.

I.e. *vector* addition.

Multiplication of complex numbers is as follows:
If

$$z = x + jy, \quad w = u + jv$$

then

$$z.w = (x + jy).(u + jv) = x.u + j.j.y.v + j.(y.u + x.v)$$

We use $j.j = -1$ (i.e.$\sqrt{-1}\sqrt{-1} = -1$) This gives:

$$z.w = (x.u - y.v) + j.(y.u + x.v)$$

Note: if complex numbers have zero imaginary parts, the rules given here collapse to the rules of normal arithmetic for real numbers.

Example Verify the last statement, i.e. set $y, v = 0$ in addition and multiplication of complex numbers.

The *complex conjugate* of a complex number, $c = a + jb$ is

$$c^* = a - jb$$

Complex Numbers and Matrices

Matrices and vectors can contain complex numbers. The rules for matrix addition, multiplication, given above, all apply; we just replace the normal addition, multiplication with the complex versions given in the previous section dealing with "Complex Numbers".

A.3.6 Further Matrix and Vector Operations

Vector Inner (Dot) Product

Let vector

$$x = \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{pmatrix}$$

i.e. $x = (x_1, x_2, \ldots x_n)^t$ (t denotes transpose), and vector $y = (y_1, y_2, \ldots y_n)^t$

The *inner product* (*dot product*, *scalar product*) of x and y is the matrix product

$$x^t y$$

Dimensions: 1×1, $1 \times n$, $n \times 1$

This is the same as:

$$xy^t = \sum_{i=1}^{n} x_i y_i$$

If the dot product of two vectors is 0, they are said to be *orthogonal*.

Vector Addition

Three $n \times 1$ vectors x, y, z:

$$z = x + y$$

with

$$\begin{pmatrix} z_1 \\ z_2 \\ . \\ . \\ z_n \end{pmatrix} = \begin{pmatrix} x_1 + y_1 \\ x_2 + y_2 \\ . \\ . \\ x_n + y_n \end{pmatrix}$$

Distance between Vectors

Considering n-dimensional vectors as points in n-dimensional space, we can talk about the distance, d, between vectors x and y. The following is the squared distance:

$$d^2(x, y) = (x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2$$

or

$$d^2(x, y) = \sum_{i=1}^{n} (x_i - y_i)^2 \tag{A.9}$$

Example Determine the Euclidean distance between the points (1,1), (1,3).

```
         |
       3 +   * (1,3)
         |
       2 +
         |
dim 2  1 +   * (1,1)
         |
         +---+---+---+---+---+---
             1   2   3   4        dim 1
```

$$d = \sqrt{(1-1)^2 + (1-3)^2} = \sqrt{0+4} = 2$$

Length or Magnitude of a Vector

Considering now an n-dimensional vector as the line joining the origin to its point in n-dimensional space, we can talk about the *length* – or, more usually, the *magnitude* – of a vector as the distance between the vectors x and the origin (0,0,0...):

$$\mid x \mid = \sqrt{(\sum_{i=1}^{n} (x_i)^2)}$$

Example Verify that the magnitude of the vector (1,0) is 1.

Example Verify that the magnitude of the vector (0,2) is 2.

Example Verify that the magnitude of the vector (1,1) is 1.414; i.e. $\sqrt{2}$. Note that it is *not* 1, as you can easily verify by sketching.

Normalized or Unit Length Vectors

Quite often we are just interested in the relative *directions* of vectors and, for easier comparison, we would like to reduce all vectors to unity magnitude – this is called *normalization.*

Normalization is performed by the following (cf. scaling dealt with above) transformation:

$$x_{i'} = x_i / \mid x \mid$$

where $x_{i'}$ is the ith component of the normalised vector, x_i is the ith component of the original vector, and $\mid x \mid$ is the magnitude of the original vector.

Example Normalize the vector (0,1); *Answer:* (0,1).

Example (a) Normalize the vector (0,2); *Answer:* (0,1) (b) Verify, with a diagram, that normalization has retained the direction of the original vector.

Example (a) Normalize the vector (1,1); *Answer:* (0.707,0.707).

(b) Verify.

Answer:

$0.707 = 1/\sqrt{2}$

magnitude $= \sqrt{x_1^2 + x_2^2} = \sqrt{1/\sqrt{2}^2 + 1/\sqrt{2}^2} = \sqrt{1/2 + 1/2} = 1$

Example Verify that, in two dimensions, all unit vectors lie on the unit circle (a circle with centre at the origin (0,0) and with radius 1).

