MAS465: Multivariate Data Analysis
MAS6011: Dependent Data (semester 1)

Frazer Jarvis
J12 Hicks Building
`A.F.Jarvis@shef.ac.uk`

Autumn 2016-17

Contents

Introduction

The aim of this course is to survey the field of Multivariate Data Analysis. When observations are made, it is rare in practice that only one piece of data is stored. Generally, observations are "multi-dimensional", i.e., one observation consists of several pieces of data. Analysing this multi-dimensional, or *multivariate*, data has challenges of its own, and we will explore several of these in this module.

As with any analysis of data, there are two sides to the analysis of multivariate data, although in practice the distinction may not be completely clear-cut.

Given a mass of data, the first task of the data analyst is to explore it from as many angles as possible. Data visualisation tends to be difficult in the multivariate case, as the number of dimensions required usually exceeds 3. We will start by developing some techniques for the visualisation of multivariate data, before moving on to some of the main techniques, involving reduction of dimensionality. The aim of this stage is *exploratory*; the idea is to pick out salient points in the data worthy of further investigation. Generally one uses summary tables and basic visualisations in this exploratory stage, but visualisation can be awkward if there are too many dimensions.

The nature of this exploratory analysis is rather different from much else you might have encountered in your statistics courses, in that there is little statistics involved! Partly the aim is to develop visualisation techniques; even more, the aim is to develop rather pure mathematical techniques to allow high-dimensional data sets to be viewed in a smaller number of dimensions.

With the exploratory analysis completed, one needs to check whether any interesting observations are *statistically significant*. Generally, in order to have a nice statistical model, some assumptions (normality, for example) need to be made about the distribution of the data. In the 1-dimensional (*univariate*) case, you will already be familiar with the methods involved. With a suitable assumption of normality, one can test various properties of the data against hypotheses by evaluating *test statistics* (χ^2, etc.). By the Central Limit Theorem, features of a large number of observations should be expected to resemble the corresponding features from normal distributions; on this assumption, one can develop test statistics and hypothesis tests for the mean more generally.

In the multivariate case, there is a good generalisation of the normal distribution, called (unsurprisingly) the *multivariate normal distribution*, and we can begin to ask similar questions in the multivariate case to those in the univariate case, developing statistics, and hypothesis testing, in a similar way.

There will be little new statistics in this module; rather, we will develop similar tests to those you already know, but in the multivariate case.

Brief outline

The course is concerned with analysing and interpreting multivariate data. Generally when observations are made, more than one piece of data is stored. We will generally use p to denote the number of pieces of data stored, and n for the number of observations taken; p is the *dimension* of the data. That is, we are performing measurement of p variables on each of n subjects.

Examples might include:

1. body temperature, renal function, blood presure, weight of 73 hypertensive subjects ($p = 4$, $n = 73$)
2. petal and sepal length and width of 150 flowers ($p = 4$, $n = 150$)
3. amounts of 9 trace elements in clay of Ancient Greek pottery fragments ($p = 9$)
4. amounts of 18 amino acids in fingernails of 92 arthritic subjects ($p = 18$, $n = 92$)
5. presence or absence of each of 286 decorative motifs on 148 bronze age vessels in North Yorkshire ($p = 286$, $n = 148$)
6. grey shade levels of each of 1024 pixels in each of 15 digitized images ($p = 1024$, $n = 15$)
7. digitization of a spectrum ($p = 10000$, $n = 100$ is typical)
8. activation levels of all genes on a genome ($p = 30000$ genes, $n = 10$ microarrays is typical)

   Note that

- measurements can be *discrete* or *continuous* (or even *Boolean*, i.e., true/false), or a mixture of both.
- typically the variables are *correlated* but individual sets of observations are *independent.*
- there may be more observations than variables, or vice versa.
- some multivariate techniques are only available when $n > p$ (e.g., discriminant analysis, formal testing of parametric hypotheses etc.). Other techniques can be used even if $n < p$ (e.g., Principal Component Analysis, Cluster Analysis).

  (Generally, this will be because some techniques will require certain $n \times p$ matrices to have rank p.)

Subject matter – some multivariate problems

There are a number of questions we seek to answer in multivariate data analysis. Here are some:

1. *Reduction of dimensionality.* Visualising and interpreting data in may dimensions is not easy, and we seek to represent the data as simply as possible, generally in fewer dimensions, without sacrificing valuable information, to ease the problem of interpreting the data. This is generally part of the exploratory analysis, and simplifies the structure of the data by, for example, ignoring variables which are highly correlated with others.

2. *Cluster analysis/classification.* Do data arise from a homogeneous source, or do they come from a variety of sources, e.g., does a medical condition have subvariants? Thus one natural question is to ask if the observations form groups, or *clusters*, or if they are more randomly scattered. Techniques for forming groups are called *cluster analysis.*

3. *Sorting and grouping.* We try to group "similar" objects or variables, based upon measured characteristics, finding rules for discriminating between groups (e.g., two known variants of a disease). This is a different type of classification problem to the last, and is called *discriminant analysis.* Here, we have a random sample from

two (or more) populations, and try to set up a rule to allocate new observations to the correct population with the greatest probability of being correct. If we can find the best rules, this allows diagnostic tests to classify future cases, and may also throw extra light on the data (e.g., in amino acids and arthritis example, there are two types of arthritis, psoriatic and rheumatoid; determining which combinations of amino acids best distinguishes them gives information on the biochemical differences between the two conditions).

4. *Scaling* This technique is appropriate when the data arises in the form of similarities or dissimilarities between individuals. The objective is to produce a “map” of the individuals in a small number of dimensions to allow easy visualisation and comparisons.

5. *Dependencies among variables.* Are all the variables mutually independent or are one or more variables dependent on the others? If there are relationships between variables, one can predict the values of one or more variables on the basis of observations of other variables.

6. *Hypothesis construction and testing.* There are (mostly) obvious generalisations of univariate problems: t-tests, analysis of variance, regression, multivariate general linear model (e.g., modelling data by $Y' = X\Theta + \epsilon$, where Y' is the $n \times p$ data matrix, X is an $n \times k$ matrix of known observations of k-dimensional regressor variables, Θ is the $k \times p$ matrix of unknown parameters, and ϵ is $n \times p$ with n values of p-dimensional error variables).

Broadly speaking, the first four of these are data-analytic techniques, where the analyst tries to find patterns in the given data (this is also called *data mining*). The last two are statistical in nature, and try to confirm the patterns observed. (However, there is scope for more statistical methods in some of the data-analytic techniques too, as we shall see.)

After an initial review of methods for displaying multivariate data graphically (“data visualisation”, or “datavis”), this course will begin with topics from the first category (dimensionality reduction) before considering other non-statistical methods and finally the estimation and testing problems which rest on distributional assumptions and which are straightforward generalisations of univariate statistical techniques.

Generally, we want to make decisions based on data which is hard to visualise and has sufficiently many observations that the amount of data is hard to handle. It should not be a surprise that increasingly methods from data science and machine learning are supplementing more “traditional” statistical techniques. The topic of multivariate data analysis was already substantial, but the enormous current work on data science and machine learning makes this field extremely active, and fast-moving. Inevitably, we can only address a small subset of the material, and must refer the reader to literature on machine learning for topics such as neural networks, self-organising maps and random forests. (I'd love to learn and talk about these in more detail too!)

R and other computer packages

The statistical package for this course is R. It is very similar to the copyright package S-PLUS and the command line commands of S-PLUS are (almost) interchangeable with those of R. A brief guide to getting started in R is available from the course homepage.

R is free and may be downloaded from `http://cran.r-project.org/`. You are strongly advised to have access to a computer with a copy of R installed. The commands given in these notes were used with R, version 3.0.0, released in April 2013; the version of R currently (September 2016) is 3.3.1, and most of the scripts in the module have been checked with the newest version.

Typing `help()` with any given function name in the parentheses will give you a useful summary of how it works.

R has a large number of libraries contributed by users to perform many sophisticated analyses. These extend the capabilities substantially. If you are considering using multivariate techniques outside this course, you would be well advised to use R.

As well as R and S-PLUS, there are other packages used in the "real world", such as MINITAB. Of course, one can do quite a lot with Excel also.

Those of you proficient in computing may also like to prepare a data analysis suite around the programming language Python, with modules `numpy` for the numerical analysis, `pandas` for the data analysis, and `matplotlib` for the graphical output. See `https://store.continuum.io/cshop/anaconda/` for a free distribution aimed at the data-analyst, including all of these Python modules (and many more).

Many further modules are in development for both R and Python which will assist data analysts more, especially with a view to applications in the field of "big data". R does not really scale all that well at the moment to really large data sets, although no doubt this will change; in any case, it is possible to integrate it within Hadoop, the open-source implementation of Google's `MapReduce` protocol for distributing very large tasks between many computers. The same is possible for Python.

We shall say almost nothing about these other options, beyond mentioning that they exist. Data scientists are fairly evenly split between Python and R, with other languages being less popular, but statisticians seem to be nearly universally working with R, at least in the academic world.

For the purposes of the MAS465 and MAS6011 examinations, students need to be familiar only with R output.

R libraries required

Most of the statistical analyses described use functions within the `base` and `stats` packages, and the `MASS` package (`MASS` is *Modern Applied Statistics with S*, the title of a book by Venables and Ripley). It is recommended that each R session should start with

```
library(MASS)
```

The `MASS` library is installed with the base system of R and the `stats` package is automatically loaded. Occasionally these notes may mention other packages.

As usual, additional packages can be installed by `install.packages("package")`, etc. In each R session you want to use (say) `package`, type `library(package)` at the prompt.

Books for the course

Don't buy any books! Books are expensive, even if you find used copies at Amazon or ABEbooks etc.

These notes are closely based on those of the previous lecturer, Dr Nick Fieller. I am told that he has recently signed a contract to turn his notes into a book, so that will probably be the most appropriate book for the module when it is published (unlikely to be before 2017!). He, in turn, based the content of the module on the books:

- **R.Gnanadesikan**, *Methods for Statistical Data Analysis of Multivariate Observations*, 2nd., ed., Wiley (1997)
- **K.Mardia, J.Kent, J.Bibby**, *Multivariate Analysis*, Wiley (1981)

He also suggested the books:

- **T.Cox**, *An Introduction to Multivariate Analysis*, Arnold (2005)
- **B.Everitt**, *An R and* S-PLUS *Companion to Multvariate Analysis*, Springer (2005), see also `http://biostatistics.iop.kcl.ac.uk/publications/everitt`

These two texts cover most of the material in the course, and are modestly priced. Another similar text, mentioned to me by someone else, and which covers the course material quite closely, is

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

Indeed, I've found this book, and that of Cox, to be very useful.

The previous lecturer also recommended

- **B.Manly**, *Multivariate statistical methods: a primer*, 3rd. ed., Chapman & Hall (2004)

as an excellent introduction to many of the techniques in this course, and is the source of some of his examples.

The next book is a rather substantial text, but I found it useful when preparing the course:

- **R.Johnson, D.Wichern**, *Applied Multivariate Statistical Analysis*, 6th. ed., Pearson (2007)

Finally, I'll mention some more advanced books. The first is now something of a classic, and will be of interest for further reading (perhaps especially for MSc students):

- **W.Venables, B.Ripley**, *Modern Applied Statistics with S*, 4th. ed., Springer (2002), support material available at `http://www.stats.ox.ac.uk/pub/MASS4`

This treats recently developed techniques widely used in industry, and which may be of use in other courses involving project work.

For more theoretical material, the best book is probably

- **T.Anderson**, *An Introduction to Multivariate Statistical Analysis*, 3rd. ed., Wiley (2003)

but we won't get far enough in this module to get beyond the theory developed in the books listed earlier.

For a more advanced introduction to all the techniques in this module (and many others), see the second book below – the first is intended as an introduction to the second:

- **G.James et al.**, *An Introduction to Statistical Learning*, Springer (2013), a download is at `http://web.stanford.edu/~hastie/pub.htm`
- **T.Hastie et al.**, *The Elements of Statistical Learning*, 2nd. ed., Springer (2011), a download is at `http://web.stanford.edu/~hastie/pub.htm`

There are many other texts on specific topics: pattern recognition, neural networks, image processing, data mining, etc.; all may contain useful material.

R graphics and data visualisation

R comes equipped with integrated graphics. These are generally sufficient for users' needs, and we shall use nothing more in this module.

However, there are additional libraries contributed by others which enhance graphical capabilities further; the book *R Graphics* by Paul Murrell (Chapman & Hall, 2nd. ed., 2011) is a good starting point. See also *R Graphics Cookbook* by Winston Chang (O'Reilly, 2012).

The package `lattice` is particularly designed for multivariate analysis, and the author, Deephayan Sarkar, has written an accompanying book *Lattice: Multivariate Data Visualisation with R* (Springer, 2008).

Overall, though, I would recommend `ggplot2`, by Hadley Wickham; he has written an accompanying book *ggplot2: Elegant Graphics for Data Analysis* (Springer, 2009, 2nd ed. 2016). See also `http://ggplot2.org/`. There's a nice introduction to `ggplot2` in *Machine Learning for Hackers* by Drew Conway and John Myles White (O'Reilly, 2012). The prefix `gg-` stands for "Grammar of Graphics", a syntax developed by Leland Wilkinson to describe any static statistical graphic. Recently, Wickham and Winston Chang have been working on `ggvis` and `shiny`, which extends `ggplot2` to cover dynamic statistical graphics (i.e., where there is some element of user interaction). I recommend the reader look at

`http://www.rstudio.com/products/rpackages/`

for many useful R packages, not just for data visualisation. The field of data visualisation seems to be moving very fast, with new packages appearing very frequently (Python is similar, but R seems to be nearer to stabilising).

It seems that the professional data analyst's workflow tends more and more to use web-based interactive graphics (not R), which can be produced with the Javascript D3 library (see `http://d3js.org/`) and the book by Scott Murray, *Interactive Data Visualisation for the Web* (O'Reilly, 2013) is a great introduction.

Again, we won't say anything further about these options.

1 Preliminary mathematical material

This chapter is not examinable explicitly, and I shall only lecture small parts of it; I will, however, draw on this material later in the module. If there's anything that you aren't sure about, you might like to revise it now! Some proofs have been included for completeness, but you will never need to know these – the results may be used later, however. The main point of the chapter is to introduce the notation that we'll be using throughout the module.

1.1 Basic notation

We deal with *observations* $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix}$, a column p-vector. That is, each observation consists of p numbers.

It will generally be convenient to store the observations in a matrix. Not only is this a convenient form to store the data, but matrix methods are going to play an important role in the module; manipulating the data will be easiest if it is already in matrix form.

The ith observation is $x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}$.

If x is a column vector, write x' for the transpose of x; it is a row vector. We use the same notation for matrices.

Form the $p \times n$ matrix X whose columns are the x_i. This matrix contains all the data. However, there is confusion throughout sources between whether to treat X or its transpose, the $n \times p$ matrix X', as the "data matrix" (obviously it doesn't really matter, but we do need to fix a choice so that our formulae work!). Partly because of the way R stores the information, as a *data frame*, we will refer to X' as the *data matrix*. Thus the data matrix is the $n \times p$ matrix:

$$X' = \begin{pmatrix} x_{11} & \dots & x_{1p} \\ x_{21} & \dots & x_{2p} \\ \vdots & \vdots & \vdots \\ x_{n1} & \dots & x_{np} \end{pmatrix},$$

in which each row consists of a multivariate observation. (As already remarked, this choice is not universal in the literature, and you should be aware of this if you consult any books!)

Note that we do not specifically indicate vectors of matrices (by underlining, bold face etc.); additionally, whether x_i denotes the ith vector observation of X, or the ith element of a vector x is dependent on context.

Define the *sample mean vector* as

$$\overline{x} = \begin{pmatrix} \overline{x}_1 \\ \vdots \\ \overline{x}_p \end{pmatrix} = \tfrac{1}{n} \begin{pmatrix} x_{11} + \cdots + x_{n1} \\ \vdots \\ x_{1p} + \cdots + x_{np} \end{pmatrix}.$$

As usual, this sample mean is a sort of "centre of gravity" of the data points.

There is a very succinct way to write $\overline{x}$: it is equal to $\frac{1}{n}X\mathbf{1}$, where $\mathbf{1}$ is the column vector of n 1s. (You should try to understand this succinct definition; the easier you find this sort of matrix manipulation, the easier the course will become!) Using standard results on transposes, we can write this equivalently as

$$\overline{x}' = \tfrac{1}{n}\mathbf{1}'X',$$

in terms of the data matrix X'. After all, we are really interested in the sample mean of the data matrix, i.e., the mean for each variable, and X' has one column for each variable; the sample mean ought to as well.

We also define the *(sample) variance* (or *variance-covariance matrix* or *covariance matrix*) as

$$S = \operatorname{var}(X') = \tfrac{1}{n-1}(X - \overline{X})(X - \overline{X})', \tag{1.1}$$

where $\overline{X} = (\overline{x}, \dots, \overline{x})$ is the $p \times n$ matrix with all columns equal to $\overline{x}$. (Again, some authors use $\frac{1}{n}$ in place of $\frac{1}{n-1}$; the justification for using $\frac{1}{n-1}$ is the same as in the univariate case, as we shall explain later.)

Exercise 1.1 Use (1.1) to check that

$$s_{ij} = \frac{1}{n-1}\sum_{k=1}^{n}(x_{ki} - \overline{x}_i)(x_{kj} - \overline{x}_j).$$

Notice that S is a $p \times p$ matrix whose diagonal entry S_{ii} is the *sample variance* of the ith variable, and the entry S_{ij} for $i \neq j$ is the *sample covariance* between the ith and jth variables. In particular, S is symmetric (this should be clear from (1.1))

Provided none of the variables is a linear combination of any of the others (so the columns are *linearly independent*), S will be non-singular (i.e., invertible, or equivalently have non-zero determinant) and will be positive definite.

We will also use the *sample correlation matrix* R, which is simply the same as the variance matrix, but with the entries scaled so that

$$r_{ij} = \frac{s_{ij}}{\sqrt{s_{ii}s_{jj}}}. \tag{1.2}$$

If L denotes the diagonal matrix whose entries are $s_{11}, \dots, s_{pp}$, and $L^{1/2}$ is then the diagonal matrix with entries $\sqrt{s_{11}}, \dots, \sqrt{s_{pp}}$, then

$$S = L^{1/2}RL^{1/2}.$$

1.2 More about the sample variance and correlation matrices

S can also be written as

$$\begin{aligned} S &= \tfrac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})(x_i - \overline{x})' \\ &= \tfrac{1}{n-1}\left(\sum_{i=1}^{n} x_i x_i' - n\overline{x}\overline{x}'\right). \end{aligned}$$

For the first equality, note that $X - \overline{X}$ is a matrix with columns $b_i = x_i - \overline{x}$, and so $B = X - \overline{X}$ has columns $(b_1, \ldots, b_n)$, and $BB' = \sum_{i=1}^n b_i b_i'$.

The second equality follows directly from multiplying out the terms in the summation sign (keeping the ordering the same, and taking the transpose inside the brackets, noting that $\overline{x}$ is a constant, and the sum of the x_i in $n\overline{x}$).

Let's quickly make some comments on the correlation matrix R. From the formula (1.2), it is easy to see:

1. R is a symmetric $p \times p$ matrix;
2. $r_{ii} = 1$ for all i;
3. $-1 \leq r_{ij} \leq 1$ for all i, j. As usual, if $r_{ij} > 0$, then there is a tendency for variables i and j to be larger at the same time, whereas if $r_{ij} < 0$, there is a tendency for variable j to be smaller when variable i is larger, and vice versa. Geometrically, r_{ij} is the cosine of the angle between the vectors of deviations of observations of the ith and jth variables from the mean.

Below, we will develop a number of results about symmetric matrices, which will apply to both S and R.

First, here is a result which justifies the use of the scalar factor $\frac{1}{n-1}$ (some texts use $\frac{1}{n}$):

Proposition 1.2 *The sample mean and variance of independent observations are unbiased estimates for the population mean and variance.*

Proof. This is a direct generalisation of the standard proof in the univariate case. So we suppose that the data is generated as i.i.d. random variables with mean μ (a p-vector) and variance Σ (a $p \times p$-matrix). Then

$$\begin{aligned} E(\overline{x}) &= E(\tfrac{1}{n}x_1 + \cdots + \tfrac{1}{n}x_n) \\ &= \tfrac{1}{n}E(x_1) + \cdots + \tfrac{1}{n}E(x_n) \\ &= \tfrac{1}{n}\mu + \cdots + \tfrac{1}{n}\mu \\ &= \mu. \end{aligned}$$

For the variance, we again argue as in the univariate case. Firstly, we see that if x_i is a vector with $E(x_i) = \mu$ and variance $\Sigma = E((x_i - \mu)(x_i - \mu)')$, then multiplying this out gives $E(x_i x_i') - \mu\mu' = \Sigma$, and so

$$E(x_i x_i') = \mu\mu' + \Sigma. \tag{1.3}$$

In particular, the $\overline{x} = \frac{1}{n}(x_1 + \cdots + x_n)$ also has expected value μ, and one easily shows that $E((\overline{x} - \mu)(\overline{x} - \mu)') = \frac{1}{n}\Sigma$, so we deduce that

$$E(\overline{x}\overline{x}') = \mu\mu' + \tfrac{1}{n}\Sigma. \tag{1.4}$$

Now

$$S = \frac{1}{n-1}\left(\sum_{i=1}^n x_i x_i' - n\overline{x}\overline{x}'\right),$$

so

$$\begin{aligned}
E(S) &= \frac{1}{n-1}\left(\sum_{i=1}^{n} E(x_i x_i') - nE(\overline{x}\overline{x}')\right) \\
&= \frac{1}{n-1}\left(n\left(\mu\mu' + \Sigma\right) - n\left(\mu\mu' + \frac{1}{n}\Sigma\right)\right) \\
&= \frac{1}{n-1}(n-1)\Sigma \\
&= \Sigma,
\end{aligned}$$

as required. □

Finally, we list a couple of simple (but useful) results of matrix manipulations. We will be applying matrix operations freely, and will need this sort of result often.

Lemma 1.3 *If w is any vector, then* $\text{var}(X'w) = w'\text{var}(X')w = w'Sw$.

Proof. Write $Y' = X'w$. This follows directly from the expression for $\text{var}(Y)$, putting $y_i = w'x_i$ etc.:

$$\begin{aligned}
\text{var}(Y') &= \tfrac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{Y})(y_i - \overline{Y})' \\
&= \tfrac{1}{n-1}\sum_{i=1}^{n}(w'x_i - w'\overline{X})(w'x_i - w'\overline{X})' \\
&= \tfrac{1}{n-1}\sum_{i=1}^{n} w'(x_i - \overline{X})(x_i - \overline{X})'w \\
&= w'Sw.
\end{aligned}$$

□

This will be useful when we consider (for example) variances of linear combinations of variables. Notice that $X'w$ is a linear combination of the variables; this shows how the variance matrix of the linear combination (a scalar) is related to the variance matrix of the whole dataset (a $p \times p$-matrix).

Lemma 1.4 *If A is any $p \times q$ matrix, then* $\text{var}(X'A) = A'\text{var}(X')A = A'SA$.

Proof. Similar to the last result. □

This will be useful when we consider (for example) certain linear transformations of variables.

1.3 Basic properties of eigenvectors and eigenvalues

Let A be a real $p \times p$ matrix.

Definition 1.5 The *eigenvalues* of A are the roots of the degree p polynomial (the *characteristic polynomial*) in λ given by $q(\lambda) = \det(A - \lambda I_p) = 0$.

Remark 1.6 Suppose these eigenvalues are $\lambda_1, \ldots, \lambda_p$. Then $q(\lambda) = \prod_i(\lambda_i - \lambda)$. Comparing the coefficients of λ^{p-1} in these two expressions for $q(\lambda)$ gives

$$\sum_{i=1}^{p} \lambda_i = \mathrm{tr}(A).$$

Further, putting $\lambda = 0$ gives $\prod_{i=1}^{p} \lambda_i = \det(A)$.

Definition 1.7 Suppose that λ_i is an eigenvalue of A. Since $A - \lambda_i I_p$ is singular (i.e., has zero determinant), there is (at least one) vector x_i, called an *eigenvector* of A, such that $(A - \lambda_i)x_i = 0$, i.e., $Ax_i = \lambda_i x_i$.

Generally, if λ_i is an eigenvalue of A with multiplicity m_i, the maximal number of linearly independent eigenvectors of A can be anywhere between 1 and m_i. In statistical applications, this is generally only an issue for the eigenvalue $\lambda = 0$ (for example, if there are not enough observations that the $p \times p$ matrix arising can possibly have rank p).

The following fact is rather useful:

Fact 1.8 If A is a real symmetric $p \times p$ matrix, then there are always p linearly independent eigenvectors.

This is a well-known result from matrix theory. Much more is known, and we'll derive what we need in the next page or so.

We will often want to transform our variables by means of some linear transformation, and will need to know how eigenvalues and eigenvectors of the variance matrix (for example) are affected by this. As we shall see, transforming the variables by a matrix C leads to a conjugation of the sample variance matrix by C.

Proposition 1.9 *If C is any $p \times p$ nonsingular square matrix, then A and CAC^{-1} have the same eigenvalues. Furthermore, if x_i is an eigenvector of A with eigenvalue λ_i, then Cx_i is an eigenvector of CAC^{-1} with eigenvalue λ_i.*

Proof. The first statement follows from the equality $\det(A - \lambda I_p) = \det(CAC^{-1} - \lambda I_p)$. For this, recall that $\det(AB) = \det(A)\det(B)$, and, in particular,

$$\begin{aligned}
\det(CAC^{-1} - \lambda I_p) &= \det(CAC^{-1} - \lambda CC^{-1}) \\
&= \det(C(A - \lambda I_p)C^{-1}) \\
&= \det(C)\det(A - \lambda I_p)\det(C^{-1}) \\
&= \det(A - \lambda I_p).
\end{aligned}$$

If $Ax_i = \lambda_i x_i$, then $(CAC^{-1})(Cx_i) = CAx_i = C(\lambda_i x_i) = \lambda_i(Cx_i)$, so the eigenvectors of CAC^{-1} are Cx_i. □

We have already stated one useful result about real symmetric matrices. Here is another.

Proposition 1.10 *Suppose that A is a real symmetric matrix, i.e., $A = A'$. Then the eigenvalues of A are real.*

Proof. Indeed, if $Ax = \lambda x$, then $\overline{x}'Ax = \lambda\overline{x}'x$; applying transposition, and taking complex conjugation implies that also $\overline{x}'Ax = \overline{\lambda}\overline{x}'x$. As $x \neq 0$, we get $\lambda = \overline{\lambda}$. □

In the same way that we can measure lengths of vectors in $\mathbb{R}^2$ and $\mathbb{R}^3$ using (essentially) Pythagoras's Theorem, giving $\|(a, b)\| = \sqrt{a^2 + b^2}$, there is an analogous statement in $\mathbb{R}^n$:

$$\|(x_1, \ldots, x_n)'\| = \sqrt{x_1^2 + \cdots + x_n^2}.$$

In fact, this extends to the notion of an *inner product* on $\mathbb{R}^n$:

$$\langle x, y\rangle = x_1y_1 + \cdots + x_ny_n,$$

so that $\|x\| = \sqrt{\langle x, x\rangle}$. The notation $\langle\cdot,\cdot\rangle$ is traditional in pure mathematics, but statisticians tend not to use it – instead, notice that if $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ and $y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$, then

$$x'y = x_1y_1 + \cdots + x_ny_n;$$

we shall just use the notation $x'y$ in this module.

Just as the existence of the dot product in $\mathbb{R}^3$ allows one to define angles etc., between vectors, the same holds in $\mathbb{R}^n$; in particular, we say that two vectors x, y are *orthogonal* if $x'y = 0$. The last main result we need about real symmetric matrices is the following:

Proposition 1.11 *Suppose that A is a real symmetric matrix. If λ and μ are distinct eigenvalues and x and y are corresponding eigenvectors, then x and y are orthogonal.*

Proof. Indeed, we have $Ax = \lambda x$ and $Ay = \mu y$. So $y'Ax = \lambda y'x$ and $x'Ay = \mu x'y$. Taking transposes of the latter gives $y'Ax = \mu y'x$. As $\lambda \neq \mu$, we get $y'x = 0$. □

Let's summarise the main properties of real symmetric matrices:

1. If A denotes a real symmetric $p \times p$ matrix, then A has p linearly independent eigenvectors.
2. All the eigenvalues of A are real.
3. Eigenvectors corresponding to distinct eigenvalues are orthogonal.

The (sample) variance matrix will be the main tool for many of our techniques, and the properties above will often be important.

Corollary 1.12 *If A is a real symmetric $p \times p$ matrix, and if X denotes the matrix whose columns are normalised eigenvectors of A, then X is an orthogonal matrix (i.e. $XX' = I$, or $X' = X^{-1}$).*

Proof. This simply follows from the observation that the columns of X have length 1, and are orthogonal. □

Recall that if A is any $p \times p$ matrix with p linearly independent eigenvectors (e.g., any matrix with p distinct eigenvalues, or any real symmetric matrix), then if X denotes the matrix whose columns are eigenvectors of A, then $X^{-1}AX = \Lambda$, where Λ is the diagonal matrix whose diagonal entries are the eigenvalues. (Indeed, this is easily verified by considering how matrix multiplication works, and computing AX; the product is evaluated by multiplying A by each column in turn, and since each column is an eigenvector, we easily find $AX = X\Lambda$.)

The decomposition $X^{-1}AX = \Lambda$ is known as the *spectral decomposition*, and can easily be seen to be equivalent to

$$A = \lambda_1 x_1 x_1' + \cdots + \lambda_p x_p x_p',$$

where x_i is an eigenvector of A with eigenvalue λ_i.

Corollary 1.13 *Positive definite real symmetric matrices (i.e., real symmetric matrices with positive eigenvalues) possess positive definite real symmetric square roots.*

Proof. If A is a positive definite real symmetric $p \times p$ matrix with orthogonal eigenvectors $x_1, \dots, x_p$ corresponding to eigenvalues $\lambda_1, \dots, \lambda_p$, we know that $X^{-1}AX = \Lambda$, i.e., that $A = X\Lambda X^{-1}$. Then we pick $\Lambda^{1/2}$ to be the diagonal matrix whose entries are the positive square roots of each λ_i. Put $B = X\Lambda^{1/2}X^{-1}$; then it is easy to see that $A = B^2$. Since $X' = X^{-1}$ (if we take the eigenvectors to be normalised), we can also write $B = X\Lambda^{1/2}X'$. □

We can apply this because of the following result:

Lemma 1.14 *The eigenvalues of the variance matrix S are non-negative.*

Proof. Suppose λ is an eigenvalue of S, with corresponding eigenvector x. Then $Sx = \lambda x$. It follows that $x'Sx = \lambda x'x$; it is easy to see that $x'Sx$ is a non-negative scalar (why?), and that $x'x$ is a positive scalar. □

Summarising all the discussion so far, the variance matrix has a unique positive definite symmetric square root. We won't need this often.

1.4 Differentiating with respect to vectors

As you will have gathered already, many techniques in the module involve matrices and vectors. Often we will be interested in the best choice of vector to optimise some function. Classically, you know that to optimise a function of one variable, we try to differentiate it, and set the result to zero. We will need something similar for vectors.

Suppose that $x = (x_1 \ \dots \ x_p)'$ is a p-vector. We say that $f = f(x)$ is a *scalar function* of x if it is a function depending on x, but which produces a number as the output. For example, we could take $f(x) = x'x$, the square length of x.

Then we define the *derivative* of f with respect to x, written $\frac{\partial f}{\partial x}$, as the vector $(\frac{\partial f}{\partial x_1} \ \dots \ \frac{\partial f}{\partial x_p})'$ (assuming all these partial derivatives exist).

The most useful special case is given by the following result.

Lemma 1.15 *Suppose that S is a symmetric $p \times p$-matrix, and that $f(x) = x'Sx$. Then $\frac{\partial f}{\partial x} = 2Sx$.*

Proof. Write $S = (s_{ij})$. Then

$$f(x) = x'Sx = \sum_{i=1}^{n} \sum_{j=1}^{n} s_{ij} x_i x_j. \tag{1.5}$$

We can work out $\frac{\partial f}{\partial x_k}$ by the product rule. We'll consider 4 cases, depending whether i or j is equal to k.

- If $i \neq k$, and $j \neq k$, then the term in (1.5) coming from i and j differentiates to 0:
$$\frac{\partial(s_{ij}x_ix_j)}{\partial x_k} = 0.$$
- If $i = k$, and $j \neq k$, then the term in (1.5) coming from i and j differentiates as:
$$\frac{\partial(s_{kj}x_kx_j)}{\partial x_k} = s_{kj}x_j.$$
- If $i \neq k$, and $j = k$, then the term in (1.5) coming from i and j differentiates as:
$$\frac{\partial(s_{ik}x_ix_k)}{\partial x_k} = s_{ik}x_i.$$
- If $i = j = k$, then the term in (1.5) coming from i and j differentiates as:
$$\frac{\partial(s_{kk}x_k^2)}{\partial x_k} = 2s_{kk}x_k.$$

Thus
$$\begin{aligned}\frac{\partial(x'Sx)}{\partial x_k} &= \sum_{j\neq k} s_{kj}x_j + \sum_{i\neq k} s_{ik}x_i + 2s_{kk}x_k \\ &= \sum_{j\neq k} s_{kj}x_j + s_{kk}x_k + \sum_{i\neq k} s_{ik}x_i + s_{kk}x_k \\ &= \sum_{j=1}^{n} s_{kj}x_j + \sum_{i=1}^{n} s_{ik}x_i \\ &= \sum_{j=1}^{n} s_{kj}x_j + \sum_{i=1}^{n} s_{ki}x_i \\ &= 2\sum_{j=1}^{n} s_{kj}x_j \\ &= 2(Sx)_k.\end{aligned}$$

Combining these gives the result. □

(The same method shows that if S is not symmetric, then $\frac{\partial(x'Sx)}{\partial x} = (S + S')x$.) As a special case, if $S = I$, then $\frac{\partial(x'x)}{\partial x} = 2x$.

Another useful example, rather easier than the one above is the following (you could treat this as an exercise). If a is a constant p-vector, and $f(x) = a'x$, then $\frac{\partial f}{\partial x} = a$.

1.5 Constrained optimisation

For several of the techniques in the first part of the module, we will need to perform a *constrained optimisation*. That is, we will need to work out the maximum value of a function *under some extra conditions*.

Suppose $x = (x_1, \dots, x_n)'$. To maximise $f(x)$ (a scalar function of x) subject to k scalar constraints $g_1(x) = 0, \dots, g_k(x) = 0$ with $k < n$, we use *Lagrange multipliers* $\lambda_1, \dots, \lambda_k$. Define $\Omega = f(x) + \sum_{j=1}^{k} \lambda_j g_j(x)$, and maximise/minimise Ω with respect to the $n + k$ variables $x_1, \dots, x_n, \lambda_1, \dots, \lambda_k$.

Example 1.16 If we are asked to minimise $x_1^2 + x_2^2$, subject to $x_1 + x_2 = 1$, we write $\Omega = x_1^2 + x_2^2 + \lambda(x_1 + x_2 - 1)$. Then $\frac{\partial\Omega}{\partial x_1} = 2x_1 + \lambda$, $\frac{\partial\Omega}{\partial x_2} = 2x_2 + \lambda$, and $\frac{\partial\Omega}{\partial\lambda} = x_1 + x_2 - 1$. It is easy to see that the solution to all these equations is $x_1 = x_2 = \frac{1}{2}$.

Here's a more complicated statistical example:

Example 1.17 Suppose that $t_1, \dots, t_n$ are unbiased estimates of θ with variances $\sigma_1^2, \dots, \sigma_n^2$; let's try to find the best linear unbiased estimate of θ.

Take a linear combination $\tau = \sum \alpha_k t_k$. We want to choose the coefficients α_i so that τ has minimum variance subject to the constraint of being unbiased. Now $E(t_i) = \theta$ for all i, so $E(\tau) = \theta$, and so we have the constraint $\sum \alpha_k = 1$. Also, $\mathrm{var}(\tau) = \sum \alpha_k^2 \sigma_k^2$. Let

$$\Omega = \sum \alpha_k^2 \sigma_k^2 + \lambda \left(\sum \alpha_k - 1 \right).$$

Then

$$\frac{\partial\Omega}{\partial\alpha_i} = 2\alpha_i\sigma_i^2 + \lambda, \qquad \frac{\partial\Omega}{\partial\lambda} = \sum \alpha_k - 1.$$

Setting these to 0, the first equations give $\alpha_i = -\frac{1}{2}\frac{\lambda}{\sigma_i^2}$, and then the final one gives $\lambda = -2/\sum \frac{1}{\sigma_k^2}$. Substituting back gives

$$\alpha_i = \frac{1}{\sigma_i^2} \left(\sum \frac{1}{\sigma_k^2} \right)^{-1},$$

and the BLUE estimate of θ is

$$\hat{\theta} = \left(\sum \frac{t_k}{\sigma_k^2} \right) \Big/ \left(\sum \frac{1}{\sigma_k^2} \right).$$

1.6 The multivariate normal distribution

When we come to perform statistical tests, we will need to make some assumptions on the data. As in the univariate case, the normal distribution will play an important role:

1. it is relatively easy to work with, and can be manipulated to give useful resuts;
2. it seems to be a reasonable model quite often in practice, even if there is no obvious reason why data should be normal;
3. there is a multivariate central limit theorem, which means that the mean of a collection of i.i.d. variables looks more and more normal, so it does have a distinguished position amongst all the distributions.

We will say more about this distribution later, but it will occasionally be useful to be able to refer to it in the early chapters as well, so we mention it briefly here. (In any case, it is likely that most taking the module will have seen parts of this material before.),

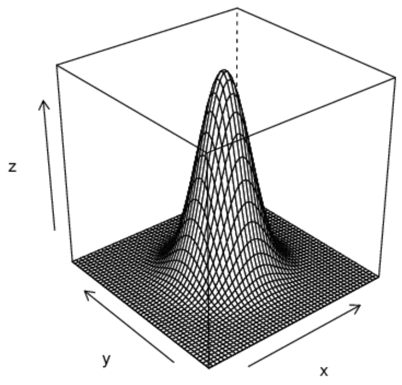
Definition 1.18 Suppose that μ is a p-vector, and that Σ is a positive definite symmetric $p \times p$-matrix. Then the *multivariate normal distribution* $N_p(\mu, \Sigma)$ is the distribution with probability density function

$$f_x(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)\right).$$

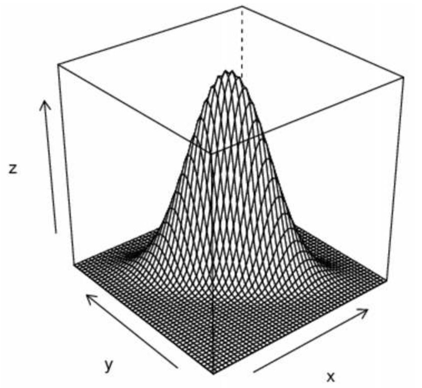
Let's first imagine that Σ is a diagonal matrix (i.e., the variables are uncorrelated). Then it is easy to see that for any positive constant c, the surface $(x-\mu)'\Sigma^{-1}(x-\mu) = c$ is an ellipsoid, centred on μ, and with axes in the direction of the co-ordinate axes.

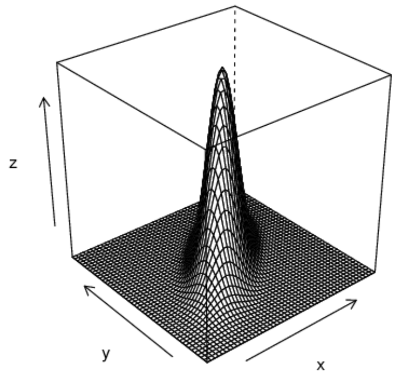
More generally, if Σ is an arbitrary (positive definite symmetric) matrix, the multivariate normal density is constant on surfaces where $(x-\mu)'\Sigma^{-1}(x-\mu)$ is constant, and these are again ellipsoids centred at μ. The axes of each ellipsoid of constant density are in the direction of the eigenvectors of Σ^{-1} (recall that these are the same as the eigenvectors of Σ, but if $\Sigma x = \lambda x$, then $\Sigma^{-1}x = \lambda^{-1}x$), and their lengths are proportional to the reciprocals of the square roots of the eigenvalues of Σ^{-1}.

Here is a picture of the distribution in the bivariate case, when $\Sigma = I_2$:



and here are two views of the distribution in the case where $\Sigma = \begin{pmatrix} 1 & 0.75 \\ 0.75 & 1 \end{pmatrix}$ – note that the elliptical horizontal cross-sections are much more pointed:





Remark 1.19 The quantity $(x-\mu)'\Sigma^{-1}(x-\mu)$ gives a notion of *squared statistical distance* of x from μ. If one component has a much larger variance than others, it will contribute less to the distance. Moreover, two highly correlated random variables will contribute less than two variables that are nearly uncorrelated. Often this, and related quantities, are named after *Mahalanobis*, the statistician who remarked on its importance.

Lemma 1.20 *If $X \sim N_p(\mu, \Sigma)$, then the mean of X is μ and the variance is Σ.*

Proof. Suppose $x \sim N_p(\mu, \Sigma)$ and $y = \Sigma^{-1/2}(x - \mu)$, where $\Sigma^{-1/2}$ is as defined earlier, then $(x-\mu)'\Sigma^{-1}(x-\mu) = y'y = \sum_{i=1}^p y_i^2$. Now the density of y is

$$f_y(y) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}y'y\right).J_{xy},$$

where J_{xy} is the Jacobian of the transformation given by $|\frac{dx}{dy}|$, where $\frac{dx}{dy}$ is the $p \times p$ matrix with ijth element $\frac{\partial x_i}{\partial y_j}$.

Now $y = \Sigma^{-1/2}(x-\mu)$, so $x = \Sigma^{1/2}y + \mu$, so $\frac{dx}{dy} = \Sigma^{1/2}$. Thus $J_{xy} = |\Sigma|^{1/2}$, giving

$$f_y(y) = \frac{1}{(2\pi)^{p/2}} \exp\left(-\frac{1}{2}y'y\right).$$

Thus the y_i are independent univariate $N(0,1)$. (Notice therefore that $f_y(y) > 0$, integrates to 1, and so is a genuine pdf, so that $f_x(x)$ is also.)

Now if $x \sim N_p(\mu, \Sigma)$, and $y = \Sigma^{-1/2}(x-\mu)$, then

$$E(y) = \Sigma^{-1/2}(E(x) - \mu), \qquad \text{var}(y) = \Sigma^{-1/2}\text{var}(x)\Sigma^{-1/2}.$$

As $E(y) = 0$ and $\text{var}(y) = I_p$, we see that $E(x) = \mu$ and $\text{var}(x) = \Sigma$. □

We list some properties of the multivariate normal distribution, without proof.

1. Suppose that $X \sim N_p(\mu, \Sigma)$, and that w is a p-vector. Then the linear combination $w'X \sim N(w'\mu, w'\Sigma w)$.

2. Suppose that $X \sim N_p(\mu, \Sigma)$, and that A is a $q \times p$-matrix. Then $AX \sim N_q(A\mu, A\Sigma A')$.

3. If $X \sim N_p(\mu, \Sigma)$, and Y denotes a subset of variables of size q, then $Y \sim N_q(\mu_Y, \Sigma_Y)$, where μ_Y denotes the q-vector formed by taking the entries of μ corresponding to Y, and Σ_Y is the $q \times q$-matrix formed from Σ in the same way.

4. If $X \sim N_p(\mu_X, \Sigma_X)$ and $Y \sim N_q(\mu_Y, \Sigma_Y)$, then the $p+q$-vector
$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim N_{p+q}\left(\begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \begin{pmatrix} \Sigma_X & 0 \\ 0 & \Sigma_Y \end{pmatrix}\right)$$
as long as X and Y are independent.

5. If
$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim N_{p+q}\left(\begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \begin{pmatrix} \Sigma_X & \Sigma \\ \Sigma' & \Sigma_Y \end{pmatrix}\right)$$
then X and Y are independent if and only if $\Sigma = 0$.

6. If $X \sim N_p(\mu, \Sigma)$, then
   (a) $(X-\mu)'\Sigma^{-1}(X-\mu)$ is distributed as χ_p^2, the chi-squared distribution with p degrees of freedom;
   (b) In particular, the solid ellipsoid $\{x \mid (x-\mu)'\Sigma^{-1}(x-\mu) \leq \chi_p^2(\alpha)\}$ has probability $1-\alpha$.

   (This follows from the proof of Lemma 1.20.)

Recall that the *characteristic function* of a univariate probability distribution is given by $E(\exp(itx))$, which completely determines the behaviour and properties of the distribution (in fact, this is the Fourier transform of the distribution). There is a multivariate counterpart: if x is a p-dimensional distribution, then for t a vector in $\mathbb{R}^p$, we define

$$\phi_x(t) = E(\exp(it'x)) = \int\limits_{\mathbb{R}^p} f(x)e^{it'x}\,dx,$$

where $f(x)$ is the pdf. Note that as $t'x$ is a scalar, we can also write

$$\phi_x(t) = \int\limits_{\mathbb{R}^p} f(x)e^{\frac{1}{2}it'x+\frac{1}{2}ix't}\,dx$$

as $t'x = (t'x)' = x't$.

