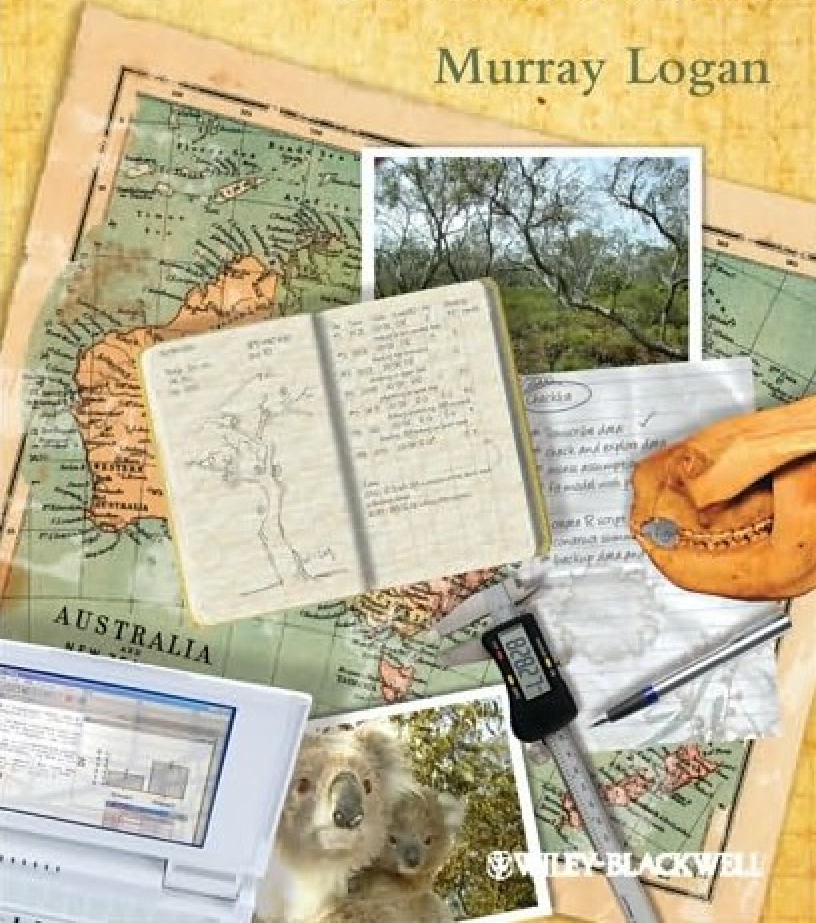
Biostatistical Design and Analysis Using R

A Practical Guide

Murray Logan



WILEY-BLACKWELL

Biostatistical Design and Analysis Using R

A Practical Guide

Murray Logan

WILEY-BLACKWELL

A John Wiley & Sons, Inc., Publication

Biostatistical Design and Analysis Using R

Companion website

A companion website for this book is available at:

www.wiley.com/go/logan/r

The website includes figures from the book for downloading.

Biostatistical Design and Analysis Using R

A Practical Guide

Murray Logan

WILEY-BLACKWELL
A John Wiley & Sons, Inc., Publication

Blackwell Publishing was acquired by John Wiley & Sons in February 2007. Blackwell's publishing program has been merged with Wiley's global Scientific, Technical and Medical business to form Wiley-Blackwell.

Registered office: John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

Editorial offices: 9600 Garsington Road, Oxford, OX4 2DQ, UK
The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK
111 River Street, Hoboken, NJ 07030-5774, USA

For details of our global editorial offices, for customer services and for information about how to apply for permission to reuse the copyright material in this book please see our website at www.wiley.com/wiley-blackwell

Library of Congress Cataloguing-in-Publication Data

Logan, Murray.
Biostatistical design and analysis using R : a practical guide / Murray Logan.
p. cm.
Includes bibliographical references and index.
ISBN 978-1-4443-3524-8 (hardcover : alk. paper) – ISBN 978-1-4051-9008-4 (pbk. : alk. paper)
1. Biometry. 2. R (Computer program language) I. Title.
QH323.5.L645 2010
570.1′5195 – dc22
2009053162

A catalogue record for this book is available from the British Library.

Typeset in 10.5/13pt Minion by Laserwords Private Limited, Chennai, India
Printed and bound in Singapore

1 2010

Contents

Companion website for this book: wiley.com/go/logan/r

Preface

R is a powerful and flexible statistical and graphical environment that is freely distributed under the GNU Public Licence[a] for all major computing platforms (Windows, MacOSX and Linux). This open source licence along with a relatively simple scripting syntax has promoted diverse and rapid evolution and contribution. As the broader scientific community continues to gain greater instruction and exposure to the overall project, the popularity of R as a teaching and research tool continues to accelerate.

It is now widely acknowledged that R proficiency as a scientific skill set is becoming increasingly more desirable and useful throughout the scientific community. However, as with most open source developments, the emphasis of the R project remains on the expansive development of tools and features. Applied documentation still remains somewhat sparse and somewhat incomprehensible to the average biologist. Whilst there are a number of excellent texts on R emerging, the bulk of these texts are devoted to the R language itself. Any featured examples therein are used primarily for the purpose of illustrating the suite of commonly used R features and procedures, rather than to illustrate how R can be used to perform common biostatistical analyses.

Coinciding with the increasing interest in R as both a learning and research tool for biostatistics, has been the success of a relatively new major biostatistics textbook (Quinn and Keough, 2002). This text provides detailed coverage of most of the major statistical concepts and tests that biologists are likely to encounter with an emphasis on the practical implementation of these concepts with real biological data. Undoubtedly, a large part of the appeal of this book is attributable to the extensive use of real biological examples to augment and reinforce the text. Furthermore, by concentrating on the information biologists need to implement their research, and avoiding the overuse of complex mathematical descriptions, the authors have appealed to those biologists who don't require (or desire) a knowledge of performing or programming entire analyses from scratch. Such biologists tend to use statistical software that is already available and specifically desire information that will help them achieve reliable statistical and biological outcomes. Quinn and Keough (2002) also advocate a number of alternative

[a] This is an open source licence that ensured that the application as well as its source code is freely available to use, modify and redistribute.

texts that provide more detailed coverage of specific topics and that also adopt this real example approach.

Typically, most biostatistical texts focus on the principles of design and analysis without extending into the practical use of software to implement these principles. Similarly, R/S-plus texts tend to concentrate on documenting and showcasing the features of R without providing much of a biostatistical account of the principles behind the features or illustrating how these tools can be extended to achieve comprehensive real world analyses. Consequently, many biological students and professionals struggle to translate the theoretical advice into computational outcomes. Although some of these difficulties can be addressed after extensively reading through a number of software references, many of the difficulties remain. The inconsistency and incompatibility between theory texts and software reference texts is mainly the result of differing intentions of the two genres and is a source of great frustration.

The reluctance of biostatistical texts to promote or instruct on any particular statistical software (except for extremely specialized cases where historically only a single dedicated program was available) is in part an acknowledgment of the diversity of software packages available (each of which differs substantially in the range of features offered as well as the user interface and output provided). Furthermore, software upgrades generally involve major alternations to the way in which preexisting tasks are performed and thus being associated with a single software package tends to restrict the longevity and audience of the text. In contrast, although contributers are constantly extending the feature set of R environments, overall the project maintains a consistent user interface. Consequently, there is currently both a need and opportunity for a text that fills the gap between biostatistics texts and software texts, so as to assist biologists with the practical side of performing statistical analysis.

Many biological researchers and students have at one stage or another used one or other of the major biostatistics texts and gained a good understanding of the principles. However, from time to time (and particularly when preparing to generate a new design or analyse a new data set), they require a quick refresher to help remind them of the issues and principles relevant to their current design and/or analysis scenarios. In most cases, they do not need to re-read the more discursive texts and in many cases express a reluctance to invest large amounts of valuable research time doing so. Therefore, there is also a need for a quick reference that summarizes the key concepts of contemporary biostatistics and leads users step-wise through each of the analysis procedures and options. Such a guide would also help users to identify their areas of statistical naivete and enable them to return to a more comprehensive text with a more focused and efficient objective.

Therefore, the intended focus of this book will be to highlight the major concepts, principles and issues in contemporary biostatistics as well as demonstrate how to use R (as a research design, analysis and presentation tool) to complete examples from major biostatistics textbooks. In so doing, this proposed text acknowledges the important role that statistical software and real examples play in reinforcing statistical principles and practices.

Hence in summary, the intentions of the book are three-fold

(i) To provide very brief refresher summaries of the main concepts, issues and options involved in a range of contemporary biostatistical analyses
(ii) To provide key guides that steps users through the procedures and options of a range of contemporary biostatistical analyses
(iii) To provide detailed R scripts and documentation that enable users to perform a range of real worked examples from statistics texts that are popular among biological and environmental scientists

Worked examples

Where possible and appropriate, this book will make use the same examples that appear in the popular biostatistical texts so as to take advantage of the history and information surrounding those examples as well as any familiarity that users may have with those examples. Having said this however, access to these other texts will not be necessary to get good value out of the materials.

Website

This book is augmented by a website (http://www.wiley.com./go/logan/r) which includes:

- raw data sets and R analysis scripts associated with all worked examples
- the `biology` *package* that contains many functions utilized in this book
- an R reference card containing links to pages within the book

Typographical convensions

Throughout this book, all R language objects and functions will be printed in courier (`monospaced`) typeface. Commands will begin with the standard R command prompt (`<`) and lines continuing on from a previous line will begin with the continuation prompt (`+`). In syntax used within the chapter keys, `dataset` is used as an example and should be replaced by the name of the actual data frame when used. Similarly, all vector names should be replaced by the names used to denote the various variables in your data set.

Acknowledgements

The inspiration for this book came primarily from Gerry Quinn and Mick Keough towards whom I am both indebted and infuriated (in equal quantities). As authors of a statistical piece themselves, they should known better than to encourage others

to attempt such an undertaking! I also wish to acknowledge the intellectualizing and suggestions of Patrick Baker and Andrew Robinson, the former of whom's regular supply of ideas remains a constant source of material and torment. Countless numbers of students and colleagues have also helped refine the materials and format of this book. As almost all of the worked examples in this book are adapted from the major biostatistical texts, the contributions of these other authors cannot be overstated. Finally, I would like to thank Nat, Kara, Saskia and Anika for your support and tolerance while I wrote this "extremely quite boring book with rid-ic-li-us pictures" (S. Logan, age 7).

R quick reference card

Session management

\> `q()` Quitting R (see page 8)
\> `ls()` List the objects in the current environment (see page 7)
\> `rm(...)` Remove objects from the current environment (see page 7)
\> `setwd(dir)` Set the current working directory (see page 7)
\> `getwd()` Get the current working directory (see page 7)

Getting help

\> `?function` Getting help on a function (see page 8)
\> `help(function)` Getting help on a function (see page 8)
\> `example(function)` Run the examples associated with the manual page for the function (see page 8)
\> `demo(topic)` Run an installed demonstration script (see page 8)
\> `apropos("topic")` Return names of all objects in search list that match "topic" (see page 9)
\> `help.search("topic")` Getting help about a concept (see page 9)
\> `help.start()` Launch R HTML documentation (see page 9)

Built in constants

\> `LETTERS` the 26 upper-case letters of the English alphabet (see page 17)
\> `letters` the 26 lower-case letters of the English alphabet (see page 17)
\> `month.name` English names of the 12 months of the year
\> `month.abb` Abbreviated English names of the 12 months of the year
\> `pi` π – the ratio of a circles circumference to diameter (see page 105)

Packages

\> `installed.packages()` List of all currently installed packages (see page 44)
\> `update.packages()` Update installed packages (see page 44)
\> `install.packages(pkgs)` Install package(s) (pkgs) from CRAN mirror (see page 45)
`R CMD INSTALL package` Install an add-on package (see page 43)
\> `library(package)` Loading an add-on package (see page 45)
\> `data(name)` Load a data set or structure inbuilt into R or a loaded package.

Importing/Exporting

\> `source("file")` Input, parse and sequentially evaluate the file (see page 45)
\> `sink("file")` Redirect non-graphical output to file
\> `read.table("file", header=T, sep=)` Read data in table format and create a data frame, with variables in columns (see page 51)
\> `read.table("clipboard", header=T, sep=)` Read data left on the clipboard in table format and create a data frame, with variables in columns (see page 51)
\> `read.systat("file.syd", to.data.frame=T)` Read SYSTAT data file and create a data frame (see page 52)
\> `read.spss("file.sav", to.data.frame=T)` Read SPSS data file and create a data frame (see page 52)
\> `as.data.frame(read.mtp("file.mtp"))` Read Minitab Portable Worksheet data file and create a data frame (see page 52)
\> `read.xport("file")` Read SAS XPORT data file and create a data frame (see page 52)
\> `write.table(dataframe, "file", row.names=F, quote=F, sep=)` Write the contents of a dataframe to file in table format (see page 53)
\> `save(object, file="file.RData")` Write the contents of the object to file (see page 53)
\> `load(file="file.RData")` Load the contents of a file (see page 53)
\> `dump(object, file="file")` Save the contents of an object to a file (see page 53)

Generating Vectors

\> `c(...)` Concatenate objects (see page 6)
\> `seq(from, to, by=, length=)` Generate a sequence (see page 12)
\> `rep(x, times, each)` Replicate each of the values of x (see page 13)

Character vectors

\> `paste(..., sep=)` Combine multiple vectors together after converting them into character vectors (see page 13)
\> `substr(x, start, stop)` Extract substrings from a character vector (see page 14)

Factors

\> `factor(x)` Convert the vector (x) into a factor (see page 15)

> **`factor(x, levels=c())`** Convert the vector (x) into a factor and define the order of levels (see page 15)
> **`gl(levels, reps, length, labels=)`** Generate a factor vector by specifying the pattern of levels (see page 15)
> **`levels(factor)`** Lists the levels (in order) of a factor (see page 54)
> **`levels(factor) <-`** Sets the names of the levels of a factor (see page 54)

Matrices

> **`matrix(x,nrow, ncol, byrow=F)`** Create a matrix with `nrow` and/or `ncol` dimensions out of a vector (x) (see page 16)
> **`cbind(...)`** Create a matrix (or data frame) by combining the sequence of vectors, matrices or data frames by columns (see page 16)
> **`rbind(...)`** Create a matrix (or data frame) by combining the sequence of vectors, matrices or data frames by rows (see page 16)
> **`rownames(x)`** Read (or set with **`<-`**) the row names of the matrix (`x`) (see page 17)
> **`colnames(x)`** Read (or set with **`<-`**) the column names of the matrix (`x`) (see page 17)

Lists

> **`list(...)`** Generate a list of named (for arguments in the form `name=x`) and/or unnamed (for arguments in the form (`x`) components from the sequence of objects (see page 17)

Data frames

> **`data.frame(...)`** Convert a set of vectors into a data frame (see page 49)
> **`row.names(dataframe)`** Read (or set with **`<-`**) the row names of the data frame (see page 49)
> **`fix(dataframe)`** View and edit a dataframe in a spreadsheet (see page 49)

Indexing

Vectors
> **`x[i]`** Select the i^{th} element (see page 21)
> **`x[i:j]`** Select the i^{th} through j^{th} elements inclusive (see page 21)
> **`x[c(1,5,6,9)]`** Select specific elements (see page 21)
> **`x[-i]`** Select all except the i^{th} element (see page 21)
> **`x["name"]`** Select the element called `"name"` (see page 21)
> **`x[x > 10]`** Select all elements greater than 10 (see page 22)
> **`x[x > 10 & x < 20]`** Select all elements between 10 and 20 (both conditions must be satisfied) (see page 22)
> **`x[y == "value"]`** Select all elements of x according to which y elements are equal to "value" (see page 22)
> **`x[x > 10 | y == "value"]`** Select all elements which satisfy either condition (see page 22)
Matricies
> **`x[i,j]`** Select element in row i, column j (see page 23)
> **`x[i,]`** Select all elements in row i (see page 23)
> **`x[,j]`** Select all elements in column j (see page 23)
> **`x[-i,]`** Select all elements in each row other than the i^{th} row (see page 23)
> **`x["name",1:2]`** Select columns 1 through to 2 for the row named `"name"` (see page 23)
> **`x[x[,"Var1"]>4,]`** Select all rows for which the value of the column named `"Var1"` is greater than 4 (see page 23)
> **`x[,x[,"Var1"]=="value"]`** Select all columns for which the value of the column named `"Var1"` is equal to "value"
Lists
> **`x[[i]]`** Select the i^{th} object of the list (see page 24)
> **`x[["value"]]`** Select the object named `"value"` from the list (see page 24)
> **`x[["value"]][1:3]`** Select the first three elements of the object named `"value"` from the list (see page 24)
Data frames
> **`x[c(i,j),]`** Select rows i and j for each column of the data frame (see page 56)
> **`x[,"name"]`** Select each row of the column named `"name"` (see page 56)
> **`x[["name"]]`** Select the column named `"name"`
> **`x$name`** Refer to a vector named `"name"` within the data frame (`x`) (see page 53)

Object information

> **`length(x)`** number of elements in x (see page 34)
> **`class(x)`** get the class of object x (see page 18)
> **`class(x)<-`** set the class of object x (see page 18)
> **`attributes(x)`** get (or set) the attributes of object x (see page 19)
> **`attr(x, which)`** get (or set) the *which* attribute of object x (see page 19)
> **`is.na(x), is.numeric(x), is.character(x), is.factor(x), ...`** methods used to assess the type of object x (`methods(is)` provides full list) (see page 18)

Object conversion

> **`as.null(x), as.numeric(x), as.character(x), as.factor(x), ...`** methods used to covert x to the

specified type (`methods(is)` provides full list) (see page 20)

Data manipulations

> **`subset(x, subset=, select=)`** Subset a vector or data frame according to a set of conditions (see page 56)
> **`sample(x, size)`** Randomly resample `size` number of elements from the `x` vector without replacement. Use the option `replace=TRUE` to sample with replacement. (see page 76)
> **`apply(x, INDEX, FUN)`** Apply the function (*FUN*) to the margins (`INDEX=1` is rows,`INDEX=2` is columns, `INDEX=c(1,2)` is both) of a vector, array or list (*x*) (see page 29)
> **`tapply(x, factorlist, FUN)`** Apply the function (*FUN*) to the vector (*x*) separately for each combination of the list of factors (see page 30)
> **`lapply(x, FUN)`** Apply the function (*FUN*) to each element of the list `x` (see page 30)
> **`replicate(n, EXP)`** Re-evaluate the expression (`EXP`) *n* times. Differs from `rep` function which repeats the result of a single evaluation (see page 28)
> **`aggregate(x, by, FUN)`** Splits data according to a combination of factors and calculates summary statistics on each set (see page 58)
> **`sort(x, decreasing=)`** Sorts a vector in increasing or decreasing (default) order (see page 26)
> **`order(x, decreasing=)`** Returns a list of indices reflecting the vector sorted in ascending or descending order (see page 26)
> **`rank(x, ties.method=)`** Returns the ranks of the values in the vector, tied values averaged by default (see page 27)
> **`which.min(x)`** Index of minimum element in `x`
> **`which.max(x)`** Index of maximum element in `x`
> **`rev(x)`** Reverse the order of entries in the vector (`x`) (see page 27)
> **`unique(x)`** Removes duplicate values (see page 337)
> **`t(x)`** Transpose the matrix or data frame (`x`) (see page 387)
> **`cut(x, breaks)`** Creates a factor out of a vector by slicing the vector `x` up into chunks. The option `breaks` is either a number indicating the number of cuts or else a vector of cut values (see page 111)
> **`which(x == a)`** Each of the elements of `x` is compared to the value of `a` and a vector of indices for which the logical comparison is true is returned
> **`match(x,y)`** A vector of the same length as `x` with the indices of the first occurance of each element of `x` within `y`
> **`choose(n,k)`** Computes the number of unique combinations in which `k` events can be arranged in a sequence of `n`
> **`combn(x,k)`** List all the unique combinations in which the elements of `x` can be arranged when taken `k` elements at a time
> **`with(x,EXP)`** Evaluate an expression (`EXP`) (typically a function) in an environment defined by `x` (see page 59)

Search and replace

> **`grep(pattern, x, ...)`** Searches a character vector (`x`) for entries that match the pattern (pattern) (see page 24)
> **`regexpr(pattern, x, ...)`** Returns the position and length of identified pattern (`pattern`) within the character vector (`x`) (see page 25)
> **`gsub(pattern, replacement, x, ...)`** Replaces ALL occurrences of the pattern (`pattern`) within the character vector (`x`) with replacement (`replacement` (see page 26)
> **`sub(pattern, replacement, x, ...)`** Replaces THE FIRST occurrence of the pattern (`pattern`) within the character vector (`x`) with replacement (`replacement` (see page 26)

Formating data

> **`ceiling(x)`** Rounds vector entries up to the nearest integer that is no smaller than the original vector entry (see page 27)
> **`floor(x)`** Rounds vector entries up to the nearest integer that is no smaller than the original vector entry (see page 27)
> **`trunc(x)`** Rounds vector entries to the nearest integer towards '0' (zero) (see page 27)
> **`round(x, digits=)`** rounds vector entries to the nearest numeric with the specified number of decimal places (`digit=`). Digits of 5 are rounded off to the nearest even digit (see page 27)
> **`formatC(x, format=, digits=, ...)`** Format vector entries according to a set of specifications (see page 28)

Math functions

Summary statistics
> **`mean(x)`** Mean of elements of `x` (see page 70)
> **`var(x)`** Variance of elements of `x` (see page 70)
> **`sd(x)`** Standard deviation of elements of `x` (see page 70)
> **`length(x)`** Number of elements of `x` (see page 34)
> **`sd(x)/sqrt(length(x))`** Standard error of elements of `x` (see page 70)
> **`quantile(x, probs=)`** Quantiles of `x` corresponding to probabilities (default: `0,0.25,0.5,0.75,1`)
> **`median(x)`** Median of elements of `x` (see page 70)
> **`min(x)`** Minimum of elements of `x` (see page 70)

> **`max(x)`** Maximum of elements of `x` (see page 70)
> **`range(x)`** Same as `c(min(x), max(x))` (see page 111)
> **`sum(x)`** Sum of elements of `x` (see page 106)
> **`cumsum(x)`** A vector the same length as *x* and whose i^{th} element is the sum of all elements up to and including *i*
> **`prod(x)`** Product of elements of `x`
> **`cumprod(x)`** A vector the same length as *x* and whose i^{th} element is the product of all elements up to and including *i*
> **`cummin(x)`** A vector the same length as *x* and whose i^{th} element is the minimum value of all elements up to and including *i*
> **`cummax(x)`** A vector the same length as *x* and whose i^{th} element is the maximum value of all elements up to and including *i*
> **`var(x,y)`** variance between *x* and y (matrix if `x` and `y` are matrices of data frames)
> **`cov(x,y)`** covariance between *x* and y (matrix if `x` and `y` are matrices of data frames)
> **`cor(x,y)`** linear correlation between *x* and y (matrix if `x` and `y` are matrices of data frames) (see page 226)

Scale trasformations

> **`exp(x)`** Transform values to exponentials (see page 212)
> **`log(x)`** Transform values to log_e (see page 69)
> **`log(x, 10)`** Transform values to $log_1 0$ (see page 69)
> **`log10(x)`** Transform values to $log_1 0$ (see page 69)
> **`sqrt(x)`** Square root transform values of `x` (see page 69)
> **`asin(sqrt(x))`** Arcsin transform values of `x` (which must be proportions) (see page 69)
> **`rank(x)`** Transform values of `x` to ranks (see page 27)
> **`scale(x, center=, scale=)`** Scales (mean of 0 and sd of 1) values of `x` to ranks. To only center data, use `scale='FALSE'`, to only reduce data use `center='FALSE'` (see page 220)

Distributions

The following are used for the following list of distribution functions

`x=` a vector of quantiles
`q=` a vector of quantiles
`p=` a vector of probabilities
`n=` the number of observations

> **`dnorm(x, mean, sd), pnorm(q, mean, sd), qnorm(p, mean, sd), rnorm(n, mean, sd)`** Density, distribution function, quantile function and random generation for the normal distribution with mean equal to `mean` and standard deviation equal to `sd` (see page 63)
> **`dlnorm(x, meanlog, sdlog), pnorm(q, meanlog, sdlog), qnorm(p, meanlog, sdlog), rnorm(n, meanlog, sdlog)`** Density, distribution function, quantile function and random generation for the log normal distribution whose logarithm has a mean equal to `meanlog` and standard deviation equal to `sdlog` (see page 63)
> **`dunif(x, min, max), punif(q, min, max), qunif(p, min, max), runif(n, min, max)`** Density, distribution function, quantile function and random generation for the uniform distribution with a minimum equal to `min` and maximum equal to `max` (see page 63)
> **`dt(x, df), pt(q, df), qt(p, df), rt(n, df)`** Density, distribution function, quantile function and random generation for the t distribution with `df` degrees of freedom
> **`df(x, df1, df2), pf(q, df1, df2), qf(p, df1, df2), rf(n, df1, df2)`** Density, distribution function, quantile function and random generation for the F distribution with `df1` and `df2` degrees of freedom
> **`dchisq(x, df), pchisq(q, df), qchisq(p, df), rchisq(n, df)`** Density, distribution function, quantile function and random generation for the chi-squared distribution with `df` degrees of freedom (see page 499)
> **`dbinom(x, size, prob), pbinom(q, size, prob), qbinom(p, size, prob), rbinom(n, size, prob)`** Density, distribution function, quantile function and random generation for the binomial distribution with parameters `size` and `prob` (see page 63)
> **`dnbinom(x, size, mu), pnbinom(q, size, mu), qnbinom(p, size, mu), rnbinom(n, size, mu)`** Density, distribution function, quantile function and random generation for the negative binomial distribution with parameters `size` and `mu` (see page 63)
> **`dpois(x, lambda), ppois(q, lambda), qpois(p, lambda), rpois(n, lambda)`** Density, distribution function, quantile function and random generation for the Poisson distribution with parameter `lambda` (see page 63)

Spatial procedures

`sp package`

> **`Polygon(xy)`** Convert a 2-column numeric matrix (`xy`) with coordinates into a object of class Polygon. Note the first point (row) must be equal to the last coordinates (row) (see page 79)
> **`Polygons(Plygn, ID)`** Combine one or more Polygon objects (*Plygn*) together into an object of class Polygons. (see page 80)
> **`SpatialPolygons(xy)`** A list of one or more Polygons . (see page 80)
> **`spsample(x, n, type=)`** Generate approximately `n` points on or within a SpatialPolygons object (`x`). The

option `type=` indicates the type of sampling (`"random"`, `"regular"`, `"stratisfied"` or `"non-aligned"`) (see page 80)

Plotting

> **`hist(x, breaks)`** Histogram of the frequencies of vector `x`. The option `breaks` specifies how the bins are constructed and is typically either a number (number of bins), a vector of breakpoints (see page 116)

> **`plot(x)`** Plot the values of `x` (on *y*-axis) ordered on *x*-axis (see page 85)

> **`plot(x, y)`** Scatterplot of `y` (on *y*-axis) against `x` (*x*-axis) (see page 37)

> **`plot(formula)`** If all vectors numeric - Scatterplot of `lhs` (on *y*-axis) against `rhs` (*x*-axis), otherwise a "box-and-whisker" plot with a separate box for each combination of `rhs` categories (see page 37)

> **`boxplot(x)`** "Box-and-whiskers" plot for vector or formula `x` (see page 119)

> **`pairs(x)`** Scatterplot matrices for multiple numeric vectors or formula `x` (see page 122)

> **`Mbargraph(dv, iv)`** Bargraph (*biology package*) of mean `dv` against categorical `iv` with error bars (see page 268)

> **`interaction.plot(x.fact, trace.fact, response)`** Plots the mean (or other summary) of the response (`response`) for two-way combinations of factors (*x*-axis factor: `x.fact` and trace factor: `trace.fact`), thereby illustrating possible interactions (see page 126)

> **`scatterplot(x)`** (`car package`) Fancy scatterplot for a pair of numeric vectors or formula `x`. Includes boxplots on margins and regression line (see page 121)

> **`scatterplot.matrix(x)`** (`car package`) Fancy scatterplot matrices for multiple numeric vectors or formula `x`. Includes univariate displays in diagonals (see page 122)

Low-level plotting commands

> **`points(x, y)`** Adds points with coordinates `x, y`. Option `type=` can be used (see page 99)

> **`lines(x, y)`** Adds lines with coordinates `x, y`. Option `type=` can be used (see page 109)

> **`abline(fit)`** Adds a regression line from the linear model `fit` (see page 109)

> **`abline(a, b)`** Adds a regression line with a *y*-intercept of `a` and a slope of `b`

> **`axis(text, at, labels, ...)`** Adds an axis to the bottom (`side=1`), left (`side=2`), top (`side=3`) or right (`side=4`) plot margin. Options `at` and `labels` can be used to specify where to draw tick marks and what labels to put at each tick mark (see page 107)

> **`box(which=, bty=, ...)`** Draws a box around the plot (`which="plot"`), figure (`which="figure"`), inner (`which="inner"`) or outer (`which="outer"`) region of the current plot. Option `bty` specifies the type of box to draw (`"o"`, `"l"`, `"7"`, `"c"`, `"u"` or `"]"` result in boxes that resembles the corresponding upper case letter) (see page 127)

> **`mtext(text, side, line=0, ...)`** Adds text (`text`) to the plot margin specified by `side` (see `axis()` above). Option `line` specifies the distance (in lines) away from the axis to put the text (see page 101)

> **`matlines(x, y, ...)`** Adds confidence or prediction (`y`) limits along a sequence (`x`) to the plot (see page 113)

> **`data.ellipse(x, y, levels, ...)`** Adds data ellipses from vectors (`x, y`) to the plot (see page 184)

> **`confidence.ellipse(x, y, levels, ...)`, `confidence.ellipse(model, ...)`** Adds confidence ellipses to the plot for linear models from vectors (`x, y`) or fitted model

Model fitting

> **`contrasts(x)`** View the contrasts associated with the factor `x` (see section 7.3.1)

> **`contrasts(x) <- value`** Set the contrasts associated with the factor `x`. The `value` parameter can either be a numeric matrix of coefficients or else a quoted name of a function that computes the matrix.(see section 7.3.1)

> **`lm(formula)`** Fit linear model from `formula` of format `response ~ predictor1 + predictor2 + ....` use `I(x*y) + I(x^2)` to include nonlinear terms (see chapters 8&10)

> **`lm.II(formula)`** (`biology package`) Fit linear model II regression from `formula` of format `response~predictor`. (see chapter 8)

> **`rlm(formula)`** (`MASS package`) Fit M-estimator linear model from `formula` of format `response~predictor`. (see chapter 8)

> **`mblm(formula)`** (`mblm package`) Fit nonparametric regression model from `formula` of format `response~predictor`. (see chapter 8)

> **`glm(formula, family)`** Fit generalized linear model from `formula`. Error distribution and link function are specified by `family` - see `family()` (see chapter 17)

> **`aov(formula)`** Fit an anova model by making a call to `lm` for each stratum within `formula` (see chapters 10-15)

> **`nls(formula, start)`** Determine the nonlinear least-squares estimates of the parameters of a nonlinear model `formula`. Starting estimates are provided as a named list or numeric vector (`start`) (see chapter 9)

> **`lme(fixed, random, correlation, ...)`** (*nlme package*) Fit linear mixed effects models from

a specification of the fixed-effects formula (`fixed`) and random-effects formula (`random`) and correlation structure (`correlation`) (see chapters 11-14)

> **`lmer(formula, ...)`** (*lme4 package*) Fit (generalized) linear mixed effects models from a specification of a formula (`formula`) (see chapters 11-14)

> **`gam(`*`formula`*`, family=, ...)`** (*gam package*) Fit generalized additive models from the formula (`formula`). Error distribution and link function are specified by `family` - see `family()` (see chapter 17)

> **`pvals.fnc(.lmer, nsim, withMCMC, ...)`** (*languageR package*) Calculate p-values from lmer models (`.lmer`) via Markov Chain Monte Carlo sampling. (see chapters 11-14)

> **`VarCorr(fit)`** (*nlme package*) Calculate variance components from a linear mixed effects model (`fit`). (see chapters 11-14)

Fit diagnostics *The following generic functions can be applied to some of the above fitted model objects*

> **`plot(fit`**) Diagnostic plots for a fitted model `fit` (see chapters 8-15)

> **`av.plots(fit`**) Added-variable (partial-regression) plots for a fitted model `fit` (see chapter 9)

> **`residuals(fit`**) Residuals from a fitted model `fit` (see chapters 8-15)

> **`deviance(fit`**) Deviance of a fitted model `fit` (see chapter 17)

> **`influence.measures(fit`**) Regression diagnostics for a fitted model `fit` (see chapters 8-15, 17)

> **`vif(fit`**) Calculate variance-inflation factor for a fitted model `fit` (see chapters 9, 17)

> **`1/vif(fit`**) Calculate tolerance for each term in a fitted model `fit` (see chapters 9, 17)

> **`predict(fit, data.frame`**) Predicted responses from a fitted model `fit` given a set of predictor values `data.frame` (see chapters 8-15, 17)

> **`confint(fit`**) Parameter confidence intervals from a fitted model `fit` (see chapters 8-15, 17)

> **`replications(formula`**) Determine the number of replicates of each term in `formula` (see chapters 11-15)

> **`is.balanced(formula`**) (*biology package*) Determine whether the design specified by the `formula` is balanced (see chapters 11-15)

> **`tukey.nonadd.test(fit)`** (*alr3 package*) Perform Tukey's test for nonadditivity from a model (`fit`) fitted via `lm()` (see chapters 13-15)

Measures of model fit

> **`extractAIC(fit, ...)`** Compute AIC for parametric model (fit). Equivalent BIC using `k=log(rnow(dataset))` argument. (see chapters 9 & 17)

> **`AIC(fit, ...)`** Compute AIC for any model (fit). Equivalent BIC using `k=log(rnow(dataset))` argument. (see chapters 9 & 17)

> **`AICc(fit)`** (*MuMIn package*) Compute AIC corrected for small sample sizes for a fitted model (*fit*). (see chapters 9 & 17)

> **`QAIC(fit)`** (*MuMIn package*) Compute quasi-AIC corrected for overdispersion for a fitted model (*fit*). (see chapters 9 & 17)

> **`QAICc(fit)`** (*biology package*) Compute quasi-AIC corrected for overdispersion and small sample sizes for a fitted model (*fit*). (see chapters 9 & 17)

> **`deviance(fit)`** Compute deviance for a fitted model (*fit*). (see chapters 9 & 17)

> **`Model.selection(fit)`** (*biology package*) Generate various model fit estimates and perform model averaging for all possible combinations of predictor variables in a supplied model `fit`. (see chapters 9 & 17)

> **`dredge(fit)`** (*MuMIn package*) Select most parsimonious model from all possible combinations of predictor variables in a supplied model `fit` based on information criteria (`rank=` either `"AICc"`, `"QAIC"` or `"BIC"`). (see chapters 9 & 17)

> **`model.avg(ml)`** (*MuMIn package*) Perform model averaging from a supplied fitted model object `ml` returned from model dredging. (see chapters 9 & 17)

Post-hoc analyses

> **`mainEffects(`*`fit, at`*`)`** (*biology package*) Perform main effects tests from the fitted model (`fit`) (see chapters 12-15, 17)

> **`glht(`*`fit, linfct=mcp(FACTOR=type)`*`)`** (*multcomp package*) Post-hoc, pairwise comparisons of factor (`FACTOR`). Option `type` specifies what type of post-hoc test to perform (`"Dunnett"`, `"Tukey"`, `"Sequen"`, `"AVE"`, `"Changepoint"`, `"Williams"`, `"Marcus"`, `"McDermott"`) (see chapter 10)

> **`npmc(`*`dataset, ...`*`)`** (*npmc package*) Non-parametric post-hoc, pairwise comparisons on a specifically constructed dataset (`dataset`). (see chapter 10)

> **`mt.rawp2adjp(`*`pvalues, proc`*`)`** (*multtest package*) Multiple pairwise comparison p-value (`pvalues`) adjustments . (see chapter 10)

> **`p.adjust(`*`pvalues, method`*`)`** Multiple pairwise comparison p-value (`pvalues`) adjustments . (see chapter 10)

Statistics and summaries

> **`t.test(`*`x, y`*`), t.test(`*`formula`*`)`** One and two sample t-tests on vectors (`x, y`) or formula `formula`. Option `var.equal` indicates whether pooled or separate variance t-test and option `paired` indicates whether independent or paired t-test (see chapter 6)

> **`cor.test(x, y), cor.test(formula)`** Correlation between sample pairs from separate vectors (`x, y, ...`) or formula `formula, ...`. Option `method` indicates the form of correlation (`'pearson', kendall` or `spearman'`) (see chapter 8)

> **`hier.part(y, data, gof)`** (*hier.part package*) Hierarchical partitioning given a vector of dependent variables `y` and a data frame `data`. Option `gof=` used to specify assessment of fit (root mean square prediction error: `"RMSPE"`, Log-Likelihood: `"logLik"` or R-squared: `"Rsqu"`) (see chapter 9)

> **`anova(fit, ...)`** Compute analysis of variance table for a fitted model `fit` or models (see chapters 8-15, 17)

> **`summary(fit)`** Summarize parameter estimates for a fitted model `fit` (see chapters 8-15, 17)

> **`AnovaM(fit, ...)`** (*biology package*) Compute analysis of variance table for a fitted model `fit` accommodating unbalanced hierarchical designs (see chapters 11-15)

> **`wilcox.JN(fit)`** (*biology package*) Perform Wilcoxon modified Johnson-Neyman procedure on fitted ANCOVA model (`fit`) (see chapter 15)

> **`tree(formula, ...)`** (*tree package*) Perform binary recursive partitioning (regression tree) from response and predictors specified in `formula`. (see chapter 9)

Robust statistics

> **`wilcox.test(x, y), t.test(formula)`** One and two sample ("Mann-Whitney") Wilcoxon testson vectors (`x, y`) or formula `formula`. Option indicates whether independent or paired Wilcoxon-test (see chapter 6)

> **`oneway.test(formula,...)`** Perform Welch's test comparing the means of two or more groups specified by formula `formula, ...`. (see chapter 10)

> **`kruskal.test(formula,...)`** Perform Kruskal-Wallis rank sum test, specified by formula `formula, ...`. (see chapter 10)

> **`friedman.test(formula,...)`** Perform Friedman rank sum test with unreplicated blocked data, specified by formula `formula, ...`. (see chapter 13)

> **`friedmanmc(DV, FACTOR, BLOCK)`** (*pgirmess library*) Multiple pairwise comparison test following Friedman's test. (see chapter 13)

Frequency analysis

> **`chisq.test(x)`** Performs chi-squared goodness-of-fit tests and contingency table tests. (see chapter 16)

> **`fisher.test(x)`** Performs fishers exact test goodness-of-fit tests and contingency table tests. (see chapter 16)

> **`ks.test(x)`** Performs Kolmogorov-Smirnov tests. (see chapter 16)

> **`g.test(x)`** (*biology package*) Performs G-test for goodness-of-fit tests and contingency table tests. (see chapter 16)

> **`oddsratios(xtab)`** (*biology package*) Calculate pairwise odds ratios from a contingency table (`xtab`). (see chapter 16-17)

Bootstrapping

> **`boot(data, stat, R, sim, rand.gen)`** (*boot package*) Generates `R` bootstrap replicates from a statistical function (`stat`) incorporating a particular simulation (`sim=` one of `"parametric"`, `"balanced"`, `"permutation"` or `"arithetic"`). Function `rand.gen` defines how randomization occurs (see page 149)

Power analysis

> **`power.t.test(n, delta, sd, power)`** Calculate one of; sample size (`n`), true difference in means (`delta`), standard deviation (`sd`) or power (`power`) of t-test. The option `type` indicates the type of t-test (`"two.sample", "one.sample", "paired"`)

> **`pwr.r.test(n, r, power)`** (*pwr package*) Calculate one of; sample size (`n`), correlation coefficient (`r`) or power (`power`) of t-test. > **`power.anova.test(groups, n, between.var, within.var, power)`** Calculate one of; number of groups (`groups`), sample size (`n`), between group variance (`between.var`), within group variation (`within.var`) or power (`power`) of ANOVA.

> **`pwr.chisq.test(w, N, df, power)`** (*pwr package*) Calculate one of; effect size (`w`), total number of observations (`N`), degrees of freedom (`df`) or power (`power`) of chi-square test.

General key to statistical methods

1 a. **Testing a specific null hypothesis or effects** .... Go to 3
b. **Not testing a specific null hypothesis** .... Go to 2

2 a. **Statistical or numerical summaries** .... Chapter 3
b. **Graphical summaries** .... Chapter 5

3 a. **Response variable continuous** .... Go to 4
b. **Response variable categorical or frequencies** .... Go to 10

4 a. **One or more categorical predictor (independent) variables** .... Go to 6
Testing a null hypotheses about group differences
b. **One or more continuous predictor (independent) variables** .... Go to 5
Investigating relationships

5 a. **Single predictor variable and linear relationship** .... Chapter 8
Correlation and simple linear regression
b. **Multiple predictor variables or curvilinear/complex relationship** ...... Chapter 9
Complex regression analysis

6 a. **A single predictor variable** .... Go to 7
b. **Multiple predictor variables** .... Go to 8

7 a. **Predictor variable with two levels (two groups)** .... Chapter 6
Simple hypothesis testing, t-tests
b. **Predictor variable with multiple levels (more than two groups)** ...... Chapter 10
Single factor Analysis of Variance (ANOVA)

8 a. **All predictor variables categorical** .... Go to 9
Multifactor and complex Analysis of Variance ANOVA
b. **Continuous and categorical predictor variables** .... Chapter 15
Analysis of Covariance

9 a. **All levels within each predictor variable fully replicated** .... Chapter 12
Multifactor Analysis of Variance ANOVA

b. All predictor variables blocked within a random factor .... Chapter 13
Unreplicated factorial designs – randomized block and simple repeated measures

c. Within and between blocking factors .... Chapter 14
Partly nested designs – split-plot and complex repeated measures

10 a. Binary response variable (presence/absence, alive/dead, yes/no etc) .. Chapter 17
Logistic regression

b. Response variable frequencies .... Chapters 16&17
Counts from classifying units according to one or more categories
Chi-squared test, contingency tables, log-linear modeling.

1

Introduction to R

1.1 Why R?

R is a language and programming environment for statistical analysis and graphics that is distributed under the GNU General Public License[a] and is largely modeled on the powerful proprietary S/Splus (from ATT Bell Laboratories). R provides a flexible and powerful environment consisting of a core set of integrated tools for classical data manipulation, analysis and display. An ever expanding library of additional modules (packages) provide extended functionality for more specialized procedures. Initially written by Ross Ihaka and Robert Gentleman of the Department of Statistics at the University of Auckland (NZ), the R project is currently maintained by an international cooperative (the 'R Core Team') who oversee and adjudicate on the continual development of the project.

The GNU General Public License and flexible language ensure that the R project has the potential to rapidly support any newly conceived procedures. Consequently, R has (and will continue to), evolved rapidly as statisticians from a wide range of scientific backgrounds recognize the power of universally adopted tools and offer their contributions. Moreover, the universality, freedom and extensibility of R has resulted in its rapid expansion in popularity among biological teaching and research professionals and students alike. Source code and binaries (executable files) are also freely available for the Windows, Mac[b] and Unix/Linux families of operating systems from the Comprehensive R Archive Network (CRAN) site at 'http://cran.r-project.org/'. Not surprisingly then, R is quickly becoming the universal statistical language of the international scientific community, and correspondingly, R proficiency skills are becoming increasingly more valuable.

As R is a copy of S, documentation on either are generally relevant (however, it should be noted that there are a number of differences between the two dialects). In particular, Everitt (1994), Pinheiro and Bates (2000) and Venables and Ripley (2002) are excellent S/S-PLUS references whilst Dalgaard (2002), Fox (2002), Maindonald and Braun (2003), Crawley (2002, 2007), Murrell (2005) and Zuur et al. (2009) are excellent R reference texts for biologists. In addition, there is an extensive amount of

[a] Under the GNU General Public License, anyone is free to use, modify and (re)distribute the software.
[b] Support for the Mac OS Classic ended with R 1.7.1.

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

information available on-line at the CRAN site ('http://r-project.org') and in the help files packaged with the distributions and extension packages.

1.2 Installing R

At the time of writing the current version of R is R.2.9.1. Since Windows, Unix/Linux and Mac OS systems differ extensively in areas of user privileges and software management, different installation files and procedures are required for each of the systems. Irrespective of the system, the latest version of an installation binary or the source code can be downloaded from the CRAN. Binary installation files or compressed source code for version R.2.9.1 can also be found on the accompanying website www.wiley.com/go/logan/r.

1.2.1 Windows

Obtain a copy of the R installation binary file (e.g. R-2.9.1-win32.exe). Run this self-extracting and self-installation file as Administrator (right click on the executable and select Run as Administrator) if you know the appropriate password. This will install R in the default (and best) location. If you do not know the Administrator password for the computer (or do not have adequate privileges), R will be installed within your user account. The installer will guide you through the installation, but for most purposes the default options are adequate. During the installation process, startup menu and desktop icon links to *RGui.exe* (the main R interface) will be automatically created.