Template Matching of Unit Vectors

Quite often we wish to compare two vectors, x and y (assume they have already been normalized).

One way is to compute the distance between them.

$$d = \sqrt{(}\sum_{i=1}^{n}(x_i - y_i)^2)$$

Then we can use the common sense rule:

Small distance $\Longrightarrow$ *similar*
Big distance $\Longrightarrow$ *different*

Alternatively, we can compute how well the corresponding components correlate, i.e. perform a template matching by multiplying corresponding components and summing (i.e. a dot product):

$$c = \sum_{i=1}^{n} x_i y_i$$

Then we can use the rule:

Big correlation value (c) $\Longrightarrow$ *similar*
Small $\Longrightarrow$ *different*
(See section 3.2.13: if $c = 0$ the two vectors are orthogonal.)
Maximizing correlation is equivalent to minimizing distance.

Intuitive proof (two-dimensions):
We expand for the case of 2-dimensions:

$$d = ((x_1 - y_1)^2 + (x_2 - y_2)^2) = (x_1^2 + y_1^2 - 2x_1y_1 + x_2^2 + y_2^2 - 2x_2y_2)$$

$$= (x_1^2 + x_2^2 + y_1^2 + y_2^2 - 2x_1y_1 - 2x_2y_2)$$

$$d = \sum_{i=1}^{2} x_i^2 + \sum_{i=1}^{2} y_i^2 - 2\sum_{i=1}^{2} x_iy_i$$

Since x and y are normalized to unit magnitude:

$$\sum_{i=1}^{2} x_i^2 = \sum_{i=1}^{2} y_i^2 = 1$$

Thus,

$$d = -2(c - 1)$$

When the vectors are the same, $x = y$, so that

$$c = \sum x_iy_i = \sum x_ix_i = 1$$

since x is unit magnitude. This is the highest possible value that c can attain (i.e. when the vectors are the *same*, the components are completely matched/correlated).

A.3.7 Vector Spaces

Vectors in Neural Networks and Pattern Recognition

Many approaches to automatic pattern recognition (especially neural networks) use representations of patterns as arrays of numbers (vectors).

The recognition process often involves finding the "most similar", of a set of stored patterns, to an unknown pattern.

Obviously, the distance is clearly a good measure of similarity: small distance large similarity; clearly also, template matching or correlation is intuitively appealing. We have shown that they both give the same result.

Neural Networks:

The computation performed by a single neuron, as used in artificial neural networks, is simply the dot product between the input excitations x_i and the weights, w_i,

$$\text{sum } = \sum_{i=1}^{n} x_i w_i$$

followed by passing *sum* through some threshold function, such as:

$$\begin{aligned} \text{output} &= 1 \text{ if sum } > T \\ &= 0 \text{ otherwise} \end{aligned}$$

Note the similarity with template matching.

Example The Figure below shows a letter 'C' in a small (3×3) part of a digital image (a digital picture). A digital picture is represented by brightness numbers (pixels) for each picture point.

Now, represent the nine pixel values as elements of a vector. Assuming the character is white-on-black and that bright (filled in with '*') corresponds to '1', and dark to '0', the components of the vector corresponding to the 'C' are:

$$x_1 = 1, x_2 = 1, x_3 = 1, x_4 = 1, x_5 = 0, x_6 = 0, x_7 = 1, x_8 = 1, x_9 = 1$$

```
  Pixel representation:

   1    2    3
  +----+----+----+
  |****|****|****|
  |****|****|****|
  +----+----+----+
4 |****|5   |6   |
  |****|    |    |
  +----+----+----+
  |****|****|****|
  |****|****|****|
  +----+----+----+
   7     8    9

    A Letter 'C'
```

The letter 'T' would give a different observation vector:

```
'T': 1,1,1,    0,1,0,    0,1,0
'O': 1,1,1,    1,0,1,    1,1,1
'C': 1,1,1,    1,0,0,    1,1,1
etc.
```

Linear Independence of Vectors

Two vectors a_i, a_j,

$$a_i = (a_{i1}, a_{i2}, \ldots, a_{ip})$$

$$a_j = (a_{j1}, a_{j2}, \ldots, a_{jp})$$

are linearly *dependent* if one can be written as a scalar product of the other,

$$a_i = ca_j = (ca_{j1}, ca_{j2}, \ldots, ca_{jp})$$

i.e. the vectors differ only by a scale factor, c, that is applied to all elements.