Proposition 1.21 *If $x \sim N_p(\mu, \Sigma)$, then*

$$\phi_x(t) = \exp\left(it'\mu - \tfrac{1}{2}t'\Sigma t\right).$$

Proof. We have

$$\begin{aligned}
\phi_x(t) &= \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \int\limits_{\mathbb{R}^p} \exp\left(-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu) + \tfrac{1}{2}it'x + \tfrac{1}{2}ix't\right) dx \\
&= \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \int\limits_{\mathbb{R}^p} \exp\left(-\frac{1}{2}(x-\mu-i\Sigma t)'\Sigma^{-1}(x-\mu-i\Sigma t) + \tfrac{1}{2}it'\mu + \tfrac{1}{2}i\mu't - \tfrac{1}{2}t'\Sigma t\right) dx \\
&= \frac{\exp(\frac{1}{2}it'\mu + \frac{1}{2}i\mu't - \frac{1}{2}t'\Sigma t)}{(2\pi)^{p/2}|\Sigma|^{1/2}} \int\limits_{\mathbb{R}^p} \exp\left(-\frac{1}{2}(x-\mu-i\Sigma t)'\Sigma^{-1}(x-\mu-i\Sigma t)\right) dx \\
&= \exp(\tfrac{1}{2}it'\mu + \tfrac{1}{2}i\mu't - \tfrac{1}{2}t'\Sigma t) \\
&= \exp(it'\mu - \tfrac{1}{2}t'\Sigma t).
\end{aligned}$$

□

Corollary 1.22 *Suppose that $x \sim N_p(\mu, \Sigma)$, and that $x_1, \ldots, x_n$ are observations of x. Define $\overline{x} = \frac{1}{n}\sum_{i=1}^n x_i$. Then*

$$\overline{x} \sim N_p(\mu, \tfrac{1}{n}\Sigma).$$

Proof. It is easy to verify the assertions about the mean and variance using standard methods (see Lemma 1.20). But the normality is less obvious. Here is an argument using characteristic functions. As $\overline{x} = \frac{x_1+\cdots+x_n}{n}$, it is easy to see that:

$$\begin{aligned}
\phi_{\overline{x}}(t) &= \prod_{i=1}^n \phi_{x_i}(t/n) \\
&= \prod_{i=1}^n \exp(it'\mu/n - \tfrac{1}{2}t'\Sigma t/n^2) \\
&= \exp(it'\mu/n - \tfrac{1}{2}t'\Sigma t/n^2)^n \\
&= \exp(it'\mu - \tfrac{1}{2}t'\Sigma t/n),
\end{aligned}$$

which is the characteristic function of $N_p(\mu, \frac{1}{2}\Sigma)$. □

In fact, this is a special case of the more general multidimensional central limit theorem, which also has an easy proof using characteristic functions:

Theorem 1.23 (Multivariate Central Limit Theorem) *Let $x_1, \ldots, x_n$ denote independent observations from any population with mean μ and finite covariance Σ. Then $\sqrt{n}(\overline{x} - \mu)$ has approximately an $N_p(0, \Sigma)$ distribution when n is large, and large relative to p.*

For completeness, we include a proof, although it is not necessary to remember it.

Proof. There is a simple proof using characteristic functions. Let $y_i = \Sigma^{-1/2}(x_i - \mu)$; then y_i has mean 0 and variance equal to I_p, the $p \times p$-identity matrix.

In general, a distribution of mean μ and variance Σ has characteristic function starting $1 + it'\mu - \frac{1}{2}t'\Sigma t$. Thus $\phi_{y_i}(t) = 1 - \frac{1}{2}t't + \cdots$, where the remaining terms are of higher order in $t't$.

Consider $z_n = \frac{1}{\sqrt{n}} \sum_{i=1}^n y_i = \frac{1}{\sqrt{n}}\Sigma^{-1/2}(\sum_{i=1}^n (x_i - \mu)) = \frac{1}{\sqrt{n}}\Sigma^{-1/2}(n\overline{x} - n\mu)$. Then

$$\begin{aligned}\phi_{z_n}(t) &= \prod_{i=1}^n \phi_{y_i}(t/\sqrt{n}) \\ &= \left(1 - \frac{t't}{2n} + \cdots\right)^n \\ &\to \exp\left(-\frac{t't}{2}\right),\end{aligned}$$

and this is the characteristic function of the $N_p(0, I_p)$ distribution. Thus $z_n = \sqrt{n}\Sigma^{-1/2}(\overline{x} - \mu) \sim N_p(0, I_p)$ for large n, and so $\sqrt{n}(\overline{x} - \mu) \sim N_p(0, \Sigma)$ for large n. □

Often one makes an assumption that data are distributed in accordance with a multivariate normal distribution in order to develop some statistical tests. Sometimes, when the sample size is large, and the test solely involves the behaviour of $\overline{x}$, we can use the Central Limit Theorem to get some approximation to mormality, and then the precise distribution is less crucial.

Generally, then, our hypothesis tests will require an assumption that the distribution of the population is (multivariate) normal, or, if the test involves only $\overline{x}$, that there are a large number of samples; then, the nearer the population is to normal, the fewer samples that will be required that the tests will be valid.

2 Data visualisation

Probably the first thing to do to make sense of a new data set is to compute simple summaries of each column. You will know that R has a `summary()` command, whose input is a vector of numbers, and whose output consists of

1. the minimum value of the vector;
2. the first quartile;
3. the median;
4. the mean;
5. the third quartile;
6. the maximum value.

In addition, the command `var` produces the variance of a set of numbers, and `sd` produces the standard deviation. For univariate data, these quantities are generally sufficient to get a good picture of the shape of the data, and it is a good first step also for multivariate data. Incidentally, the command `mean` produces an overall mean; better is to use the command `apply(dataset,2,mean)` to find the mean of each variable. Try `help(apply)` for more details: the "2" in the command here means that the `mean` operation is applied to the *columns*.

Note that in the multivariate case, `sd` and `var` do slightly different things: `sd(dataframe)^2` produces a list of the variances of all the p individual variables, while `var(dataframe)` produces the whole $p \times p$ variance matrix.

Computing numerical summaries of data is obviously valuable, but often visualising the data through pictures is a more effective way to understand the information and discover patterns. The human brain seems to be designed to find patterns in data, but sometimes it seems to do so even when the patterns are just the result of chance. This is why a proper statistical analysis is also necessary.

We can classify our data sets by the number of dimensions involved; that is, the number of measurements taken in each observation.

When the number of dimensions is small, there are some well-known techniques for data visualisation. We closely follow Chapter 2 of Everitt's book, mentioned earlier.

2.1 Scatterplots etc.

To illustrate the variety of plots which are available to us, we will very closely follow the second chapter of Everitt's book, mentioned above.

Of course, given a multivariate data set, we can plot each component using the usual univariate methods: histograms, stem-and-leaf plots and box plots. But it is more important that we can study the relationships between the variables. For 2 variables, scatter plots are the most common technique.

First, we need a data source, and we use Everitt's data on air pollution in various US cities. Download the data from `http://biostatistics.iop.kcl.ac.uk/publications/everitt/`) in a data frame `airpoll`, and list the contents (note that the syntax for these datasets is different from the `.Rdata` files):

```
> airpoll<-source("D:\\frazer\\teaching-ug\\MAS465\\chap2airpoll.dat")$value
> airpoll
         Rainfall Education Popden Nonwhite NOX SO2 Mortality
akronOH        36      11.4   3243      8.8  15  59     921.9
albanyNY       35      11.0   4281      3.5  10  39     997.9
allenPA        44       9.8   4260      0.8   6  33     962.4
atlantGA       47      11.1   3125     27.1   8  24     982.3
baltimMD       43       9.6   6441     24.4  38 206    1071.0
birmhmAL       53      10.2   3325     38.5  32  72    1030.0
bostonMA       43      12.1   4679      3.5  32  62     934.7
bridgeCT       45      10.6   2140      5.3   4   4     899.5
bufaloNY       36      10.5   6582      8.1  12  37    1002.0
cantonOH       36      10.7   4213      6.7   7  20     912.3
chatagTN       52       9.6   2302     22.2   8  27    1018.0
chicagIL       33      10.9   6122     16.3  63 278    1025.0
cinnciOH       40      10.2   4101     13.0  26 146     970.5
clevelOH       35      11.1   3042     14.7  21  64     986.0
colombOH       37      11.9   4259     13.1   9  15     958.8
dallasTX       35      11.8   1441     14.8   1   1     860.1
daytonOH       36      11.4   4029     12.4   4  16     936.2
denverCO       15      12.2   4824      4.7   8  28     871.8
detrotMI       31      10.8   4834     15.8  35 124     959.2
flintMI        30      10.8   3694     13.1   4  11     941.2
ftwortTX       31      11.4   1844     11.5   1   1     891.7
grndraMI       31      10.9   3226      5.1   3  10     871.3
grnborNC       42      10.4   2269     22.7   3   5     971.1
hartfdCT       43      11.5   2909      7.2   3  10     887.5
houstnTX       46      11.4   2647     21.0   5   1     952.5
indianIN       39      11.4   4412     15.6   7  33     968.7
kansasMO       35      12.0   3262     12.6   4   4     919.7
lancasPA       43       9.5   3214      2.9   7  32     844.1
losangCA       11      12.1   4700      7.8 319 130     861.8
louisvKY       30       9.9   4474     13.1  37 193     989.3
memphsTN       50      10.4   3497     36.7  18  34    1006.0
miamiFL        60      11.5   4657     13.5   1   1     861.4
milwauWI       30      11.1   2934      5.8  23 125     929.2
minnplMN       25      12.1   2095      2.0  11  26     857.6
nashvlTN       45      10.1   2082     21.0  14  78     961.0
newhvnCT       46      11.3   3327      8.8   3   8     923.2
neworlLA       54       9.7   3172     31.4  17   1    1113.0
newyrkNY       42      10.7   7462     11.3  26 108     994.6
philadPA       42      10.5   6092     17.5  32 161    1015.0
pittsbPA       36      10.6   3437      8.1  59 263     991.3
portldOR       37      12.0   3387      3.6  21  44     894.0
provdcRI       42      10.1   3508      2.2   4  18     938.5
readngPA       41       9.6   4843      2.7  11  89     946.2
richmdVA       44      11.0   3768     28.6   9  48    1026.0
rochtrNY       32      11.1   4355      5.0   4  18     874.3
stlousMO       34       9.7   5160     17.2  15  68     953.6
sandigCA       10      12.1   3033      5.9  66  20     839.7
```

```
sanfrnCA        18       12.2   4253       13.7 171  86      911.7
sanjosCA        13       12.2   2702        3.0  32   3      790.7
seatleWA        35       12.2   3626        5.7   7  20      899.3
springMA        45       11.1   1883        3.4   4  20      904.2
syracuNY        38       11.4   4923        3.8   5  25      950.7
toledoOH        31       10.7   3249        9.5   7  25      972.5
uticaNY         40       10.3   1671        2.5   2  11      912.2
washDC          41       12.3   5308       25.9  28 102      968.8
wichtaKS        28       12.1   3665        7.5   2   1      823.8
wilmtnDE        45       11.3   3152       12.1  11  42     1004.0
worctrMA        45       11.1   3678        1.0   3   8      895.7
yorkPA          42        9.0   9699        4.8   8  49      911.8
youngsOH        38       10.7   3451       11.7  13  39      954.4
```

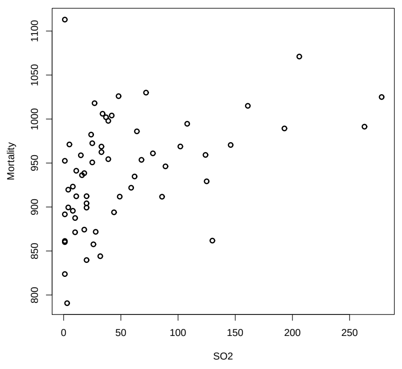
Before we plot some graphs, let's look at the summary data:

```
> summary(airpoll)
    Rainfall        Education          Popden         Nonwhite
 Min.   :10.00   Min.   : 9.00   Min.   :1441   Min.   : 0.80
 1st Qu.:32.75   1st Qu.:10.40   1st Qu.:3104   1st Qu.: 4.95
 Median :38.00   Median :11.05   Median :3567   Median :10.40
 Mean   :37.37   Mean   :10.97   Mean   :3866   Mean   :11.87
 3rd Qu.:43.25   3rd Qu.:11.50   3rd Qu.:4520   3rd Qu.:15.65
 Max.   :60.00   Max.   :12.30   Max.   :9699   Max.   :38.50
      NOX              SO2           Mortality
 Min.   :  1.00   Min.   :  1.00   Min.   : 790.7
 1st Qu.:  4.00   1st Qu.: 11.00   1st Qu.: 898.4
 Median :  9.00   Median : 30.00   Median : 943.7
 Mean   : 22.65   Mean   : 53.77   Mean   : 940.4
 3rd Qu.: 23.75   3rd Qu.: 69.00   3rd Qu.: 983.2
 Max.   :319.00   Max.   :278.00   Max.   :1113.0
```

Note that this gives summaries of each variable individually, but takes no account of any interactions (i.e., correlations) that the variables may have.

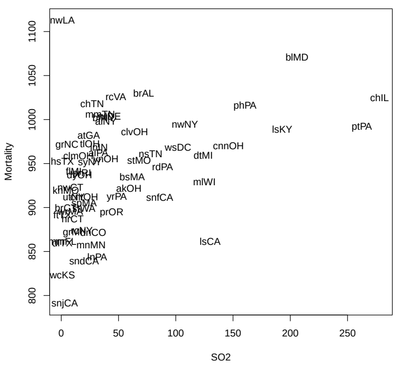
Now we want to plot some graphs. We will focus initially on the relation between the columns `SO2` and `Mortality`. Here is the basic scatterplot (you need to type `attach(airpoll)` first):

```
plot(SO2,Mortality,pch=1,lwd=2)
```



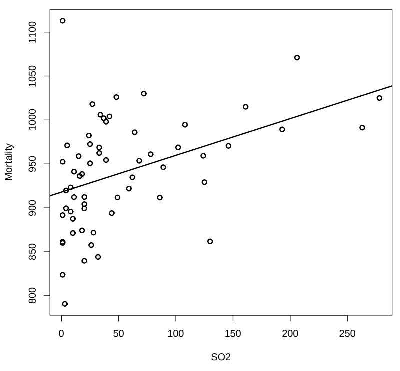
We will plot various things on top of this graph, but it is useful to be able to identify individual points (e.g., outliers), so we will start by adding names to the points:

```
names<-abbreviate(row.names(airpoll))
plot(SO2,Mortality,lwd=2,type="n")
text(SO2,Mortality,labels=names,lwd=2)
```



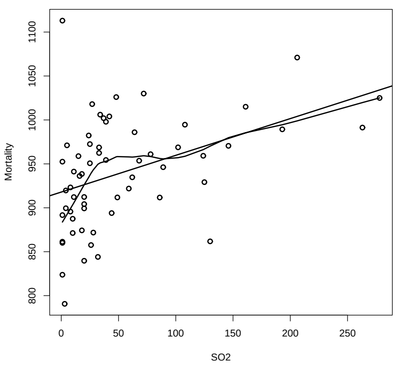
We can add a regression line:

```
plot(SO2,Mortality,pch=1,lwd=2)
abline(lm(Mortality~SO2),lwd=2)
```



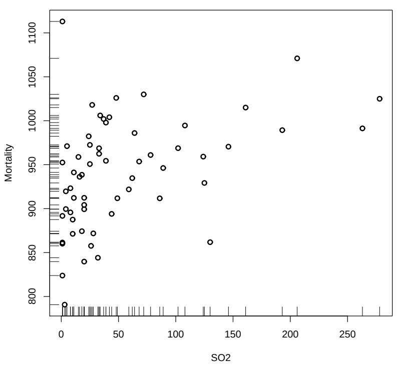
We can add also a more local line of best fit:

```
plot(SO2,Mortality,pch=1,lwd=2)
abline(lm(Mortality~SO2),lwd=2)
lines(lowess(SO2,Mortality),lwd=2)
```



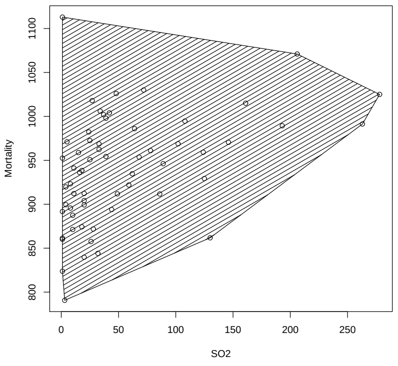
And we can add the “marginal” distributions to the plot as well:

```
plot(SO2,Mortality,pch=1,lwd=2)
rug(SO2,side=1)
rug(Mortality,side=2)
```



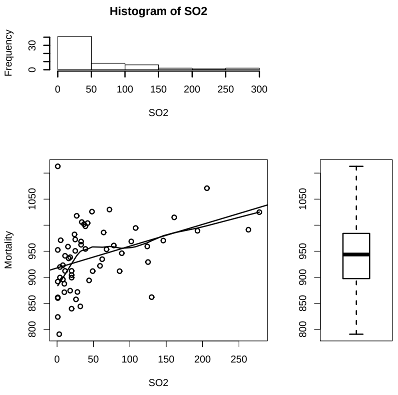
We can draw in the convex hull, to aid with finding outliers etc.:

```
hull<-chull(SO2,Mortality)
plot(SO2,Mortality,pch=1)
polygon(SO2[hull],Mortality[hull],
   density=15,angle=30)
```



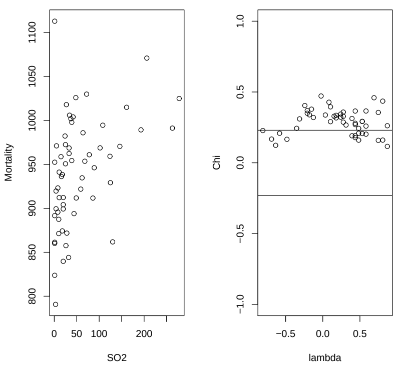
To analyse the data, Everitt mentions a number of tools. It is often useful to study the marginal distributions as well as the joint distribution; we have already seen one way to plot these on the same diagram, but Everitt gives another. We can look at a histogram of the `SO2` data and a boxplot of the `Mortality` data together in a matrix:

```
par(fig=c(0,0.7,0,0.7))
plot(SO2,Mortality,lwd=2)
abline(lm(Mortality~SO2),lwd=2)
lines(lowess(SO2,Mortality),lwd=2)
par(fig=c(0,0.7,0.65,1),new=TRUE)
hist(SO2,lwd=2)
par(fig=c(0.65,1,0,0.7),new=TRUE)
boxplot(Mortality,lwd=2)
```



Scatterplots are very good for summarising the data, but it is sometimes hard to see whether, for example, the two variables are independent. The *chiplot* is a way to see this; it plots points which one would expect to lie in a central region if the two variables are independent, and otherwise they may tend to lie outside. For details, see Everitt, pp.24–25. The function `chiplot` is not part of R, but can be downloaded from Everitt's website (see the file `rsplus2.r`, which makes use of the functions in `functions.txt`, which you should run first).

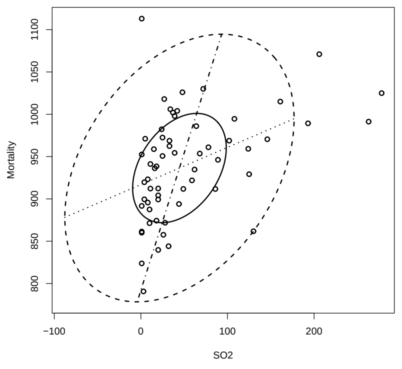
```
chiplot(SO2,Mortality,
   vlabs=c("SO2","Mortality"))
```



Here, one can see that the two variables seem *not* to be independent, as they largely lie outside the central box.

Everitt also provides a function `bvbox` to give bivariate boxplots. This superimposes a pair of concentric ellipses onto the scatterplot; the inner (the *hinge*) contains the innermost 50% of points, and the outer (the *fence*) excludes outliers:

```
bvbox(cbind(SO2,Mortality),xlab="SO2",
   ylab="Mortality")
```



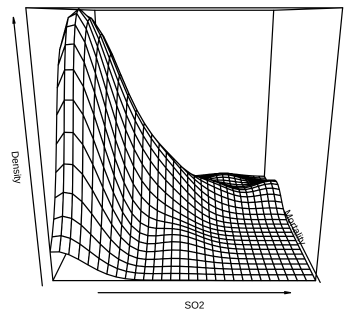
We'll explain later why ellipses are the appropriate shape for bivariate normal data.

The data can also be visualised using 3-dimensional techniques. Although scatter plots are excellent when there are a fairly small number of points, these techniques are often better with more points. With more points, other options include:

- bivariate histograms, drawn in perspective
- bivariate boxplots
- 2-dimensional frequency plots
- augmented scatter plots

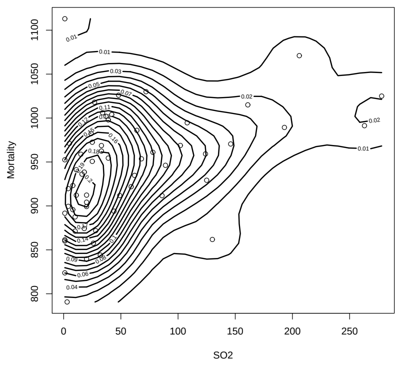
A 3-dimensional view of the data is given by the following:

```
den1<-bivden(SO2,Mortality)
persp(den1$seqx,den1$seqy,den1$den,
   xlab="SO2",ylab="Mortality",
   zlab="Density",lwd=2)
```



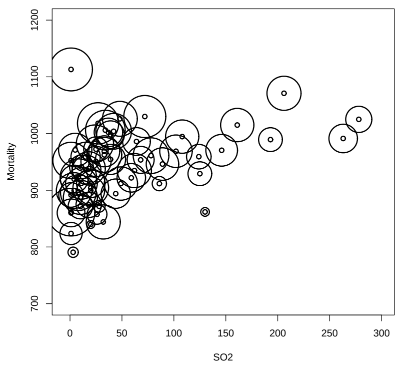
and a contour plot of this surface superimposed on the original data is given by

```
den1<-bivden(SO2,Mortality)
plot(SO2,Mortality)
contour(den1$seqx,den1$seqy,den1$den,
   lwd=2,nlevels=20,add=TRUE)
```



A third variable (e.g., `Rainfall`) can be plotted in various ways. We can augment the scatterplot with extra symbols; the following code draws circles around each point with radius proportional to the rainfall, to give a *bubbleplot*:

```
plot(SO2,Mortality,pch=1,lwd=2,
   ylim=c(700,1200),xlim=c(-5,300))
symbols(SO2,Mortality,circles=Rainfall,
   inches=0.4,add=TRUE,lwd=2)
```

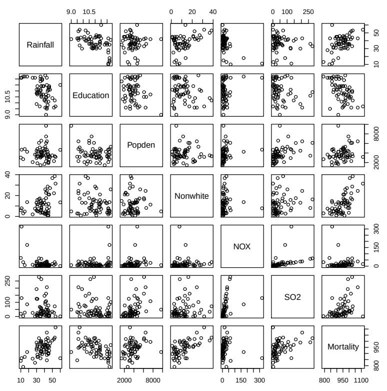


Or we can plot various 2-dimensional random views, or 3-dimensional scatter plots drawn in perspective or rotated interactively. It is easy to construct datasets with outliers which one cannot see from any of the co-ordinate projections (one can do this also in 2 dimensions). This means that if one is to make a 2-dimensional visualisation by picking viewpoints, one has to be careful. Modern computer software allows a more dynamic approach, displaying a slowly changing view of all the data, or allowing the user to manipulate the view by rotating the plots interactively.

The reader should also be aware of *conditioning plots* and the more general *trellis graphics*. We will not go into details here, but the help system details for the R command `coplot` will give some information.

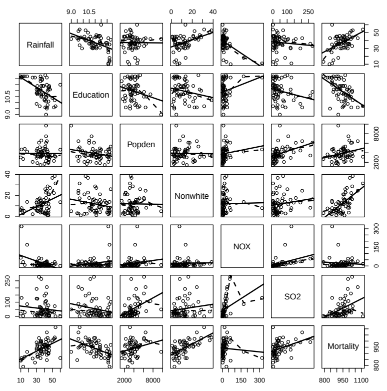
With more dimensions, the key tool in displaying multivariate data is scatterplots of pairwise components, arranged as a *matrix plot*.

```
pairs(airpoll)
```



With a little work, one can add all the regression lines also:

```
pairs(airpoll,panel=function(x,y)
   {abline(lsfit(x,y)$coef,lwd=2)
    lines(lowess(x,y),lty=2,lwd=2)
    points(x,y)})
```



As soon as the number of dimensions becomes large (i.e., more than about 5), it is difficult to make sense of the display. If the number of observations is small, then some other technique (e.g., star plots, see below) can handle quite large numbers of variables. If there are large numbers of both variables and observations, then we will need to determine the "most interesting" dimensions (this is the purpose of the dimensionality-reduction techniques that we will come to next), and then do matrix plots.

2.2 Andrews plots

See D.F.Andrews (1972) *Plots of high-dimensional data*, Biometrics **28**, 125–136.

Another, rather different way, to visualise data is by using *Andrews plots*. As usual, we take the data to be vector observations $\{x_i \mid i = 1, \ldots, n\}$, so x_{ij} is the jth element of the ith observation.

Then we define

$$f_{x_i}(t) = \frac{1}{\sqrt{2}} x_{i1} + x_{i2} \sin t + x_{i3} \cos t + x_{i4} \sin 2t + \cdots + x_{ip} \frac{\sin}{\cos} \left(\left\lfloor \frac{p}{2} \right\rfloor t \right).$$

This maps p-dimensional data $\{x_i\}$ onto 1-dimensional $\{f_{x_i}(t)\}$ for any t. If we plot $f_{x_i}(t)$ over $-\pi < t \leq +\pi$, we obtain a 1-dimensional representation of the data with properties:

1. preserves means, i.e.,
$$f_{\overline{x}}(t) = \frac{1}{n} \sum_{i=1}^{n} f_{x_i}(t);$$

2. preserves distances, i.e., if we define the square of the distance between two functions as
$$\|f_{x_1}(t) - f_{x_2}(t)\|^2 = \int_{-\pi}^{\pi} (f_{x_1}(t) - f_{x_2}(t))^2 \, dt = \pi \sum_{j=1}^{p} (x_{1j} - x_{2j})^2,$$
we get $\pi\times$ the square of the Euclidean distance between x_1 and x_2, so close points appear as close functions (though not necessarily vice versa);

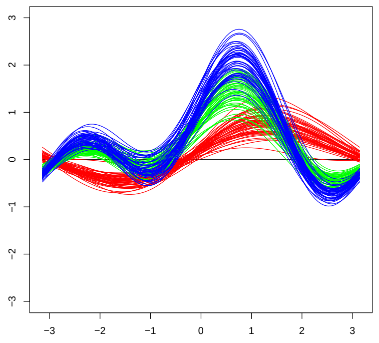
3. yields 1-dimensional views of the data: at $t = t_0$ we obtain the projection of the data onto the vector
$$f_1(t_0) = (1/\sqrt{2}, \sin t_0, \cos t_0, \sin 2t_0, \ldots)'$$
(i.e., viewed from some position, the data look like the 1-dimensional scatter plot given on the vector $f_1(t_0)$, i.e., the line intersecting the Andrews plot at $t = t_0$).

4. significance tests and distributional results are available.

- the plot does depend on the order in which the components are taken. A preliminary transformation of data may be sensible (e.g., principal components);
- other sets of orthonormal functions are possible;
- the "coverage" decreases rapidly as the dimensionality increases, i.e., for high dimensions, many "views" of the data are missed, and we miss separation between groups of points or other important features.

Many packages have facilities for producing Andrews plots. The R package `andrews` can be installed from the CRAN website. Here is a plot of a famous data set on iris lengths on three varieties of iris plants (note that the groupings of the plants into the three varieties can clearly be seen from the colour).

```
data(iris)
   andrews(iris,clr=5,ymax=3)
```



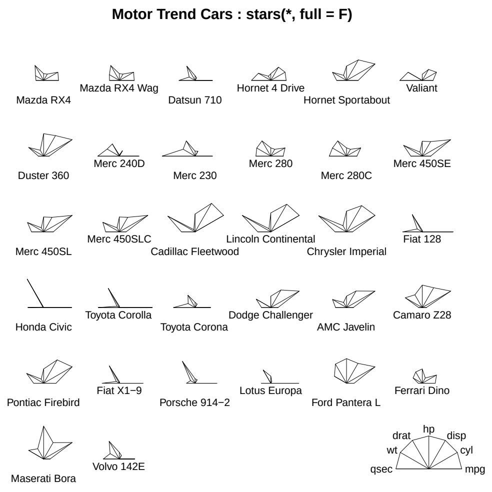
We will say more about this iris data in the next chapter.

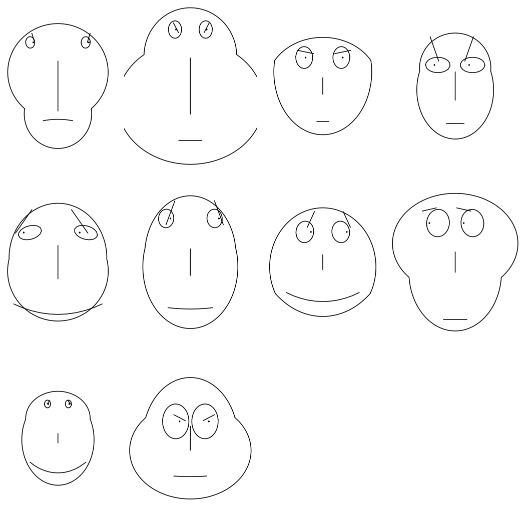
2.3 Star plots and Chernoff faces

When the number of observations is fairly small (less than 50, say), an alternative way to view the data uses "star plots". A *star plot* is one where each observation is represented by a star or polygon where the length of the vector to each vertex corresponds to the value of a particular variable.

Another way to produce visualisations is to use *Chernoff faces*. This is not a very serious technique but is widely known and available in some packages (e.g., in R, use the library `aplpack` or `TeachingDemos` with the function `faces()`). The idea is to code each variable in some feature of a face (e.g., length of nose, curvature of mouth). Then you can display each observation as a face and look for "family resemblances". It is thought that humans' ability to recognise and interpret faces has evolved to a very high extent, and this technique tries to exploit this.

Here are some examples:





2.4 Conclusions

Simple scatter plots arranged in a matrix work well when there are not too many variables. If there are more than about 5 variables, it is difficult to focus on particular displays and so understand the structure. If there are not too many observations, we can construct a symbol (e.g., a star or polygon) for each separate observation which indicates the values of the variables for that observation. Obviously star plots will not be useful if either there are a large number of observations (more than will fit on one page) or if there are a large number of variables (perhaps more than 20).

However, many multivariate statistical analyses involve very large numbers of variables; more than 50 is routine, and new areas of application often involve more than 1000.

What is required is a display of "all of the data" using just a few scatterplots, i.e., using just a few variables. That is, we need to select "the most interesting variables". This may mean concentrating on the "most interesting" actual measurements, or it may mean combining the variables together in some way to create a few new ones which are the "most interesting". That is, we need some technique of *dimensionality reduction*.

3 Dimensionality reduction I: Principal Components Analysis (PCA)

Suppose we have a large amount of data, involving many attributes. For example, we could be surveying 10000 people, and collecting data on age, height, weight, income, blood pressure, etc., so that we associate many pieces of information with one person. Then we have many many data points which we plot in a high-dimensional space, which we want to analyse, and find patterns in.

As we have explained in the previous chapter, it is not easy to visualise data in the multivariate setting: when the number of dimensions is large, we cannot easily use graphical techniques. It is often desirable to try to view the data in smaller dimensions.

As an analogy, consider the planets of the solar system. While they naturally live in 3-dimensional space (ignoring relativistic distortions etc.!), they approximately orbit in a 2-dimensional plane. We would try to view the planets as orbiting in a 2-dimensional world.

This leads to the problem:

> Given n points in p dimensional space $\mathbb{R}^p$, and some number $q < p$, find the "best" q-dimensional space $X \subset \mathbb{R}^p$, and projections of the n points into X so that as little information is lost as possible.

If we can find this set X, we can then regard the data points as lying (nearly) in the smaller dimensional set, and can then hope that this reduction in dimensions allows us to visualise the data better.

Using fewer dimensions for the data has other benefits too: the data can be manipulated more easily, and the data can be stored in less space (*data compression*).

Often the data can be correlated (e.g., height and weight may be related), and this means that the data points might be appropriately represented by some combination of variables (e.g., size). We therefore try to replace the variables with a smaller set (the 2 dimensions of height and weight might be replaced with a 1-dimensional measure of size), keeping as much information as we can.

While one can technically try to perform this reduction to a shape of any sort, we can use the full power of matrix techniques only by doing the reduction to a linear space, and it is this case which we shall focus on.

3.1 Introduction

As usual, our data matrix is given by $X' = \{x_{ij} \mid i = 1, \dots, n,\ j = 1 \dots, p\} = \{x_i \mid i = 1, \dots, n\}$.

The objective is to find a *linear* transformation given by a "change of basis", so that new variables are linear combinations of old variables with nice properties. In particular, such linear transformations correspond to multiplying the data matrix X' to get another matrix Y'; we want the map $X' \longrightarrow Y'$ such that the first component of Y' is the "most interesting", the second is the "second most interesting", etc., i.e., we want to choose a new coordinate system so that the data, when referred to this new system Y', are such that the first component contains "most information", the second contains the next "most information", etc.

Note that "most interesting" and "most information" translate to "maximum variance".

With luck, the first few components contain "nearly all" the information in the data, and the remaining components contain relatively little information, and can be discarded (i.e., the statistical analysis can be concentrated on just the first few components – multivariate analysis is much easier in 2 or 3 dimensions, where the data can be visualised easily).

A linear transformation $X' \longrightarrow Y'$ is given by $Y' = X'A$ where A is a $p \times p$-matrix; it makes statistical sense to restrict attention to *non-singular* matrices A. If A happens to be an *orthogonal* matrix, i.e., $A'A = I_p$, then the transformation $X' \longrightarrow Y'$ is an orthogonal transformation (i.e., just a rotation and/or a reflection of the n points in p-dimensional space).

3.2 Principal components

The basic idea is to find a set of orthogonal coordinates such that the sample variances of the data with respect to these coordinates are in decreasing order of magnitude, i.e., the projection of the points onto the first principal component has maximal variance among all such linear projections, the projection onto the second has maximal variance subject to being uncorrelated with the first, the third has maximal variance subject to being uncorrelated with the first two, etc.

We want to take the linear combination $X'a$ of the data matrix by choosing a so that the variance of $X'a$ is maximised. A moment's thought will show you that this is not quite enough – we could replace a by ka, and the variance increases by a factor of k^2. We need to normalise the choice of a in some way.

Definition 3.1 The *first principal component* is the vector a_1 such that the projection of the data X' onto a_1, i.e., $X'a_1$, has maximal variance, subject to the normalising constraint that $a_1'a_1 = 1$ (i.e., a_1 has length 1).

Let us now explain how to find a_1.

We know that $\text{var}(X'a_1) = a_1'\text{var}(X')a_1 = a_1'Sa_1$. Notice that a_1' is a $1 \times p$ matrix, S is a $p \times p$ matrix and a_1 is a $p \times 1$ matrix, so $a_1'Sa_1$ is a scalar. So the problem is to maximize the scalar $a_1'Sa_1$ subject to the constraint that $a_1'a_1 = 1$.

This is the first of several points in the module that we will need to perform some maximization procedure for functions of several variables, constrained to satisfy some conditions. We have already developed the necessary techniques in Chapter 1. The general procedure is the following:

1. Introduce a Lagrange multiplier and define a new objective function;
2. Differentiate with respect to x (generally a vector) and set to 0;
3. Recognise that this is an eigenequation with the Lagrange multiplier as eigenvalue;
4. Deduce that there are *only* a limited number of possible values for this eigenvalue (all of which can be calculated numerically);
5. Use some extra step to determine which eigenvalue gives the maximum (typically use the constraint somewhere).

Remark 3.2 We will need to differentiate with respect to a vector below. If $x = (x_1 \ \dots \ x_p)'$ is a p-vector, and $f = f(x)$ is a scalar function of x, recall that $\frac{\partial f}{\partial x}$ is the vector $(\frac{\partial f}{\partial x_1} \ \dots \ \frac{\partial f}{\partial x_p})'$. If $f(x) = x'Sx$, where S is a symmetric $p \times p$ matrix, then $\frac{\partial f}{\partial x} = 2Sx$, by Lemma 1.15.

For this problem, this procedure works as follows:

- We have already remarked that we assume that $a_1'a_1 = 1$.
- Define $\Omega_1 = a_1'Sa_1 - \lambda_1(a_1'a_1 - 1)$ where λ_1 is a Lagrange multiplier; we now need to maximize Ω_1 with respect to both a_1 and λ_1.
- Setting $\frac{\partial \Omega_1}{\partial \lambda_1} = 0$ gives $a_1'a_1 = 1$.

  Differentiating with respect to (the vector) a_1 gives $\frac{\partial \Omega_1}{\partial a_1} = 2Sa_1 - 2\lambda_1 a_1$, and setting this equal to 0 gives $Sa_1 - \lambda_1 a_1 = 0$, so that a_1 is an eigenvector of S corresponding to the eigenvalue λ_1.
- Providing S is non-singular, S has p non-zero eigenvalues. So which eigenvalue is λ_1?
- If $Sa_1 = \lambda_1 a_1$, then $a_1'Sa_1 = \lambda_1 a_1'a_1 = \lambda_1$, so to maximize $\text{var}(X'a_1) = a_1'Sa_1$, we must take λ_1 to be the *largest* eigenvalue of S and a_1 as the corresponding eigenvector, and then the maximum value of this variance is λ_1 (which is also the required value of the Lagrange multiplier).

We want the second component to be uncorrelated with the first. This means that we need $a_2'Sa_1 = 0$. However, $Sa_1 = \lambda_1 a_1$, and so this is equivalent to $a_2'a_1 = 0$, i.e., that a_2 and a_1 are orthogonal.

Let's work out the second principal component: a_2 should have the property that projection of X' onto a_2 has maximal variance subject to $a_2'a_2 = 1$ and $a_2'a_1 = 0$. We should maximize

$$\Omega_2 = a_2'Sa_2 - \mu a_2'a_1 - \lambda_2(a_2'a_2 - 1),$$

where μ and λ_2 are Lagrange multipliers. Differentiating with respect to μ and λ_2 just expresses the constraints. Differentiating with respect to a_2 gives

$$2Sa_2 - \mu a_1 - 2\lambda_2 a_2 = 0.$$

Then

$$\begin{aligned} 2a_1'Sa_2 - \mu a_1'a_1 - 2\lambda_2 a_1'a_2 &= 0 \\ 2a_2'Sa_2 - \mu a_2'a_1 - 2\lambda_2 a_2'a_2 &= 0 \end{aligned}$$

and we see that

$$\mu = 2a_1'Sa_2 = 2(Sa_1)'a_2 = 2(\lambda_1 a_1)'a_2 = 2\lambda_1 a_1'a_2 = 0$$

and so $2Sa_2 - 2\lambda_2 a_2 = 0$, and a_2 is an eigenvector of S with eigenvalue λ_2. Further,

$$2a_2'Sa_2 - \mu a_2'a_1 - 2\lambda_2 a_2'a_2 = 0,$$

or $\text{var}(X'a_2) = \lambda_2$, so λ_2 must be the second largest eigenvalue of S, with eigenvector a_2.

Proceeding in this way, it is easy to see that the same argument shows that

Theorem 3.3 *The p principal components of data X' are the p eigenvectors $a_1, \dots, a_p$ corresponding to the p ordered eigenvalues $\lambda_1 \geq \cdots \geq \lambda_p$ of S, the variance of X'.*

There is a slight difficulty if $\lambda_i = \lambda_{i+1}$ since then a_i and a_{i+1} can be chosen in arbitrarily many ways, mutually orthogonal but anywhere in the eigenspace corresponding to $\lambda_i = \lambda_{i+1}$. However, this happens only with probability 0 in practice.

There is another difficulty if $\lambda_p = 0$, since then a_p is not uniquely defined, but this only happens when one variable is a linear combination of the others (very unlikely if $n \geq p$), and in any case, components with $\lambda_p \sim 0$ are unlikely to be interesting.

Remark 3.4 If one of the eigenvalues is 0, you should check the data set carefully. It might be that there are columns which are aggregates of others; for example, a data set consisting of exam scores might contain marks for individual modules, as well as an overall average. Clearly this latter column will be a linear combination of the individual module marks – you should ignore this column in the data set.

3.3 A simple example

We'll go through the procedure for one really simple 2-dimensional toy example, and perform all the procedures by hand for this set. The aim is to ensure that you can see, in a very simple example, how each procedure works, before we give the R code to do it automatically on larger sets.

Here is our data set, consisting of observations (x_i, y_i):

$$(1,4), \quad (3,7), \quad (5,8), \quad (7,11), \quad (9,15).$$

We compute the mean of each variable, and see that the mean is $(\overline{x}, \overline{y}) = (5, 9)$. Then we get the variance matrix as $S = \begin{pmatrix} 10 & 13 \\ 13 & 17.5 \end{pmatrix}$. As you can see from the description above, the whole analysis depends only on this variance matrix. It has two eigenvalues, which are computed as $\lambda_1 = 27.2800\dots$ and $\lambda_2 = 0.2199\dots$. Thus $\frac{\lambda_1}{\lambda_1+\lambda_2} = 0.992$, and it follows that the first principal component contains 99.2% of the information in the points. (This reflects the fact that the five points aren't so far from lying in a straight line; they are very highly correlated.)

Let's project the data onto a 1-dimensional space (i.e., a line). The (normalised) eigenvector corresponding to the eigenvalue λ_1 is given by $e_1 = \begin{pmatrix} 0.6011\dots \\ 0.7991\dots \end{pmatrix}$, and the line we want is the line whose gradient is given by e_1, and which passes through the mean. Since this line is $x = \overline{x} + te_1$, it is easy, with the help of some simple calculations, to translate each data point x onto the line.

Let's do one example: take $x_1 = (1, 4)'$.

Then we want to work out the corresponding value of t (this should be regarded as the *co-ordinate* of the principal component e_1) for x_1.

Write $x_1 = \overline{x} + te_1 + ue_2$. As $x_1 = (1, 4)'$ and $\overline{x} = (5, 9)'$, we have to solve $(-4, -5)' = te_1 + ue_2$. We could use simultaneous equations, but it is easier to use the orthogonality of e_1 and e_2; we simply pre-multiply by e_1':

$$e_1'(-4, -5) = te_1'e_1 + ue_1'e_2 = t.1 + u.0 = t.$$

So we compute $t = e_1'(-4,-5)' = e_1' \begin{pmatrix} -4 \\ -5 \end{pmatrix} = -4 \times 0.6011 \ldots - 5 \times 0.7991 \ldots = -6.4002 \ldots$.
We can similarly work out u, by computing $e_2'(-4,-5)'$. However, we do not need this, as we are going to ignore the e_2 component!

We see that x_1 is mapped to $\overline{x} + te_1$ under the transformation, and this is

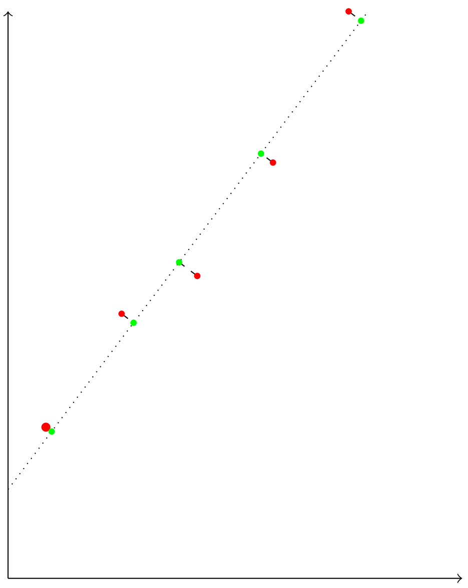
$$\overline{x} + te_1 = \begin{pmatrix} 1.1522 \ldots \\ 3.8854 \ldots \end{pmatrix}.$$

This is the point that $(1,4)'$ is mapped to on the first principal component.

Similarly, we can see where all the points are mapped to:

$$\begin{aligned} (1,4) &\text{ is mapped to } (1.1522, 3.8854) \\ (3,7) &\text{ is mapped to } (3.3163, 6.7620) \\ (5,8) &\text{ is mapped to } (4.5195, 8.3614) \\ (7,11) &\text{ is mapped to } (6.6836, 11.2379) \\ (9,15) &\text{ is mapped to } (9.3281, 14.7531) \end{aligned}$$

The similarities with lines of best fit should be clear! We are trying to project the data points onto a line so that as little information is lost as possible. To test your understanding, you might like to program this procedure in R (or whichever programming language you wish) and try to replicate these results.



3.4 Issue – Units: covariance or correlation?

Notice that principal components are not invariant under all linear transformations of original variables; in particular, separate scaling of the variables. For example, principal components of data of people's heights, weights and incomes measured in inches, pounds and pounds sterling will not generally be the same as those on the same data converted to centimetres, kilograms and euros. This means one should be careful in the interpretation of analyses of data on mixed types. This is especially true if the variables have widely different variances, even if they are all of the same type. If one variable has a very large variance, then the first principal component will be dominated by this one variable.