1.2.2 Unix/Linux

Obtain a copy of the compressed R source code (e.g. R.2.9.1.tgz) and unpack it to an appropriate location (typically `/usr/local`) with:

```
tar xvfz R.2.9.1.tgz
```

Note: if you do not have root status, or you wish to have R installed in an alternative location for some reason, you are referred to the R-admin.html help file included in the packed source. From the top directory of the unpacked source, issue the following commands to configure, build and check the system:

```
./configure
make
make check
```

If there are no failures, the manuals can be built in dvi, pdf and/or info formats using the following commands:

```
make dvi
make pdf
make info
```

Install the R tree (and manuals) on your system using the following commands:

```
make install
make install-dvi
make install-pdf
make install-info
```

A symbolic link (`R`) will be added to `/usr/local/bin` and thus R can be run by entering `R` at a terminal command prompt.

1.2.3 MacOSX

Obtain a copy of the R disk image file (e.g. R.2.9.1.tgz). Start the installation by running (double-clicking on) the disk image file. This will bring up a new Finder window containing the installation package. Run the installation package (double-click) and if you are not already logged in as Administrator, you will be prompted for the administrator password. The installer will then guide you through the installation, but for most purposes the default options are adequate.

1.3 The R environment

Let's begin with a few important definitions:

Object R is an object oriented language and everything in R is an object. For example, a single number is an object, a variable is an object, output is an object, a data set is an object that is itself a collection of objects, etc.

Vector A collection of one or more *objects* of the same type (e.g. all numbers or all characters etc).

Function A set of instructions carried out on one or more objects. Functions are typically used to perform specific and common tasks that would otherwise require many instructions. For example, the *function* `mean()` is used to calculate the arithmetic mean of the values in a given *numeric vector*. Functions consist of a name followed by parentheses containing either a set of *parameters* (expressed as *arguments*) or left empty.

Parameter The kind of information that can be passed to a function. For example, the `mean()` *function* declairs a single required parameter (a valid object for which the mean is to be calculated is a compulsary) as well as a number of optional parameters that facilitate finer control over the function.

Argument The specific information passed to a function to determine how the function should perform its task. Arguments are expressions (in the form of `name=value`) given between the parentheses that follow the name of the function. For example, the `mean()` function requires at least one argument - either the name of an object that contains the values from which the mean is to be generated or a vector of values.

Operator Is a symbol that has a pre-defined meaning. Familiar operators include `+ - *` and `/`, which respectively perform addition, subtraction, multiplication and division. The `=` operator is used within functions to assign values to arguments. Logical operators are queries returning either a TRUE or FALSE response. Familiar logical operators include < ('is the left hand side less than the right?'), > ('greater than?'), <= ('less than or equal?') and >= ('greater than or equal?'), while less familiar logical operators include == (which translates to 'does the entry on the left hand side of the == operator equal the entry on the right hand side?'), `!=` (logical NOT – 'is the left hand side not equal to the right?'), `&&` (logical AND – 'are both left hand and right hand conditions TRUE?') and `||` (logical OR – 'is either condition TRUE?').

1.3.1 The console (command line)

The R command prompt (>) is where you interact with R by entering commands (expressions). Commands are evaluated once the **Enter** key has been pressed, however, they can also be separated from one another on a single line by a semicolon character (;). A continuation prompt (+) is used by R to indicate that the command on the preceding line was syntactically incomplete. R ignores all characters on a line that are followed by a hash character (#). These statements or *comments* are commonly used in R literature and scripts for explaining or detailing the surrounding commands.
Enter the following command at the R command prompt (>):

```
> 5 + 1
[1] 6
```

R evaluates the command `5+1` (5 plus 1) and returns the value of an object whose first (and only) element is 6. The `[1]` indicates that this is the first (and in this case only) element in the object returned.

Command history

Each time a command is entered at the R command prompt, the command is also added to a list known as the command history. The up and down arrow keys scroll backward and forward respectively through the session's command history list and place the top most command at the current R command prompt. Scrolling through the command history enables previous commands to be rapidly re-executed, reviewed or modified and executed.

1.4 Object names

All objects have unique names to which they are refered. Names given to any object in R can comprise virtually any sequence of letters and numbers providing that the following rules are adhered to:

- Names must begin with a letter (names beginning with numbers or operators are not permitted)
- Names cannot contain the following characters; `space , - + * / # % & [ ] { } ( ) ~`

Whilst the above rules are necessary, the following naming conventions are also recommended:

- Avoid names that are the names of common predefined functions as this can provide a source of confusion for both you and R. For example, to represent the mean of a head length variable, use something like `MEAN.HEAD.LENGTH` or `MeanHeadLength` rather than `mean`.
- In R, **all commands are case sensitive** and thus `A` and `a` are different and refer to different objects. Almost all inbuilt names in R are lowercase. Therefore, one way to reduce the likelihood of assigning a name that is already in use by an inbuilt object is to only use uppercase names for any objects that you create. This is a convention practiced in this book.
- Names should reflect the content of the object. One of the powerful features of R is that there is virtually no limit to the number of objects (variables, datasets, results, models, etc) that can be in use at a time. However, without careful name management, objects can rapidly become misplaced or ambiguous. Therefore, the name of an object should reflect what it is, and what has happened to it. For example, the name `Log.FISH.WTS` might be given to an object that contains log transformed fish weights.
- Although there are no restrictions on the length of names, shorter names are quicker to type and provide less scope for typographical errors and are therefore recommended (of course within the restrictions of the point above).
- Separate any words in names by a decimal point. For example, the name `HEAD.LENGTH` might be used to represent a numeric vector of head lengths.

Attempts have been made to always adhere to the above naming conventions throughout the rest of the worked examples in this book, so as to provide a more extensive guide to good naming practices.

1.5 Expressions, Assignment and Arithmetic

An **expression** is a command that is entered at the R command prompt, evaluated by R, printed to the current output device (usually the screen), and then discarded. For example:

```
> 2 + 3                          ← an expression
[1] 5                            ← the evaluated output
```

Assignment assigns a name to a new object that may be the result of an evaluated expression or any other object. The assignment operator `<-` is interpreted by R as 'evaluate the expression on the right hand side and assign it the name supplied on the left hand side'[c]. If the object on the left hand side does not already exist, then it is created, otherwise the object's contents are replaced. The contents of the object can be viewed (printed) by entering the name of the object at the command prompt.

```
> VAR1 <- 2 + 3                  ← assign expression to the object VAR1
> VAR1                           ← print the contents of the object VAR1
[1] 5                            ← evaluated output
```

[c] Assignment can also be made left to right using the `->` assignment operator.

A single command may be spread over multiple lines. If either a command is not complete by the end of a line, or a carriage return is entered before R considers that the command syntax is complete, the following line will begin with the prompt + to indicate that the command is incomplete.

```
> VAR2 <-          ← an incomplete assignment/expression
+ 2 + 3            ← assignment/expression completed
> VAR2             ← print the contents of VAR2, the evaluated output
[1] 5
```

When the contents of a vector are numeric (see section 1.10 below), standard arithmetic procedures can be applied.

```
> VAR2 - 1                ← print the contents of VAR2 minus 1
[1] 4
> ANS1 <- VAR1 * VAR2     ← evaluated expression assigned to ANS1
> ANS1                    ← print the contents of ANS1 the evaluated output
[1] 25
```

Objects can be concatenated (joined together) to create objects with multiple entries using the `c()` (concatenation) *function*.

```
> c(1, 2, 6)              ← concatenate 1, 2 and 6
[1] 1 2 6                 ← printed output
> c(VAR1, ANS1)           ← concatenate VAR1 and ANS1 contents
[1] 5 25                  ← printed output
```

In addition to the typical addition, subtraction, multiplication and division operators, there are a number of special operators, the simplest of which are the quotient or integer divide operator (`%/%`) and the remainder or modulus operator (`%%`).

```
> 7/3
[1] 2.333333
> 7%/%3
[1] 2
> 7%%3
[1] 1
```

1.6 R Sessions and workspaces

1.6.1 Cleaning up

So far we have created a number of objects. To view a list of all current objects that have been created:

```
> ls()                    ← list current objects in R environment
[1] "ANS1" "VAR1" "VAR2"
```

The `ls()` *function* is also useful for searching for the name of objects that you created and can't remember:

```
> ls(pat = "VAR")            ← list objects that begin with VAR
[1] "VAR1" "VAR2"
> ls(pat = "A*1")            ← list objects that contain an A and a 1 with
[1] "ANS1" "VAR1"              any number of characters in between.
```

Since objects are easily created (and forgotten about) in R, an R session's workspace can rapidly become cluttered with extraneous and no longer required objects. To avoid this, it is good practice to remove objects as they become obsolete. This is done with the `rm()` *function*.

```
> rm(VAR1, VAR2)             ← remove the VAR1 and VAR2 objects
> rm(list = ls())            ← remove all user defined objects
```

1.6.2 Workspaces

Throughout an R session, all objects (including loaded packages, see section 1.19) that have been added are stored within the R global environment, called the workspace. Occasionally, it is desirable to save the workspace and thus all those objects (vectors, functions, etc) that were in use during a session so that they are automatically available during subsequent sessions. This can be done using the `save.image()` *function*. Note, this will save the workspace to a file called `.RData` in the current working directory (usually the R startup directory, see section 1.6.3), unless a filename (and path) is supplied as an argument to the `save.image()` *function*. A previously saved workspace can be loaded by providing a full path and filename as an argument to the `load()` *function*. Whilst saving a workspace image can sometimes be convenient, it can also contribute greatly to organizational problems associated with large numbers of obsolete or undocumented objects.

1.6.3 Current working directory

By default, files are read and written to the current working directory-the R startup directory (location of the R executable file) unless otherwise specified. To enable read and write operations to take place in other locations, the current working directory can be changed with the `setwd()` *function* which requires a single argument (the full path of the directory[d]). The current working directory can be reviewed using the `getwd()` *function*

```
> setwd("~/Documents/")      ← set the current working directory
> getwd()                    ← review the current working directory
[1] "/home/murray/Documents"
```

[d] Note that R using the Unix/Linux style directory subdivision markers. That is, R uses the forward slash / in path names rather than the regular \ of Windows.

```
> list.files(getwd())
[1] "addressbook.vcf"
[2] "Introduction.rnw"           ← list all in the current working directory
[3] "Introduction.rnw.map"
[4] "Rplots.ps"
[5] "Rscripts.R"
```

1.6.4 Quitting R

To quit R elegantly, use the `q()` *function*. You will be asked whether or not you wish to save the workspace image. If you answer yes (`y`), the current state of your environment or workspace (including all the objects and packages[e] that were added during the session) will be stored within the current working directory.

1.7 Getting help

There are a variety of ways to obtain help on either specific functions or more general procedures within the R environment. Specific information on any inbuilt and add-in objects (such as functions) as well as the R language can be obtained by either providing the name of the object as a character string argument for the `help()` *function* or by using the name of the object as a suffix to a `?` character[f]. As an example, the following two statements both display the R manual page on the `mean()` *function*:

```
> help(mean)
> ?mean
```

Help files are in a standard format such that they all include a description of the object(s), a template of how the object(s) are used, a description of all the arguments and options, more information on any important specific details of the use of the object(s), a list of authors, a list of similar objects and finally a set of examples that illustrate the use of the object(s).

The examples within a manual page can also be run on the R command line using the `example()` *function*. To see an example use of the `mean` *function*:

```
> example(mean)
```

R includes some inbuilt demonstration scripts that showcase the general use of functions on certain topics. The `demo()` *function* provides a user-friendly interface for running these demonstrations. For example, to get an overview of the use of some of the basic graphical procedures in R, run the `graphics demo`:

```
> demo(graphics)
```

[e] Packages provide a flexible means of extending the functionality of R, see section 1.19.
[f] Help on objects within a package is only available when the package is loaded.

Calling the `demo()` *function* without any arguments returns a list of demonstration topics available on your system:

```
> demo()
```

The `apropos()` *function* returns a set of object names from the current search list that match a specific pattern, and is therefore useful for recalling the name of functions. For example, the following expression returns the name of all currently available objects that contain the characters `"mea"` in their names.

```
> apropos("mea")
 [1] "colMeans"           "influence.measures"
 [3] "kmeans"             "mean"
 [5] "mean.data.frame"    "mean.Date"
 [7] "mean.default"       "mean.difftime"
 [9] "mean.POSIXct"       "mean.POSIXlt"
[11] "rowMeans"           "weighted.mean"
```

The `help.search()` and `help.start()` *functions* both provide ways of searching through all the installed R manuals on your system for specific terms. The name of the term or 'keyword' is provided as a character string argument to the `help.search()` *function* which returns a list of relevant manual pages and their brief descriptions.

```
> help.search("mean")
```

The `help.start()` *function* is a more comprehensive and general help system that launches a web browser that displays various local HTML documents containing specific R documentation, a search engine and links to other resources.

There are also numerous books written on the use of R (and/or S/PLUS), see section 1.22 for a list of recent publications.

1.8 Functions

Functions are sets of commands that are conveniently wrapped together such that they can be initiated via a single command that encapsulates all the user inputs to any of the internal commands. Hence, functions provide a friendly way to interact with a set of commands. Most functions require one or more inputs (called *arguments*), and, while a particular function may have a number of arguments, not all need to be specified each time the function is *called*. Consider the `seq()` *function*, which generates a sequence of values (a *vector*) according to the values of the arguments. This function has the following common usage structures:

```
> seq(from, to)
```

← *a sequence of numbers from* `'from'` *to* `'to'` *incrementing by 1*

```
> seq(from, to, by = )
> seq(from, to, length.out = )
```

← *a sequence of numbers from* `'from'` *to* `'to'` *incrementing by* `'by='`
← *a sequence of* `'length.out'` *numbers from* `'from'` *to* `'to'`

If only the first two arguments are provided (as in the first form above), the result is a sequence of integers from `'from'` to `'to'`. Note that this is equivalent to the sequence generator of the form `'from:to'`. When the arguments are provided unnamed (such as `seq(5,9)`), the order of arguments is assumed to be as provided in the usage structure. Therefore, the following two expressions do **not** yield the same sequences:

```
> seq(5, 9)
> seq(9, 5)
```

Named arguments are used to distinguish between alternative uses of a function. For example, in the expression `seq(2,10,4)`, the `4` could mean either that the sequence should increment by 4 (`by=4`) or that the sequence should consist of 4 numbers (`length.out=4`). Furthermore, when named arguments are provided, the order in which the arguments are included is no longer important. Thus, the following are equivalent:

```
> seq(from = 5, to = 9, by = 2)
> seq(to = 9, by = 2, from = 5)
```

Argument names can also be truncated provided the names are not ambiguous. Therefore, the above examples could be shortened to `seq(f=5, t=9, b=2)`. If a function had the arguments `length` and `letter`, for that particular function, the arguments could be truncated to `len` and `let` respectively.

Many functions also provide default values for some compulsory arguments. The default values represent the 'typical' conditions under which the function is used, and these arguments are only required if they are to be different from the default. For example, the `mean` *function* calculates the arithmetic mean of one or more numbers. In addition to an argument that specifies an object containing numbers (to be averaged), the function has the arguments `trim=0` and `na.rm=FALSE` which respectively indicate what fraction of the data to trim to calculate the trimmed mean and whether or not to remove missing entries before calculation. The expression `mean(X)` is therefore equivalent to `mean(X, trim=0, na.rm=FALSE)`.

1.9 Precedence

The rules of operator precedence are listed (highest to lowest) in Table 1.1. Additionally, expressions within parentheses '`()`' always have precedence. Arguments and expressions within a function are always evaluated before the function. Consider the following set of commands that use the `c()` (concatenation) *function* to generate a

Table 1.1 Precedence and description of operators within R listed from highest to lowest.

Operator	Description
`[ [[`	indexing
`::`	name space
`$`	component
`^`	exponentiation (evaluated right to left)
`- +`	sign (unary)
`:`	sequence
`%special%`	special operators (e.g. `%/%, %%`)
`* \`	multiplication, division
`+ -`	addition and subtraction
`< > <= >= == !=`	ordering and comparison
`!`	logical negation (not)
`& &&`	logical AND
`| ||`	logical OR
`~`	formula
`-> ->>`	assignment (left to right)
`=`	argument assignment (right to left)
`<- <<-`	assignment (right to left)
`?`	help

vector of two numbers (2 and 4) and then use the `rep()` (repeat) *function* to repeat the vector thrice.

```
> X <- c(2, 4)
> rep(X, 3)
[1] 2 4 2 4 2 4
```

Alternatively, by nesting the `c()` *function* within the `rep()` *function*, the same result can be achieved with a single command:

```
> rep(c(2, 4), 3)
[1] 2 4 2 4 2 4
```

1.10 Vectors - variables

The basic data storage unit in R is called a *vector*. A vector is a collection of one or more entries of the same *class* (type). Table 1.2 below defines the four major vector classes and provides simple examples of their use. Vectors are one-dimensional arrays of entries. That is, a vector is a single column (or row) of entries whose length is the number of rows in the column or vice versa. Each entry has a unique index number that is equivalent to a row number that can be used to refer to that particular entry within the vector.

Table 1.2 Object vector classes in R. The *operator* `:` is used to generate a sequence of integers. The *function* called `c()` is short (very short) for concatenate and can be used to generate a vectors. The *operator* `==` evaluates whether the left hand side is equal to the right hand side.

Vector class	Example	
`integer`	`> 2:4`	`#vector of integers from 2 to 4`
(Whole numbers)	`[1] 2 3 4`	
	`> c(1,3,9)`	`#vector of integers`
	`[1] 1 3 9`	
`numeric`	`> c(8.4, 2.1)`	`#vector of real numbers`
(Real numbers)	`[1] 8.4 2.1`	
`character`	`> c('A', 'ABC')`	`#vector of letters`
(Letters)	`[1] "A" "ABC"`	
`logical`	`> c(2:4)==3`	`#evaluate the expression`
(TRUE or FALSE)	`[1] FALSE TRUE FALSE`	`#the printed logical vector`

Biological variables are collections of observations of the same kind (e.g. a temperature variable contains a collection of temperature measurements) and are therefore, appropriately represented by vectors. Continuous biological variables are represented by *numeric vectors*, whereas, categorical variables are best represented by *character vectors*. For example, a *numeric vector* (variable) might represent the air temperature within ten (10) quadrats.

```
> TEMPERATURE <- c(36.1, 30.6, 31, 36.3, 39.9, 6.5,
+     11.2, 12.8, 9.7, 15.9)
> TEMPERATURE
 [1] 36.1 30.6 31.0 36.3 39.9  6.5 11.2 12.8  9.7 15.9
```

1.10.1 Regular or patterned sequences

Inclusive sequences of integers can be generated using the `:` operator

```
> #a sequence from 10 to 18 inclusive
> 10:18
[1] 10 11 12 13 14 15 16 17 18
> #a sequence from 18 to 10 inclusive
> 18:10
[1] 18 17 16 15 14 13 12 11 10
```

The `seq()` *function* is used to generate numeric sequences

```
> #every 4th number from 2 to <= 20
> seq(from=2, to=20, by=4)
[1]  2  6 10 14 18
```

```
> seq(from = 2, to = 20, length = 5)
[1]  2.0  6.5 11.0 15.5 20.0
```

Sequences of repeated entries are supported with the `rep()` *function*.

```
> rep(4, 5)                  #repeat the number 4 five times
[1] 4 4 4 4 4
> rep("no", 4)               #repeat the word 'no' four times
[1] "no" "no" "no" "no"
> rep(c(2, 5), 3)            #repeat the series 2 & 5 three times
[1] 2 5 2 5 2 5
> rep(c(2, 5), c(3, 2))      #repeat the number 2 three times
[1] 2 2 2 5 5                # and then the number 5 twice
```

Note that in the two examples immediately above, there are functions within functions. That is the `c()` *function* is used within the `rep()` *function*. When there are functions within functions, the inner most function is evaluated first. Hence in the above examples, the `c()` *function* is evaluated and expanded first and then the `rep()` function uses the resulting object(s) as an argument.

1.10.2 Character vectors

Names of experimental or sampling units (such as sites, quadrats, individuals...) can be stored into character vectors.

```
> QUADRATS <- c("Q1", "Q2", "Q3", "Q4", "Q5", "Q6",
+     "Q7", "Q8", "Q9", "Q10")
> QUADRATS
 [1] "Q1"  "Q2"  "Q3"  "Q4"  "Q5"  "Q6"  "Q7"  "Q8"  "Q9"
[10] "Q10"
```

A more elegant way to generate the above character vector is to use the `paste()` *function*. This function converts multiple vectors into character vectors before combining the elements of each vector together into a single character vector. A `sep=` argument is used to indicate a separation character (or set of characters) to appear between combined vector elements:

```
> QUADRATS <- paste("Q", 1:10, sep = "")
> QUADRATS
 [1] "Q1"  "Q2"  "Q3"  "Q4"  "Q5"  "Q6"  "Q7"  "Q8"  "Q9"
[10] "Q10"

> paste("Quad", 1:10, sep = ".")
 [1] "Quad.1"  "Quad.2"  "Quad.3"  "Quad.4"  "Quad.5"
 [6] "Quad.6"  "Quad.7"  "Quad.8"  "Quad.9"  "Quad.10"
```

Such a character vector can then be used to name the elements of a vector. For example, we could use the `names()` *function* to name the elements of the `TEMPERATURE` vector according to their quadrat labels:

```
> names(TEMPERATURE) <- QUADRATS
> TEMPERATURE
  Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9  Q10
36.1 30.6 31.0 36.3 39.9  6.5 11.2 12.8  9.7 15.9
```

The `paste()` *function* can also be used in conjunction with other functions to generate lists of labels. For example, we could combine a vector in which the letters A, B, C, D and E (generated with the `LETTERS` *constant*) are each repeated twice consecutively (using the `rep()` *function*) with a vector that contains a 1 and a 2 to produce a character vector that labels sites in which the quadrats may have occurred.

```
> SITE <- paste(rep(LETTERS[1:5], each = 2), 1:2,
+     sep = "")
> SITE
 [1] "A1" "A2" "B1" "B2" "C1" "C2" "D1" "D2" "E1" "E2"
```

The `substr()` *function* is used to extract parts of string (set of characters) entries within character vectors and thus is useful for making truncated labels (particularly for graphical summaries). For example, if we had a character vector containing the names of the Australian capital cities and required abbreviations (first 3 characters) for graph labels:

```
> AUST <- c("Adelaide", "Brisbane", "Canberra",
+     "Darwin", "Hobart", "Melbourne", "Perth",
+     "Sydney")

> substr(AUST, 1, 3)
[1] "Ade" "Bri" "Can" "Dar" "Hob" "Mel" "Per" "Syd"
```

Alternatively, we could use the `abbreviate()` *function*.

```
> abbreviate(AUST, minlength = 3)
 Adelaide  Brisbane  Canberra    Darwin    Hobart Melbourne
    "Adl"     "Brs"     "Cnb"     "Drw"     "Hbr"     "Mlb"
    Perth    Sydney
    "Prt"     "Syd"
```

Categorical variables with discrete levels can be represented by *character vectors*. For example, a *character vector* might represent whether or not each of the quadrats (from which the above temperatures were measured) were shaded. The first entry in each vector (the numerical temperature vector and the categorical shade vector), corresponds to the first quadrat measured, and so on such that both vectors (variables) are of the same length.

```
> SHADE <- c("no", "no", "no", "no", "no", "full",
+     "full", "full", "full", "full")
> SHADE
 [1] "no"   "no"   "no"   "no"   "no"   "full" "full" "full"
 [9] "full" "full"
```

1.10.3 Factors

To properly accommodate factorial (categorical) variables, R has an additional class of vector called a *factor* which stores the vector along with a list of the levels of the factorial variable. The `factor()` *function* converts a vector into a factor vector.

```
> SHADE <- factor(SHADE)
> SHADE
 [1] no   no   no   no   no   full full full full full
Levels: full no
```

Note the differences between the output of the factor vector and the previous character vector. Firstly, the absence of quotation marks indicate that the vector is no longer a character vector. Internally, the factor vector (`SHADE`) is actually a numeric variable containing only 1's and 2's and in which 1 is defined as the level 'full' and 2 is defined as the level 'no' (levels of a factor are defined alphabetically by default). Hence, when printed, each entry is represented by a label and the levels contained in the factor are listed below.

There are a number of more convenient ways to generate factors in R. Combinations of the `rep()` *function* and concatenation (`c()`) *function* can be used in a variety of ways to produce identical results:

```
> SHADE <- factor(c(rep("no", 5), rep("full", 5)))
> SHADE <- factor(rep(c("no", "full"), c(5, 5)))
> SHADE <- factor(rep(c("no", "full"), each = 5))
> SHADE
 [1] no   no   no   no   no   full full full full full
Levels: full no
```

Another convenient method of generating a factor when each level of the factor has an equal number of entries (replicates) is to use the `gl()` *function*. The `gl()` *function* requires the number of factor levels, the number of consecutive replicates per factor level, the total length of the factor, and a list of factor level labels, as arguments.

```
#generate a factor with the levels 'no' and 'full', each repeated
times in a row

> SHADE <- gl(2, 5, 10, c("no", "full"))
> SHADE
 [1] no   no   no   no   no   full full full full full
Levels: no full
```

```
> SHADE <- gl(2, 1, 10, c("no", "full"))
> SHADE
 [1] no   full no   full no   full no   full no   full
Levels: no full
```

Notice that by default, the `factor()` *function* arranges the factor levels in alphabetical order, whereas the `gl()` *function* orders the factor levels in the order in which they are included in the expression. Issues relating to the ordering of factor levels will be covered in section 2.6.1.

1.11 Matrices, lists and data frames

1.11.1 Matrices

A vector has only a single dimension – it has length. However, a vector can be converted into a matrix (2 dimensional array), whereupon it will display height and width. For example, we could convert the `TEMPERATURE` vector into a matrix by specifying the number of rows (or columns) within the `matrix()` *function*:

```
> matrix(TEMPERATURE, nrow = 5)
     [,1] [,2]
[1,] 36.1  6.5
[2,] 30.6 11.2
[3,] 31.0 12.8
[4,] 36.3  9.7
[5,] 39.9 15.9
```

By default, the matrix is filled by columns. The optional argument `byrow=T`, causes filling by rows instead.

Matrices can also be used to represent the binding of two or more vectors of equal length (and class[g]). For example, we may have the X and Y coordinates for five quadrats within a grid. Vectors are combined into a single matrix using the `cbind()` (combine by columns) or `rbind()` (combine by rows) *functions*:

```
> X <- c(16.92, 24.03, 7.61, 15.49, 11.77)
> Y <- c(8.37, 12.93, 16.65, 12.2, 13.12)
> XY <- cbind(X, Y)
> XY
         X     Y
[1,] 16.92  8.37
[2,] 24.03 12.93
[3,]  7.61 16.65
[4,] 15.49 12.20
[5,] 11.77 13.12
```

[g] when vectors of different types are combined, they are all be converted into a suitable common type.

```
> rbind(X, Y)
   [,1]  [,2]  [,3]  [,4]  [,5]
X 16.92 24.03  7.61 15.49 11.77
Y  8.37 12.93 16.65 12.20 13.12
```

Row and column names can be set (and viewed) using the `rownames()` and `colnames()` *functions*:

```
> colnames(XY)
[1] "X" "Y"
> rownames(XY) <- LETTERS[1:5]
> XY
      X     Y
A 16.92  8.37
B 24.03 12.93
C  7.61 16.65
D 15.49 12.20
E 11.77 13.12
```

The object, `LETTERS`, is a 26 character vector inbuilt into R that contains the uppercase letters of the English alphabet. Similarly, `letters`, contains the equivalent lowercase letters.

1.11.2 Lists

Whilst matrices store vectors of the same type (class) and length, *lists* are used to store collections of objects that can be of differing lengths and types. Lists are constructed using the `list()` *function*. For example, we have previously created a number of isolated vectors (temperature, shade and names and coordinates of sites) that may actually represent data or information from a single experiment. These objects can be grouped together such that they all become *components* of a *list* object:

```
> EXPERIMENT <- list(SITE = SITE, COORDINATES = paste(X,
+     Y, sep = ","), TEMPERATURE = TEMPERATURE,
+     SHADE = SHADE)
> EXPERIMENT
$SITE
 [1] "A1" "A2" "B1" "B2" "C1" "C2" "D1" "D2" "E1" "E2"

$COORDINATES
[1] "16.92,8.37"  "24.03,12.93" "7.61,16.65"  "15.49,12.2"
[5] "11.77,13.12"

$TEMPERATURE
  Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9  Q10
36.1 30.6 31.0 36.3 39.9  6.5 11.2 12.8  9.7 15.9
```

```
$SHADE
 [1] no   full no   full no   full no   full no   full
Levels: no full
```

Note that this list consists of four components made up of two character vectors (`SITE` and `COORDINATES`: a vector of XY coordinates for sites A, B, C, D and E), a numeric vector (`TEMPERATURE`) and a factor (`SHADE`). Note also that while three of the components have a length of 10, the `COORDINATES` component has only five.

1.11.3 Data frames - data sets

Rarely are single biological variables collected in isolation. Rather, data are usually collected in sets of variables reflecting investigations of patterns between and/or among the different variables. Consequently, data sets are best organized into matricies of variables (*vectors*) all of the same lengths yet not necessarily of the same type. Hence, neither lists nor matrices represent natural storages for data sets. This is the role of *data frames* which are used to store a list of vectors of the same length (yet potentially different types) in a rectangular matrix.

Data frames are generated by combining multiple vectors together such that each vector becomes a separate column in the data frame. In this way, a data frame is similar to a matrix in which each column can represent a different vector type. For a data frame to faithfully represent a data set, the sequence in which observations appear in the vectors must be the same for each vector, and each vector should have the same number of observations. For example, the first, second, third...etc entries in each vector must represent respectively, the observations collected from the first, second, third...etc sampling units.

Since the focus of this book is in the exploration, analysis and summary of data sets, and data sets are accommodated in R by data frames, the generation, importation/exportation, manipulation and management of data frames receives extensive coverage in chapter 2.

1.12 Object information and conversion

1.12.1 Object information

Everything in R is an object and all objects are of a certain type or *class*. The class of an object can be examined using the `class()` *function*. For example:

```
> class(TEMPERATURE)
[1] "numeric"
```

There is also a family of *functions* prefixed with `is.` that evaluate whether or not an object is of a particular class (or type) or not. Table 1.3 lists the common object query functions. All object query functions return a *logical vector*. Enter `methods(is)` for a more comprehensive list.

Table 1.3 Common object query functions and their corresponding return values.

Function	Returns TRUE:
`is.numeric(x)`	if **all** elements of x are numeric or integer (`x <-c(1,-3.5)`)
`is.null(x)`	if x is NULL (the object has no length) (`x <-NULL`)
`is.logical(x)`	if **all** elements of x are logical (`x <- c(TRUE,FALSE)`)
`is.character(x)`	if **all** elements of x are character strings (`x <- c(,A,,,Quad,)`)
`is.vector(x)`	if the object x is a vector (a single dimension). Returns `FALSE` if object has any attributes other than `names`
`is.factor(x)`	if the object x is a factor
`is.matrix(x)`	if the object x is a matrix (2 dimensions but not a data frame)
`is.list(x)`	if the object x is a list
`is.data.frame(x)`	if the object x is a data frame
`is.na(x)`	for **each** missing (`NA`) element in x (`x <- c(NA,2)`)
`!`	('not') character as a prefix converts the above functions into '`is.not.`'

Many R objects also have a set of *attributes*, the number and type of which are specific to each class of object. For example, a matrix object has a specific number of dimensions as well as row and column names. The attributes of an object can be viewed using the `attributes()` *function*:

```
> attributes(XY)
$dim
[1] 5 2

$dimnames
$dimnames[[1]]
[1] "A" "B" "C" "D" "E"

$dimnames[[2]]
[1] "X" "Y"
```

Similarly, the `attr()` *function* can be used to view and set individual attributes of an object, by specifying the name of the object and the name of the attribute (as a character string) as arguments. For example:

```
> attr(XY, "dim")
[1] 5 2
> attr(XY, "description") <- "coordinates of quadrats"
> XY
      X     Y
A 16.92  8.37
B 24.03 12.93
```

```
C  7.61 16.65
D 15.49 12.20
E 11.77 13.12
attr(,"description")
[1] "coordinates of quadrats"
```

Note that in the above example, the attribute `"description"` is not a inbuilt attribute of a matrix. When a new attribute is set, this attribute is displayed along with the object. This provides a useful way of attaching a description to an object, thereby reducing the risks of the object becoming unfamiliar.

1.12.2 Object conversion

Objects can be converted or coerced into other objects using a family of *functions* with a `as.` prefix. Note that there are some obvious restrictions on these conversions as most objects cannot be completely accommodated by all other object types, and therefore some information (such as certain attributes) may be lost or modified during the conversion. Objects and elements that cannot be successfully coerced are returned as `NA`. Table 1.4 lists the common object coercion functions. Use `methods(as)` for a more comprehensive list.

Table 1.4 Common object coercion functions and their corresponding return values.

Function	Converts object to
`as.numeric(x)`	a numeric vector ('integer' or 'real'). Factors converted to integers.
`as.null(x)`	a `NULL`
`as.logical(x)`	a logical vector. Values of `>1` converted to `TRUE`, otherwise `FALSE`
`as.character(x)`	a character vector
`as.vector(x)`	a vector. All attributes (including `names`) are removed.
`as.factor(x)`	a factor. This is an abbreviated version of `factor`
`as.matrix(x)`	a matrix. Any non-numeric elements result in all matrix elements being converted to character strings
`as.list(x)`	a list
`as.data.frame(x)`	a data frame. Matrix columns and list columns are converted into a separate vectors of the data frame, and character vectors are converted into factors. All previous attributes are removed

1.13 Indexing vectors, matrices and lists

This section makes use of a number of objects created in earlier sections. Importantly, the `TEMPERATURE` object is a named vector and thus output will differ slightly from unnamed vectors in that returned elements are headed by their row names.

1.13.1 Vector indexing

It is possible to print or refer to a subset of a vector by appending an *index vector* (enclosed in square brackets, `[]`), to the vector name. There are four common forms of vector indexing used to extract a sub-set of vectors:

(i) **Vector of positive integers**. A set of integers that indicate which elements of the vector are to be selected. Selected elements are concatenated in the specified order.

- Select the n^{th} element

```
> TEMPERATURE[2]
  Q2
30.6
```

- Select elements n through m

```
> TEMPERATURE[2:5]
  Q2   Q3   Q4   Q5
30.6 31.0 36.3 39.9
```

- Select a specific set of elements

```
> TEMPERATURE[c(1, 5, 6, 9)]
  Q1   Q5   Q6   Q9
36.1 39.9  6.5  9.7
```

(ii) **Vector of negative integers**. A set of integers that indicate which elements of the vector are to be excluded from concatenation.

- Select all but the n^{th} element

```
> TEMPERATURE[-2]
  Q1   Q3   Q4   Q5   Q6   Q7   Q8   Q9  Q10
36.1 31.0 36.3 39.9  6.5 11.2 12.8  9.7 15.9
```

(iii) **Vector of character strings**. This form of vector indexing is only possible for vectors whose elements have been named. A vector of element names can be used to select elements for concatenation.

- Select the named element

```
> TEMPERATURE["Q1"]
  Q1
36.1
```

- Select the names elements

```
> TEMPERATURE[c("Q1", "Q4")]
  Q1   Q4
36.1 36.3
```

(iv) **Vector of logical values**. The vector of logical values must be the same length as the vector being sub-setted and usually are the result of an evaluated condition. Logical values of `T` (`TRUE`) and `F` indicate respectively to include and exclude corresponding elements of the main vector from concatenation.

– Select elements for which the logical condition is true

```
> TEMPERATURE[TEMPERATURE < 15]
  Q6   Q7   Q8   Q9
 6.5 11.2 12.8  9.7
> TEMPERATURE[SHADE == "no"]
  Q1   Q3   Q5   Q7   Q9
36.1 31.0 39.9 11.2  9.7
```

– Select elements for which multiple logical conditions are true

```
> TEMPERATURE[TEMPERATURE < 34 & SHADE == "no"]
  Q3   Q7   Q9
31.0 11.2  9.7
```

– Select elements for which one or other logical conditions are true

```
> TEMPERATURE[TEMPERATURE < 10 | SHADE == "no"]
  Q1   Q3   Q5   Q6   Q7   Q9
36.1 31.0 39.9  6.5 11.2  9.7
```

1.13.2 Matrix indexing

Like vectors, matrices can be indexed from vectors of positive integers, negative integers, character strings and logical values. However, whereas vectors have only a single dimension (length) (thus enabling each element to be indexed by a single number), matrices have two dimensions (height and width) and, therefore, require a set of two numbers for indexing. Consequently, matrix indexing takes on the form of `[row.indices, col.indices]`, where `row.indices` and `col.indices` respectively represent sequences of row and column indices of the form described for vectors in section 1.13.1.

Before proceeding, re-examine the `XY` matrix generated in section 1.11.1:

```
> XY
      X     Y
A 16.92  8.37
B 24.03 12.93
C  7.61 16.65
D 15.49 12.20
E 11.77 13.12
attr(,"description")
[1] "coordinates of quadrats"
```

The following examples will illustrate the variety of matrix indexing possibilities:

```
> XY[3, 2]                    # select the element at row 3,
[1] 16.65                       column 2

> XY[3, ]                     # select the entire 3rd row
    X     Y
 7.61 16.65

> XY[, 2]                     # select the entire 2nd column
    A     B     C     D     E
 8.37 12.93 16.65 12.20 13.12

> XY[, -2]                    # select all columns except the
    A     B     C     D     E   2nd
16.92 24.03  7.61 15.49 11.77

> XY["A", 1:2]                #select columns 1 through 2 for
    X     Y                     row A
16.92  8.37

> XY[, "X"]                   #select the column named 'X'
    A     B     C     D     E
16.92 24.03  7.61 15.49 11.77

> XY[XY[, "X"] > 12, ]        #select all rows for which the
      X     Y                   value of the column X is
A 16.92  8.37                   greater than 12
B 24.03 12.93
D 15.49 12.20
```

1.13.3 List indexing

Lists consist of collections of objects that need not be of the same size or type. The objects within a list are indexed by appending an *index vector* (enclosed in double square brackets, `[[]]`), to the list name. A single object within a list can also be referred to by appending a string character (`$`) followed by the name of the object to the list names (e.g. `list$object`). The elements of objects within a list are indexed according to the object type. *Vector indices* to objects within other objects (lists) are placed within their own square brackets outside the list square brackets:

Recall the `EXPERIMENT` list generated in section 1.11.2

```
> EXPERIMENT
$SITE
 [1] "A1" "A2" "B1" "B2" "C1" "C2" "D1" "D2" "E1" "E2"
```

```
$COORDINATES
[1] "16.92,8.37"  "24.03,12.93" "7.61,16.65"  "15.49,12.2"
[5] "11.77,13.12"

$TEMPERATURE
  Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9  Q10
36.1 30.6 31.0 36.3 39.9  6.5 11.2 12.8  9.7 15.9

$SHADE
 [1] no   full no   full no   full no   full no   full
Levels: no full
```

The following examples illustrate a variety of list indexing possibilities:

```
> #select the first object in the list
> EXPERIMENT[[1]]
 [1] "A1" "A2" "B1" "B2" "C1" "C2" "D1" "D2" "E1" "E2"

> #select the object named 'TEMPERATURE' within the list
> EXPERIMENT[['TEMPERATURE']]
  Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9  Q10
36.1 30.6 31.0 36.3 39.9  6.5 11.2 12.8  9.7 15.9

> #select the first 3 elements of 'TEMPERATURE' within
> #'EXPERIMENT'
> EXPERIMENT[['TEMPERATURE']][1:3]
  Q1   Q2   Q3
36.1 30.6 31.0

> #select only those 'TEMPERATURE' values which correspond
> #to SITE's with a '1' as the second character in their name
> EXPERIMENT$TEMPERATURE[substr(EXPERIMENT$SITE,2,2) == '1']
  Q1   Q3   Q5   Q7   Q9
36.1 31.0 39.9 11.2  9.7
```

1.14 Pattern matching and replacement (character search and replace)

It is often desirable to select a subset of data on the basis of character entries that match more general patterns. Furthermore, the ability to search and replace character strings within a character vector can be very useful.

1.14.1 grep - pattern searching

The `grep()` *function* searches within a vector for matches to a pattern and returns the index of all matching entries.

```
# select only those 'SITE' values that contain an 'A'
> grep("A", EXPERIMENT$SITE)
[1] 1 2
> EXPERIMENT$SITE[grep("A", EXPERIMENT$SITE)]
[1] "A1" "A2"
```

By default, the pattern comprises any valid *regular expression*[h] which provides great pattern searching flexibility.

```
# convert the EXPERIMENT list into a data frame
> EXP <- as.data.frame(EXPERIMENT)
# select only those rows that contain correspond to a 'SITE'
  value of either an A, B or C followed by a '1'
> grep("[A-C]1", EXP$SITE)
[1] 1 3 5
> EXP[grep("[A-C]1", EXP$SITE), ]
   SITE COORDINATES TEMPERATURE SHADE
Q1   A1  16.92,8.37        36.1    no
Q3   B1  7.61,16.65        31.0    no
Q5   C1 11.77,13.12        39.9    no
```

1.14.2 regexpr - position and length of match

Rather than return the indexes of matching entries, the `regexpr()` *function* returns the position of the match within each string as well as the length of the pattern within each string (`-1` values correspond to entries in which the pattern is not found).

```
#recall the AUST character vector that lists the Australian
  capital cities
> AUST
[1] "Adelaide"  "Brisbane"  "Canberra"  "Darwin"
[5] "Hobart"    "Melbourne" "Perth"     "Sydney"
#get the position and length of string of characters containing
  an 'a' and an 'e' separated by any number of characters
> regexpr("a.*e", AUST)
[1]  5  6  2 -1 -1 -1 -1 -1
attr(,"match.length")
[1]  4  3  4 -1 -1 -1 -1 -1
```

[h] A regular expression is a formal computer language consisting of normal printing characters and special *metacharacters* (which represent wildcards and other features) that together provide a concise yet flexible way of matching strings.

1.14.3 gsub - pattern replacement

The `gsub()` *function* replaces all instances[i] of an identified pattern within a character vector with an alternative set of characters.

```
> gsub("no", "Not shaded", EXP$SHADE)
 [1] "Not shaded" "full"       "Not shaded" "full"
 [5] "Not shaded" "full"       "Not shaded" "full"
 [9] "Not shaded" "full"
```

It is also possible to extend the functionality to accomodate perl-compatible regular expressions.

```
#convert all the capital values entries into uppercase identify
  (and store) all words (\\w) convert stored pattern (\\1) to
  uppercase (\\U)
> gsub("(\\w)", "\\U\\1", AUST, perl = TRUE)
[1] "ADELAIDE"  "BRISBANE"  "CANBERRA"  "DARWIN"
[5] "HOBART"    "MELBOURNE" "PERTH"     "SYDNEY"
```

1.15 Data manipulation

1.15.1 Sorting

The `sort()` *function* is used to sort vector entries in increasing (or decreasing) order. Note that the elements of the `TEMPERATURE` vector were earlier named (see section 1.10.2). This assists in the distinction of the following functions, however it does result in slightly different format (each element has a name above it, and the braced index is absent).

```
> sort(TEMPERATURE)
  Q6   Q9   Q7   Q8  Q10   Q2   Q3   Q1   Q4   Q5
 6.5  9.7 11.2 12.8 15.9 30.6 31.0 36.1 36.3 39.9

> sort(TEMPERATURE, decreasing = T)
  Q5   Q4   Q1   Q3   Q2  Q10   Q8   Q7   Q9   Q6
39.9 36.3 36.1 31.0 30.6 15.9 12.8 11.2  9.7  6.5
```

The `order()` *function* is also used to sort vector entries in increasing (or decreasing) order, but rather than return a sorted vector, it returns the position (order) or the sorted entries in the original vector. For example:

```
> order(TEMPERATURE)
 [1]  6  9  7  8 10  2  3  1  4  5
```

[i] The similar `sub()` *function* replaces only the first match of a pattern within a vector.

Indicating that the smallest entry in the `TEMPERATURE` vector was at position (index) 6 and so on.