In such a case, the directions of the vectors are the same; only their length differs by the scaing factor c.

If we have

$$b = \sum_{j=1}^{n} c_j a_j$$

then b is a linear combination of the a_js and is linearly dependent on them.

Normally, as in the next section (rank) we are interested in the linear independence of vectors formed by rows of a matrix. If we have one row of a matrix that is linearly dependent on (some – or all) the others, this means that the simultaneous equation associated with that row contributes no new information.

Rank of a Matrix

Given a q-row × p-column matrix, A

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & \ldots & \ldots & a_{1p} \\ a_{21} & a_{22} & \ldots & \ldots & \ldots & a_{2p} \\ \ldots & \ldots & \ldots & a_{rc} & \ldots & \ldots \\ a_{q1} & a_{q2} & \ldots & \ldots & \ldots & a_{qp} \end{pmatrix}$$

the rank of A is the number of linearly independent rows in it.

If $p = q$ the matrix is square, and we may need to invert it, it will only invert if all the rows are linearly independent; otherwise, the matrix is *singular* – non-invertable. One simple way of viewing this problem is that, for a system of simultaneous equations to be solvable, we need p equations in p unknowns; if one, or more, of the equations is linearly dependent on the others, this equation contributes no new information, i.e. we effectively have only $p - 1$ 'useful' equations, and the system is unsolvable – its rank is $p - 1$.

In pattern recognition and estimation, the incidence of singular or nearly singular matrices is insidious and common, e.g. a common source is taking reading for a dependent variable, y, say, for the same, or nearly the same values of the independent variable, x. It can lead to nonsense results – analogous to what happens close to divide by 0 in floating point arithmetic.

Eigenvalues and Eigenvectors

For any positive-definite matrix R, there exists a unitary matrix U that satisfies the following equation:

$$U^T RU = \Lambda$$

where

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \lambda_p \end{pmatrix}$$

Λ is a diagonal matrix containing the *eigenvalues* of R, and

$$U = \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ u_i \\ \dots \\ u_p \end{pmatrix}$$

is the matrix formed by the *eigenvectors*, u_i, of R.

Another equation governing eigenvalues and eigenvectors is

$$Ru_i = \lambda_i u_i$$

or, in matrix form, showing all the eigenvectors and eigenvalues:

$$RU = U\Lambda$$

General References

The following have been referenced in the text. For bibliographies relating to the different techniques studied, see the relevant sections in the respective chapters.

1. H.–M. Adorf, "Classification of low–resolution stellar spectra via template matching — a simulation study", *Data Analysis and Astronomy*, (Proceedings of International Workshop on Data Analysis and Astronomy, Erice, Italy, April 1986) Plenum Press, New York (1986, forthcoming).

2. M.R. Anderberg, *Cluster Analysis for Applications*, Academic Press, New York, 1973.

3. P.R. Bevington, *Data Reduction and Error Analysis for the Physical Sciences*, McGraw–Hill, New York, 1969.

4. R.K. Blashfield and M.S. Aldenderfer, "The literature on cluster analysis", *Multivariate Behavioral Research*, **13**, 271–295, 1978.

5. A. Boggess, A. Carr, D.C. Evans, D. Fischel, H.R. Freeman, C.F. Fuechsel, D.A. Klinglesmith, V.L. Krueger, G.W. Longanecker, J.V. Moore, E.J. Pyle, F. Rebar, K.O. Sizemore, W. Sparks, A.B. Underhill, H.D. Vitagliano, D.K. West, F. Macchetto, B. Fitton, P.J. Barker, E. Dunford, P.M. Gondhalekar, J.E. Hall, V.A.W. Harrison, M.B. Oliver, M.C.W. Sandford, P.A. Vaughan, A.K. Ward, B.E. Anderson, A. Boksenberg, C.I. Coleman, M.A.J. Snijders and R. Wilson, "The IUE spacecraft and instrumentation", *Nature*, **275**, 372–377, 1978a.