It should be slightly alarming that different choices of units give different principal components (even if the components are scaled to the same units, we get different answers). However, it does tend to be the case that the many different ways allow one to draw relatively similar qualitative conclusions.

Where there is no reason to treat the variables on a similar scale, a natural way to standardise the data is to use the correlation matrix, rather than the variance matrix. This ensures that the co-ordinates are equally spread, and are originally viewed as having the same importance. (It is equivalent to *scaling* the data so that the variance of each variable becomes 1.)

Generally, if data X' are measurements of p variables, all of the same type (e.g., all concentrations of amino acids or all linear dimensions in the same units, but not, for example, age/income/weights), then the coefficients of principal components can be interpreted as "loadings" of the original variables and hence the principal components can be interpreted as contrasts in the original variables, as with some of the examples later.

This interpretation is less easy if the data are of mixed "types" or if the variables have widely different variances, even if the eigenanalysis is based on the correlation rather than the variance matrix.

In practice, however, it tends to be the case that one of the variables has much larger variance than another, or that differing scales are used, and it is the correlation matrix which is most often used (indeed, it is the default in many computer packages).

However, each individual data set should be considered individually; there is scope for adjusting the weightings associated to each variable.

Of course, we can perform PCA in R; the default in R is to use the covariance matrix, but one can specify the correlation matrix instead. We shall say more about computing principal components in R later.

Let's take the same data set as above, but do the PCA using the correlation matrix instead of the variance matrix. Here are the points:

$$(1,4), \quad (3,7), \quad (5,8), \quad (7,11), \quad (9,15).$$

Step 1 Compute the correlation matrix. This is $\begin{pmatrix} 1 & 0.9827 \\ 0.9827 & 1 \end{pmatrix}$ (since the points are so nearly collinear, they are very highly correlated).

Step 2 Calculate the *eigenvectors and eigenvalues* of the variance matrix.

Let's work out the characteristic polynomial:

$$\det(S - \lambda I) = \det \begin{pmatrix} \lambda - 1 & -0.98270 \\ -0.98270 & \lambda - 1 \end{pmatrix} = \lambda^2 - 2\lambda + 0.0343,$$

and the roots (the eigenvalues) are 1.9827 and 0.0173. The eigenvector corresponding to the eigenvalue of 1.9827 is $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and the eigenvector corresponding to the eigenvalue of 0.0173 is $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$. It is natural (and useful) to normalise these vectors to have length 1, so that the eigenvectors will be $\begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$ and $\begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}$.

(When there are only 2 columns of data, these are always the eigenvectors, as you might like to explain, but things get more interesting with more columns!)

Step 3 Choose the most *important* eigenvectors.

Clearly the dominant eigenvector is $e_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$, we shall ignore the other.

Step 4 Convert the data to the principal components.

We've got some data, and now we've chosen the principal components. We need to compress our data and give it in terms of these eigenvectors.

We do the same as we did with the variance matrix. There's a slight subtlety because of the scaling implicit in the data to get from the covariances to the correlations. We'll do the first point again, before giving all the results.

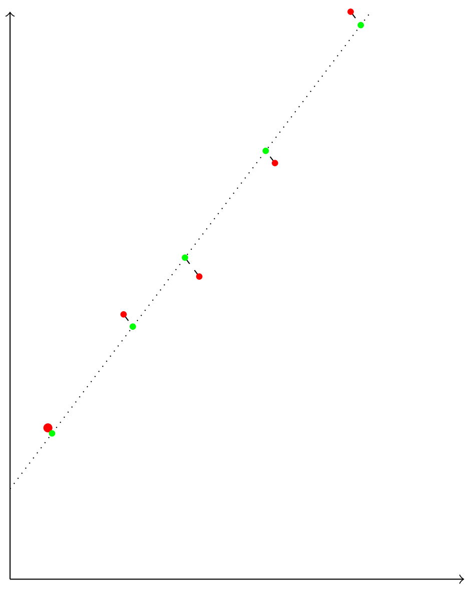
The easiest way is to start with x_1, and subtract the mean, to get a point $(1,4)-(5,9)=(-4,-5)$. Then scale the first coordinate by its standard deviation, and the second by its standard deviation. That is, we scale all the first co-ordinates by $1/\sqrt{10}$, and all the second by $1/\sqrt{17.5}$, so that they have variance 1.

Now the shifted first point is $(-4/\sqrt{10}, -5/\sqrt{17.5}) = (-1.2649, -1.1952)$.

Then we want to write this in terms of the two eigenvectors, and especially to calculate the contribution of the main eigenvector. So we compute $e_1'(-1.2649, -1.1952) = -1.7396$, and take this multiple of $(1/\sqrt{2}, 1/\sqrt{2})$, to get $(-1.2301, -1.2301)$.

Now we translate back, by scaling the first coordinate back by $\sqrt{10}$ and the second by $\sqrt{17.5}$, and then add the mean $(5,9)$ to get $(1.1102, 3.8542)$, and this is the projection we want.

$$\begin{aligned}
(1,4) &\text{ is mapped to } (1.1102, 3.8542) \\
(3,7) &\text{ is mapped to } (3.2441, 6.6771) \\
(5,8) &\text{ is mapped to } (4.6220, 8.5) \\
(7,11) &\text{ is mapped to } (6.7559, 11.3229) \\
(9,15) &\text{ is mapped to } (9.2679, 14.6458)
\end{aligned}$$



You might notice that the projection lines don't seem to be orthogonal to the principal component – you should think why this is a consequence of the scaling of the coordinates.

In this example, the two computations of the principal components didn't lead to very different answers (but they are definitely different!). The reason is that the second component was almost negligible. In examples where components other than the first have more significance, or where the variances of the variables are more widely different, the answers can be quantitatively somewhat different. In general, this does not seem to affect the analysis qualitatively, but the data analyst needs to be aware of this issue.

3.5 Computation

Given that we have this algorithmic approach to the PCA, it will be no surprise that we can implement this easily on a computer, nor that it is already implemented in R.

We have $\lambda_1, \dots, \lambda_p$ satisfying $(S - \lambda I_p)a = 0$. This is a system of p simultaneous equations in the components of a, so the determinant of the coefficients must be 0 for non-trivial solutions to exist, i.e., we have $\det(S - \lambda I_p) = 0$.

This is a polynomial of degree p in λ. In principle, this case be solved for $\lambda_1, \dots, \lambda_p$ and then each eigenvector can be read off, and then normalised with the constraint that $a_i'a_i = 1$.

There are many algorithms for computing eigenvalues of matrices, and which are implemented in very many computer packages. In particular, we can do this in R (and other mathematical and statistical packages).

In R, with data in matrix X, do

```
S<-var(X)
```

to get the variance matrix, and then

```
S.eig<-eigen(S)
```

to put the eigenvectors of S (i.e., the loadings of the principal components) in the variable `S.eig$vectors` and the eigenvalues in `S.eig$values`.

The basic function for principal components analysis in R is `princomp(mydata)`, where `mydata` is a dataframe.

The command `mydata.pca<-princomp(mydata)` puts the principal components of `mydata` in `mydata.pca$loadings` and the *square roots* of the eigenvalues in `mydata.pca$sdev` and the scores (i.e., the values rotated onto the principal components) into `mydata.pca$scores`.

`plot(mydata.pca)` produces a bar chart of the variances, `plot(mydata.pca$x[,1:2])` plots the scores on principal component 2 against those on principal component 1. The basic results (proportions of variance for each principal component etc.) can be examined with `summary(mydata.pca)`.

The bar chart produced by `plot(mydata.pca)` is no substitute for a scree graph of cumulative proportions of variance contained by successive principal components. R does not have an inbuilt function to do this, but it is simple to write one, see below. This function is on the course web page in the script file `scree.R` (see below).

Remark 3.5 There is an alternative version of all these commands. Instead of `eigen`, you can use `svd`; the command `S.svd<-svd(S)` puts the eigenvectors in `S.svd$v` and the eigenvalues in `S.svd$c`; the command `prcomp(mydata)` runs PCA on the data frame `mydata` and is based on `svd()` rather than `eigen()`. The command `mydata.pr<-prcomp(mydata)` puts the principal components of `mydata` in `mydata.pr$rotation` and the *square roots* of the eigenvalues in `mydata.pr$sdev` and the scores (i.e., the values rotated onto the principal components) into `mydata.pr$x`. See the R help system for more details. In fact, `svd()` is said to be more numerically stable than `eigen`, and also works when the number of variables is larger than the number of observations.

The examples given below use `princomp()`. It is suggested that you repeat these using `prcomp()`, taking care to change one or two details of the calls involved.

3.6 By how much can we reduce dimension?

Generally, if the columns are quite closely correlated, one would expect the first principal component to contain much of the information in the data set, so PCA is particularly effective in this case – we can replace the correlated columns with a single column corresponding to some pattern accounting for the correlation between the columns. So one initial test might be to compute some correlations between columns; if the columns tend to be well correlated, one would expect that fewer principal components would be required. Let's say a little more.

The principal components $a_1, \ldots, a_p$ provide a useful coordinate system to which to refer the data and for displaying them graphically, i.e., better than the original coordinates

$$\begin{pmatrix}1\\0\\0\\\vdots\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\\\vdots\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\\vdots\\0\end{pmatrix}, \ldots, \begin{pmatrix}0\\0\\0\\\vdots\\1\end{pmatrix}$$

Let $A = (a_1, \ldots, a_p)$, the $p \times p$ matrix with jth column a_j. As the columns are length 1 eigenvectors of a real symmetric matrix, we see that $A'A = I_p$, so that A is an orthogonal matrix.

Then the projection of the data X' onto this coordinate system is $Y' = X'A$. Since A is orthogonal, the transformation $X' \longrightarrow Y'$ is a rotation/reflection (i.e., no overall change of scale or origin).

If we measure the "total" variation in the original data as the sum of the variances of the original components, i.e.,

$$s_{11} + s_{22} + \cdots + s_{pp} = \text{tr}(S),$$

then the "total" variation of Y' is $\lambda_1 + \cdots + \lambda_p$, and $\sum_i \lambda_i = \text{tr}(S)$ as the trace of a matrix is the same as the sum of its eigenvalues. (One can view this as a kind of "conservation of total information" in the PCA process.)

So we can interpret $\lambda_1 / \sum \lambda_i$ as the proportion of the total variation "explained" by the first principal component, and $(\lambda_1 + \lambda_2)/\sum \lambda_i$ as the proportion of the total variation explained by the first two principal components etc.

If the first few principal components explain most of the variation in the data, then the later principal components are redundant and little information is lost if they are discarded or ignored.

Quite how much data we need to retain depends on the data and area of application. Some areas might be satisfied if we choose enough components to make up 40% of the total variation, while others might require 90%. A figure needs to be chosen as a trade-off between the convenience of a small value of k and a large value of the cumulative relative proportion of variance explained.

If p is large, an informal way of choosing a good k is graphically, with a scree plot, plotting j against $(\sum_1^j \lambda_i)/\text{tr}(S)$ against j. The graph will be monotonic and convex; it will typically increase steeply for the first few values of j and then begin to level off. The point where it starts levelling off is the point where bringing in more principal components brings less returns in terms of variance explained.

An R function `screeplot()`

This function (due to the previous lecturer of the module, Nick Fieller) produces screeplots of cumulative partial sums of variances on each principal component against the number of dimensions. By default, the principal components are calculated from the variance matrix, and the number of dimensions used is 10. Both of these can be overridden in the call statement – see the examples. To use in an R session, copy and paste the complete list of commands to a script window, and run them. It can be downloaded from the course webpage.

```
# function to draw screeplots of cumulative
# eigenvalues in principal component analysis

screeplot<-function(mydata,cor=F,maxcomp=10) {
my.pc<-princomp(mydata,cor=cor)
k<-min(dim(mydata),maxcomp)
x<-c(0:k)
y<-my.pc$sdev[1:k]*my.pc$sdev[1:k]
y<-c(0,y)
z<-100*cumsum(y)/sum(my.pc$sdev*my.pc$sdev)

plot(x,z,type="l",xlab="number of dimensions",
 cex.main=1.5, lwd=3, col="red",
 ylim=c(0,100),
 ylab="cumulative percentage of total variance",
  main="Scree plot of variances",
 xaxt="n", yaxt="n")

axis(1,at=x,lwd=2)
axis(2,at=c(0,20,40,60,80,100),lwd=2)
abline(a=100,b=0,lwd=2,lty="dashed",col="orange")
text(x,z,labels=x,cex=0.8,adj=c(1.2,-.1),col="blue")
}

# Examples of calls to it are
# screeplot(mydata) # default uses covariance, maximum 10 components
# screeplot(mydata,T) #  uses correlations, maximum 10 components
# screeplot(mydata,maxcomp=7) # default use covariance, maximum 7 components
# screeplot(mydata,T,maxcomp=8) # use correlations, maximum 8 components
```

3.7 Geometrical interpretation

It is valuable to get some idea of what PCA is really doing geometrically, and to see also the role played by the variance matrix in the geometry of the sample.

We can explain this most easily in 2 dimensions. Suppose we have some 2-dimensional set of data points, and we work out the variance matrix and its eigenanalysis.

Then this means that the data is quite well approximated by an ellipse whose axes are the eigenvectors of the variance matrix, with corresponding lengths proportional to the square roots of the eigenvalues.

(See the plot in the notes in Chapter 2 for Everitt's function `bvbox`.)

Then PCA merely rotates the ellipse so that the axes lie along the co-ordinate axes.

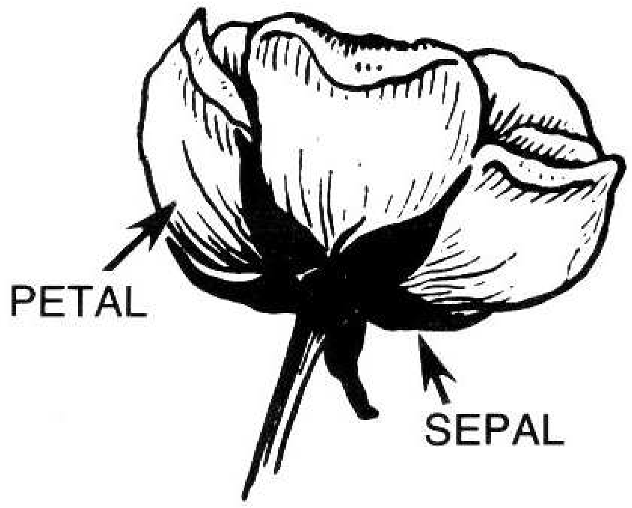
The same happens in more dimensions – the data is approximately ellipsoidal in shape, where the axes are given by the eigenvectors of the variance matrix, with lengths proportional to the square roots of the eigenvalues, and PCA rotates the ellipsoid onto the standard co-ordinate axes.

3.8 Example: Iris data (Anderson/Fisher)

Perhaps the most analysed data set in history, is one collected by Anderson for analysis by Fisher, consisting of 50 examples of each of 3 varieties of iris flower.

The data set is on MOLE (it is `irisnf.Rdata`); each observation measures the petal and sepal lengths and widths of the flowers.

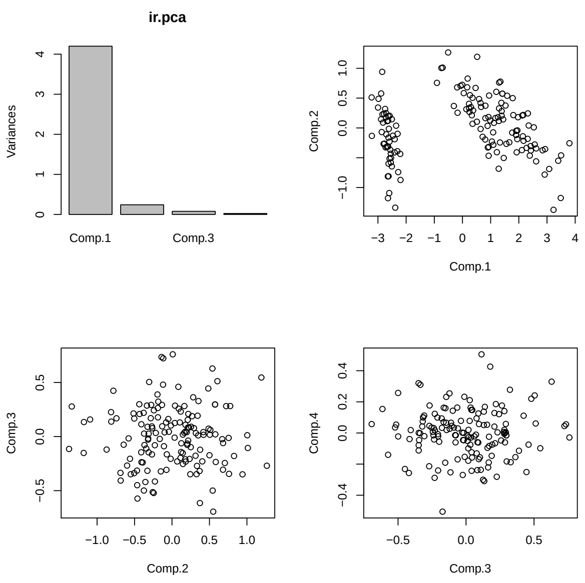
Petals and sepals



Here is the R analysis:

```
> ir<-iris[,-5]
> ir.pca<-princomp(ir)
> ir.pca
Call:
princomp(x = ir)

Standard deviations:
   Comp.1    Comp.2    Comp.3    Comp.4
2.0494032 0.4909714 0.2787259 0.1538707
```

```
 4  variables and  150 observations.
> summary(ir.pca)
Importance of components:
                          Comp.1     Comp.2     Comp.3      Comp.4
Standard deviation     2.0494032 0.49097143 0.27872586 0.153870700
Proportion of Variance 0.9246187 0.05306648 0.01710261 0.005212184
Cumulative Proportion  0.9246187 0.97768521 0.99478782 1.000000000
> plot(ir.pca)
> loadings(ir.pca)

Loadings:
             Comp.1 Comp.2 Comp.3 Comp.4
Sepal.Length  0.361 -0.657 -0.582  0.315
Sepal.Width         -0.730  0.598 -0.320
Petal.Length  0.857  0.173        -0.480
Petal.Width   0.358         0.546  0.754

               Comp.1 Comp.2 Comp.3 Comp.4
SS loadings      1.00   1.00   1.00   1.00
Proportion Var   0.25   0.25   0.25   0.25
Cumulative Var   0.25   0.50   0.75   1.00
> par(mfrow=c(2,2))
> plot(ir.pca)
> ir.pc<-predict(ir.pca)
> plot(ir.pc[,1:2])
> plot(ir.pc[,2:3])
> plot(ir.pc[,3:4])
```



We see that the first principal component contrasts flowers with big petals and long sepals with those with small petals and short sepals, i.e., big flowers with small flowers. The second contrasts big sepals with small sepals.

The first scatter plot contains 98% of the variation in the data. It can be seen that much of this variation is because the data separates into (at least) two groups along the first principal component.

As we've seen, we can type `iris.pc<-princomp(irisnf)` to do a full principal components analysis.

New data can be imposed on old data using `predict()`. To illustrate this, let's just do a PCA on the first two varieties, i.e., the first 100 observations. For this, type:

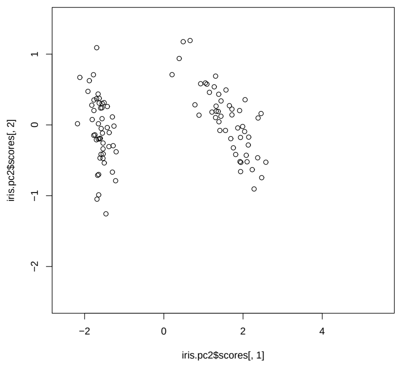
```
iris.pc2<-princomp(irisnf[1:100,-5])
```

Then we can consider the last variety, and rotate all of these observations onto the principal components using `predict`:

```
irisnew<-predict(iris.pc2,irisnf[101:150,-5])
```

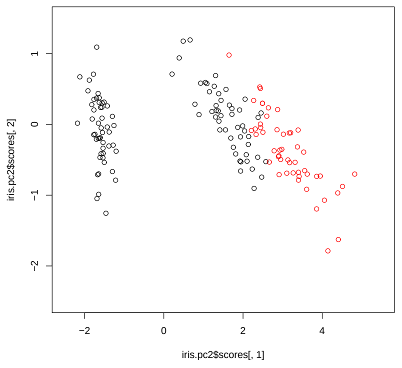
Let's now plot the first 100 observations:

```
plot(iris.pc2$scores[,1]
  ,iris.pc2$scores[,2]
  ,xlim=c(-2.5,5.5),ylim=c(-2.5,1.5))
```



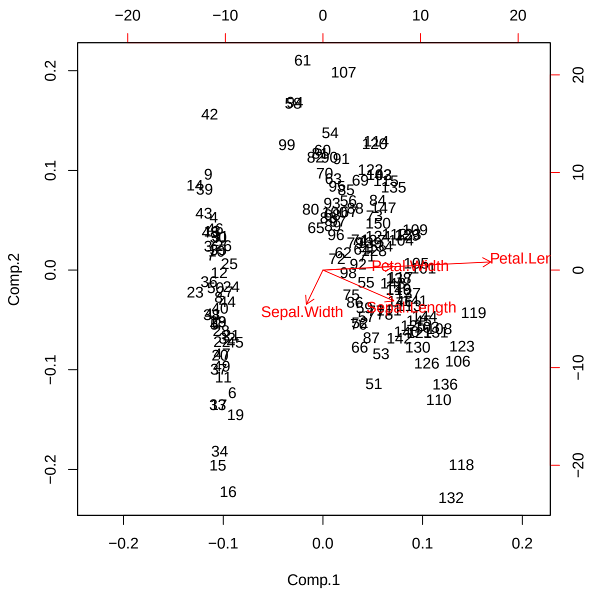
Note that we've specified the dimensions of the plot, so that the points we are about to add will fit on nicely. Then we'll add the other 50 points, stored in `irisnew`, to the plot, in a different colour to make them distinguishable:

```
points(irisnew[,1],irisnew[,2],col=2)
```



It is possible to represent the original variables on a plot of the data on the first two principal components (a *biplot*). The variables are represented as arrows, with lengths proportional to the standard deviations, and angles between a pair of arrows proportional to the covariances. The orientation of the arrows on the plot relates to the correlations between the variables and the principal components and so can be an aid to interpretation.

```
biplot(ir.pca)
```



3.9 Example: Turtle data (Morrison)

Measurements are taken in millimetres of length, width and height of shells of 24 female painted turtles (*Chrysemys picta marginata*) collected in a single day in the St Lawrence valley. This is a rather artificial example, since the dimensionality is already low ($n = 24$ and $p = 3$), but it illustrates some of the calculations and interpretation. The sample variance matrix is symmetric with values $S = \begin{pmatrix} 451.39 & 271.17 & 168.70 \\ * & 171.73 & 103.29 \\ * & * & 66.65 \end{pmatrix}$, with trace 689.77. Then

	a_1	a_2	a_3
length	0.8126	-0.5454	-0.2054
width	0.4955	0.8321	-0.2491
height	0.3068	0.1008	0.9465
variance (λ_j)	680.4	6.5	2.86

Checks: the sum of variances is 689.76, and we can check easily that $Sa_i - \lambda_i a_i = 0$.

The first component accounts for 98.64% of the total variance. This is typical of data on *sizes* of items, and it reflects variation in overall sizes of the turtles. Its value for any particular turtle (or score on the first principal component) is

$$Y_1 = 0.81 \times \text{length} + 0.50 \times \text{width} + 0.31 \times \text{height},$$

and reflects the general size. The second principal component is

$$Y_2 = -0.55 \times \text{length} + 0.83 \times \text{width} + 0.10 \times \text{height},$$

and this component is dominated by variations in length and width, and is therefore a contrast between them. High values of Y_2 will come from nearly round shells and low values from elongated elliptical shells. So Y_2 is a measure of roundness. Finally,

$$Y_3 = -0.21 \times \text{length} - 0.25 \times \text{width} + 0.95 \times \text{height},$$

so this contrasts height against (length + width), so is a measure of how peaked or pointed the shell is. The conclusions are that the shells vary most in overall size, next most in shape, and lastly in peakedness.

Exercise 3.6 Verify the calculations in R (remember that you don't need the full data set, just the sample variance matrix).

Exercise 3.7 Repeat the calculations in R using the correlation matrix.

3.10 Example: Chicken dimensions (Wright)

The data for this example are given in Wright (1954), "The interpretation of multivariate systems", in *Statistics and Mathematics in Biology* (State University Press, Iowa, eds. O.Kempthorne, T.A.Bancroft, J.W.Gowen, J.L.Lush), 11–33.

Six bone measurements $x_1, \ldots, x_6$ were made on each of 275 white leghorn fowl. These were x_1: skull length, x_2: skull breadth, x_3: humerus, x_4: ulna, x_5: femur, x_6; tibia (so the first two were skull measurements, the next two were wing measurements, and the last two were leg measurements). The table below gives the coefficients of the six principal components calculated from the variance matrix.

variable	$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$
x_1	0.35	0.53	0.76	−0.04	0.02	0.00
x_2	0.33	0.70	−0.64	0.00	0.00	0.03
x_3	0.44	−0.19	−0.05	0.53	0.18	0.67
x_4	0.44	−0.25	0.02	0.48	−0.15	−0.71
x_5	0.44	−0.28	−0.06	−0.50	0.65	−0.13
x_6	0.44	−0.22	−0.05	−0.48	−0.69	0.17

To interpret these coefficients, we will round them heavily to either just one decimal place, and ignore values close to 0, giving:

variable	$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$
x_1	0.4	0.6	0.7	0	0	0
x_2	0.4	0.6	−0.7	0	0	0
x_3	0.4	−0.2	0	0.5	0	0.7
x_4	0.4	−0.2	0	0.5	0	−0.7
x_5	0.4	−0.2	0	−0.5	0.6	0
x_6	0.4	−0.2	0	−0.5	−0.6	0

and then consider the signs:

variable	$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$
x_1	+	+	+			
x_2	+	+	−			
x_3	+	−		+		+
x_4	+	−		+		−
x_5	+	−		−	+	
x_6	+	−		−	−	

We can then see that the first component a_1 is proportional to the sum of all the measurements. Large fowl will have all x_i large, and so their scores on the first principal component $y_1 (= x'a_1)$ will be large, similarly small birds will have low scores of y_1. If we produce a scatter plot using the first principal component aas the horizontal axis, then the large birds will appear on the right-hand side, and small ones on the left. Thus the first principal component measures *overall size.*

The second component is of the form (skull) − (wing + leg), and so high positive scores of $y_2 (= x'a_2)$ will come from birds with large heads and small wings and legs. If we plot y_2 against y_1, then the birds with relatively small heads for the size of their wings and legs will appear at the bottom, and those with relatively big heads at the top. So the second principal component measures *overall body shape.*

The third component is a measure of *skull shape* (i.e., skull width against skull length). the fourth is wing size compared with leg size, so is another measure of *body shape* (but one not involving the head). The fifth and sixth are contrasts between upper and lower parts of the wing and leg respectively, so y_5 measures *leg shape* and y_6 measures *wing shape.*

For comparison, we see the effect of using the correlation matrix for the principal component analysis instead of the variance matrix. The correlation matrix is:

	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1.000	0.505	0.569	0.602	0.621	0.603
x_2	0.505	1.000	0.422	0.467	0.482	0.450
x_3	0.569	0.422	1.000	0.926	0.877	0.878
x_4	0.602	0.467	0.926	1.000	0.874	0.894
x_5	0.621	0.482	0.877	0.874	1.000	0.937
x_6	0.603	0.450	0.878	0.894	0.937	1.000

Then we get the principal components as:

variable	$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$
x_1	0.35	0.40	0.85	0.05	0.01	−0.03
x_2	0.29	0.81	−0.50	0.02	0.01	−0.04
x_3	0.44	−0.26	−0.11	0.50	0.60	−0.33
x_4	0.45	−0.20	−0.10	0.47	−0.60	0.41
x_5	0.45	−0.16	−0.10	−0.50	0.37	0.58
x_6	0.45	−0.21	−0.07	−0.47	−0.38	−0.62
eigenvalue	4.46	0.78	0.46	0.17	0.08	0.05

Comparing these results with those from the variance matrix, note that the overall interpretation of the individual components will be exactly the same, except perhaps the last two.

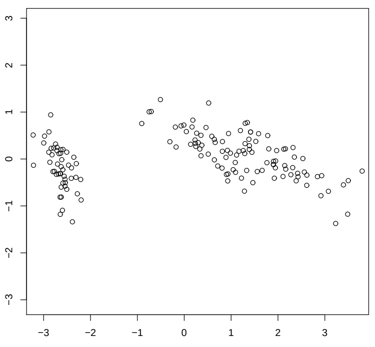
Note that the sum of the eigenvalues is 6.0 (the correlation matrix had trace 6.0, since it is a 6×6 matrix with 1 in each diagonal entry). The first component accounts for almost 75% of the variance of the data – strictly speaking, it is 75% of the standardised data, and not the original data. In fact, if the analysis were based on the variance matrix, then the first principal component would appear even more dominant; this is typical with measurements of physical sizes. For dimensionality reduction in particular, it is sensible to base assessments on the analysis of the correlation matrix, or else look at the proportions of variance explained as a total of all the components *excluding* the first (you may not care about size so much as differences in species, for example, where differences in the shape of the animal may be more important than the size, which may just be an indicator of the age of the animal). The first three components account for 95% of the variance of the standardised data, and this might be a satisfactory initial reduction to these three components.

(Note that different packages may reverse all the signs in one of the columns; recall that if x_i is an eigenvector, so is $-x_i$, so there is some arbitrariness in the sign.)

3.11 Remarks

- "Experience" shows that if you include a few discrete variables with several continuous ones (or if you have a large number of discrete ones without any continuous ones), then principal component analysis based upon the correlation matrix works, i.e., you obtain interpretable answers.
- Since the key objective of principal components analysis is to extract information, i.e., to *partition the internal variation*, it is sensible to plot the data using equal scaling on the axes. This is easy in R or S-PLUS using the `MASS` function `eqscplot()`:

```
eqscplot(ir.pc[,1:2])
```



  Note that unlike `plot()`, the axes are not automatically labelled by `eqscplot()` and you need to do this by including `xlab="first p.c.",ylab="second p.c."` in the call to it.

- *Supplementary data* (i.e., further data not included in calculating the principal component transformation but measured on the same variables) can be displayed on a PCA plot (i.e., apply the same transformation to the new data). This idea is particularly of use in related techniques such as discriminant and correspondence analysis. This can be handled in R with the function `predict()`; see the help system for details.
- Numerical interpretation of loadings is only viable with a small number of original variables. In many modern examples with several hundred variables, other techniques have to be used, e.g., plot loadings against variable number (assuming that there is some structure to the variable numbering, e.g., digitising spectra). This may reveal, for example, that the first few principal components are combinations from the middle of the spectra, next few from one third of the way along.

- PCA is obtained by choosing projections to maximise variances of projected data. Choosing a different criterion can give interesting views of data: *projection pursuit methods*, e.g., maximise projected kurtosis (see e.g., Venables and Ripley) and *independent component analysis*.

3.12 Non-linear techniques

Linear PCA attempts to find a linear coordinate system which is in accordance with the data configuration; it is particularly useful if the data contain a linear singularity or near linear singularity.

If the data contain a non-linear singularity, then linear principal component analysis may fail to find it, and a more general technique is required. The technique is similar to polynomial regression, and its relationship with multiple regression.

Example 3.8 Suppose that $p = 2$ and we seek quadratic principal components.

Given data $x = (x_1\ x_2)'$, the first step is to find $z = ax_1 + bx_2 + cx_1^2 + dx_1x_2 + ex_2^2$ such that the variance of z is maximal among all such quadratic functions of x_1, x_2.

Let $x_3 = x_1^2$, $x_4 = x_1x_2$ and $x_5 = x_2^2$. If $x^* = (x_1, \ldots, x_5)'$ and $a^* = (a, b, c, d, e)'$, then the problem is to maximise $\text{var}(a^{*\prime}x^*)$ just as in the linear case.

For general p, augment the original p variables by $p(p+1)/2$ derived ones, and perform PCA on the new set of $p + p(p+1)/2$ variables. This is only practical for small p since you need to have more than $p(p+1)/2$ negligible eigenvalues for the dimensionality of the original problem to be reduced.

It would in principle be possible to define new augmenting variables such as $\sin x_i$ or more complicated functions just as with multiple regression. However, determining which such functions are appropriate is not easy, unlike multiple regression with simple plotting of residuals, etc., available, and it is not a routine technique except in special cases where there is some theoretical suggestion from the nature of the problem. For example, in a study on dimensions of items of timber with circumference of trunk and length of trunk both included as measurements, there may be some reason for thinking that the volume of the timber may be roughly constant, so including a new augmenting variable defined as length $\times$ circumference^2 could be useful. However, such situations are rare in practice.

3.13 Further reading

PCA is the most widely used dimensionality reduction technique, but there are others which may be of interest. We shall say nothing about these techniques here, beyond a very short description; more details are easily accessible via an internet search, for example, or in some of the textbooks on the subject.

Factor analysis is a method which supposes that the p observed variables are combinations of q underlying variables, at least approximately. Then the aim is to determine the "best" choice of the underlying variables. Again, the value of q is open to choice; unlike PCA, the results for a given value of q will differ completely if $q + 1$ is used instead. This method is widely used in social science, where researchers have tried to explain observed numerical variables by assuming the existence of some underlying variables which aren't measurable (e.g., "intelligence"), but its use has sometimes been controversial.

Projection pursuit takes another viewpoint on looking at the data. In PCA, what we are essentially doing is to choose views of the data which are as wide as possible, and decreasing. (Imagine that the data is essentially an ellipsoid – then the first principal component is the longest axis, and the view of the data is orthogonal to this axis.) In projection pursuit, one looks for the views of the data which are most interesting, in the sense that they are as *nongaussian* as possible. This is measured by a quantity called *kurtosis.*

Independent Component Analysis looks for components that are *statistically independent* and *nongaussian.* It is used often in signal processing techniques, and is closely related to a process known as *blind source separation* in that field. As usual, we suppose that the observed data is a linear combination of variables; here we try to make the variables chosen as independently as possible.

Increasingly, techniques from data science and machine learning are making their way into this area. Look on Wikipedia for some data mining techniques: *(Kohonen's) self-organising maps*, *generative topographic mapping*, etc., and follow links from there.

4 Dimensionality reduction II: Multidimensional Scaling (MDS)

4.1 Introduction

Multidimensional scaling is another technique that aims to plot data in a small number of dimensions. We have already seen principal component analysis, where the aim was to plot the points with as little loss of information as possible. The aim of multidimensional scaling is to plot the points with as little distortion as possible, in some sense.

Our data sets thus far have appeared as collections of observations for each subject. However, in practice, data sets are often given in terms of *similarities* or *dissimilarities* between n observations, and the resulting $n \times n$ matrix is known as a *proximity matrix.*

A nice example, which we will consider more below, is given in a paper of G.Ekman, "Dimensions of Color Vision", Journal of Psychology **38** (1954). Here, Ekman conducted an experiment where subjects looked at a screen with two circular glass windows, lit by projectors into which colour filters could be placed. Fourteen colour filters were used, trasmitting light of increasing wavelength; 31 subjects were asked to rate how similar the colours were on a five-point scale (0 =no similarity, 4 =identical). After averaging, the similarities were divided by 4 to lie within the unit interval. The data set can be loaded into R with `data(ekman)` (with the library `smacof`). Here is the proximity matrix (it is symmetric with 1s along the diagonal):

```
> library(smacof)
> data(ekman)
> ekman
     434  445  465  472  490  504  537  555  584  600  610  628  651
445 0.86
465 0.42 0.50
472 0.42 0.44 0.81
490 0.18 0.22 0.47 0.54
504 0.06 0.09 0.17 0.25 0.61
537 0.07 0.07 0.10 0.10 0.31 0.62
555 0.04 0.07 0.08 0.09 0.26 0.45 0.73
584 0.02 0.02 0.02 0.02 0.07 0.14 0.22 0.33
600 0.07 0.04 0.01 0.01 0.02 0.08 0.14 0.19 0.58
610 0.09 0.07 0.02 0.00 0.02 0.02 0.05 0.04 0.37 0.74
628 0.12 0.11 0.01 0.01 0.01 0.02 0.02 0.03 0.27 0.50 0.76
651 0.13 0.13 0.05 0.02 0.02 0.02 0.02 0.02 0.20 0.41 0.62 0.85
674 0.16 0.14 0.03 0.04 0.00 0.01 0.00 0.02 0.23 0.28 0.55 0.68 0.76
```

Note that the row and column headings refer to the frequency of the transmitted light in μm.

Examples 4.1 Many sets of data can be given in this way.

- We could measure the "distance" between places by measuring the physical distance, or the distance along motorways, or the length of time of train journeys;
- The "distance" between two Morse code symbols could be the number of times one Morse code symbol is mistaken for another;

- A very coarse measure of distance between two départements of France could be $\delta_{ij} = 1$ if they have a common border and $\delta_{ij} = 0$ otherwise;
- To compare two different prehistoric graves which (potentially) contain artefacts of M types, we could mark 0 if the artefact is absent from the grave and 1 if it is present, and define δ_{ij} as the number of artefacts in common between grave i and grave j;
- Finally, any data set of p observations on n subjects can be converted to a proximity matrix given an appropriate way to measure the distance between observations. A natural choice is to take the Euclidean distance between the observations, regarding them as points in $\mathbb{R}^p$, and taking the usual square root of sum of squares of differences. This suffers from the same issue of units that happened in PCA; it would be better to normalise the variables so that each had unit variance, and still better is to use the squared statistical distance (Mahalanobis distance) that we introduced earlier. In general, there are many ways to convert $n \times p$ data matrices into $n \times n$ proximity matrices.

Measures of similarity can be converted to measures of dissimilarity (and vice versa) in many obvious ways – subtraction, taking reciprocals, etc.

The aim of multidimensional scaling is to get some visualisation of this sort of data, by plotting points on a map whose Euclidean distances closely reflect the similarities between the points. For example, in the proximity matrix above, the colours with frequencies 434μm and 445μm seem to be much closer than the colours with frequencies 465μm and 600μm say. We try to plot the points in some Euclidean space in such a way that their distances reflect the proximity in the original data.

That is, we are considering data where the relationships between the points are the most important features.

Just like PCA, we need to choose an appropriate dimension for our map, but visualising the data becomes easier if we can fit the data quite well into 2 or 3 dimensions.

We choose a dimension k, and try to find a set of points in k-dimensional space whose distances approximate the distances in the distance matrix of the original data. This is necessarily an approximation in general; there may be no configuration of points in k-dimensional space with exactly these distances. (For example, if we had a data set consisting of 4 observations, all at a distance of 1 from each other, we would not be able to view these in 2-dimensional space exactly, as there is no configuration of points with these distances from each other.) However, if we can find some approximation, we can hope to retain enough of the structure of the data that we can still draw suitable conclusions – this is particularly interesting if k is small enough that we can genuinely visualise the data in k dimensions.

Thus we consider the following problem:

> Given an $n \times n$ (symmetric) matrix (δ_{ij}) of dissimilarities, can a configuration of points be found in Euclidean k-space (where k is open to choice) such that the calculated distance matrix (d_{ij}) reasonably matches the given dissimilarity matrix (δ_{ij})?

The answer is generally yes for sufficiently large k (e.g., $k = n - 1$, with some restrictions on the δ_{ij}). The interest is when it can be done for very small k, for example, with $k \leq 5$.

In this case, the data approximately lives in these k dimensions, and one can hope to visualise the data more easily.

The objectives of multidimensional scaling analysis can be any or some of:

- To learn more about the measure of dissimilarity itself. We have already mentioned Ekman's example – we might hope to understand the factors which cause the brain to confuse different colours.
- To discover some underlying structure in the data themselves. For example, if the dissimilarities can be closely matched by a string of points lying on a line, can the distance along the line be interpreted as some variable of interest such as time in the prehistoric graves example?
- To discover whether the units divide "naturally" into groups – these techniques are widely used in market research for "market segmentation"; do the customers divide into different target groups that are reached by different forms of advertising?
- To discover some "gap" in the market that can be filled by a new product.

The available techniques for multidimensional scaling are mostly *ad hoc* methods – i.e., based on intuition rather than solid mathematical theory – and are mostly non-metric (i.e., use only the orderings of the given dissimilarities and not their actual absolute values). The one exception is the "classical solution" which is often used as the initial trial solution for the start of one of the iterative methods.

4.2 Metric methods

In metric multidimensional scaling, the aim is to find a configuration of points whose Euclidean distances strongly resemble the original proximity matrix

4.2.1 Principal co-ordinates analysis, or Classical metric MDS

First, we assume that the $n \times n$ proximity matrix is a matrix of Euclidean distances between points in $\mathbb{R}^k$. We shall explain how to find the n points. (Notice that distances are preserved under various operations on $\mathbb{R}^k$, such as translation, reflections, rotations, etc., so the solution will not be unique.)

Suppose that $D = (\delta_{ij})$ is an $n \times n$ distance matrix, coming from a data matrix X'. Then we define $B = X'X$. Notice that both D and B are $n \times n$ matrices, and that $b_{ij} = x_i'x_j = \sum_{k=1}^p x_{ik}x_{jk}$. Further B is a positive semidefinite matrix; given an n-vector y, $y'By = y'X'Xy = (Xy)'(Xy)$, the squared length of Xy, which is non-negative.

Then

$$\begin{aligned}\delta_{ij}^2 &= \sum_{k=1}^p (x_{ik} - x_{jk})^2 \\ &= \sum_{k=1}^p x_{ik}^2 + \sum_{k=1}^p x_{jk}^2 - 2\sum_{k=1}^p x_{ik}x_{jk} \\ &= b_{ii} + b_{jj} - 2b_{ij}.\end{aligned}$$

Thus, given a data matrix, we can form the positive semidefinite matrix $B = X'X$, and can easily compute the distance matrix.

Conversely, if we have a distance matrix (δ_{ij}), we want to try to find a data matrix X' with these distances. Firstly, we try to find a matrix B related to D as above. This is equivalent to writing b_{ij} in terms of the δ_{ij}^2. As already remarked, if X' exists, we can find it where all the columns add to 0, and all the rows add to 0, simply by translating, so we will assume that $\sum_{i=1}^n x_{ij} = 0$ for all j and that $\sum_{j=1}^n x_{ij} = 0$ for all i. It is elementary (but quite long!) to invert the system of equations $\delta_{ij}^2 = b_{ii} + b_{jj} - 2b_{ij}$ to get

$$b_{ij} = -\tfrac{1}{2}\left[\delta_{ij}^2 - \delta_{i.}^2 - \delta_{.j}^2 + \delta_{..}^2\right],$$

where

$$\begin{aligned}
\delta_{i.}^2 &= \tfrac{1}{n}\sum_{k=1}^n \delta_{ik}^2 \\
\delta_{.j}^2 &= \tfrac{1}{n}\sum_{k=1}^n \delta_{kj}^2 \\
\delta_{..}^2 &= \tfrac{1}{n^2}\sum_{k,l=1}^n \delta_{kl}^2.
\end{aligned}$$

Thus given the distance matrix D, we can recover the matrix B as above. Now we want to write $B = X'X$ for some data matrix X. It turns out that there is an easy way to do this in terms of the eigenvectors of B.

We simply compute the non-zero eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p > 0$ of B, and let $e_1, \ldots, e_p$ denote the corresponding eigenvectors of unit length. Let $f_i = \sqrt{\lambda_i} e_i$, and let X' be the matrix whose columns are f_i.

Let's check that $B = X'X$. We can work out Be_i and $X'Xe_i$ and check that we get the same thing for each i:

$$X'Xe_i = X'\begin{pmatrix} f_1' \\ \vdots \\ f_i' \\ \vdots \\ f_p' \end{pmatrix} e_i = X'\begin{pmatrix} 0 \\ \vdots \\ \sqrt{\lambda_i} \\ \vdots \\ 0 \end{pmatrix} = \sqrt{\lambda_i} f_i = \lambda_i e_i.$$

But $Be_i = \lambda_i e_i$ too, and the e_i form a basis of $\mathbb{R}^p$, so $B = X'X$.

This proves:

Theorem 4.2 *D is a matrix of Euclidean distances if and only if B is positive semidefinite (equivalently, if all eigenvalues of B are non-negative).*

The above result gives a practical way to find a configuration of points with (approximately) a given data matrix, Given a distance matrix D,

1. Find the matrix B (computationally, it works out as HAH, where A is the matrix $(-\frac{1}{2}\delta_{ij}^2)$ and H is the matrix $I_n - \frac{1}{n}J_n$, where J_n is the $n \times n$ matrix with all entries equal to 1).
2. Find the eigenanalysis of B.

3. Transpose the matrix of eigenvectors.
4. Take the columns of this transposed matrix as the *principal coordinates* of the points.

If the eigenvalues are all non-negative, then the distance matrix will be an exact match. If some are negative, then (pragmatically) take just the positive ones and the match will be approximate. Eigenvectors taken in order of magnitude of the positive eigenvalues will give increasingly better approximations to the desired configuration of points.

Aside 4.3 For those interested, here is the inversion mentioned above.

Let's evaluate $n^2(\delta_{ij}^2 - \delta_{i.}^2 - \delta_{.j}^2 + \delta_{..}^2)$. We have:

$$\begin{aligned}
& n^2(\delta_{ij}^2 - \delta_{i.}^2 - \delta_{.j}^2 + \delta_{..}^2) \\
= \; & n^2\delta_{ij}^2 - n\sum_{k=1}^{2}\delta_{ik}^2 - n\sum_{k=1}^{n}\delta_{kj}^2 + \sum_{k,l=1}^{n}\delta_{kl}^2 \\
= \; & n^2(b_{ii} + b_{jj} - 2b_{ij}) - n\left[\sum_{k=1}^{n} b_{ii} + b_{kk} - 2b_{ik}\right] \\
& -n\left[\sum_{k=1}^{n} b_{kk} + b_{jj} - 2b_{jk}\right] + \left[\sum_{k,l=1}^{n} b_{kk} + b_{ll} - 2b_{kl}\right].
\end{aligned}$$

We expand this and simplify as:

$$\begin{aligned}
& n^2 b_{ii} + n^2 b_{jj} - 2n^2 b_{ij} - n\sum_{k=1}^{n} b_{kk} + 2n\sum_{k=1}^{n} b_{ik} - n\sum_{k=1}^{n} b_{kk} \\
& -n\sum_{k=1}^{n} b_{kk} - n^2 b_{jj} + 2n\sum_{k=1}^{n} b_{jk} + n\sum_{k=1}^{n} b_{kk} + n\sum_{l=1}^{n} b_{ll} - 2\sum_{k,l=1}^{n} b_{kl} \\
= \; & -2n^2 b_{ij} + 2n\sum_{k=1}^{n} b_{ik} - n\sum_{k=1}^{n} b_{kk} + 2n\sum_{k=1}^{n} b_{jk} + n\sum_{l=1}^{n} b_{ll} - 2\sum_{k,l=1}^{n} b_{kl} \\
= \; & -2n^2 b_{ij} + 2n\sum_{k=1}^{n} b_{ik} + 2n\sum_{k=1}^{n} b_{jk} - 2\sum_{k,l=1}^{n} b_{kl},
\end{aligned}$$

the last line following as l and k are just dummy variables over which the summation is being done.