The `rank()` *function* is used to indicate the ranking of each entry in a vector:

```
> rank(TEMPERATURE)
 Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9 Q10
  8    6    7    9   10    1    3    4    2   5
```

Indicating that the first entry in the `TEMPERATURE` vector was ranked eighth in increasing order. Ranks from decreasing order can be produced by then reversing the returned vector using the `rev()` *function*.

```
> rev(rank(TEMPERATURE))
Q10   Q9   Q8   Q7   Q6   Q5   Q4   Q3   Q2   Q1
  5    2    4    3    1   10    9    7    6    8
```

1.15.2 Formatting data

Rounding

The `ceiling()` *function* rounds vector entries up to the nearest integer

```
> ceiling(TEMPERATURE)
 Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9 Q10
 37   31   31   37   40    7   12   13   10  16
```

The `floor()` *function* rounds vector entries down to the nearest integer

```
> floor(TEMPERATURE)
 Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9 Q10
 36   30   31   36   39    6   11   12    9  15
```

The `trunc()` *function* rounds vector entries to the nearest integer towards '0' (zero)

```
> trunc(seq(-2, 2, by = 0.5))
[1] -2 -1 -1  0  0  0  1  1  2
```

The `round()` *function* rounds vector entries to the nearest numeric with the specified number of decimal places. Digits of 5 are rounded off to the nearest even digit.

```
> round(TEMPERATURE)
 Q1   Q2   Q3   Q4   Q5   Q6   Q7   Q8   Q9 Q10
 36   31   31   36   40    6   11   13   10  16

> round(seq(-2, 2, by = 0.5))
[1] -2 -2 -1  0  0  0  1  2  2
```

```
> round(TEMPERATURE/2.2, 2)
   Q1    Q2    Q3    Q4    Q5    Q6    Q7    Q8    Q9   Q10
16.41 13.91 14.09 16.50 18.14  2.95  5.09  5.82  4.41  7.23

> round(TEMPERATURE, -1)
 Q1  Q2  Q3  Q4  Q5  Q6  Q7  Q8  Q9 Q10
 40  30  30  40  40  10  10  10  10  20
```

Other formating

Occasionally (mainly for graphical displays), it is necessary to be able to adjust the other aspects of the formatting of vector entries. For example, you may wish to have numbers expressed in scientific notation (2.93e-04 rather than 0.000293) or insert commas every 3 digits left of the decimal point. These procedures are supported via the `formatC()` *function*.

```
> seq(pi, pi * 10000, length = 5)
[1]     3.141593  7856.337828 15709.534064 23562.730300
[5] 31415.926536

# scientific notation
> formatC(seq(pi, pi * 10000, length = 5), format = "e",
+     digits = 2)
[1] "3.14e+00" "7.86e+03" "1.57e+04" "2.36e+04" "3.14e+04"

# scientific notation only if it saves space
> formatC(seq(pi, pi * 10000, length = 5), format = "g",
+     digits = 2)
[1] "3.1"     "7.9e+03" "1.6e+04" "2.4e+04" "3.1e+04"

# floating point format with 1000's indicators
> formatC(seq(pi, pi * 10000, length = 5), format = "f",
+     big.mark = ",", digits = 2)
[1] "3.14"      "7,856.34"  "15,709.53" "23,562.73"
[5] "31,415.93"
```

1.16 Functions that perform other functions repeatedly

The `replicate()` *function* repeatedly performs the function specified in the second argument the number of times indicated by the first argument. The important distinction between the `replicate()` *function* and the `rep()` *functions* described in section 1.10.1, is that the former repeatedly performs the function whereas the later performs the function only once and then duplicates the result multiple times. Since most functions produce the same result each time they are performed, for many uses, both functions produce identical results. The one group of functions that do not produce identical results each time, are those involved in random number generation. Hence, the `replicate()` *function* is usually used in conjunction with random number generators (such as `runif()`, which will be described in greater detail in chapter 4) to produce sets of random numbers. Consider first the difference between `rep()` and `replicate()`:

```
> rep(runif(1), 5)
[1] 0.4194366 0.4194366 0.4194366 0.4194366 0.4194366

> replicate(5, runif(1))
[1] 0.467324683 0.727337794 0.797764456 0.007025032
[5] 0.155971928
```

When the function being run within `runif()` itself produces a vector of length > 1, the `runif()` *function* combines each of the vectors together as separate columns in a matrix:

```
> replicate(5, runif(5))
          [,1]      [,2]      [,3]      [,4]        [,5]
[1,] 0.3266058 0.3313832 0.2113326 0.4744742 0.257732622
[2,] 0.5241960 0.9801652 0.6642341 0.5292882 0.799982207
[3,] 0.1894848 0.8300792 0.7178351 0.7262750 0.698298026
[4,] 0.1464055 0.6758495 0.9940731 0.3015559 0.288537242
[5,] 0.5491748 0.4052211 0.9923927 0.4074775 0.002170782
```

1.16.1 Along matrix margins

The `apply()` *function* applies a function to the margins (1=row margins and 2=column margins) of a matrix. For example, we might have a matrix that represents the abundance of three species of moth from three habitat types:

```
> MOTH <- cbind(SpA = c(25, 6, 3), SpB = c(12, 12,
+     3), SpC = c(7, 2, 19))
> rownames(MOTH) <- paste("Habitat", 1:3, sep = "")
> MOTH
         SpA SpB SpC
Habitat1  25  12   7
Habitat2   6  12   2
Habitat3   3   3  19
```

The `apply()` *function* could be used to calculate the column means (mean abundance of each species across habitat types):

```
> apply(MOTH, 2, mean)
      SpA       SpB       SpC
11.333333  9.000000  9.333333
```

1.16.2 By factorial groups

The `tapply()` *function* applies a function to the vector separately for each level of a factor combination. This provides a convenient way to calculate group statistics (pivot tables). For example, if we wanted to calculate the mean `TEMPERATURE` for each level of the `SHADE` factor:

```
> tapply(TEMPERATURE, SHADE, mean)
   no  full
25.58 20.42
```

1.16.3 By objects

The `lapply()` and `sapply()` *functions* apply a function separately to each of the objects in a list and return a list and vector/matrix respectively. For example, to find out the length of each of the objects within the `EXPERIMENT` list:

```
> lapply(EXPERIMENT, length)
$SITE
[1] 10

$COORDINATES
[1] 5

$TEMPERATURE
[1] 10

$SHADE
[1] 10

> sapply(EXPERIMENT, length)
       SITE COORDINATES TEMPERATURE       SHADE
         10           5          10          10
```

1.17 Programming in R

Although the library of built-in and add-on tools available for the R environment is extensive (and continues to grow at an incredible rate), occasionally there is the need to perform a task for which there are no existing functions. Since R is itself a programming language (in fact most of the available functions are written in R), extending its functionality to accommodate additional procedures can be a relatively simple exercise (depending of course, on the complexity of the procedure and your level of R proficiency).

1.17.1 Grouped expressions

Multiple commands can be issued on a single line by separating each command by a semicolon (`;`). When doing so, commands are evaluated in order from left to right:

```
> A <- 1;  B <- 2;  C <- A + B
> C
[1] 3
```

When a series of commands are grouped together between braces (such as {`command1; command2;...`}), the whole group of commands are evaluated as a single expression and the value of the last evaluated command within the group is returned:

```
> D <- {A <- 1; 2 -> B; C <- A + B}
> D
[1] 3
```

Grouped expressions are useful for wrapping up sets of commands that work together to produce a single result and since they are treated as a single expression, they too can be further nested within braces as part of a larger grouped expression.

1.17.2 Conditional execution – `if` and `ifelse`

Conditional execution is when a sequence of tasks is determined by whether a condition is met (`TRUE`) or not (`FALSE`), and is useful when writing code that needs to be able to accommodate more than one set of circumstances. In R, conditional execution has the forms:

```
if(condition) true.task
if(condition) true.task else false.task
ifelse(condition) true.task false.task
```

If `condition` returns a `TRUE`, the statement `true.task` is evaluated, otherwise the `false.task` is evaluated (if provided). If `condition` cannot be coerced into a logical (a yes/no answer), an error will be reported.

To illustrate the use of the `if` conditional execution, imagine that you were writing code to calculate means and you anticipated that you may have to accommodate two different classes of objects (vectors and matrices). I will use the vector `TEMPERATURE` and the matrix `MOTH`:

```
> NEW.OBJECT <- TEMPERATURE
> if (is.vector(NEW.OBJECT)) mean(NEW.OBJECT)
+      else apply(NEW.OBJECT, 2, mean)
[1] 23
```

```
> NEW.OBJECT <- MOTH
> ifelse(is.vector(NEW.OBJECT), mean(NEW.OBJECT),
+     apply(NEW.OBJECT, 2, mean))
[1] 11.33333
```

1.17.3 Repeated execution – looping

Looping enables sets of commands to be performed (executed) repeatedly.

for

A `for` loop iteratively loops through a vector of integers (a counter), each time executing the set of commands, and takes on the general form of:

```
for (counter in sequence) task
```

where `counter` is a loop variable, whose value is incremented according to the integer vector defined by `sequence`. The `task` is a single expression or *grouped expression* (see section 1.17.1) that utilizes the incrementing variable to perform a specific operation on a sequence of objects. For a simple example of a for loop, consider the following snippet that counts to six:

```
> for (i in 1:6) print(i)
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
```

As a more applied example, let's say we wanted to calculate the distances between each pair of sites in the XY matrix generated in section 1.11.1. The distance between any two sites (e.g. `'A'` and `'B'`) could be determined using Pythagoras' theorem ($a^2 + b^2 = c^2$).

```
> sqrt((XY["A", "X"] - XY["B", "X"])^2 + (XY["A",
+     "Y"] - XY["B", "Y"])^2)

# OR equivalently
> sqrt((XY[1, 1] - XY[2, 1])^2 + (XY[1, 2] - XY[2,
+     2])^2)
[1] 8.446638
```

A *for loop* can be used to produce a 5×5 matrix of pairwise distances between each of the sites:

```
# Create empty object
> DISTANCES <- NULL
```

```
> for (i in 1:5) {
+     X.DIST <- (XY[i, 1] - XY[, 1])^2
+     Y.DIST <- (XY[i, 2] - XY[, 2])^2
+     DISTANCES <- cbind(DISTANCES, sqrt(X.DIST +
+         Y.DIST))
+ }
> colnames(DISTANCES) <- rownames(DISTANCES)
> DISTANCES
          A         B         C        D         E
A  0.000000  8.446638 12.459314 4.088251  7.006069
B  8.446638  0.000000 16.836116 8.571143 12.261472
C 12.459314 16.836116  0.000000 9.049691  5.455868
D  4.088251  8.571143  9.049691 0.000000  3.832075
E  7.006069 12.261472  5.455868 3.832075  0.000000
```

while

A `while` loop executes a set of commands repeatedly while a condition is `TRUE` and exits when the condition evaluates to `FALSE`, and takes the general form:

```
> while (condition) task
```

where `task` is a single expression or *grouped expression* (see section 1.17.1) that performs a specific operation as long as `condition` evaluates to `TRUE`.

To illustrate the use of a `while` loop, consider the situation where a procedure needs to generate a temporary object, but you want to be sure that no existing objects are overwritten. A simple solution is to append the object name with a number. A `while` loop can be used to repeatedly assess whether an object name (`TEMP`) already exists in the current R environment (each time incrementing a suffix) and eventually generate a unique name. The first three commands in the following syntax are included purely to generate a couple of existing names and confirm their existence.

```
> TEMP <- NULL
> TEMP1 <- NULL
> ls()
 [1] "A"            "AUST"         "B"            "C"
 [5] "D"            "DISTANCES"    "EXP"          "EXPERIMENT"
 [9] "i"            "MOTH"         "NEW.OBJECT"   "op"
[13] "QUADRATS"     "SHADE"        "SITE"         "TEMP"
[17] "TEMP1"        "TEMPERATURE"  "X"            "X.DIST"
[21] "XY"           "Y"            "Y.DIST"
#object name suffix, initially empty
> j <- NULL
# proposed temporary object
> NAME <- "TEMP"
# iteratively search for a unique name
```

```
> while (exists(Nm <- paste(NAME, j, sep = ""))) {
+     ifelse(is.null(j), j <- 1, j <- j + 1)
+ }
# assign the unique name to a numeric vector
> assign(Nm, c(1, 3, 3))
# Reexamine list of objects, note the new object, TEMP2
> ls()
 [1] "A"          "AUST"        "B"        "C"
 [5] "D"          "DISTANCES"   "EXP"      "EXPERIMENT"
 [9] "i"          "j"           "MOTH"     "NAME"
[13] "NEW.OBJECT" "Nm"          "op"       "QUADRATS"
[17] "SHADE"      "SITE"        "TEMP"     "TEMP1"
[21] "TEMP2"      "TEMPERATURE" "X"        "X.DIST"
[25] "XY"         "Y"           "Y.DIST"
```

The `exists()` *function* assesses whether an object of the given name already exists and `assign()` *function* makes the first argument an object name and assigns it the value of the second argument.

1.17.4 Writing functions

For all but the most trivial cases, lines of R code should be organized into a new *function* which can then be used in the same way as the built in functions. Functions are defined using the `function()` *function*:

```
> name <- function(argument1, argument2, ...) expression
```

The new function (called `name`) will use the arguments (`argument1, argument2, ...`) to evaluate the `expression` (usually *grouped expressions* – see section 1.17.1) and return the result of the evaluated expression. Once defined, the function is called by issuing a statement in the form:

```
> name(argument1, argument2, ...)
```

Functions not only provide a more elegant way to interact with a procedure (as all arguments are provided in one location, and the internal workings are hidden from view), they form a reusable extension of the R environment. As such, there are a couple of general programming conventions that are worth adhering to. Firstly, each function should only perform a single task. If a series of tasks are required, consider writing a number of functions that in turn are called from another function. Secondly, where possible, provide default options, thereby simplifying the use of the function for most regular occasions. Thirdly, user defined functions should be in either upper case or camel case so as to avoid conflicting with functions built into R or one of the many extension packages.

For example, we could extend the functionality of R by writing a function that estimates the standard error of the mean. The standard error of the mean can be estimated using the formula $sd/\sqrt{n-1}$, where sd is the standard deviation of the sample and n is the number of observations.

```
> SEM <- function(x, na.rm = FALSE) {
+     if (na.rm == TRUE)
+         VAR <- x[!is.na(x)]
+     else VAR <- x
+     SD <- sd(VAR)
+     N <- length(VAR)
+     SD/sqrt(N - 1)
+ }
```

The function first assesses whether missing values (values of `'NA'`) should be removed (based on the value of `na.rm` supplied by the function user). If the function is called with `na.rm=TRUE`, the `is.na()` *function* is used to deselect such values, before the standard deviation and length are calculated using the `sd`[j] and `length` *functions*. Finally, the standard error of the mean is calculated and returned. This function could then be used to calculate the standard error of the mean for the `TEMPERATURE` vector:

```
> SEM(TEMPERATURE)
[1] 4.30145
```

1.18 An introduction to the R graphical environment

In addition to providing a highly adaptable statistical environment, R is also a graphical environment in which figures suitable for publication can be generated. The R graphical environment consists of one or more graphical devices along with an extensive library of functions for manipulating objects on these devices. A graphical device is an output stream such as a window, file or printer that is capable of receiving and interpreting graphical/plotting instructions. The exhaustive number of graphical functions can be broadly broken down into three categories:

- **High-level** graphics (plotting) functions are used to generate a new plot on a graphical device, and, unless directed otherwise, accompanying axes, labels and the appropriate (yet basic) points/bars/boxes etc are also automatically generated. When these functions are issued, a graphical device (a window unless otherwise specified) is opened and activated. If the device is already active, the previous plot will be overwritten. Whilst these functions form the basis of all graphics in R, they are rarely used in isolation to produced graphs, as they offer only limited potential for customization.
- **Low-level** graphics functions are used to customize and enhance existing plots by adding more objects and information, such as additional points, lines, words, axes, colors etc.
- **Interactive** graphics functions allow information to be added or extracted interactively from existing plots using the mouse. For example, a label may be added to a plot at the location of the mouse pointer, thereby simplifying the interaction with the graphical device's coordinate system.

[j] The `sd` *function* returns a `'NA'` when a vector containing missing values is encountered.

The R graphical environment also includes a set of graphical parameters that operate over and above these functions to control the settings of the graphical device, such as its dimensions and where a plot is positioned within the device.

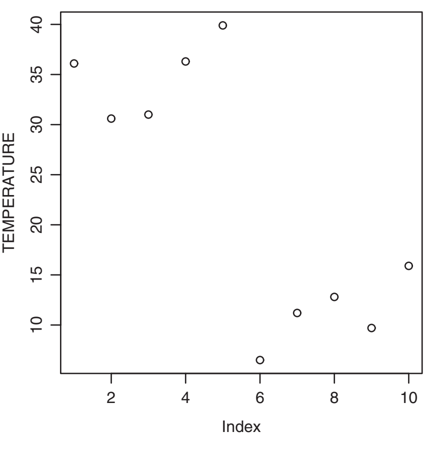
As this section aims to provide only an introductory overview of the R graphical environment, documentation will be limited to just some high level graphics functions. Documentation on low level and interactive graphical functions as well as graphical parameters will be reserved until chapter 5.

1.18.1 The `plot()` function

The `plot()` *function* is actually a generic function that produces different types of plots depending on the class of objects upon which it is acting. The `plot()` *function* evaluates the class of the arguments and then passes the objects on to the plotting function most appropriate for those objects. Notice that the first time a plotting statement is issued, a graphical device (window) is opened and a plot generated. Thereafter, the plots on this graphical device are replaced.

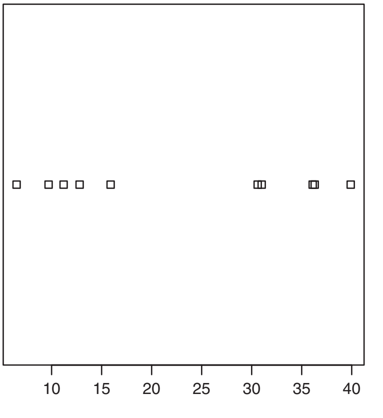
`plot(x)` – if `x` is a *numeric vector* this form of the `plot()` *function* produces a time series plot, a plot of `x` against index numbers.

```
> plot(TEMPERATURE)
```



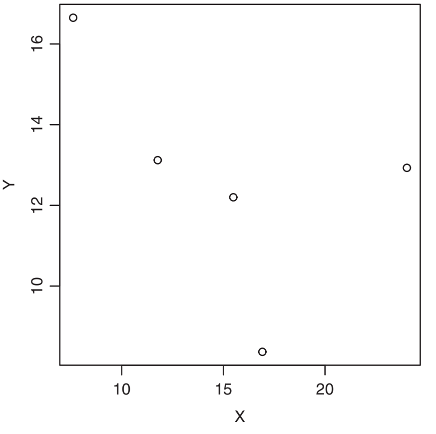
`plot(~x)` – if `x` is a *numeric vector* this form of the `plot()` *function* produces a stripchart for `x`. The same could be achieved with the `stripplot()` *function*. The ~ indicates a formula in which the left side is modeled against the right.

```
> plot(~TEMPERATURE)
```



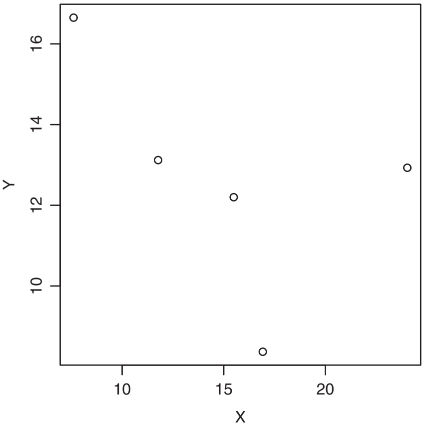
plot(x,y) – if x and y are *numeric vectors* this form of the plot() *function* produces a scatterplot of y against x.

```
> plot(X, Y)
```



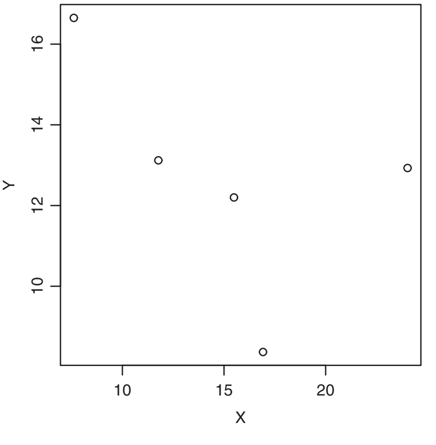
plot(y~expr) – if y is a *numeric vector* and expr is an *expression*, this form of the plot() *function* plots y against each vector in the expression.

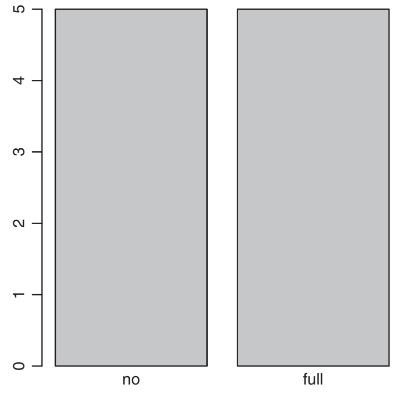
```
> plot(Y ~ X)
```



plot(xy) – if xy is a either a two-column *matrix* or a *list* containing the entries x and y, this form of the plot() *function* produces a plot of y (column 2) against x (column 1). If x is *numeric*, this will be a scatterplot, otherwise it will be a boxplot.

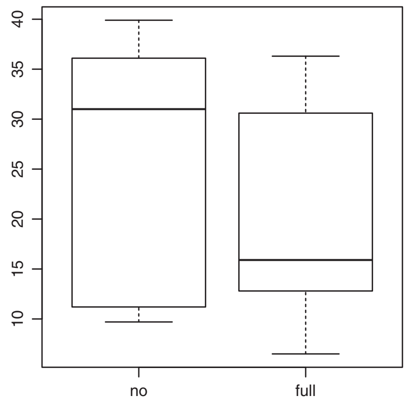
```
> plot(XY)
```





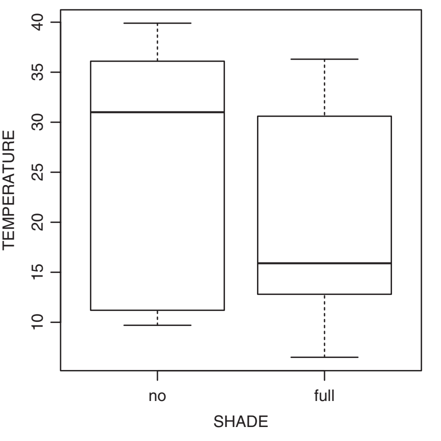
`plot(fact)` – if `fact` is a *factor vector*, this form of the `plot()` *function* produces a bar graph (bar chart) with the height of bars representing the number of entries of each level of the factor. The same could be achieved with the `barplot()` *function*.

```
> plot(SHADE)
```



`plot(fact, dv)` – if `fact` is a *factor vector* and `dv` is a `numeric vector`, this form of the `plot()` *function* produces boxplots of `dv` for each level of `fact`. The same could be achieved with the `boxplot()` *function*.

```
> plot(SHADE, TEMPERATURE)
```



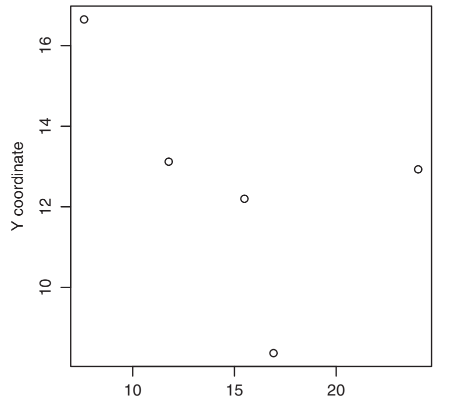
`plot(dv~fact)` – if `fact` is a *factor vector* and `dv` is a `numeric vector`, this form of the `plot()` *function* produces boxplots of `dv` for each level of `fact`.

```
> plot(TEMPERATURE ~ SHADE)
```

There are a limited number of options available to modify the appearance of these plots. Consider the following example:

ylab= and xlab= – these arguments specify the labels used on the vertical and horizontal axes respectively.

```
> plot(X, Y, ylab = "Y coordinate",
+ xlab = "")
```



Other useful high-level plotting functions and options will be illustrated in chapter 5.

1.18.2 Graphical devices

By default, R uses the `window()` graphical device (`X11()` in UNIX/Linux and typically `quartz()` in MacOSX), which provides a representation of graphics on the screen within the R application. However, it is often necessary to produce graphics that can be printed or used within other applications. This is achieved by starting an alternative device (such as a graphics file) driver, redirecting graphical commands to this alternative device, and finally completing the process by closing the alternative device driver. The device driver is responsible for converting the graphical command(s) into a format that is appropriate for that sort of device.

Most installations of R come complete with a number of alternative graphics devices, each of which have their own set of options. A list of graphics devices available on your installation can be obtained by examining the `Devices` help file after issuing the following command[k].

```
> ?Devices
```

Table 1.5 lists some of the major alternative graphics devices and illustrates the common options used for each. Note that in all cases, unless full path names are supplied in the filenames, files are written to the current working directory[l]

The `bitmap()` *function* can also be used to provide a consistent interface to a number of device drivers. The `type=` *argument* can be used to select from a large

[k] A function name preceded by a question mark (?) instruct R to bring up the help file on that function. Help files are introduced in section 1.7.

[l] The current working directory is the location in which files user files are read and written. The working directory can be altered to any available directory on your system and is discussed in section 1.6.3.

Table 1.5 List of useful alternative R graphical devices[a].

Device	Example of use	
jpeg	`> jpeg(file="figure1.jpg",`	
	`+ width=500, height=500,`	dimensions of device (pixels)
	`+ quality=75)`	degree of non-compression
	`> ....`	graphical commands
	`> dev.off()`	close the device
postscript	`> postscript(file="figure1.ps",`	
	`+ width=6, height=6,`	dimensions of graphics region (inches)
	`+ paper="special",`	size of the device, if `paper="special"`
	`+ horiz=F,`	portrait orientation
	`+ family="Helvetica")`	font family to use
	`> ....`	graphical commands
	`> dev.off()`	close the device
pdf	`> pdf(file="figure1.pdf",`	
	`+ width=6, height=6,`	dimensions of graphics region (inches)
	`+ paper="special",`	size of the device, if `paper="special"`
	`+ family="Helvetica")`	font family to use
	`> ....`	graphical commands
	`> dev.off()`	close the device

[a] Not all graphical devices are available on all systems.

range of device types including, `"jpeg"`, `"pcx256"`, `"bmp256"` and `"png256"`. This function has a modest set of arguments (options), the most important of which are the device dimensions (width and height) that are specified in inches.

The `dev2bitmap()` *function* converts a screen graphics device into a graphics file device, thereby providing a simple (yet restrictive) way to save a completed graphic to file without the need to reissue the commands. This function takes the same argument set as the `bitmap()` *function*.

1.18.3 Multiple graphics devices

It is also possible to have multiple devices (of the same or different type) open at once, thereby enabling multiple graphics to be viewed and/or modified concurrently. Each opened graphics device is given a number[m] (starting with 2) and the number reflects the order in which it was created.

To create multiple devices, issue the `dev.set(1)` *function* multiple times. Multiple blank windows will be created, the most recently created of which will be the *active*

[m] Graphical device 1 is a null device – an indicator that there are no currently opened devices.

device (the device in which graphical functions will next act). To view the list of currently open devices, issue the following:

```
> dev.set(1)
null device
          1
> dev.list()
pdf pdf
  2   3
```

This indicates that there are currently two `pdf` graphics devices open in my current session. To list the currently active device:

```
> dev.cur()
pdf
  3
```

To make a graphical device *active* and thus ready to accept the next graphical function, specify the device number as an argument to the `dev.set()` *function*. For example, to make graphical device 2 the *active device*:

```
> dev.set(2)
pdf
  2
```

R returns the type and number of the device as confirmation. The active device can be closed by issuing the `dev.off()` *function* without an argument, whereas a specific device can be closed by specifying the device number as the argument.

A graphics device can be copied from one open device to another (or even to a new device) using the `dev.copy()` *function*. To copy the active device to graphics device 3 (assuming that there is a device numbered 3 and that this is not the active device):

```
> dev.copy(which = 3)
pdf
  3
```

To copy the active device to a new display device (e.g. window, X11 or quartz), specify the device type as an argument:

```
> dev.copy(device = X11)
```

The `dev.copy()` *function* can also be used to copy the active device to other device types, such as graphics files. To do so, the `dev.copy()` *function* is able to receive and forward arguments on to the relevant graphics device driver function (see Table 1.5).

```
> dev.copy(device = jpeg, file = "figure1.jpg",
+     height = 600, width = 600)
```

Note that the jpeg graphics file will not be written until the device has been closed by specifying the device number as an argument to the `dev.off()` *function*.

As an alternative, the `dev.print()` *function* can be used. This operates identically to the `dev.copy()` *function* except that it closes the new device once the graphic has been copied to it. In this way, it is similar to the `dev2bitmap()` *function* and is also useful for sending graphics to a printer.

1.19 Packages

The functionality of the core R system is extended through an ever expanding library of add-on *packages*. As new procedures are developed, they can be supported by specific add-on packages rather than necessitating re-writes of the entire application. Packages define a set of functions designed to perform more specific statistical or graphical tasks. Packages also include help files, example data sets and command scripts to provide information about the full use of the functions. All packages that are made available through the official Comprehensive R Archive Network (CRAN) and its many mirror sites, must comply with very specific regulations set and enforced by the R core development team. Authors of packages are also encouraged not to 'reinvent the wheel', but rather make use of the functionality of other packages where possible. These factors help maximize stability, uniformity and consistency across and between R and all of its packages, thereby ensuring that users of R who have attained a reasonable level of proficiency can rapidly master new packages.

The modularized nature of R also means that only the packages that are necessary to perform the current tasks need to be loaded into memory at any one time. This results in a very 'light-weight', fast statistical and graphical system.

As with procedures for installing and running R itself, procedures for installing packages differ between operating systems and are usually best performed with Administrator (super user) privileges[n].

1.19.1 Manual package management

Obtaining packages

The core R system includes only a subset of the available packages – those packages that have been identified by the R core development team as essential for supporting the common and traditional data exploration, analysis and summary procedures. Additional packages can be obtained from the CRAN web site (http://cran.r-project.org) by following the 'packages' hyperlink and locating the specific package(s). Windows users

[n] Installing with Administrator rites ensures that installations take place in the correct locations (with system wide access). Regular users typically do not have write access to these locations and thus installations with lesser privileges result in packages being installed in the users data directories. In Windows, R can be run as an Administrator by right clicking on the RGui.exe file, folder or shortcut and selecting Run As Administrator from the drop-down menu. Linux and MacOSX users usually know how to act as a super user.

should download the .zip versions, Unix/Linux users download the .tar.gz versions and MacOSX users download .tgz versions.

Note that the philosophy of cross-package reliance to reduce the number of replicated procedures, means that many packages depend on other packages. A package's dependencies are listed in the package description. Ensure that when downloading a package, all other packages that are required have either been previously acquired or are also downloaded. The `library()` *function* without any arguments returns a list of installed and currently available packages on your system. This can be useful for checking potential dependency violations.

Installing packages

Windows

To install packages directly from one of the CRAN mirrors or Bioconductor (Bioinformatics packages) repositories, start by selecting the **Packages** menu from within RGui. For CRAN repositories, select the most local CRAN mirror to you from the list that appears after selecting **Set CRAN mirror...** from the **Packages** menu. Anytime thereafter you can install packages from that mirror by selecting the **Install package(s)...** submenu and then selecting the desired package(s) from the list. To install packages from the Bioconductor packages repository, first alter the repository via the **Select repositories...** submenu.

It is also possible to install packages from pre-downloaded package binaries. Select the **Packages** menu, then the **Install from local zip files..** submenu and locate the downloaded .zip file(s) and click the OK button.

Unix/Linux

Typically only root (or a superuser) can install packages. As root, and from the directory containing the compressed package, enter the following command at a terminal prompt:

```
R CMD INSTALL package_name.tar.gz
```

where `package_name` is the name of the package to be installed.

MacOSX

The MacOSX port of R is able to install packages from source packages using the methods outlined for Unix/Linux systems. However, it is also able to install from pre-packaged binary packages. Whilst the latter is sometimes (for some packages) specific to which OS version is in use (typically only the latest), no other additional compiler tools are required for installation. Hence, installation from binary packages is the simplest method.

To install packages directly from one of the CRAN mirrors or Bioconductor (Bioinformatics packages) repositories, after selecting the **Package Installer** submenu, select the appropriate repository and package type (typically CRAN (binaries)) before pressing **Get List**. Select the package(s) you want installed, check the "Install Dependencies" check-box just below the "Install Selected" button to ensure all the necessary dependencies[o] are also retrieved. You are also able to chose where the packages are installed. There are four radio buttons corresponding to the possible locations. The default is "At System Level (in R framework)". For those with Administrator privileges and password, this is recommended. The others are "At User Level", "In Other Location (Will Be Asked Upon Installation)", and "As Defined by .libpaths()" Finally, click the **Install Selected** button.

To install from downloaded binary packages, select the **Package Installer** submenu from the **Packages & Data** menu. Selecting **Local Source Package** and pressing **Install** will bring up a new Finder window form which you should navigate to and select the downloaded package(s).

Package management within R

The R statistical and graphical environment is equipped with a number of tools to help install and update packages on your system. A list of all the currently installed packages can be obtained by issuing:

```
> installed.packages()
```

```
        Package   LibPath                           Version   Priority Bundle Contains
abind   "abind"   "/usr/local/lib/R/site-library" "1.1-0"   NA       NA     NA
akima   "akima"   "/usr/local/lib/R/site-library" "0.5-2"   NA       NA     NA
alr3    "alr3"    "/usr/local/lib/R/site-library" "1.1.7"   NA       NA     NA
Biobase "Biobase" "/usr/local/lib/R/site-library" "2.4.1"   NA       NA     NA
biology "biology" "/usr/local/lib/R/site-library" "1.0"     NA       NA     NA
bitops  "bitops"  "/usr/local/lib/R/site-library" "1.0-4.1" NA       NA     NA
        Depends                          Imports Suggests                 Enhances OS_type Built
abind   "R (>= 1.5.0)"                   NA      NA                       NA       NA      "2.6.2"
akima   NA                               NA      NA                       NA       NA      "2.9.1"
alr3    NA                               NA      NA                       NA       NA      "2.6.2"
Biobase "R (>= 2.7.0), methods, utils" NA      "tools, tkWidgets, ALL" NA       NA      "2.9.1"
biology "car"                            NA      NA                       NA       NA      "2.9.1"
bitops  NA                               NA      NA                       NA       NA      "2.9.1"
```

Note, I have included only the first six packages to save space. The `installed.packages()` *function* returns the name of the installed packages as well as information about the packages including the version number, dependencies and the version of R on which the package was built.

Packages are often updated in the CRAN repositories. The easiest way to update the installed packages is to use the `update.packages()` *function*

```
> update.packages()
```

[o] In the spirit of modularization, many packages build upon functions contributed by other packages. Consequently, packages that depend on function within other packages list those packages as dependencies. For a given package to install correctly, all its dependencies must already be installed.

You will be prompted for a repository mirror (web locations that provide copies of the official R repositories). You should select the mirror closest to you. The `update.packages()` *function* will then compare your currently installed packages to those on the repositories, download any updated packages and install them on your system. It is also possible to provide a `repos=` *argument* in order to explicitly specify the base URL of the repository you wish to access the package from.

Individual packages can also be installed from a CRAN mirror. The name of the package (without the version codes) is supplied as an argument to the `install.packages()` *function*. As described above, the `repos=` *argument* can be used. The following syntax could be used to install the `car` (Companion to Applied Regression) *package* from the University of Melbourne CRAN mirror.

```
> install.packages(car, repos = "http://cran.ms.unimelb.edu.au")
```

1.19.2 Loading packages

Although packages only need to be installed once, before a package can be used in a session, it needs to be loaded into memory. This ensures that while you may have a very large number of packages installed on your system, only those packages that are actually required to perform the current tasks are taking up valuable resources. A package is loaded by providing the name of the package (without any extensions) as an argument for the `library()` *function*. For example, to load the package `gdata` which provides various data manipulation functions:

```
> library(gdata)
Loading required package: gtools
```

In this case R, informs you that it first loaded a package called `gtools` that `gdata` depends on.

1.20 Working with scripts

One of the advantages of command driven software is that if the commands used to perform certain tasks can be stored, then the tasks can be easily repeated exactly. A collection of one or more commands is called a script. In R, a script is a plain text file with a separate command on each line and can be created and read in any text editor. A script is read into R by providing the full filename (and path if not in the current working directory – see section 1.6.3) of the script file as an argument in the `source()` *function*. By convention, filenames for R scripts end in the extension `.R`. For example:

```
> source("filename.R")
```

A typical script may look like the following:

```
# Temperature.R script
# Written by Murray Logan Aug09
# Sets up temperature and shade variables and calculates mean
# temperature in and out of shade

# Generates a numeric vector called TEMPERATURE
TEMPERATURE <- c(36.1, 30.6, 31.0, 36.3, 39.9, 6.5, 11.2, 12.8,
  9.7, 15.9)
# Define quadrat labels for row names
names(TEMPERATURE) <- paste('Q', 1:10, sep="")

# Generate a factor with the levels 'no' and 'full'
SHADE <- gl(2,5,10,c('no','full'))
# Calculate the mean TEMPERATURE for each level of SHADE
tapply(TEMPERATURE, SHADE, mean)
```

The above script illustrates a couple of important points about R scripts. Firstly, commands within scripts do not begin with a (>) prompt. Expressions can be split over multiple lines (and a '+' prompt is not required) and extra spaces and tabs are completely ignored by R. Finally, the benefits of regular comments throughout a script cannot be overstated. Since scripts are so valuable as a lasting record of analyses, it is of vital importance that each step be thoroughly documented for future reference.

When a script is sourced, each line of the script is parsed[p] (checked for errors), interpreted, and run as if it had been typed directly at the R command prompt. This is an extremely useful feature as it enables complicated and/or lengthy sequences of commands to be stored, modified and reused rapidly as well as acting as a record of data analysis and a repository of analysis techniques. All the commands used in this book are provided as scripts on the accompanying website www.wiley.com/go/logan/r.

1.21 Citing R in publications

The full R citation (and convenient BibTeX entry) is obtained by issuing the following:

```
> citation()
To cite R in publications use:
```

[p] Parsing is a process by which information is first verified before use.

```
  R Development Core Team (2009). R: A language and
  environment for statistical computing. R Foundation
  for Statistical Computing, Vienna, Austria. ISBN
  3-900051-07-0, URL http://www.R-project.org.

A BibTeX entry for LaTeX users is

  @Manual{,
    title = {R: A Language and Environment
             for Statistical Computing},
    author = {{R Development Core Team}},
    organization = {R Foundation for Statistical Computing},
    address = {Vienna, Austria},
    year = {2009},
    note = {{ISBN} 3-900051-07-0},
    url = {http://www.R-project.org},
  }

We have invested a lot of time and effort in creating
R, please cite it when using it for data analysis.
See also 'citation("pkgname")' for citing R packages.
```

1.22 Further reading

Crawley, M. J. (2002). *Statistical computing: an introduction to data analysis using S-PLUS*. John Wiley & Sons, UK.

Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

Dalgaard, P. (2002). *Introductory Statistics with R*. Springer-Verlag, New York.

Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

Ihaka, R., and R. Gentleman. (1996). *R: A Language for Data Analysis and Graphics*. Journal of Computational and Graphical Statistics **5**:299–314.

Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

Pinheiro, J. C., and D. M. Bates. (2000). *Mixed effects models in S and S-PLUS*. Springer-Verlag, New York.

R Development Core Team, (2005). R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org`.

Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS, 4th edn*. Springer-Verlag, New York.

2

Data sets

2.1 Constructing data frames

Data frames are generated by amalgamating vectors of the same length together. To illustrate the translation of a data set (collection of variables) into an R data frame (collection of vectors), a portion of a real data set by Mac Nally (1996) in which the bird communities were investigated from 37 sites across five habitats in southeastern Australia will be used. Although the original data set includes the measured maximum density of 102 bird species from the 37 sites, for simplicity's sake only two bird species (GST: gray shrike thrush, EYR: eastern yellow robin) and the first eight of the sites will be included. The truncated data set, comprises a single factorial (or categorical) variable, two continuous variables, and a set of site (row) names, and is as follows:

Site	HABITAT	GST	EYR
Reedy Lake	Mixed	3.4	0.0
Pearcedale	Gipps.Manna	3.4	9.2
Warneet	Gipps.Manna	8.4	3.8
Cranbourne	Gipps.Manna	3.0	5.0
Lysterfield	Mixed	5.6	5.6
Red Hill	Mixed	8.1	4.1
Devilbend	Mixed	8.3	7.1
Olinda	Mixed	4.6	5.3

Firstly, generate the three variables (excluding the site labels as they are not variables) separately:

```
> HABITAT <- factor(c("Mixed", "Gipps.Manna", "Gipps.Manna",
+     "Gipps.Manna", "Mixed", "Mixed", "Mixed", "Mixed"))
> GST <- c(3.4, 3.4, 8.4, 3, 5.6, 8.1, 8.3, 4.6)
> EYR <- c(0, 9.2, 3.8, 5, 5.6, 4.1, 7.1, 5.3)
```

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

Next, use the names of the vectors as arguments in the `data.frame()` *function* to amalgamate the three separate variables into a single data frame (data set) which we will call MACNALLY (after the author).

```
> MACNALLY <- data.frame(HABITAT, GST, EYR)
> MACNALLY
      HABITAT GST EYR
1       Mixed 3.4 0.0
2 Gipps.Manna 3.4 9.2
3 Gipps.Manna 8.4 3.8
4 Gipps.Manna 3.0 5.0
5       Mixed 5.6 5.6
6       Mixed 8.1 4.1
7       Mixed 8.3 7.1
8       Mixed 4.6 5.3
```

Notice that each vector (variable) becomes a column in the data frame and that each row represents a single sampling unit (in this case, each row represents a different site). By default, the rows are named using numbers corresponding to the number of rows in the data frame. However, these can be altered to reflect the names of the sampling units by assigning a list of alternative names to the `row.names()` *property* of the data frame.

```
> row.names(MACNALLY) <- c("Reedy Lake", "Pearcedale", "Warneet",
+     "Cranbourne", "Lysterfield", "Red Hill", "Devilbend",
+     "Olinda")
> MACNALLY
                HABITAT GST EYR
Reedy Lake        Mixed 3.4 0.0
Pearcedale  Gipps.Manna 3.4 9.2
Warneet     Gipps.Manna 8.4 3.8
Cranbourne  Gipps.Manna 3.0 5.0
Lysterfield       Mixed 5.6 5.6
Red Hill          Mixed 8.1 4.1
Devilbend         Mixed 8.3 7.1
Olinda            Mixed 4.6 5.3
```

2.2 Reviewing a data frame - `fix()`

As with all other objects, a data frame can be viewed by issuing the name of the data frame. A data frame can also be viewed as a simple spreadsheet in a separate window by using the name of the data frame as an argument in the `fix()` *function*. The `fix()` *function* also enables simple editing of the data frame. The arrow keys are used for navigating the spreadsheet and any alterations will be made to the data frame when the window is closed. Try the following:

```
> fix(MACNALLY)
```

Warning - only make alterations to numeric variables, alterations to the entries of factorial variables will not update the factors list of levels and thus the factor will appear to act irrationally in analysis and graphical procedures.

2.3 Importing (reading) data

Generally, statistical systems are not very well suited to tasks of data entry and management. This is the roll of spreadsheets, of which there are many available. Although the functionality of R continues to expand, it is unlikely that R itself will ever duplicate the extensive spreadsheet and database capabilities of other software[a]. R development has roots in the Unix/Linux programming philosophy that dictates that tools should be dedicated to performing specific tasks that they perform very well and rely on other tools to perform other tasks. Consequently, the emphasis of R is, and will continue to be, purely an environment for statistical and graphical procedures. It is expected that other software will be used to generate and maintain data sets.

Unfortunately, data importation into R can be a painful exercise that overshadows the benefits of using R for new users. In part, this is because there are a large number of competing methods that can be used to import data and from a wide variety of sources. This section does not intend to cover all the methods. Rather, it will highlight the simplest and most robust methods of importing data from the most popular sources.

Unless file path names are specified, all data reading functions search for files in the current working directory. Refer to section 1.6.3 for information of reviewing and altering the current working directory.

2.3.1 Import from text file

The easiest form of importation is from a pure text file. Since most software that accepts file input can read plain text files, they can be created in all spreadsheet, database and statistical software packages and are also the default outputs of most data collection devices. In a text file, data are separated or delimited by a specific character, which in turn defines what sort of text file it is. The text file should broadly represent the format of the data frame. That is, variables should be in columns and sampling units in rows. The first row should contain the variable names and if there are row names, these should be in the first column.

[a] However, there are numerous projects in early stages of development that are being designed to offer an interface to R from within major spreadsheet packages.

The following examples illustrate the format of the abbreviated Mac Nally (1996) data set created as both comma delimited (left) and tab delimited (right) files as well as the corresponding `read.table()` commands used to import the files.

Comma delimited text file *.csv

```
HABITAT,GST,EYR
Reedy Lake,Mixed,3.4,0.0
Pearcedale,Gipps.Manna,3.4,9.2
Warneet,Gipps.Manna,8.4,3.8
Cranbourne,Gipps.Manna,3.0,5.0
....
```

```
> MACNALLY <- read.table(
+ 'macnally.csv', header=T,
+ row.names=1, sep=',')
```

Tab delimited text file *.txt

```
             HABITAT       GST   EYR
Reedy Lake   Mixed         3.4   0.0
Pearcedale   Gipps.Manna   3.4   9.2
Warneet      Gipps.Manna   8.4   3.8
Cranbourne   Gipps.Manna   3.0   5.0
....
```

```
> MACNALLY <- read.table(
+ 'macnally.txt', header=T,
+ row.names=1, sep='\t')
```

The first argument to the `read.table()` *function* specifies the name (in quotation marks) of the text file to be imported (and path if not in the current working directory, see section 1.6.3). The `header=T` argument indicates that the first row of the file is a header that defines the variable (vector) names. The `row.names=` argument indicates which column in the data set contains the row names. If there are no row names in the data set, then the `row.names=` argument should be omitted. Finally, the `sep=` *argument* specifies which character is used as the delimiter to separate data entries. The syntax (`'\t'`) indicates a tab character. Field (data) separators are not restricted to commas or tabs, just about any character can be defined as a separator.