6. A. Boggess, R.C. Bohlin, D.C. Evans, H.R. Freeman, T.R. Gull, S.R. Heap, D.A. Klinglesmith, G.R. Longanecker, W. Sparks, D.K. West, A.V. Holm, P.M. Perry, F.H. Schiffer III, B.E. Turnrose, C.C. Wu, A.L. Lane, J.L. Linsky, B.D. Savage, P. Benvenuti, A. Cassatella, J. Clavel, A. Heck, F. Macchetto, M.V. Penston, P.L. Selvelli, E. Dunford, P.M. Gondhalekar, M.B. Oliver, M.C.W. Sandford, D.J. Stickland, A. Boksenberg, C.I. Coleman, M.A.J. Snijders and R. Wilson, "In–flight performance of the IUE", *Nature*, **275**, 377–385, 1978b.

7. J.M. Chambers, *Computational Methods for Data Analysis*, Wiley, New York, 1977.

8. A. Cucchiaro, M. Jaschek and C. Jaschek, *An Atlas of Ultraviolet Stellar Spectra*, Liège and Strasbourg, 1978.

9. E. Davoust and W.D. Pence, "Detailed bibliography on the surface photometry of galaxies", *Astronomy and Astrophysics Supplement Series*, **49**, 631–661, 1982.

10. B.S. Everitt, *Cluster Analysis*, Heinemann Educational Books, London, 1980 (2nd ed.).

11. A.D. Gordon, *Classification*, Chapman and Hall, London, 1981.

12. J.C. Gower and P. Legendre, "Metric and Euclidean properties of dissimilarity coefficients" *Journal of Classification* **3**, 5–48, 1986.

13. P.A.V. Hall and G.R. Dowling, "Approximate string matching", *Computing Surveys*, **12**, 381–402, 1980.

14. D.J. Hand, *Discrimination and Classification*, Wiley, New York, 1981.

15. A. Heck, "UV stellar spectral classification", in Y. Kondo et al. (eds.), *Scientific Accomplishments of the IUE*, D. Reidel, Dordrecht, 1986 (in press).

16. A. Heck, D. Egret, M. Jaschek and C. Jaschek, *IUE Low Resolution Spectra Reference Atlas — Part 1. Normal Stars*, European Space Agency Special Publication 1052, 1984a.

17. A. Heck, D. Egret, Ph. Nobelis and J.C. Turlot, "Statistical classification of IUE stellar spectra by the Variable Procrustean Bed (VPB) approach", in *Fourth European IUE Conference*, European Space Agency Special Publication 218, pp. 257–261, 1984b.

18. A. Heck, D. Egret, Ph. Nobelis and J.C. Turlot, "Statistical confirmation of the UV stellar classification system based on IUE low–dispersion stellar spectra", *Astrophysics and Space Science* **120**, 223–237, 1986a.

19. A. Heck, D. Egret, B.J.M. Hassall, C. Jaschek, M. Jaschek and A. Talavera, "IUE low–dispersion spectra reference atlas — Volume II. Peculiar stars" in *New Insights in Astrophysics — Eight Years of UV Astronomy with IUE*, European Space Agency Special Publication 263, 661–664, 1986b.

20. C. Jaschek, "Classification spectrale" in *Classification Stellaire*, Ecole de Goutelas, ed. D. Ballereau, Observatoire de Meudon, 1979.

21. M. Jaschek and C. Jaschek, "Classification of ultraviolet spectra", in *The MK Process and Stellar Classification*, ed. R.F. Garrison, David Dunlap Observatory, 290, 1984.

22. Y. Kondo, A. Boggess, M. Grewing, C. de Jager, A.L. Lane, J.L. Linsky, W. Wamsteker and R. Wilson (eds.), *Scientific Accomplishments of the IUE*, D. Reidel, Dordrecht, 1986 (in press).

23. J.B. Kruskal, "An overview of sequence comparison: time warps, string edits, and macromolecules", *SIAM Review*, **25**, 201–237, 1983.

24. J.B. Kruskal and M. Wish, *Multidimensional Scaling*, Sage, Beverly Hills, 1978.

25. M.J. Kurtz, "Classification methods: an introductory survey", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 47–58, 1983.

26. O. Lefèvre, A. Bijaoui, G. Mathez, J.P. Picat and G. Lelièvre, "Electronographic BV photometry of three distant clusters of galaxies", *Astronomy and Astrophysics*, **154**, 92–99, 1986.

27. L. Lebart, A. Morineau and K.M. Warwick, *Multivariate Statistical Analysis*, Wiley, New York, 1984.

28. T.E. Lutz, "Estimation — comments on least squares and other topics", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 179–185, 1983.