Now recall that we are assuming that $B = X'X$, and that the rows of X sum to 0. This means that $X\mathbf{1}_p = \mathbf{0}_n$. So $X'X'\mathbf{1}_p = \mathbf{0}_p$, and so the rows of $X'X$ also sum to 0. For row i, this is the statement that $\sum_{k=1}^{n} b_{ik} = 0$. The same argument applies to $\sum_{k,l=1}^{n} b_{kl}$, which is $\sum_{k=1}^{n}\left(\sum_{l=1}^{n} b_{kl}\right)$, and the terms in the brackets all vanish as above.

Finally, the columns of X' add to 0, i.e., $\mathbf{1}_p'X' = \mathbf{0}_n'$, and a similar argument shows that the columns of B add to 0; this is $\sum_{k=1}^{n} b_{kj} = 0$.

Finally, we see that

$$n^2(\delta_{ij}^2 - \delta_{i.}^2 - \delta_{.j}^2 + \delta_{..}^2) = -2n^2 b_{ij},$$

as required.

In R, there is a built-in function `cmdscale()` to perform classical scaling.

Let's try to find a 1-dimensional space for scaling with our data set

$$(1,4), \quad (3,7), \quad (5,8), \quad (7,11), \quad (9,15).$$

The matrix of δ_{ij}^2 is easily computed as

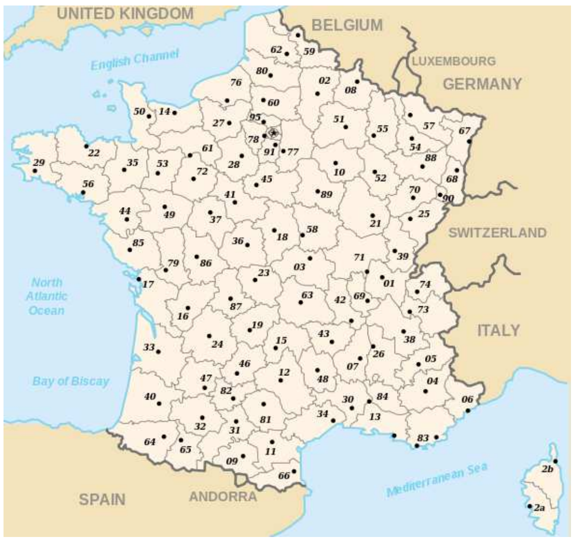
$$\begin{pmatrix} 0 & 13 & 32 & 85 & 185 \\ 13 & 0 & 5 & 32 & 100 \\ 32 & 5 & 0 & 13 & 65 \\ 85 & 32 & 13 & 0 & 20 \\ 185 & 100 & 65 & 20 & 0 \end{pmatrix}.$$

We can then form $B = HAH = \begin{pmatrix} 41 & 18 & 5 & -18 & -46 \\ 18 & 8 & 2 & -8 & -20 \\ 5 & 2 & 1 & -2 & -6 \\ -18 & 8 & -2 & 8 & 20 \\ -46 & -20 & -6 & 20 & 52 \end{pmatrix}$. Three eigenvalues of this matrix are 0, and the other two are $\lambda_1 = 109.12$, with eigenvector $e_1 = \begin{pmatrix} -0.8890 \\ -0.3890 \\ -0.1110 \\ 0.3890 \\ 1 \end{pmatrix}$ and $\lambda_2 = 0.8798$, with eigenvector $e_2 = \begin{pmatrix} 0.4640 \\ 0.9640 \\ -1.4640 \\ -0.9640 \\ 1 \end{pmatrix}$. If we just want a 1-dimensional space, we choose 1 eigenvector (the one with largest eigenvalue), namely e_1. We scale e_1 so that it has square length equal to λ_1, so that e_1 is replaced by $e_1 = \begin{pmatrix} -6.4003 \\ -2.8006 \\ -0.7991 \\ 2.8006 \\ 7.1994 \end{pmatrix}$. Then the principal coordinates of the points are exactly the entries in e_1. Let's see what this gives. We compute the square distances between the values $(e_{1i} - e_{1j})^2$ as

$$\begin{pmatrix} 0 & 12.9578 & 31.3732 & 84.6561 & 184.9516 \\ 12.9578 & 0 & 4.0059 & 31.3732 & 99.9998 \\ 31.3732 & 4.0059 & 0 & 12.9578 & 63.9762 \\ 84.6561 & 31.3732 & 12.9578 & 0 & 19.3496 \\ 184.9516 & 99.9998 & 63.9762 & 19.3496 & 0 \end{pmatrix},$$

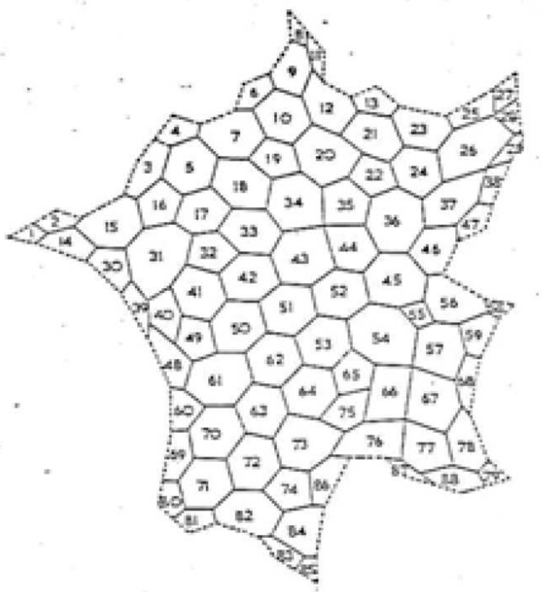
a very reasonable fit with the original distance matrix.

For another example, where the distance matrix is clearly not Euclidean, let's take a map of France, divided into départements:



A very coarse measure of similarity would be to take $\delta_{ij} = 1$ if départements i and j shared a border, and 0 if not. (This is similar to the *adjacency matrix* in graph theory, although here we also take $\delta_{ii} = 1$ as well.)

Applying classical MDS gives the following "map" of the data (where we have rotated and reflected the answer to be able to compare with the map above, and replaced vertices by suitable polygons surrounding them):



4.2.2 Implementation in R

In R, type `cmdscale()` (in the `MASS` library).

Let's look at Ekman's data in R. After typing

```
> library(MASS)
> library(smacof)
> data(ekman)
```

to load the `MASS` library (for the `cmdscale()` function) and the `smacof` library (for the `ekman` data set), and to read in the `ekman` data set into the path, type

```
> ekman.cmds<-cmdscale(1-ekman)
```

to store the result in `ekman.cmds`.

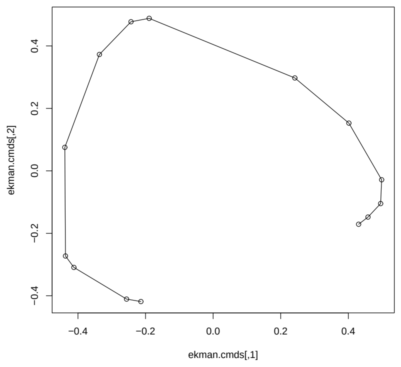
- Note that Ekman's data is a matrix of *similarities*, whereas `cmdscale()` uses *dissimilarities*, so we need to convert the data (subtracting every entry from 1 is appropriate with Ekman's data) before applying `cmdscale`.

  To see the results, type

```
> plot(ekman.cmds)
> lines(ekman.cmds)
```

which plots the points and joins neighbouring entries in the data matrix with lines:

```
> plot(ekman.cmds)
> lines(ekman.cmds)
```



Typing `cmdscale(1-ekman,eig=TRUE)` also gives the eigenvalues as

$$1.98, 1.30, 0.44, 0.37, 0.16, 0.10, 0.04, 0.03, 0.02, 0.00, 0.00, 0.00, -0.03, -0.05$$

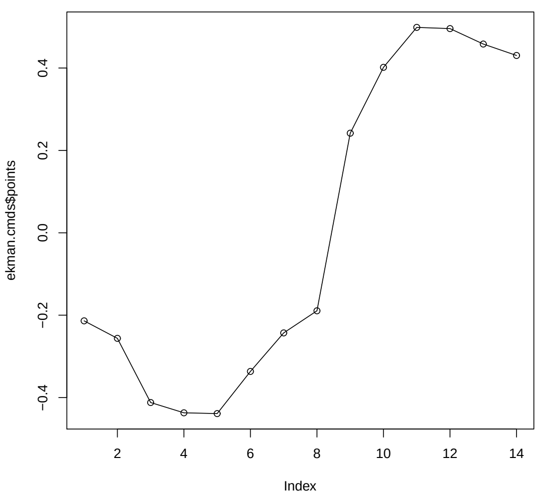
and the existence of negative eigenvalues tells you that the distance matrix is not Euclidean.

Thus any plot (in any number of dimensions) will have to have some distortion.

The command `cmdscale()` uses `k` for the number of dimensions. The default number of dimensions for `cmdscale()` is $k = 2$ (for ease of visualisation), but one can change this. Choosing $k > 2$ will make visualisation harder, but it may well be that the set of points is still of interest.

For example, `ekman.cmds<-cmdscale(1-ekman,k=1)` gives the best 1-dimensional map of the data.

This graph plots the points against the indexes (which are in order of the frequencies).



You can see that the 1-dimensional graph folds over itself; this led Ekman to conclude that human perception of colour is more than 1-dimensional.

We'll look at more examples later.

4.2.3 Duality with PCA

If we start with a data set and then calculate the Euclidean distance matrix for all the points (with function `dist()`) and then apply principal co-ordinate analysis, i.e., classical metric scaling (with `cmdscale()`), then we obtain precisely the same results as principal component analysis, except perhaps for arbitrary changes of sign of one or more axes, though we do not automatically have the additional information on proportions of variance or loadings available.

Indeed, suppose that we have a mean-centred data matrix X'. Then the variance matrix is essentially $S = XX'$, while the analysis above is an eigenanalysis of $B = X'X$. However, if e_i is an eigenvector of $X'X$, then we claim that Xe_i is an eigenvector of XX' with the same eigenvalue:

$$(XX')(Xe_i) = X(X'Xe_i) = X(\lambda e_i) = \lambda_i(Xe_i).$$

So the non-zero eigenvalues of the two matrices S and B are the same, and, further, the eigenvectors of S are simply obtained from the eigenvectors of B by multiplying by X. Similarly, the eigenvectors of B are obtained from the eigenvectors of S by multiplying by X'.

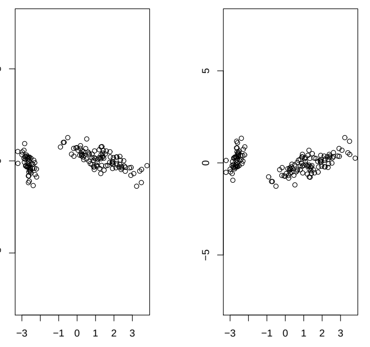
Suppose that $a_i = Xe_i$ denotes the eigenvector of S corresponding to eigenvalue λ_i. Then $X'a_i$ is the vector giving the component scores for the ith principal component, and the sum of squares of $X'a_i$ is λ_i. So if $\widehat{e}_i = e_i/\|e_i\|$ and $\widehat{a}_i = a_i/\|a_i\|$ are normalised to have unit length, then

$$X'\widehat{a}_i = \lambda_i\widehat{e}_i.$$

So instead of calculating the component scores by transforming the eigenvectors of $S = XX'$, we can get them directly from the eigenvectors of $B = X'X$.

You might like to verify that in our simple example, the distances between the images of the original points under the PCA reduction to 1-dimension are exactly the distances (up to rounding errors, at least) given by the principal co-ordinate analysis above.

```
> library("MASS")
> ir<-iris[,-5]
> ir.pr<-prcomp(ir)
> ir.scal<-cmdscale(dist(ir))
> ir.pc<-predict(ir.pr)
> par(mfrow=c(1,2))
> library("MASS")
> eqscplot(ir.pc)
> eqscplot(ir.scal)
```



The `cmdscale()` plot on the right is identical to the PCA plot on the left apart from a reflection about the vertical axis.

4.2.4 Another metric method

To motivate the non-metric method, here is another way of performing this sort of analysis.

We recall that we are given a matrix (δ_{ij}) of distances between n points. We are looking for a small dimensional set of points whose distances resemble this matrix – in order words, we are looking for a new data matrix with fewer variables which is close to the original matrix in that it gives a very similar distance matrix.

If we have a guess for such a set, where the associated distance matrix is (d_{ij}), then we could measure the closeness of the approximation using sums of squares. However, this depends on the units used to measure the distances, so is not ideal. Instead, we normalise the sum of squares and then (like the standard deviation) take the square root so that it is in the same units as the original distances. This gives a measure

$$S = \sqrt{\frac{\sum_{i<j}(d_{ij} - \delta_{ij})^2}{\sum_{i<j} d_{ij}^2}}$$

of how well the guess fits the distance matrix. This measure is called the *stress*. There are other alternative measures of stress (e.g., one could replace d_{ij} with δ_{ij} in the denominator). The hope is that the stress is close to 0, which would indicate that the guess is a good one.

From an original guess, the idea is to perturb the points until a minimum is reached. This may be a local minimum rather than the global minimum, but repeating the method with other initial starting points or different perturbations may give a number of different answers from which the best may be selected.

Generally a steepest descent method is used in the perturbation step, and the process ends when the stress is at a minimum (or when it appears that further perturbation only reduces the stress by some small pre-specified tolerance level).

4.3 Non-metric methods: ordinal MDS

Often it is not the exact value of δ_{ij} that matters, but its relative value compared with other distances. In social science experiments, for example, subjects may be asked to give a subjective assessment, and then we are only really interested in the relative ordering of the distances.

In the example of Ekman's data, the translation of 'completely different' to 'identical' onto the scale 0–4 is arbitrary; the answers given are subjective, etc.

What really matters in the data is the relative sizes of the similarities. It does seem reasonable to suggest that one pair of colours are more similar than another if the similarity coefficients Ekman computes are larger.

In non-metric, or ordinal, multidimensional scaling, our aim is to find configurations of points whose distances d_{ij} have the same relative ordering as the δ_{ij}(at least weakly, i.e., $\leq$ if not actually $<$).

In outline (just as for the description of the metric method above), the basis of all the techniques is to obtain a trial configuration of points, calculate the distance matrix (d_{ij}) for this configuration, and see how well the ordering of the d_{ij} matches that of the given δ_{ij}, then to perturb the trial configuration to try to improve the match of the orderings.

From the d_{ij}, we construct *disparities* $\hat{d}_{ij}$ such that they are in the same rank order as the δ_{ij}. These form "smoothed" versions of the d_{ij}, and are constructed using *monotonic regression.* That is, given the pairs (δ_{ij}, d_{ij}) with the δ_{ij} increasing, we try to fit a monotonic curve to the points while making the sum of the squares of the vertical deviations as small as possible. This gives a curve passing through the points $(\delta_{ij}, \hat{d}_{ij})$. If the d_{ij} are in the same order as the δ_{ij}, we will have $\hat{d}_{ij} = d_{ij}$. Generally, however, the orderings will differ, and there will be pairs with $\hat{d}_{ij} \neq d_{ij}$. So we are interested in how close the distances d_{ij} are to the disparities $\hat{d}_{ij}$, and we can measure the fit by replacing δ_{ij} with $\hat{d}_{ij}$ in our measures of stress. Therefore we replace the stress function above with the stress function

$$S = \sqrt{\frac{\sum_{i<j}(d_{ij} - \hat{d}_{ij})^2}{\sum_{i<j} d_{ij}^2}},$$

and try to make this smaller; we try to minimise this stress function (or some variant).

Example 4.4 Take a 4×4 distance matrix of dissimilarities

$$\begin{pmatrix} 0.0 & 0.3 & 0.5 & 0.9 \\ 0.3 & 0.0 & 0.4 & 0.8 \\ 0.5 & 0.4 & 0.0 & 0.6 \\ 0.9 & 0.8 & 0.6 & 0.0 \end{pmatrix}$$

and suppose that we have 4 points with $d_{12} = 0.5$, $d_{13} = 0.9$, $d_{14} = 1.0$, $d_{23} = 0.4$, $d_{24} = 1.1$, $d_{34} = 1.0$.

Then we can make the table:

	δ_{12}	δ_{23}	δ_{13}	δ_{34}	δ_{24}	δ_{14}
	0.3	0.4	0.5	0.6	0.8	0.9
d_{ij}	0.5	0.4	0.9	1.0	1.1	1.0
$\hat{d}_{ij}$	0.45	0.45	0.9	1.0	1.05	1.05

The first row consists of the ordered list of δ_{ij}, and the second of their values. Then the third gives the corresponding list of d_{ij}, before the fourth gives the monotone least squares best fit, i.e., a non-decreasing list of values such that $\sum(d_{ij} - \hat{d}_{ij})^2$ is minimised.

From this configuration, we would iteratively improve it by moving points short distances so that the stress slightly decreases each time. When further changes to the configuration do not reduce the stress, the process ends.

Whichever stress function we select (and we shall discuss other choices), we choose the $\hat{d}_{ij}$ to minimise S subject to the ordering constraint, i.e., the $\hat{d}_{ij}$ are monotonic nondecreasing with the δ_{ij}. Then $S_{\min}$ is a measure of how well the trial configuration matches the original δ_{ij}. If we regard $S_{\min}$ as a function of the coordinates of the n points in k-space defining d_{ij} (i.e., a function of nk variables) then we can minimise stress with respect to these variables (using iterative techniques, as mentioned in the example) and find the least stressful configuration in k-space.

We calculate the minimal stress for $k = 1, 2, 3, \ldots$ and stop when increasing k does not decrease stress very much. This determines the dimensionality of the problem.

Here's a summary of the procedure:

1. Choose an initial configuration, normalised so that it is centred at the origin, and with unit mean square distance from the origin.
2. Find the distances (d_{ij}), compute the disparities $(\hat{d}_{ij})$, and calculate the stress.
3. Consider the stress S as a function of the coordinates defining the points, and compute $\frac{\partial S}{\partial x}$ at the point.
4. If it is 0 (up to rounding errors), the point is at a (possibly local) minimum.
5. Otherwise, move to a new configuration along the path of steepest descent.

Because of the possibility of being at a local minimum rather than a global minimum, one should start with several different initial configurations.

The final value of the stress function is an indicator of how successful the procedure has been.

We carry out this procedure for $k = 1, 2, 3, \ldots$, and stop when increasing k doesn't decrease the optimal stress significantly: we can again use a scree plot for this.

We can use a scree plot to help choose a good value of k, plotting the choice of k against the corresponding values of minimal stress. One might find, for example, that if $k = 1$ (i.e., we try to represent our set with a 1-dimensional map) that the minimal stress is quite high, whereas with $k = 2$, the minimal stress becomes considerably lower, and plotting with $k = 3$, little further improvement is obtained. In this case, we would regard the 2-dimensional map as an appropriate representation of our points.

This is the standard choice of stress function, but others that are used are a straightforward generalisation proposed by Kruskal:

$$S_{\text{Kruskal}} = \left(\frac{\sum_{i<j} \left(\theta(\hat{d}_{ij}) - d_{ij} \right)^2}{\sum_{i<j} d_{ij}^2} \right)^{\frac{1}{2}}$$

for some monotonic increasing function $\theta(\ )$, and a rather different one by Sammon:

$$S_{\text{Sammon}} = \frac{1}{\sum_{i<j} d_{ij}} \sum_{i<j} \frac{(\hat{d}_{ij} - d_{ij})^2}{d_{ij}}.$$

The Sammon measure of stress takes the size of the distances more into account: small dissimilarities are more heavily weighted than large ones, meaning that they tend to be more accurately reproduced, and points tend to be spread out more.

Standard programs are available to minimise S (using monotonic regression algorithms) and hence to determine the optimal configuration. R and S-PLUS functions `isoMDS()` (for Kruskal stress) and `sammon()` (for Sammon stress) in the `MASS` library perform variants of non-metric scaling very easily; the distinction between them is in the form of the stress function used. As can be seen above, the process can be quite computationally intensive.

The general experience is that Sammon mapping produces configurations that are more evenly spread out and Kruskal or classical scaling may produce ones which are more tightly clustered. Which is preferable is open to choice and depend upon the particular application.

Given two solutions to the MDS problem, they can be compared using *Procrustes analysis* (in Greek mythology, Procrustes was a bandit who made his victims fit his bed by stretching their limbs or by cutting them off). This is a method of comparing two shapes by keeping one fixed, and then transforming the second so that it superimposes as closely as possible onto the first; then one can do a sum of squares computation to measure the difference between the two configurations. We shall say no more here.

4.4 Examples

4.4.1 The Ekman data set again

After loading the libraries `smacof` (for the data set) and `MASS` (for the MDS functions), type

```
> ekman.mds<-isoMDS(1-ekman)
```

to get the result in `ekman.mds`.

- Recall that the Ekman data is a matrix of similarities, while `isoMDS` uses dissimilarities – since Ekman's similarities are between 0 and 1, we subtract everything from 1 to get a matrix of dissimilarities before applying `isoMDS`, just as we did for `cmdscale`.

Then type

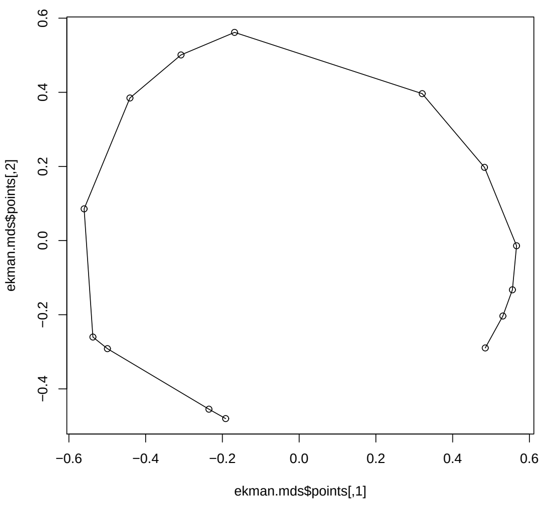
```
> ekman.mds
```

to look at the points; notice that `ekman.mds$points` stores the co-ordinates of the points, and `ekman.mds$stress` gives the stress as 2.922.

Now type

```
> plot(ekman.mds$points)
> lines(ekman.mds$points)
```

to get the picture:

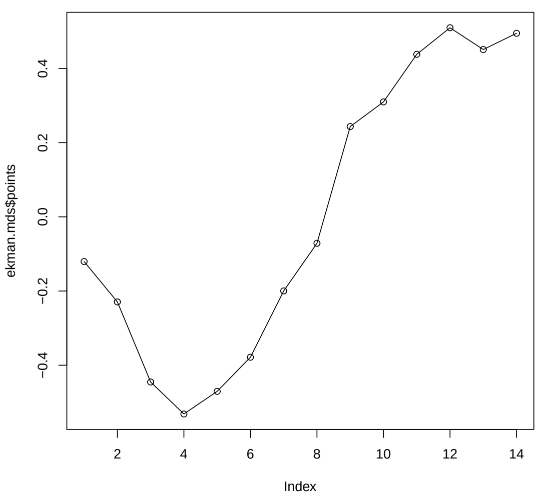


The default number of dimensions for `isoMDS` is 2, since visualisation is one of the key goals of the method.

However, we can change the default with `ekman.mds<-isoMDS(1-ekman,k=1)`, again just like `cmdscale()`.

We get a 1-dimensional view of the data, with stress 28.919.

Here's a picture of the 1-dimensional view. Notice that `isoMDS` doesn't plot it on a single axis, but in the same way as `cmdscale`. Imagine that the plot is squashed onto 1-dimension – and notice that the plot folds over onto itself, and does *not* increase directly with the wavelength.



On the other hand, the 2-dimensional plot consists of a line that really does join points of increasing wavelength.

The command `ekman.mds<-isoMDS(1-ekman,k=?)` also gives the initial stress (using `cmdscale`) and the stress after the iterative process.

Let's tabulate these, so that we can see the improvements that the iterative part of the algorithm brings, and also to give a scree plot of the final stresses, so that we can judge the best number of dimensions to take.

k	1	2	3	4	5	6	7
`cmdscale`	31.85	5.84	4.36	2.98	1.42	1.19	0.82
`isoMDS`	28.92	2.92	1.76	0.75	0.25	0.10	0.08

We could plot these points in a scree plot, to help us choose a dimension with a suitable level of stress. In this case, the default of $k = 2$ seems to produce a pretty reasonable fit.

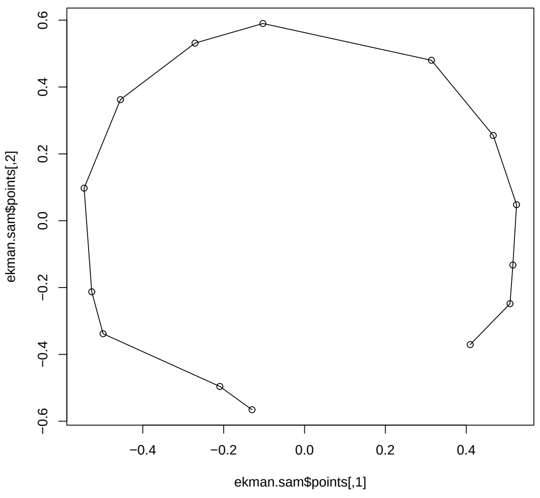
We can repeat the analysis with Sammon scaling.

Type `ekman.sam<-sammon(1-ekman)`. This stores additional information, but the points of the configuration are in `ekman.sam$points`.

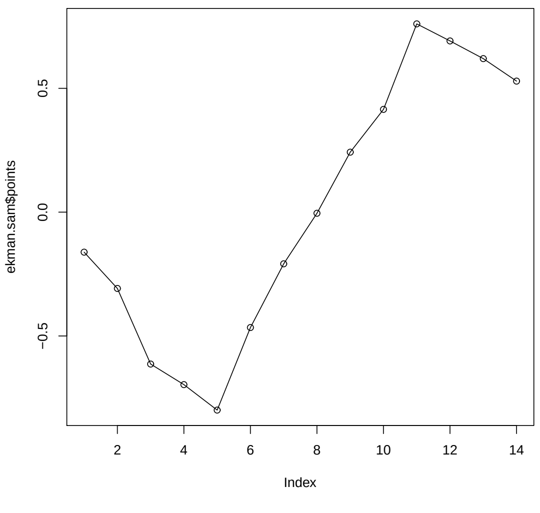
We can plot them as before:

```
plot(ekman.sam$points)
lines(ekman.sam$points)
```

to get the picture:



and with `ekman.sam<-sammon(1-ekman,k=1)`, we get



Comparing the graphs produced by `isoMDS` and by `sammon` gives some mild evidence that Sammon stress does produce more evenly spaced points.

As before, we can choose an appropriate dimension by computing the stress (in different units to the earlier table):

k	1	2	3	4	5
initial	0.282	0.064	0.027	0.010	0.003
`sammon`	0.176	0.022	0.006	0.002	0.000

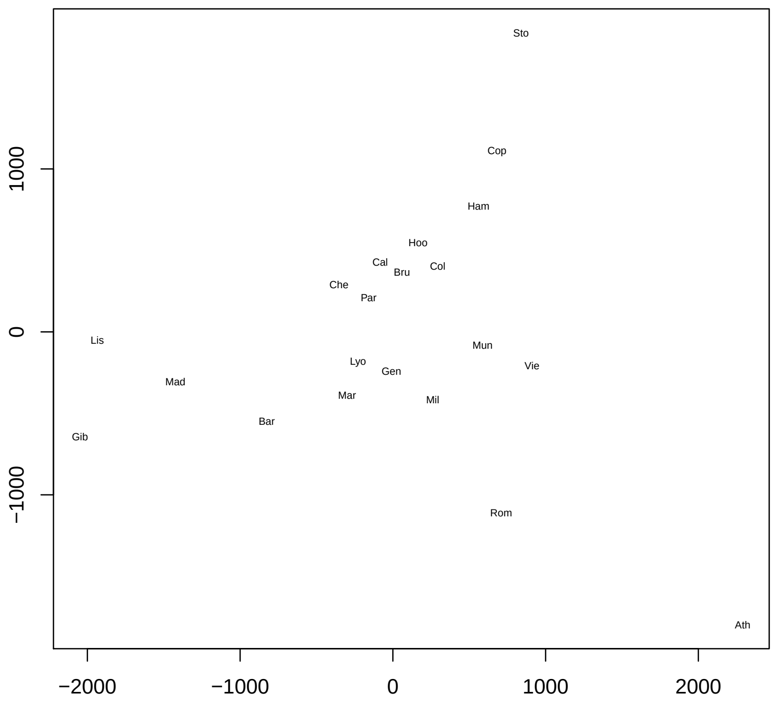
4.4.2 Road distances between European cities

The data set `eurodist` (in the `datasets` library) gives the distances by road in kilometres between 21 cities in Europe, and is given below. The analysis is in R:

```
                Athens Barcelona Brussels Calais Cherbourg Cologne Copenhagen
Barcelona         3313
Brussels          2963      1318
Calais            3175      1326      204
Cherbourg         3339      1294      583    460
Cologne           2762      1498      206    409       785
Copenhagen        3276      2218      966   1136      1545     760
Geneva            2610       803      677    747       853    1662       1418
...
```

Then we can apply classical scaling, and plot the results:

```
> data(eurodist)
> eurodist.cmds<-cmdscale(eurodist)
> eurodist.cmds
                       [,1]        [,2]
Athens          2290.274680  1798.80293
Barcelona       -825.382790   546.81148
Brussels          59.183341  -367.08135
Calais           -82.845973  -429.91466
Cherbourg       -352.499435  -290.90843
Cologne          293.689633  -405.31194
Copenhagen       681.931545 -1108.64478
Geneva            -9.423364   240.40600
[...]
> x<-eurodist.cmds[,1]
> y<- -eurodist.cmds[,2]
> plot(x,y,type="n",xlab="",ylab="")
> text(x,y,names(eurodist),cex=0.5)
```



This reproduces the geography of Europe very closely (it needs a slight rotation) and suggests that the technique itself works, and so can be applied to other data where we don't know what the answer is beforehand, such as the next example, which resembles Ekman's data.

4.4.3 Confusion between Morse Code signals

The data are derived from a set presented by Rothkopf (1957). The original data give the percentages of times that 598 subjects responded "Same!" when two symbols were transmitted in quick succession, i.e., when symbol i (row) was transmitted first followed by symbol j (column). The original matrix (for all 26 letters and 10 numerals) is not hard to find online. These are *similarity matrices*. The file `morsefull.Rdata` gives a symmetric version of a *distance matrix* and indicates the percentages of times that two symbols were declared to be different. It is derived from the original by first taking the symmetric part (i.e., half the sum of the original matrix plus its transpose) and then subtracting these from 100. Finally, the diagonal elements were set to 0 to ensure that it was a true distance matrix.

The file `morse.Rdata` gives just the submatrix of `morsefull.Rdata` that contains the digits 0–9:

$$1 : \cdot - - - -$$
$$2 : \cdot \cdot - - -$$
$$3 : \cdot \cdot \cdot - -$$
$$4 : \cdot \cdot \cdot \cdot -$$
$$5 : \cdot \cdot \cdot \cdot \cdot$$
$$6 : - \cdot \cdot \cdot \cdot$$
$$7 : - - \cdot \cdot \cdot$$
$$8 : - - - \cdot \cdot$$
$$9 : - - - - \cdot$$
$$0 : - - - - -$$

```
> load("morse.Rdata")
> attach(morse)
> morse.cmd<-cmdscale(morse,k=9,eig=TRUE)
Warning message:
In cmdscale(morse, k = 9, eig = TRUE) :
  only 7 of the first 9 eigenvalues are > 0
> morse.cmd$eig
 [1]  1.143502e+04  6.354744e+03  4.482303e+03  2.330540e+03  1.758846e+03
 [6]  9.104095e+02  1.524900e+01 -2.501110e-12 -7.103261e+02 -1.128180e+03
```

Using the option `k=9` means that the eigenvectors corresponding to the first 9 non-negative eigenvalues are recorded and stored in the matrix `morse.cmd$points`. Omitting the option `k=9` would result in the first two eigenvectors only being calculation and they would be stored in a matrix `morse.cmd[,.,]`. Using the option `eig=TRUE` means that the eigenvalues are stored in the vector `morse.cmd$eig`.

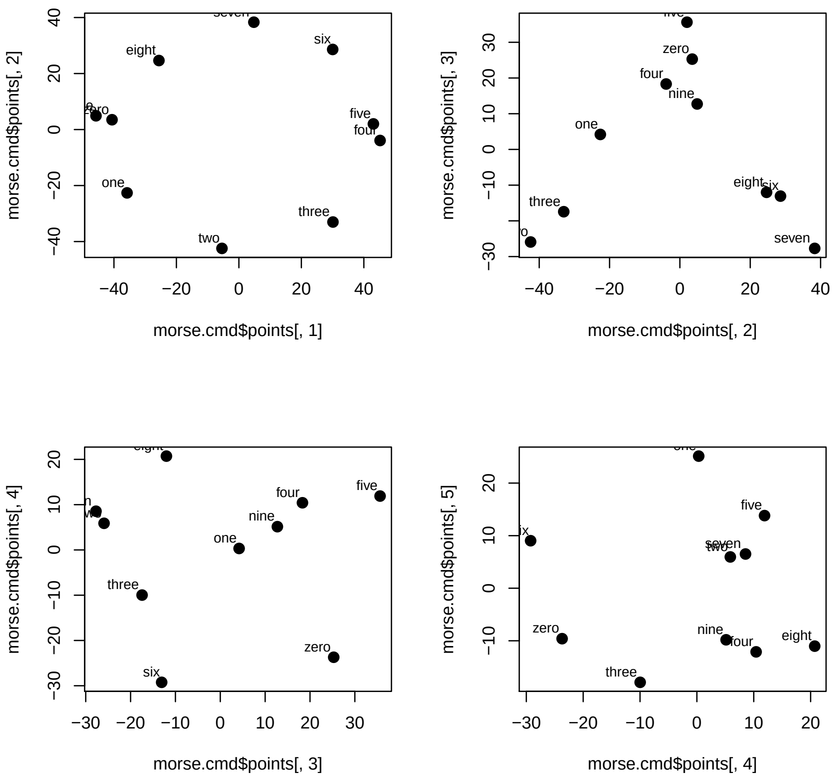
Note the warning message signalling that some eigenvalues are negative; this means it is not possible to find an exact Euclidean solution.

```
> plot(morse.cmd$points[,1],morse.cmd$points[,2],pch=16,cex=1.5)
> text(morse.cmd$points[,1],morse.cmd$points[,2],row.names(morse),
    cex=0.8,adj=c(1.1,-0.6))
>
> plot(morse.cmd$points[,2],morse.cmd$points[,3],pch=16,cex=1.5)
> text(morse.cmd$points[,2],morse.cmd$points[,3],row.names(morse),
    cex=0.8,adj=c(1.1,-0.6))
>
> plot(morse.cmd$points[,3],morse.cmd$points[,4],pch=16,cex=1.5)
> text(morse.cmd$points[,3],morse.cmd$points[,4],row.names(morse),
>
> plot(morse.cmd$points[,4],morse.cmd$points[,5],pch=16,cex=1.5)
> text(morse.cmd$points[,4],morse.cmd$points[,5],row.names(morse),
     cex=0.8,adj=c(1.1,-0.6))
```

After producing the first plot, it is useful to make the plot the active window and then click on `History` in the menu bar and select `Recording`. This allows scrolling between all subsequent graphs and the first one with the page up and down keys. Alternatively, issuing the command

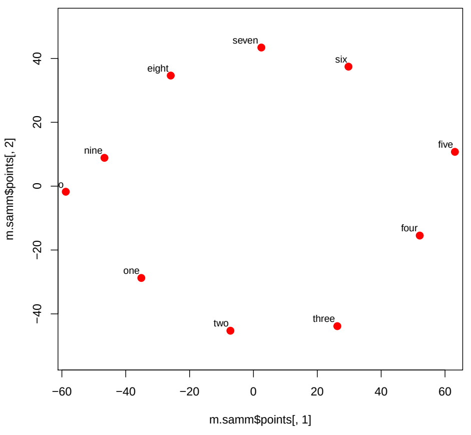
```
> par(mfrow=c(2,2))
```

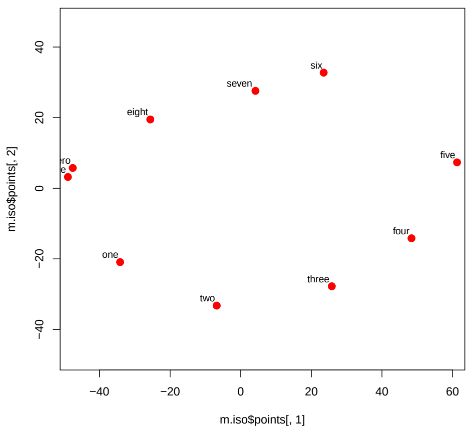
before the plot commands produces all the plots in a convenient matrix.



To compare with non-metric methods, we have:

```
> m<-as.matrix(morse)
> m.samm<-sammon(m)
Initial stress        : 0.06906
stress after  10 iters: 0.02855, magic = 0.500
stress after  20 iters: 0.02829, magic = 0.500
stress after  30 iters: 0.02829, magic = 0.500
>
> eqscplot(m.samm$points[,1],m.samm$points[,2],pch=16,col="red",cex=1.5)
> text(m.samm$points,names(morse),cex=0.8,adj=c(1.1,-0.6))
>
> m.iso<-isoMDS(m,cmdscale(morse))
initial  value 13.103235
iter   5 value 9.362759
iter  10 value 7.943274
final  value 7.652321
converged
>
> eqscplot(m.iso$points[,1],m.iso$points[,2],pch=16,col="red",cex=1.5)
> text(m.iso$points,names(morse),cex=0.8,adj=c(1.1,-0.6))
```





Note that in the call to `isoMDS()` a starting configuration was required and this was given by the matrix `cmdscale(morse)`. The object `morse.cmd` could not be used since this was created with the `k=9` option (to obtain all 9 dimensional and so `morse.cmd` is not a matrix). By default, `cmdscale()` with no optional arguments produces a matrix with the 2-dimensional configuration.

4.4.4 Language concordancies

One way we might define the dissimilarity between two languages could be to count the words for the numbers 1–10 with different first letters.

So the dissimilarity between English and French would be 6: one/un, two/deux, four/quatre, five/cinq, eight/huit, ten/dix (but not three/trois, six/six, seven/sept or nine/neuf).

An analysis was done on the ten languages: Arabic, Danish, Dutch, English, Finnish, French, German, Hungarian, Polish and Russian (where appropriate transliterations from other scripts were made).

The matrix of dissimilarities is as follows:

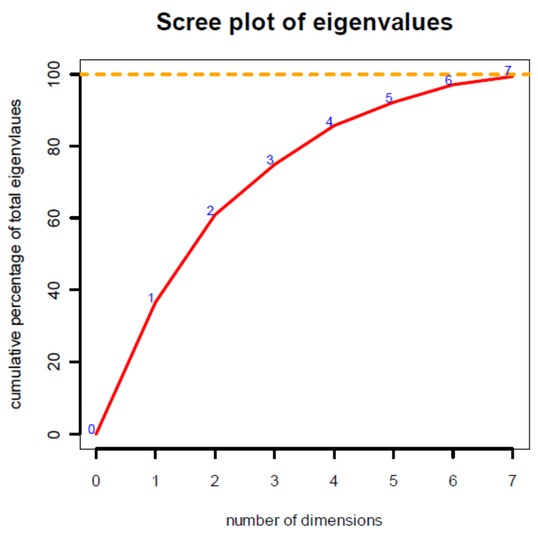
	Eng	Dan	Dut	Ger	Fre	Pol	Hun	Fin	Rus	Ara
Eng	0	2	7	6	6	7	9	9	7	6
Dan	2	0	6	5	6	6	8	9	7	7
Dut	7	6	0	5	9	10	8	9	10	10
Ger	6	5	5	0	7	8	9	9	8	8
Fre	6	6	9	7	0	5	10	9	5	7
Pol	7	6	10	8	5	0	10	9	2	7
Hun	9	8	8	9	10	10	0	8	10	10
Fin	9	9	9	9	9	9	8	0	9	10
Rus	7	7	10	8	5	2	10	9	0	8
Ara	6	7	10	8	7	7	10	10	8	0

The eigenvalues are

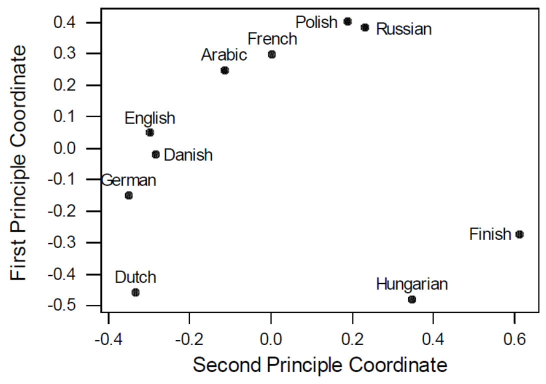
$$101.9, 66.9, 39.4, 29.9, 18.0, 13.2, 7.0, 4.7, 0.0, -3.7$$

- There is no set of points with these Euclidean distances, as there is a negative eigenvalue
- A scree plot suggests that two components don't really account for a reasonable proportion of eigenvalues, so there is probably distortion in the 2-dimensional plot.

Here's a scree plot of the stresses against dimension:



Let's plot the first two principal co-ordinates:



We'll return to this example later: one place you can see that there is distortion is in the position of Arabic – according to the data matrix, it should be closer to English than to French, which is clearly not the case in this plot. Later, we'll think about one way to automate this observation.

4.5 Notes

- In the examples on the French départements, Munsingen graves and Morse Code, the measures were really *similarities*, whereas those on distances between towns and the iris data, the measures were *dissimilarities*. A "similarity" is easily converted to a "dissimilarity" by changing the sign, or taking the reciprocal or subtracting from 100 or many similar devices. In fact, one of the exercises shows that provided you convert between similarities and dissimilarities in a particular way, then applying the classical solution above gives precisely the same answers whether you start with a matrix of dissimilarities (as assumed there) or a matrix of similarities. Generally, it does not matter very much how dissimilarities are converted into similarities and vice versa and the resulting displays are very similar.

- If two points are well separated in a scaling plot, then it must be the case that they are dissimilar, but the converse is not true. If two plots are close together then they may or may not be similar; plots on further dimensions might separate them. One way to assess this is to look at the percentage of stress captured in the plot. If the greater part of the stress is represented in the plot, then there cannot be enough room to separate them. Another convenient way to assess this graphically is to superimpose a *minimum spanning tree* on the scaling plot. If this is annotated with the actual distances of each branch (only realistic for small numbers of plots) then this gives a full picture for interpretation. Exactly the same is true for PCA plots.

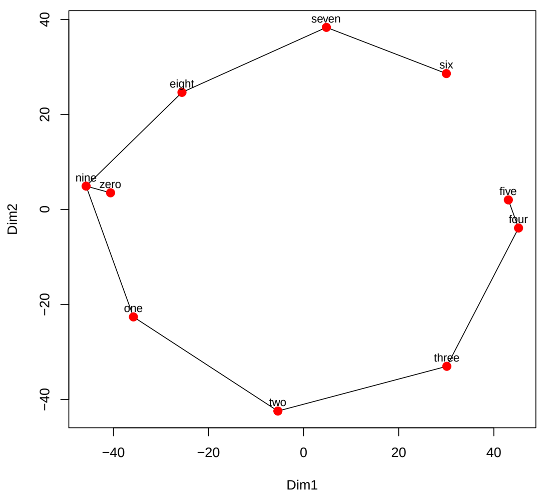
4.6 Minimum spanning trees

The minimum spanning tree is calculated using the distance information (across all dimensions) using the original dissimilarity matrix. It provides a unique path such that (i) all points are connected, and (ii) there are no circuits, and has the shortest possible path of all such trees. It has the property that for nodes r and s, d_{rs} must be greater than the maximum link in the unique path from r to s. The idea is to give a direct visual impression of each individual's "nearest neighbours" since in general, these will be the individuals connected to it by the edges of the tree. So if two points appear to be close together in a scaling plot, but the tree does not join them directly, then it can be concluded that in fact they may not be very similar, though note that the converse does not hold. Nearby points on the plot which are not joined by edges indicate possible areas of distortion. There are two main algorithms for constructing minimal spanning trees: Kruskal's algorithm and Prim's algorithm. Further details can be found on Wikipedia, for example; both are very simple algorithms to explain, but naive methods for implementation can be slower than the best methods.