2.3.2 Importing from the clipboard

The `read.table()` *function* can also be used to import data (into a data frame) that has been placed on the clipboard[b] by other software, thereby providing a very quick and convenient way of obtaining data from spreadsheets. Simply replace the filename argument with the word `'clipboard'` and indicate a tab field separator (`\t`). For example, to import data placed on the clipboard from Microsoft Excel, use the following syntax;

```
> MACNALLY <- read.table("clipboard", header = T, row.names = 1,
+     sep = "\t")
```

2.3.3 Import from other software

As previously stated, virtually all software packages are able to export data in text file format and usually with a choice of delimiters. However, the `foreign` *package* offers

[b] The clipboard is allocated space in virtual memory from which information can be copied and pasted within and between different applications.

more direct import of native file formats from a range of other popular statistical packages. To illustrate the use of the various relevant functions within the `foreign` *package*, importation of a subset of the Mac Nally (1996) data set from the various formats will be illustrated.

Systat[c]

```
> library(foreign)
> MACNALLY <- read.systat("macnally.syd", to.data.frame = T)
```

Spss

```
> library(foreign)
> MACNALLY <- read.spss("macnally.sav", to.data.frame = T)
```

Minitab

```
> library(foreign)
> MACNALLY <- as.data.frame(read.mtp("macnally.mtp"))
```

Note, the file must be in Minitab Portable Worksheet format.

Sas

```
> library(foreign)
> MACNALLY <- read.xport("macnally")
```

Note, the file must be in the SAS XPORT format. If there is only a single dataset in the XPORT format library, then the `read.xport()` *function* will return a data frame, otherwise it will return a list of data frames.

Excel

Excel is more than just a spreadsheet – it contains macros, formulae, multiple worksheets and formatting. The easiest ways to import data from Excel is either to save the worksheet as a text file (comma or tab delimited) and import the data as a text file (see section 2.3.3), or to copy the data to the clipboard in Excel and import the clipboard data into R (see section 2.3.2).

2.4 Exporting (writing) data

Although plain text files are not the most compact storage formats, they do offer two very important characteristics. Firstly, they can be read by a wide variety of other applications, ensuring that the ability to retrieve the data will continue indefinitely.

[c] Cannot be used to import files produced with the MacOS version of SYSTAT due to incompatible file formats.

Secondly, as they are neither compressed nor encoded, a corruption to one section of the file does not necessarily reduce the ability to correctly read other parts of the file. Hence, this is also an important consideration for the storage of datasets.

The `write.table()` *function* is used to save data frames. Although there are a large number of optional arguments available for controlling the exact format of the output file, typically only a few are required. The following example illustrates the exportation of the Mac Nally (1996) data set as a comma delimited text file.

```
> write.table(MACNALLY, "macnally.csv", quote = F, row.names = T,
+     sep = ",")
```

The first and second arguments specify respectively the name of the data frame and filename (and path if necessary) to be exported. The `quote=F` argument indicates that words and factor entries should not be exported with surrounding double quotation marks. The `row.names=T` argument indicates that the row names in the data frame are also to be exported (they will be the first column in the file). If there are no defined row names in the data frame, alter the argument to `row.names=F`. Finally, specify the field separator for the file (comma specified in above example).

2.5 Saving and loading of R objects

Any object in R (including data frames) can also be saved into a native R workspace image file (*.RData) either individually, or as a collection of objects using the `save()` *function*. For example;

```
> #save just the MACNALLY data frame
> save(MACNALLY, file='macnally.RData')
> #calculate the mean GST
> meanGST <- mean(MACNALLY$GST)
> #display the mean GST
> meanGST
[1] 4.878378
> #save the MACNALLY data frame as well as the mean GST object
> save(MACNALLY, meanGST, file='macnallystats.RData')
```

The saved object(s) can be loaded during subsequent sessions by providing the name of the saved workspace image file as an argument to the `load()` *function*. For example;

```
> load("macnallystats.RData")
```

Similarly, a straight un-encoded text version of an object (including a dataframe) can be saved or added to a text file using the `dump()` *function*.

```
> dump("MACNALLY", file = "macnally")
```

If the file character string is left empty, the text representation of the object will be written to the console. This can then be viewed or copied and pasted into a script file, thereby providing a convenient way to bundle together data sets along with graphical and analysis commands that act on the data sets.

```
> dump("MACNALLY", file = "")
```

Thereafter, the dataset is automatically included when the script is sourced and cannot accidentally become separated from the script.

2.6 Data frame vectors

In generating a data frame from individual vectors (such as above), copies of the original vectors, rather than the actual original vectors themselves are amalgamated. Consequently, while the vectors contained in the data frame contain the same information (entries) as the original vectors, they are completely distinct from the original vectors. So from the examples above, the R workspace will contain the vectors `HABITAT`, `GST` and `EYR` as well as `HABITAT`, `GST` and `EYR` within the `MACNALLY` data frame.

To refer to a vector within a data frame, the name of the vector is proceeded by the name of the data frame and the two names are separated by a $ character. For example, to refer to the `GST` vector of the `MACNALLY` data frame:

```
> MACNALLY$GST
 [1]  3.4  3.4  8.4  3.0  5.6  8.1  8.3  4.6  3.2  4.6  3.7  3.8
[13]  5.4  3.1  3.8  9.6  3.4  5.6  1.7  4.7 14.0  6.0  4.1  6.5
[25]  6.5  1.5  4.7  7.5  3.1  2.7  4.4  3.0  2.1  2.6  3.0  7.1
[37]  4.3
```

Modification made to the original vectors **will not** affect the vectors within a data frame. Therefore, it is important to remember to use the data frame prefix. To avoid confusion, it is generally recommended that following the successful generation of the data frame from individual vectors, the original vectors should be deleted.

```
> rm(HABITAT, GST, EYR)
```

Thereafter, any inadvertent reference to the original vector (`GST`) rather than vector within the data frame (`MACNALLY$GST`) will result in a error informing that the object does not exist.

```
> GST
Error: Object "GST" not found
```

2.6.1 Factor levels

When factors are generated directly using the `factor()` *function* or a data set is imported using one of the above importation methods (which themselves use the `factor()` *function* to convert character vectors into factors), factor levels are automatically arranged alphabetically. For example, examine the levels of the `MACNALLY$HABITAT` factor;

```
> levels(MACNALLY$HABITAT)
[1] "Box-Ironbark"       "Foothills Woodland" "Gipps.Manna"
[4] "Mixed"              "Montane Forest"     "River Red Gum"
```

Although the order of factor levels has no bearing on most statistical procedures and for many applications, alphabetical ordering is as valid as any other arrangement, for some analyses (particularly those involving contrasts, see section 7.3) it is necessary to know the arrangement of factor levels. Furthermore, for graphical summaries of some data, alphabetical factor levels might not represent the natural trends among groups. Consider a dataset that includes a factorial variable with the levels `'high'`, `'medium'` and `'low'`. Presented alphabetically, the levels of the factor would be `'high' 'low' 'medium'`. Those data would probably be more effectively presented in the more natural order of `'high' 'medium' 'low'` or `'low' 'medium' 'high'`.

When creating a factor, the order of factor levels can be specified as a list of labels. For example, consider a factor with the levels `'low'`,`'medium'` and `'high'`:

```
> FACTOR <- gl(3, 2, 6, c("low", "medium", "high"))
> FACTOR
[1] low    low    medium medium high   high
Levels: low medium high
```

The order of existing factor levels can also be altered by redefining a factor:

```
> # examine the default order of levels
> levels(MACNALLY$HABITAT)
[1] "Box-Ironbark"       "Foothills Woodland" "Gipps.Manna"
[4] "Mixed"              "Montane Forest"     "River Red Gum"

> # redefine the order of levels
> MACNALLY$HABITAT<-factor(MACNALLY$HABITAT, levels=c(
+ 'Montane Forest', 'Foothills Woodland','Mixed', 'Gipps.Manna',
+ 'Box-Ironbark','River Red Gum'))

> # examine the new order of levels
> levels(MACNALLY$HABITAT)
[1] "Montane Forest"     "Foothills Woodland" "Mixed"
[4] "Gipps.Manna"        "Box-Ironbark"       "River Red Gum"
```

In addition, some analyses perform different operations on factors that are defined as 'ordered' compared to 'unordered' factors. Regardless of whether you have altered the ordering of factor levels or not, by default all factors are implicitly considered 'unordered' until otherwise defined using the `ordered()` *function*[d].

[d] Alternatively, the argument `ordered=TRUE` can be supplied to the `factor` *function* when defining a vector as a factor.

```
> FACTOR <- ordered(FACTOR)
> FACTOR
[1] low    low    medium medium high   high
Levels: low < medium < high
```

2.7 Manipulating data sets

2.7.1 Subsets of data frames – data frame indexing

Indexing of data frames follows the format of `data frame[rows,columns]`, see Table 2.1.

As an alternative to data frame indexing, the `subset()` *function* can be used:

```
> #extract all the bird densities from sites that have GST values
> #greater than 3
> subset(MACNALLY, GST>3)

                        HABITAT  GST EYR
Reedy Lake                Mixed  3.4 0.0
Pearcedale          Gipps.Manna  3.4 9.2
Warneet             Gipps.Manna  8.4 3.8
Lysterfield               Mixed  5.6 5.6
Red Hill                  Mixed  8.1 4.1
```

Table 2.1 Data frame indexing.

Action	Example indexing syntax
Indexing by rows (sampling units)	Select the first 5 rows of each of the vectors in the data frame `> MACNALLY[1:5,]`
	Select each of the vectors for the row called 'Pearcedale' from the data frame `> MACNALLY['Pearcedale',]`
Indexing by columns (variables)	Select all rows but just the second and forth vector of the data frame `> MACNALLY[,c(2,4)]`
	Select the GST and EYR vectors for all sites from the dataframe `> MACNALLY[,c('GST','EYR')]`
Indexing by conditions	Select the data for sites that have GST values greater than 3 `> MACNALLY[MACNALLY$GST>3,]`

	Select data for 'Mixed' habitat sites that have GST values greater than 3
	`> MACNALLY[MACNALLY$GST>3 &`
	`+ MACNALLY$HABITAT=='Mixed',]`

```
Devilbend                    Mixed  8.3 7.1
Olinda                       Mixed  4.6 5.3
Fern Tree Gum       Montane Forest  3.2 5.2
Sherwin         Foothills Woodland  4.6 1.2
...

> #extract the GST and EYR densities from sites in which GST
> #is greater than 3
> subset(MACNALLY, GST>3, select=c('GST','EYR'))

              GST EYR
Reedy Lake    3.4 0.0
Pearcedale    3.4 9.2
Warneet       8.4 3.8
Lysterfield   5.6 5.6
Red Hill      8.1 4.1
Devilbend     8.3 7.1
Olinda        4.6 5.3
Fern Tree Gum 3.2 5.2
Sherwin       4.6 1.2
...
```

The `subset()` *function* can be used within many other analysis functions and therefore provides a convenient way of performing data analysis on subsets of larger data sets.

2.7.2 The `%in%` matching operator

It is often desirable to subset according to multiple alternative conditions. The `%in%` *operator* searches through all of the entries in the object on the lefthand side for matches with any of the entries within the vector on the righthand side.

```
> #subset the MACNALLY dataset according to those rows that
> #correspond to HABITAT 'Montane Forest' or 'Foothills Woodland'
> MACNALLY[MACNALLY$HABITAT %in% c("Montane Forest",
+ "Foothills Woodland"),]
                           HABITAT  GST EYR
Fern Tree Gum       Montane Forest  3.2 5.2
Sherwin         Foothills Woodland  4.6 1.2
Heathcote Ju        Montane Forest  3.7 2.5
Warburton           Montane Forest  3.8 6.5
Panton Gap          Montane Forest  3.8 3.8
St Andrews      Foothills Woodland  4.7 3.6
Nepean          Foothills Woodland 14.0 5.6
Tallarook       Foothills Woodland  4.3 2.9
```

Convieniently, the `%in%` *operator* can also be used in the `subset` *function*.

2.7.3 Pivot tables and aggregating datasets

Sometimes it is necessary to calculate summary statistics of a vector separately for different levels of a factor. This is achieved by specifying the numeric vector, the factor (or list of factors) and the summary statistic function (such as mean) as *arguments* in the `tapply()` *function*.

```
> #calculate the mean GST densities per HABITAT
> tapply(MACNALLY$GST, MACNALLY$HABITAT, mean)
    Montane Forest Foothills Woodland              Mixed
          3.625000           6.900000           5.035294
       Gipps.Manna        Box-Ironbark      River Red Gum
          5.325000           4.575000           3.300000
```

When it is necessary to calculate the summary statistic for multiple variables at a time, or to retain the dataset format to facilitate subsequent analyses or graphical summaries, the `aggregate()` *function* is very useful.

```
> #calculate the mean GST and EYR densities per habitat
> aggregate(MACNALLY[c('GST','EYR')],
+ list(Habitat=MACNALLY$HABITAT), mean)
             Habitat      GST      EYR
1     Montane Forest 3.625000 4.500000
2 Foothills Woodland 6.900000 3.325000
3              Mixed 5.035294 4.264706
4        Gipps.Manna 5.325000 6.925000
5       Box-Ironbark 4.575000 1.450000
6      River Red Gum 3.300000 0.000000
```

Alternatively, the `gsummary()` *functon* within the `nlme` and `lme4` *packages* performs similarly. The `gsummary()` *function* performs more conveniently than `aggregate()` on grouped data (data containing hierarchical blocking or nesting).

```
> library(nlme)
> gsummary(MACNALLY[c("GST", "EYR")], groups = MACNALLY$HABITAT)
                        GST      EYR
Montane Forest     3.625000 4.500000
Foothills Woodland 6.900000 3.325000
Mixed              5.035294 4.264706
Gipps.Manna        5.325000 6.925000
Box-Ironbark       4.575000 1.450000
River Red Gum      3.300000 0.000000
```

2.7.4 Sorting datasets

Often it is necessary to rearrange or sort datasets according to one or more variables. This is done by using the `order()` *function* to generate the row indices. By default, data are sorted in increasing order, however this can be reversed by supplying the `decreasing=T` *argument* to the `order()` *function*. It is possible to sort according to multiple variables simply by specifying a comma separated list of the vector names (see example below), whereby the data are sorted first by the first supplied vector, then the next and so on. Note however, when multiple vectors are supplied, all are sorted in the same direction.

```
> MACNALLY[order(MACNALLY$HABITAT, MACNALLY$GST), ]

                           HABITAT  GST EYR
Fern Tree Gum      Montane Forest  3.2 5.2
Heathcote Ju       Montane Forest  3.7 2.5
Warburton          Montane Forest  3.8 6.5
Panton Gap         Montane Forest  3.8 3.8
Tallarook      Foothills Woodland  4.3 2.9
Sherwin        Foothills Woodland  4.6 1.2
St Andrews     Foothills Woodland  4.7 3.6
Nepean         Foothills Woodland 14.0 5.6
Donna Buang                 Mixed  1.5 0.0
...
```

To appreciate how this is working, examine just the `order` component

```
> order(MACNALLY$HABITAT, MACNALLY$GST)
 [1]  9 11 12 15 37 10 20 21 26 19 35 14  1 17 23  8 27 13  5 18
[21] 22 28  6  7 16  4  2 24  3 33 34 25 36 30 32 29 31
```

Hence when this sequence is applied as row indices to MACNALLY, it would be interpreted as 'display row 13, then row 27, 29 etc'.

2.7.5 Accessing and evaluating expressions within the context of a dataframe

For times when you find it necessary to repeatedly include the name of the dataframe within functions and expressions, the `with()` *function* is very convenient. This function evaluates an expression (which can include functions) within the context of the dataframe. Hence, the above `order()` illustration could also be performed as:

```
> with(MACNALLY, order(HABITAT, GST))
 [1]  9 11 12 15 37 10 20 21 26 19 35 14  1 17 23  8 27 13  5 18
[21] 22 28  6  7 16  4  2 24  3 33 34 25 36 30 32 29 31
```

2.7.6 Reshaping dataframes

Data sets are typically constructed such that variables (vectors) are in columns and replicates are in rows. This standard format (known as long format) allows a huge variety of graphical and numerical summaries and analyses to be performed with minimal need for data alterations. Nevertheless, there are a small number of analyses (such as paired *t*-tests, repeated measures and multivariate analysis of variance (MANOVA)) that can be performed on, or else require data to be arranged in *wide* format. In wide format, the rows represent blocks or individuals and the repeated measurements (responses to treatments within each block) are arranged in columns. Conversion between long and wide data formats is provided by the `reshape()` *function*. To illustrate, we will use the Walter and O'Dowd (1992) randomized block dataset in which the number of mites encountered on leaves with and without domatia blocked within plants were modelled.

```
> walter<-read.table('walter.csv', header=TRUE, sep=',')
> #view first six rows of the walter data set
> head(walter)
  LEAVES BLOCK TREAT MITE
1     a1     1     1    9
2     a2     1     2    1
3     b1     2     1    2
4     b2     2     2    1
5     c1     3     1    0
6     c2     3     2    2
```

Using the `reshape()` *function* to convert the long format into wide format:

```
> walter.wide <- reshape(walter, v.names = "MITE",
+     timevar = "TREAT", idvar = "BLOCK", direction = "wide",
+     drop = "LEAVES")
> walter.wide
   BLOCK MITE.1 MITE.2
1      1      9      1
3      2      2      1
5      3      0      2
7      4     12      4
9      5     15      2
11     6      3      1
13     7     11      0
15     8      6      0
17     9      7      1
19    10      6      0
21    11      5      1
23    12      8      1
25    13      3      1
27    14      6      0
```

In the above, `v.names=` specifies the names of vectors from the long format whose values will fill the repeated measures columns of the wide format, `timevar=` specifies the names of categorical vectors in the long format whose levels will define the separate repeated measures columns, `idvar=` specifies the names of categorical vectors in the long format that define the blocks or individuals. The `direction=` *argument* specifies the format of the resulting dataframe and `drop=` specifies the name of any vectors in the long format that can be removed prior to reshaping. Similarly, the `reshape()` *function* can be used to convert wide to long format. Reshaping from wide to long format is often desirable, since while the long format is necessary for most analysis and summaries, the wide format is typically more compact and suitable for field data collection sheets and spreadsheet entry. For the purpose of an example, the following wide data set represents seal counts from ten sites at three different times of the day (08:00, 12:00 and 16:00). The researcher wishes to reshape it to long format to facilitate analyses.

```
> seals <- data.frame(Seal = paste("Site", 1:10, sep = ""),
+     T8.00 = c(10, 35, 67, 2, 49, 117, 26, 85, 20,
+         15), T12.00 = c(15, 47, 88, 3, 46, 132, 41,
+         101, 36, 18), T16.00 = c(9, 31, 62, 0, 39,
+         86, 11, 3, 14, 7))
> seals.long <- reshape(seals, varying = c("T8.00",
+     "T12.00", "T16.00"), v.names = "Count", timevar = "TIME",
+     times = paste("T", seq(8, 16, by = 4), sep = ""),
+     idvar = "Seal", direction = "long")
> seals.long
              Seal TIME Count
Site1.T8     Site1   T8    10
Site2.T8     Site2   T8    35
Site3.T8     Site3   T8    67
Site4.T8     Site4   T8     2
Site5.T8     Site5   T8    49
Site6.T8     Site6   T8   117
Site7.T8     Site7   T8    26
Site8.T8     Site8   T8    85
Site9.T8     Site9   T8    20
Site10.T8   Site10   T8    15
Site1.T12    Site1  T12    15
Site2.T12    Site2  T12    47
Site3.T12    Site3  T12    88
Site4.T12    Site4  T12     3
Site5.T12    Site5  T12    46
Site6.T12    Site6  T12   132
Site7.T12    Site7  T12    41
Site8.T12    Site8  T12   101
Site9.T12    Site9  T12    36
Site10.T12  Site10  T12    18
Site1.T16    Site1  T16     9
Site2.T16    Site2  T16    31
```

```
Site3.T16   Site3  T16    62
Site4.T16   Site4  T16     0
Site5.T16   Site5  T16    39
Site6.T16   Site6  T16    86
Site7.T16   Site7  T16    11
Site8.T16   Site8  T16     3
Site9.T16   Site9  T16    14
Site10.T16 Site10  T16     7
```

2.8 Dummy data sets - generating random data

Most statisticians strongly recommend that research questions be designed around sets of well defined statistical procedures. This ensures that the eventual data analyses remain possible and relatively straightforward. Furthermore, many would recommend the generation and mock analysis of dummy data sets that approximate the anticipated structure and variability of the anticipated data. This enables many of the common data analysis problems to be anticipated, thereby allowing solutions to be considered prior to data collection. Dummy data sets are usually created by filling the response variable(s) (and continuous predictor variables) with random data.

R uses the Mersenne-Twister Random Number Generator (RNG) with a random number sequence cycle of $2^{19937} - 1$. All random number generators have what is known as a 'seed'. This is a number that uniquely identifies a series of random number sequences. Strictly, computer generated random numbers are 'pseudo-random' as the sequences themselves are predefined. However, with such a large number of possible sequences ($2^{19937} - 1$), for all intents and purposes they are random.

By default, the initial random seed is generated from the computer clock (milliseconds field) and is therefore unbiased. However, it is also possible to specify a random seed. This is often useful for error-checking functions. Additionally, it also facilitates learning how to perform randomizations, as the same outcomes can be repeated.

R has a family of functions (see Table 2.2) that extract random numbers from a range of mathematical distributions that represent the common sampling and statistical distributions encountered in biology.

For example, imagine that you were interested in examining the effect of four different nitrogen treatments (N1, N2, N3, N4) on the growth rate of a particular species of plant. An ANOVA (see chapter 10) appeared suitable for your intended experimental design, and you prudently decided to run a mock analysis prior to data collection. Previous studies had indicated that the growth rate of the plant species was normally distributed with a mean of around 250 mm per year with a standard deviation of about 20 mm, and you had decided (for whatever reason) to have 10 replicates of each treatment. Using these criteria it is possible to generate a dummy data set.

Table 2.2 Random number generation functions for different sampling distributions.

Distribution	Example syntax

Normal	`> # generate 5 random numbers from a normal`
	`> # distribution with a mean of 10 and a standard`
	`> # deviation of 1`
	`> rnorm(5,mean=10,sd=1)`
	`[1] 11.564555  9.732885  8.357070  8.690451 12.272846`

Log-Normal	`> # generate 5 random numbers from a log-normal`
	`> # distribution whose logarithm has a mean of 2 and a`
	`> # standard deviation`
	`> # of 1`
	`> rlnorm(5,mean=2,sd=1)`
	`[1]  8.157636 30.914781 20.175299  5.071559 16.364014`

Uniform	`> # generate 5 random numbers from a uniform`
	`> # distribution with a minimum of 2 and a`
	`> # maximum of 10`
	`> runif(5,min=1,max=10)`
	`[1] 4.710560 8.155589 8.272690 6.898405 4.226496`

Poisson	`> # generate 5 random numbers from a Poisson`
	`> # distribution with a lambda parameter of 4`
	`> rpois(5,min=1,max=10)`
	`[1] 4 4 2 6 1`

Binomial	`> # generate 5 random numbers from a binomial`
	`> # distribution based on 10 Bernoulli trials and`
	`> # a prob. of 0.5`
	`> rbinom(5,size=10,prob=.5)`
	`[1] 4 4 1 4 6`

Negative binomial	`> # generate 5 random numbers from a negative binomial`
	`> # distribution based on 10 Bernoulli trials and`
	`> # an alternative parameterization (mu) of 4`
	`> rnbinom(5,size=10,mu=4)`
	`[1] 5 7 1 4 5`

Exponential	`> # generate 5 random numbers from a exponential`
	`> # distribution with a lambda rate of 2`
	`> rexp(5,rate=2)`
	`[1] 0.3138283 1.1896221 0.2466995 0.4090852 1.1757822`

```
> # create the response variable with four sets of 10 random
> # numbers from a normal distribution
> GROWTH.RATE <- c(rnorm(10, 250,20), rnorm(10, 250,20),
+ rnorm(10, 250,20),rnorm(10, 250,20))
> # create the nitrogen treatment factor with four levels each
> # replicated 10 times
> TREATMENT <- gl(4,10,40,c('N1', 'N2', 'N3', 'N4'))
> # combine the vectors into a dataframe
> NITROGEN <- data.frame(GROWTH.RATE, TREATMENT)
```

For multifactor designs, the `expand.grid()` *function* provides a convenient way to generate dataframes containing all combinations of one or more factors. Following from the previous example, imagine you now wanted to create mock data for a two factor (nitrogen treatment and season) ANOVA design. A dummy data set could be created as follows:

```
> # create the nitrogen treatment factor with four levels
> TREATMENT <- c("N1","N2","N3","N4")
> # create the season factor with two levels
> SEASON <- c("WINTER", "SUMMER")
> # use the expand.grid function to create a dataframe with each
> # combination replicated 5 times
> TS<-expand.grid(TREATMENT=TREATMENT,SEASON=SEASON, reps=1:5)
> # combine a normally distributed response variable to the
> # factor combinations using the data.frame function
> NITROGEN<-data.frame(TS,GROWTH.RATE=rnorm(40,250,20))
```

The data can now be subject to the statistical and graphical procedures. Dummy data sets are also useful for examining the possible impacts of missing data and unbalanced designs.

3

Introductory statistical principles

Statistics is a branch of mathematical sciences that relates to the collection, analysis, presentation and interpretation of data and is therefore central to most scientific pursuits. Fundamental to statistics is the concept that samples are collected and statistics are calculated to *estimate populations* and their *parameters*.

Statistical populations can represent natural biological populations (such as the Victorian koala population), although more typically they reflect somewhat artificial constructs (e.g. Victorian male koalas). A statistical population strictly refers to all the possible observations from which a sample (a subset) can be drawn and is the entity about which you wish to make conclusions.

The population parameters are the characteristics (such as population mean, variability etc) of the population that we are interested in drawing conclusions about. Since it is usually not possible to observe an entire population, the population parameters must be estimated from corresponding statistics calculated from a subset of the population known as a sample (e.g sample mean, variability etc). Provided the sample adequately represents the population (is sufficiently large and unbiased), the sample statistics should be reliable estimates of the population parameters of interest. It is primarily for this reason that most statistical procedures assume that sample observations have been drawn randomly from populations (to maximize the likelihood that the sample will truly represent the population). Additional terminology fundamental to the study of biometry are listed in Table 3.1.

In addition to estimating population parameters, various statistical functions (or *statistics*) are often calculated to express the relative magnitude of trends within and between populations. For example, the degree of difference between two populations is usually described by a t-statistic (see chapter 6). Another important concept in statistics is the idea of *probability*. The probability of an event or outcome is the proportion of times that the event or outcome is expected to occur in the long-run (after a large number of repeated procedures). For many statistical analyses, probabilities of occurrence are used as the basis for conclusions, inferences and predictions.

Consider the vague research question "How much do Victorian male koalas weigh?". This could be interpreted as:

- How much do each of the Victorian male koalas weigh individually?
- What is the total mass of all Victorian male koalas added together?
- What is the mass of the typical Victorian male koala?

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

Table 3.1 List of important terms. Examples pertain to a hypothetical research investigation into estimating the protein content of koala milk.

Term	Definition	Example
Measurement	A single piece of recorded information reflecting a characteristic of interest (e.g. length of a leaf, pH of a water aliquot mass of an individual, number of individuals per quadrat etc)	Protein content of the milk of a single female koala
Observation	A single measured sampling or experimental unit (such as an individual, a quadrat, a site etc)	A small quantity of milk from a single koala
Population	All the possible observations that could be measured and the unit of which wish to draw conclusions about (note a statistical population need not be a viable biological population)	The milk of all female koalas
Sample	The (representative) subset of the population that are observed	A small quantity of milk collected from 15 captive female koalas[a]
Variable	A set of measurements of the same type that comprise the sample. The characteristic that differs (varies) from observation to observation	The protein content of koala milk.

[a] Note that such a sample may not actually reflect the defined population. Rather, it could be argued that such a sample reflects captive populations. Nevertheless, such extrapolations are common when field samples are difficult to obtain.

Arguably, it is the last of these questions that is of most interest. We might also be interested in the degree to which these weights differ from individual to individual and the frequency of individuals in different weight classes.

3.1 Distributions

The set of observations in a sample can be represented by a *sampling* or *frequency distribution*. A frequency distribution (or just distribution) represents how often observations in certain ranges occur (see Figure 3.1a). For example, how many male koalas in the sample weigh between 10 and 11 kg, or how many weigh more than 12 kg. Such a sampling distribution can also be expressed in terms of the probability (long-run likelihood or chance) of encountering observations within certain ranges. For example, the probability of encountering a male koala weighing more than 12 kg is equal to the proportion of male koalas in the sample that weighed greater than 12 kg. It is then referred to as a probability distribution. When a frequency distribution can be described by a mathematical function, the probability distribution is a curve. The total area under this curve is defined as 1 and thus, the area under sections of the

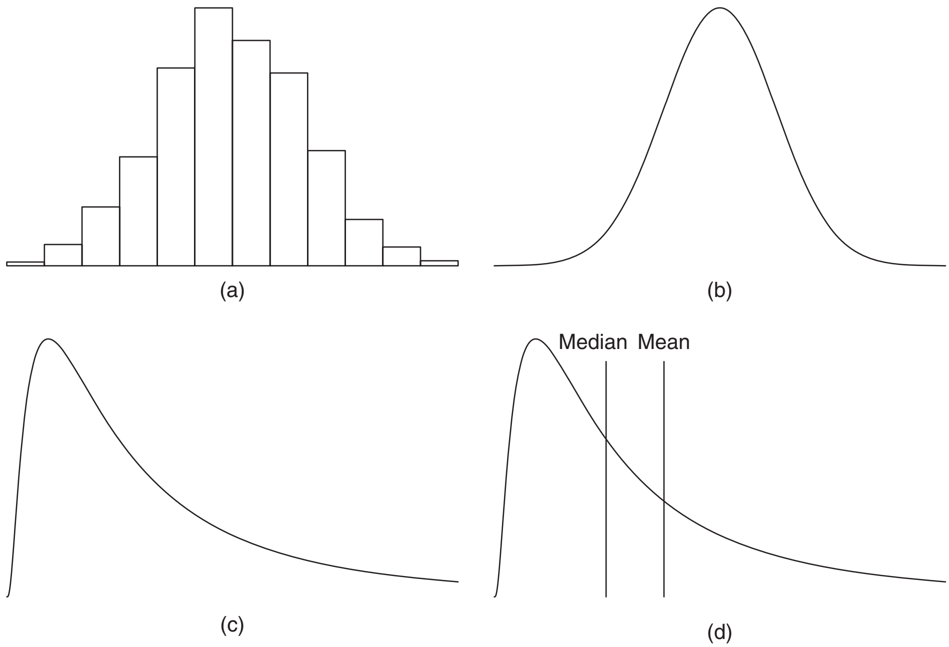


Fig 3.1 Fictitious histogram (a) and (b) normal and (c-d) log-normal probability distributions.

curve represent the probability of values falling in the associated interval. Note, it is not possible to determine the probability of discrete events (such as the probability of encountering a koala weighing 12.183 kg) only ranges of values.

3.1.1 The normal distribution

It has been a long observed mathematical phenomenon that the accumulation of a set of independent random influences tend to converge upon a central value (**central limit theorem**) and that the distribution of such accumulated values follow a specific 'bell shaped' curve called a *normal* or *Gaussian* distribution (see Figure 3.1b). The normal distribution is a symmetrical distribution in which values close to the center of the distribution are more likely and that progressively larger and smaller values are less commonly encountered.

Many biological measurements (such as the weight of a Victorian male koala) are likewise influenced by an almost infinite number of factors (many of which can be considered independent and random) and thus many biological variables also follow a normal distribution. Since many scientific variables behave according to the central limit theorem, many of the common statistical procedures have been specifically derived for (and thus assume) normally distributed data. In fact, the reliability of inferences based on such procedures is directly related to the degree of conformity to this assumption of normality. Likewise, many other statistical elements rely on normal distributions, and thus the normal distribution (or variants thereof) is one of the most important mathematical distributions.

3.1.2 Log-normal distribution

Many biological variables have a lower limit of zero (at least in theory). For example, a koala cannot weigh less than 0 kg or there cannot be fewer than zero individuals in a quadrat. Such circumstances can result in asymmetrical distributions that are highly truncated towards the left with a long right tail (see Figure 3.1c). In such cases, the mean and median present different values (the latter arguably more reflective of the 'typical' value), see Figure 3.1d. These distributions can often be described by a log-normal distribution. Furthermore, some variables do not naturally vary on a linear scale. For example, growth rates or chemical concentrations might naturally operate on logarithmic or exponential scales. Consequently, when such data are collected on a linear scale, they might be expected to follow a non-normal (perhaps log-normal) distribution.

3.2 Scale transformations

Essentially, data transformation is the process of converting the scale in which the observations were measured into another scale. I will demonstrate the principles of data transformation with two simple examples. Firstly, to illustrate the legitimacy and commonness of data transformations, imagine you had measured water temperature in a large number of streams. Let's assume that you measured the temperature in °C. Supposing later you required the temperatures be in °F. You would not need to re-measure the stream temperatures. Rather, each of the temperatures could be converted from one scale (°C) to the other (°F). Such transformations are very common.

Imagine now that a botanist wanted to examine the leaf size of a particular species. The botanist decides to measure the length of a random selection of leaves using a standard linear, metric ruler and the distribution of sample observations are illustrated in Figure 3.2a. The growth rate of leaves might be expected to be greatest in small leaves and deccelerate with increasing leaf size. That is, the growth rate of leaves might be expected to be logarithmic rather than linear. As a result, the distribution of leaf sizes using a linear scale might also be expected to be non-normal (log-normal). If, instead of using a linear scale, the botanist had used a logarithmic ruler, the distribution of leaf sizes may have been more like that depicted in Figure 3.2b.

If the distribution of observations is determined by the scale used to measure of the observations, and the choice of scale (in this case the ruler) is somewhat arbitrary (a linear scale is commonly used because we find it easier to understand), then it is justifiable to convert the data from one scale to another after the data has been collected and explored. It is not necessary to re-measure the data in a different scale. Therefore, to normalize the data, the botanist can simply convert the data to logarithms.

The important points in the process of transformations are;

(i) The order of the data has not been altered (a large leaf measured on a linear scale is still a large leaf on a logarithmic scale), only the spacing of the data has changed
(ii) Since the spacing of the data is purely dependent on the scale of the measuring device, there is no reason why one scale is more correct than any other scale
(iii) For the purpose of normalization, data can be converted from one scale to another

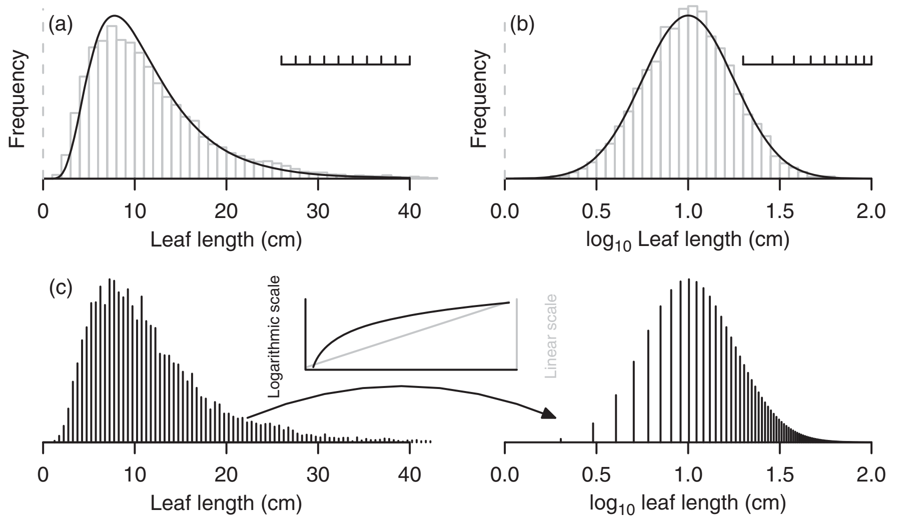


Fig 3.2 Ficticious illustration of scale transformations. Leaf length measurements collected on a linear a) and logarithmic b) scale yielding log-normal and normal sampling distributions respectively. Leaf length measurements collected on a linear scale can be *normalized* by applying a logarithmic function (inset) to each measurement. Such a scale transformation only alters the relative spacing of measurements c). A largest leaf has the largest values on both scales.

Table 3.2 Common data transformations.

Nature of data	Transformation	R syntax
Measurements (lengths, weights, etc)	$\log_e$	`log(x)`
	$\log_{10}$	`log(x, 10)`
	$\log_{10}$	`log10(x)`
	$\log x + 1$	`log(x+1)`
Counts (number of individuals, etc)	$\sqrt{}$	`sqrt(x)`
Percentages (must be proportions)	arcsin	`asin(sqrt(x))*180/pi`

where `x` is the name of the vector (variable) whose values are to be transformed.

The purpose of scale transformation is purely to normalize the data so as to satisfy the underlying assumptions of a statistical analysis. As such, it is possible to apply any function to the data. Nevertheless, certain data types respond more favourably to certain transformations due to characteristics of those data types. Common transformations and R syntax are provided in Table 3.2.

3.3 Measures of location

Measures of location describe the center of a distribution and thus characterize the typical value of a population. There are many different measures of location (see Table 3.3), all of which yield identical values (in the center of the distribution) when

Table 3.3 Commonly estimated population parameters[a].

Parameter	Description	R syntax
Estimates of Location		
Arithmetic mean (μ)	The sum of the values divided by the number of values (n)	`mean(X)`
Trimmed mean	The arithmetic mean calculated after a fraction (typically 0.05 or 5%) of the lower and upper values have been discarded	`mean(X, trim=0.05)`
Winsorized mean	The arithmetic mean is calculated after the trimmed values are replaced by the upper and lower trimmed quantiles	`library(psych)` `winsor(X, trim=0.05)`
Median	The middle value	`median(X)`
Minimum, maximum	Smallest and largest values	`min(X),max(X)`
Estimates of Spread		
Variance (σ^2)	Average deviation of observations from the mean	`var(X)`
Standard deviation (σ)	Square-root of variance	`sd(X)`
Median absolute deviation	The median difference of observations from the median value	`mad(X)`
Inter-quartile range	Difference between the 75% and 25% ranked observations	`IQR(X)`
Precision and confidence		
Standard error of $\bar{y}$ ($s_{\bar{y}}$)	Precision of the estimate $\bar{y}$	`sd(X)/sqrt(length(X))`
95% confidence interval of μ	Interval with a 95% probability of containing the true mean	`library(gmodels)` `ci(X)`

[a]Only L-estimators are provided. L-estimators are linear combinations of weighted statistics on ordered values. M-estimators (of which maximum likelihood is an example) are calculated as the minimum of some function(s).

the population (and sample) follows an exactly symmetrical distribution. Whilst the mean is highly influenced by unusually large or small values (outliers) and skewed distributions, the median is more *robust*. The greater the degree of asymmetry and outliers, the more disparate the different measures of location.

3.4 Measures of dispersion and variability

In addition to having an estimate of the typical value (center of a distribution), it is often desirable to have an estimate of the spread of the values in the population. That is, do all Victorian male koalas weigh the same or do the weights differ substantially?

In its simplest form, the variability, or spread, of a population can be characterized by its range (difference between maximum and minimum values). However, as ranges can only increase with increasing sample size, sample ranges are likely to be a poor estimate of population spread. *Variance* (s^2) describes the typical deviation of values from the typical (mean) value:

$$s^2 = \sum \frac{(y_i - \bar{y})^2}{n - 1}$$

Note that by definition, the mean value must be in the center of all the values, and thus the sum of the positive and negative deviations will always be zero. Consequently, the deviances are squared prior to summing. Unfortunately, this results in the units of the spread estimates being different to the units of location. *Standard deviation* (the square-root of the variance) rectifies this issue.

Note also, that population variance (and standard deviation) estimates are calculated with a denominator of $n - 1$ rather than n. The reason for this is that since the sample values are likely to be more similar to the sample mean (which is of course derived from these values) than to the fixed, yet unknown population mean, the sample variance will always underestimate the population variance. That is, the sample variance and standard deviations are biased estimates of the population parameters. Division by *n-1* rather than n is an attempt to partly offset these biases.

There are more robust (less sensitive to outliers) measures of spread including the inter-quartile range (difference between 75% and 25% ranked observations) and the median absolute deviation (MAD: the median difference of observations from the median value).

3.5 Measures of the precision of estimates - standard errors and confidence intervals

Since sample statistics are used to estimate population parameters, it is also desirable to have a measure of how good the estimates are likely to be. For example, how well the sample mean is likely to represent the true population mean. The proximity of an estimated value to the true population value is its *accuracy*. Clearly, as the true value of the population parameter is never known (hence the need for statistics), it is not possible to determine the accuracy of an estimate. Instead, we measure the *precision* (repeatability, consistency) of the estimate. Provided an estimate is repeatable (likely to be obtained from repeated samples) and that the sample is a good, unbiased representative of the population, a precise estimate should also be accurate.

Strictly, precision is measured as the degree of spread (standard deviation) in a set of sample statistics (e.g. means) calculated from multiple samples and is called the *standard error*. The standard error can be estimated from a single sample by dividing the sample standard deviation by the square-root of the sample size ($\frac{\sigma}{\sqrt{n}}$). The smaller the standard error of an estimate, the more precise the estimate is and thus the closer it is likely to approximate the true population parameter.

The central limit theorem (which predicates that any set of averaged values drawn from an identical population will always converge towards being normally distributed) suggests that the distribution of repeated sample means should follow a normal distribution and thus can be described by its overall mean and standard deviation (=standard

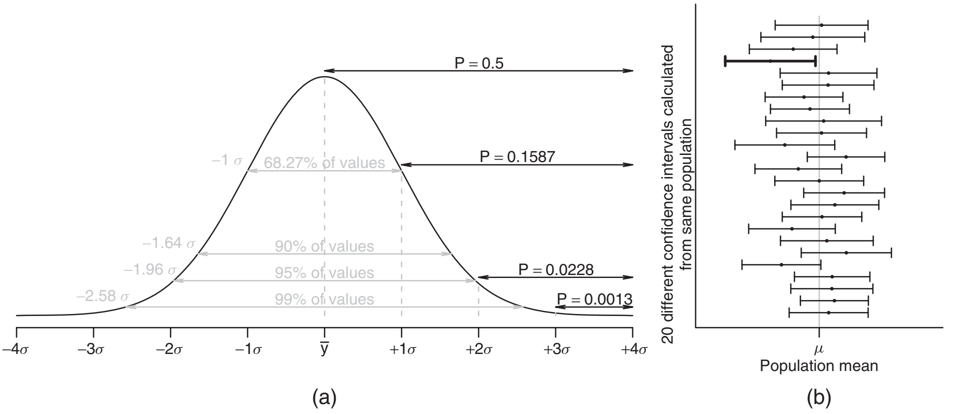


Fig 3.3 (a) Normal distribution displaying percentage quantiles (grey) and probabilities (areas under the curve) associated with a range of standard deviations beyond the mean. (b) 20 possible 95% confidence intervals from 20 samples ($n = 30$) drawn from the one population. Bold intervals are those that do not include the true population mean. In the long run, 5% of such intervals will not include the population mean (μ).

error). In fact, since the standard error of the mean is estimated from the same single sample as the mean, its distribution follows a special type of normal distribution called a t-distribution. In accordance to the properties of a normal distribution (and thus a t-distribution with infinite degrees of freedom), 68.27% of the repeated means fall between the true mean and $\pm$ one sample standard error (see Figure 3.3). Put differently, we are 68.27% percent confident that the interval bound by the sample mean plus and minus one standard error will contain the true population mean. Of course, the smaller the sample size (lower the degrees of freedom), the flatter the t-distribution and thus the smaller the level of confidence for a given span of values (interval).

This concept can be easily extended to produce intervals associated with other degrees of confidence (such as 95%) by determining the percentiles (and thus number of standard errors away from the mean) between which the nominated percentage (e.g. 95%) of the values lie (see Figure 3.3a). The 95% confidence interval is thus defined as:

$$P\left\{\bar{y} - t_{0.05(n-1)}s_{\bar{y}} \leq \mu \leq \bar{y} + t_{0.05(n-1)}s_{\bar{y}}\right\}$$

where $\bar{y}$ is the sample mean, $s_{\bar{y}}$ is the standard error, $t_{0.05(n-1)}$ is the value of the 95% percentile of a t distribution with $n-1$ degrees of freedom, and μ is the unknown population mean. For a 95% confidence interval, there is a 95% probability that the interval will contain the true mean (see Figure 3.3b). Note, this interpretation is about the interval, not the true population value, which remains fixed (albeit unknown). The smaller the interval, the more confidence is placed in inferences about the estimated parameter.

3.6 Degrees of freedom

The concept of degrees of freedom is sufficiently abstract and foreign to those new to statistical principles that it warrants special attention. The *degrees of freedom* refers to how many observations in a sample are 'free to vary' (theoretically take on any value) when calculating independent estimates of population parameters (such as population variance and standard deviation).