29. M. Lybanon, "A better least–squares method when both variables have uncertainties", *American Journal of Physics* **52**, 22–26, 1984.

30. W.W. Morgan, "The MK system and the MK process", in *The MK Process and Stellar Classification*, ed. R.F. Garrison, David Dunlap Observatory, 18, 1984.

31. F. Murtagh, "Structure of hierarchic clusterings: implications for information retrieval and for multivariate data analysis", *Information Processing and Management*, **20**, 611–617, 1984.

32. F. Murtagh, *Multidimensional Clustering Algorithms*, COMPSTAT Lectures Volume 4, Physica–Verlag, Wien, 1985.

33. J.V. Narlikar, "Statistical techniques in astronomy", *Sankhya: The Indian Journal of Statistics, Series B, Part 2*, **44**, 125–134, 1982.

34. S. Okamura, "Global structure of Virgo cluster galaxies", in eds. O.G. Richter and B. Binggeli, *ESO Workshop on The Virgo Cluster of Galaxies*, ESO Conference and Workshop Proceedings No. 20, 201–215, 1985.

35. W.D. Pence and E. Davoust, "Supplement to the detailed bibliography on the surface photometry of galaxies", *Astronomy and Astrophysics Supplement Series*, **60**, 517–526, 1985.

36. J. Pfleiderer, "Data set description and bias", in *Statistical Methods in Astronomy*, European Space Agency Special Publication 201, 3–11, 1983.

37. A. Rapoport and S. Fillenbaum, "An experimental study of semantic structures", in eds. A.K. Romney, R.N. Shepard and S.B. Nerlove, *Multidimensional Scaling; Theory and Applications in the Behavioral Sciences. Vol. 2, Applications*, Seminar Press, New York, 93–131, 1972.

38. F.J. Rohlf, "Generalization of the gap test for the detection of multivariate outliers", *Biometrics*, **31**, 93–101, 1975.

39. G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*, McGraw–Hill, New York, 1983.

40. D. Sankoff and J.B. Kruskal (Eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, Addison–Wesley, New York, 1983.

41. M.J. Seaton, "Interstellar extinction in the UV", *Monthly Notices of the Royal Astronomical Society* **187**, 73P–76P, 1979.

42. B.T. Smith *et al.*, *Matrix Eigensystem Routines — EISPACK Guide*, Lecture Notes in Computer Science 6, Springer Verlag, Berlin and New York, 1976.

43. P.H.A. Sneath and R.R. Sokal, *Numerical Taxonomy*, Freeman, San Francisco, 1973.

44. H. Späth, *Cluster Dissection and Analysis: Theory, Fortran Programs, Examples*, Ellis Horwood, Chichester, 1985.

45. B. Takase, K. Kodaira and S. Okamura, *An Atlas of Selected Galaxies*, University of Tokyo Press, Tokyo, 1984.

46. M. Thonnat, "Automatic morphological description of galaxies and classification by an expert system", INRIA Rapport de Recherche, Centre Sophia Antipolis, No. 387, 1985.

47. A. Tucker, *Applied Combinatorics*, Wiley, New York, 1980.

48. M. Volle, *Analyse des Données*, Economica, Paris, 1981.

49. J.V. Wall, "Practical statistics for astronomers. I. Definitions, the normal distribution, detection of signal", *Quarterly Journal of the Royal Astronomical Society*, **20**, 130–152, 1979.

50. M. Watanabe, K. Kodaira and S. Okamura, "Digital surface photometry of galaxies toward a quantitative classification. I. 20 galaxies in the Virgo cluster", *Astrophysics Journal Supplement Series.*, **50**, 1–22, 1982.

51. G.B. Wetherill, *Elementary Statistical Methods*, Chapman and Hall, London, 1972.

52. D. Wishart, "Mode analysis: a generalization of nearest neighbour which reduces chaining effects", in ed. A.J. Cole, *Numerical Taxonomy*, Academic Press, New York, 282–311, 1969.

53. D. York, "Least squares fitting of a straight line", *Canadian Journal of Physics* **44**, 1079–1086, 1966.

54. C.T. Zahn, "Graph–theoretical methods for detecting and describing Gestalt clusters", *IEEE Transactions on Computers*, **C–20**, 68–86, 1971.