R has several facilities for calculating and plotting minimum spanning tress within the contributed packages. A selection of these is `mst()` in package `ape`, `dino.mst()` and `nmds()` in package `fossil`, `mstree()` in package `spdep`, `spantree()` in package `vegan`. Only the last of these is illustrated here; we return to the example of Morse code confusion.

```
> library(vegan)
> morse.tr<-spantree(morse)
> plot(morse.tr,cmdscale(morse),pch=16,col=2,cex=1.5)
> text(cmdscale(morse),names(morse),cex=0.8,adj=c(0.5,-0.6))
```

Here is the resulting picture:

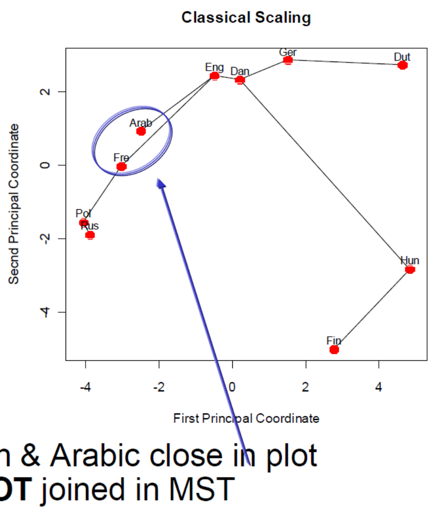


Examining the minimum spanning tree shows that the points “nine” and “zero” are joined, and also “four” and “five”. However, all we can tell from this is that these pairs are closer together than to any other point. Of course, with this small number of points, this could be seen by examining the distance matrix. In this case, it shows that “zero” and “nine” are indeed close, but “four” and “five” are well separated (which of course can be seem from the earlier plot of dimensions 2 and 3). To examine the distances on the minimum spanning tree (and so fewer numbers to scan for a large dataset) do:

```
> morse.tr$kid
[1]  1 2 3 4 7 8 9 1 9
> morse.tr$dist
[1] 38 41 62 44 35 35 42 43 21
```

`morse.tr$kid` gives the child note of the parent, starting from parent number two, and `morse.tr$dist` gives the corresponding distances.

Let’s return to the example of language concordancies, where we saw some distortion. We’ll take the plot of the first two principal co-ordinates, and add the minimum spanning tree:



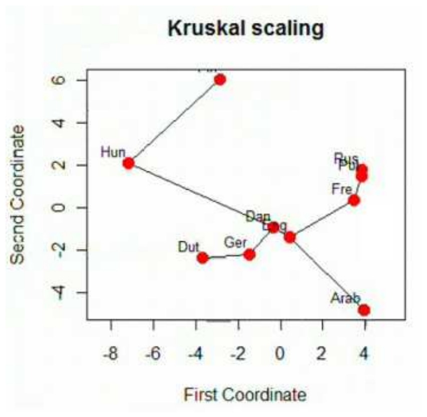
- French & Arabic close in plot but **NOT** joined in MST

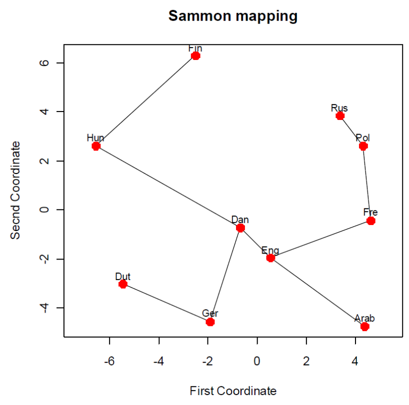
French and Arabic look close in the plot, but they are not adjacent in the minimum spanning tree; the path between them passes through English.

We can see that Arabic is nearer English than French (in this measure). This shows that there is distortion in the 2-dimensional view.

But this is just a projection from higher dimensions, and so we see that French and Arabic are more widely separated in the third and higher dimensions.

Let’s look at the results of applying Kruskal and Sammon non-metric scaling. We see that both techniques separate French and Arabic more successfully.





4.7 Summary and conclusions

- Multidimensional scaling is concerned with producing a representation of points in low dimensional space starting from a matrix of interpoint distances.
- The distances can be a general measure of similarity or (equivalently) dissimilarity.
- The purpose of multidimensional scaling analyses may be to learn about the measure of (dis)similarity itself or to identify structure in the points themselves. One might, for example, discover that objects are clustered in separate groups.
- Applying the technique to an example where the "answer is known" (e.g., French départements) gives confidence when applying it to similar but unknown situations.
- The classical solution, also known as Principal Co-ordinate Analysis, gives a method for constructing a display.
  - If the distance matrix is Euclidean, it gives an exact representation (up to arbitrary rotations and reflections), and if the negative eigenvalues are small, one might not expect too much distortion, so that it should be a reasonably accurate measure.
  - We can reduce dimensionality by ignoring small eigenvalues.
  - Otherwise it can be a starting point for iterative techniques of monotonic regression.
- Scree plots of eigenvalues or stress values can give an informal aid to determining a suitable number of dimensions.
- Some eigenvalues determined in Principal Co-ordinate Analysis from non-Euclidean distance matrices will be negative. Care needs to be taken in construction of scree plots in such cases.
- Principal Co-ordinate Analysis is the dual of Principal Component Analysis.
- The axes produced by metric or non-metric scaling analysis are arbitrary. It may be possible to assign intuitive interpretations to them by examining the data, but these are informal and not a property of the analysis (unlike the axes in PCA or Crimcoords below).
- In R and S-PLUS, the key commands are `cmdscale()`, `sammon()` and `kruskal()`.

4.8 Further reading

- Other “unsupervised learning” techniques for investigating similarities include Cluster Analysis and Kohonen self-organising maps (see internet sources for more details).
- To see how closely shapes correspond, one can use *Procrustes analysis.* The name “Procrustes” refers to an innkeeper from Greek mythology who made his victims fit his bed either by stretching their limbs or cutting them off. We won’t say anything about this, but if you are doing an MSc dissertation and need to try to analyse whether your maps are suitable, you might want to read up on it.

Further reading: Visualisation of contingency tables

Introduction

We have now seen ways to visualise two forms of data set in a small number of dimensions: a typical $n \times p$ data matrix can be visualised with principal components analysis (or other techniques), and an $n \times n$ proximity matrix can be visualised with multidimensional scaling.

But data sets arise in other ways too. One way is a contingency table. We will first explain how to transform contingency tables to distance matrices, amenable to visualisation via scaling methods. The problem with this is that one can visualise the rows collectively, or the columns collectively, but there is no obvious relation between them, and no obvious way to relate the row and column categories. But there is another way, called *correspondence analysis*, which does allow this to happen, and we sketch it after discussing the transformation into distances.

Visualisation with MDS

Before we introduce correspondence analysis, let's give the first way to visualise contingency tables, using the methods of multidimensional scaling. So we will need to convert our contingency tables into distances.

We'll borrow a running example from Chapter 7 of Cox's book, mentioned on p.5 (and correct all his calculations!).

The Wall Street Journal (September 1st 1987) gave an analysis of 298 women in corporate sales, classified by position, and type of dress. Position is measured by the labels:

P1	Top
P2	Above average
P3	Average or below
P4	Failing

Dress is classified by:

Prof	Professionally dressed
Appr	Appropriately dressed
Fash	Fashionably dressed
Poor	Poorly dressed

Then the data on corporate sales is given by:

	Prof	Appr	Fash	Poor	Total
P1	12	12	7	1	32
P2	56	39	18	16	129
P3	20	42	32	26	120
P4	4	2	2	9	17
Total	92	95	59	52	298

The Euclidean distance between rows is heavily influenced by the total number in each row, so we need to perform some normalisation. A similar consideration applies to the columns. We will standardise the data in three ways. Firstly, we standardise all the data by dividing all the entries by the total number of entries:

	Prof	Appr	Fash	Poor	Total
P1	0.040	0.040	0.023	0.003	0.107
P2	0.188	0.131	0.060	0.054	0.433
P3	0.067	0.141	0.107	0.087	0.403
P4	0.013	0.007	0.007	0.030	0.057
Total	0.308	0.319	0.197	0.174	1.000

We'll write X' for this standardised $n \times p$ data matrix (Cox calls this X), so

$$X' = \begin{pmatrix} 0.040 & 0.040 & 0.023 & 0.003 \\ 0.188 & 0.131 & 0.060 & 0.054 \\ 0.067 & 0.141 & 0.107 & 0.087 \\ 0.013 & 0.007 & 0.007 & 0.030 \end{pmatrix}.$$

We also let

$$D_R = \begin{pmatrix} 0.107 & 0 & 0 & 0 \\ 0 & 0.433 & 0 & 0 \\ 0 & 0 & 0.403 & 0 \\ 0 & 0 & 0 & 0.057 \end{pmatrix} = \mathrm{diag}(R_1, R_2, R_3, R_4)$$

and

$$D_C = \begin{pmatrix} 0.308 & 0 & 0 & 0 \\ 0 & 0.319 & 0 & 0 \\ 0 & 0 & 0.197 & 0 \\ 0 & 0 & 0 & 0.174 \end{pmatrix} = \mathrm{diag}(C_1, C_2, C_3, C_4)$$

be the diagonal matrices formed of the (standardised) row and column totals, where R_i is the proportion of the women in position P_i, and C_i is similarly the proportion dressed in the given category.

Standardising the data in this way doesn't address the issue of different row or column totals, so we'll need to standardise the rows and columns individually by dividing each row individually by the row and column totals. Then the row profiles are given by:

	Prof	Appr	Fash	Poor	Total
P1	0.375	0.375	0.219	0.031	1.000
P2	0.434	0.302	0.140	0.124	1.000
P3	0.167	0.350	0.267	0.217	1.000
P4	0.235	0.118	0.118	0.529	1.000

and notice that the matrix of row profiles is just $D_R^{-1}X'$:

$$D_R^{-1}X' = \begin{pmatrix} 0.375 & 0.375 & 0.219 & 0.031 \\ 0.434 & 0.302 & 0.140 & 0.124 \\ 0.167 & 0.350 & 0.267 & 0.217 \\ 0.235 & 0.118 & 0.118 & 0.529 \end{pmatrix},$$

and similarly standardising the column profiles gives (note that there is an error in Cox here):

$$X'D_C^{-1} = \begin{pmatrix} 0.130 & 0.126 & 0.119 & 0.019 \\ 0.609 & 0.410 & 0.305 & 0.308 \\ 0.217 & 0.442 & 0.542 & 0.500 \\ 0.043 & 0.021 & 0.034 & 0.173 \end{pmatrix}.$$

(In correspondence analysis below, we will work with the transpose $D_C^{-1}X$.) Now the χ^2-distance between rows r and s is defined as

$$d_{rs}^2 = \sum_{i=1}^{p} \frac{1}{C_i} \left(\frac{x_{ri}}{R_r} - \frac{x_{si}}{R_s} \right)^2,$$

a weighted Euclidean distance between the rows of the row profile matrix, and similarly, the χ^2-distance between two columns is

$$d_{ij}^2 = \sum_{r=1}^{n} \frac{1}{R_r} \left(\frac{x_{ri}}{C_i} - \frac{x_{rj}}{C_j} \right)^2.$$

In our example, the matrix of χ^2-distances between rows is

$$\begin{pmatrix} 0.000 & 0.330 & 0.593 & 1.320 \\ 0.330 & 0.000 & 0.608 & 1.086 \\ 0.593 & 0.608 & 0.000 & 0.926 \\ 1.320 & 1.086 & 0.926 & 0.000 \end{pmatrix},$$

and the matrix of χ^2-distances between columns is

$$\begin{pmatrix} 0.000 & 0.474 & 0.692 & 0.904 \\ 0.474 & 0.000 & 0.233 & 0.738 \\ 0.692 & 0.233 & 0.000 & 0.660 \\ 0.904 & 0.738 & 0.660 & 0.000 \end{pmatrix}.$$

Now that one has matrices of distances, methods from MDS can be used to visualise them. (Alternatively, they can be analysed using cluster analysis, sketched briefly below.)

Correspondence analysis

This is a technique for investigating/displaying the relationship between two categorical variables (i.e., frequency data in a contingency table) in a type of scatterplot. Broadly analgous to PCA, except that instead of partitioning the total variance into successive components, it partitions the χ^2-statistic (known as the *total inertia*, $\sum(O-E)^2/E$) into components attributable to orthogonal underlying components. Interpretations of "proportions of inertia explained" loadings of categories etc., are just as in PCA. Mathematically it again relies on the eigenanalysis of an appropriate matrix.

In our example above, the method starts by finding the *generalised singular value decomposition* of the data matrix X' with respect to the row and column weightings. More precisely, we need to solve $X' = ADB'$, where D is a diagonal matrix of the singular values of X' (a generalisation of the notion of eigenvalue to nonsquare matrices), subject to the requirements that

$$A'D_R^{-1}A = I$$
$$B'D_C^{-1}B = I.$$

Just like PCA, we choose a dimension k for the visualisation (usually $k = 2$ or possibly $k = 3$), and will later just plot the dimensions corresponding to the largest k singular values.

The matrices A and B are computed using row and column operations. But it is easier to do it in R; this sort of generalised SVD can be done with the command `SVDgen` in the `PTAk` package. Write

$$U = D_R^{-1}A, \qquad V = D_C^{-1}B;$$

then the row profiles are $D_R^{-1}X = (UD)B'$, and the column profiles are $D_C^{-1}X = (VD)A'$. Our normalisations correspond to equalities

$$U'D_RU = I, \qquad V'D_CV = I.$$

This means that the row profiles of X' are rotated by B into UD-space (i.e., the space spanned by the columns of UD). Similarly, the column profiles are rotated by A into VD-space. Since these are rotations, Euclidean distances between row profiles are unchanged (and similarly for column profiles).

Let's look at our example again. First we find the generalised singular value decomposition for X':

$$X' = \begin{pmatrix} 0.107 & 0.111 & -0.099 & -0.272 \\ 0.433 & 0.364 & 0.244 & 0.231 \\ 0.403 & -0.350 & -0.317 & 0.132 \\ 0.057 & -0.125 & 0.172 & -0.092 \end{pmatrix} \begin{pmatrix} 1.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.311 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.218 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.009 \end{pmatrix}$$
$$\times \begin{pmatrix} 0.309 & 0.360 & 0.259 & -0.129 \\ 0.319 & 0.047 & -0.249 & 0.391 \\ 0.198 & -0.106 & -0.239 & -0.301 \\ 0.174 & -0.300 & 0.229 & 0.039 \end{pmatrix}.$$

So

$$A = \begin{pmatrix} 0.107 & 0.111 & -0.099 & -0.272 \\ 0.433 & 0.364 & 0.244 & 0.231 \\ 0.403 & -0.350 & -0.317 & 0.132 \\ 0.057 & -0.125 & 0.172 & -0.092 \end{pmatrix}, \qquad B' = \begin{pmatrix} 0.309 & 0.360 & 0.259 & -0.129 \\ 0.319 & 0.047 & -0.249 & 0.391 \\ 0.198 & -0.106 & -0.239 & -0.301 \\ 0.174 & -0.300 & 0.229 & 0.039 \end{pmatrix}.$$

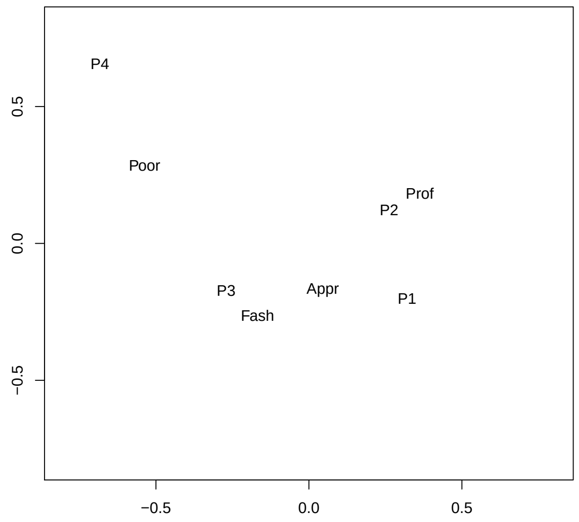
Then

$$U = D_R^{-1}A = \begin{pmatrix} 1.000 & 1.032 & -0.924 & -2.529 \\ 1.000 & 0.842 & 0.563 & 0.533 \\ 1.000 & -0.869 & -0.787 & 0.329 \\ 1.000 & -2.195 & 3.023 & -1.605 \end{pmatrix},$$
$$V = D_C^{-1}B = \begin{pmatrix} 1.000 & 1.165 & 0.840 & -0.419 \\ 1.000 & 0.146 & -0.781 & 1.227 \\ 1.000 & -0.535 & -1.206 & -1.520 \\ 1.000 & -1.722 & 1.310 & 0.224 \end{pmatrix}.$$

We ignore the eigenvalue 1 (which is always present). We can rotate the row profiles onto UD-space as above, plotting (say) the first two dimensions, and also rotate the column profiles onto VD-space similarly. But

$$\begin{aligned} UD &= (UD)(B'{}_C^{-1}B) \\ &= (D_R^{-1}X')V \\ &= (D_R^{-1}X')(VD)D^{-1}, \end{aligned}$$

and similarly $VD = (D_C^{-1}X)(UD)D^{-1}$, so it is straightforward to move between UD-space and VD-space; this allows us to plot both sets of variables on the same graph, and to relate the two sets of variables graphically.



We can see how the two divisions into categories are interacting – failing saleswomen (P4) tend to be poorly dressed, for example, since `P4` and `Poor` are fairly close on the graph above.

We want to measure how much of the information about the distances is retained by working in a correspondence analysis in k dimensions.

For this, we introduce a measure called *inertia*, relating to the dispersion of the points. The inertia has a rather complicated definition, but simplifies to $I = \mathrm{tr}(D^2)$ (where we ignore the singular value 1), and is therefore equal to the sum of the square singular values. Visualising in k dimensions retains a proportion of the total inertia corresponding to the first k singular values – a proportion of

$$\sum_{i=1}^{k} \lambda_i^2 / \sum_{i=1}^{n} \lambda_i^2.$$

R file for the graph above

```
library(MASS)
library(PTAk)
X<-matrix(c(12,12,7,1,56,39,18,16,20,42,32,26,4,2,2,9),nrow=4,ncol=4,byrow=TRUE)
x<-X/sum(X)
one<-c(1,1,1,1)
rowtot<-X%*%one
coltot<-t(one)%*%X
DR<-diag(c(rowtot))
DC<-diag(c(coltot))
DR<-DR/sum(X)
DC<-DC/sum(X)
s<-SVDgen(x,solve(DC),solve(DR))
A<-t(s[[1]]$v)
```

```
D<-diag(s[[2]]$d)
B<-t(x)%*%t(solve(A))%*%solve(D)
U<-solve(DR)%*%A
V<-solve(DC)%*%B
UD<-U%*%D
VD<-V%*%D
rown<-c("P1","P2","P3","P4")
coln<-c("Prof","Appr","Fash","Poor")
a1<-UD[,2]
b1<-UD[,3]
a2<-VD[,2]
b2<-VD[,3]
a<-c(a1,a2)
b<-c(b1,b2)
n<-c(rown,coln)
plot(a,b,type="n",xlab="",ylab="")
text(a,b,labels=n)
```

More on correspondence analysis

Canonical correspondence analysis is correspondence analysis incorporating continuous covariates, i.e., it analyses the relationship between two categorical variables after allowance has been made for the dependence of one of them on a covariate (for example, data on frequences of occurrence of species on various sites arranged in a sites$\times$species contingency table, suitable for correspondence analysis, to see which species tend to group together).

Multiple Correspondence Analysis is a generalisation of correspondence analysis to three- or higher-dimensional contingency tables.

We will not say anything more here, through lack of time and space, but wanted to make the reader (perhaps with an MSc dissertation in mind) aware of this technique. The reader is referred to Wikipedia, or to Cox, mentioned on p.5.

Further reading: Cluster analysis

One aim of multidimensional scaling, and data visualisation more generally, is to find structure in data. In particular, we might hope to use visualisation techniques to spot group structure in data. The problem of separating observations into groups is known as "unsupervised learning" in the machine learning literature. (In the next section, we will consider "supervised learning", where we know group structure in advance, but want to classify new observations into one of the groups.)

So in "unsupervised learning", the machine is given a load of data, and attempts to separate the observations into groups. The statistical/data analytical terminology for this sort of problem is *cluster analysis*.

Given some data, you may (or may not) wish to delete missing entries, and to rescale the variables so that their variances are the same; otherwise some variables can distort the clustering. One can do this with

```
data<-na.omit(data)
data<-scale(data)
```

Hierarchical clustering, or connectivity-based clustering

Hierarchical cluster analysis typically imagines the n observations in p dimensions, and starts by trying to merge two points into a single cluster – the natural first choice here are the two points which are closest (using an appropriate definition of closest).

Having started with n distinct observations, we now have $n-1$ clusters; $n-2$ consist of single observations, but there is also 1 cluster consisting of a pair of observations.

We repeat this – we need to find which two of the clusters are closest.

It is easy to measure the distance between two observations, but how do we measure the distance between an observation, and the cluster consisting of two observations?

More generally, as more and more clusters are formed by the merging process, we need to measure the distance between two clusters of observations.

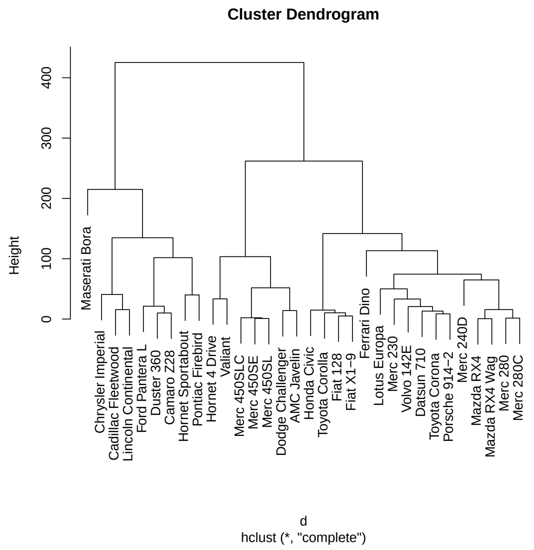
There are many methods for doing this:

- *Single linkage* – the distance between two clusters is defined as the shortest distance between a point in one cluster and a point in the second cluster.
- *Complete linkage* – the distance between two clusters is defined as the furthest distance between a point in one cluster and a point in the second cluster.
- *Average linkage* – the distance between two clusters is defined as the distance between their centroids (i.e., means).
- We can try to minimise the sums of squares between centroids of clusters.
- etc.

Conversely, one can start with a single cluster, and split clusters into two where appropriate using these linkages. This is known as *divisive* hierarchical clustering (and the first method given above is called *agglomerative*, if one wants to refer to this).

In R, one uses the command `hclust()` to perform agglomerative hierarchical clustering; by default, it uses the complete linkage method (but type `?hclust` in R to find alternatives). Results are generally displayed on a *dendrogram*:

```
d<-dist(as.matrix(mtcars))
hc<-hclust(d)
plot(hc)
```



Going down the height axis, and drawing a horizontal line at any desired point, shows the clusters at that level. In the example, a height of 200 has 4 clusters, one of which is a single outlier while the other three are of similar sizes. Further analysis of the dendrogram can be done via the command `as.dendrogram()`; in the example above, we can type `dc<-as.dendrogram(hc)`, and then typing `str(dc)` gives a string-based output of the dendrogram, which can be adjusted in various ways, e.g., by just restricting to a part of the tree, merging two parts, looking more closely at the structure, and so on (type `?dendrogram` for more).

Non-hierarchical methods

Nowadays, hierarchical methods are regarded as less useful than the alternative *non-hierarchical*) approaches to clustering by the data mining community.

Centroid-based clustering specifies a number k of clusters in advance. Then the idea is to choose k cluster centres, and assign observations to the cluster with the nearest centre. This gives a clustering of the n objects into k clusters, which depends on the initial choice of cluster centres.

The problem comes in how to determine the best choices of cluster centres. We need to decide some way to score a choice of cluster centres – the most common algorithm here is *k-means clustering*, where the score is the sum of the squared distances between the data points and the nearest cluster centre – and try to choose the cluster centres so that this score is minimised. There are other measures of scoring choices of cluster centres, but k-means clustering is by far the most commonly used method in practice.

This is a hard optimisation problem, but there are iterative methods that converge reasonably fast to local optimum configurations:

1. Choose k cluster centres randomly.

2. Assign each data point to the cluster centre to which it is closest, and compute the total squared distances.

3. Replace the cluster centres by the centroids (averages) of the clusters to which they belong.

4. Repeat the last two steps (note that sometimes data points will now be assigned to different clusters).

Evenutally (quite quickly, usually), this converges, and we get a clustering of the n objects into k clusters, but this may depend on the initial choice of cluster centres; you may wish to run this several times, and choose the best.

We can run this in R with the command `kmeans`; typing `kmeans(mtcars,4)` runs the k-means clustering algorithm with 4 clusters. The output computes the best cluster centres, and then assigns a group to each data point. To measure how successful this is, R computes the percentage of the contribution of the total sum of squares coming from points in different clusters; we hope that distances within clusters will be small, so we hope that this percentage is large. In the case of the `mtcars` dataset, 4 clusters account for 88.1% of the total sum of squares (note that since different choices of initial centres may converge to different local optimum values, if you repeat this, you may well get a different answer).

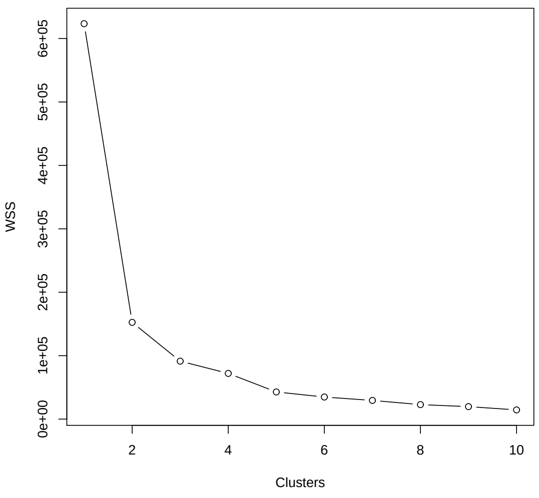
Then there remains the question of which value of k we should use. We can repeat this method for each $k = 2, 3, \ldots$, and form a scree plot of the optimum total sum of squares against k, or alternatively the percentage of sum of squares coming from between clusters, and select an appropriate value of k by looking for an "elbow" in the plot.

All the methods try to minimise variance within the clusters and maximise variance between clusters. So the method resembles a kind of reverse ANOVA, in that we are trying to move observations around so as to get the most significant ANOVA results.

One way is to use the within clusters sum of squares, and make a scree plot:

```
> wss<-(nrow(mtcars)-1)*sum(apply(mtcars,2,var))
> for(i in 2:10) wss[i]<-sum(kmeans(mtcars,centers=i)$withinss)
> plot(1:10,wss,type="b",xlab="Clusters",ylab="WSS")
```

produces the scree plot:

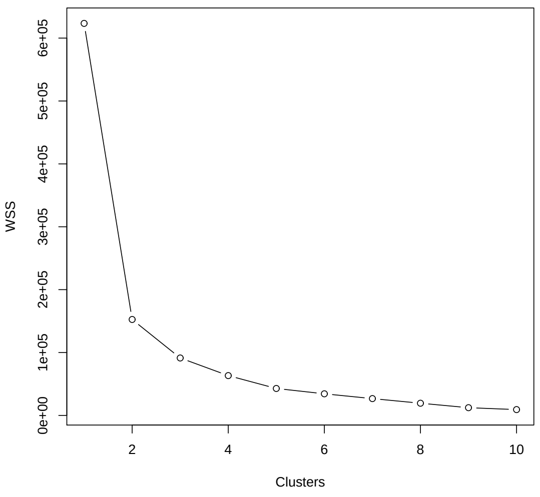


The first line assigns the value of `wss[1]`, the value when there is only 1 cluster; the second line gives the within sum of squares result for between 2 and 10 clusters, then plotting this on a scree plot. With this scree plot, one might be tempted to go for either 3 or 5 clusters, since there seem to be suitable kinks in the plot here.

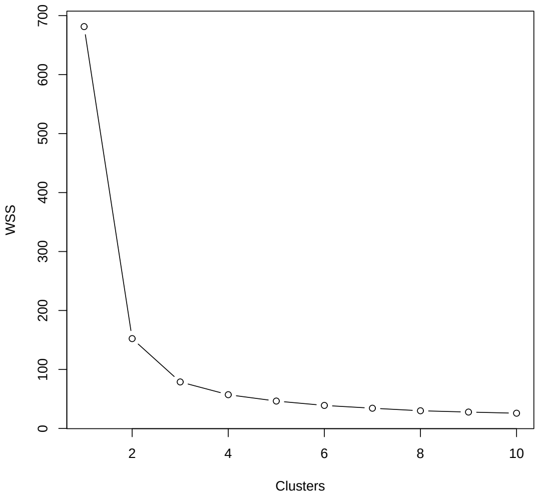
But remember that, as already mentioned, running this several times gives different plots each time, as the initial cluster centres are chosen randomly, and the final result is strongly dependent on this initial choice. Better is to run the command several times and take the best solution:

```
out<-0
wss<-(nrow(mtcars)-1)*sum(apply(mtcars,2,var))
for(i in 2:10){
  for(j in 1:100){
    out[j]<-sum(kmeans(mtcars,i)$withinss)
  }
  wss[i]<-min(out)
}
plot(1:10,wss,type="b",xlab="Clusters",ylab="WSS")
```

and this produces the scree graph



where the number of clusters is not so clear. However, with data which we know is clustered (e.g., the `iris` dataset), we get the scree plot:



and perhaps it is now easier to see that 2 or 3 clusters seem to be appropriate.

Further reading

Wikipedia lists alternative methods. For example, some methods suppose some statistical distribution of each cluster, and try to measure how good an assignment of points to clusters is by using the Mahalanobis distance (try the `Mclust()` command in the `mclust` package). You should look for "model-based clustering" in the literature.

The CRAN page `https://cran.r-project.org/web/views/Cluster.html` gives an excellent index to the R packages and commands available.

See also an excellent Stack Overflow answer.

5 Discriminant analysis

So far the data analytic techniques considered have regarded the data as arising from a homogeneous source – i.e., as all one data set. A display on principal components might reveal unforeseen features: outliers, subgroup structure as well as perhaps singularities (i.e., dimensionality reduction).

Suppose now we know that the data consist of observations classified into k groups (cf. 1-way classification in univariate data analysis) so that data from different groups might be different in some ways. We can take advantage of this knowledge with the objectives being:

- Effective data display using this extra information.
- Dimensionality reduction whilst retaining information on differences between groups.
- Informal interpretation by examination of loadings of the nature of differences between groups.
- Classification of future observations into one of the known groups.

Linear Discriminant Analysis (LDA) can aid all of these objectives, but if the primary aim is the final one of classification of future observations, then alternatives may do better on particular data sets, e.g., quadratic discriminant analysis, or neural networks (see internet sources).

As with PCA etc., it is based on linear combinations of the variables.

LDA is a purely data analytic method. If we know distributions of the data, we can begin to ask more statistical questions about, for example, the probabilities of misclassification of data, confidence intervals for predictions of classification of new observations etc. We'll do that in the last chapter of the course.

Linear Discriminant Analysis finds the best linear combinations of variables for separating the groups; if there are k different groups then it is possible to find $k-1$ separate linear discriminant functions which partition the *between groups variance* into decreasing order, similar to the way that principal component analysis partitions the *within group variance* into decreasing order. The data can be displayed on these discriminant coordinates (also known as *crimcoords.* Boundaries between classification regions are linear.

The effectiveness of the classification can be assessed by simulation methods (random-relabelling, jack-knifing) or classifying further observations of known categories.

This method requires more observations than variables (i.e., $n > p$), since it requires the invertibility of a $p \times p$ matrix whose rank is at most n.

5.1 Outline

The starting point is to define a measure of separation between the known groups: this is a matrix generalisation of the F-statistic used for testing in one-way analysis of variance, and is essentially the ratio of between group variance to within group variance.

Step one is to identify that linear combination of variables which maximises the F-statistic for testing differences between the groups in one-way analysis of variance on the univariate data (cf. maximising variances in derivation of PCA).

We shall see that this is achieved by the eigenanalysis of the ratio of between group variance to within group variance matrix (cf. the eigenanalysis of the variance matrix in PCA).

The first eigenvector is the first linear discriminant (also known as *Fisher's linear discriminant*). By analogy with PCA, subsequent eigenvectors are extracted as second, third,..., discriminant coordinates (or *crimcoords*), and data plotted on these axes, although strictly these axes are not orthogonal, i.e., the transformation to discriminant coordinates is not just a rotation/reflection.

Successive eigenvalues (and cumulative proportions etc.) are interpreted as "successive amounts of discrimination achieved" by each axis.

5.2 The algorithm and crimcoords

We suppose that there are k groups, labelled $G_1, \ldots, G_k$, and that group G_i has n_i observations (as usual, with p dimensions). Write $n = \sum n_i$ for the total number of observations as usual.

Write X_i' for the data matrix for the group G_i (so X_i' is a $n_i \times p$ matrix), and so the data are $\{x_j^{(i)} \mid i = 1, \ldots, k,\ j = 1, \ldots, n_i\}$ where $x_j^{(i)}$ is a $p \times 1$ vector. Write $\overline{x}$ for the p-vector which is the overall mean of the n observations, and $\overline{x}_i$ for the p-vector which is the mean of group G_i. In matrix terms, the $n \times p$ data matrix of the whole sample is X', where $X = (X_1; \cdots; X_k)$, and $\overline{x} = \frac{1}{n} X 1_n$ and $\overline{x}_i = \frac{1}{n_1} X_i 1_{n_i}$; here $\overline{X}_i = \overline{x}_i 1_{n_i}'$ is the $p \times n_i$ matrix with each column equal to $\overline{x}_i$.

Let

$$S_i = \frac{1}{n_i - 1}(X_i - \overline{X}_i)(X_i - \overline{X}_i)'$$

be the *within group i* variance matrix. Then

$$S_i = \frac{1}{n_i - 1}\sum_{j=1}^{n_i}(x_j^{(i)} - \overline{x}_i)(x_j^{(i)} - \overline{x}_i)'$$

(so S_i is $p \times p$).

Let

$$W = \frac{1}{n-k}\sum_{i=1}^{k}(n_i - 1)S_i$$

be the *within groups variance* (again a $p \times p$ matrix). This measures the variability inside the groups.

Define

$$B = \frac{1}{k-1}\sum_{i=1}^{k} n_i(\overline{x}_i - \overline{x})(\overline{x}_i - \overline{x})',$$

the *between groups variance*. This measures the variability between the groups.

W and B are analogous to the within and between groups mean squares in a univariate one-way analysis of variance, and if we set $p = 1$ then the usual formulae for these are retrieved. We want to compare W and B; since these are matrices, we can't take a quotient, but we will give a method below where the product $W^{-1}B$ comes out naturally, and when $p = 1$, this is indeed just the quotient. We should regard $W^{-1}B$ as a measure of comparison between W and B; we need some measure of how "large" $W^{-1}B$ is, and there are many possibilities – determinant, largest eigenvalue, sum of eigenvalues (trace), etc., and none really stands out as a natural choice.

Remark 5.1 It can be shown that if

$$T = (X - \overline{X})(X - \overline{X})' = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (x_j^{(i)} - \overline{x})(x_j^{(i)} - \overline{x})',$$

(the total sum of squares), then $T = (n - k)W + (k - 1)B$ (which is the multivariance analysis of variance of total variance into within and between components).

This will be exploited more formally later, when considering multivariate analysis of variance (MANOVA), but here we consider just a simple analogy with 1-way analysis of variance.

If we projected all the data into one dimension, then we could perform a 1-way analysis of variance and compare the between groups mean square with the within groups mean square: we would calculate the F-statistic, which is the ratio of between to within groups mean squares. If there are large differences between the groups, then the between group mean square will be relatively large and so the F-statistic will be large.

As we take different projections, this ratio will vary; sometimes it may be very small, when the differences between the groups are hidden by the projection, and at other times it may be very significant, when the projection reveals differences between the groups.

The objective in determining discriminant coordinates is to choose the projection to highlight the differences between the groups.

We project all the p-dimensional data onto vector a_1 (a column p-vector), so the projected data matrix for the ith group G_i becomes $X_i' a_1$.

Then the within group i mean square (i.e., the sample variance of the 1-dimensional projected data of the ith group) is $a_1' S_i a_1$ (note that this is a scalar!), and

$$\begin{aligned} a_1' S_i a_1 &= \frac{1}{n_i - 1} a_1' \sum_{j=1}^{n_i} (x_j^{(i)} - \overline{x}_i)(x_j^{(i)} - \overline{x}_i)' a_1 \\ &= \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (a_1' x_j^{(i)} - a_1' \overline{x}_i)^2. \end{aligned}$$

Similarly, the within group mean square of the projected data is $a_1' W a_1$ and the between group mean square is $a_1' B a_1$. To highlight the distinction between the groups, we want to choose a_1 to maximise $F_1 = \frac{a_1' B a_1}{a_1' W a_1}$ (i.e., just the usual F-ratio in a classical one-way analysis of variance, though note we have not as yet introduced the background of multivariate normality, so we will not as yet claim that this has a statistical distribution related to the variance-ratio Snedecor F-distribution used to assess significance in a one-way analysis of variance).

So the problem is to maximise $F_1 = \frac{a_1' B a_1}{a_1' W a_1}$ with respect to a_1. Note that F_1 is a ratio of quadratic forms in a_1, so if a_1 is multiplied by any scalar, then the value of F_1 is unaltered. This means that we can impose any scalar constraint on a_1 without altering the maximisation problem on F_1.

Noting that F_1 is a ratio of 2 quadratic forms, a convenient constraint to impose is that the denominator is 1. So now the problem becomes "maximise $a_1' B a_1$ subject to $a_1' W a_1 = 1$".

This allows us to convert the original problem to a constrained maximisation problem which will be solved using a Lagrange multiplier, thus making it an eigenvalue problem, which can be solved easily (numerically at least) by R.

Introduce a Lagrange multiplier λ_1, and let $\Omega_1 = a_1'Ba_1 - \lambda_1(a_1'Wa_1 - 1)$. Then $\frac{\partial\Omega_1}{\partial a_1} = 2Ba_1 - 2\lambda_1 Wa_1 = 0$, i.e., $W^{-1}Ba_1 - \lambda_1 a_1 = 0$, i.e., a_1 is an eigenvector of $W^{-1}B$ corresponding to the eigenvalue λ_1.

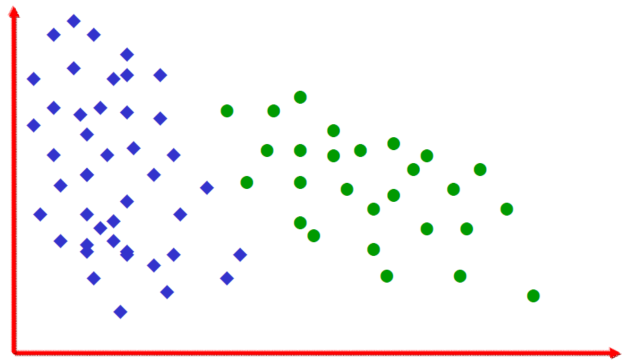
Notice that this is the point where we use the invertibility of the $p \times p$ matrix W, which requires that it has sufficiently large rank.

To see which eigenvector is needed to maximise F_1, note that $Ba_1 = \lambda_1 Wa_1$ implies that $a_1'Ba_1 = \lambda_1 a_1'Wa_1$, so $a_1'Ba_1 = \lambda_1$, and we take λ_1 as the largest eigenvalue of $W^{-1}B$ and a_1 as the corresponding eigenvector.

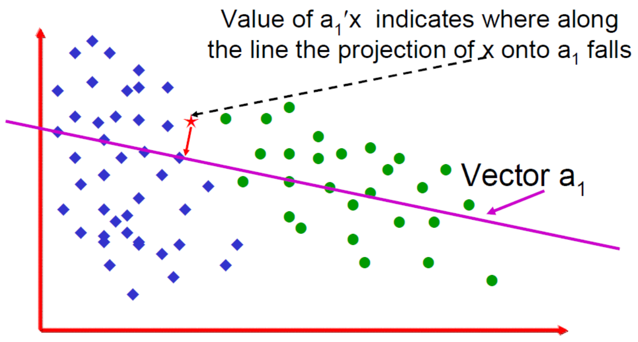
Remark 5.2 The function $f(x) = a_1'x$ is known as *Fisher's linear discriminant function.*

This choice of a_1 is the direction that makes the separation between the groups most clear.

Given a new observation x, we use a rule based on the value of $f(x)$. Here's a set with two groups:



We'll add the vector a_1 (translated so that it passes through the mean), and project onto the line it spans:



With two groups, this is the best function to make this prediction; with more than two, this is not necessarily the case, and other functions (or other methods) may be better.

In Principal Components Analysis, the other eigenvectors and eigenvalues were also useful. We could do the same here, and consider the other eigenvectors of $W^{-1}B$. Only λ_1 and a_1 have genuine interpretations (unlike PCA, when all the eigenvectors came out of the analysis).

In general, $W^{-1}B$ has several non-zero eigenvalues $\lambda_1, \ldots, \lambda_r$, where r is the rank of $W^{-1}B$, and is equal to $\min(k-1, p)$ unless there are pathological collinearities in either the data points in one or more groups or in group means.

Note that the eigenvectors of $W^{-1}B$ are not orthogonal – although both W and B are symmetric, $W^{-1}B$ need not be.

But if λ_i and λ_j are distinct eigenvalues of $W^{-1}B$ with eigenvectors a_i and a_j, then a_i and a_j are not necessarily orthogonal. However, they have a sort of orthogonality property: we have

$$\begin{aligned} Ba_i &= \lambda_i W a_i \\ Ba_j &= \lambda_j W a_j, \end{aligned}$$

and so $a_j' B a_i - \lambda_i a_j' W a_i = 0$ and $a_i' B a_j - \lambda_j a_i' W a_j = 0$. As W and B are symmetric, $a_j' B a_i = a_i' B a_j$ and $a_j' W a_i = a_i' W a_j$, and (assuming $\lambda_i \neq \lambda_j$), we see that $a_i' B a_j = a_i' W a_j = 0$ for $i \neq j$.

This gives an orthogonality "with respect to W" (or B). The within groups variance W reflects the different variances and covariances of the variables.

If all of the variables are independent, and have the same variance σ^2, then $W = \sigma^2 I_p$, and the condition really is an genuine orthogonality condition.

If W is diagonal, then $a_i' a_j = \sum_k \sigma_{kk}^2 a_{ik} a_{jk} = 0$. W "corrects" for the variance structure of the variables.

Definition 5.3 The functions $a_i' x$ are called the *discriminant coordinates*, or *crimcoords*, and the space they span is called the *discriminant space*. (We may also use this terminology for the space spanned by the first t of them.)

In practice, one chooses a suitable number t of co-ordinates (just like PCA), and views the data with these crimcoords.

Sometimes these functions are referred to as *canonical variates*, particularly in the context of interpreting the loadings of the variables so as to describe the nature of the difference between the groups. Examination of the loadings of variables in the crimcoords gives an idea of which variables provide discrimination between groups in an analogous way to the interpretation of principal components.

We cna choose a suitable t by examination of the sequence $\lambda_1, \lambda_2, \ldots$ using a scree plot (by analogy with PCA), but the non-orthogonality of the eigenvectors means that the amount of discrimination is not partitioned into separate bits, unlike principal components and classical scaling.

Although the crimcoords are not orthogonal, it is traditional to plot them on orthogonal axes; this distorts the original space of points, but in a sensible way. Transforming to discriminant space is a good way to display relationships and distinctions between groups.

5.3 R code, and an example (iris data)

The `MASS` library in R provides a function `lda()` which provides a full facility (as usual, type `help(lda)` to find out more); the syntax is `lda(group~V1+V2+...+Vk)`, where `Vi` denotes the variables, and `group` is the group (unsurprisingly!).

Then co-ordinates in the crimcoords are got with `predict.lda()$x` – this is the data matrix multiplied by the eigenvectors of $W^{-1}B$. Then new data can be plotted with `predict.lda(.,newdata)$x`.

Analysis in R

Returning yet again to Anderson's iris data, given below is a record of an R session to produce a display on crimcoords.