In order for any inferences about a population to be reliable, each population parameter estimate (such as the mean and the variance) must be independent of one another. Yet they are usually all obtained from a single sample and to estimate variance, a prior estimate of the mean is required. Consequently, mean and variance estimated from the same sample cannot strictly be independent of one another.

When estimating the population variance (and thus standard deviation) from sample observations, not all of the observations can be considered independent of the estimate of population mean. The value of at least one of the observations in the sample is constrained (not free to vary). If, for example, there were four observations in a sample with a mean of 5, then the first three of these can theoretically take on any value, yet the forth value must be such that the sum of the values is still 20. The degrees of freedom therefore indicates how many **independent** observations are involved in the estimation of a population parameter. A 'cost' of a single degree of freedom is incurred for each prior estimate required in the calculation of a population parameter.

The shape of the probability distributions of coefficients (such as those in linear models etc) and statistics depend on the number of degrees of freedom associated with the estimates. The greater the degrees of freedom, the narrower the probability distribution and thus the greater the statistical power[a]. Degrees of freedom (and thus power) are positively related to sample size (the greater the number of replicates, the greater the degrees of freedom and power) and negatively related to the number of variables and prior required parameters (the greater the number of parameters and variables, the lower the degrees of freedom and power).

3.7 Methods of estimation

3.7.1 Least squares (LS)

Least squares (LS) parameter estimation is achieved by simply **minimizing** the overall differences between the observed sample values and the estimated parameter(s). For example, the least squares estimate of the population mean is a value that minimizes the differences between the sample values and this estimated mean. Least squares estimation has no inherent basis for testing hypotheses or constructing confidence

[a] Power is the probability of detecting an effect if an effect genuinely occurs.

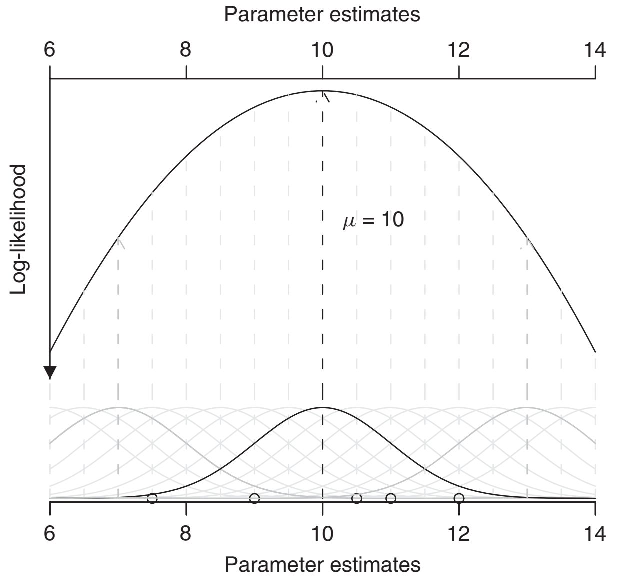


Fig 3.4 Diagrammatic illustration of ML estimation of μ.

intervals and is thus primarily for parameter estimation. Least squares estimation is used extensively in simple model fitting procedures (e.g. regression and analysis of variance) where optimization (minimization) has an exact solution that can be solved via simultaneous equations.

3.7.2 Maximum likelihood (ML)

The maximum likelihood (ML) approach estimates one or more population parameters such that the (log) likelihood of obtaining the observed sample values from such populations is **maximized** for a nominated probability distribution.

Computationally, this involves summing the probabilities of obtaining each observation for a range of possible population parameter estimates, and using integration to determine the parameter value(s) that maximize the likelihood. A simplified example of this process is represented in Figure 3.4.

Probabilities of obtaining observations for any given parameter value(s) are calculated according to a specified exponential probability distribution (such as normal, binomial, Poisson, gamma or negative binomial). When the probability distribution is normal (as in Figure 3.4), ML estimators for linear model parameters have exact computational solutions and are identical to LS solutions (see section 3.7.1). However for other probability distributions (for which LS cannot be used), ML estimators involve complex iterative calculations. Unlike least squares, the maximum likelihood estimation framework also provides standard errors and confidence intervals for estimations and therefore provides a basis for statistical inference. The major draw back of this method is that it typically requires strong assumptions about the underlying distributions of the parameters.

3.8 Outliers

Outliers are extreme or unusual values that do not fall within the normal range of the data. As many of the commonly used statistical procedures are based on means and variances (both of which are highly susceptible to extreme observations), outliers tend to bias statistical outcomes towards these extremes. For a statistical outcome to reliably reflect population trends, it is important that all observed values have an equal influence on the statistical outcomes. Outliers, however, have a greater influence on statistical outcomes than the other observations and thus, the resulting statistical outcomes may no longer represent the population of interest.

There are numerous mathematical methods that can be used to identify outliers. For example, an outlier could be defined as any value that is greater than two standard deviations from the mean[b]. Alternatively, outliers could be defined as values that are greater than two times the inter-quartile range from the inter-quartile range.

Outliers are caused by a variety of reasons including errors in data collection or transcription, contamination or unusual sampling circumstances, or the observation may just be naturally unusual. Dealing with outliers therefore depends on the cause and requires a great deal of discretion.

- If there are no obvious reasons why outlying observations could be considered unrepresentative, they must be retained although it is often worth reporting the results of the analyses with and without these influential observations
- Omitting outliers can be justified if there is reason to suspect that they are not representative (due to sampling errors etc), although their exclusion should always be acknowledged.
- There are many statistical alternatives that are based on more robust (less affected by departures from normality or the presence of outliers) measures that should be employed if outliers are present.

3.9 Further reading

Fowler, J., L. Cohen, and P. Jarvis. (1998). *Practical statistics for field biology*. John Wiley & Sons, England.

Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

[b] This method clearly assumes that the observations are normally distributed.

4

Sampling and experimental design with R

A fundamental assumption of nearly all statistical procedures is that samples are collected randomly from populations. In order for a sample to truly represent a population, the sample must be collected without bias (intentional or otherwise). R has a rich array of randomization tools to assist researches randomize their sampling and experimental designs.

4.1 Random sampling

Biological surveys involve the collection of observations from naturally existing populations. Ideally, every possible observation should have an equal likelihood of being selected as part of the sample. The `sample()` *function* facilitates the drawing of random samples.

Selecting sampling units from a numbered list

Imagine wanting to perform bird surveys within five forested fragments which are to be randomly selected from a list of 37 fragments:

```
> sample(1:37, 5, replace=F)
[1]  2 16 28 30 20

> MACNALLY <- read.table("macnally.csv", header=T, sep=",")
> sample(row.names(MACNALLY), 5, replace=F)
[1] "Arcadia"     "Undera"      "Warneet"     "Tallarook"
[5] "Donna Buang"
```

Selecting sample times

Consider a mammalogist who is about to conduct spotlighting arboreal mammal surveys at 10 different sites (S1→S10). The mammalogist wants to randomize the time (number of minutes since sundown) that each survey commences so as to restrict any sampling biases or confounding dial effects. Since the surveys are to take exactly

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

20 minutes and the maximum travel time between sites is 10 minutes, the survey starting times need to be a minimum of 30 minutes apart. One simple way to do this is to generate a sequence of times at 30 minute intervals from 0 to 600 (60 × 10) and then randomly select 10 of the times using the `sample()` *function*:

```
> sample(seq(0,600, by=30), 10, replace=F)
 [1] 300  90 270 600 480 450  30 510 120 210
```

However, these times are not strictly random, as only a small subset of possible times could have been generated (multiples of 30). Rather, they are a regular sequence of times that could potentially coincide with some natural rhythm, thereby confounding the results. A more statistically sound method is to generate an initial random starting time and then generate a set of subsequent times that are a random time greater than 30 minutes, but no more than (say) 60 minutes after the preceding time. A total of 10 times can then be randomly selected from this set.

```
> # First step is to obtain a random starting (first survey)
> # time. To do this retain the minimum time from a random set of
> # times between 1 (minute) and 60*10 (number of minutes in
> # 10 hours)
> TIMES <- min(runif(20,1,60*10))
> # Next we calculate additional random times each of which is a
> # minimum and maximum of 30 and 60 minutes respectively after
> # the previous
> for(i in 2:20) {
+ TIMES[i] <- runif(1,TIMES[i-1]+30,TIMES[i-1]+60)
+ if(TIMES[i]>9*60) break
+ }
> # Randomly select 10 of these times
> TIMES <- sample(TIMES, 10, replace=F)
> # Generate a Site name for the times
> names(TIMES) <- paste('Site',1:10, sep='')
> # Finally sort the list and put it in a single column
> cbind('Times'=sort(TIMES))
           Times
Site6   53.32663
Site9   89.57309
Site5  137.59397
Site1  180.17486
Site4  223.28241
Site2  312.30799
Site3  346.42314
Site10 457.35221
Site7  513.23244
Site8  554.69444
```

Note, that potentially any times could have been generated, and thus this is a better solution. This relatively simple example could be further extended with the use of some of the Date-Time functions.

```
> # Convert these minutes into hs, mins, seconds
> hrs <- TIMES%/%60
> mins <- trunc(TIMES%%60)
> secs <- trunc(((TIMES%%60)-mins)*60)
> RelTm <- paste(hrs,sprintf("%2.0f",mins),secs, sep=":")
> # We could also express them as real times
> # If sundown occurs at 18:00 (18*60*60 seconds)
> RealTm<-format(strptime(RelTm, "%H:%M:%S")+(18*60*60),
+ "%H:%M:%S")
> # Finally sort the list and put it in a single column
> data.frame('Minutes'=sort(TIMES),
+ 'RelativeTime'=RelTm[order(TIMES)],
+ RealTime=RealTm[order(TIMES)])
         Minutes RelativeTime RealTime
Site6   53.32663      0:53:19 18:53:19
Site9   89.57309      1:29:34 19:29:34
Site5  137.59397      2:17:35 20:17:35
Site1  180.17486      3: 0:10 21:00:10
Site4  223.28241      3:43:16 21:43:16
Site2  312.30799      5:12:18 23:12:18
Site3  346.42314      5:46:25 23:46:25
Site10 457.35221      7:37:21 01:37:21
Site7  513.23244      8:33:13 02:33:13
Site8  554.69444      9:14:41 03:14:41
```

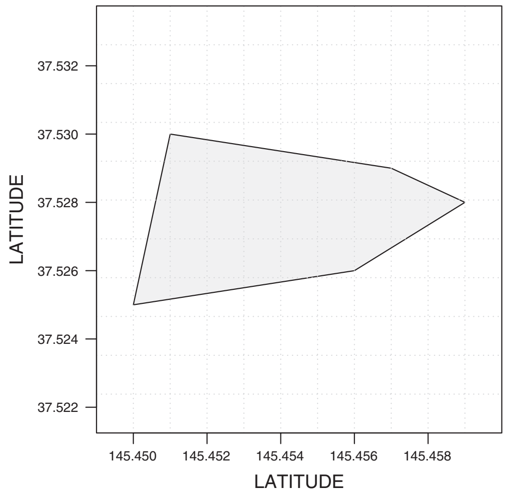
Selecting random coordinates from a rectangular grid

Consider requiring 10 random quadrat locations from a 100 × 200 m grid. This can done by using the `runif()` *function* to generate two sets of random coordinates:

```
> data.frame(X=runif(10,0,100), Y=runif(10,0,200))
           X          Y
1  87.213819 114.947282
2   9.644797  23.992531
3  41.040160 175.342590
4  97.703317  23.101111
5  52.669145   1.731125
6  63.887850  52.981325
7  56.863370  54.875307
8  27.918894  46.495312
9  94.183309 189.389244
10 90.385280 151.110335
```

Random coordinates of an irregular shape

Consider designing an experiment in which a number of point quadrats (lets say five) are to be established in a State Park. These points are to be used for stationary 10 minute bird surveys and you have decided that the location of each of the point quadrats within each site should be determined via random coordinates to minimize sampling bias. As represented in figure to the right, the site is not a regular rectangle and therefore the above technique is not appropriate. This problem is solved by first generating a matrix of site boundary coordinates (GPS latitude and longitude), and then using a specific set of *functions* from the `sp`[a] *package* to generate the five random coordinates.

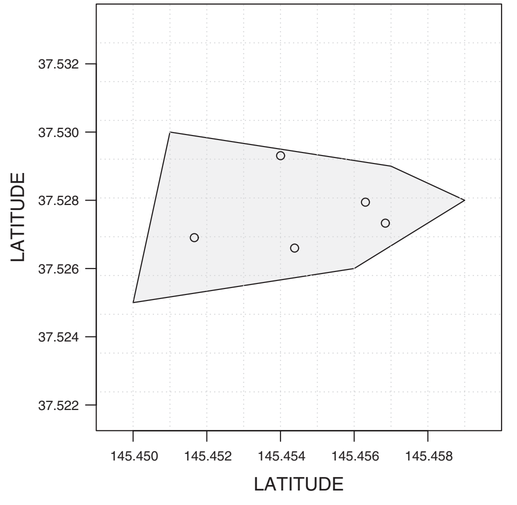



```
> LAT <- c(145.450, 145.456, 145.459, 145.457, 145.451, 145.450)
> LONG <- c(37.525, 37.526, 37.528, 37.529, 37.530,37.525)
> XY <- cbind(LAT,LONG)
> plot(XY, type='l')
> library(sp)
> XY.poly <- Polygon(XY)
> XY.points <- spsample(XY.poly, n=8, type='random')
> XY.points
SpatialPoints:
           r1       r2
[1,] 145.4513 37.52938
[2,] 145.4526 37.52655
[3,] 145.4559 37.52746
[4,] 145.4573 37.52757
[5,] 145.4513 37.52906
[6,] 145.4520 37.52631
[7,] 145.4569 37.52871
[8,] 145.4532 37.52963
Coordinate Reference System (CRS) arguments: NA
```

[a] Note that the *function* responsible for generating the random coordinates (`spsample()`) is only guaranteed to produce approximately the specified number of random coordinates, and will often produce a couple more or less. Furthermore, some locations might prove to be unsuitable (if for example, the coordinates represented a position in the middle of a lake). Consequently, it is usually best to request a 50% more than are actually required and simply ignore any extras.

These points can then be plotted on the map.

```
> points(XY.points[1:5])
```



Lets say that the above site consisted of two different habitats (a large heathland and a small swamp) and you wanted to use stratified random sampling rather than pure random sampling so as to sample each habitat proportionally. This is achieved in a similar manner as above, except that multiple spatial rings are defined and joined into a more complex spatial data set.

```
> LAT1 <- c(145.450, 145.456, 145.457, 145.451,145.450)
> LONG1 <- c(37.525, 37.526, 37.529, 37.530, 37.525)
> XY1 <- cbind(LAT1,LONG1)
> LAT2 <- c(145.456,145.459,145.457,145.456)
> LONG2 <- c(37.526, 37.528, 37.529,37.526)
> XY2 <- cbind(LAT2,LONG2)
> library(sp)
> XY1.poly <- Polygon(XY1)
> XY1.polys <- Polygons(list(XY1.poly), "Heathland")
> XY2.poly <- Polygon(XY2)
> XY2.polys <- Polygons(list(XY2.poly), "Swamp")
> XY.Spolys <- SpatialPolygons(list(XY1.polys, XY2.polys))
> XY.Spoints <- spsample(XY.Spolys, n=10, type='stratified')
> XY.Spoints
SpatialPoints:
            x1       x2
 [1,] 145.4504 37.52661
 [2,] 145.4529 37.52649
 [3,] 145.4538 37.52670
 [4,] 145.4554 37.52699
```

```
 [5,] 145.4515 37.52889
 [6,] 145.4530 37.52846
 [7,] 145.4552 37.52861
 [8,] 145.4566 37.52738
 [9,] 145.4578 37.52801
[10,] 145.4510 37.52946
Coordinate Reference System (CRS) arguments: NA
```

The `spsample()` *function* supports random sampling (`'random'`), stratified random sampling (`'stratified'`), systematic sampling (`'regular'`) and non-aligned systematic sampling (`'nonaligned'`). Visual representations of each of these different sampling designs are depicted in Figure 4.1.

Random distance or coordinates along a line

Random locations along simple lines such as linear transects, can be selected by generating sets of random lengths. For example, we may have needed to select a single point along each of ten 100 m transects on four occasions. Since we effectively require $10 \times 4 = 40$ random distances between 0 and 100 m, we generate these distances

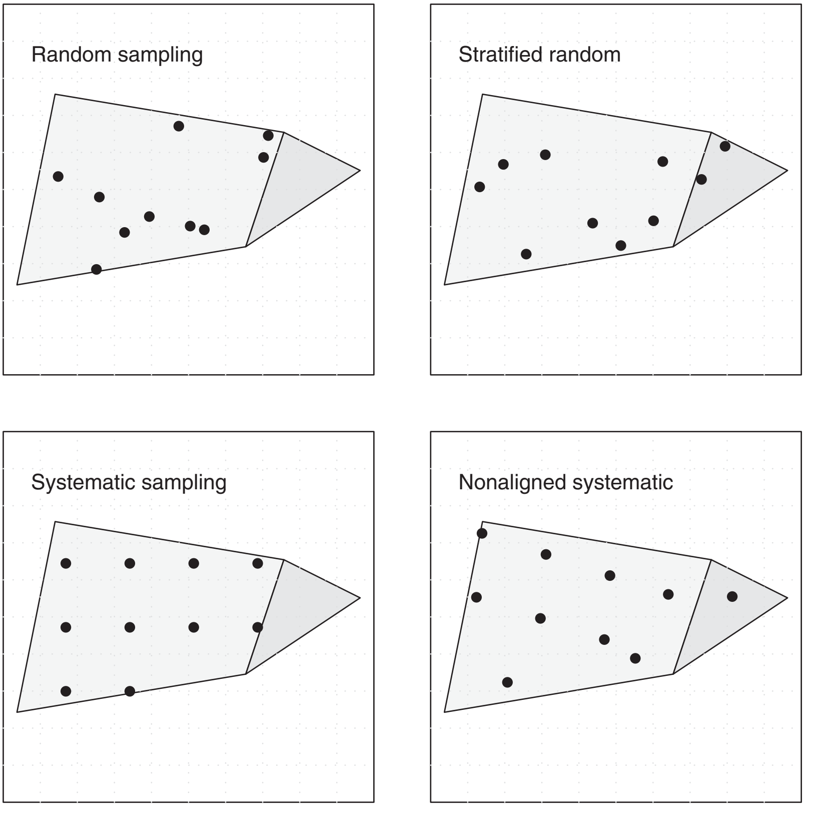


Fig 4.1 Four different sampling designs supported by the `spsample()` *function*.

and arrange them in a 10 × 4 matrix where the rows represent the transects and the columns represent the days:

```
> DIST <- matrix(runif(40,0,100),nrow=10)
> DIST
           [,1]       [,2]      [,3]      [,4]
 [1,]  7.638788 89.4317359 24.796132 24.149444
 [2,] 31.241571  0.7366166 52.682013 38.810297
 [3,] 87.879788 88.2844160  2.437215 32.059111
 [4,] 28.488424  6.3546905 78.463586 60.120835
 [5,] 25.803398  4.8487586 98.311620 87.707566
 [6,] 10.911730 25.5682093 90.443998  9.097557
 [7,] 63.199593 36.7521530 62.775836 29.430201
 [8,] 20.946571 42.7538255  4.389625 81.236970
 [9,] 94.274397 21.9937230 64.892213 70.588414
[10,] 13.114078  9.7766933 43.903295 90.947627
```

To make the information more user friendly, we could put apply row and column names and round the distances to the nearest centimeter:

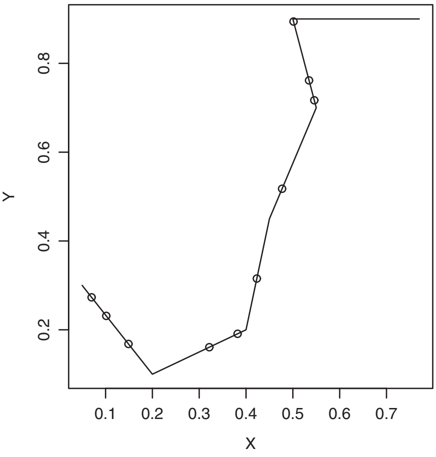
```
> rownames(DIST) <- paste("Transect", 1:10, sep='')
> colnames(DIST) <- paste("Day", 1:4, sep='')
> round(DIST, digits=2)
            Day1  Day2  Day3  Day4
Transect1   7.64 89.43 24.80 24.15
Transect2  31.24  0.74 52.68 38.81
Transect3  87.88 88.28  2.44 32.06
Transect4  28.49  6.35 78.46 60.12
Transect5  25.80  4.85 98.31 87.71
Transect6  10.91 25.57 90.44  9.10
Transect7  63.20 36.75 62.78 29.43
Transect8  20.95 42.75  4.39 81.24
Transect9  94.27 21.99 64.89 70.59
Transect10 13.11  9.78 43.90 90.95
```

If the line represents an irregular feature such as a river, or is very long, it might not be convenient to have to measure out a distance from a particular point in order to establish a sampling location. These circumstances can be treated similar to other irregular shapes. First generate a matrix of X,Y coordinates for major deviations in the line, and then use the `spsample()` *function* to generate a set of random coordinates.

```
> X <- c(0.77,0.5,0.55,0.45,0.4, 0.2, 0.05)
> Y <- c(0.9,0.9,0.7,0.45,0.2,0.1,0.3)
> XY <- cbind(X,Y)
> library(sp)
> XY.line <- Line(XY)
> XY.points <- spsample(XY.line,n=10,'random')
```

```
> plot(XY, type="l")
> points(XY.points)
```

```
> coordinates(XY.points)
              X         Y
 [1,] 0.5538861 0.9000000
 [2,] 0.4171638 0.2858188
 [3,] 0.3869956 0.1934978
 [4,] 0.4579028 0.4697570
 [5,] 0.3109703 0.1554851
 [6,] 0.1238188 0.2015750
 [7,] 0.5398741 0.6746852
 [8,] 0.4826300 0.5315749
 [9,] 0.1745837 0.1338884
[10,] 0.5248993 0.6372481
```



4.2 Experimental design

Randomization is also important in reducing confounding effects. Experimental design incorporates the order in which observations should be collected and/or the physical layout of the manipulation or survey. Good experimental design aims to reduce the risks of bias and confounding effects.

4.2.1 Fully randomized treatment allocation

Lets say that you were designing an experiment in which you intended to investigate the effect of fertilizer on the growth rate of a species of plant. You intended to have four different fertilizer treatments (A, B, C and D) and a total of six replicate plants per treatment. The plant seedlings are all in individual pots housed in a greenhouse and to assist with watering, you want to place all the seedlings on a large table arranged in a 4×6 matrix. To reduce the impacts of any potentially confounding effects (such as variations in water, light, temperature etc), fertilizer treatments should be assigned to seedling positions completely randomly.

This can be done by first generating a factorial vector (containing the levels A, B, C, and D, each repeated six times), using the `sample` *function* to randomize the treatment orders and then arranging it in a 4×6 matrix:

```
> TREATMENTS <- gl(4,6,24,c('A','B','C','D'))
> matrix(sample(TREATMENTS),nrow=4)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,] "C"  "D"  "A"  "B"  "C"  "A"
[2,] "A"  "B"  "C"  "C"  "C"  "B"
[3,] "A"  "D"  "A"  "B"  "D"  "D"
[4,] "B"  "D"  "C"  "B"  "A"  "D"
```

Note that when the optional size argument (number of random entries to draw) is not supplied, the `sample()` *function* performs a random permutation of the elements of the vector.

4.2.2 Randomized complete block treatment allocation

When the conditions under which an experiment is to be conducted are expected to be sufficiently heterogeneous to substantially increase the variability in the response variable (and thus obscure the effects of the main factor), experimental units are grouped into blocks (units of space or time that are likely to have less variable background conditions). Each level of the treatment factor is then applied to a single unit within each block.

In the previous example, treatments were randomly positioned throughout the 4×6 matrix. However, if the conditions in the greenhouse were not homogeneous (perhaps the light was better at one end and the sprinkler system favoured a certain section of the table), the ability to detect any effects of fertilizer treatment might be impeded. A randomized complete block (in which each level of fertilizer is randomly positioned within each block) design is achieved by repeating the `sample()` *function* six times (one per block) and combining the result into a matrix:

```
> TREATMENTS <- replicate(6,sample(c('A','B','C','D')))
> colnames(TREATMENTS) <- paste('Block',1:6,sep='')
> TREATMENTS
     Block1 Block2 Block3 Block4 Block5 Block6
[1,] "B"    "C"    "B"    "C"    "D"    "A"
[2,] "A"    "D"    "D"    "B"    "A"    "D"
[3,] "C"    "B"    "A"    "A"    "B"    "C"
[4,] "D"    "A"    "C"    "D"    "C"    "B"
```

5

Graphical data presentation

Graphical summaries provide three very important rolls in data analyses. Firstly, they are an important part of the initial exploratory data analyses that should precede any formal statistical analyses. Secondly, they provide visual representations of the patterns and trends revealed in complex statistical analyses. Finally, in some instances (such as regression trees and ordination plots), graphical representations are the primary result of the analyses. R accommodates many of the standard exploratory data analyses via specific plotting functions. Many of these functions require little user input and produce very rudimentary plots – although the quality of such exploratory data analyses is rarely of great importance (as they are typically only for the researcher). Nevertheless, the plotting functionality within R is also highly customizable in order to produce rich, publication quality graphical and analytical summaries.

Typically, a graphic begins with a **high-level** plotting function that defines the coarse structure of the graphic including its dimensions, axes scales, plotting symbol types and titles before creating a new plotting region on the graphics device. The most frequently used high-level plotting function is the `plot()` *function* which is a generic, overloaded[a] function that produces different plots depending on the *class* of object passed as its first argument. A range of the graphics produced by plot were illustrated on page 36. Other commonly used high-level plotting functions include `hist()`, `boxplot()`, `scatterplot()` and `pairs()`. Additional elements (such as text and lines) are added using the rich set of **low-level** graphical functions available. Common low-level plotting functions include `lines()`, `points()`, `text()` and `axis()`. These functions cannot define the dimensions of the plotting region and thus can only be added to existing plots.

It is not the intention of this chapter to produce finalized versions of graphical summaries. Rather, emphasis will be on illustrating the range of the commonly used high and low level plotting functions as well as some of the many graphical options available to help achieve rich and professional graphics. Subsequent chapters will build upon these foundations and illustrate the production of publication quality figures appropriate for the designs and analyses.

[a] A function is overloaded when many separate functions contain the same name (e.g. *plot*), yet differ from each other in the arguments (input) they except and the output they produce. Function overloading provides a common, convenient name to interface a suite of functions (thereby reducing the number of names that need to be learned).

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

In the plotting system described above, graphics are built up by sequentially adding items (lines, points, text, etc) to a base plot. Each graphical element is evaluated individually. However, for data that can be naturally split into subsets (subjects, blocks), **Trellis** graphics provide an alternative system in which entire sets of graphical elements are applied consistently to multiple subplots within a grid (or trellis). The resulting multipanel displays are produced by a single set of integrated instructions that also handle the otherwise difficult tasks of coordinating the control of axes scales and aspect ratios. Furthermore, the plots represent the underlying data in a manner that closely matches their hierarchical treatment in linear modelling.

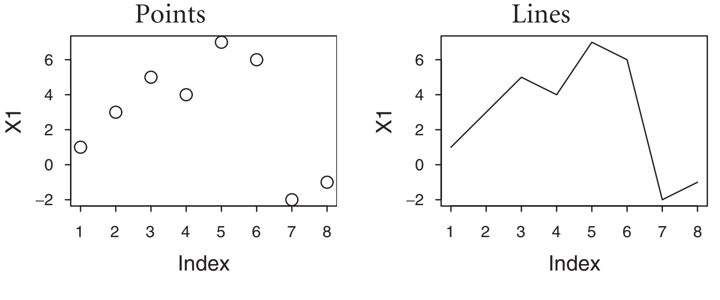
All plotting functions are handled via graphics device drivers. When R starts up, it automatically opens a graphics device driver (x11 on linux, windows on Windows and quartz or x11 on Mac OS X) ready to accept plotting commands. These graphics devices are referred to as *display* or screen devices since the output is displayed on the screen. There are also numerous *file* graphics devices (such as postscript, pdf, jpeg, etc) in which the graphical information is stored in standard formats for incorporation into other applications. Importantly, plotting commands can only be sent to a single graphical device at a time and the capabilities of different types of graphical devices vary.

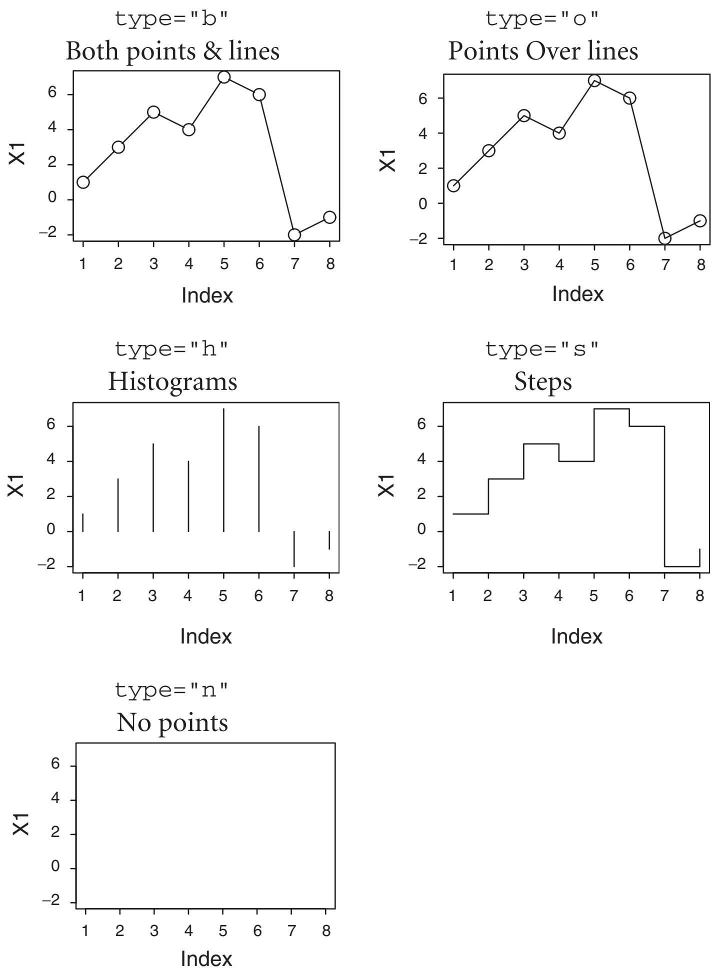
5.1 The `plot()` *function*

The `plot()` *function* is a generic (overloaded) function, the output of which depends on the class of objects passed to it as arguments (see page 36). There are many other parameters that can be used to control various aspects of the `plot()` *function*. Some of these parameters (summarized below) provide convenient ways to control the scaling and overall form of the plot and are specific to the `plot()` high level plotting function (along with many of its derivatives). Others (graphical parameters, see section 5.2) provide even finer control of the overall plot and where relevant, can be applied to most other high and low level plotting functions.

5.1.1 The `type` *parameter*

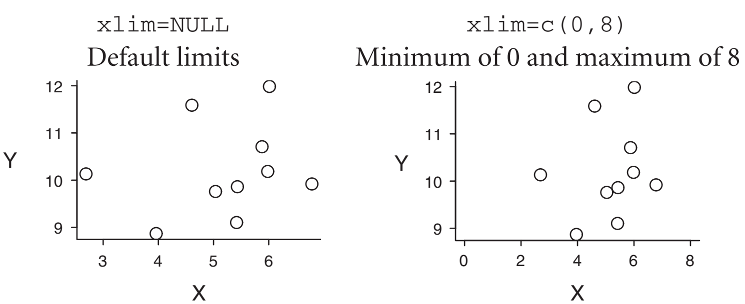
The `type` *parameter* takes a single character argument and controls how the points should be presented.





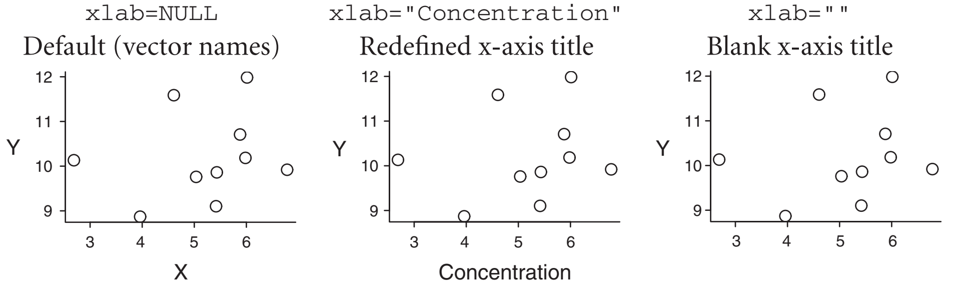
5.1.2 The `xlim` and `ylim` *parameters*

`xlim` and `ylim` control the x-axis and y-axis range respectively. These parameters take a vector with two elements (`c(min,max)`) representing the minimum and maximum scale limits.



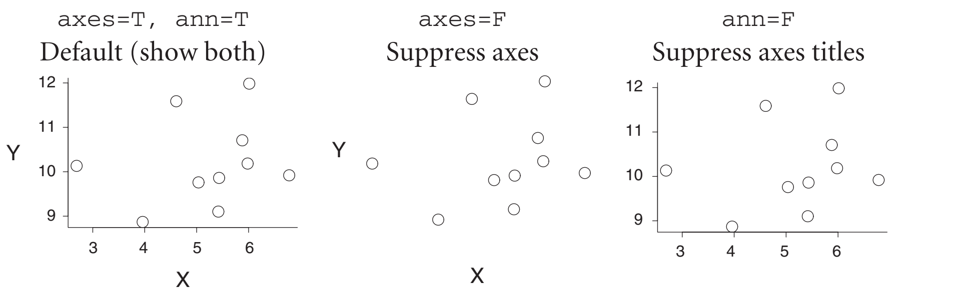
5.1.3 The `xlab` and `ylab` *parameters*

`xlab` and `ylab` define the titles for the x-axis and y-axis respectively. These parameters take a character string.



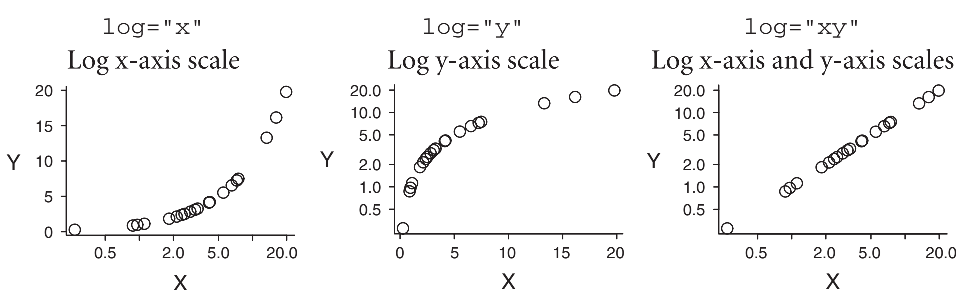
5.1.4 The `axes` and `ann` *parameters*

The `axes` and `ann` *parameters* indicates whether (=`TRUE`) or not (=`FALSE`) ALL the axes and axes titles should be plotted respectively.



5.1.5 The `log` *parameter*

The `log` *parameter* indicates whether or which axes should be plotted on a logarithmic scale.



5.2 Graphical Parameters

The graphical parameters provide consistent control over most of the plotting features across a wide range of high and low plotting functions. Any of these parameters can be set by passing them as arguments to the `par()` *function*. Once set via the `par()` *function*, they become global graphical parameters that apply to all subsequent functions that act on the current graphics device.

All of the graphical parameters have default values that are applied when a new graphical device is instantiated. When the `par()` *function* is used to alter a parameter setting, it returns a list containing the previous values of any altered parameters. Applying this list as an argument to the `par()` *function* thereby restores the previous graphical parameters.

```
> opar <- par(mar=c(4,5,1,1)
> # the plot margins of the current or new device are set
> # to be four, five, one and one text lines from the bottom,
> # left, top and right of the figure boundary
> opar

$mar
[1] 5.1 4.1 4.1 2.1

> par(opar)
> # restore plotting margins to be 5.1, 4.1, 4.1 and 2.1 text
> # lines thick.
```

Similarly, calling the `par()` *function* without any arguments returns a list containing ALL the current parameter values (altered or not) in alphabetical order. Whilst it might be tempting to use this list to apply settings to other graphical devices (or even the currently active device at a later date), since the settings will be restored alphabetically, parameters further along the alphabet will overwrite or nullify alternative parameters. For example, both `mai` and `mar` provide alternative ways of altering the plot margin dimensions, however the latter will have the final say. A safer practice for storing current settings for reuse is to call the `par()` *function* with the altered parameters twice. The first time will store the previous settings and the second will store the current altered settings.

```
> # on a new or restored device
> opar <- par(mar=c(4,5,1,1)
> npar <- par(mar=c(4,5,1,1)
> npar

$mar
[1] 4 5 1 1

> par(npar)
```

5.2.1 Plot dimensional and layout parameters

The graphical parameters responsible for controlling the dimensions and layout of graphics can only be set via the `par()` *function* and are itemized in Table 5.1 and represented in Figure 5.1.

Table 5.1 Dimensional and layout graphical parameters.

Parameter tag	Value	Description
`din`, `fin`, `pin`	`=c(width,height)`	Dimensions (width and height) of the device, figure and plotting regions (in inches)
`fig`	`=c(left,right,bottom,top)`	Coordinates of the figure region within the device. Coordinates expressed as a fraction of the device region.
`mai`, `mar`	`=c(bottom,left,top,right)`	Size of each of the four figure margins in inches and lines of text (relative to current font size).
`mfg`	`=c(row,column)`	Position of the currently active figure within a grid of figures defined by either `mfcol` or `mfrow`.
`mfcol`, `mfrow`	`=c(rows,columns)`	Number of rows and columns in a multi-figure grid.
`new`	`=TRUE` or `=FALSE`	Indicates whether to treat the current figure region as a new frame (and thus begin a new plot over the top of the previous plot (`TRUE`) or to allow a new high level plotting function to clear the figure region first (`FALSE`).
`oma`, `omd`, `omi`	`=c(bottom,left,top,right)`	Size of each of the four outer margins in lines of text (relative to current font size), inches and as a fraction of the device region dimensions
`plt`	`=c(left,right,bottom,top)`	Coordinates of the plotting region expressed as a fraction of the device region.
`pty`	`="s"` or `="m"`	Type of plotting region within the figure region. Is the plotting region a square (`="s"`) or is it maximized to fit within the shape of the figure region.
`usr`	`=c(left,right,bottom,top)`	Coordinates of the plotting region corresponding to the axes limits of the plot.

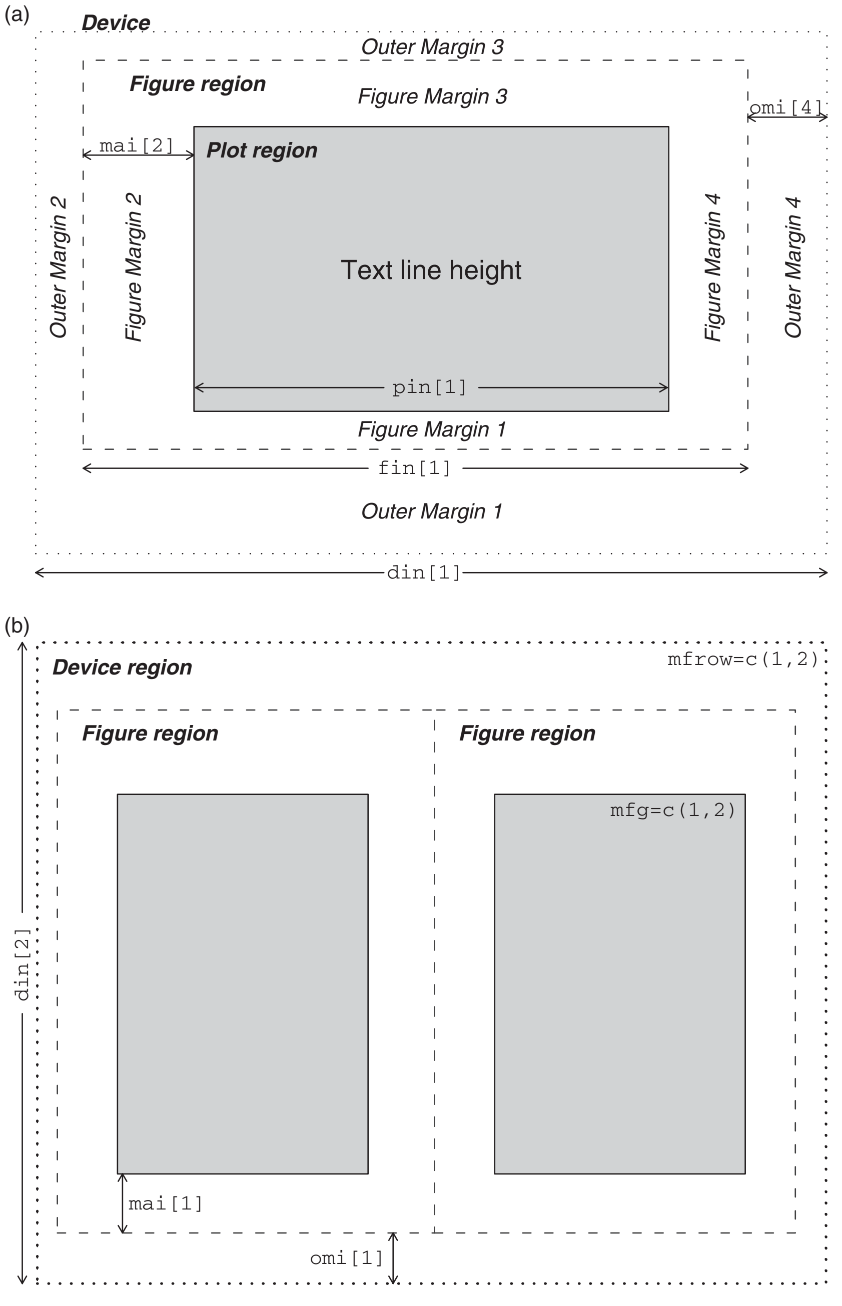


Fig 5.1 Device, figure and plotting regions along with examples of the graphical parameters that control each of the respective dimensions for (a) single figure and (b) multifigure graphics.

5.2.2 Axis characteristics

The parameters that provide finer control of the scale and formatting of the plot axes are listed in Table 5.2.

Table 5.2 Graphical parameters controlling characteristics of axes.

Parameter tag	Value	Description
`ann`, `axes`	`=T` or `=F`	High level plotting parameters that specify whether or not titles (main, sub and axes) and axes should be plotted.
`bty`	`="o","l","7","c","u" or "]"`	Single character whose upper case letter resembles the sides of the box or axes to be included with the plot.
`lab`	`=c(x,y,length)`	Specifies the length and number of tickmarks on the x and y axes.
`las`	`=0, 1, 2 or 3`	Specifies the style of the axes tick labels. `0` = parallel to axes, `1` = horizontal, `2` = perpendicular to axes, `3` = vertical
`mgp`	`=c(title,labels,line)`	Distance (in multiples of the height of a line of text) of the axis title, labels and line from the plot boundary.
`tck`, `tcl`	`=length`	The length of tick marks as a fraction of the plot dimensions (`tck`) and as a fraction of the height of a line of text (`tcl`)
`xaxp`, `yaxp`	`=c(min,max,num)`	Minimum, maximum and number of tick marks on the x and y axes
`xaxs`, `yaxs`	`="r"` or `="i"`	Determines how the axes ranges are calculated. The `"r"` option results in ranges that extend 4% beyond the data ranges, whereas the `"i"` option uses the raw data ranges.
`xaxt`, `yaxt`	`="y"`, `="n"` or `="s"`	Essentially determines whether or not to plot the axes. The `"s"` option is for compatibility with S.
`xlog`, `ylog`	`=FALSE` or `=TRUE`	Specifies whether or not the x and y axes should be plotted on a (natural) logarithmic scale.
`xpd`	`=FALSE`, `=TRUE` or `='NA'`	Specifies whether plotting is clipped to the plotting (`=FALSE`), figure (`=TRUE`) or device (`='NA'`) region

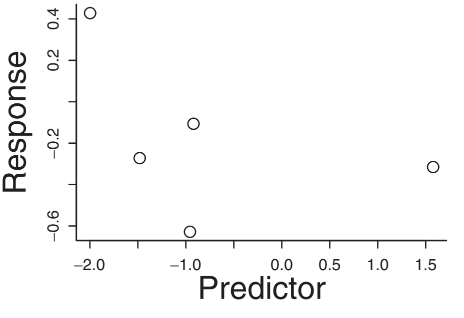
Table 5.3 Character expansion parameters.

Parameter	Applies to
`cex`	All subsequent characters
`cex.axis`	Axes tick labels
`cex.lab`	Axes titles
`cex.main`	Main plot title
`cex.sub`	Plot sub-titles

5.2.3 Character sizes

The base or default character size of text and symbols on a graphic is defined when the graphics device is initiated. Thereafter, the sizes of characters (including symbols) can be controlled by the character expansion (`cex`) *parameter*. The (`cex`) *parameter* determines the amount by which characters should be magnified relative to the base character size and can be set as an argument to the `par()` *function* as well as to individual high and low level plotting functions. In addition to the overall character expansion parameter, there are also separate character expansion parameters that control the sizes of text within each of the major components of a figure (see Table 5.3) relative to `cex`.

```
> set.seed(12)
> plot(rnorm(5,0,1), rnorm(5,0,1),
    xlab="Predictor",
    ylab="Response", cex=2,
    cex.lab=3, cex.axis=1.5,
    bty="l")
```



5.2.4 Line characteristics

Many of the characteristics of lines are controlled by arguments to the `par()` *function* or to high and low level plotting functions (see Table 5.4).