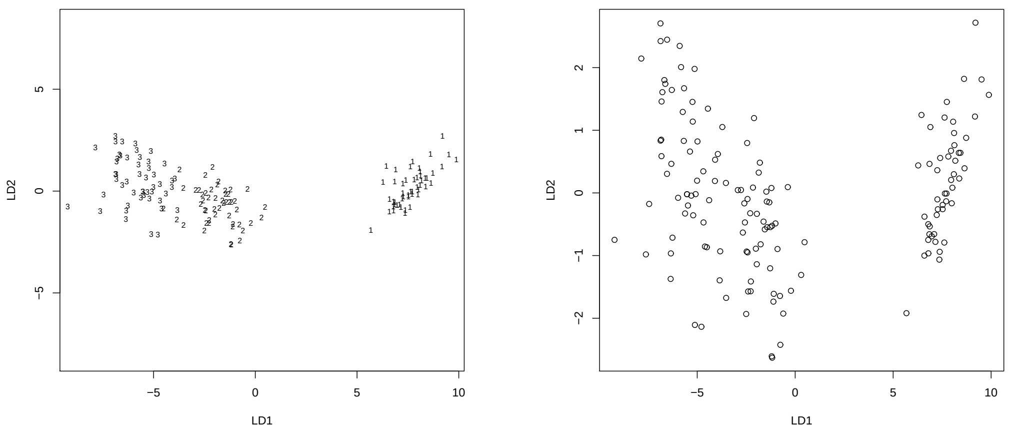
```
> attach(irisnf)
> library(MASS)
> iris.lda<-lda(Variety~Sepal.l+Sepal.w+Petal.l+Petal.w)
> iris.crimcoord<-predict(iris.lda)$x
> iris.lda
Call:
lda(Variety ~ Sepal.l + Sepal.w + Petal.l + Petal.w)

Prior probabilities of groups:
        1         2         3
0.3333333 0.3333333 0.3333333

Group means:
  Sepal.l Sepal.w Petal.l Petal.w
1   5.006   3.428   1.462   0.246
2   5.936   2.764   4.260   1.326
3   6.588   2.974   5.552   2.026

Coefficients of linear discriminants:
               LD1          LD2
Sepal.l  0.8129912  0.006189401
Sepal.w  1.5569539  2.157270079
Petal.l -2.1987600 -0.919346099
Petal.w -2.8266164  2.825877950

Proportion of trace:
   LD1    LD2
0.9912 0.0088
> iris.crimcoord
[produces the coordinates of all 150 data points, projected
into discriminant space]
> plot(iris.lda)
> plot(iris.crimcoord)
```

Notice that `iris.lda` prints everything in `iris.lda`. Most interesting are the `Coefficients of linear discriminants` which give the eigenvectors of $W^{-1}B$ corresponding to non-zero eigenvalues (elements give loadings of variables in each crimcoord). The proportion of the trace gives the relative sizes of the corresponding eigenvalues.

In this case, we have 50 objects in each group; the `lda` function uses a Bayesian approach, where the prior probabilities of the groups are also used in the analysis.

The first eigenvector can be interpreted here as contrasting the size of the petal with the size of the sepal, and shows that this is the primary way to discriminate between the groups. The second eigenvector needs to be interpreted with care – after all, it didn't really come out of the analysis in the same way as the first – but it does seem to be a general measure of width, as neither the sepal nor petal lengths seem to play a large part in the eigenvector. In any case, because the second eigenvalue is so much smaller than the first, the second eigenvector plays hardly any role in discriminating between the groups.

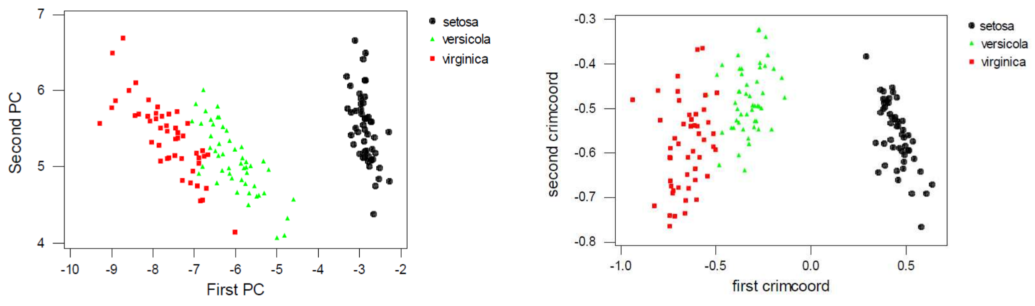
The two plot commands produce:



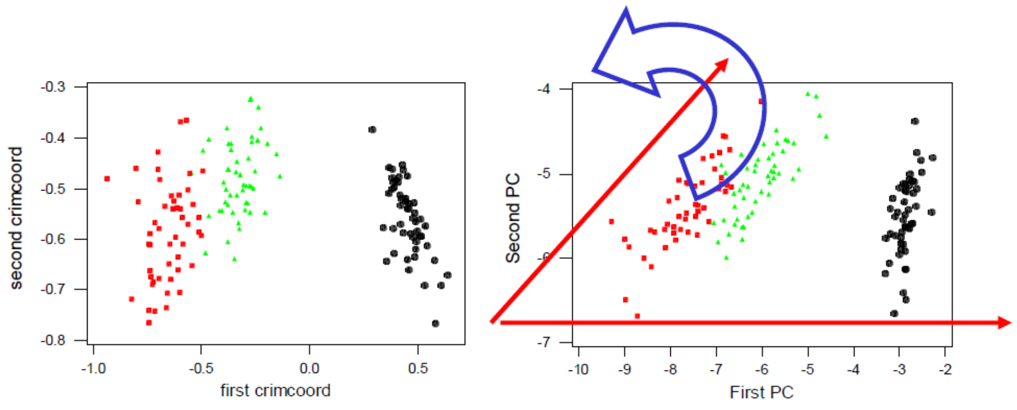
Note that the crimcoords plot doesn't label the points by group; on the other hand, it does fill the plotting space more.

The crimcoords plot gives slightly better separation than the plots on principal components. We can use the same methods as PCA to interpret the variables which separate the groups most.

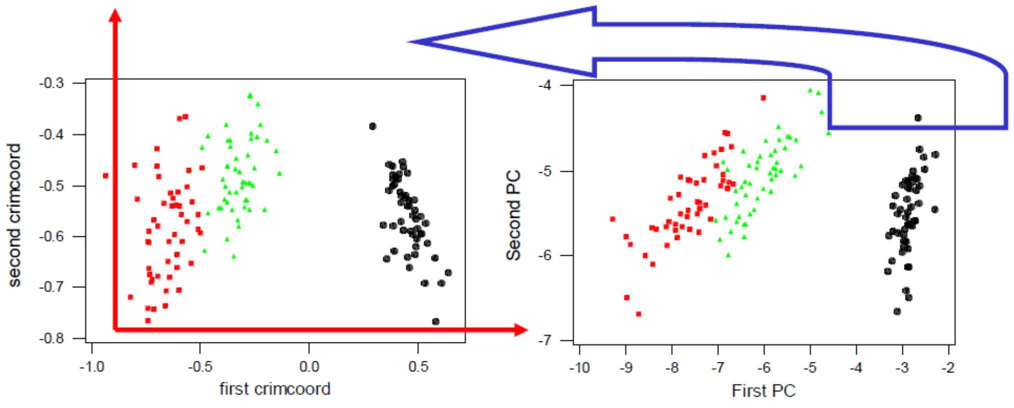
Let's plot the iris data on principal components and crimcoords:



Note the distortion of the data on the crimcoords – it is as if the true axes have been stretched from an acute angle:



and made into a right angle by pulling out one axis to stretch the data space:



Prior probabilities

In R, the function `lda()` assumes by default that prior probabilities of group membership are equal to the observed proportions in the training data of the k groups. To override this, the call to `lda()` should include a parameter `prior=c`$(\mathtt{p}_1, \mathtt{p}_2, \ldots, \mathtt{p}_k)$. In the example on the iris data, the training data had 50 observations in each group, and so the prior probabilities were taken, by default, to be equal. The effect of taking prior probabilities different from the observed proportions is to alter the estimate of the within-groups variance W so that instead of using weights $(n_i - 1)/(n - k)$, it uses weights p_i, i.e., W is estimated as $W = \sum_{i=1}^{k} p_i S_i$, instead of $\frac{1}{n-k} \sum_{i=1}^{k} (n_i - 1) S_i$.

However, the major difference is in the function `predict.lda()` where by default the prior probabilities are taken from those in the original call to `lda()`. This can be overridden by specifying `prior` as something different. This can be useful if there is good prior information on group membership probabilities of test data. Similar comments apply to the function `qda()` which performs quadratic discriminant analysis (see the help system in R).

Costs

It may be that classifying an object in group 1 as a member of group 2 has a much higher "cost" than classifying an object in group 2 as a member of group 1. For example, if a medical diagnosis might involve a potentially fatal condition, then failing to diagnose it is significantly more costly than concluding that the patient has the condition when it is, in fact, not.

We shall not say anything about costs, but refer the reader to textbooks such as that by Johnson and Wichern.

5.4 Application to informal (data analytic) classification

One of the main uses of linear discriminant analysis is to informal data classification.

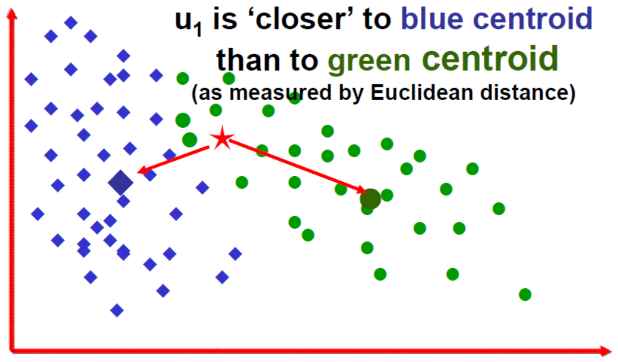
Suppose we have k groups of "reference" data and a set of r unknown points u_j, $j = 1, \ldots, r$, and we wish to classify each of the r points into one of the k groups (e.g., multivariate data from reliably diagnosed patients with one of k different conditions, wish to diagnose r new patients on whom the same measurements are made).

Suppose we have calculated the k reference group means or *centroids* $\{\overline{x}_i \mid i = 1, \ldots, k\}$. It is natural to assign an observation u to that group to which it is nearest, in some sense.

We might measure the distance of u from the group i centroid as $D(i)$, where $D^2(i) = (u - \overline{x}_i)' M (u - \overline{x}_i)$, where M is some suitable weighting matrix which we require to be positive semi-definite to ensure that $D^2(i) \geq 0$.

There are many possible choices for the weight matrix M. The simplest is to take $M = I_p$, and then D is the Euclidean distance

$$D^2(i) = (u - \overline{x}_i)'(u - \overline{x}_i).$$



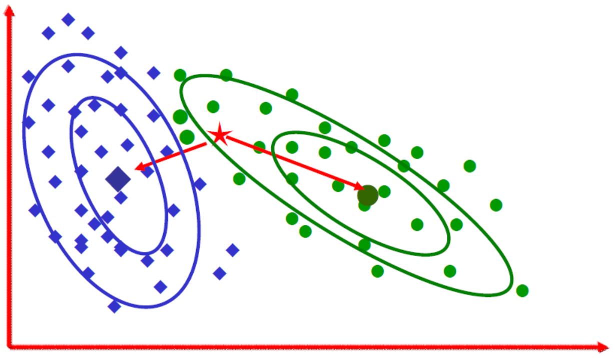
But u_1 is clearly in the middle of a group of green points, and the issue seems to be that the green data appears to be more widely separated than the blue data.

The same can happen in the univariate case: imagine one set of points distributed with a normal distribution with mean -1 and variance 0.01, and another normal distribution with mean $+1$ and variance 100. Then a new reading of -0.5 is 5 standard deviations from the first mean, and only 0.15 standard deviations from the second, so it is clearly more likely to be associated with the second distribution. (In fact, points like -2 are also more likely to come from the second distribution, even though -2 is the other side of -1 from $+1$.) So this method isn't really very good! The problem is that Euclidean distance isn't taking account of the spread of the data.

Even better is to take $M = S_i^{-1}$, giving

$$D^2(i) = (u - \overline{x}_i)' S_i^{-1} (u - \overline{x}_i),$$

which requires $n > p$ to ensure that S_i is non-singular. (This is the *group i squared Mahalanobis distance*). This takes account the shape of the density of the data:



The new point now looks more likely to be green than blue, because of the shape of the spread of the data.

But another alternative is to use the crimcoords. So we take $M = A_t A_t'$, where A_t is the matrix of the first t eigenvectors of $W^{-1}B$ (i.e., it projects the data onto the first t

crimcoords); this gives

$$D^2(i) = (u - \overline{x}_i)' A_t A_t' (u - \overline{x}_i) = [A_t'(u - \overline{x}_i)]'[A_t'(u - \overline{x}_i)],$$

and so is equivalent to projecting the data into discriminant space and then using Euclidean distance. This is *discriminant space distance.* Yet another possibility is to take $M = W^{-1}$, giving

$$D^2(i) = (u - \overline{x}_i)' W^{-1} (u - \overline{x}_i),$$

which is the "average" of the measures – sensible if there are good reasons for expecting the covariances in the different groups to be similar. This is known as the squared *Mahalanobis distance of u from the group mean.*

All of these, and many more, are used in practice, and they may give slightly different results in classification. It may be difficult to determine precisely which criterion ready-made analyses in packages actually use. The most common used criteria are the third (discriminant space distance), and the Mahalanobis distance. If the different groups have substantially different variances, then the group i Mahalanobis distance is a possibility, but extensions of the method to quadratic discriminant analysis may also be useful.

Let's look at how to implement this with the `iris` data set:

```
> attach(irisnf)
> library(MASS)
> samp<-c(sample(1:50,25),sample(51:100,25),sample(101:150,25))
```

The final command here creates a random vector with 25 elements taken from each of the range 1–50, 51–100 and 101–150.

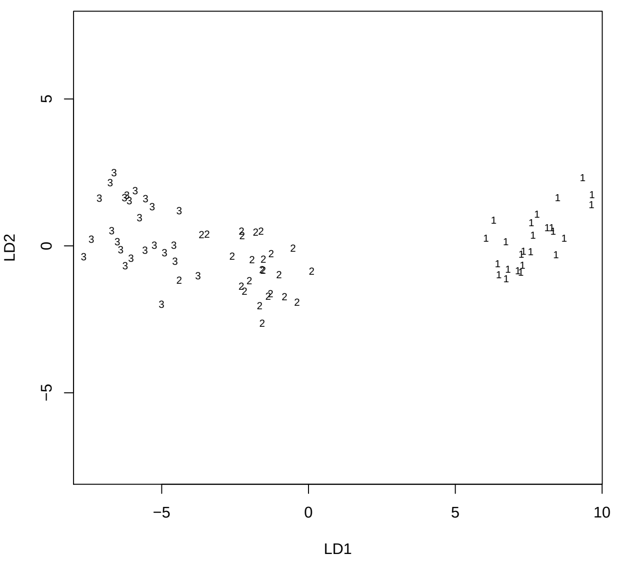
```
> irisnftraining<-irisnf[samp,]
> irisnftest<-irisnf[-samp,]
```

Now `irisnftraining` contains the 75 sample points from `irisnf`, and `irisnftest` contains the remaining 75 points. We want to "train" LDA on the training set, and then use the result to predict the classification of the test data. First, we need to run LDA on the training set, so we need to make this the current data set:

```
> detach(irisnf)
> attach(irisnftraining)
```

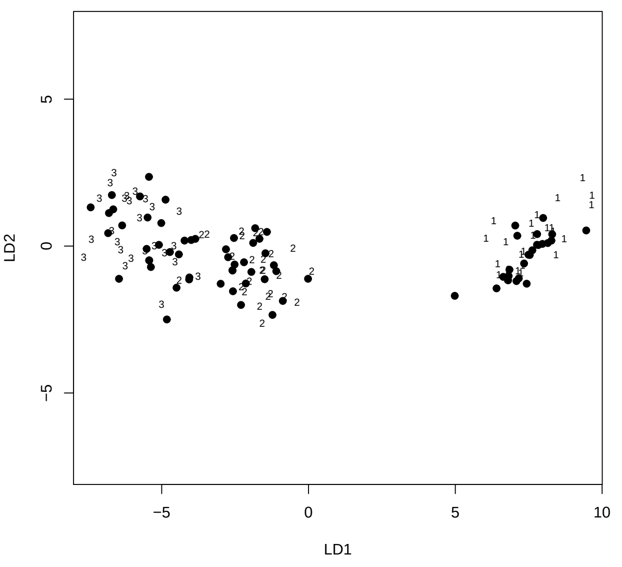
Then run LDA on the training set, and plot the results:

```
> iristrain.lda<-lda(Variety~Sepal.l+Sepal.w+Petal.l+Petal.w)
> plot(iristrain.lda)
```



and then we add to the plot the points coming from the test data (with the final column removed):

```
> points(predict(iristrain.lda,irisnftest[,-5])$x,pch=19)
```



5.5 Summary and conclusions

- Crimcoords, or discriminant coordinates, highlight differences between known groups of observations. Displays on these are strictly distorting the data slightly, since the axes are not orthogonal, but are conventionally drawn as such.
- The first crimcoord is also known as Fisher's Linear Discriminant Function.
- Interpretations of factor loadings and use of scree plots is performed by analogy with PCA.

- Referral to crimcoords allows informal classification of other points of unknown origin.
- Other informal methods for classification are used, such as classifying by minimum Mahalanobis distance.
- The method requires more observations than variables (i.e., $n > p$).

5.6 Further reading

- Look in Wikipedia or other sources for *quadratic discriminant analysis*
- Other techniques for investigating discrimination between known categories are logistic regression, classification trees and neural networks.

Further reading: Canonical Correlation Analysis

Canonical correlation analysis aims to investigate the relationship between two sets of variables – here referred to as the X variables and the Y variables, rather than the dependence of one set on the other which is the purpose of regression analysis (mentioned later). This means that canonical correlation analysis is symmetric in the X and Y variables, though we can accommodate their having different dimensions, say p and q respectively. As with regression methods, we require at least more observations than dimensions, i.e., $n \geq \max(p, q)$, and in particular we need various matrices to be non-singular and so possess inverses.

The approach is to find linear combinations of the X and Y variables that have maximum correlation with each other amongst all such linear combinations. This is analogous to principal component analysis where linear combinations of variables with maximal variance are sought.

Suppose that $X = (X_1, \ldots, X_q)$ and $Y = (Y_1, \ldots, Y_p)$, and n observations are available on each (measured simultaneously). Suppose that $\text{var}(X) = \Sigma_{XX}$, $\text{var}(Y) = \Sigma_{YY}$ and $\text{cov}(X, Y) = \Sigma_{XY}$, where these may be taken as either the population (theoretical) values, if there is a known or assumed underlying distribution generating the data, or the sample values based on the n observations. Let a and b be p- and q-vectors respectively, and consider the correlation between $a'X$ and $b'Y$, noting that their variances are $a'\Sigma_{XX}a$ and $b'\Sigma_{YY}b$ respectively. This is

$$\rho_{XY} = \frac{a'\Sigma_{XY}b}{\sqrt{a'\Sigma_{XX}ab'\Sigma_{YY}b}}.$$

This is independent of the scale of both a and b, so we may impose scale constraints without loss of generality, most conveniently to ensure that the denominator is unity, i.e., that $a'\Sigma_{XX}a = b'\Sigma_{YY}b = 1$.

Maximizing ρ_{XY} subject to these constraints (by introducing λ and μ as Lagrange multipliers) yields

$$\begin{aligned} \Sigma_{XY}b - \lambda\Sigma_{XX}a &= 0 \\ \Sigma_{XY}a - \mu\Sigma_{YY}b &= 0 \end{aligned}$$

after differentiating ρ_{XY} with respect to a and b and setting the results equal to 0. Premultiplying these by a' and b' respectively and recalling the constraints gives $\lambda = \mu = a'\Sigma_{XY}b = \rho_{XY}$. Premultiply the first equation by $\rho_{XY}(\Sigma_{XX})^1$ and the second by $\Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{YY}^{-1}$ and adding gives

$$\Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{XY}a - (\rho_{XY})^2 a = 0,$$

showing that a is an eigenvector of $\Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{XY}$ with eigenvalue ρ_{XY}^2. A further step shows that to maximise the correlation, we need to take the largest eigenvalue. Similar analysis shows that b is the eigenvector corresponding to the largest eigenvalue of $\Sigma_{YY}^{-1}\Sigma_{XY}\Sigma_{XX}^{-1}\Sigma_{XY}$ which is also ρ_{XY}^2.

It is easy to show that further eigenvectors $a_2, \ldots$ and $b_2, \ldots$ maximise the correlation between linear functions of the X and Y variables subject to the constraints of orthogonality with earlier ones.

Example 5.4 (Everitt, chapter 8) The data set `chap8headsize` from the web page associated with Everitt's book give the length (X_1) and breadth (X_2) of first sons, and Y_1 and Y_2 the corresponding measurement for second sons.

Analysis gives

$$\Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{XY} = \begin{pmatrix} 0.323 & 0.317 \\ 0.302 & 0.302 \end{pmatrix}$$
$$\Sigma_{YY}^{-1}\Sigma_{XY}\Sigma_{XX}^{-1}\Sigma_{XY} = \begin{pmatrix} 0.301 & 0.300 \\ 0.319 & 0.323 \end{pmatrix}$$

Both have eigenvalues 0.62 and 0.0029, giving canonical correlations of 0.7885 and 0.0537. The eigenvectors are $a_1 = \begin{pmatrix} 0.727 \\ 0.687 \end{pmatrix}$, $a_2 = \begin{pmatrix} 0.704 \\ -0.710 \end{pmatrix}$, $b_1 = \begin{pmatrix} 0.684 \\ 0.730 \end{pmatrix}$ and $b_2 = \begin{pmatrix} 0.709 \\ -0.705 \end{pmatrix}$.

The first two canonical variates are essentially averages of length and breadth, and so proportional to circumference, showing that the major characteristic shared by brothers is overall head size, with correlation estimated as 0.79. The second canonical variates are contrasts in length and breadth, and so reflect shape. The correlation between the shapes of elder and younger brothers is 0.05, and thus the shapes are virtually unrelated.

Here are a few further comments:

- It may be shown that if one of the variables, say the Y variable, is a group indicator or set of binary dummy variables, then the canonical variates of the X variables are precisely the discriminant functions between the groups. This is the reason for the latter sometimes being referred to as *canonical variates* – they are the linear combinations of the X variables that most highly correlate with the group structure, i.e., discriminate highly between them. The fact that a different scaling constraint is used in the analysis is immaterial since the result is invariant to scale.
- The choice of sign for the eigenvectors is arbitrary, as with PCA.
- Plots of the data referred to the canonical variates (either a_i against a_j for just the X variables or a_i against b_i for all the variables) may be useful ways of displaying the data to investigate structure.
- The number of nonzero eigenvalues (i.e., canonical correlations) is at most $\min(p, q)$.
- As with multivariate regression, we require various matrices to be non-singluar.
- Interpretation of loadings in the canonical variates is similar to that in PCA and LDA, and gives insight into aspects of the data structure.
- R functions for performing canonical correlation analysis are `cancor()` in the `stats` library, and (better) `cc()` in the `CCA` package.

6 Introduction to multivariate statistics

Thus far, we have only considered data-analytic techniques, i.e., methods that depend only on the data received, and which make no assumptions on an underlying distribution of the data generated.

This exploratory data analysis is crucial in data science – it allows the researcher to get a feel for the data set, and might simplify later more formal statistical analyses.

The second part of the course, beginning with this section, considers more formal statistical models and techniques for the analysis of multivariate data.

Unsurprisingly, there are multivariate generalisations of all the standard univariate distributions, as well as distributions that only really make sense in a multivariate context (imagine points distributed uniformly on the surface of a sphere, for example).

However, for much the same reasons as in the univariate case, it is the normal distribution, in a multivariate formulation, which plays the most important role, and we won't consider any other multivariate distributions. There are several reasons why it is most important:

- Often data looks like it is distributed in this way;
- The statistical tests are fairly simple;
- There is a multivariate version of the Central Limit Theorem which states that whatever distribution you choose, for large sample sizes, its mean is distributed as a multivariate normal distribution.

The section starts with the definition of the p-dimensional multivariate normal distribution and its basic properties (mean, variance and sampling properties such as maximum likelihood estimation). This will allow the construction of likelihood ratio tests and thus the extension to several dimensions of the routine 1-dimensional tests such as t-tests and analysis of variance.

In the next chapter, we will develop some of these tests, also with some more complex hypotheses, which can only arise in several dimensions, such as whether the population mean is somewhere on the unit sphere. This requires use of Lagrange multipliers to maximise likelihoods subject to constraints. We will also see a new method for constructing hypothesis tests (the *Union-intersection principle*), which in some circumstances can give a different form of test from a likelihood ratio test, and in others provide a useful additional interpretation of likelihood ratio tests. This topic links in with the idea of projecting data into one dimension, choosing the dimension appropriately, which was encountered in the construction of principal components and crimcoords.

You should re-read the section on the multivariate normal distribution from the first chapter, although the lectures will run through it again briefly.

6.1 The multivariate normal distribution

Let's summarise the results from Section 1.

Definition 6.1 Suppose that μ is a p-vector, and that Σ is a positive definite symmetric $p \times p$-matrix. Then the *multivariate normal distribution* $N_p(\mu, \Sigma)$ is the distribution with probability density function

$$f_x(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)\right).$$

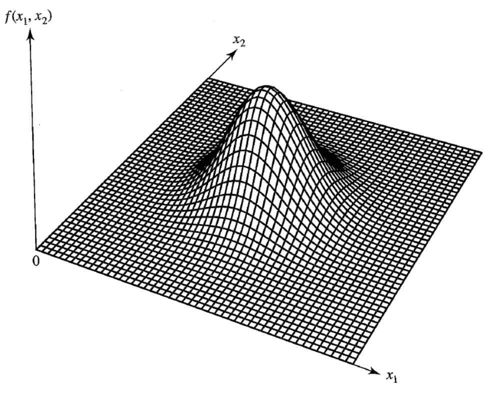
It is an exercise for you to check that this has volume 1, so really is a probability distribution; this is not too hard, once you recall that $y = \Sigma^{-1/2}(x-\mu)$ is distributed as $N_p(0, I_p)$; then the integral breaks up into a product of p integrals which arise in the analogous univariate calculation, and the Jacobian of the transformation is $|\Sigma|^{-1/2}$.

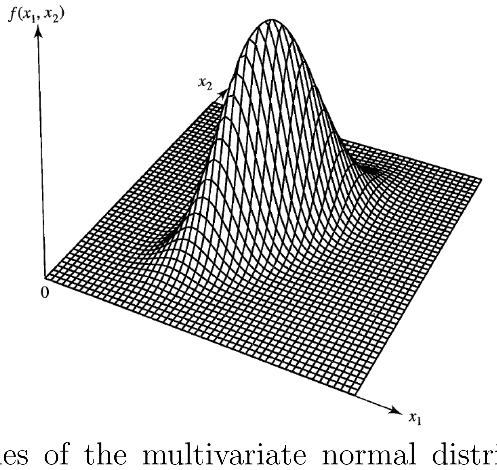
Remark 6.2 The mean of $N_p(\mu, \Sigma)$ turns out to be μ, and the variance is Σ (Lemma 1.20).

Notice that when $p = 1$, we recover the usual normal distribution $N(\mu, \sigma^2)$, where $\Sigma = \sigma^2$; the quantity $(x-\mu)'\Sigma^{-1}(x-\mu)$ simplifies to $\frac{(x-\mu)^2}{\sigma^2}$. We have already interpreted it as a "squared statistical distance" (the squared *Mahalanobis distance*) between x and μ.

It should be clear that the paths of values yielding a constant density are ellipsoids: the multivariate normal density is constant on surfaces where $(x-\mu)'\Sigma^{-1}(x-\mu)$ is constant. The axes of each ellipsoid of constant density are in the direction of the eigenvectors of Σ^{-1} (recall that these are the same as the eigenvectors of Σ, but if $\Sigma x = \lambda x$, then $\Sigma^{-1}x = \lambda^{-1}x$), and their lengths are proportional to the reciprocals of the square roots of the eigenvalues of Σ^{-1}.

Here are a couple of pictures of the distribution (bivariate: the first picture has $\Sigma = I_2$, and the second has $\Sigma = \begin{pmatrix} 1 & \frac{3}{4} \\ \frac{3}{4} & 1 \end{pmatrix}$):





Various (largely unsurprising) properties of the multivariate normal distribution were given without proof in section 1.

The most useful is the following:

If $x \sim N_p(\mu, \Sigma)$, then

1. $(x-\mu)'\Sigma^{-1}(x-\mu)$ is distributed as χ^2_p, the chi-squared distribution with p degrees of freedom;

2. In particular, the solid ellipsoid $\{x \mid (x-\mu)'\Sigma^{-1}(x-\mu) \leq \chi^2_p(\alpha)\}$ has probability $1-\alpha$.

This follows from the proof of Lemma 1.20, but we sketch a proof next:

If $x \sim N_p(\mu, \Sigma)$, then $y = \Sigma^{-1/2}(x - \mu)$ is distributed like $N_p(0, I_p)$, and this shows that

$$(x-\mu)'\Sigma^{-1}(x-\mu) = y'y = \sum_{i=1}^{p} Y_i^2 \sim \chi_p^2,$$

where each Y_i^2 is the square of a $N(0,1)$ (univariate) random variable, and the result follows.

One of the consequences of the properties is that the marginal distributions of the individual variables of a multivariate normal distribution is a univariate normal distribution.

6.2 Assessing normality and detecting outliers

Assessing (multivariate) normality is not always easy; there are many things that can go wrong. It is possible to construct a multivariate distribution which is not normal, but whose marginal distributions are normal. However, distributions which look normal in low dimensions, but not in high dimensions are rarely encountered in practice; if a distribution is not normal, one can generally see this in 1 or 2 dimensions. However, although it is sensible to check each marginal component of sample data for normality (e.g., by probability plotting), it does not follow that the multivariate data are satisfactorily multivariate normally distributed for the statistical tests and other procedures to be appropriate.

With a large number of samples, a plot is often a good idea; one can make a Q-Q plot (quantile-quantile), where, say, one orders the data (perhaps by squared statistical distance from the mean), and then reads off from tables what the result would be using a normal distribution. One can then compute a correlation coefficient to compare the data with the normal distribution, and compare it against some critical point to see whether a hypothesis of normality at some given level of significance should be rejected.

More precisely, the further check is provided by the squared statistical distances of each observation from the mean

$$D_i^2 = (x_i - \overline{x})'S^{-1}(x_i - \overline{x}),$$

which should have approximately a χ^2 distribution with p degrees of freedom. These distances will not actually be independent, but are nearly so, consequently a test of normality is provided by assessing the D_i^2 as a sample of observations from a χ_p^2-distribution. Everitt provides a function `chisplot()` for producing a χ^2-probability plot (i.e., ordered observations against quantiles of χ_p^2).

If data is not normal, it can be that further thought/analysis can be made to make the data look more normal. See, for example, Johnson-Wichern for more.

The same technique can be used to detect outliers; as well as plotting the points, if possible, to detect unusual observations visually, we can calculate the squared statistical distances from the mean for each point, and consider whether any are unusually far away.

6.3 Random samples

Suppose that $x \sim N_p(\mu, \Sigma)$, and that $x_1, \dots, x_n$ are observations of x. Define $\overline{x} = \frac{1}{n}\sum_{i=1}^n x_i$, and

$$S = \frac{1}{n-1}(x-\overline{x})(x-\overline{x})' = \frac{1}{n-1}\left(\sum_{i=1}^n x_i x_i' - n\overline{x}\overline{x}'\right).$$

(Recall that x is a vector of length p, so that $(x-\overline{x})(x-\overline{x})'$ is the product of a $p \times 1$ and $1 \times p$ matrix, so is a $p \times p$ matrix.) Then $E(\overline{x}) = \mu$ and $\operatorname{var}(\overline{x}) = \frac{1}{n^2}\sum_{i=1}^n \operatorname{var}(x_i) = \frac{1}{n}\Sigma$.

We have already shown that $\overline{x} \sim N_p(\mu, \frac{1}{n}\Sigma)$ (Corollary 1.22). In fact, this is a special case of the more general multidimensional central limit theorem (proved in Chapter 1):

Theorem 6.3 (Multivariate Central Limit Theorem) *Let $x_1, \dots, x_n$ denote independent observations from any population with mean μ and finite covariance Σ. Then $\sqrt{n}(\overline{x} - \mu)$ has approximately an $N_p(0, \Sigma)$ distribution when n is large, and large relative to p.*

Often one makes an assumption that data are distributed in accordance with a multivariate normal distribution in order to develop some statistical tests. Sometimes, when the sample size is large, and the test solely involves the behaviour of $\overline{x}$, we can use the Central Limit Theorem to get some approximation to mormality, and then the precise distribution is less crucial.

Generally, then, our hypothesis tests will require an assumption that the distribution of the population is (multivariate) normal, or, if the test involves only $\overline{x}$, that there are a large number of samples; then, the nearer the population is to normal, the fewer samples that will be required that the tests will be valid.

6.4 The distribution of the sample mean and variance matrix

Suppose that we sample n observations of a multivariate normal distribution $N_p(\mu, \Sigma)$. This sample has a sample mean $\overline{x}$ and a sample variance matrix S; in this section, we try to understand the distributions of $\overline{x}$ and S.

We have already seen the distribution of $\overline{x}$ (Corollary 1.22):

Corollary 6.4 *Suppose that $x \sim N_p(\mu, \Sigma)$, and that $x_1, \dots, x_n$ are observations of x. Define $\overline{x} = \frac{1}{n}\sum_{i=1}^n x_i$. Then*

$$\overline{x} \sim N_p(\mu, \tfrac{1}{n}\Sigma).$$

We will now also give the distribution of the variance matrix S, and will introduce another multivariate distribution.

In the univariate case, we know that $\overline{x}$ is normally distributed with mean μ and variance σ^2/n, and we have seen the same result in the multivariate case also. We also know the distribution of the sample variance in the univariate case: we know that $(n-1)s^2 = \sum_{i=1}^n (x_i - \overline{x})^2$ is distributed as σ^2 times a χ^2_{n-1}-distribution. We can write this as the sum of $n-1$ squares of i.i.d. $N(0, \sigma^2)$-variables, $Z_1^2 + \cdots + Z_{n-1}^2$.

In the multivariate case, we have a similar distribution, known as the *Wishart distribution.* In particular,

Definition 6.5 Define the *p-dimensional Wishart distribution with scale matrix* Σ *and* m *degrees of freedom* by

$$W_p(\Sigma, m) = \sum_{i=1}^{m} Z_i Z_i',$$

where each $Z_i \sim N_p(0, \Sigma)$.

This is a matrix generalisation of the χ^2-distribution: if $p = 1$, then $W = \sum_{i=1}^{m} x_i^2$ with $x_i \sim N(0, \sigma^2)$. Note that it is a $\frac{1}{2}p(p+1)$-dimensional distribution ($W = ZZ'$ is symmetric). Its "standard form" is when $\Sigma = I_p$.

Then (a proof is given in Cox's book, p.34)

$$(n-1)S \sim W_p(\Sigma, n-1) \quad \text{independently of } \overline{x}.$$

Cox also explicitly gives the pdf of $W_p(\Sigma, m)$.

Its properties are generalisations of those of the χ^2-distribution, e.g., additivity on the degrees of freedom parameter: if $U \sim W_p(\Sigma, m)$ and $V \sim W_p(\Sigma, n)$ independently, then $U + V \sim W_p(\Sigma, m+n)$. Its key use is as an intermediate step in deriving the distribution of things of real interest.

Since Σ is unknown, we cannot use the distribution of $\overline{x}$ to make inferences about μ. However, S provides information about Σ, and the distribution of S does not depend on μ. This allows us to develop a statistic for making inferences about μ.

6.5 Testing for normal population mean

Consider a univariate distribution $N(\mu, \sigma^2)$, where μ is not known, and let's ask whether data is consistent with a hypothesis $H_0: \ \mu = \mu_0$. Then the alternative hypothesis is $H_A: \ \mu \neq \mu_0$.

Given data $x_1, \ldots, x_n$, we form the test statistic

$$t = \frac{(\overline{x} - \mu_0)}{s/\sqrt{n}},$$

where $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the sample mean, and $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$ is the sample variance.

This test statistic has a t-distribution with $n-1$ degrees of freedom. We reject H_0 is the observed value of $|t|$ is too large. Equivalently, we reject H_0 if its square

$$t^2 = \frac{(\overline{x} - \mu_0)^2}{s^2/n} = n(\overline{x} - \mu_0)(s^2)^{-1}(\overline{x} - \mu_0)$$

is large. This RHS gives another occurrence of our squared statistical distance.

We reject H_0 at significance level α when

$$n(\overline{x} - \mu_0)(s^2)^{-1}(\overline{x} - \mu_0) > t_{n-1}^2(\alpha/2). \tag{6.1}$$

On the other hand, if H_0 is accepted, then μ_0 is a plausible value for the normal mean. Indeed, any μ_0 where the inequality of (6.1) fails, would be accepted. This means that μ_0 would lie in the $100(1-\alpha)\%$ confidence interval

$$\overline{x} - t_{n-1}(\alpha/2)\frac{s}{\sqrt{n}} \leq \mu_0 \leq \overline{x} + t_{n-1}(\alpha/2)\frac{s}{\sqrt{n}}.$$

By analogy with the above, we can ask when a given $p \times 1$-vector μ_0 is a plausible mean for a multivariate normal distribution. The multivariate analogue of the t-distribution is:

$$T^2 = (\overline{x} - \mu_0)' \left(\frac{1}{n}S\right)^{-1} (\overline{x} - \mu_0) = n(\overline{x} - \mu_0)' S^{-1} (\overline{x} - \mu_0). \tag{6.2}$$

This statistic is *Hotelling's T^2-statistic.* Since this statistic is p-dimensional, and has $n-1$ degrees of freedom, we also write $T^2(p, n-1)$ for the quantity in equation (6.2).

Luckily, we do not need to have separate tables for the T^2-distribution; Hotelling showed that

Theorem 6.6 $T^2(p, n-1) = \frac{(n-1)p}{n-p} F_{p,n-p}$.

We omit the proof (but one is in Chatfield and Collins, pp.109–111).

We explain why this crops up. Recall that $n(\overline{x}-\mu)'\Sigma^{-1}(\overline{x}-\mu) \sim \chi^2_p$, and this allows us to perform hypothesis testing on μ. Often, however, Σ is unknown, but we still want to do hypothesis testing on μ, and we need to use the sample variance S instead.

This leads to Hotelling's T^2-test:

> If $x_1, \ldots, x_n$ are a random sample from an $N_p(\mu, \Sigma)$ population, with sample mean and variance $\overline{x}$ and S respectively, then
>
> $$P\left(n(\overline{x} - \mu)' S^{-1} (\overline{x} - \mu) > \frac{(n-1)p}{n-p} F_{p,n-p}(\alpha)\right) = \alpha, \tag{6.3}$$
>
> where $F_{p,n-p}(\alpha)$ is the upper $100\alpha\%$ percentile of the $F_{p,n-p}$-distribution.

In particular, if we want to make a test of the hypothesis $H_0 : \ \mu = \mu_0$ against $H_A : \ \mu \neq \mu_0$, we reject H_0 at the α-level of significance if the observed T^2-statistic

$$T^2 = n(\overline{x} - \mu_0)' S^{-1} (\overline{x} - \mu_0) > \frac{(n-1)p}{n-p} F_{p,n-p}(\alpha).$$

We can relate Hotelling's T^2-distribution to the Wishart distribution. For this, we rewrite

$$\begin{aligned} T^2 &= n(\overline{x} - \mu_0)' S^{-1} (\overline{x} - \mu_0) \\ &= \sqrt{n}(\overline{x} - \mu_0)' \left(\frac{\sum_{i=1}^n (x_i - \overline{x})(x_i - \overline{x})'}{n-1}\right)^{-1} . \sqrt{n}(\overline{x} - \mu_0), \end{aligned}$$

and the middle matrix is the one giving the Wishart distribution:

$$T^2(p, n-1) = N_p(0, \Sigma)' \left(\frac{1}{n-1} W_p(\Sigma, n-1)\right)^{-1} N_p(0, \Sigma).$$

This should remind you of an analogous statement in the univariate case:

$$t^2_{n-1} = \sqrt{n}(\overline{x} - \mu_0)(s^2)^{-1}\sqrt{n}(\overline{x} - \mu_0),$$

where the middle term is distributed like a (scaled) χ^2-variable, and the outer terms like a normal random variable.

6.6 Confidence regions

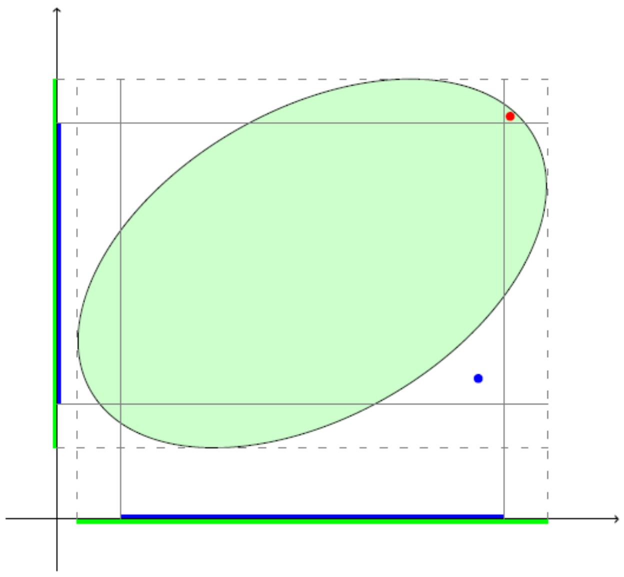
Hotelling's T^2-statistic gives us a *confidence region* for the mean μ of a p-dimensional multivariate normal distribution, given a set of observations $x_1, \ldots, x_n$:

> A $100(1-\alpha)\%$ confidence region for the mean μ of a p-dimensional multivariate normal distribution is the ellipsoid of all μ such that
>
> $$n(\overline{x}-\mu)'S^{-1}(\overline{x}-\mu) \leq \frac{np-p}{n-p}F_{p,n-p}(\alpha).$$

So the confidence region consists of all μ_0 for which the T^2-test would not reject $H_0: \ \mu = \mu_0$ at significance level α.

Before we go on, we should notice that the ellipsoidal shape of the confidence region already implies that multivariate and simultaneous marginal univariate tests will look different: in the multivariate case, confidence regions are ellipsoidal, whereas we will now explain that simultaneous univariate regions will be cuboidal.

Here is a picture in 2 dimensions of a bivariate confidence region.



Let's suppose that the green elliptical region represents a 95% (say) multivariate confidence region for a hypothesis test (assuming normality). That means, essentially, that were we to run an experiment more and more often, eventually the proportion of results which would land in the green region would be 95%.

Now consider the marginal distributions of each component. If we project the green ellipse onto the x-axis, clearly *more than* 95% of the time, our results will end up in the projection, since this includes not only the green region, but also other regions. So there is some smaller interval on the x-axis which represents the 95% confidence interval for the corresponding variable. The same happens with the other axis, and one might expect (assuming some sort of independence) that 0.95^2 of the time, our data will have both its components in the univariate confidence intervals. To get a 95% rectangular box, we'd need to use $\sqrt{0.95}$-confidence intervals on each axis, perhaps the intervals marked in blue.

This means that 95% of the time, our data has both its x-component in the horizontal blue interval, and its y-component in the vertical blue interval.

But the main thing to note is that these regions are different shapes, and this means that each region has a point not in the other. In particular, the red point lies inside the ellipse, but outside the rectangular box, so the red point corresponds to data which is not significant in a multivariate test, but would be significant for simultaneous univariate tests. Indeed, as drawn, the red point would be significant for each univariate test, even at the higher $\sqrt{0.95}$ level of significance, even though it is not significant for a multivariate test. In the same way, the blue point is not significant for either univariate test, but since it lies outside the ellipsoid, it is significant for the multivariate test.

So multivariate tests really are different from univariate tests, simply because of the shape of the confidence regions. One can imagine ellipsoids in 3 dimensions of various shapes, and you can try to convince yourself that this sort of phenomenon is likely to be even more frequent in higher dimensions, and is also more likely when the variables are correlated.

However, the situation is very different if we think about *all* possible projections. Indeed, since the shape of the confidence ellipsoid is convex, if a statistic lies outside the confidence interval, then there will be some 1-dimensional projection (*not necessarily onto a variable*) whose projection lies outside the projection of the ellipsoid. This means that our confidence of lying inside the ellipsoid is exactly equal to the confidence that the projection lies inside all the 1-dimensional projections of the ellipsoid.

Indeed, this gives another way to recover the T^2-statistic, which we introduced earlier merely as a higher-dimensional analogue of a 1-dimensional test. We need to find the vector a such that the projection $a'X$ is as significant as possible. But if $X \sim N_p(\mu, \Sigma)$, then $a'X \sim N(a'\mu, a'\Sigma a)$. Consider the linear combination $z = a'x = a_1x_1 + \cdots + a_px_p$ of some data. Then the mean and variance of z are related to the mean and variance of x by:

$$\overline{z} = a'\overline{x}$$
$$s_z^2 = a'Sa$$

(recall that z is univariate, so we denote its variance by s_z^2). For a single fixed a, a $100(1-\alpha)\%$ confidence interval for $\mu_z = a'\mu$ is given by

$$t = \frac{\overline{z} - \mu_z}{s_z/\sqrt{n}} = \frac{\sqrt{n}(a'\overline{x} - a'\mu)}{\sqrt{a'Sa}},$$

and leads to the interval

$$a'\overline{x} - t_{n-1}(\alpha/2)\frac{\sqrt{a'Sa}}{\sqrt{n}} \leq a'\mu \leq a'\overline{x} + t_{n-1}(\alpha/2)\frac{\sqrt{a'Sa}}{\sqrt{n}}. \tag{6.4}$$

Note that this is the *blue* interval in the picture on p.105. We will work out the equations of the green interval later.

Under the null hypothesis $H_0 : \mu = \mu_0$, we expect z to have mean $a'\mu_0$, and use a t-test for the z-values given by

$$\mathcal{T}^2 = \frac{|a'(\overline{x} - \mu_0)|}{\sqrt{a'Sa/n}}.$$

This is maximised when its square is maximised, so we need to maximise

$$\frac{n(a'(\overline{x} - \mu_0))^2}{a'Sa},$$

and the usual Lagrange multiplier argument shows that the maximum is attained when a is proportional to $S^{-1}(\overline{x} - \mu_0)$, and then the statistic we measure is

$$\mathcal{T}^2 = n(\overline{x} - \mu_0)' S^{-1}(\overline{x} - \mu_0),$$

exactly Hotelling's T^2-test.

Let's just do this. We want to maximise $\mathcal{T}^2$, so we scale a (which leaves $\mathcal{T}^2$ unchanged) so that $a'Sa = 1$. Then we have the objective function

$$\begin{aligned} \Omega &= n(a'(\overline{x} - \mu_0))^2 - \lambda(a'Sa - 1) \\ &= na'(\overline{x} - \mu_0)(\overline{x} - \mu_0)'a - \lambda(a'Sa - 1). \end{aligned}$$

Then

$$\frac{\partial \Omega}{\partial a} = 2n(\overline{x} - \mu_0)(\overline{x} - \mu_0)'a - 2\lambda Sa,$$

so a is an eigenvector of $nS^{-1}(\overline{x} - \mu_0)(\overline{x} - \mu_0)'$, with eigenvalue λ (multiplying the displayed equation by a' shows that λ is actually the value of $\mathcal{T}^2$). But $(\overline{x} - \mu_0)(\overline{x} - \mu_0)'$ has rank 1, so the same is true for $nS^{-1}(\overline{x} - \mu_0)(\overline{x} - \mu_0)'$, and there is therefore only one non-zero eigenvalue. We can verify that $S^{-1}(\overline{x} - \mu_0)$ is an eigenvector:

$$\begin{aligned} & n(S^{-1}(\overline{x} - \mu_0)(\overline{x} - \mu_0)')(S^{-1}(\overline{x} - \mu_0)) \\ = \; & n(S^{-1}(\overline{x} - \mu_0))((\overline{x} - \mu_0)'S^{-1}(\overline{x} - \mu_0)) \\ = \; & n((\overline{x} - \mu_0)'S^{-1}(\overline{x} - \mu_0))(S^{-1}(\overline{x} - \mu_0)) \end{aligned}$$

as $(\overline{x} - \mu_0)'S^{-1}(\overline{x} - \mu_0)$ is a scalar. So a should be proportional to $S^{-1}(\overline{x} - \mu_0)$, and its eigenvalue is $\lambda = n(\overline{x} - \mu_0)'S^{-1}(\overline{x} - \mu_0)$, which is exactly Hotelling's T^2-statistic.

Looking at 1-dimensional projections is often a useful trick in multivariate statistics (and we'll see more examples later).

Although the ellipsoid is hard to visualise in many dimensions, we do know how to compute the relative lengths of the axes and their directions. Indeed, the centre of the ellipsoid is at $\widehat{\mu} = \overline{x}$ (the MLE, as we shall see); the directions of the axes are the eigenvectors of S (or, equivalently, of S^{-1}), and their lengths are proportional to the square roots of the eigenvalues. More precisely:

Proposition 6.7 *If $e_1, \dots, e_p$ are the eigenvectors of S, with corresponding eigenvalues $\lambda_1, \dots, \lambda_p$, then the axes of the confidence ellipsoid are*

$$\pm\sqrt{\lambda_i}\sqrt{\frac{np - p}{n(n - p)} F_{p,n-p}(\alpha)}\, e_i$$

and the centre is at $\overline{x}$.

We omit the proof, but hopefully you will try to see why the terms appear. We'll now work out the projections of the ellipsoid onto a vector a, to get the *green* intervals in the picture on p.105. We'll write $c^2 = \frac{np-p}{n-p} F_{p,n-p}(\alpha)$ for the threshold given by the T^2-statistic in Hotelling's Theorem 6.6. Then the interior of the confidence ellipsoid is given by $n(\overline{x} - \mu)'S^{-1}(\overline{x} - \mu) < c^2$, and the boundary is given by $n(\overline{x} - \mu)'S^{-1}(\overline{x} - \mu) = c^2$.

Theorem 6.8 *The projection of the confidence ellipsoid onto a vector a is given by the following interval:*

$$a'\overline{x} - c\frac{\sqrt{a'Sa}}{\sqrt{n}} \le a'\mu \le a'\overline{x} + c\frac{\sqrt{a'Sa}}{\sqrt{n}}.$$

Proof. For simplicity, we'll assume that our sample satisfies $\overline{x} = 0$, just to ease the notation. This time, we are given our data set, and can work out $\overline{x}$ and S, and we are also given a.

We will find the value of μ lying on the boundary of the ellipsoid which maximises the projection $a'\mu$. This projection is then the extremity of the projection of the ellipsoid onto a.

So we want to choose μ to maximise $a'\mu$, subject to $n\mu' S^{-1}\mu = c^2$, the boundary of the ellipsoid. Equivalently, we maximise $(a'\mu)^2 = \mu' a.a'\mu$ (as $a'\mu$ is a scalar, $\mu' a = (a'\mu)' = a'\mu$).

This is again a Lagrange multiplier problem: form

$$\Omega = \mu' aa'\mu - \lambda(n\mu' S^{-1}\mu - c^2),$$

and then

$$\frac{\partial\Omega}{\partial\mu} = 2aa'\mu - 2n\lambda S^{-1}\mu.$$

Setting this equal to 0 gives

$$aa'\mu = n\lambda S^{-1}\mu,$$

and multiplying by S shows that μ is an eigenvector of the rank 1 matrix Saa'. There is only one non-zero eigenvalue, and it is easy to see that $Saa'(Sa) = (Sa)(a'Sa) = (a'Sa)(Sa)$ as $a'Sa$ is a scalar, so Sa is the unique eigenvector with non-zero eigenvalue. Then $\mu = t.Sa$ for some t.

But μ satisfies $n\mu' S^{-1}\mu = c^2$, so this gives

$$c^2 = nta'S.S^{-1}.(tSa) = nt^2 a'Sa,$$

and we read off that $t^2 = \frac{c^2}{na'Sa}$. This gives μ, and then we need $a'\mu = \pm t.a'Sa = \pm c\sqrt{a'San}$. This is the confidence interval for $a'\mu$ around 0. Substituting in the value of c, we get the interval

$$-c\frac{\sqrt{a'Sa}}{\sqrt{n}} \le a'\mu \le c\frac{\sqrt{a'Sa}}{\sqrt{n}},$$

as required. $\square$

Thus:

Corollary 6.9 *Let $x_1, \ldots, x_n$ be a random sample from an $N_p(\mu, \Sigma)$ distribution. Then, for all a, the interval*

$$\left(a'\overline{x} - \sqrt{\frac{np-p}{n(n-p)}F_{p,n-p}(\alpha)a'Sa}, a'\overline{x} + \sqrt{\frac{np-p}{n(n-p)}F_{p,n-p}(\alpha)a'Sa}\right)$$

will contain $a'\mu$ with probability at least $1-\alpha$.

Note that this simultaneous region is the same as the multivariate one, since we are not just considering projections onto the axes (i.e., the marginal distributions), but projections onto *all* possible directions.

However, let's stress again that this is *not* the same as the region we get simply by looking at projections onto each axis. If we choose a to be a co-ordinate vector

$(0, \dots, 0, 1, 0, \dots, 0)$, we get the green projections onto each axis, but then the simultaneous confidence region that this gives is then smallest box with sides parallel to the co-ordinate axes, and which contains the confidence ellipsoid. Since this box is, in general, much larger than the ellipsoid, one can see that generally the probability that a random sample lies in this box will be much larger than α, and that this method is producing a very cautious overestimate.

6.7 Large sample inferences about a population mean

With large samples, we can form confidence tests even without an assumption of normality (essentially due to the Central Limit Theorem). We have already seen that

$$n(\overline{x} - \mu)' S^{-1} (\overline{x} - \mu)$$

is approximately distributed as a χ^2_p-distribution. Then we find a test for the population mean:

> Let $x_1, \dots, x_n$ be a random sample from a distribution with mean μ and (positive definite) variance matrix Σ. When $n - p$ is large, we can reject $H_0 : \mu = \mu_0$ at a level of significance (approximately) α if the observed statistic
> $$n(\overline{x} - \mu)' S^{-1} (\overline{x} - \mu) > \chi^2_p(\alpha).$$

Notice that if $x_1, \dots, x_n$ come from a normal distribution, then this test differs from the one in (6.3). However, for n large relative to p, it turns out that $\frac{np-p}{n-p} F_{p,n-p}(\alpha)$ and $\chi^2_p(\alpha)$ are approximately equal (and are asymptotically equal as $n \to \infty$).

Again, we can make confidence regions, and simultaneous confidence intervals, just like the normal theory.

6.8 A 2-sample test

The *(Mahalanobis) distance* between two populations with means μ_1 and μ_2, and common variance Σ is defined as Δ, where

$$\Delta^2 = (\mu_1 - \mu_2)' \Sigma^{-1} (\mu_1 - \mu_2).$$

If we have samples of sizes n_1 and n_2, means $\overline{x}_1$ and $\overline{x}_2$, variances S_1 and S_2, then we define the *sample Mahalanobis distance* as D, where

$$D^2 = (\overline{x}_1 - \overline{x}_2)' S^{-1} (\overline{x}_1 - \overline{x}_2),$$

where

$$S = \frac{(n_1 - 1)S_1 + (n_2 - 1)S_2}{n - 2}$$

is the pooled variance, and $n = n_1 + n_2$.

For general μ_1 and μ_2, one can show that

$$T^2 = \frac{n_1 n_2}{n} \left((\overline{x_1} - \overline{x_2}) - (\mu_1 - \mu_2) \right)' S^{-1} \left((\overline{x_1} - \overline{x_2}) - (\mu_1 - \mu_2) \right)$$

is distributed as

$$\frac{(n-2)p}{n-p-1} F_{p,n-p-1}.$$

As a special case, when $\mu_1 = \mu_2$, we have $D^2 \sim \frac{n}{n_1 n_2} T^2(p, n-2)$.

So to test $\mu_1 = \mu_2$, we compute D^2 as above, and reject $H_0 : \mu_1 = \mu_2$ at significance level α if $\frac{n_1 n_2 (n-p-1)}{n(n-2)p} D^2 > F_{p,n-p-1}(\alpha)$.

These one- and two-sample tests are easy to compute in R by direct calculation using the matrix arithmetic facilities. The two-sample test can be calculated using the general MANOVA facilities (see section 7.16)

The library `ICSNP` contains a function `HotellingsT2()` which provides one- and two-sample tests.

But in general, it is not hard to carry out the analysis by hand. Here is a one-sample test:

Example 6.10 Recall the data on head length of first and second sons in 25 families from the further reading section on Canonical Correlation Analysis.

Let's do a 1-sample T^2-test on the hypothesis $H_0 : \mu_0 = (182, 182)$, i.e., both mean head lengths are 182. We need to evaluate $T^2 = n(\overline{x} - \mu_0)' S^{-1} (\overline{x} - \mu_0)$. This is easy to do by hand; the data give

$$\overline{x} = \begin{pmatrix} 185.72 \\ 183.84 \end{pmatrix}, \qquad S = \begin{pmatrix} 91.481 & 66.875 \\ 66.875 & 96.775 \end{pmatrix}.$$

So it is easy to compute $\overline{x} - \mu_0 = \begin{pmatrix} 3.72 \\ 1.84 \end{pmatrix}$, and as $n = 25$, we can simply compute the value of T^2 as approximately 4.17. Since $T^2(p, n) = \frac{np}{n-p+1} F_{p,n-p+1}$, we see that $T^2(p, n-1) = \frac{(n-1)p}{n-p} F_{p,n-p}$, so this gives an F-statistic of $F_{p,n-p} = \frac{25-2}{24\times 2} T^2 = 2.00$. However, $F_{2,23}(95\%) = 3.4$, so there is little evidence that the mean head lengths are not both equal to 182.

6.9 Worked examples

We will briefly give some examples of how to operate these more elementary tests.

We'll borrow (plagiarise!) some numbers from the excellent examples in the book of Chafield and Collins, chapter 7.

Chatfield and Collins make up some artificial data regarding 2-year old boys from a "high-altitude region in Asia", where MUAC means the *mid-upper-arm circumference.* Measurements are in centimetres.

Individual	Height	Chest circumference	MUAC
1	78	60.6	16.5
2	76	58.1	12.5
3	92	63.2	14.5
4	81	59.0	14.0
5	81	60.8	15.5
6	84	59.5	14.0

Example 6.11 Lowland children of the same age have expected measurements equal to 90cm, 58cm and 16cm respectively. Let's test the hypothesis that the highland children have the same means.

We consider $H_0 : \mu = \mu_0$, where $\mu_0 = \begin{pmatrix} 90 \\ 58 \\ 16 \end{pmatrix}$. We easily compute $\overline{x} = \begin{pmatrix} 82.0 \\ 60.2 \\ 14.5 \end{pmatrix}$, so that $\overline{x} - \mu_0 = \begin{pmatrix} -8.0 \\ 2.2 \\ -1.5 \end{pmatrix}$. We can compute (using R, for example, or by hand, if you are brave)

$$S = \begin{pmatrix} 31.600 & 8.040 & 0.500 \\ 8.040 & 3.172 & 1.310 \\ 0.500 & 1.310 & 1.900 \end{pmatrix},$$

and

$$S^{-1} = \begin{pmatrix} 0.186 & -0.632 & 0.387 \\ -0.632 & 2.584 & -1.615 \\ 0.387 & -1.615 & 1.538 \end{pmatrix},$$

and we can compute the T^2-statistic as 420.445. We need to compare $\frac{n-p}{np-p}T^2 = 84.09$ with $F_{3,3}$; indeed, R gives a p-value (`1-pf(84.09,3,3)`) of slightly over 0.002, so there is very strong evidence against the null hypothesis.

Example 6.12 Let's now study the confidence intervals of each component individually at the 95% level.

The formula in Theorem 6.9 involves various constants, and we can evaluate those first:

$$K = \sqrt{\frac{np-p}{n(n-p)} F_{3,3}(0.95)} = 2.780;$$

then using the result with $a = (1, 0, 0)'$, we see that

$$\mu_1 \in 82.0 \pm K\sqrt{31.600} = 82.0 \pm 15.6.$$

In the same way,

$$\mu_2 \in 60.2 \pm K\sqrt{3.172} = 60.2 \pm 5.0,$$
$$\mu_3 \in 14.5 \pm K\sqrt{1.900} = 14.5 \pm 3.8.$$

None of these intervals excludes the original null hypothesis values, although the multivariate test was even significant at the 99% level.

Just using the data as given, and testing each variable as a univariate test, we get different results:

$$\mu_1 \in 82.0 \pm 6.6,$$
$$\mu_2 \in 60.2 \pm 1.9,$$
$$\mu_3 \in 14.5 \pm 1.6.$$

The first two intervals here are significant, but the last is not.

Exercise 6.13 If we were testing the hypothesis that $\mu = \mu_0 = (88.0, 58.4, 16.0)'$, show that the values are significant for the multivariate test with p-value around 0.004, but not significant for the univariate tests, using either Theorem 6.9 or the t-tests above.

Example 6.14 Let's test that μ_0 is proportional to $(12, 8, 2)$, i.e., that $\mu_{0,1}/6 = \mu_{0,2}/4 = \mu_{0,3}$.

We can test this using matrix methods; let $C = \begin{pmatrix} 2 & 1 \\ -3 & 0 \\ 0 & -6 \end{pmatrix}$, so that $C' = \begin{pmatrix} 2 & -3 & 0 \\ 1 & 0 & -6 \end{pmatrix}$. Then our test can be reformulated as $H_0 : C'\mu_0 = 0$ (check!).

We know the distribution of $C'\mu_0$ (assuming normality etc.), and this reduces (in this case) to a bivariate normal distribution test.

The reader should verify (exercise!) that the p-value here is about 0.008.

To demonstrate the 2-sample procedures, Chatfield and Collins make up some more data, about 2-year old girls in the same region:

Individual	Height	Chest circumference	MUAC
1	80	58.4	14.0
2	75	59.2	15.0
3	78	60.3	15.0
4	75	57.6	13.0
5	79	59.5	14.0
6	78	58.1	14.5
7	75	58.0	12.5
8	64	55.5	11.0
9	80	59.2	12.5

Example 6.15 We might want to test the hypothesis that there is no difference in the mean vectors for 2-year old boys and girls.

So we compute $n_2 = 9$, $\overline{x}_2 = \begin{pmatrix} 76.0 \\ 58.4 \\ 13.5 \end{pmatrix}$ and $S_2 = \begin{pmatrix} 24.500 & 5.638 & 4.313 \\ 5.638 & 1.970 & 1.456 \\ 4.313 & 1.456 & 1.813 \end{pmatrix}$.

We can compute the overall variance matrix $S = \begin{pmatrix} 27.231 & 6.562 & 2.846 \\ 6.562 & 2.432 & 1.400 \\ 2.846 & 1.400 & 1.846 \end{pmatrix}$ on 13 degrees of freedom. We can compute

$$S^{-1}(\overline{x}_1 - \overline{x}_2) = \begin{pmatrix} 0.1278 \\ 0.3493 \\ 0.0797 \end{pmatrix},$$

then the value of the T^2-statistic $\mathcal{T}^2 = 5.3117$, and then compare $\frac{13-3+1}{13\times 3}\mathcal{T}^2 = 1.50$ with $F_{3,11}(0.05) = 3.59$, and see that the difference in means is not significant, so we cannot reject the null hypothesis.

Exercise 6.16 In this Example, verify that the difference in mean height is significant at the 5% level (just). Similarly, verify that the difference in mean chest measurement is also significant at this level.

If you would like to test your understanding of this section, have a go at a couple of the questions from the Exercises to Chapter 7 in Chatfield and Collins:

1. Test the hypothesis that the population of girls has $\mu = (80, 60, 15)'$.
2. Test the hypothesis that the population of girls has $\mu_1/5 = \mu_2/4 = \mu_3$.

6.10 Statistical discriminant analysis

We've already discussed the problem of discrimination between groups, but now we will assume that each group has the extra property of being distributed with a multivariate normal distribution. This extra information allows us to make more precise some of the informal conclusions from section 5.

As above, we suppose that we collect data on "objects" which can be classified into one or other of k known categories ($k \geq 2$). For example, "objects" could be patients, and the $k = 3$ categories are Rheumatoid Arthritis, Psorathic Arthritis and Psoriasis. Or "objects" could be iris flowers, and the $k = 3$ categories could be setosa, versicolor and virginica.

Suppose further that we measure p variates on each "object" to obtain a p-dimensional datum x, so $x \in \mathbb{R}^p$.

A *discriminant rule* d is a partition of $\mathbb{R}^p$ into k regions $R_1, \ldots, R_k$ (so every point of $\mathbb{R}^p$ lies in precisely one of the regions) such that if $x \in R_j$, then we classify x as in category j.

This is a formal way of saying that a discriminant rule d is a way of deciding unambiguously which category an object x belongs to, just on the basis of the measurements x.

Statistical discriminant analysis is concerned with

- regarding the measurements x as observations of some random variable whose distribution depends upon which category x belongs to, i.e., we suppose that the distribution of measurements in a given category arises from some given density rule.
- how can we construct a discriminant rule from data $x_1, \ldots, x_n$ known to belong to specified categories.
- what the (statistical) properties of the rule are, and in particular, how we can compare two rules d_1 and d_2.

We may know the precise density rules f_j (e.g., multivariate normal with precisely defined parameters), or the form of the density rules without knowing all the parameters (e.g., we know that the groups have a multivariate normal distribution, but we don't know the means or variances), or we may not know anything. In the last case, we fall back on the informal analysis above. We will consider some examples of the first case (unlikely in practice!), but the second case can be dealt with similarly, using the maximum likelihood estimators for the parameters, which we will consider in the next chapter.

We first consider some rules for discrimination:

6.10.1 The maximum likelihood discriminant rule

The maximum likelihood discriminant rule is to allocate x to that category which gives the greatest likelihood to x.

Note that technically we should only consider likelihoods of *parameters* for given data, not of data themselves. Here we are effectively considering the index of the categories as the parameter and so are considering the likelihood of j for data x.

Example 6.17 Two one-dimensional normal populations. Here $p = 1$, $k = 2$ and $x \sim N(\mu_i, \sigma_i^2)$ if x is in category i. We allocate x to category 1 if $f_1(x) > f_2(x)$, i.e., if

$$\frac{\sigma_2}{\sigma_1} \exp\left(-\frac{1}{2}\left(\frac{x-\mu_1}{\sigma_1}\right)^2 + \frac{1}{2}\left(\frac{x-\mu_2}{\sigma_2}\right)^2 \right) > 1,$$

i.e., if

$$Q(x) = x^2\left(\frac{1}{\sigma_1^2} - \frac{1}{\sigma_2^2}\right) - 2x\left(\frac{\mu_1}{\sigma_1^2} - \frac{\mu_2}{\sigma_2^2}\right) + \left(\left(\frac{\mu_1^2}{\sigma_1^2} - \frac{\mu_2^2}{\sigma_2^2}\right) - 2\log\left(\frac{\sigma_2}{\sigma_1}\right)\right) > 0.$$

Suppose that $\sigma_1 > \sigma_2$, and the coefficient of x^2 in $Q(x)$ is negative. Then $Q(x) < 0$ if x is sufficiently small or sufficiently big.

If $\sigma_1 = \sigma_2$, then $\log(\sigma_2/\sigma_1) = 0$, so the rule becomes allocate to category 1 if $|x - \mu_1| < |x - \mu_2|$.

Note in passing that the discrimination boundary is a quadratic surface in general, and is a linear space if the two variances coincide. This suggests that quadratic discriminant analysis might be a better tool for dealing with discrimination problems in the normal setting than linear discriminant analysis, unless you know that the variances are very similar. Let's see that something similar holds in more dimensions:

Example 6.18 p dimensions, k normal populations, means μ_i and common variance Σ. Then

$$f_i(x) = (2\pi)^{-p/2}|\Sigma|^{-1/2}\exp(-(x-\mu_i)'\Sigma^{-1}(x-\mu_i)/2),$$

which is maximized when $(x-\mu_i)'\Sigma^{-1}(x-\mu_i)$ is minimised, i.e., allocate x to the category whose mean has the smallest Mahalanobis distance from x.

Note that when $k = 2$, this reduces to the rule that we should allocate x to category 1 if

$$(\mu_1 - \mu_2)'\Sigma^{-1}(x - \mu) > 0,$$

where $\mu = (\mu_1 + \mu_2)/2$, i.e., the dividing point/line/plane/hyperplane between two populations with the same variances is linear (it is a hyperplane passing through μ, though not necessarily perpendicular to this line).

In the case where we don't know the parameters in advance, we would estimate a common variance (if this looked appropriate) with the pooled sample variance W, and the group means with the sample means. (We'll see later that the sample means are, unsurprisingly, the maximum likelihood estimators for the group means.) For example, with two groups, and two normal populations $N_p(\mu_i, \Sigma)$ (common variance), then estimate μ_1 and μ_2 by the sample means, and Σ by the pooled sample variance, and then the rule becomes to allocate x to category 1 if $(\overline{x}_1 - \overline{x}_2)'W^{-1}(x - (\overline{x}_1 + \overline{x}_2)/2) > 0$, where $W = \frac{1}{n-2}\sum_{i=1}^{2}(n_i - 1)S_i$, the pooled sample variance.

Example 6.19 (Iris Setosa and Versicolor) Taking just sepal length and width gives

$$\overline{x}_1 = \begin{pmatrix} 5.01 \\ 3.43 \end{pmatrix}, \ \overline{x}_2 = \begin{pmatrix} 5.94 \\ 2.77 \end{pmatrix}, \ W = \begin{pmatrix} 0.195 & 0.092 \\ 0.092 & 0.121 \end{pmatrix},$$

so $(\overline{x}_1 - \overline{x}_2)'W^{-1} = (-11.44, 14.14)$, so the rule is to allocate $x = (x_1\ x_2)'$ if

$$(-11.44\ 14.14)\begin{pmatrix} x_1 - \frac{1}{2}(5.01 + 5.94) \\ x_2 - \frac{1}{2}(3.43 + 2.77) \end{pmatrix} = -11.44x_1 + 14.14x_2 + 18.74 > 0.$$

6.10.2 Bayes Discriminant rule

In some circumstances, it is sensible to recognise that there are differing a priori probabilities of membership of categories. For example, in medical diagnosis some conditions may be very rare and others very common, although the symptoms (i.e., measurements available) may be very similar for the differing conditions (e.g., flu and polio). In these cases, it is reasonable to make some allowance for this, and shift the balance of classifying towards the more common category.

If the k categories have prior probabilities $\pi_1, \dots, \pi_k$, then the Bayes Discriminant Rule is to allocate x to that category for which $\pi_i f_i(x)$ is greatest. The Maximum Likelihood Rule is equivalent to a Bayes Discriminant Rule when the probabilities are equal.

6.10.3 The Likelihood Ratio Discriminant Rule

Let H_i: x and the n_i observations known to be from category i are all from category i, all others from known categories.

The rule is to allocate x to category j where $\ell_{\max}(H_j) = \max(\ell_{\max}(H_i))$. The distinction is that x is included in the ML estimation.

Example 6.20 (Two normal populations $N_p(\mu_i, \Sigma)$) n_i observations from category i with means $\overline{x}_i$ and variances S_i. Let H_1: x from population 1. Then under H_1 we have n_1+1 observations from population 1 which have mean $\frac{n_1\overline{x}_1+x}{n_1+1}$ and n_2 from population 2.

So under H_1 we have $\hat{\mu}_1 = \frac{n_1\overline{x}_1+x}{n_1+1}$ and $\hat{\mu}_2 = \overline{x}_2$, and the MLE of Σ is

$$\hat{\Sigma}_1 = \frac{1}{n_1+n_2+1}\left((n_1-1)S_1 + (n_2-1)S_2 + \frac{n_1}{n_1+1}(x-\overline{x}_1)(x-\overline{x}_1)'\right),$$

and under H_2 we have $\hat{\mu}_2 = \frac{n_2\overline{x}_2+x}{n_2+1}$ and $\hat{\mu}_1 = \overline{x}_1$, and the MLE of Σ is

$$\hat{\Sigma}_2 = \frac{1}{n_1+n_2+1}\left((n_1-1)S_1 + (n_2-1)S_2 + \frac{n_2}{n_2+1}(x-\overline{x}_2)(x-\overline{x}_2)'\right).$$

Now $\ell_{\max}(H_1) - \ell_{\max}(H_2) = \frac{1}{2}(n_1+n_2+1)(\log|\hat{\Sigma}_2| - \log|\hat{\Sigma}_1|)$, and

$$\begin{aligned}
|\hat{\Sigma}_i| &= \left|\frac{1}{n_1+n_2+1}\left((n_1-1)S_1 + (n_2-1)S_2 + \frac{n_i}{n_i+1}(x-\overline{x}_i)(x-\overline{x}_i)'\right)\right| \\
&= (n_1+n_2+1)^{-p}\left|T + \frac{n_i}{n_i+1}(x-\overline{x}_i)(x-\overline{x}_i)'\right| \\
&= (n_1+n_2+1)^{-p}\left|T\left(I + T^{-1}\frac{n_i}{n_i+1}(x-\overline{x}_i)(x-\overline{x}_i)'\right)\right| \\
&= (n_1+n_2+1)^{-p}|T|.\left|I + \frac{n_i}{n_i+1}(x-\overline{x}_i)'T^{-1}(x-\overline{x}_i)\right|
\end{aligned}$$

the final equality coming from the identity $|I+AB| = |I+BA|$ on Task Sheet 3, and where $T = (n_1-1)S_1 + (n_2-1)S_2$. So $\ell_{\max}(H_1) > \ell_{\max}(H_2)$ and x is allocated to population 1 if

$$\frac{n_2}{n_2+1}(x-\overline{x}_2)'T^{-1}(x-\overline{x}_2) > \frac{n_1}{n_1+1}(x-\overline{x}_1)'T^{-1}(x-\overline{x}_1);$$

if $n_1 = n_2$, then this is the same as the sample ML rule, and if n_1 and n_2 are both large, then it is almost so. However, if either sample size is small, and $n_1 \neq n_2$, then the allocation is different.

6.11 Fisher's linear discriminant function revisited

In §5, we showed that if the data are projected into one dimension, the projection which maximizes the ratio of the between to within groups sums of squares, $a'Ba/a'Wa$, is the (right) eigenvector of $W^{-1}B$ corresponding to its largest eigenvalue.

New observations, x, can be classified according to any of the criteria above. The most common practice is to calculate the discriminant score $a'x$ and allocate to the group j where $\min(|a'x - a'\overline{x}_i|) = a'x - a'\overline{x}_j$. In the case $k = 2$, B has rank 1, and $B = \frac{n_1n_2}{n}(\overline{x}_1 - \overline{x}_2)(\overline{x}_1 - \overline{x}_2)' = \frac{n_1n_2}{n}dd'$, say, and $W^{-1}B$ has only one non-zero eigenvalue which is $\frac{n_1n_2}{n}d'W^{-1}d$ and eigenvector $W^{-1}d$ and the rule is to allocate to category 1 if

$$(\overline{x}_1 - \overline{x}_2)'W^{-1}(x - (\overline{x}_1 + \overline{x}_2)/2) > 0,$$

i.e., the same as the sample discriminant rule.

The function $h(x) = a'x$ where a is the first eigenvector of $W^{-1}B$ is called *Fisher's Linear Discriminant Function.*

6.12 Probabilities of miscalculation

6.12.1 Examples

When we know the distribution of the groups, we can estimate the probabilities of misclassification.

Let $p_{ij} = P(\text{a type } j \text{ object is classified as type } i)$; the *performance* of a discriminant rule is described by the p_{ij}. Good rules have p_{ij} small for $i \neq j$ and big for $i = j$.

Example 6.21 (Two normal populations) Consider two normal populations $N_p(\mu_i, \Sigma)$. Let $\mu = \frac{1}{2}(\mu_1 + \mu_2)$. Then when x is in the first population, we have $\alpha'(x - \mu) \sim N(\frac{1}{2}\alpha'(\mu_1 - \mu_2), \alpha'\Sigma\alpha)$ for any vector α.

When we classify x on the value of the discriminant function $h(x) = \alpha'(x - \mu)$, with $\alpha = \Sigma^{-1}(\mu_1 - \mu_2)$ (classifying as population 1 if $h(x) > 0$), then $h(x) \sim N(\frac{1}{2}\Delta^2, \Delta^2)$, where $\Delta^2 = (\mu_1 - \mu_2)'\Sigma^{-1}(\mu_1 - \mu_2)$.

Similarly, if x is in population 2, then $h(x) \sim N(-\frac{1}{2}\Delta^2, \Delta^2)$.

So $p_{12} = P(h(x) > 0 | x \text{ is in population 2}) = \Phi = \Phi(\frac{-\Delta^2/2}{\sqrt{\Delta^2}}) = \Phi(-\frac{1}{2}\Delta)$, and similarly $p_{21} = \Phi(-\frac{1}{2}\Delta)$. Thus for two normal populations with a common variance, the misclassification probabilities are equal.

Example 6.22 (Correlation in the bivariate normal) Consider the case $p = 2$, with normal populations $N_2(0, \Sigma)$ and $N_2(\mu, \Sigma)$, where $\mu = (\mu_1 \; \mu_2)'$, and suppose $\mu_1, \mu_2 > 0$, and $\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \sigma^2$.

1. First, we examine how the correlation may affect the power of the discriminant function.

   We have$\Delta^2 = \mu'\Sigma^{-1}\mu = \frac{\mu_1^2 + \mu_2^2 - 2\rho\mu_1\mu_2}{\sigma^2(1-\rho^2)}$. If the variables were uncorrelated, then we would have $\Delta^2 = \frac{\mu_1^2 + \mu_2^2}{\sigma^2} = \Delta_0^2$.

Now the correlation will "improve" the discrimination (i.e., reduce p_{12} if $\Delta^2 > \Delta_0^2$, which rearranges to

$$\rho\left(\left(1+\left(\frac{\mu_2}{\mu_1}\right)^2\right)\rho - 2\left(\frac{\mu_2}{\mu_1}\right)\right) > 0,$$

i.e., if $\rho < 0$ or $\rho > \frac{2\mu_1\mu_2}{\mu_1^2+\mu_2^2}$. In particular, if $\mu_1 = \mu_2$, then any positive correlation reduces the power of discrimination.

2. Now we compare the rule with that based just on the first variable. So if the rule were based just on measurements of the first variable, then $p_{12} = \Phi(-\frac{1}{2}\Delta^*)$, where $\Delta^{*2} = \frac{\mu_1^2}{\sigma^2}$, and so using both variables instead of just one improves the discrimination if $\frac{\mu_1^2+\mu_2^2-2\rho\mu_1\mu_2}{1-\rho^2} > \mu_1^2$, which rearranges to $(\rho\mu_1 - \mu_2)^2 > 0$, which is always the case.

6.12.2 Misclassification probabilities under estimation

If parameters are estimated, then these can be used to estimate the p_{ij}. For example, in Example 6.21 we can obtain $\widehat{p}_{12} = \Phi(-\frac{1}{2}\hat{\Delta})$, where

$$\hat{\Delta}^2 = (\overline{x}_1 - \overline{x}_2)' S^{-1} (\overline{x}_1 - \overline{x}_2).$$

In Example 6.19 of discriminating between iris setosa and iris versicola, this gives $p_{12} = 0.013$. Generally such estimates tend to be over-optimistic – i.e., biased downwards.

Alternatively, given any discriminant rule $\{R_j\}$, we can work out the number of misclassifications as $n_{ij} = |\{x \text{ in population } j \text{ and } x \in R_i\}|$; then we can estimate p_{ij} by $\hat{p}_{ij} = \frac{n_{ij}}{\sum_i n_{ij}}$, the proportion from population j classified as i.

Again, this tends to be over-optimistic – and can be improved by using the techniques below.

Jack-knifing, or cross-validation

Here, the discriminant rule is calculated by leaving out each observation in turn, and then using that rule to classify the omitted observation. The overall correct classification rate is then a *jackknife estimate* of the true probability of correct classification.

Permutation and randomisation tests

To assess whether the observed correct classification rate is "significantly" higher than would be achieved by chance, it is possible to perform the complete discriminant analysis on the same observations but by permuting the labels of group membership and calculating the correct classification rate for this permuted set of labels. Observing where the rate obtained from the correct labels falls in this *permutation distribution* provides the *permutation test*: if it is in the upper tail of the distribution then there is evidence that the rate obtained is higher than would be achieved by chance.

Typically the number of permutations is too large to compute the complete permutation distribution, and so a reasonable number of random permutations are used to construct a *randomisation test*. Random permutations can be obtained by sampling with replacement from the label variable.

Further reading: Multivariate regression analysis

Now that we have developed the theory of the multivariate normal distribution, we can explain the generalisation of regression analysis to the multivariate case.

There is little new in terms of statistical concepts, and the univariate mathematical development carries through almost unchanged symbolically to the multivariate case; we will see that this generalisation actually requires no new mathematics. Formal statistical tests are not described here, but are essentially applications of multivariate analysis of variance considered later.

First, recall the univariate case, with just one independent variable X. The parameters α and β in the model $E[Y] = \alpha + \beta X$, or $y_i = \alpha + \beta x_i + \epsilon_i$, where $\epsilon_i \sim N(0, \sigma^2)$ are i.i.d., have least squares estimates (and maximum likelihood estimates) obtained by minimising $\sum_{i=1}^{n}(y_i - \hat{\alpha} - \hat{\beta}x_i)^2$ which yields

$$\hat{\alpha} = \overline{y} - \hat{\beta}\overline{x}$$
$$\hat{\beta} = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sum(x_k - \overline{x})^2}$$

Note that the sample correlation coefficient is slightly different,

$$\rho_{XY} = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum(x_i - \overline{x})^2 \sum(y_i - \overline{y})^2}}.$$

Note also that the correlation coefficient is symmetric in X and Y, but the regression coefficient is not, emphasizing that the correlation coefficient is used for investigating the relationship between the X and Y variables, whereas regression analysis investigates the dependence of the dependent variable Y on the independent variable X (perhaps with the aim of predicting one from the other).

If there are several independent variables, $X_1, \ldots, X_q$, then the appropriate model can be written as $E[Y] = X\beta$ or $Y = X\beta + \epsilon$, where $\epsilon \sim N_n(0, \sigma^2 I_n)$, and $\beta' = (\beta_1, \ldots, \beta_q)$, and X is the $n \times q$ matrix of observations of the q independent variables (and usually the first X_1 would be taken to be the unit vector $\mathbf{1}_n$, so the model includes a constant term). Note here the use of X as the $n \times q$ matrix of observations, rather than X', as in other chapters, so as to conform with general usage in presentation of regression.

Minimising the sum of squares of the residuals, i.e., $\Omega = (Y - X\beta)'(Y - X\beta)$, yields after differentiating Ω with respect to β:

$$2X'(Y - X\beta) = 0,$$

and thus $\hat{\beta} = (X'X)^{-1}X'Y$ (provided $n > p$, or more precisely that $X'X$ is nonsingular). The extension to the case when Y is a matrix of vector observations is now straightforward.

The p-dimensional multivariate linear model is $E[Y] = X\beta$ or $Y = X\beta + \epsilon$, where Y is an $n \times p$ matrix of n observations of p-dimensional variables, X is $n \times q$, β is $q \times p$ and ϵ is an $n \times p$ matrix random variable. The matrix of residual sums of squares and cross products is $\Omega = (Y - X\beta)'(Y - X\beta)$; differentiating Ω with respect to β yields $2X'(Y - X\beta) = 0$, and thus $\hat{\beta} = (X'X)^{-1}X'Y$ (provided $n > p$, or more precisely that $X'X$ is nonsingular). Note that Ω is a matrix, so its differentiation may need to be taken on trust, as also the fact that this minimises both the determinant and the trace of Ω. This means that the p individual $\hat{\beta}_i$ in $\hat{\beta}$ are identical to those which would be obtained by separate multiple regressions of each Y_i on the independent variables $X_1, \dots, X_q$. This is true of least squares estimation and of maximum likelihood estimation if the errors are assumed to be normal with independence from one observation to the next, though they could be correlated between one variable to the next on the same observation, i.e., that the rows of ϵ are i.i.d $N_p(0, \Sigma)$. Predicted values of Y are given by $\hat{Y} = X\hat{\beta}$ and Σ is estimated as $\hat{\Sigma} = (Y - X\hat{\beta})'(Y - X\hat{\beta})/(n - q - 1)$.

Example 6.23 (Cox) This data concerns the price and consumption of pork and beef, and is at `http://lib.stat.cmu.edu/DASL/Datafiles/agecondat.html`:

YEAR	PBE	CBE	PPO	CPO	PFO	DINC	CFO	RDINC	RFP
1925	59.7	58.6	60.5	65.8	65.8	51.4	90.9	68.5	877
1926	59.7	59.4	63.3	63.3	68.0	52.6	92.1	69.6	899
1927	63.0	53.7	59.9	66.8	65.5	52.1	90.9	70.2	883
1928	71.0	48.1	56.3	69.9	64.8	52.7	90.9	71.9	884
1929	71.0	49.0	55.0	68.7	65.6	55.1	91.1	75.2	895
1930	74.2	48.2	59.6	66.1	62.4	48.8	90.7	68.3	874
1931	72.1	47.9	57.0	67.4	51.4	41.5	90.0	64.0	791
1932	79.0	46.0	49.5	69.7	42.8	31.4	87.8	53.9	733
1933	73.1	50.8	47.3	68.7	41.6	29.4	88.0	53.2	752
1934	70.2	55.2	56.6	62.2	46.4	33.2	89.1	58.0	811
1935	82.2	52.2	73.9	47.7	49.7	37.0	87.3	63.2	847
1936	68.4	57.3	64.4	54.4	50.1	41.8	90.5	70.5	845
1937	73.0	54.4	62.2	55.0	52.1	44.5	90.4	72.5	849
1938	70.2	53.6	59.9	57.4	48.4	40.8	90.6	67.8	803
1939	67.8	53.9	51.0	63.9	47.1	43.5	93.8	73.2	793
1940	63.4	54.2	41.5	72.4	47.8	46.5	95.5	77.6	798
1941	56.0	60.0	43.9	67.4	52.2	56.3	97.5	89.5	830

where

- `PBE` is the price of beef (cents/lb)
- `CBE` is the consumption of beef per capita (lbs)
- `PPO` is the price of pork (cents/lb)
- `CPO` is the consumption of pork per capita (lbs)
- `PFO` is the retail food price index $(1947 - 1949 = 100)$
- `DINC` is the disposable income per capita index $(1947 - 1949 = 100)$
- `CFO` is the food consumption per capita index $(1947 - 1949 = 100)$
- `RDINC` is the index of real disposable income per capita $(1947 - 1949 = 100)$
- `RFP` is the retail food price index adjusted by the CPI $(1947 - 1949 = 100)$

For this illustration, we will consider the dependence of consumption of beef and pork, i.e., $Y = [\texttt{CBE}, \texttt{CPO}]$ upon the prices of beef and pork and disposable income, and include a constant term in the regression, so $X = [1_{17}, \texttt{PBE}, \texttt{PPO}, \texttt{DINC}]$.

Thus $p = 2$, $q = 4$ and $n = 17$.

We find:

$$Y = \begin{pmatrix} 58.6 & 65.8 \\ 59.4 & 63.3 \\ 53.7 & 66.8 \\ 48.1 & 69.9 \\ 49.0 & 68.7 \\ 48.2 & 66.1 \\ 47.9 & 67.4 \\ 46.0 & 69.7 \\ 50.8 & 68.7 \\ 55.2 & 62.2 \\ 52.2 & 47.7 \\ 57.3 & 54.4 \\ 54.4 & 55.0 \\ 53.6 & 57.4 \\ 53.9 & 63.9 \\ 54.2 & 72.4 \\ 60.0 & 67.4 \end{pmatrix}, \quad X = \begin{pmatrix} 1 & 59.7 & 60.5 & 51.4 \\ 1 & 59.7 & 63.3 & 52.6 \\ 1 & 63.0 & 59.9 & 52.1 \\ 1 & 71.0 & 56.3 & 52.7 \\ 1 & 71.0 & 55.0 & 55.1 \\ 1 & 74.2 & 59.6 & 48.8 \\ 1 & 72.1 & 57.0 & 41.5 \\ 1 & 79.0 & 49.5 & 31.4 \\ 1 & 73.1 & 47.3 & 29.4 \\ 1 & 70.2 & 56.6 & 33.2 \\ 1 & 82.2 & 73.9 & 37.0 \\ 1 & 68.4 & 64.4 & 41.8 \\ 1 & 73.0 & 62.2 & 44.5 \\ 1 & 70.2 & 59.9 & 40.8 \\ 1 & 67.8 & 51.9 & 43.5 \\ 1 & 63.4 & 41.5 & 46.5 \\ 1 & 56.0 & 43.9 & 56.3 \end{pmatrix}$$

The least squares estimate of β is

$$\hat{\beta} = \begin{pmatrix} 101.40 & 79.60 \\ -0.753 & 0.153 \\ 0.254 & -0.687 \\ -0.241 & 0.283 \end{pmatrix}.$$

You should check these calculations, and also check that the same answers would be obtained with separate univariate linear regressions.

The advantage of the multivariate approach is that the correlation between pork and beef consumption is modelled, and this allows construction of simultaneous confidence intervals.

Σ is estimated as $\hat{\Sigma} = \begin{pmatrix} 4.412 & -7.572 \\ -7.572 & 16.835 \end{pmatrix}$.

- Multivariate regression models the dependence of a p-dimensional random variable Y on a q-dimensional variable X.
- The individual equations relating the p individual Y variables on the X variables are identical to those obtained by separate univariate regressions of the Y_i on the X variables.
- The advantage of the multivariate approach is that the correlation structure of the dependent variables is modelled, and this allows construction of simultaneous confidence intervals.
- Formal tests of hypothesis (e.g., whether $\beta = 0$) are available.
- It does not matter whether $q < p$ or $q > p$.
- It is required that $n > \max(p, q)$ and that $X'X$ is non-singular.
- In R, type `lm(cbind(y1,y2)~x1+x2+x3)`, say, to test the dependency of two variables `y1` and `y2` on `x1`, `x2` and `x3`.

Further reading: Partial Least Squares

The methods of linear discriminant analysis (LDA), multivariate regression analysis and canonical correlation analysis (CCA) considered above all required more observations than variables measured. This contrasts with the exploratory technique of PCA where there is no such restrction; if there are fewer obvservations than variables, then the last few eigenvalues are not informative.

In the contexts of the development of these methods, and appreciating the computations restrctions of the time in the mid 20th century, the "$n > p$" requirement was generally not restrictive; the data sets considered were small, and typically n might be a few hundred, and p might be around 10. It was generally easy to obtain more observations (interview a few more people, or measure a few more iris flowers) or else drop a few variables from consideration using expert knowledge (e.g., that certain variables are more or less useful than others to the purpose at hand).

However, in the last 30 years or so, this situation has changed, and it is now very common to encounter data sets where p is significantly larger than n, and it is not easy to reduce the number of variables or measure more objects. For example, in measuring gene expression levels, it is routine to measure thousands of variables simultaneously on relatively few subjects. Measurement of each variable (i.e., each gene) may be cheap, but each high-dimensional observation is expensive. If 13000 genes are measured on 100 samples, it is not realistic to reduce the number of variables to fewer than 100, especially if the objective of the study is to discover which genes are the most important, and there is no existent expert knowledge in the area (hence the reason for the study), nor is it realistic to obtain more samples if each costs tens of pounds.