5.2.5 Plotting character *parameter* - `pch`

The plotting character (`pch`) *parameter* can be set with the `par()` *function*, and can also be set as arguments within individual high and low level plotting functions.

```
> set.seed(12)
> # plot points as solid circles
> plot(rnorm(5,0,1), rnorm(5,0,1), pch=16, axes=F,
    ann=F, cex=4)
```

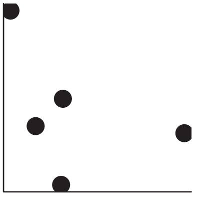


Table 5.4 Line characteristics.

Parameter	Description	Examples
`lty`	The type of line. Specified as either a single integer in the range of 1 to 6 (for predefined line types) or as a string of 2 or 4 numbers that define the relative lengths of dashes and spaces within a repeated sequence.	lty=1 lty=2 lty=3 lty=4 lty=5 lty=6 lty=7 lwd='1234' lwd='9111'
`lwd`	The thickness of a line as a multiple of the default thickness (which is device specific)	lwd=0.5 lwd=0.75 lwd=1 lwd=2 lwd=4
`lend`	The line end style (square, butt or round)	lend=2 lend=1 lend=0
`ljoin`	The style of the join between lines	ljoin=0 ljoin=1 ljoin=2

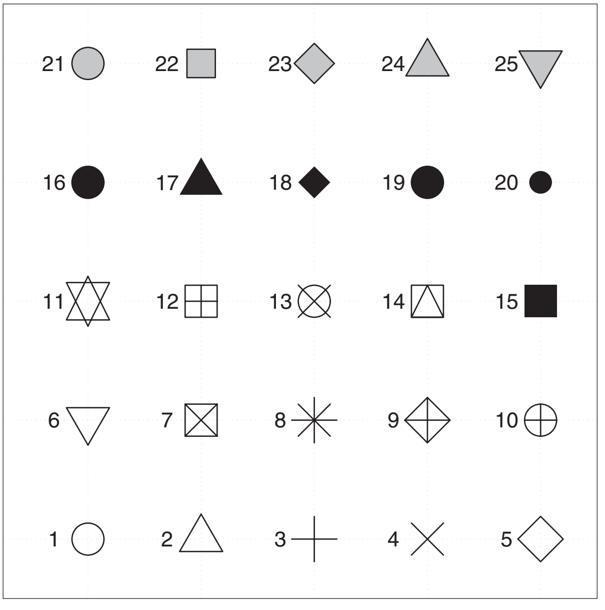


Fig 5.2 Basic `pch` plotting symbols.

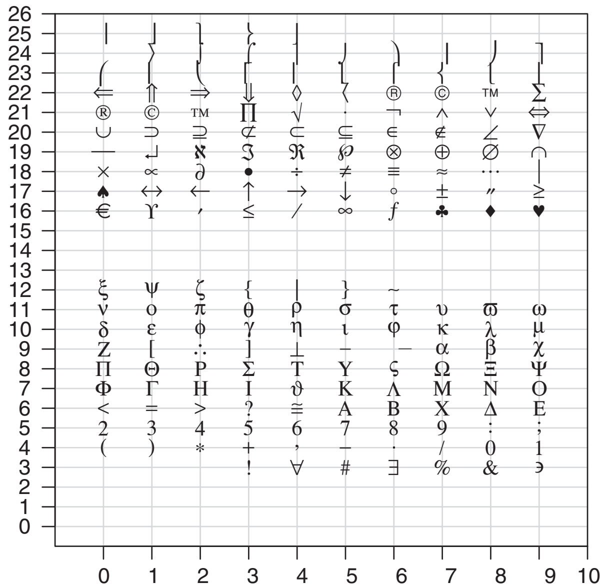
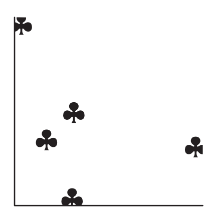


Fig 5.3 Extended `pch` plotting symbols for the symbol font (`font=5`). The plotting character number is determined from the grid by adding the x coordinate to 10 times the y coordinate. Hence, symbol ♣ is character number 167.

There are 25 basic plotting symbols (see Figure 5.2) that can be used to define the point character (`pch`) within many high and low level plotting functions. The numbers to the left of the symbols in the figure indicate the integer value used as the argument.

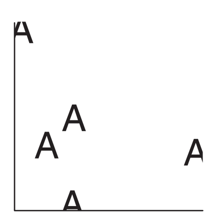
In addition to these standard plotting characters, when used in conjunction with a *symbol* font face, the `pch` *parameter* can accept any integer between 1:128 and 160:254 to yield an extended point character set (see Figure 5.3).

```
> set.seed(12)
> plot(rnorm(5,0,1), rnorm(5,0,1), pch=167, cex=4,
    font=5)
```



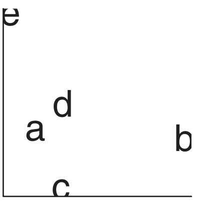
The `pch` *parameter* can also accept any other keyboard printing character (letter, number, punctuation etc) as an argument.

```
> set.seed(12)
> plot(rnorm(5,0,1), rnorm(5,0,1), pch="A",
    axes=F, cex=4)
```



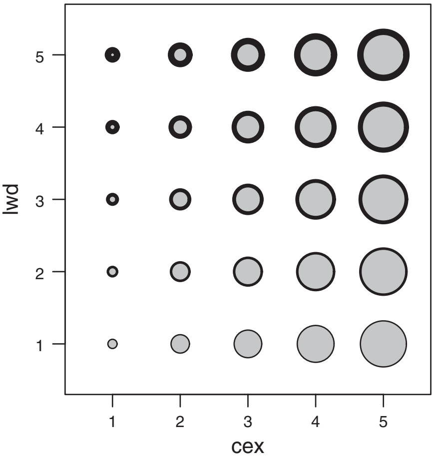
Upper and lower case letters can also be plotted respectively via the predefined `Letters[]` and `letters[]` *vectors*.

```
> set.seed(12)
> plot(rnorm(5,0,1), rnorm(5,0,1),
    pch=letters[1:5], axes=F, cex=4)
```



The size and weight of plotting symbols is controlled respectively by the `cex` (character expansion factor) and `lwd` (line width) *parameters*.

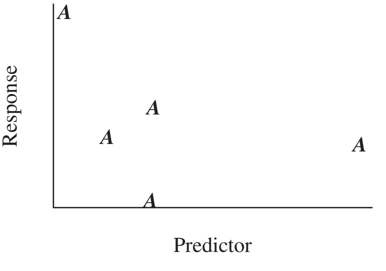
```
> m <- matrix(rep(1:5,5),nrow=5,
    byrow=F)
> plot(m, t(m), pch=21,
    bg="grey", cex=m,
    lwd=t(m), xlim=c(.5,5.5),
    ylim=c(.5,5.5), las=1,
    xlab="cex", ylab="lwd")
```



5.2.6 Fonts

The shape of text characters is controlled by the *family* (the overall visual appearance of a group of fonts - otherwise known as the typeface) and the *font* (plain, bold, italics, etc), see Figure 5.4. The font families supported varies for each graphical device as do the names by which they are referred (see Table 5.5).

```
> set.seed(12)
> # plot points with a italic serif
> # font
> plot(rnorm(5,0,1), rnorm(5,0,1),
    pch="A", family="serif", font=4,
    xlab="Predictor", ylab="Response")
```



Different fonts can also be applied to each of the main plotting components (`font.axis`: axes labels, `font.lab`: axes titles, `font.main`: Main plot title and `font.sub`: plot sub-title).

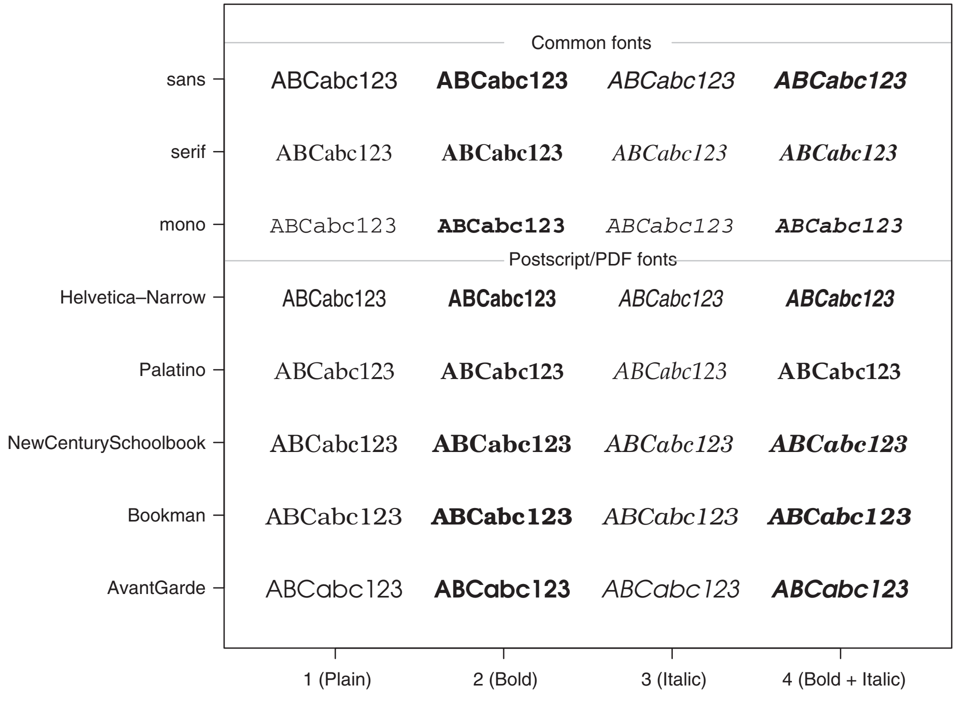


Fig 5.4 Appearance of major family (y-axis) and font (x-axis) sequences.

Table 5.5 Family names appropriate for the most common devices.

Device	Serif	Sans serif	Monospaced
Display devices			
`X11()` (Unix/Linux)	`"serif"`	`"sans"`	`"mono"`
`quartz()` (Mac OS X)	`"serif"`	`"sans"`	`"mono"`
`window()` (Windows)	`"serif"`	`"sans"`	`"mono"`
File devices			
postscript	`"Times"`	`"Helvetica"`	`"Courier"`
pdf	`"Times"`	`"Helvetica"`	`"Courier"`

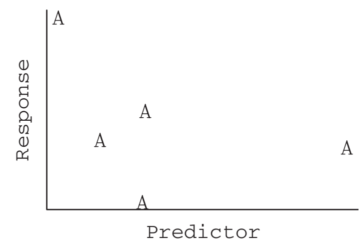
Hershey fonts

R also supports Hershey (vector) fonts that greatly extend the range of characters and symbols available. In contrast to regular (bitmap) fonts that consist of a set of small images (one for each character of each style and size), vector fonts consist of the coordinates of each of the curves required to create the character. That is, vector fonts store the information on how to draw the character rather than store the character itself.

Hershey fonts can therefore be scaled to any size without distortion. Unfortunately however, Hershey fonts cannot be combined with regular fonts in a single plotting statement and thus they cannot be easily incorporated into mathematical formulae. An extensive selection of the Hershey font characters available can be obtained by issuing the command below and following the prompts:

```
> demo(Hershey)
```

```
> set.seed(12)
> plot(rnorm(5,0,1), rnorm(5,0,1),
    pch="A", family="HersheySerif",
    xlab="Predictor", ylab="Response")
```



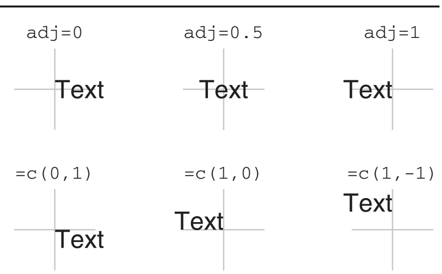
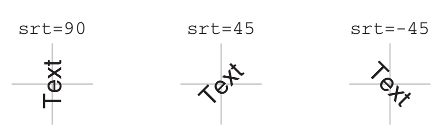
5.2.7 Text orientation and justification

The orientation and justification of characters and strings are also under the control of a set of graphics parameters (see Table 5.6).

5.2.8 Colors

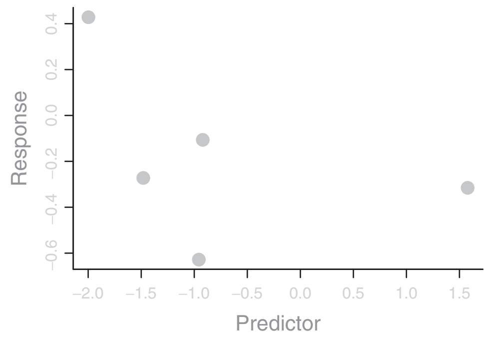
The color of all plotting elements is controlled by a set of parameters. The default color for plotting elements is specified using the `col` *parameter*. There are also separate parameters that control the color of each of the major components of a figure (`col.axis`: the axes tick labels, `col.lab`: the axes titles, `col.main`: the main plot title, `col.sub`: plot sub-titles) and when specified, take precedence over the `col` *parameter*. Two additional parameters, `bg` and `fg` can be used to control the color

Table 5.6 Text orientation and justification characteristics.

Parameter	Description	Examples
`adj`	Specifies the justification of a text string relative to the coordinates of its origin. A single number between `0` and `1` specifies horizontal justification. A vector of two numbers (`=c(x,y)`) indicates justification in horizontal and vertical directions.	
`crt`, `srt`	Specifies the amount of rotation (in degrees) of single characters (`crt`) and strings (`srt`)	

of the background and foreground (boxes and axes) respectively. The color of other elements (such as the axes themselves) is manipulated by using the `col` *parameter* within low-level plotting functions.

```
> set.seed(12)
> plot(rnorm(5,0,1),
    rnorm(5,0,1),
    xlab="Predictor",
    ylab="Response", col=8,
    col.lab="grey50",
    col.axis="grey90", bty="l")
```



There are numerous ways that colors can be specified:

- by an index (numbers 0-8) to a small palette of eight colors (0 indicates the background color). The colors in this palette can be reviewed with the `palette()` *function*.
- by name. The names of the 657 defined colors can be reviewed with the `colors()` *function*. The `epitools` *package* provides the `colors.plot()` *function* which generates a graphic that displays a matrix of all the colors. When used with the `locator=TRUE` *argument*, a series of left mouse clicks on the color squares, terminated by a right mouse click, will result in a matrix of corresponding color names.
- extract an arbitrary number (`n`) of contiguous colors from built-in color palettes
  - `rainbow(n)` - Red→Violet
  - `heat.colors(n)` - White→Orange→Red
  - `terrain.colors(n)` - White→Brown→Green
  - `topo.colors(n)` - White→Brown→Green→Blue
  - `grey(n)` - White→Black
- by direct specification of the red, green and blue components of the RGB spectrum as a character string in the form `"#RRGGBB"`. This string consists of a `#` followed by a pair of hexadecimal digits in the range `00:FF` for each component.

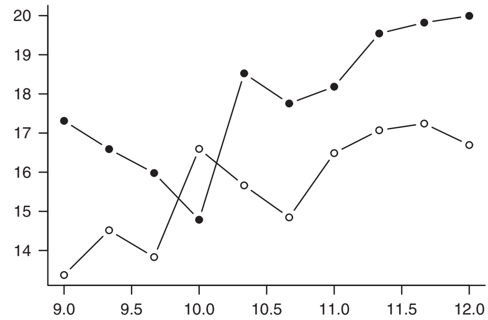
5.3 Enhancing and customizing plots with low-level plotting functions

In addition to their specific parameters, each of the following functions accept many of the graphical parameters. In the function definitions, these capabilities are represented by three consecutive dots (`...`). Technically, `...` indicates that any supplied arguments that are not explicitly part of the definition of a function are passed on to the relevant underlying functions (in this case, `par`).

5.3.1 Adding points - `points()`

Points can be added to a plot using the `points(x, y, pch, ...)` *function*. This function plots a plotting character (specified by the `pch` *parameter*) at the coordinates specified by the vectors `x,y`. Alternatively, the coordinates can be passed as a formula of the form, `y~x`.

```
> set.seed(1)
> X<-seq(9,12,l=10)
> Y1<-(1*X+2)+rnorm(10,3,1)
> Y2<-(1.2*X+2)+rnorm(10,3,1)
> plot(c(Y1,Y2)~c(X,X),
    type="n", axes=T, ann=F,
    bty="l", las=1)
> points(Y1~X,pch=21, type="b")
> points(Y2~X,pch=16, type="b")
```



5.3.2 Adding text within a plot - `text()`

The `text()` *function* adds text strings (`labels` *parameter*) to the plot at the supplied coordinates (`x,y`) and is defined as:

```
> text (x, y = NULL, labels = seq_along(x), adj = NULL,
  pos = NULL, offset = 0.5, vfont = NULL, cex = 1, col = NULL,
  font = NULL, ...)
```

Descriptions and examples of the arguments not previously outlined in the graphical parameters section, are outlined in Table 5.7.

`paste()`

The `paste()` *function* concatenates vectors together after converting each of the elements to characters. This is particularly useful for making labels and is equally

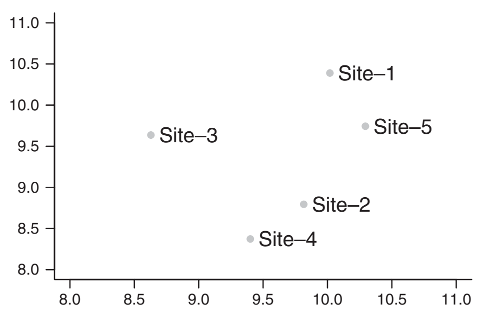
Table 5.7 `text()` arguments.

Parameter	Description	Examples
`pos`	Simplified text justification that overrides the `adj` *parameter*. `1`=below, `2`=left, `3`=above and `4`=right.	`pos=1` Text; `pos=2` Text; `pos=3` Text; `pos=4` Text
`offset`	Offset used by `pos` as a fraction of the width of a character.	`pos=1,offset=1` Text; `pos=1,offset=2` Text
`vfont`	Provision for Hershey (vector) font specification (`vfont=c(typeface, style)`.	`lab='ABCabc123'` `vfont=c('serif','plain')` ABCabc123 `lab=c('\VE','\MA','\#H0844')` `vfont=c('serif','plain')` ♀ ♂ ☆

useful in non-graphical applications. Paste has two other optional *parameters* (`sep` and `collapse`) which define extra character strings to be placed between strings joined. `sep` operates on joins between paired vector elements whereas `collapse` operates on joints of elements within a vector respectively.

```
> cc <- c("H","M","L")
> cc
[1] "H" "M" "L"
> paste(cc,1:3, sep=":")
[1] "H:1" "M:2" "L:3"
> paste(cc, collapse=":")
[1] "H:M:L"
> paste(cc, 1:3,sep="-",collapse=":")
[1] "H-1:M-2:L-3"
```

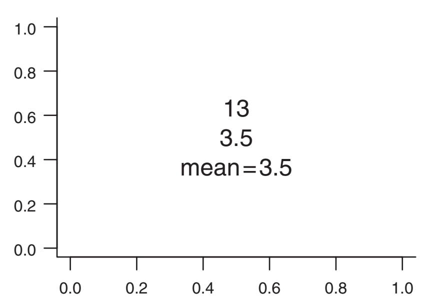
```
> set.seed(10)
> X<-rnorm(5,10,1)
> Y<-rnorm(5,10,1)
> plot(X,Y, type="n",axes=T,
    ann=F, bty="l", las=1,
    xlim=c(8,11), ylim=c(8,11))
> points(X,Y,col="grey", pch=16)
> text(X,Y,paste("Site",1:5,
    sep="-"), cex=2, pos=4)
```



Non-character arguments

Most other objects[b] passed as a label object are evaluated before being coerced into a string for plotting. In so doing, the output of other functions can be plotted.

```
> plot(c(0,1),c(0,1),type="n",
    axes=T, ann=F, bty="l", las=1)
> text(.5,.75, 5*2+3, cex=2)
> text(.5,.5, mean(c(2,3,4,5)),
    cex=2)
> text(.5,.25, paste("mean=",
    mean(c(2,3,4,5))), cex=2)
```



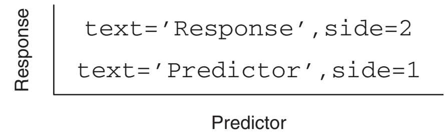
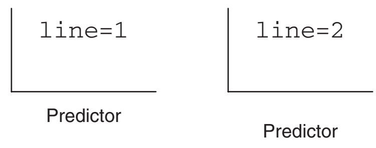
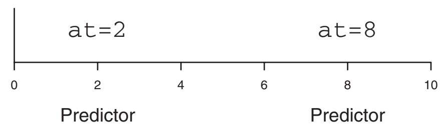
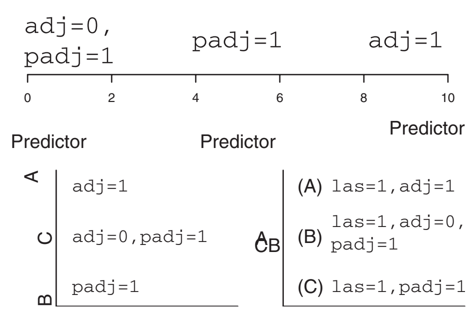
5.3.3 Adding text to plot margins - `mtext()`

The `mtext()` *function* adds text (`text`) to the plot margins and is typically used to create fancy or additional axes titles. The `mtext()` *function* is defined as:

```
> mtext(text, side = 3, line = 0, outer = FALSE, at = NA,
  adj = NA, padj = NA, cex = NA, col = NA, font = NA, ...)
```

[b] *Language objects* are treated differently (see section 5.3.5).

Table 5.8 `mtext()` arguments.

Parameter	Description	Examples
`side`	Specifies which margin the title should be plotted in. `1`=bottom, `2`=left, `3`=top and `4`=right.	
`line`	Number of text lines out from the plot region into the margin to plot the marginal text	
`outer`	For multi-plot figure, if `outer=TRUE`, put the marginal text in the outer margin (if there is one).	
`at`	Position along the axis (in user coordinates) of the text	
`adj`, `padj`	Adjustment (justification) of the position of the marginal text parallel (`adj`) and perpendicular (`padj`) to the axis. Justification depends on the orientation of the text string and the margin (axis).	

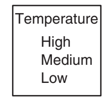
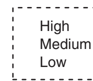
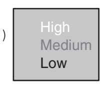
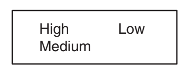
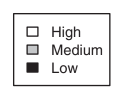
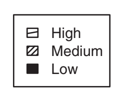
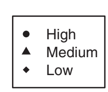
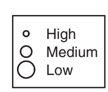
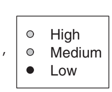
Descriptions and examples of the arguments not previously outlined in the graphical parameters section, are outlined in Table 5.8.

5.3.4 Adding a legend - `legend()`

The `legend()` *function* brings together a rich collection of plotting functions to produce highly customizable figure legends in a single call. A sense of the rich functionality of the `legend` *function* is reflected in Table 5.9 and the function definition:

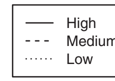
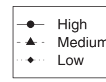
```
> legend(x, y = NULL, legend, fill = NULL, col = par("col"),
    lty, lwd, pch, angle = 45, density = NULL, bty = "o",
    bg = par("bg"), box.lwd = par("lwd"), box.lty = par("lty"),
    pt.bg = NA, cex = 1, pt.cex = cex, pt.lwd = lwd,
    xjust = 0, yjust = 1, x.intersp = 1, y.intersp = 1,
    adj = c(0, 0.5), text.width = NULL, text.col = par("col"),
    merge = do.lines && has.pch, trace = FALSE,
    plot = TRUE, ncol = 1, horiz = FALSE, title = NULL,
    inset = 0)
```

Table 5.9 `legend()` arguments. To save space, some parameter descriptions are combined, others are omitted.

Parameter	Description	Examples
`legend`	A vector of strings or expressions to comprise the labels of the legend.	
`title`	A string or expression for a title at the top of the legend	`title='Temperature'` 
`bty`, `box.lty`, `box.lwd`	The type (`"o"` or `"n"`), line thickness and line style of box framing the legend.	`box.lwd=1.5, box.lty=2` 
`bg`, `text.col`	The colors used for the legend background and legend labels	`bg='grey', text.col=c('white','grey40','black')` 
`horiz`	Whether or not to produce a horizontal legend instead of a vertical legend	`horiz=TRUE` 
`ncol`	The number of columns in which to arrange the legend labels	`ncol=2` 
`cex`	Character expansion for all elements of the legend relative to the plot `cex` graphical parameter.	
Boxes	If any of the following parameters are set, the legend labels will be accompanied by boxes.	
`fill`	Specifies the fill color of the boxes. A vector of colors will result in different fills.	`fill=c('white','grey','black')` 
`angle`, `density`	Specifies the angle and number of lines that make up the stripy fill of boxes. Negative density values result in solid fills.	`fill=c('white','grey','black')` 
Points		
`pch`	Specifies the type of plotting character.	`col=c('white','grey','black')` 
`pt.cex`, `pt.lwd`	Specifies the character expansion and line width of the plotting characters.	`pch=21,pt.cex=1:3, pt.lwd=2` 
`col`, `pt.bg`	Specifies the foreground and background color of the plotting characters (and lines for `col`).	`pch=16, pt.bg=c('grey80','grey','black'), col=1` 

(*continued overleaf*)

Table 5.9 (*continued*)

Parameter	**Description**	**Examples**
Lines	If any of the following parameters are set, the legend labels will be accompanied by lines.	
`lwd`, `lty`	Specifies the width and type of lines.	`lwd=c(1.5), lty=c(1,2,3)` 
`merge`	Whether or not to merge points and lines.	`lwd=c(1.5), lty=c(1,2,3)` 

In addition to the usual methods for specifying the positioning coordinates, convenient keywords reflecting the four corners (`"bottomleft"`, `"bottomright"`, `"topleft"`, `"topright"`) and boundaries (`"bottom"`, `"left"`, `"top"`, `"right"`) of the plotting region can alternatively be specified.

5.3.5 More advanced text formatting

The text plotting functions described above (`text()`, `mtext()` and `legend()`) can also build plotting text from objects that constitute the R language itself. These are referred to as *language objects* and include:

- **names** - the names of objects
- **expressions** - unevaluated syntactically correct statements that could otherwise be evaluated at the command prompt
- **calls** - these are specific expressions that comprise of an unevaluated named function (complete with arguments)

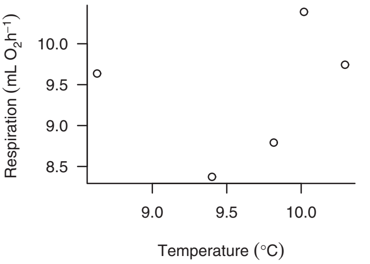
Any *language object* passed as an argument to one of the text plotting functions described above (`text()`, `mtext()` and `legend()`) will be coerced into an expression and evaluated as a mathematical expression prior to plotting. In so doing, the text plotting functions will also apply TeX-like formatting (the extensive range of which can be sampled by issuing the `demo(plotmath)` command) where appropriate. Hence, advanced text construction, formatting and plotting is thus achieved by skilled use of a variety of functions (described below) that assist in the creation of *language objects* for passing to the text plotting functions.

`expression()`

The `expression` *function* is used to build complex expressions that incorporate TeX-like mathematical formatting. Hence, the `expression` *function* is typically nested within one of the text plotting functions to plot complex combinations of characters and symbols.

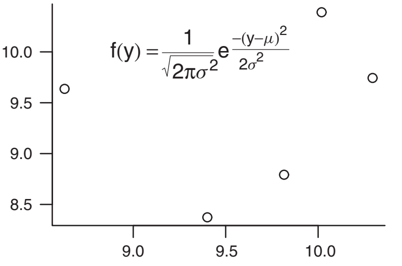
The `expression()` *function* is useful for generating axes titles with complex units.

```
> set.seed(10)
> X<-rnorm(5,10,1)
> Y<-rnorm(5,10,1)
> plot(X,Y, type="p", axes=T,
    ann=F, bty="l", las=1)
> mtext(expression(Temperature~
    (degree*C)), side=1, line=3,
     cex=1.5)
> mtext(expression(Respiration~
    (mL~O[2]~h^-1)), side=2,
     line=3.5, cex=1.5)
```



The `expression()` *function* is also useful for plotting complex mathematical formula within the plots.

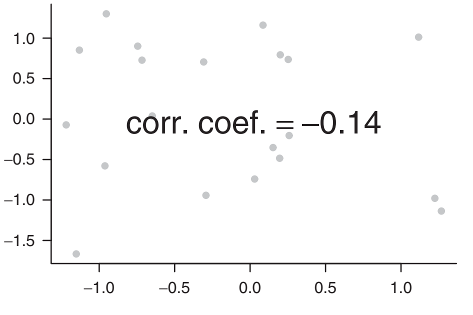
```
> set.seed(10)
> X<-rnorm(5,10,1)
> Y<-rnorm(5,10,1)
> plot(X,Y,type="p",axes=T, ann=F,
    bty="l", las=1)
> text(9.3,10, expression(f(y) ==
    frac(1,sqrt(2*pi*sigma^2))*
    e^frac(-(y-mu)^2, 2*sigma^2)),
    cex=1.25)
```



bquote()

The `bquote()` *function* generates a *language object* by converting the argument after first evaluating any objects wrapped in '`.()`'. This provides a way to produce text strings that combine mathematical formatting and the output statistical functions.

```
> set.seed(3)
> X<-rnorm(20,0,1)
> Y<-rnorm(20,0,1)
> # calculate correlation
> # between X and Y
> cc<-cor(X,Y)
> plot(X,Y,type="n",axes=T,
    ann=F, bty="l", las=1)
> points(X,Y,col="grey", pch=16)
> text(0,0,bquote(corr.~coef.==.
    (round(cc,2))), cex=4)
> text(0,0,names(cc))
```

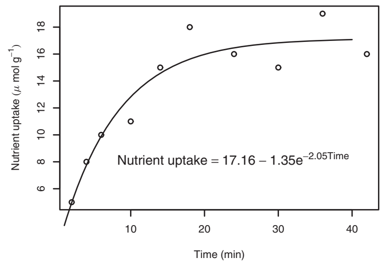


Note the required use of the tilde (~) character to allow spaces[c]. A space character at that point would have resulted in a syntactically incorrect mathematical expression.

substitute()

Alternatively, for situations in which substitutions are required within non-genuine mathematical expressions (such as straight character strings), the `substitute()` function is useful.

```
> X<-c(2,4,6,10,14,18,24,30,36,42)
> Y<-c(5,8,10,11,15,18,16,15,19,16)
> n<-nls(Y~SSasymp(X,a,b,c))
> plot(Y~X, type='p', ann=F)
> lines(1:40, predict(n,
    data.frame(X=1:40)))
> a<-round(summary(n)$coef[1,1],2)
> b<-round(summary(n)$coef[2,1],2)
> c<-round(summary(n)$coef[3,1],2)
> text(40,8,substitute(y == a
    - b*e^c*x,list(y="Nutrient
    uptake",a=a,b=b,c=c,x="Time")),
    cex=1.25, pos=2)
```




```
> mtext("Time (min)",1,line=3)
> mtext(expression(Nutrient~uptake~(mu~mol~g^-1)),
  2, line=3)
```

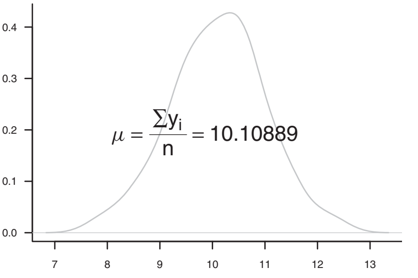
Combinations of advanced text formatting functions

It is possible to produce virtually any text representation on an R plot, however, some representations require complex combinations of the above functions. Whilst, these functions are able to be nested within one another, the combinations often appear to behave counter-intuitively. Great understanding and consideration of the exact nuances of each of the functions is required in order to successfully master their combined effects. Nevertheless, the following scenarios should provide some appreciation of the value and uses of some of these combinations.

The formula for calculating the mean of a sample ($\mu = \frac{\sum y_i}{n}$) as represented by an R mathematical expression is: `mu == frac(sum(y[i]),n)`. What if however, we wished to represent not only the formula applied to the data, but the result of the formula as well (e.g. ($\mu = \frac{\sum y_i}{n} = 10$))? To substitute the actual result, the `bquote()` function is appropriate. However, the following mathematical expression is not syntactically correct, as a mathematical expression cannot have two relational operators (=) in the one statement. `mu == frac(sum(y[i]),n) == .(meanY)` . Building such an expression is achieved by combining the `bquote()` *function* with a `paste()` *function*.

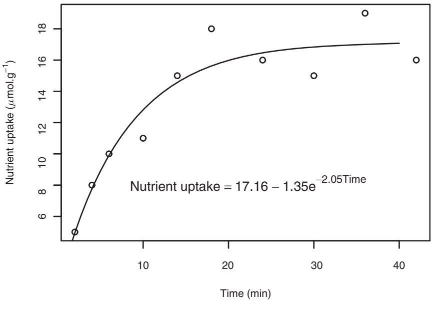
[c] Alternatively, space can be provided by the keyword `phantom(char)`, where `char` is a character whose width is equal to the amount of space required.

```
> set.seed(1)
> Y<-rnorm(100,0,1)
> plot(density(Y),type="l", axes=T,
    ann=F, bty="l", las=1,
    col="grey")
> text(10,0.2,bquote(paste(mu ==
    frac(sum(y[i]),n)) ==
    .(mean(Y))), cex=2)
```



The more observant and discerning reader may have noticed the y-axis label in the `substitute()` example above had a space between the μ and the word 'mol'. Using just the `expression()` *function*, this was unavoidable. A more eligant solution would have been to employ a `expression(paste())` combination.

```
> X<-c(2,4,6,10,14,18,24,30,36,42)
> Y<-c(5,8,10,11,15,18,16,15,19,16)
> n<-nls(Y~SSasymp(X,a,b,c))
> plot(Y~X, type='p', ann=F)
> ...
> mtext(expression(paste("Nutrient
   uptake", " (", mu, "mol.",
   g^-1, ")", sep="")), 2, line=3)
```



5.3.6 Adding axes - `axis()`

Although most of the high-level plotting functions provide some control over axes construction (typically via graphical parameters), finer control over the individual axes is achieved by constructing each axis separately with the `axis()` *function* (see Table 5.10). The `axis()` *function* is defined as:

```
> axis(side, at = NULL, labels = TRUE, tick = TRUE, line = NA,
    pos = NA, outer = FALSE, font = NA, lty = "solid", lwd = 1,
    col = NULL, hadj = NA, padj = NA, ...)
```

```
> set.seed(1)
> X<-rnorm(200,10,1)
> m<-mean(X)
> s<-sd(X)
> plot(density(X), type="l",
    axes=F, ann=F)
> axis(1, at=c(0, m, m+s, m-s,
    m+2*s, m+2*-s, 100), lab=
    expression(NA, mu, 1*sigma,
    -1*sigma, 2*sigma, -2*sigma,
    NA), pos=0, cex.axis=2)
```

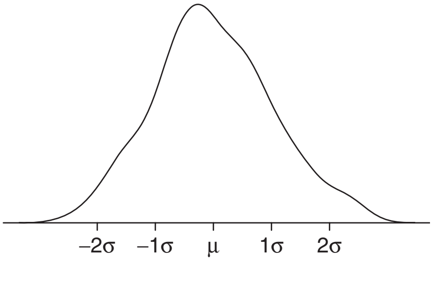
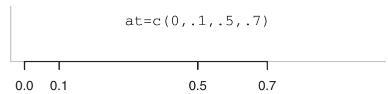
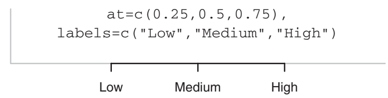
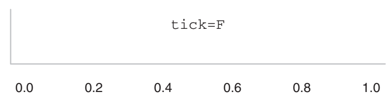
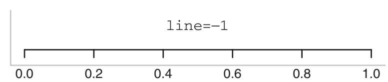
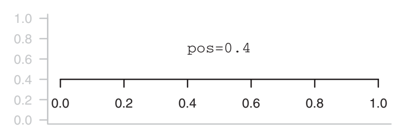
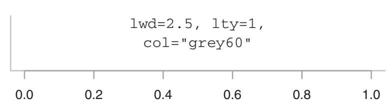
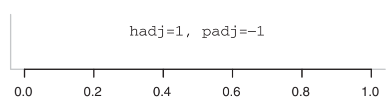


Table 5.10 `axis()` arguments.

Parameter	Description	Examples
`side`	Simplifies which axis to construct. 1=bottom, 2=left, 3=top and 4=right.	
`at`	Where the tick marks are to be drawn. Axis will span between minimum and maximum values supplied.	
`labels`	Specifies the labels to draw at each tickmark. • `TRUE` or `FALSE` - should labels be drawn • a character or expression vector defining the text appear at each tickmark specified by the `at` parameter.	
`tick`	Specifies whether or not (`TRUE` or `FALSE`) the axis line and tickmarks should be drawn.	
`line`	Specifies the number of text lines into the margin to place the axis (along with the tickmarks and labels).	
`pos`	Specifies where along the perpendicular axis, the current axis should be drawn.	
`outer`	Specifies whether or not (`TRUE` or `FALSE`) the axis should be drawn in the outer margin.	
`font`	The font used for the tickmark labels.	
`lwd`, `lty`, `col`	Specifies the line width, style and color of the axis line and tickmarks.	
`hadj`, `padj`	Specifies the parallel and perpendicular adjustment of tick labels to the axis. Units of movement (for example) are `padj=0`: right or top, `padj=1`: left or bottom. Other values are multipliers of this justification.	

5.3.7 Adding lines and shapes within a plot

There are a number of low-level plotting functions for plotting lines and shapes. Individually and collectively, they provide the tools to construct any custom graphic.

The following demonstrations will utilize a dataset by Christensen et al. (1996) that consists of course woody debris (CWD) measurements as well as a number of human impact/land use characteristics for riparian zones around freshwater lakes in North America.

```
> christ <- read.table("christ.csv", header=T, sep=",")
```

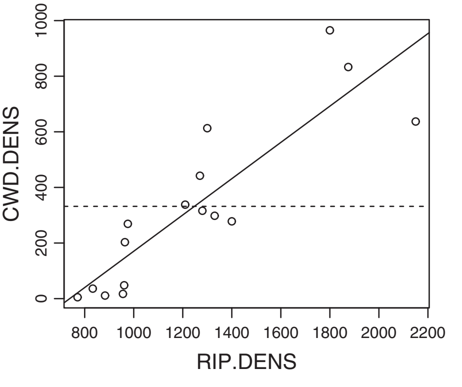
Straight lines - `abline()`

The low-level plotting `abline()` *function* is used to fit straight lines with a given intercept (`a`) and gradient (`b`) or single values for horizontal (`h`) or vertical (`v`) lines. The function can also be passed a fitted linear model (`reg`) or coefficient vector from which it extracts the intercept and slope parameters. The definition of the `abline()` *function* is:

```
> abline(a = NULL, b = NULL, h = NULL, v = NULL, reg = NULL,
    coef = NULL, untf = FALSE, ...)
```

Assessing departures from linearity and homogeneity of variance can be assisted by fitting a linear (least squares regression) line through the data cloud.

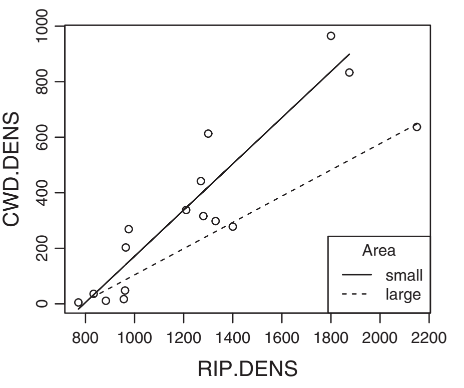
```
> plot(CWD.DENS ~ RIP.DENS,
    data=christ)
> # use abline to add a
> # regression trendline
> abline(lm(CWD.DENS ~ RIP.DENS,
    data=christ))
> # use abline to represent the
> # mean y-value
> abline(h=mean(christ$CWD.DENS),
    lty=2)
```



Lines joining a succession of points - `lines()`

The `lines()` *function* can be used to add lines between points and is particularly useful for adding multiple trends (or non-linear trends, see 'Smoothers') through a data cloud. As with the `points()` *function*, the `lines()` *function* is a generic function whose actions depend on the type of objects passed as arguments. Notably, for simple coordinate vectors, the `points()` and `lines()` *functions* are virtually interchangeable (accept in the type of points they default to). Consequently, a more complex example involving the `predict()` *function* (a function that predicts new values from fitted models) will be used to demonstrate the power of the lines function.

Assessing departures from linearity and homogeneity of variance can be assisted by fitting a linear (least squares regression) line through the data cloud.



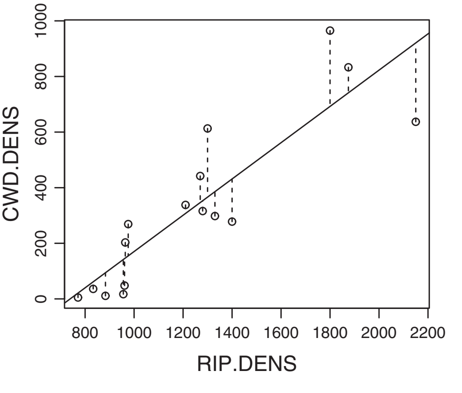

```
> plot(CWD.DENS ~ RIP.DENS,
+  data=christ, type="p")
> # divide the dataset up
> # according to lake size
> area <- cut(christ$AREA,2,
+ lab=c("small", "large"))
> # explore trend for each
> # area separately
> lm.small <- lm(CWD.DENS ~ RIP.DENS, data=christ,
+ subset=area=="small")
> lm.large <- lm(CWD.DENS ~ RIP.DENS, data=christ,
+ subset=area=="large")
> lines(christ$RIP.DENS[area=="small"], predict(lm.small))
> lines(christ$RIP.DENS[area=="large"], predict(lm.large), lty=2)
> legend("bottomright",title="Area",legend=c("small","large"),
+ lty=c(1,2))
```

Lines between pairs of points - `segments()`

The `segments` *function* draws straight lines between points ((`x0`,`y0`) and (`x1`,`y1`)). When each of the coordinates are given as vectors, multiple lines are drawn.

```
> segments(x0, y0, x1, y1, col = par("fg"), lty = par("lty"),
    lwd = par("lwd"), ...)
```

Assessing departures from linearity and homogeneity of variance can also be further assisted by adding lines to represent the residuals (segments that join observed and predicted responses for each predictor). This example also makes use of the `with()` *function* which evaluates any expression or call (in this case the `segments` *function*) in the context of a particular data frame (`christ`) or other environment.



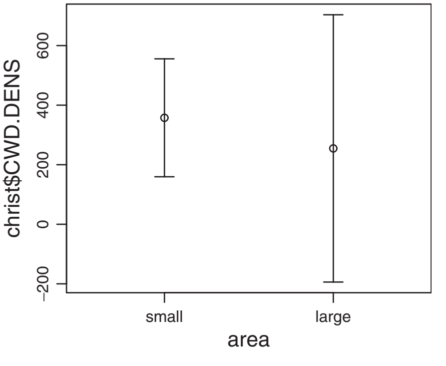

```
> plot(CWD.DENS ~ RIP.DENS,
    data=christ)
```

```
> abline(lm(CWD.DENS ~ RIP.DENS, data=christ))
> # fit the linear model
> christ.lm <- lm(CWD.DENS ~ RIP.DENS, data=christ)
> abline(christ.lm)
> with(christ, segments(RIP.DENS, CWD.DENS, RIP.DENS,
    predict(christ.lm), lty=2))
```

Arrows and connectors - `arrows()`

The `arrows()` *function* builds on the `segments` *function* to add provisions for simple arrow heads. Furthermore, as the length, angle and end to which the arrow head applies are all controllable, the `arrows()` *function* is also particularly useful for annotating figures and creating flow diagrams. The function can also be useful for creating customized error bars (as demonstrated in the following example).

```
> area<-cut(christ$AREA,2,
    lab=c("small","large"))
> library(gmodels)
> s<-tapply(christ$CWD.DENS,
    area,ci)
> plot(christ$CWD.DENS ~ area,
    border="white", ylim=range(s))
> points(1,s$small["Estimate"])
> points(2,s$large["Estimate"])
> with(s, arrows(1,
    small["CI lower"], 1,
    small["CI upper"], length=0.1,
    angle=90, code=3))
> with(s, arrows(2,
    large["CI lower"], 2,
    large["CI upper"], length=0.1,
    angle=90, code=3))
```



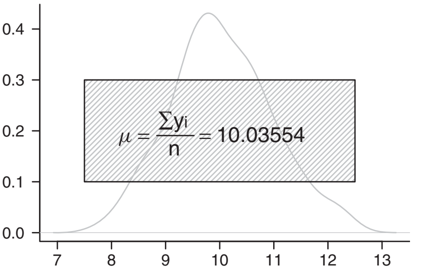
Rectangles - `rect()`

The `rect()` *function* draws rectangles from left-bottom, right-top coordinates that can be filled with solid or striped patterns (according to the line type, width, angle, density and color):

```
> rect(xleft, ybottom, xright, ytop, density = NULL, angle = 45,
    col = NA, border = NULL, lty = par("lty"), lwd = par("lwd"),
    ...)
```

The main use of rectangles is to produce frames for items within plots.

```
> set.seed(1)
> Y<-rnorm(200,10,1)
> plot(density(Y),type="l",axes=T,
    ann=F, bty="l", las=1,
    col="grey")
> rect(7.5,.1,12.5,.3, ang=45,
    density=20, col="grey",
    border="black")
> text(10,0.2,bquote(paste(mu ==
    frac(sum(y[i]),n)) ==
    .(mean(Y))),cex=2)
```



Irregular shapes between a succession of points - `polygon()`

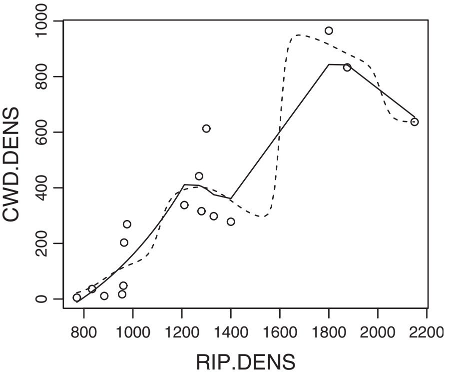
Given a vector of x coordinates and a corresponding vector of y coordinates, the `polygon()` *function* draws irregular shapes:

```
> polygon(x, y = NULL, density = NULL, angle = 45, border = NULL,
    col = NA, lty = par("lty"), ...)
```

Smoothers

Smoothing functions can be useful additions to scatterplots, particularly for assessing (non)linearity and the nature of underlying trends. There are many different types of smoothers see section 8.3 and Table 8.2.

Smoothers are added to a plot by first fitting the smoothing function (`loess()`, `ksmooth()`) to the data before plotting the values predicted by this function across the span of the data.



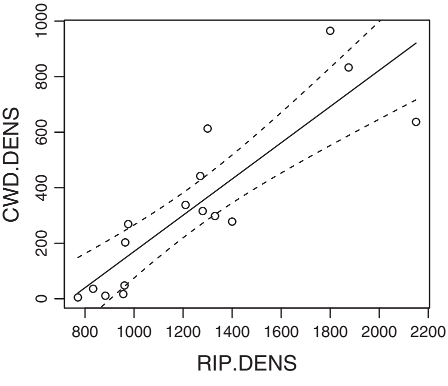

```
> plot(CWD.DENS ~ RIP.DENS,
    data=christ)
> # fit the loess smoother
> christ.loess<-loess(CWD.DENS ~
    RIP.DENS, data=christ)
> # created a vector of the sorted
> # X values
> xs<-sort(christ$RIP.DENS)
> lines(xs, predict(christ.loess, data.frame(RIP.DENS=xs)))
> # fit and plot a kernel smoother
> christ.kern <- ksmooth(christ$RIP.DENS, christ$CWD.DENS,
    "norm", bandwidth=200)
> lines(christ.kern, lty=2)
```

Confidence ellipses - `matlines()`[d]

The `matlines()` *function*, along with the similar `matplot()` and `matpoints()` *functions* plot multiple columns of matrices against one another, thereby providing a convenient means to plot predicted trends and confidence intervals in a single statement.

Confidence bands are added by using the value(s) returned by a `predict()` *function* as the second argument to the `matlines()` *function*.

```
> plot(CWD.DENS ~ RIP.DENS,
    data=christ)
> christ.lm<-lm(CWD.DENS ~
    RIP.DENS, data=christ)
> xs<-with(christ,
    seq(min(RIP.DENS),
    max(RIP.DENS), l=1000))
> matlines(xs,
    predict(christ.lm,
    data.frame(RIP.DENS=xs),
    interval="confidence"),
    lty=c(1,2,2), col=1)
```



5.4 Interactive graphics

The majority of plotting functions on the majority of graphical devices operate by sending all of the required information to the device at the time of the call - no additional information is required or accepted from the user. The display devices (`X11()`, `windows()` and `quartz()`) however, also support a couple of functions designed to allow interactivity between the user and the current plotting region.

5.4.1 Identifying points - `identify()`

The `identify()` *function* allows the user to label points interactively. After issuing the identify() *function* with arguments corresponding to the x and y axis vectors, R awaits mouse input in the form of left mouse button clicks in the plotting region of the current display device. Each time the left mouse button is clicked on the display device, the coordinates of the mouse pointer are retrieved and the nearest data points (determined by comparing the mouse pointer coordinates to the point coordinates supplied as arguments) are labelled. A right mouse click ('ESC' on MAC OS X) terminates the function which returns a vector of point indices. In its simplest form, `identify()` *function* can be used to identify potentially problematic observations. Additional arguments can be supplied to provide finer control over the relative positioning and text of the labels.

[d] Note, the same could be achieved via three seperate `lines()` calls.

5.4.2 Retrieving coordinates - `locator()`

The `locator()` *function* returns the coordinates of the mouse pointer each time the left mouse button is clicked on the display device. A right mouse click on the display ('ESC' on MacOSX) terminates the function which returns a list of x, y coordinates. Alternatively, the function can be supplied with an argument indicating the number of points to locate (`n`). Furthermore, if the `type=` parameter is set to one of the plotting point types, the points will be echoed onto the current plotting region. The `locator()` *function* provides a convenient way to construct mock data sets, trace objects as well as construct simple maps.

5.5 Exporting graphics

Graphics can also be written to several graphical file formats via specific graphics *devices* which oversee the conversion of graphical commands into actual graphical elements. In order to write graphics to a file, an appropriate graphics device must first be 'opened'. A graphics device is opened by issuing one of the device functions listed below and essentially establishes the devices global parameters and readies the device stream for input. Opening such a device also creates (or overwrites) the nominated file. As graphical commands are issued, the input stream is evaluated and accumulated. The file is only written to disk when the device is closed via the `dev.off()` *function*.

Note that as the capabilities and default global parameters of different devices differ substantially, some graphical elements may appear differently on different devices. This is particularly true of dimensions, locations, fonts and colors.

5.5.1 Postscript - `poscript()` and `pdf()`

Postscript is actually a programming language that defines both the nature of the content and exactly how the content should be displayed or printed on a page. As a result, postscript is device independent and scalable to any size and is therefore the preferred format of most publishers. Whilst there are many other arguments that can be passed to the `postscript()` *function*, common use is as follows:

```
> postscript(file, family, fonts = NULL, width, height,
    horizontal, paper)
```

where `file` is a file name (and path), `font` and `family` declare all the fonts required in the device, `width` and `height` define the dimensions (in inches) of the graphic, `paper` defines the size of the printer paper (or 'special' for graphics in which width and height is defined) and `horizontal` determines the orientation of the graphic relative to the paper type.

Like postscript, pdf (Portable Document Format) files contain information on exactly how the printed page should appear. Pdf documents can also contain a great deal of additional information on how the information should behave in different contexts. Such 'advanced' postscript features are largely designed to enhance the capabilities of documents displayed on screens and are therefore rarely utilized from R. Importantly, unlike R's postscript device, the pdf device does not embed a prologue of font metrics, and thus only fonts that can be assumed to be present on the target devices (printers and other computers) should be used.

5.5.2 Bitmaps - `jpeg()` and `png()`

R also supports a range of bitmap file formats, the range of which depends on the underlying operating system and the availability of external applications.

```
> jpeg(filename, width = 480, height = 480, units = "px",
    pointsize = 12, quality = 75, bg = "white", res = NA, ...)
```

where `filename` defines the name of the file (including path), `width` and `height` define the dimensions of the graphic (in pixels) and quality defines the compression quality (100 indicates no compression). The graphical capabilities of the bitmap devices are largely tied to the default display device.

5.5.3 Copying devices - `dev.copy()`

Alternatively, graphics can be exported to file by copying the contents of one device (such as a display device) to another device (such as a file device) using the `dev.copy()` *function*.

5.6 Working with multiple graphical devices

It is possible to have multiple graphical devices open simultaneously. However, only one device can be active (receptive to plotting commands) at a time. Once a device has been opened (see section 5.5), the device object is given an automatically iterated reference number in the range of 1 to 63. Device 1 will always be a `null` device that cannot accept plotting commands and is essentially just a placeholder for the device counter. The set of functions for managing multiple devices are described in Table 5.11. To appreciate the workings of these functions, first create multiple display devices. To do so, issue one of the commands listed below (the one appropriate for your system) multiple times:

Windows	MacOSX[e]	Linux
`windows()`	`quartz()`	`X11()`

Note that the device title bars will indicate the device reference number as well as whether the device is currently active or inactive. The last one created will be active.

[e] The default graphics device for MacOSX is `X11`, however, many prefer `quartz`.

Table 5.11 Functions for managing multiple graphics devices.

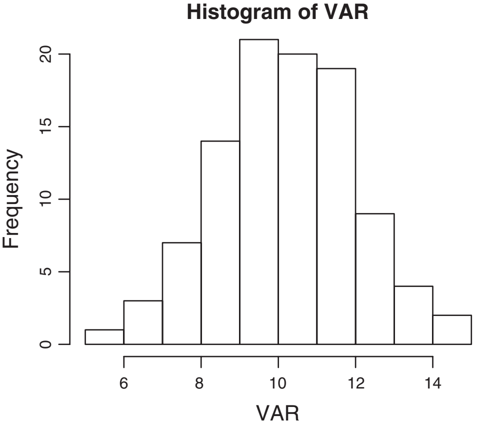
Function	Description	Example
`dev.list()`	Returns the numbers of open devices (with device types as column headings)	X11 X11 2 3
`dev.cur()`	Returns the number (and name) of the currently active device	X11 3
`dev.next()`	Returns the number (and name) of the next available device after the device specified by the `which=` argument (after current if `which=` absent)	X11 2
`dev.prev()`	Returns the number (and name) of the previous available device after the device specified by the `which=` argument (before current if `which=` absent)	X11 2
`dev.set()`	Makes the device specified by the `which=` argument the currently active device and returns the number (and name) of this device. If `which=` argument absent, it is set to the next device.	X11 2
`dev.off()`	Closes the device specified by the `which=` argument (or current device if `which=` argument absent), makes the next device active and returns the number (and name) of this device.	X11 3

5.7 High-level plotting functions for univariate (single variable) data

5.7.1 Histogram

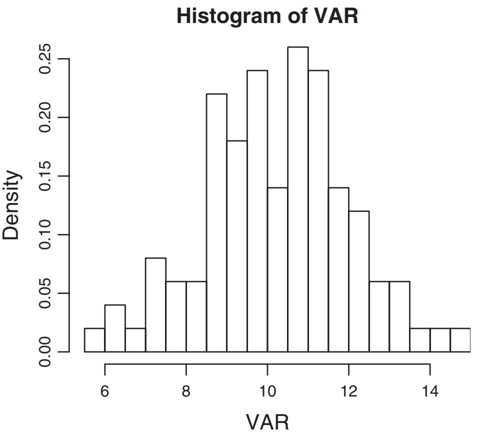
Histograms are useful at representing the distribution of observations for large (> 30) sample sizes.

```
> set.seed(1)
> VAR <- rnorm(100,10,2)
> hist(VAR)
```



The number or size of the bins can be controlled by passing respectively a single number or vector of bin breakpoints with the `breaks=` argument[f]. Specifying the `probability=T` argument will express the number counts in each bin as a density (probability) rather than as a frequency.

```
> hist(VAR, breaks=18,
    probability=T)
#OR equivalently in this case
> hist(VAR, breaks=seq(5.5,15,
    by=.5), probability=T)
```

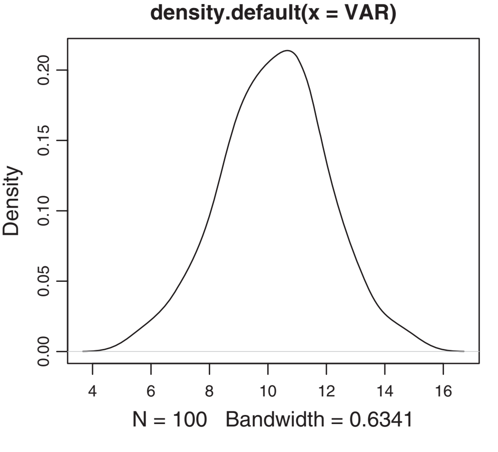


5.7.2 Density functions

Probability density functions are also useful additions or alternatives to histograms as they further assist in describing the patterns of the underlying distribution. Typical kernel density functions fit a series of kernels (symmetric probability functions) to successive subsets (windows) of the ordered dataset from which new estimates of the observations are calculated. The resolution and texture (smoothness) of the density function is controlled by a smoothing parameter which essentially defines the width of the kernel window.

A density function can be plotted using the `density()` *function* as an argument to the high-level overloaded `plot()` *function*.

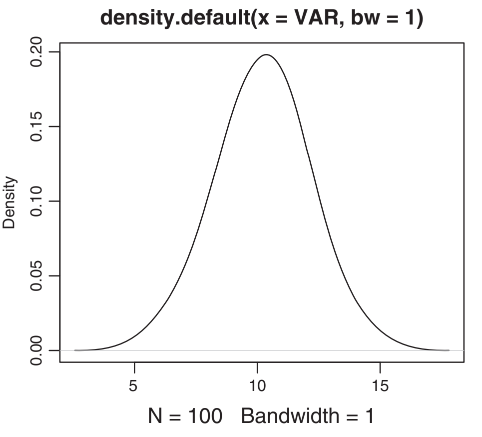
```
> plot(density(VAR))
```



[f] It is also possible to pass a function that computes the number of breaks or the name of a breaking algorithm.

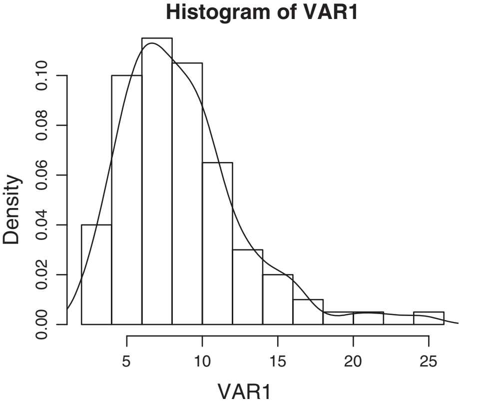
The type of smoothing kernel (normal or gaussian by default) can be defined by the `kernel=` *argument* and the degree of smoothing is controlled by the `bw=` (bandwidth) *argument*. The higher the smoothing bandwidth, the greater the degree of smoothing.

```
> plot(density(VAR, bw=1))
```



The density function can also be added to a histogram using the `density()` *function* as an argument to a the low-level `lines()` *function*.

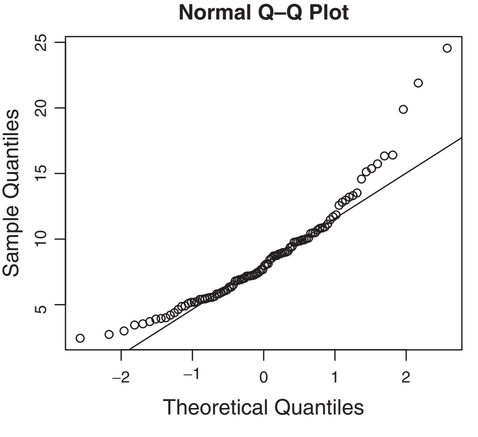
```
> set.seed(1)
> VAR1 <- rlnorm(100,2,.5)
> hist(VAR1, prob=T)
> lines(density(VAR1))
```



5.7.3 Q-Q plots

Q-Q normal plots can also be useful at diagnosing departures from normality by comparing the data quantiles[g] to those of a standard normal distribution. Substantial deviations from linearity, indicate departures from normality.

```
> qqnorm(VAR1)
> qqline(VAR1)
```



[g] Quantiles are a regular spacing of points throughout an ordered data set.

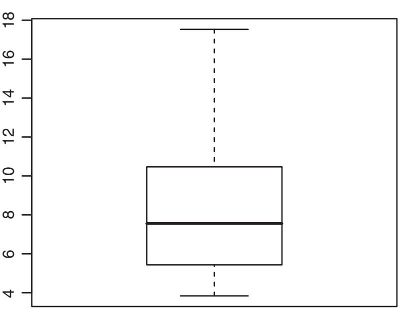
5.7.4 Boxplots

For smaller sample sizes, histograms and density functions can be difficult to interpret. Boxplots (or box-and-whisker plots) provide an alternative means of depicting the location (average), variability and shape of the distribution of data. The dimensions of a boxplot are defined by the five-number summaries (minimum value, lower quartile (Q1), median (Q2), upper quartile (Q3) and maximum value - each representing 25%) of the data (see Figure 5.5).

Recall that boxplots are typically used to explore the distributions of small samples. The volatility of quantiles from small samples offers little confidence in any single component of a boxplot. Hence, the key characteristic of a boxplot that is indicative of a departure from normality is that *each segment* of the boxplot gets progressively larger (or smaller). Only in such a circumstance, could you be confident that the sample could not have come from a normal distribution of values. The following boxplots provide an illustration of such a departure from normality (log-normal boxplot).

Univariate boxplots are generated by passing a vector to the `boxplot()` *function.*

```
> set.seed(6)
> VAR2<-rlnorm(15,2,.5)
> boxplot(VAR2)
```



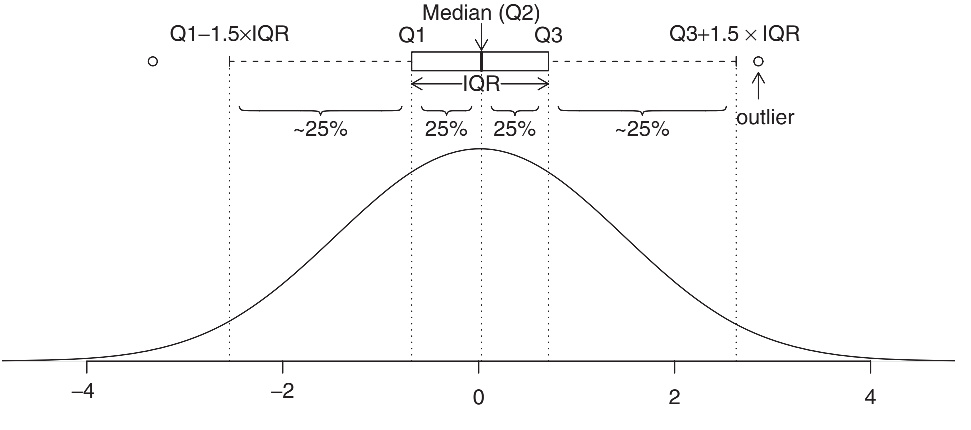
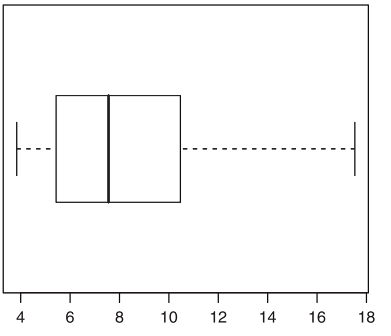


Fig 5.5 Boxplot of a standard normal distribution (mean=0, sd=1).

The `horizontal=T` *argument* is used to produce horizontally aligned boxplots

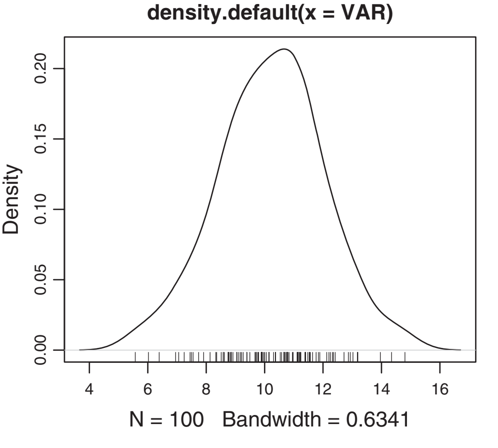
```
> boxplot(VAR2, horizontal=T)
```



5.7.5 Rug charts

Another representation of the data that can be added to existing plots is a rug chart that displays the values as a series of ticks on the axis. Rug charts can be particularly useful at revealing artifacts in the data that are "smoothed" over by histograms, boxplots and density functions.

```
> set.seed(1)
> VAR <- rnorm(100,10,2)
> plot(density(VAR))
> rug(VAR,side=1)
```



5.8 Presenting relationships

When two or more continuous variables are collected, we often intend to explore the nature of the relationships between the variables. Such trends can be depicted graphically in scatterplots. Scatterplots display a cloud of points, the coordinates of which correspond to the values of the variables that define the horizontal and vertical axes.

5.8.1 Scatterplots

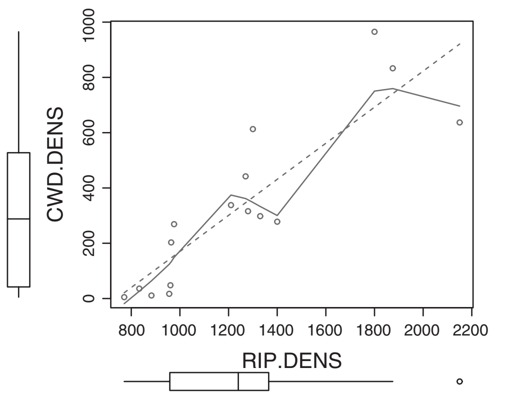
Although scatterplots do not formally distinguish between response (dependent) and predictor (independent) variables, when such distinctions occur, independent variables are conventionally plotted along the horizontal (x) axis.

Scatterplots are used prior to analyses to help assess the suitability of the data to particular analytical procedures. Of particular importance are the insights they provide into the linearity and patterns of variability of trends. They are also presented post analysis as summaries of the trends and analyses.

The following demonstrations will again utilize the course woody debris (CWD) dataset by Christensen et al. (1996). As previously demonstrated, scatterplots can generated with the `plot()` *function*. Additional features (such as trendlines, smoothers and other features that assist in assessing departures from linearity and homogeneity of variance) can then be added with various low-level plotting functions.

To facilitate all of these diagnostic features as well as marginal boxplots, the high-level `scatterplot()` *function* (`car` *package*) is very useful. Note, the `scatterplot()` *function* fits a lowess rather than loess smoother.

```
> library(car)
> scatterplot(CWD.DENS ~
    RIP.DENS, data=christ)
```



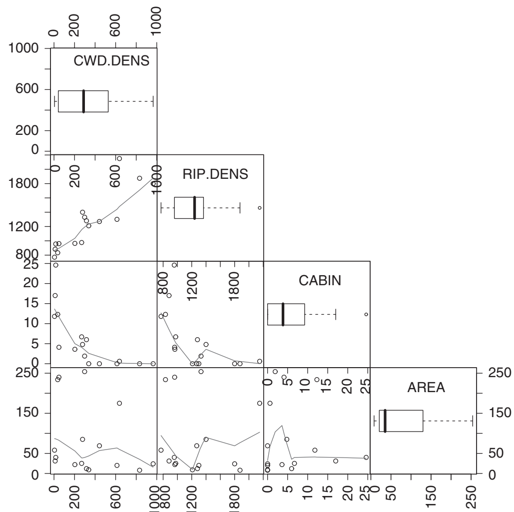
Scatterplot matrices (SPLOMS)

Scatterplot matrices display a panel of scatterplots between each pair of variables when there are three or more continuous variables. A given variable makes up the x-axis of each of the panels up the column and the y-axis of each of the panels along the row. The diagnal panels are often populated with univariate plots such as boxplots, histograms or density functions. The upper right panels are a mirror of the lower left panels. There are a few high-level plotting functions for producing scatterplot matrices:

- the `pairs()` *function* is an extension of the regular `plot()` *function*
  Different functions can be applied to the lower, upper and diagonal panels of the grid. A lowess smoother is supported by the `panel.smooth` *function*. It is also possible to define alternative functions. This example illustrates the application of horizontal boxplots into the diagonal panels. Since, the upper panels are a mirror of the lower panels, the upper panels can be removed with by setting the `upper.panel=` parameter to NULL.

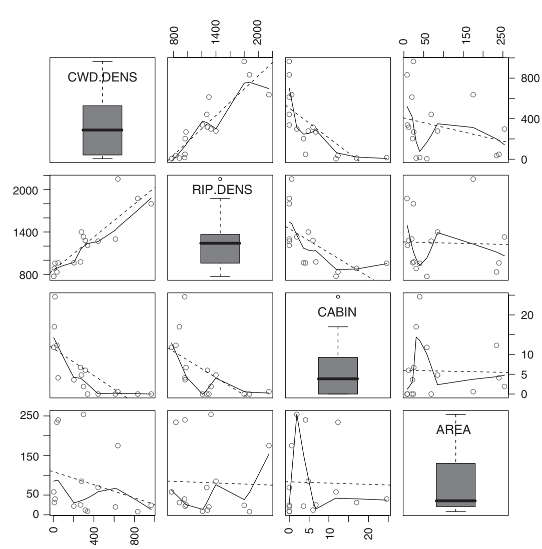
```
> # define a boxplot panel function
> panel.bxp <- function(x, ...)
> \{
> usr <- par("usr"); on.exit(par(usr))
```

```
> par(usr = c(usr[1:2],0,2))
> boxplot(x, add=TRUE, horizontal=T)
> \}
> pairs(~CWD.DENS + RIP.DENS + CABIN + AREA, data=christ,
    lower.panel=panel.smooth, diag.panel=panel.bxp,
    upper.panel=NULL, gap=0)
```



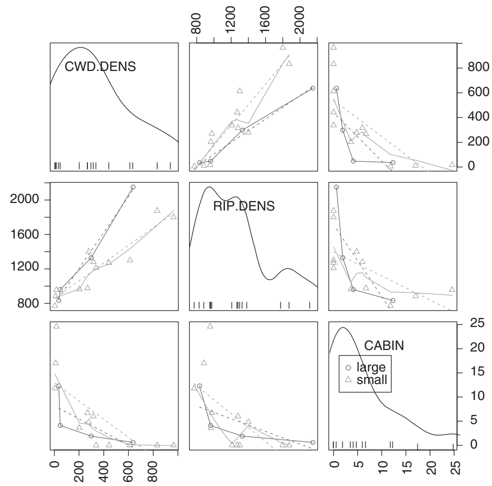
- the `scatterplot.matrix()` *function* (`car` *package*) is an extension of the regular `scatterplot()` *function*.

```
> library(car)
> scatterplot.matrix(~CWD.DENS + RIP.DENS + CABIN + AREA,
    data=christ, diag="boxplot")
```



The `scatterplot.matrix()` *function* can differentiate trends for different levels (groups) of a categorical variable. To illustrate, we will use the `cut()` *function* to convert the `AREA` vector into a categorical variable with two levels (small and large).

```
> scatterplot.matrix(~CWD.DENS + RIP.DENS + CABIN,
    groups=cut(christ$AREA,br=2, lab=c("small","large")),
    by.groups=T, data=christ, diag="density")
```



3D scatterplots

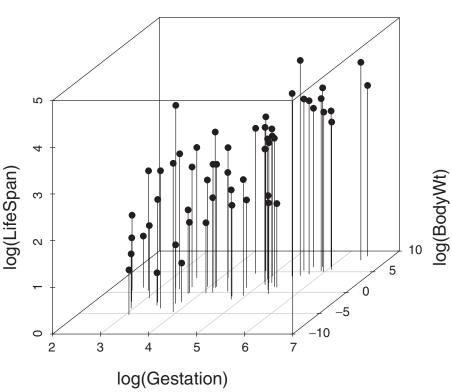
Three dimensional scatterplots can be useful for exploring multivariate patterns between combinations of three or more variables. To illustrate 3D scatterplots in R, we will make use of a dataset by Allison and Cicchetti (1976) that compiles sleep, morphology and life history characteristics 62 species of mammal along with predation indices.

```
> allison <- read.table("allison.csv", header=T, sep=",")
```

- the `scatterplot3d` *function* (`scatterplot3d` *package*)

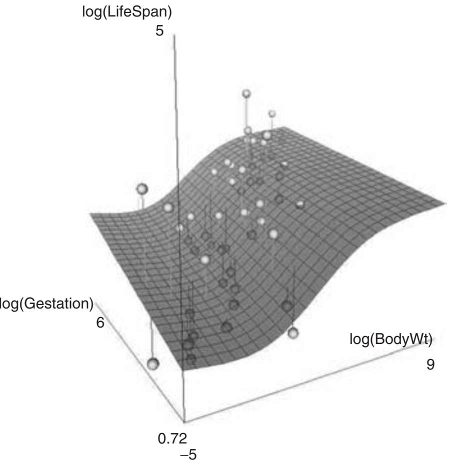
```
> library(scatterplot3d)
> with(allison,
    scatterplot3d(log
    (Gestation), log(BodyWt),
    log(LifeSpan), type="h",
    pch=16))
```

The `type="h"` *parameter* specifies that points should be connected to the base by a line and the `pch=16` *parameter* specifies solid points. All variables were expressed as their natural logarithms using the `log()` *function*.



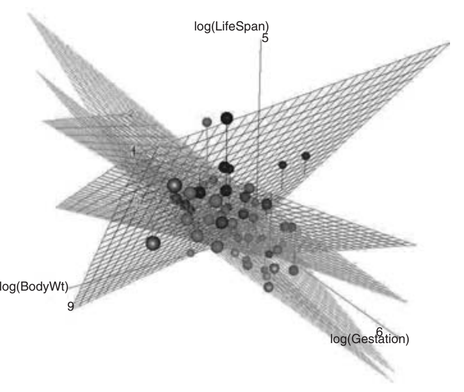
- the `scatter3d` *function* (`Rcmdr` *package*) displays rotating three dimensional plots.

```
> library(Rcmdr)
> with(allison,
    scatter3d(log(Gestation),
    log(LifeSpan), log(BodyWt),
    fit="additive", rev=1))
```



The `fit=` *parameter* specifies the form of surface to fit through the data. The option selected (`"additive"`) fits an additive non-parametric surface through the data cloud and is useful for identifying departures from multivariate linearity. The `rev=` *parameter* specifies the number of full revolutions the plot should make. Axes rotations can also be manipulated manually by dragging the mouse over the plot.

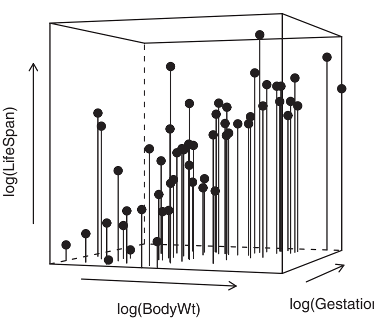
```
> library(Rcmdr)
> with(allison,
    scatter3d(log(Gestation),
    log(LifeSpan), log(BodyWt),
    fit="linear", parallel=F,
    groups=factor(Predation),
    fill=F))
```



The `parallel=F` *argument* specifies that separate surfaces are generated for each of the levels in the factorial variable specified by the `groups=` *argument*. In this case, the `factor()` *function* was used to convert the numeric predation vector to a factor. The `fill=F` *argument* specifies that the surfaces should not be filled in.

- the `cloud()` *function* (`lattice` *package*). Refer to section 5.11 for more information on trellis graphics.

```
> library(lattice)
> cloud(log(LifeSpan) ~
    log(BodyWt) *
    log(Gestation),
    data=allison, pch=16,
    type=c("p","h"),
    screen=c(x=-90, y=-20),
    zlab=list(rot=90))
```



The data are specified as a formula of the format `z~x*y`. The `type=c("p","h")` *argument* specifies that both points and connected lines should be used. The `screen=` *argument* specifies the amount of axes rotation for the x, y and z axes. The `zlab` *list* specifies that the z axis label should be rotated 90 degrees.

5.9 Presenting grouped data

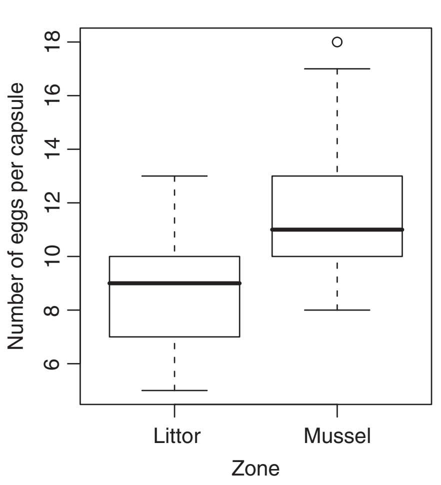
Data for which a response has been measured from two or more groups of sampling units are summarised graphically by estimates of location (such as mean and median) and spread (standard error and standard deviation). As with summaries of relationships, graphical summaries for grouped data serve as both exploratory data analysis tools as well as visual representations of statistical analyses.

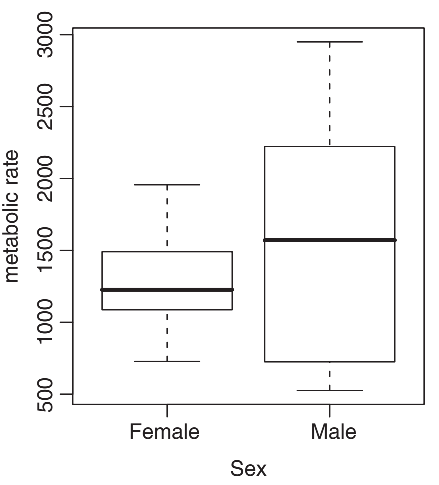
5.9.1 Boxplots

Plotting multiple boxplots side by side (one for each level of a factorial variable), provides a useful means of examining homogeneity (equal) of variance assumptions. To illustrate boxplots, we will reproduce Figure 4.5 from Quinn and Keough (2002) using data sets from Ward and Quinn (1988) and Furness and Bryant (1996).

```
> ward<-read.table("ward.csv",
    header=T, sep=",")
> boxplot(EGGS~ZONE, data=ward,
    ylab="Number of eggs per
    capsule", xlab="Zone")
```

```
> furness<-read.table("furness
    .csv", header=T, sep=",")
> boxplot(METRATE~SEX, data=
    furness, ylab="metabolic
    rate", xlab="Sex")
```



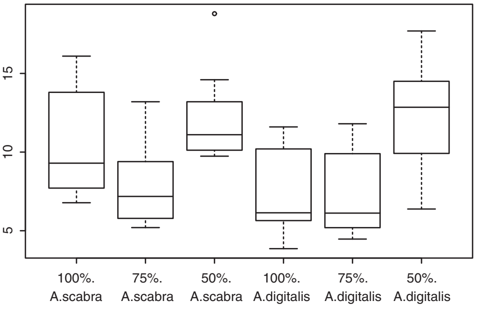


5.9.2 Boxplots for grouped means

Technically, the normality and homogeneity of variance assumptions pertain to the residuals (difference between values observed and those predicted by the proposed model) and thus the model replicates. For multi-factor analysis of variance designs, the appropriate replicates for a hypothesis test are usually the individual observations from each combination of factors. Hence, boxplots should also reflect this level of replication.

To illustrate, a data set introduced in Box 11.2 of Sokal and Rohlf (1997) on the oxygen consumption of two species of limpets under three seawater concentrations will be used.

```
> limpets <-read.table("limpets
   .csv", header=T, sep=",")
> boxplot(O2~SEAWATER*SPECIES,
   limpets)
```

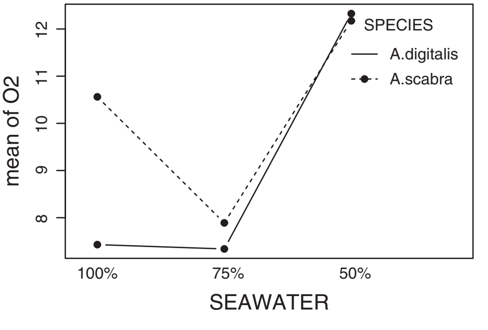


5.9.3 Interaction plots - means plots

Interactions are outcomes in which the effects of one factor are dependent on the levels of other factor(s). That is, the effect of one factor is not consistent across all levels of the other factors. Interaction plots depict the mean response value of each combination of factor levels (groups) and are therefore useful for interpreting interactions.

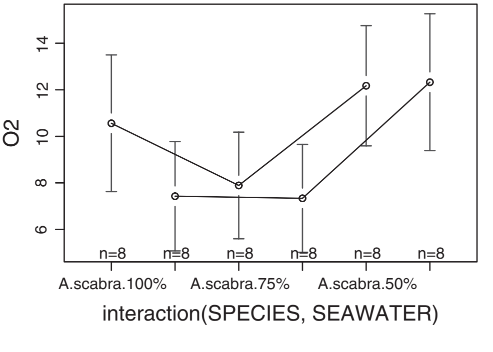
- the `interaction.plot()` *function* (`car` *package*).

```
> library(car)
> limpets <-read.table
   ("limpets.csv", header=T,
   sep=",")
> with(limpets, interaction.
   plot(SEAWATER, SPECIES,
   O2, type="b", pch=16))
```



- the `plotmeans()` *function* (`gplots` *package*)

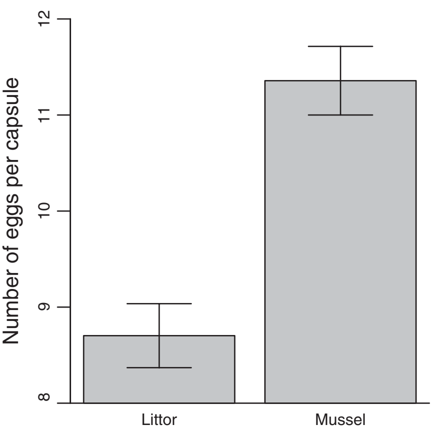
```
> library(gplots)
> plotmeans(O2 ~ interaction
   (SPECIES, SEAWATER),
   limpets, connect=list
   (c(1,3,5), c(2,4,6)))
```



5.9.4 Bargraphs

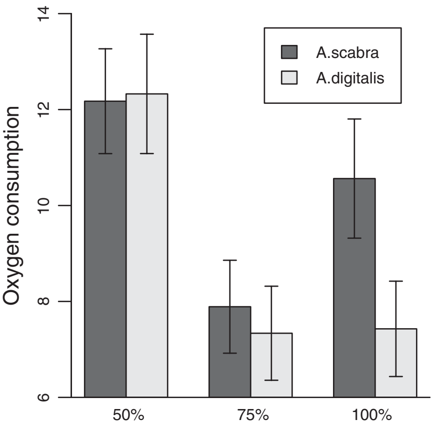
Bargraphs are plots where group means are represented by the tops of bars or columns. Pure statisticians (who refer to these plots as 'dynamite plots') argue that bars should only be used to represent frequencies (totals) and are not appropriate for representing means (since the body of the bar has no logical interpretation). Furthermore, they implicitly assume parametric assumptions and can misleadingly conceal the true nature of the data. Consequently, there are no high-level bargraph plotting functions (and it is unlikely that the R Core Development Team would ever support such a function). Such professionals prefer boxplots (see section 5.9.2), means plots (means represented by points) and violin plots (see section 5.9.5). Nevertheless, biologist often find bargraph useful graphical summaries and they do provide a greater area for displaying colors and shading to distinguish different treatment combinations. Such is the power of R, they are relatively simple to construct using a series of low-level plotting functions.

```
> means<-with(ward, tapply(EGGS,
    ZONE, mean))
> sds <-with(ward, tapply(EGGS,
    ZONE, sd))
> ns<-with(ward, tapply(EGGS, ZONE,
    length))
> ses <- sds/sqrt(ns)
> b<-barplot(means, ylim=c(min(pretty
    ( means - ses)), max(pretty
    (means+ses))), xpd=F,
    ylab="Number of eggs per capsule")
> arrows(b, means+ses, b, means-ses,
    angle=90, code=3)
> box(bty="l")
```



Similarly, multifactor bargraphs can also be constructed from first principles.

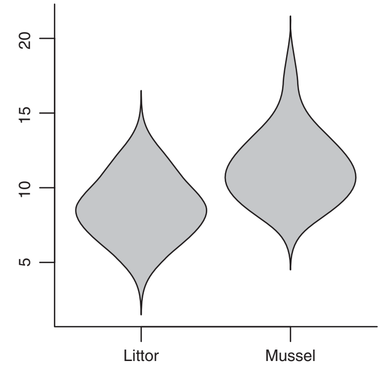
```
> means<-with(limpets, tapply(O2,
    list(SPECIES,SEAWATER), mean))
> sds <-with(limpets, tapply(O2,
    list(SPECIES,SEAWATER), sd))
> ns<-with(limpets, tapply(O2,
    list(SPECIES,SEAWATER), length))
> ses <- sds/sqrt(ns)
> b<-barplot(means, ylim=c(min(pretty
    ( means-ses)), max(pretty
    (means+ses))), beside=T, xpd=F,
    ylab="Oxygen consumption",
    legend.text=rownames(means))

> arrows(b,means+ses,b,means-ses,
    angle=90, code=3,length=0.05)
> box(bty="l")
```



5.9.5 Violin plots

Violin plots are an alternative to boxplots and bargraphs for representing the characteristics of multiple samples.

```
> library(UsingR)
> simple.violinplot(EGGS~ZONE, ward,
+ col="gray", bw="SJ")
> box(bty="l")
```



5.10 Presenting categorical data

Associations between two or more categorical variables (such as those data modelled by contingency tables and log-linear modelling) can be summarized graphically by mosaic and association plots. To illustrate graphical summaries for categorical data, we will use a data set by Young and Winn (2003) in which encountered eels were cross-classified according to species and location (grass beds, sand/rubble or bordering the previous two).

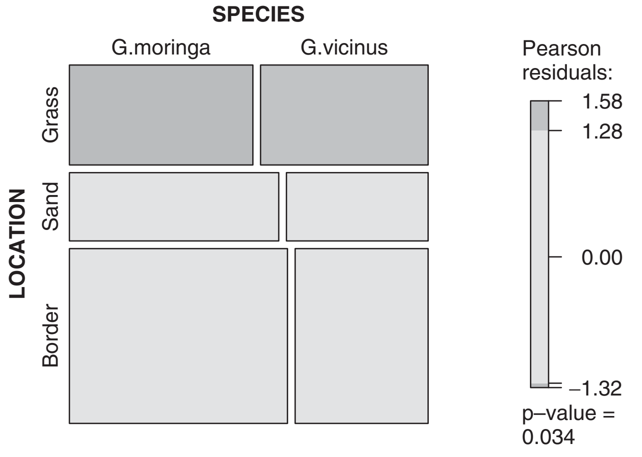
```
> eels <-read.table("eels.csv", header=T, sep=",")
> eels.xtab <- xtabs(COUNT ~ LOCATION + SPECIES, eels)
```

5.10.1 Mosaic plots

Mosaic plots represent each of the various cross-classifications as a mosaic of rectangles, the sizes of which are proportional to the observed frequencies[h]. In addition, the rectangles can be shaded to reflect the magnitudes and significance[i] of the residuals, thereby providing an indication of which cross-classifications contribute to a lack of independence.

```
> library(vcd)
> strucplot(eels.xtab, gp=shading_max)
```

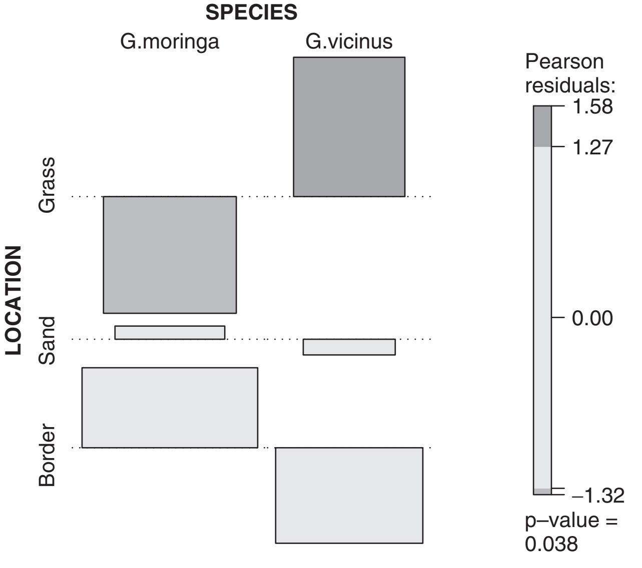
[h] Actually, the widths and heights are proportional to the marginal and conditional percentages respectively.
[i] Significance is determined via a permutation test, and thus exact probabilities differ from run to run.



5.10.2 Association plots

Association plots depict cross-classifications as rectangles whose heights reflect the relative sizes and polarity of Pearson residuals and whose areas reflect the raw residuals. As with mosaic plots, shading can be used to reflect the magnitude and significance of residuals.

```
> assoc(eels.xtab, gp=shading_max)
```



5.11 Trellis graphics

Trellis graphics provide the means of plotting the trends amongst a set of variables separately according to the levels of other variables and can therefore be more

Table 5.12 Incomplete list of high-level lattice (Trellis) plotting functions.