However, the underlying questions of what is the relationship between some dependent variable and the set of independent variables and how can we discriminate between groups of subjects are still of interest, but if $n < p$ then methods of multivariate regression and LDA are not available. Consequently, there is a need for techniques which address these problems directly, and which are immune to the singularity of the matrices required in regression and discriminant analysis. This section considers such methods, which go under the general name of *partial least squares*.

In passing, an obvious approach might be to reduce the dimensionality with PCA as a preliminary step. That is, if $p > n$, then instead of attempting to use all X_1, X_2, $\dots$, X_p in the regression analysis (noting that this will fail if $p > n$ as $X'X$ is singular), one could regress the dependent variable Y on the first k principal components of the X_i, where $k < n < p$. This approach is particularly useful if $X'X$ is singular because of the inherent multicollinearities in the X_i rather than because there are too few observations. In such cases, the first few principal components corresponding to the nonzero eigenvalues genuinely contain all the available information. In more general cases, however, the drawback is that the principal components are not derived with any regard to the dependence of the Y variables on the X_i. Similar drawbacks are present in dimensionality reduction by PCA as a preliminary to LDA, though in both situations it is a useful technique to try since it is quick and easy and the methods of partial least squares need a little more effort, at least currently, and especially in terms of interpretation.

Partial least squares is a dimensionality reduction method with components chosen with the response variable kept in mind, where the response variable may be a continuous dependent variable (possibly multivariate, "*PLS regression*") or a group indicator ("*PLS classification*"). Essentially, PLS obtains linear components from the high-dimensional data which maximise covariance with the response variable. Unlike correlations, covariances require no inversion of matrices and so avoid problems of singularity of matrices calculated when $n < p$.

Facilities for implementation of PLS are increasingly widely available. In R, there are seeral contributed packages and these are well-documented with examples and references.

Brief accounts of PLS are given in Cox (2005), p.190. Some readable references with applications are given in Boulesteix, A.-L. (2004) "PLS dimension reduction for classification of microarray data", Statistical Applications ion Genetics and Molecular Biology 3, and Boulesteix, A.-L. and Strimmer, K. (2006) "Partial Least Squares: a versatile tool for the analysis of high-dimensional genomic data", Briefings in Bioinformatics.

Boulesteix is also the lead author of the R package `plsgenomics`, which is available from the R website, which provides routines `pls.regression()` and `pls.lda()`.

7 Further statistical tests

7.1 Likelihood ratio tests

The one- and two-sample test statistics for $\mu = \mu_0$ and $\mu_1 = \mu_2$ given above are easily shown to be likelihood ratio statistics (i.e., "optimal"). Likelihood Ratio Tests are a useful general procedure for constructing tests and can often be implemented numerically using general purpose function maximization routines even when analytic closed forms for maximum likelihood estimates are not obtainable.

Suppose data are available from a distribution depending on a parameter θ, where θ may be a vector parameter, i.e., consist of several separate parameters. For example, $\theta = (\mu, \sigma)$, the parameters of a univariate normal distribution with 2 separate parameters, or $\theta = (\mu, \Sigma)$, the parameters of a p-dimensional normal distribution, with $p+\frac{1}{2}p(p+1) = \frac{1}{2}p(p+3)$ separate parameters (μ is a p-vector and Σ is a *symmetric* $p\times p$-matrix). Suppose that Θ denotes the set of all possible parameters. For each possible parameter $\theta \in \Theta$, the *likelihood* $L(\theta)$ is defined by evaluating the joint density of the variables at the observed values with these parameters.

Typically, the null hypothesis H_0 will specify the values of some of these, e.g., in the second case, $H_0:\ \mu = 0$ specifies p parameters. The null hypothesis restricts the possible parameters to a subset $\Theta_0 \subseteq \Theta$.

A likelihood ratio test of $H_0:\ \theta \in \Theta_0$ rejects H_0 if

$$\Lambda = \frac{\max_{\theta\in\Theta_0} L(\theta)}{\max_{\theta\in\Theta} L(\theta)} < c,$$

where c is suitably chosen. Thus we reject H_0 if the maximum likelihood when the parameters are in Θ_0 is much smaller than the maximum likelihood when the parameters are allowed to range over the larger set Θ. In order to specify c, we need to know more precisely the distribution of the left-hand side. However, when the sample size is large (and certain other conditions hold), the sampling distribution of $-2\ln\Lambda$ is well approximated by a χ^2-distribution.

Thus, the general procedure for constructing a likelihood ratio test (i.e., finding the LRT statistic) of H_0 verses H_A is:

1. Find the maximum likelihood estimates of all parameters θ, assuming H_0 is true, to get $\hat\theta_0$. For example, with $N_p(\mu, \Sigma)$, if $H_0:\ \mu = 0$, then estimate Σ assuming $\mu = 0$, giving $\hat\Sigma = \frac{1}{n}\sum_i x_i x_i'$, and then $\hat\theta_0 = (0, \frac{1}{n}\sum_i x_i x_i')$.

2. Find the maximum value of the log likelihood under H_0 (i.e., substitute the MLEs of the parameters under H_0 into the log likelihood function) to get $\ell_{\max}(H_0) = \ell(\hat\theta_0)$.

3. Find the maximum likelihood estimates of all parameters assuming H_A is true, $\hat\theta_A$. Typically, these will be the ordinary MLEs.

4. Find the maximum value of the log likelihood under H_A (i.e., substitute the MLEs of the parameters under H_A into the log likelihood function) to get $\ell_{\max}(H_A) = \ell(\hat\theta_A)$.

5. Calculate twice the difference in maximized log likelihoods:

$$\lambda = 2(\ell_{\max}(H_A) - \ell_{\max}(H_0)).$$

6. Use Wilks's Theorem, which says that under H_0 this statistic is approximately distributed as χ^2 with degrees of freedom given by the difference in the numbers of estimated parameters under H_0 and H_A, i.e., $\lambda \sim \chi^2_k$, where $k = \dim(H_A) - \dim(H_0)$, or

7.2 Log likelihood for the multivariate normal distribution

Suppose $x_1, \ldots, x_n$ are independent observations of $x \sim N_p(\mu, \Sigma)$. Then

$$Lik(\mu, \Sigma; X) = \frac{1}{(2\pi)^{np/2}|\Sigma|^{n/2}} \exp\left(-\tfrac{1}{2}\sum_{i=1}^{n}(x_i - \mu)'\Sigma^{-1}(x_i - \mu)\right),$$

so $\ell(\mu, \Sigma; X) = \ln(Lik(\mu, \Sigma; X))$ is equal to

$$\begin{aligned}\ell(\mu, \Sigma; X) &= -\frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)'\Sigma^{-1}(x_i - \mu) - \tfrac{1}{2}np\log(2\pi) - \tfrac{1}{2}n\log|\Sigma| \\ &= -\frac{1}{2}\left[\sum_{i=1}^{n}(x_i - \overline{x})'\Sigma^{-1}(x_i - \overline{x}) + n(\overline{x} - \mu)'\Sigma^{-1}(\overline{x} - \mu)\right] - \tfrac{1}{2}np\log(2\pi) - \tfrac{1}{2}n\log|\Sigma|\end{aligned}$$

So $\frac{\partial \ell}{\partial \mu} = n\Sigma^{-1}(\overline{x} - \mu)$, and thus $\hat{\mu} = \overline{x}$.

Further, if we set $T = \Sigma^{-1}$, it can be shown that

$$\frac{\partial \ell}{\partial T} = [n\Sigma - (n-1)S - n(\overline{x} - \mu)(\overline{x} - \mu)'] - \tfrac{1}{2}\text{diag}\,[n\Sigma - (n-1)S - n(\overline{x} - \mu)(\overline{x} - \mu)']$$

where the derivative of the scalar ℓ with respect to the $p \times p$ matrix T is the $p \times p$ matrix formed by the partial derivatives of ℓ with respect to each of the elements t_{ij} of T and where diag(A) is the diagonal matrix formed by just the diagonal elements of the $p \times p$ matrix A, zeroes elsewhere.

So if also $\frac{\partial \ell}{\partial T} = 0$, then $\Sigma = \hat{\Sigma} = \frac{n-1}{n}S + (\overline{x} - \hat{\mu})(\overline{x} - \hat{\mu})'$ and when $\hat{\mu} = \overline{x}$ this gives the (unrestricted) maximum likelihood estimates of μ and Σ as

$$\hat{\mu} = \overline{x}, \qquad \hat{\Sigma} = \frac{n-1}{n}S.$$

More generally, whatever the MLE of μ, then

$$\hat{\Sigma} = \frac{n-1}{n}S + (\overline{x} - \hat{\mu})(\overline{x} - \hat{\mu})'.$$

This form is sometimes useful in constructing likelihood ratio tests of hypotheses that put some restriction on μ and so under the null hypothesis the maximum likelihood estimate of μ is not $\overline{x}$. In these cases, we can easily obtain the MLE of Σ and thus the value of the maximum likelihood under the null hypothesis.

For the construction of likelihood ratio tests, we need the actual form of the maximised likelihood under null and alternative hypotheses. Typically, the alternative hypothesis gives no restrictions on μ and Σ, and so the MLEs under the alternative hypothesis are the usual ones, i.e., $\hat{\mu} = \overline{x}$, $\hat{\Sigma} = \frac{n-1}{n}S$. The null hypothesis will either impose some constraint on Σ (e.g., H_0: $\Sigma = \Sigma_0$) or some constraint on μ (e.g., H_0: $\mu = \mu_0$ or H_0: $\mu\mu' = 1$). In these cases we obtain the estimate of μ and then use the more general form above.

For example, under H_0: $\mu = \mu_0$, we have $\hat{\mu} = \mu_0$, and so this gives

$$\hat{\Sigma} = \frac{n-1}{n} S + (\overline{x} - \mu_0)(\overline{x} - \mu_0)' = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_0)(x_i - \mu_0)'.$$

Recall that

$$\ell(\mu, \Sigma; X) = -\frac{1}{2}\left[\sum_{i=1}^{n}(x_i - \overline{x})'\Sigma^{-1}(x_i - \overline{x}) + n(\overline{x} - \mu)'\Sigma^{-1}(\overline{x} - \mu)\right] - \tfrac{1}{2}np\log(2\pi) - \tfrac{1}{2}n\log|\Sigma|$$

and observe that:

$$\begin{aligned}
\sum_{i=1}^{n}(x_i - \overline{x})'\Sigma^{-1}(x_i - \overline{x}) &= \mathrm{tr}\left(\sum_{i=1}^{n}(x_i - \overline{x})'\Sigma^{-1}(x_i - \overline{x})\right) \\
&= \sum_{i=1}^{n}\mathrm{tr}\left((x_i - \overline{x})'\left[\Sigma^{-1}(x_i - \overline{x})\right]\right) \\
&= \sum_{i=1}^{n}\mathrm{tr}\left(\left[\Sigma^{-1}(x_i - \overline{x})\right](x_i - \overline{x})'\right) \\
&= \mathrm{tr}\left(\Sigma^{-1}\sum_{i=1}^{n}(x_i - \overline{x})(x_i - \overline{x})'\right) \\
&= \mathrm{tr}\left(\Sigma^{-1}(n-1)S\right) \\
&= (n-1)\mathrm{tr}\left(\Sigma^{-1}S\right).
\end{aligned}$$

So

$$\boxed{\ell(\mu, \Sigma; X) = -\tfrac{1}{2}(n-1)\mathrm{tr}(\Sigma^{-1}S) - \tfrac{1}{2}n\mathrm{tr}(\Sigma^{-1}(\overline{x} - \mu)(\overline{x} - \mu)') - \tfrac{1}{2}np\log(2\pi) - \tfrac{1}{2}n\log|\Sigma|}$$

and so its maximum value, $\ell(\hat{\mu}, \hat{\Sigma}; X)$, is

$$-\tfrac{1}{2}(n-1)\mathrm{tr}(\hat{\Sigma}^{-1}S) - 0 - \tfrac{1}{2}np\log(2\pi) - \tfrac{1}{2}n\log|\hat{\Sigma}|$$

and this equals

$$-\tfrac{1}{2}(n-1)\mathrm{tr}(nS^{-1}S/(n-1)) - \tfrac{1}{2}np\log(2\pi) - \tfrac{1}{2}n\log|(n-1)S/n|$$

or

$$-\tfrac{1}{2}np - \tfrac{1}{2}np\log((n-1)/n) - \tfrac{1}{2}n\log|S| - \tfrac{1}{2}np\log(2\pi).$$

We'll now give a few examples of Likelihood Ratio Tests, using these calculations.

7.3 LRT of $H_0: \ \mu = \mu_0$ against $H_A: \ \mu \neq \mu_0$ with $\Sigma = \Sigma_0$ known

Suppose that $x_1, \ldots, x_n$ are independent observations of $x \sim N_p(\mu, \Sigma_0)$.

To test $H_0: \ \mu = \mu_0$ against $H_A: \ \mu \neq \mu_0$, with Σ_0 known

Now

$$\begin{aligned}\ell(\mu;X) &= \log \text{lik}(\mu;X)\\ &= -\tfrac{1}{2}np\log(2\pi) - \tfrac{1}{2}n\log|\Sigma_0| - \tfrac{1}{2}(n-1)\text{tr}(\Sigma_0^{-1}S) - \tfrac{1}{2}n(\overline{x}-\mu)'\Sigma_0^{-1}(\overline{x}-\mu)\\ &= K - \tfrac{1}{2}n(\overline{x}-\mu)'\Sigma_0^{-1}(\overline{x}-\mu).\end{aligned}$$

So under H_0 we have $\ell(\mu_0;X,H_0) = K - \frac{1}{2}n(\overline{x}-\mu_0)'\Sigma_0^{-1}(\overline{x}-\mu_0)$, i.e.,

$$\ell_{\max}(H_0) = K - \tfrac{1}{2}n(\overline{x}-\mu_0)'\Sigma_0^{-1}(\overline{x}-\mu_0).$$

Under H_A, the MLE of μ is $\overline{x}$, giving $\ell_{\max}(H_A) = K$.

So the LRT statistic is

$$\lambda = 2(\ell_{\max}(H_A) - \ell_{\max}(H_0)) = n(\overline{x}-\mu_0)'\Sigma_0^{-1}(\overline{x}-\mu_0),$$

and the test is to reject this if it is improbably large when compared with χ^2_p, noting that there are p parameters to be estimated under H_A, but none under H_0. This is the same test we found in the previous chapter.

Also, note that this is an exact result (i.e., not a Wilks's Theorem approximation), since $\lambda = yy' = \sum y_i^2$ with $y_i \sim N(0,1)$, where $y = n^{1/2}\Sigma_0^{-1/2}(\overline{x}-\mu_0) \sim N_p(0, I_p)$.

7.4 LRT of $H_0:\ \Sigma = \Sigma_0$ against $H_A:\ \Sigma \neq \Sigma_0$ with μ unknown

Suppose that $x_1, \ldots, x_n$ are independent observations of $x \sim N_p(\mu, \Sigma)$.

To test $H_0:\ \Sigma = \Sigma_0$ against $H_A:\ \Sigma \neq \Sigma_0$.

Under H_0, we have $\hat{\mu} = \overline{x}$ and $\Sigma = \Sigma_0$.

Under H_A, we have $\hat{\mu} = \overline{x}$ and $\hat{\Sigma} = \frac{n-1}{n}S = S^*$, say.

Thus $\ell_{\max}(H_0) = -\frac{1}{2}n\text{tr}(\Sigma_0^{-1}S^*) - \frac{1}{2}np\log(2\pi) - \frac{1}{2}n\log|\Sigma_0|$, and $\ell_{\max}(H_A) = -\frac{1}{2}np - \frac{1}{2}np\log(2\pi) - \frac{1}{2}n\log|S^*|$.

So

$$\lambda = 2(\ell_{\max}(H_A) - \ell_{\max}(H_0)) = n\text{tr}(\Sigma_0^{-1}S^*) - n\log|\Sigma_0^{-1}S^*| - np,$$

and the test is to reject H_0 if λ is improbably large when compared with a χ^2 distribution with $\frac{1}{2}p(p+1)$ degrees of freedom (using the asymptotic result of Wilks's Theorem).

(Notice in this case that λ depends only on the set of eigenvalues of $\Sigma_0^{-1}S^*$.)

7.5 LRT of $\mu'\mu = 1$ with known $\Sigma = I_p$

Suppose that $x_1, \ldots, x_n$ are independent observations of $x \sim N_p(\mu, \Sigma)$. Here's a test which is only really applicable in the multivariate case:

To test $H_0:\ \mu'\mu = 1$ against $H_A:\ \mu'\mu \neq 1$, with $\Sigma = I_p$.

Let $\ell(\mu; X) = -\frac{1}{2}(n-1)\text{tr}(S) - \frac{1}{2}n(\overline{x}-\mu)'(\overline{x}-\mu) - \frac{1}{2}np\log(2\pi)$. To maximize $\ell(\mu)$ under H_0, we need to impose the constraint $\mu\mu' = 1$ and so introduce a Lagrange multiplier and let $\Omega = \ell(\mu) - \lambda(\mu'\mu - 1)$. Then $\frac{\partial\Omega}{\partial\mu} = n(\overline{x}-\mu) - 2\lambda\mu$, and differentiating Ω with respect to λ gives $\mu'\mu = 1$. So we require $\hat{\mu} = \frac{n\overline{x}}{n+2\lambda}$ and then $\hat{\mu}'\hat{\mu} = 1$ implies that $(n+2\lambda)^2 = n^2\overline{x}'\overline{x}$. So $\hat{\mu} = \frac{\overline{x}}{\sqrt{\overline{x}'\overline{x}}}$, and

$$\begin{aligned}\ell_{\max}(H_0) &= -\tfrac{1}{2}(n-1)\text{tr}(S) - \tfrac{1}{2}n\left(\overline{x} - \frac{\overline{x}}{\sqrt{\overline{x}'\overline{x}}}\right)'\left(\overline{x} - \frac{\overline{x}}{\sqrt{\overline{x}'\overline{x}}}\right) - \tfrac{1}{2}np\log(2\pi) \\ &= \tfrac{1}{2}(n-1)\text{tr}(S) - \tfrac{1}{2}n\left(\sqrt{\overline{x}'\overline{x}} - 1\right)^2 - \tfrac{1}{2}np\log(2\pi).\end{aligned}$$

Under H_A, we have $\hat{\mu} = \overline{x}$, and so

$$\ell_{\max}(H_A) = \tfrac{1}{2}(n-1)\text{tr}(S) - \tfrac{1}{2}np\log(2\pi),$$

giving $\lambda = n(\sqrt{\overline{x}'\overline{x}} - 1)^2$, and the test is to reject H_0 when this is improbably large compared to a χ^2_1 distribution.

Note that there is only 1 degree of freedom, since μ has p independent parameters, so p are estimated under H_A, and under H_0 with one constraint, we have effectively $p-1$ parameters.

7.6 LRT of $\Sigma = \lambda\Sigma_0$; $\mu = \mu_0$ known, Σ_0 known

Suppose that $x_1, \dots, x_n$ are independent observations of $x \sim N_p(\mu, \Sigma)$.

$$\boxed{\text{To test } H_0 : \ \Sigma = \lambda\Sigma_0 \text{ against } H_A : \ \Sigma \neq \lambda\Sigma_0, \text{ both } \mu_0 \text{ and } \Sigma_0 \text{ known.}}$$

(Note that under H_0, λ is the only unknown parameter, but under H_A, all $\frac{1}{2}p(p+1)$ parameters of Σ are unknown.)

Under H_0,

$$\begin{aligned}\ell(\lambda; X) &= -\tfrac{1}{2}n\text{tr}((\lambda\Sigma_0)^{-1}S^\dagger) - \tfrac{1}{2}n\log|\lambda\Sigma_0| - \tfrac{1}{2}np\log(2\pi) \\ &= -\tfrac{1}{2}n\lambda^{-1}\text{tr}(\Sigma_0^{-1}S^\dagger) - \tfrac{1}{2}np\log\lambda - \tfrac{1}{2}n\log|\Sigma_0| - \tfrac{1}{2}np\log(2\pi),\end{aligned}$$

where $S^\dagger = \frac{1}{n}\sum_{i=1}^n (x_i - \mu_0)(x_i - \mu_0)'$. So

$$\frac{d\ell}{d\lambda} = \tfrac{1}{2}n\lambda^{-2}\text{tr}(\Sigma_0^{-1}S^\dagger) - \tfrac{1}{2}np\lambda^{-1},$$

giving $\hat{\lambda} = \frac{1}{p}\text{tr}(\Sigma_0^{-1}S^\dagger)$, and so

$$\ell_{\max}(H_0) = -\tfrac{1}{2}np - \tfrac{1}{2}\log\hat{\lambda} - \tfrac{1}{2}n\log|\Sigma_0| - \tfrac{1}{2}np\log(2\pi).$$

Under H_A, we have $\hat{\Sigma} = S^\dagger$, and so

$$\ell_{\max}(H_A) = -\tfrac{1}{2}np - \tfrac{1}{2}n\log|S^\dagger| - \tfrac{1}{2}np\log(2\pi).$$

Then the LRT statistic is $2(\ell_{\max}(H_A) - \ell_{\max}(H_0))$, and this would be compared with χ^2_r, where $r = \frac{1}{2}p(p+1) - 1$, using Wilks's Theorem.

Although this is not in a simple algebraic form, it can be calculated numerically in practice, and the test evaluated.

7.7 Comments

The first pair of these tests are multivariate generalisations of equivalent univariate hypotheses, and a useful check is to put $p = 1$ and verify that the univariate test is obtained.

The other two tests illustrate the more structured hypotheses that can be tested in multivariate problems; they have no counterpart in univariate models.

Such LRTs are a powerful all-purpose method of constructing tests, and can often be implemented numerically even if algebraic analysis cannot produce MLEs in closed form.

Further, they are an elegant application of general statistical theory, and have various desirable properties – they are guaranteed to be (asymptotically) most powerful, i.e., they are more likely than any other test to be able to detect successfully that the null hypothesis is false, provided that the sample is large enough, and provided that the parent distribution of the data is indeed that presupposed (e.g., multivariate normal).

One difficulty in multivariate problems involving hypothesis testing is that when a hypothesis is rejected, it may not be apparent just why it is false. This is not so in univariate problems; if we have a model that univariate $x \sim N(\mu, \sigma^2)$ and we reject $H_0 : \ \mu = \mu_0$, then we know whether $\overline{x} > \mu_0$ or $\overline{x} < \mu_0$ and hence why H_0 is false. In contrast, if we have a model that multivariate $x \sim N_p(\mu, \Sigma)$ and we reject $H_0 : \ \mu = \mu_0$, then all we know is that there is evidence that they are different, and we do not know the direction of departure from H_0; it may be that only one component in μ_0 is not correct. That is, a likelihood ratio test may be able to reject a hypothesis but not actually reveal anything interesting about the structure of the data, e.g., knowing that H_0 was nearly correct, and only one component was wrong could provide a useful insight into the data, but this might be missed by a LRT.

The next series of tests is an alternative strategy for constructing tests which might provide more information for data analysis, though if they actually produce a different test from the LRT then they may not be so powerful (at least for sufficiently large data sets).

7.8 Union-intersection tests

Union-intersection tests (UITs) provide a different strategy for constructing multivariate tests. They are not available in all situations (unlike LRTs), they do not have any general statistical optimal properties (again unlike LRTs) and sometimes they produce test statistics that can only be assessed for statistical significance by simulation or related procedures. However, they will automatically provide an indication of the *direction of departure* from a hypothesis.

The method is to project the data into one dimension (just as with many multivariate exploratory data analytic techniques) and test the hypothesis in that one dimension. The particular dimension chosen is that which shows the greatest deviation from the null hypothesis.

The validity of the procedure relies on the Cramér-Wold Theorem, which establishes the connection between the set of all 1-dimensional projections and the multivariate distribution.

Theorem 7.1 (Cramér-Wold) *The distribution of a p-vector x is completely determined by the set of all 1-dimensional distributions of 1-dimensional projections of x,* $t'x$*,* *where t runs over all fixed p-vectors.*

Proof. Let $y = t'x$, then, for any t, the distribution (and hence the characteristic function) of y is known and is

$$\phi_y(s) = E(e^{isy}) = E(e^{ist'x}).$$

Putting $s = 1$ shows that $\phi_y(1) = E(e^{it'x})$ is known for all $t \in \mathbb{R}^p$. But this is just the characteristic function of x, and therefore the distribution of x is known. □

The importance is that any multivariate distribution can be defined by specifying the distribution of all of its linear combinations (not just the p marginal distributions), e.g., if we specify that the mean of all 1-dimensional projections of x is 0, then necessarily the mean of the p-dimensional distribution must be 0 (the converse is also true, of course).

7.9 Example of a UIT

Suppose that $x \sim N_p(\mu, I_p)$. Then for any p-vector β, we have that if $y_\beta = \beta' x$ then $y_\beta \sim N(\beta'\mu, \beta'\beta)$ (and the Cramér-Wold Theorem shows that the converse is true; it shows that no other distribution can have the same marginal distributions).

Suppose that we want to test the hypothesis $H_0 : \ \mu = 0$, based just on the single observation x.

Then, under H_0, we have that for all β, $H_{0,\beta} : \ y_\beta \sim N(0, \beta'\beta)$ is true, and the converse for all β implies that H_0 is true.

Then H_0 is the "intersection" of all univariate hypotheses $H_{0,\beta}$. For any β, $H_{0,\beta}$ is a "component" of H_0.

Now, for any specific β, $H_{0,\beta}$ is the hypothesis that the mean of a normal distribution with known variance $\sigma^2 = \beta'\beta$ is zero, and we would reject H_0 at level α if $|\frac{y_\beta}{\sqrt{\beta'\beta}}| > c_\alpha$ where c_α is the upper $100(1 - \alpha/2)\%$ point of $N(0, 1)$. Thus we reject $H_{0,\beta}$ if

$$x \in R_\beta = \left\{ x \ : \ \left| \frac{\beta' x}{\sqrt{\beta'\beta}} \right| > c_\alpha \right\}.$$

Further, H_0 is true if and only if every $H_{0,\beta}$ is true. So a sensible rejection region for H_0 is the "union" of all the rejection regions for the component hypotheses $H_{0,\beta}$, i.e., we reject H_0 if $x \in \bigcup_\beta R_\beta$.

If H_0 is rejected, then we know which β (or βs) cause the rejection, and hence the direction of deviation from H_0.

Definition 7.2 A *union-intersection* test of a multivariate hypothesis is a test whose rejection region can be written as a union of rejection regions R_β, where R_β is the rejection region of a component hypothesis $H_{0,\beta}$, where H_0 is the intersection of the $H_{0,\beta}$.

In the above case, we reject H_0 if for any β we have $|\frac{y_\beta}{\sqrt{\beta'\beta}}| > c_\alpha$, so H_0 is *not* rejected if $\max_\beta |\frac{y_\beta}{\sqrt{\beta'\beta}}| \leq c_\alpha$, i.e., if $\max_\beta \frac{y'_\beta y_\beta}{\beta'\beta} \leq c_\alpha^2$, i.e., if $\max_\beta \frac{\beta' x x' \beta}{\beta'\beta} \leq c_\alpha^2$.

Now $\frac{\beta' x x' \beta}{\beta'\beta}$ is invariant under scalar multiplication of β, so we can impose $\beta'\beta = 1$, and maximise the numerator subject to this constraint.

Introducing a Lagrange multiplier gives the problem:

Maximise $\Omega = \beta' x x' \beta - \lambda(\beta'\beta - 1)$ with respect to β and λ.

Differentiating with respect to β gives $2xx'\beta - 2\lambda\beta = 0$, so β is an eigenvector of xx'. Now xx' is of rank 1 (if $x = (x_1 \ \cdots \ x_p)'$, then xx' is the $p \times p$ matrix with $(xx')_{ij} = x_i x_j$, so every row is a multiple of x' and every column is a multiple of x – thus its rank is 1), so only has one non-zero eigenvalue. This eigenvector of xx' is x, which has eigenvalue $x'x$. So the UIT of H_0 is to reject H_0 if $x'x > c_\alpha$, where c_α is chosen to give the desired size of test. Now $x'x \sim \chi^2_p$, so for a size α test, take $c_\alpha = 100(1-\alpha)\%$ point of χ^2_p.

This is actually the same as the LRT. For this problem, and clearly telling the direction of deviation from $\mu = 0$ is not difficult with just a single observation. The following examples illustrate cases where more information is obtained from the UIT than from the LRT.

7.10 UIT of $H_0: \ \mu = \mu_0$ against $H_A: \ \mu \neq \mu_0$ with Σ unknown

Let $x_1, \ldots, x_n$ be independent observations of $x \sim N_p(\mu, \Sigma)$.

To test $H_0: \ \mu = \mu_0$ against $H_A: \ \mu \neq \mu_0$ with Σ unknown.

Let β be any p-vector, and $y_\beta = \beta' x$. Then $y_\beta \sim N(\beta'\mu, \beta'\Sigma\beta)$, i.e., $y_\beta \sim N(\mu_y, \sigma^2_y)$, say.

A component hypothesis is $H_{0,\beta}: \ \mu_y = \mu_{0y}$ ($\mu_{0y} = \beta'\mu_0$). This needs a test of a univariate normal mean, with unknown variance; a usual 1-sample t-test, and we look at

$$t_\beta = \frac{\overline{y} - \mu_{0y}}{\sqrt{\frac{1}{n} s^2_y}},$$

where $s^2_y = \frac{1}{n-1}\sum_{i=1}^n (y_i - \overline{y})^2$ which expands to $\frac{1}{n-1}\sum_{i=1}^n \beta'(x_i - \overline{x})(x_i - \overline{x})'\beta = \beta' S \beta$. Also, $\overline{y} - \mu_{0y} = \beta'(\overline{x} - \mu_0)$ and

$$(\overline{y} - \mu_{0y})^2 = \beta'(\overline{x} - \mu_0)(\overline{x} - \mu_0)'\beta,$$

so

$$t^2_\beta = \frac{n\beta'(\overline{x} - \mu_0)(\overline{x} - \mu_0)'\beta}{\beta' S \beta},$$

and the component hypothesis $H_{0,\beta}$ is rejected if this is large.

The union-intersection test statistics is obtained by maximising t^2_β with respect to β. Now this is invariant under scalar multiplication of β, so impose $\beta' S \beta = 1$, and maximise

$$\Omega = n\beta'(\overline{x} - \mu_0)(\overline{x} - \mu_0)'\beta - \lambda(\beta' S \beta - 1)$$

with respect to β and λ.

Differentiating with respect to β shows that β satisfies

$$n(\overline{x} - \mu_0)(\overline{x} - \mu_0)'\beta - \lambda S \beta = 0,$$

so β is the eigenvector of the rank 1 $p \times p$-matrix $nS^{-1}(\overline{x} - \mu_0)(\overline{x} - \mu_0)'$ corresponding to the only non-zero eigenvalue (as above, $(\overline{x} - \mu_0)(\overline{x} - \mu_0)'$ has rank 1, and it easily follows that $S^{-1}(\overline{x}-\mu_0)(\overline{x}-\mu_0)'$ also does). By inspection, this is (a scalar multiple with length 1 of) $S^{-1}(\overline{x}-\mu_0)$. So $t^2 = n(\overline{x}-\mu_0)'S^{-1}(\overline{x}-\mu_0)$ which is Hotelling's T^2, and thus the UIT is identical to the LRT.

Further, if H_0 is rejected, then this shows that the direction of deviation is along $S^{-1}(\overline{x}-\mu_0)$, and we can interpret this direction by looking at the magnitude of the loadings on the individual components.

In the special case where $S = \sigma^2 I_p$ is a scalar matrix, then the direction of deviation is along $\overline{x}-\mu_0$ (i.e., this is the direction where the deviation from the null hypothesis is most pronounced).

7.11 UIT of $H_0: \ \Sigma = \Sigma_0$ against $H_A: \ \Sigma \neq \Sigma_0$ with μ unknown

Let $x_1, \dots, x_n$ be independent observations of $x \sim N_p(\mu, \Sigma)$.

To test $H_0: \ \Sigma = \Sigma_0$ against $H_A: \ \Sigma \neq \Sigma_0$ with μ unknown.

(Recall that the LRT for this problem was considered in §7.4.)

Then $H_{0,\beta}: \ \sigma_y^2 = \sigma_{0y}^2$, tested by $U_\beta = (n-1)s_y^2/\sigma_{0y}^2$ (which is χ^2_{n-1} under $H_{0,\beta}$) rejecting if $U_\beta < c_{1\beta}$ or if $U_\beta > c_{2\beta}$. So the UIT is obtained by rejecting H_0 if $\min_\beta(U_\beta) < c_1$ or $\max_\beta(U_\beta) > c_2$. Here, c_1 and c_2 are chosen to give the test the desired size.

Now $U_\beta = (n-1)\beta' S\beta/\beta'\Sigma_0\beta$, which is optimised when $(n-1)\Sigma_0^{-1}S\beta - \lambda\beta = 0$ and $\beta'\Sigma_0\beta = 1$. Then $(n-1)\beta' S\beta = \lambda\beta'\Sigma_0\beta = \lambda$. So the maximum and minimum values of U_β are the largest and smallest eigenvalues of $\Sigma_0^{-1}S$.

Thus the test

- is not the same as the LRT;
- indicates the direction of deviation is along one or other of the first or last eigenvectors
- requires simulation to apply in practice, since there are no general results for UITs comparable to Wilks's Theorem for LRTs: instead, one needs to essentially make your own tables – simulate some random data many times, computing the value of the appropriate statistic, and see how unlikely the value of your test statistic with the actual data appears when you do this.

7.12 Multisample tests – multivariate analysis of variance

Let's recall the set up from Chapter 5.

We suppose that there are k groups, labelled $G_1, \dots, G_k$, and that group G_i has n_i observations (as usual, with p dimensions). Write $n = \sum n_i$ for the total number of observations as usual.

Write X_i' for the data matrix for the group G_i (so X_i' is a $n_i \times p$ matrix), and so the data are $\{x_j^{(i)} \mid i = 1, \dots, k, \ j = 1, \dots, n_i\}$ where $x_j^{(i)}$ is a $p \times 1$ vector. Write $\overline{x}$ for the p-vector which is the overall mean of the n observations, and $\overline{x}_i$ for the p-vector which is the mean of group G_i. In matrix terms, the $n \times p$ data matrix of the whole sample is X', where $X = (X_1; \cdots ; X_k)$, and $\overline{x} = \frac{1}{n} X 1_n$ and $\overline{x}_i = \frac{1}{n_1} X_i 1_{n_i}$; here $\overline{X}_i = \overline{x}_i 1'_{n_i}$ is the $p \times n_i$ matrix with each row equal to $\overline{x}'_i$.

Let

$$S_i = \frac{1}{n_i - 1}(X_i - \overline{X}_i)(X_i - \overline{X}_i)'$$

be the *within group i* variance matrix. Then

$$S_i = \frac{1}{n_i - 1}\sum_{j=1}^{n_i}(x_j^{(i)} - \overline{x}_i)(x_j^{(i)} - \overline{x}_i)'$$

(so S_i is $p \times p$).

Let

$$W = \frac{1}{n-k}\sum_{i=1}^{k}(n_i - 1)S_i$$

be the *within groups variance* (again a $p \times p$ matrix). This measures the variability inside the groups. It is also known as the *pooled sample variance.*

Define

$$B = \frac{1}{k-1}\sum_{i=1}^{k} n_i(\overline{x}_i - \overline{x})(\overline{x}_i - \overline{x})',$$

the *between groups variance.* This measures the variability between the groups.

In linear discriminant analysis, we found that the product $W^{-1}B$ appeared naturally. We should regard this as a measure of comparison between W and B.

The fundamental equality in MANOVA is as follows. Suppose that

$$T = (X - \overline{X})(X - \overline{X})' = \sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_j^{(i)} - \overline{x})(x_j^{(i)} - \overline{x})' = (n-1)S$$

is the total sum of squares (where S is the overall variance), then

$$\boxed{T = (n-k)W + (k-1)B}$$

(which is the multivariance analysis of variance of total variance into within and between components). Readers may wish to check this by hand.

We suppose now that each group G_i arises as independent samples of a multivariate normal $N_p(\mu_i, \Sigma)$ distribution (notice that the variance matrix is the same for each group, which is a strong assumption in the multivariate case), and we want to develop a statistical test for the following null hypothesis:

$$\boxed{\text{To test } H_0 : \ \mu_1 = \cdots = \mu_k \text{ against } H_A\text{: } \mu_i \neq \mu_j \text{ for some pair } i \neq j}$$

The full theory of MANOVA (Multivariate ANalysis Of VAriance) resembles the univariate theory rather closely, although the notation is considerably more complicated due to the multivariate nature of the observations.

Esentially, though, the hypothesis test comes down to some measurement of the relative contributions of the within-groups matrix W and the between-groups matrix B.

Anderson's book discusses the choice of test procedures in more detail: he shows (section 8.10) that the only statistics invariant under translation (i.e., the choice of origin) and scaling of the data are functions of the roots of the equation

$$\det(B - \lambda W) = 0,$$

i.e., of eigenvalues of $W^{-1}B$ (assuming that W is invertible). That is, to test the null hypothesis above, we need to calculate the eigenvalues of $W^{-1}B$, and make some decision based on the results.

A number of choices can be used. Perhaps the simplest is to consider the largest eigenvalue of $W^{-1}B$, as proposed by Roy. Lawley and Hotelling proposed using the trace of $W^{-1}B$, i.e., the sum of the eigenvalues. Roy has also proposed using $\prod \frac{\lambda_i}{1+\lambda_i}$, and Pillai has suggested using $\sum \frac{\lambda_i}{1+\lambda_i}$, which can be interpreted as the determinant and trace of $(B(W+B)^{-1})$. We will tend to use Wilks's statistic, as it follows from the likelihood ratio theory:

7.13 MANOVA: Likelihood Ratio approach – Wilks Λ-test

Wilks developed the likelihood ratio theory for this test, and developed a test, known as *Wilks's* Λ-*test*, based on the statistic $|S|/|W| = |W^{-1}S|$, rejecting H_0 if this is improbably large.

As $(k-1)B = \sum_{i=1}^{k} n_i(\overline{x}_i - \overline{x})(\overline{x}_i - \overline{x})' = (n-1)S - (n-k)W$, an equivalent test statistic is $|W^{-1}((k-1)B + (n-k)W)|$, or equivalently $|I_p + \frac{k-1}{n-k}W^{-1}B|$, rejecting if this is large, or equivalently $\Lambda = |I_p + \frac{k-1}{n-k}W^{-1}B|^{-1}$, rejecting if this is small.

Let $A = \frac{k-1}{n-k}W^{-1}B$ have eigenvalues $\lambda_1, \ldots, \lambda_p$. Then Wilks's Λ is just

$$\Lambda = \prod_{i=1}^{p} \frac{1}{1+\lambda_i}.$$

Λ is said to have a Wilks Λ-distribution $\Lambda(p, n-k, k-1)$, which for some values of p, n, k (in particular, $k = 2$ or 3) is closely related to an F-distribution. For other values of p, n and k, approximations are available (see Chatfield-Collins, p.148). For $k = 2$, this test reduces to the 2-sample Hotelling's T^2-test (§6.5).

7.14 MANOVA: The Union-intersection test

As above, if β is any vector, then the test statistic for testing $H_{0,\beta}$ is $F_\beta = \beta' B\beta / \beta' W\beta$, whose maximum value is the largest eigenvalue of $W^{-1}B$ (see §5.2). An equivalent statistic is to look at the largest eigenvalue of A.

For $k = 2$, this reduces to the 2-sample Hotelling's T^2-test (the same as the LRT), but for $k > 2$, the UIT and LRT are different.

The null distribution of this largest eigenvalue is closely related to Roy's Greatest Root Distribution.

This gives another way of measuring the size of $W^{-1}B$.

7.15 MANOVA: R

R provides Roy's statistic as well as Wilks's statistic in the routines referred to in §7.13. In addition, they produce two further statistics: Pillai's trace, which is defined as $\sum_{i=1}^{p} \frac{\lambda_i}{1+\lambda_i}$ (and is the default test that R uses), and the Lawley-Hotelling trace, defined by $\sum_{i=1}^{p} \lambda_i$.

Thus we have four statistics for measuring the failure of the null hypothesis. All four of these statistics measure or reflect the "magnitude" of the matrix $W^{-1}B$ which is the obvious multivariate generalisation of the F-statistic in univariate 1-way analysis of variance. Generally all four tests should lead to equivalent conclusions – if they do not, there is something very unusual about the data which needs further investigation.

Hotelling's T^2-statistic is most easily computed as $(n-2)$ times the Lawley-Hotelling trace, using the MANOVA option described above. (However, using `manova` does require that one has all the data, not only the summary statistics.)

To use MANOVA in R, start by `attach`ing the data frame, and running MANOVA with

```
> frame.manova<-manova(cbind(variables)~group)
```

where `variables` should be replaced by the list of variables you are considering, and `group` by the name of the group in the data set. Then type one (or more) of

```
> summary(frame.manova,test="Pillai")
> summary(frame.manova,test="Wilks")
> summary(frame.manova,test="Hotelling-Lawley")
> summary(frame.manova,test="Roy")
```

Although they produce different statistics, when there are only two groups, all four tests have the same F-value and thus the same p-value. This is not the case for $k > 2$. For example, with the iris data, we can run

```
attach(irisnf)
iris.manova<-manova(cbind(Sepal.l,Sepal.w,Petal.l,Petal.w)~Variety)
summary(iris.manova,test="Pillai")
summary(iris.manova,test="Roy")
summary(iris.manova,test="Hotelling-Lawley")
summary(iris.manova,test="Wilks")
```

and see that the F-values vary considerably (all are, however, so large that the p-value is essentially 0 in every case).

In principle, further extensions of MANOVA (e.g., 2-way or General Multivariate Linear Model) are possible.

Remark 7.3 MANOVA is rarely the only stage in the analysis, not least because the interpretation of the results is often difficult. It is always useful to look at the separate univariate ANOVAs, supplemented by the first eigenvector of $W^{-1}B$.

A key advantage of MANOVA over p separate univariate ANOVAs is when an experiment consists of measuring lots of variables on the same individuals in the hope that at least one (or even some) will show differences between the groups, but it is not known which of the p variables will do so. This is a multiple comparison problem which is partially overcome by performing an initial MANOVA to see whether there are any differences at all between the groups. If the MANOVA fails to reveal any differences, then there is little point in investigating differences on separate variables further. If there is some overall difference between the groups, then examination of the coefficients in the first eigenvector of $W^{-1}B$, together with informal examination of the individual ANOVAs will indicate which variables or combination of variables (i.e., directions) contribute to the differences.

7.16 Example of Hotelling's T^2-test using MANOVA...

At the end of chapter 6, we mentioned a couple of ways of performing Hotelling's T^2-test. But the most convenient of all uses the MANOVA facilities in R, which we shall mention briefly next.

Now let's do a 2-sample T^2-test.

Example 7.4 The task sheets have provided questions on British Museum mummy pots, where there were two batches of sizes $n_1 = 15$ and $n_2 = 10$. Two measures were given, the rim and base circumferences. To test whether the two batches have the same mean circumferences, we need to calculate

$$\frac{n_1 n_2 (n-p-1)}{n(n-2)p}(\overline{x}_1 - \overline{x}_2)' S^{-1} (\overline{x}_1 - \overline{x}_2),$$

and compare it with $F_{p,n-p-1}$.

From the data, we have sample means

$$\overline{x}_1 = \begin{pmatrix} 476.33 \\ 225.67 \end{pmatrix}, \qquad \overline{x}_2 = \begin{pmatrix} 544.00 \\ 242.50 \end{pmatrix}$$

and group sample variances

$$S_1 = \begin{pmatrix} 1483.81 & 113.33 \\ 113.33 & 420.95 \end{pmatrix}, \qquad S_2 = \begin{pmatrix} 1915.55 & 786.11 \\ 786.11 & 784.72 \end{pmatrix}.$$

We write

$$S = \frac{(n_1-1)S_1 + (n_2-1)S_2}{n_1+n_2-2} = \begin{pmatrix} 1652.75 & 376.59 \\ 376.59 & 563.30 \end{pmatrix}$$

for the overall variance. This gives all the information required to compute the statistic

$$\frac{n_1 n_2 (n-p-1)}{n(n-2)p}(\overline{x}_1 - \overline{x}_2)' S^{-1} (\overline{x}_1 - \overline{x}_2) = 7.96,$$

which we need to compare with the 95% point (say) of the $F_{2,22}$ distribution, which is 3.44. We conclude that there is strong evidence of a difference in mean circumferences between the two batches.

Now let's do it in R, using the `manova` facility.

```
> attach(brmuseum)
> br.manova<-manova(cbind(rim.cir,base.circ)~batch)
> summary(br.manova)
          Df  Pillai approx F num Df den Df   Pr(>F)
```

```
batch      1 0.41989   7.9619      2     22 0.002504 **
Residuals 23
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
> summary(br.manova,test="Hotelling-Lawley")
          Df Hotelling-Lawley approx F num Df den Df   Pr(>F)
batch      1          0.72381   7.9619      2     22 0.002504 **
Residuals 23
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

From this, we can compute the value of Hotelling's T^2-statistic as $(n-2) \times 0.72381 = 16.6476$ and also read off the F-value as $\frac{n-p-1}{np}T^2 = \frac{23}{50} \times 16.65 = 7.962$, as calculated above. The `manova` function also gives the p-value, namely 0.0025, so this is highly significant. (Notice that the Pillai statistic is the default in `summary(br.manova)`, and while this gives the same p-value, it doesn't help in calculating the value of Hotelling's T^2.)