Plotting function	Description
Univariate	
`densityplot()`	Conditional kernel smoothing density plot
`histogram()`	Conditional histograms
`dotplot()`	Conditional dotplots
Bivariate	
`xyplot()`	Conditional scatterplots
`qq()`	Conditional quantile-quantile plots
`qqmath()`	Conditional qq-normal plots
`barchart()`	Conditional barcharts
`bwplot()`	Conditional boxplots
Multivariate	
`cloud()`	Conditional 3D scatterplots
`splom()`	Matrix of scatterplots

appropriate for exploring trends within grouped data[j]. The separate trends are presented in multiple panels within a grid and/or as different plotting symbols within plots. Many of the high-level plotting functions described above have trellis equivalents (see Table 5.12), all of which are provided by the `lattice` *package*.

Trellis (`lattice`) graphics provide a richer, more customizable set of graphical procedures that can also be easily modified and committed multiple times to multiple devices. The cost however, is that they are substantially more complex. An excellent source of reference on trellis graphics (and graphics in general) within R is Murrell (2005).

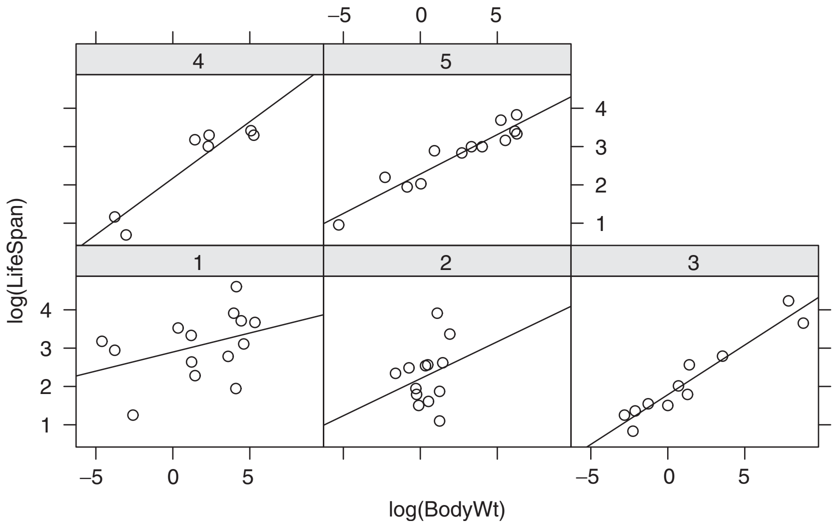
To illustrate trellis graphics we will again make use of the Allison and Cicchetti (1976) data in which the amount of sleep time, morphology and predation risks were compiled for 62 species of mammal. Predation risk was measured on a scale of 1 through 5 where 1 is very low and 5 is very high.

```
> allison <- read.table("allison.csv", header=T, sep=",")
```

A basic conditioning plot, might depict the relationship between the life span of mammals against body mass separately for each level of predation. Such a plot could be constructed using the `xyplot()` *function*. Grouped data can be specified in one of two ways. Firstly, if the plotting formula contains a factor vector separated by a |, separate panels are constructed for each level of the factor. The `xyplot()` *function* introduces the `type="r"` *argument* which specifies regression trendlines.

[j] Such as those data modelled by blocking and repeated measured designs.

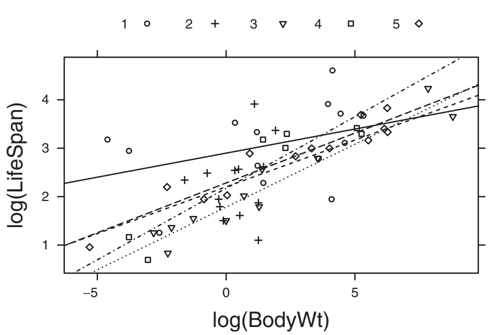
```
> xyplot(log(LifeSpan)~log(BodyWt) | factor(Predation),
    data=allison, type=c("p","r"))
```



It is clear that the relationship between longevity and body mass is conditional on the level of predation risk.

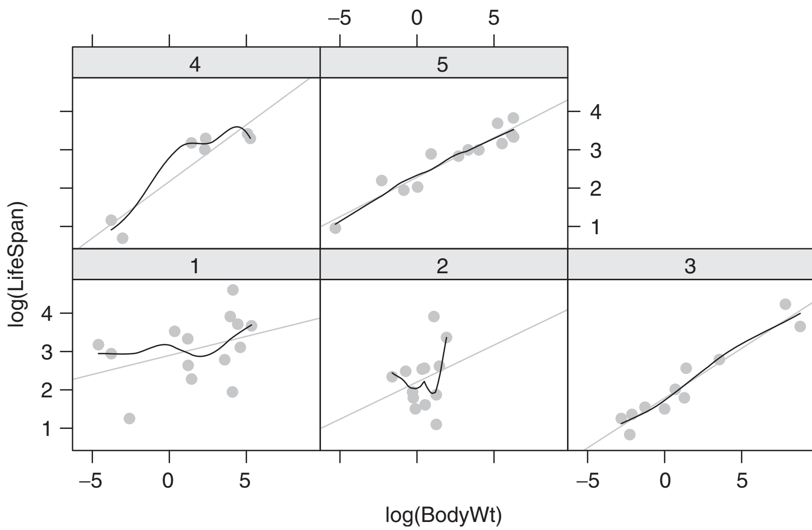
Alternatively, each of the trends can be included on the one plot by passing the factorial vector as a `group=` *argument*.

```
> xyplot(log(LifeSpan)~
    log(BodyWt), groups=factor
    (Predation), data=allison,
    type=c("p","r"),
    auto.key=list(columns=5))
```



Additional graphical features can be added to the panels using the `panels=` *argument*. This argument accepts a range of predefined functions, as well as user defined functions to achieve specific results and is called by the plotting function for each panel in the lattice.

```
> myFunc<-function(x,y) a<-lm(y~x); panel.points(x,y, pch=16,
    col="grey"); panel.abline(a,col="grey"); panel.loess(x,y)
> xyplot(log(LifeSpan) ~ log(BodyWt) | factor(Predation),
    data=allison, panel=myFunc)
```



Accordingly, there are also lattice equivalents of most of the low level plotting functions described in section 5.3. Typically, these functions are called by the name of the basic low level function name with a `panel.` prefex.

Unlike the basic plotting system described earlier, lattice plots are not a biproduct of the plotting functions. Instead, the output is returned by the function. Consequently, an entire trellis can be stored as an object and subsequently updated (modified) using the overloaded `update()` *function*. The overall graphic is not committed until the object is printed[k].

```
> myPlot<-xyplot(log(LifeSpan) ~ log(BodyWt) |
  factor(Predation), data=allison, panel=myFunc)
> print(myPlot)
```

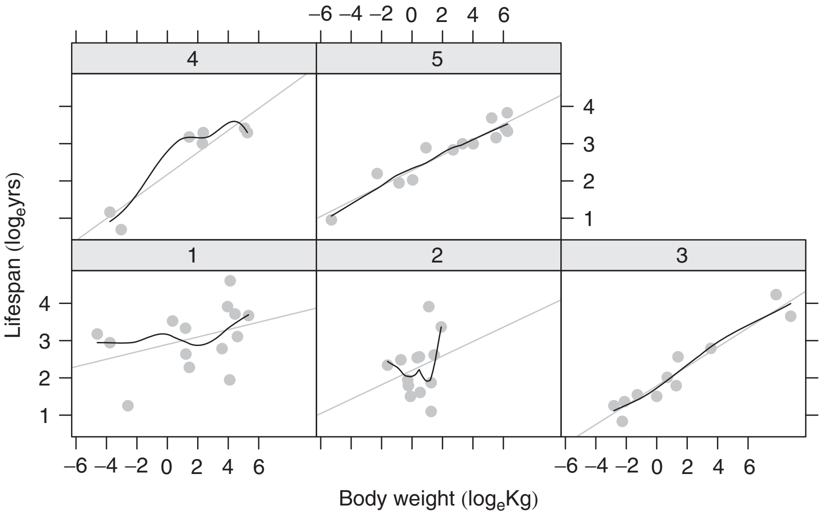
This produces the same as above.

5.11.1 `scales()` *parameters*

Many of the elements associated with the panel axes can be customized using the `scales` *parameter*. This parameter accepts a lists of arguments associated with the x and y axes.

```
> update(myPlot,  xlab=expression(paste("Body weight ",
    (log[e]*Kg))), ylab=expression(paste("Lifespan ",
    (log[e]*yrs))), scales=list(x=list(at=seq(-6,6,l=7))))
```

[k] As with most non-plotting functions in R, when a lattice plotting function is called without assigning a name for the output object, the result is automatically passed onto an appropriate print method before being discarded. If the function's output is assigned a name, the object is not "printed", it is stored.



5.12 Further reading

Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

Murrell, P. (2005). *R Graphics (Computer Science and Data Analysis)*. Chapman & Hall/CRC.

6

Simple hypothesis testing – one and two population tests

6.1 Hypothesis testing

Chapter 3 illustrated how samples can be used to estimate numerical characteristics or parameters of populations[a]. Importantly, recall that the standard error is an estimate of how variable *repeated* parameter estimates (e.g. population means) are likely to be from repeated (long-run) population re-sampling. Also recall, that the standard error can be estimated from a single collected sample given the degree of variability and size of this sample. Hence, sample means allow us make inferences about the population means, and the strength of these inferences is determined by estimates of how precise (or repeatable) the estimated population means are likely to be (standard error). The concept of precision introduces the value of using the characteristics of a single sample to estimate the likely characteristics of repeated samples from a population.This same philosophy of estimating the characteristics of a large number of possible samples and outcomes forms the basis of frequentist approach to statistics in which samples are used to objectively test specific hypotheses about populations.

A biological or research **hypothesis** is a concise statement about the predicted or theorized nature of a population or populations and usually proposes that there *is* an effect of a treatment (e.g. the means of two populations are different). Logically however, theories (and thus hypothesis) cannot be proved, only disproved (*falsification*) and thus a **null hypothesis** (H_0) is formulated to represent all possibilities except the hypothesized prediction. For example, if the hypothesis is that there *is* a difference between (or relationship among) populations, then the null hypothesis is that there is *no* difference or relationship (effect). Evidence against the null hypothesis thereby provides evidence that the hypothesis is likely to be true.

The next step in hypothesis testing is to decide on an appropriate statistic that describes the nature of population estimates in the context of the null hypothesis taking into account the precision of estimates. For example, if the null hypothesis is

[a] Recall that in a statistical context, the term population refers to all the possible observations of a particular condition from which samples are collected, and that this does not necessarily represent a biological population.

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

that the mean of one population is different to the mean of another population, the null hypothesis is that the population means are equal. The null hypothesis can therefore be represented mathematically as: $H_0 : \mu_1 = \mu_2$ or equivalently: $H_0 : \mu_1 - \mu_2 = 0$.

The appropriate test statistic for such a null hypothesis is a t-statistic:

$$t = \frac{(\bar{y}_1 - \bar{y}_2) - (\mu_1 - \mu_2)}{s_{\bar{y}_1 - \bar{y}_2}} = \frac{(\bar{y}_1 - \bar{y}_2)}{s_{\bar{y}_1 - \bar{y}_2}}$$

where $(\bar{y}_1 - \bar{y}_2)$ is the degree of difference between sample means of population 1 and 2 and $s_{\bar{y}_1 - \bar{y}_2}$ expresses the level of precision in the difference. If the null hypothesis is true and the two populations have identical means, we might expect that the means of samples collected from the two populations would be similar and thus the difference in means would be close to 0, as would the value of the t-statistic. Since populations and thus samples are variable, it is unlikely that two samples will have identical means, even if they are collected from identical populations (or the same population). Therefore, if the two populations were repeatedly sampled (with comparable collection technique and sample size) and t-statistics calculated, it would be expected that 50% of the time, the mean of sample 1 would be greater than that of population 2 and *visa versa*. Hence, 50% of the time, the value of the t-statistic would be greater than 0 and 50% of the time it would be less than 0. Furthermore, samples that are very different from one another (yielding large positive or negative t-values), although possible, would rarely be obtained.

All the possible values of the t-statistic (and thus sample combinations) calculated for a specific sample size for the situation when the null hypothesis is true could be collated and a histogram generated (see Figure 6.1a). From a frequentist perspective, this represents the sampling or probability distribution for the t-statistic calculated from repeated samples of a specific sample size (*degrees of freedom*) collected under the situation when the null hypothesis is true. That is, it represents all the possible expected t-values we might expect when there is no effect. When certain conditions (assumptions) are met, these t-values follow a known distribution called a t-distribution (see Figure 6.1b) for which the exact mathematical formula is known. The area under the entire t-distribution (curve) is one, and thus, areas under regions of the curve

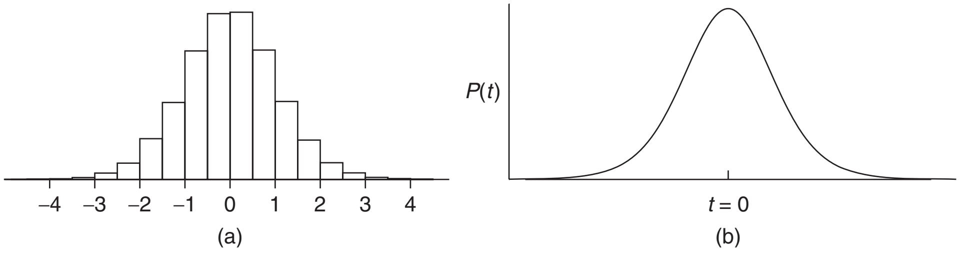


Fig 6.1 Distribution of all possible values of the t-statistic calculated from samples (each comprising of 10 observations) collected from two identical populations (situation when null hypothesis is true) represented as a (a) histogram and (b) t-distribution with 18 degrees of freedom ($df = (n_1 - 1) + (n_2 - 1) = 18$).

can be calculated, which in turn represent the relative frequencies (probabilities) of obtaining t-values in those regions. From the above example, the probability (p-value) of obtaining a t-value of greater than zero when the null hypothesis is true (population means equal) is 0.5 (50%).

When real samples are collected from two populations, the null hypothesis that the two population means are equal is tested by calculating the real value of the t-statistic, and using an appropriate t-distribution to calculate the probability of obtaining the observed (data) t-value or ones more extreme when the null hypothesis is true. If this probability is very low (below a set critical value, typically 0.05 or 5%), it is unlikely that the sample(s) could have come from such population(s) and thus the null hypothesis is unlikely to be true. This then provides evidence that the hypothesis is true.

Similarly, all other forms of hypothesis testing follow the same principal. The value of a test statistic that has been calculated from collected data is compared to the appropriate probability distribution for that statistic. If the probability of obtaining the observed value of the test statistic (or ones more extreme) when the null hypothesis is true is less than a predefined critical value, the null hypothesis is rejected, otherwise it is not rejected.

Note that the probability distributions of test statistics are strictly defined under a specific set of conditions. For example, the t-distribution is calculated for theoretical populations that are exactly normal (see chapter 3) and of identical variability. The further the actual populations (and thus samples) deviate from these ideal conditions, the less reliably the theoretical probability distributions will approximate the actual distribution of possible values of the test statistic, and thus, the less reliable the resulting hypothesis test.

6.2 One- and two-tailed tests

Two-tailed tests are any test used to test a null hypotheses that can be rejected by large deviations from expected in either direction. For example, when testing the null hypothesis that two population means are equal, the null hypothesis could be rejected if either population was greater than the other. By contrast one-tailed tests are those tests that are used to test more specific null hypotheses that restrict null hypothesis rejection to only outcomes in one direction. For example, we could use a one-tailed test to test the null hypothesis that the mean of population 1 was greater or equal to the mean of population 2. This null hypothesis would only be rejected if population 2 mean was significantly greater than that of population 1.

6.3 *t*-tests

Single population t-tests

Single population t-tests are used to test null hypotheses that a population parameter is equal to a specific value ($H_0 : \mu = \theta$, where θ is typically 0), and are thus useful for testing coefficients of regression and correlation or for testing whether measured differences are equal to zero.

Two population t-tests

Two population t-tests are used to test null hypotheses that two independent populations are equal with respect to some parameter (typically the mean, e.g. $H_0 : \mu_1 = \mu_2$). The t-test formula presented in section 6.1 above is used in the original student or pooled variances t-test. The separate variances t-test (Welch's test), represents an improvement of the t-test in that more appropriately accomodates samples with modestly unequal variances.

Paired samples t-tests

When observations are collected from a population in pairs such that two variables are measured from each sampling unit, a paired t-test can be used to test the null hypothesis that the population mean difference between paired observations is equal to zero ($H_0 : \mu_d = 0$). Note that this is equivalent to a single population t-test testing a null hypotheses that the population parameter is equal to the specific value of zero.

6.4 Assumptions

The theoretical t-distributions were formulated for samples collected from theoretical populations that are 1) **normally distributed** (see section 3.1.1) and 2) **equally varied**. Consequently, the theoretical t-distribution will only strictly represent the distribution of all possible values of the t-statistic when the populations from which real samples are collected also conform to these conditions. Hypothesis tests that impose distributional assumptions are known as *parametric tests*. Although substantial deviations from normality and/or homogeneity of variance reduce the reliability of the t-distribution and thus p-values and conclusions, t-tests are reasonably robust to violations of normality and to a lesser degree, homogeneity of variance (provided sample sizes equal).

As with most hypothesis tests, t-tests also assume 3) **that each of the observations are independent** (or that pairs are independent of one another in the case of paired t-tests). If observations are not independent, then a sample may not be an unbiased representation of the entire population, and therefore any resulting analyses could completely misrepresent any biological effects.

6.5 Statistical decision and power

Recall that probability distributions are typically symmetrical, bell-shaped distributions that define the relative frequencies (probabilities) of all possible outcomes and suggest that progressively more extreme outcomes become progressively less frequent or likely. By convention however, the statistical criteria for any given hypothesis test is a watershed value typically set at 0.05 or 5%. Belying the gradational decline in probabilities, outcomes with a probability less than 5% are considered unlikely whereas values equal to or greater are considered likely. However, values less than 5% are of course possible and could be obtained if the samples were by chance not centered similarly to the population(s) – that is, if the sample(s) were atypical of the population(s).

When rejecting a null hypothesis at the 5% level, we are therefore accepting that there is a 5% change that we are making an error (a **Type I error**). We are concluding that there is an effect or trend, yet it is possible that there really there is no trend, we just had unusual samples. Conversely, when a null hypothesis is not rejected (probability of 5% or greater) even though there really is a trend or effect in the population, a **Type II error** has been committed. Hence, a Type II error is when you fail to detect an effect that really occurs.

Since rejecting a null hypothesis is considered to be evidence of a hypothesis or theory and therefore scientific advancement, the scientific community projects itself against too many false rejections by keeping the statistical criteria and thus Type I error rate low (5%). However, as Type I and Type II error rates are linked, doing so leaves the Type II error rate (β) relatively large (approximately 20%).

The reciprocal of the Type II error rate, is called *power*. Power is the probability that a test will detect an effect (reject a null hypothesis, not make a Type II error) if one really occurs. Power is proportional to the statistical criteria, and thus lowering the statistical criteria compromises power. The conventional value of $\alpha = 0.05$) represents a compromise between Type I error rate and power.

Power is also affected by other aspects of a research framework and can be described by the following general representation:

$$power(1-\beta) \propto \frac{ES\sqrt{n}\,\alpha}{\sigma}$$

Statistical power is:

- directly proportional to the effect size (*ES*) which is the absolute size or magnitude of the effect or trend in the population. The more subtle the difference or effect, the lower the power
- directly proportional to the sample size (n). The greater the sample size, the greater the power
- directly proportional to the significance level ($\alpha = 0.05$) as previously indicated
- inversely proportional to the population standard deviation (σ). The more variable the population, the lower the power

When designing an experiment or survey, a researcher would usually like to know how many replicates are going to be required. Consequently, the above relationship is often transposed to express it in terms of sample size for a given amount of power:

$$n \propto \frac{(power\ \sigma)^2}{ES\ \alpha}$$

Researchers typically aim for power of at least 0.8 (80% probability of detecting an effect if one exists). Effect size and population standard deviation are derived from either pilot studies, previous research, documented regulations or gut feeling.

6.6 Robust tests

There are a number of more robust (yet less powerful) alternatives to independent samples t-tests and paired t-tests. The **Mann-Whitney-Wilcoxon** test[b] is a non-parametric (rank-based) equivalent of the independent samples t-test that uses the ranks of the observations to calculate test statistics rather than the actual observations and tests the null hypothesis that the two sampled populations have equal distributions. Similarly, the non-parametric **Wilcoxon signed-rank** test uses the sums of positive and negative signed ranked differences between paired observations to test the null hypothesis that the two sets of observations come from the one population. While neither test dictate that sampled populations must follow a specific distribution, the Wilcoxon signed-rank test does assume that the population differences are symmetrically distributed about the median and the Mann-Whitney test assumes that the sampled populations are equally varied (although violations of this assumption apparently have little impact). **Randomization tests** in which the factor levels are repeatedly shuffled so as to yield a probability distribution for the relevant statistic (such as the t-statistic) specific to the sample data do not have any distributional assumptions. Strictly however, randomization tests examine whether the sample patterns could have occurred by chance and do not pertain to populations.

6.7 Further reading

- Theory

  Fowler, J., L. Cohen, and P. Jarvis. (1998). *Practical statistics for field biology*. John Wiley & Sons, England.

  Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. John Wiley & Sons, New York.

  Manly, B. F. J. (1991). *Randomization and Monte Carlo methods in biology*. Chapman & Hall, London.

  Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

  Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

  Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practice - R

  Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

  Dalgaard, P. (2002). *Introductory Statistics with R*. Springer-Verlag, New York.

  Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

  Wilcox, R. R. (2005). *Introduction to Robust Estimation and Hypothesis Testing*. Elsevier Academic Press.

[b] The Mann-Whitney U-test and the Wilcoxon two-sample test are two computationally different tests that yield identical statistics.

6.8 Key for simple hypothesis testing

1 a. **Mean of single sample compared to a specific fixed value (such as a predicted population mean)** (one-sample *t*-test) . . . . Go to 3

b. **Two samples used to compare the means of two populations** . . . . Go to 2

2 a. **Two completely independent samples (different sampling units used for each replicate of each condition)** (independent samples *t*-test) . . . . Go to 3

FACTOR	DV
A	.
A	.
..	..
B	.
B	.
..	..

Dataset should be constructed in long format such that the variables are in columns and each replicate is in is own row.

b. **Two samples specifically paired (each of the sampling units measured under both conditions) to reduce within-group variation** (paired *t*-test) . . . . Go to 3

Pair	FACTOR	DV
1	A	.
2	A	.
..	..	..
1	B	.
2	B	.
..	..	

Dataset can be constructed in either long format (left) such that the variables are in columns and each replicate is in is own row or in wide format (right) such that each pair of measurements has its own row.

Pair	DV1	DV2
1	.	.
2	.	.
3	.	.
4	.	.
5	.	.
..	..	..

3 a. **Check parametric assumptions**

- **Normality of the response variable at both level of the categorical variable - boxplots**
  - *one-sample t-test*

    ```
    > boxplot(DV, dataset)
    ```
  - *two-sample t-test*

    ```
    > boxplot(DV ~ Factor, dataset)
    ```
  - *paired t-test*

    ```
    > with(dataset, boxplot(DV1 - DV2))
    > diffs <- with(dataset, DV[FACTOR == "A"]
    +     - DV[FACTOR == "B"])
    > boxplot(diffs)
    ```

where `DV` *and* `Factor` *are response and factor variables respectively in the* `dataset` *data frame.* `DV1` *and* `DV2` *represent the paired responses for group one and two of a paired t-test. Note, paired t-test data is traditionally setup in wide format (see section 2.7.6)*

- **Homogeneity of variance (two-sample *t*-tests only) - boxplots (as above) and scatterplot of mean vs variance**

```
> boxplot(DV ~ Factor, dataset)
```

where `DV` *and* `FACTOR` *are response and factor variables respectively in the* `dataset` *data frame*

Parametric assumptions met . . . . Go to 4
b. Parametric assumptions NOT met . . . . Go to 5
4 a. Perform one-sample *t*-test

```
> t.test(DV, dataset)
```

b. Perform (separate variances) independent-sample *t*-test . . . . See Example 6B

- *one-tailed* ($H_0 : \mu_A > \mu_B$)

```
> t.test(DV ~ FACTOR, dataset, alternative = "greater")
```

- *two-tailed* ($H_0 : \mu_A = \mu_B$)

```
> t.test(DV ~ FACTOR, dataset)
```

for pooled variances t-tests, include the `var.equal=T` *argument (see Example 6A).*
c. Perform (separate variances) paired *t*-test . . . . See Example 6C

- *one-tailed* ($H_0 : \mu_A > \mu_B$)

```
> t.test(DV1, DV2, dataset, alternative = "greater")
> t.test(DV ~ FACTOR, dataset, alternative = "greater",
+     paired = T)
```

- *two-tailed* ($H_0 : \mu_A = \mu_B$)

```
> t.test(DV1, DV2, dataset)
> t.test(DV ~ FACTOR, dataset, paired = T)
```

for pooled variances t-tests, include the `var.equal=T` *argument.*
5 a. Attempt a scale transformation (see Table 3.2 for common transformations) . . . . Go to 3
b. Transformations unsuccessful or inappropriate . . . . Go to 6
6 a. Underlying distribution of the response variable and residuals is non-normal, yet known . . . . GLM chapter 17
b. Underlying distribution of the response variable and residuals is non-normal and is NOT known . . . . Go to 7
7 a. Observations independent or specifically paired, variances not wildly unequal (Wilcoxon rank sum nonparametric test) . . . . Go to 8
b. Variances not wildly unequal, random sampling not possible (Randomization test) . . . . See Example 6E

```
> library(boot)
> data.boot <- boot(dataset, stat, R = 999, sim = "parametric",
+     rand.gen = rand.gen)
> plot(data.boot)
> print(data.boot)
```

where `stat` *is the statistic to repeatedly calculate and* `rand.gen` *defines how the data are randomized.*

8 a. **Perform one-sample Wilcoxon (rank sum) test**

```
> wilcox.test(DV, dataset)
```

b. **Perform independent-sample Mann-Whitney Wilcoxon test** ..... See Example 6D

- *one-tailed* ($H_0 : \mu_A > \mu_B$)

```
> wilcox.test(DV ~ FACTOR, dataset, alternative = "greater")
```

- *two-tailed* ($H_0 : \mu_A = \mu_B$)

```
> wilcox.test(DV ~ FACTOR, dataset)
```

c. **Perform paired Wilcoxon (signed rank) test**

- *one-tailed* ($H_0 : \mu_A > \mu_B$)

```
> wilcox.test(DV1,DV2, dataset, alternative="greater")
> #OR for long format
> wilcox.test(DV~FACTOR, dataset, alternative="greater",
+ paired=T)
```

- *two-tailed* ($H_0 : \mu_A = \mu_B$)

```
> wilcox.test(DV1, DV2, dataset)
> wilcox.test(DV ~ FACTOR, dataset, paired = T)
```

6.9 Worked examples of real biological data sets

Example 6A: Pooled variances, student t-test

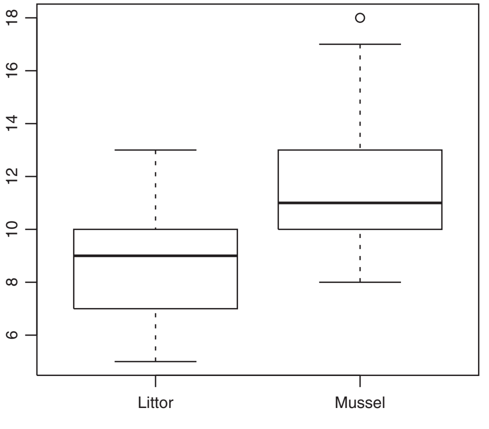
Ward and Quinn (1988) investigated differences in the fecundity (as measured by egg production) of a predatory intertidal gastropod (*Lepsiella vinosa*) in two different intertidal zones (mussel zone and the higher littorinid zone) (Box 3.2 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Ward and Quinn (1988) data set.

```
> ward <- read.table("ward.csv", header = T, sep = ",")
```

Step 2 (Key 6.3) - Assess assumptions of normality and homogeneity of variance for the null hypothesis that the population mean egg production is the same for both littorinid and mussel zone *Lepsiella*.

```
> boxplot(EGGS ~ ZONE, ward)
```



```
> with(ward, rbind(MEAN = tapply(EGGS, ZONE, mean),
+     VAR = tapply(EGGS, ZONE, var)))
       Littor    Mussel
MEAN 8.702703 11.357143
VAR  4.103604  5.357143
```

Conclusions - There was no evidence of non-normality (boxplots not grossly asymmetrical) or unequal variance (boxplots very similar size and variances very similar). Hence, the simple, studentized (pooled variances) *t*-test is likely to be reliable.

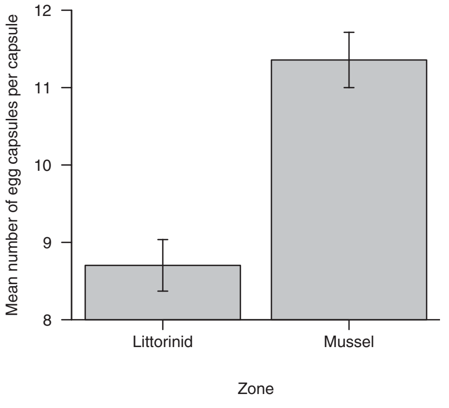
Step 3 (Key 6.4b) - Perform a pooled variances *t*-test to test the null hypothesis that the population mean egg production is the same for both littorinid and mussel zone *Lepsiella*.

```
> t.test(EGGS ~ ZONE, ward, var.equal = T)
        Two Sample t-test

data:  EGGS by ZONE
t = -5.3899, df = 77, p-value = 7.457e-07
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -3.635110 -1.673770
sample estimates:
mean in group Littor mean in group Mussel
            8.702703            11.357143
```

Conclusions - Reject the null hypothesis. Egg production by predatory gastropods (*Lepsiella vinosa* was significantly greater ($t_{77} = -5.39, P < 0.001$) in mussel zones than littorinid zones on rocky intertidal shores.

Summarize the trends with a bargraph.

```
> ward.means <- with(ward, tapply(EGGS, ZONE, mean))
> ward.sds <- with(ward, tapply(EGGS, ZONE, sd))
> ward.ns <- with(ward, tapply(EGGS, ZONE, length))
> ward.se <- ward.sds/sqrt(ward.ns)
> xs <- barplot(ward.means, ylim = range(pretty(c(ward.means +
+     ward.se, ward.means - ward.se))), axes = F, xpd = F,
+     axisnames = F, axis.lty = 2, legend.text = F, col = "gray")
> arrows(xs, ward.means + ward.se, xs, ward.means - ward.se,
+     code = 3, angle = 90, len = 0.05)
> axis(2, las = 1)
> axis(1, at = xs, lab = c("Littorinid", "Mussel"), padj = 1,
+     mgp = c(0, 0, 0))
> mtext(2, text = "Mean number of egg capsules per capsule",
+     line = 3, cex = 1)
> mtext(1, text = "Zone", line = 3, cex = 1)
> box(bty = "l")
```



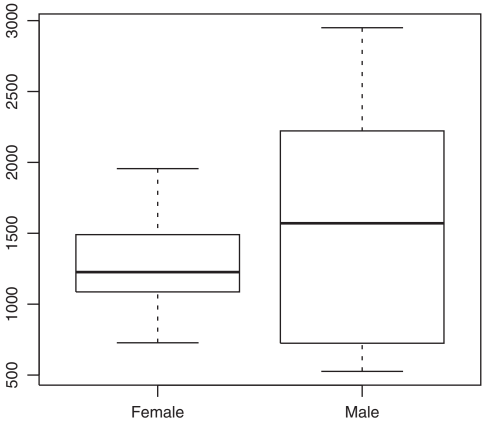
Example 6B: Separate variances, Welch's t-test
Furness and Bryant (1996) measured the metabolic rates of eight male and six female breeding northern fulmars and were interesting in testing the null hypothesis that there was no difference in metabolic rate between the sexes (Box 3.2 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Furness and Bryant (1996) data set.

```
> furness <- read.table("furness.csv", header = T, sep = ",")
```

Step 2 (Key 6.3) - Assess assumptions of normality and homogeneity of variance for the null hypothesis that the population mean metabolic rate is the same for male and female breeding northern fulmars.

```
> boxplot(METRATE ~ SEX, furness)
```




```
> with(furness, rbind(MEAN = tapply(METRATE, SEX, mean),
+      VAR = tapply(METRATE, SEX, var)))
          Female         Male
MEAN    1285.517     1563.775
VAR   177209.418   799902.525
```

Conclusions - Whilst there is no evidence of non-normality (boxplots not grossly asymmetrical), variances are a little unequal (although perhaps not grossly unequal - one of the boxplots is not more than three times smaller than the other). Hence, a separate variances *t*-test is more appropriate than a pooled variances *t*-test.

Step 3 (Key 6.4b) - Perform a separate variances (Welch's) *t*-test to test the null hypothesis that the population mean metabolic rate is the same for both male and female breeding northern fulmars.

```
> t.test(METRATE ~ SEX, furness, var.equal = F)
        Welch Two Sample t-test

data:  METRATE by SEX
t = -0.7732, df = 10.468, p-value = 0.4565
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -1075.3208   518.8042
sample estimates:
mean in group Female       mean in group Male
                  1285.517                   1563.775
```

Conclusions - Do not reject the null hypothesis. Metabolic rate of male breeding northern fulmars was not found to differ significantly ($t = -0.773, df = 10.468, P = 0.457$) from that of females.

Example 6C: Paired t-test

To investigate the effects of lighting conditions on the orb-spinning spider webs Elgar et al. (1996) measured the horizontal (width) and vertical (height) dimensions of the webs made by 17 spiders under light and dim conditions. Accepting that the webs of individual spiders vary considerably, Elgar et al. (1996) employed a paired design in which each individual spider effectively acts as its own control. A paired *t*-test performs a one sample *t*-test on the differences between dimensions under light and dim conditions (Box 3.3 of Quinn and Keough (2002)).

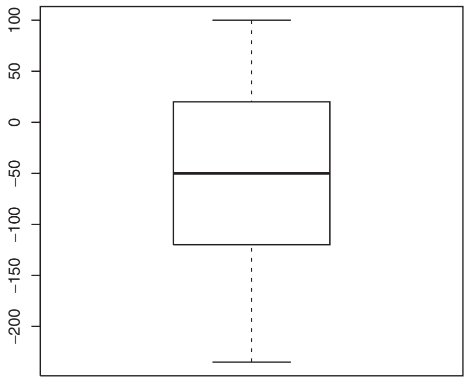
Step 1 - Import (section 2.3) the Elgar et al. (1996) data set.

```
> elgar <- read.table("elgar.csv", header = T, sep = ",")
```

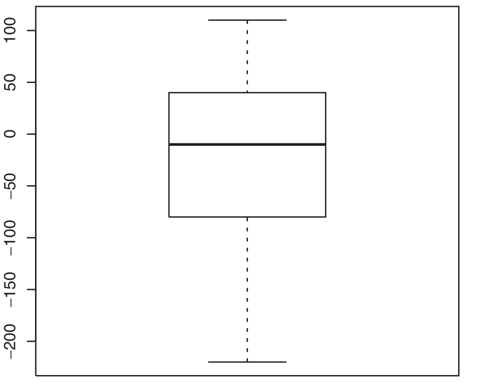
Note the format of this data set. Rather than organizing the data into the usual long format in which variables are represented in columns and rows represent individual replicates, these data have been organized in wide format. Wide format is often used for data containing repeated measures from individual or other sampling units. Whilst, this is not necessary (as paired *t*-tests can be performed on long format data), traditionally it did allow more compact data management as well as making it easier to calculate the differences between repeated measurements on each individual.

Step 2 (Key 6.3) - Assess whether the differences in web width (and height) in light and dim light conditions are normally distributed.

```
> with(elgar, boxplot(HORIZLIG -
+ HORIZDIM))
```




```
> with(elgar, boxplot(VERTLIGH -
+ VERTDIM))
```



Conclusions - There is no evidence of non-normality for either the difference in widths or heights of webs under light and dim ambient conditions. Therefore paired *t*-tests are likely to be reliable tests of the hypotheses that the mean web dimensional differences are equal to zero.

Step 3 (Key 6.4c) - Perform two separate paired *t*-tests to test the test the respective null hypotheses.

- No effect of lighting on web width

```
> with(elgar, t.test(HORIZLIG, HORIZDIM, paired = T))
        Paired t-test

data:  HORIZLIG and HORIZDIM
t = -2.1482, df = 16, p-value = 0.04735
alternative hypothesis: true difference in means is not
 equal to 0
95 percent confidence interval:
 -91.7443687  -0.6085725
sample estimates:
mean of the differences
              -46.17647
```

- No effect of lighting on web height

```
> with(elgar, t.test(VERTLIGH, VERTDIM, paired = T))
        Paired t-test

data:  VERTLIGH and VERTDIM
t = -0.9654, df = 16, p-value = 0.3487
alternative hypothesis: true difference in means is not
 equal to 0
```

```
95 percent confidence interval:
 -65.79532  24.61885
sample estimates:
mean of the differences
              -20.58824
```

Conclusions - Orb-spinning spider webs were found to be significantly wider ($t = 2.148$, $df = 16, P = 0.047$) under dim lighting conditions than light conditions, yet were not found to differ ($t = 0.965, df = 16, P = 0.349$) in height.

Example 6D: Non-parametric Mann-Whitney-Wilcoxon signed rank test

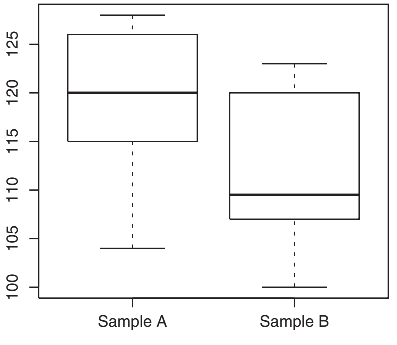
Sokal and Rohlf (1997) presented a dataset comprising the lengths of cheliceral bases (in μm) from two samples of chigger (*Trombicula lipovskyi*) nymphs. These data were used to illustrate two equivalent tests (Mann-Whitney U-test and Wilcoxon two-sample test) of location equality (Box 13.7 of Sokal and Rohlf (1997)).

Step 1 - Import (section 2.3) the nymph data set.

```
> nymphs <- read.table("nymphs.csv", header = T, sep = ",")
```

Step 2 (Key 6.3) - Assess assumptions of normality and homogeneity of variance for the null hypothesis that the population mean metabolic rate is the same for male and female breeding northern fulmars.

```
> boxplot(LENGTH ~ SAMPLE, nymphs)
```




```
> with(nymphs, rbind(MEAN = tapply(LENGTH, SAMPLE, mean),
+      VAR = tapply(LENGTH, SAMPLE, var)))
      Sample A  Sample B
MEAN 119.68750 111.80000
VAR   53.29583  60.17778
```

Conclusions - Whilst there is no evidence of unequal variance, there is some (possible) evidence of non-normality (boxplots slightly asymmetrical). These data will therefore be analysed using a non-parametric Mann-Whitney-Wilcoxon signed rank test.

Step 3 (Key 6.8b) - Perform a Mann-Whitney Wilcoxon test to investigate the null hypothesis that the mean length of cheliceral bases is the same for the two samples of nymphs of chigger (*Trombicular lipovskyi*).

```
> wilcox.test(LENGTH ~ SAMPLE, nymphs)
        Wilcoxon rank sum test with continuity correction

data:  LENGTH by SAMPLE
W = 123.5, p-value = 0.02320
alternative hypothesis: true location shift is not equal to 0
```

Conclusions - Reject the null hypothesis. The length of the cheliceral base is significantly longer in nymphs from sample 1 ($W = 123.5, df = 24, P = 0.023$) than those from sample 2.

Example 6E: Randomization t-test

Powell and Russell (1984, 1985) investigated differences in beetle consumption between two size classes of eastern horned lizard (*Phrynosoma douglassi brevirostre*) represented respectively by adult females in the larger class and adult male and yearling females in the smaller class (Example 4.1 from Manly, 1991).

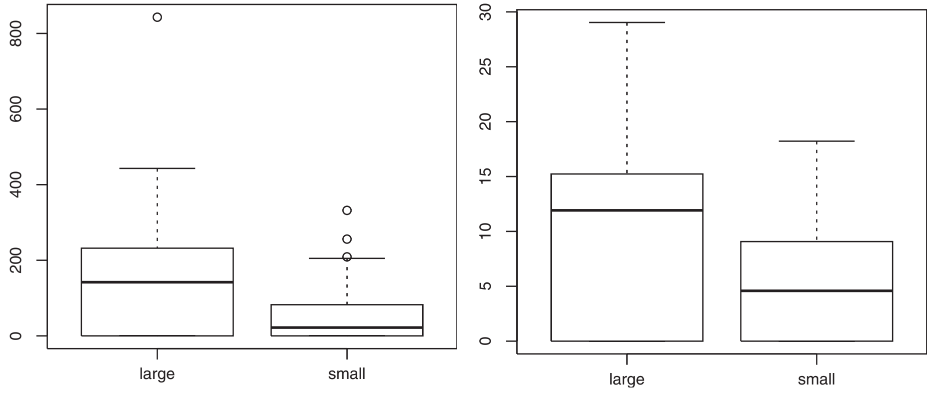
Step 1 - Import (section 2.3) the Powell and Russell (1984, 1985) beetle data set.

```
> beetles <- read.table("beetle.csv", header = T, sep = ",")
```

Step 2 (Key 6.3) - Assess normality/homogeneity of variance using boxplot of ant biomass against month. Cube root transformation also assessed, but not shown.

```
> boxplot(BEETLES~SIZE,
+ beetles)
```

```
> boxplot(sqrt(BEETLES)~SIZE,
+ beetles)
```



Conclusions - strong evidence of non-normality and lots of zero values. As a result a randomization test in which the *t*-distribution is generated from the samples, might be more robust than a standard *t*-test that assumes each of the populations are normally distributed.

Furthermore, the observations need not be independent, provided we are willing to concede that we are no longer testing hypotheses about populations (rather, we are estimating the probability of obtaining the observed differences in beetle consumption between the size classes just by chance).

Step 3 (Key 6.7b) - define the statistic to use in the randomization test – in this case the *t*-statistic (without replacement).

```
> stat <- function(data, indices) {
+     t.test <- t.test(BEETLES ~ SIZE, data)$stat
+     t.test
+ }
```

Step 4 (Key 6.7b) - define how the data should be randomized – randomly reorder the which size class that each observation belonged to.

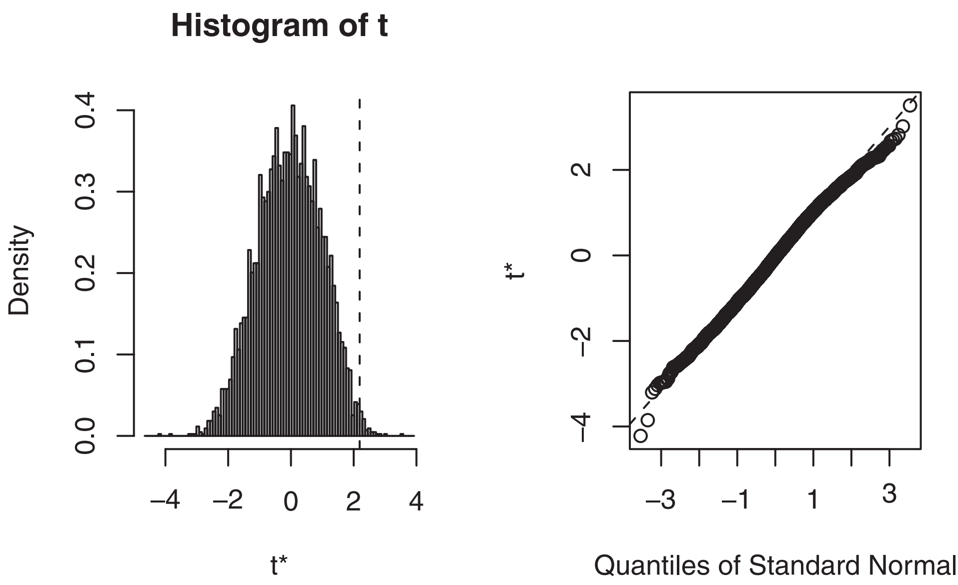
```
> rand.gen <- function(data, mle) {
+     out <- data
+     out$SIZE <- sample(out$SIZE, replace = F)
+     out
+ }
```

Step 5 (Key 6.7b) - call a bootstrapping procedure to randomize 5000 times (this can take some time).

```
> library(boot)

> beetles.boot <- boot(beetles, stat, R = 5000, sim = "parametric",
+     ran.gen = rand.gen)
```

Step 6 (Key 6.7b) - examine the distribution of *t*-statistics generated from the randomization procedure

```
> print(beetles.boot)
PARAMETRIC BOOTSTRAP


Call:
boot(data = beetles, statistic = stat, R = 5000, sim = "parametric",
    ran.gen = rand.gen)


Bootstrap Statistics :
    original    bias    std. error
t1* 2.190697 -2.237551    1.019904

> plot(beetles.boot)
```



Conclusions - The observed t-value was 2.191. Note that the t-distribution is centered around zero and thus a t-value of 2.191 is equivalent to a t-value of -2.191. Only the magnitude of a t-value is important, not the sign.

Step 7 (Key 6.7b) - calculate the number of possible t-values (including the observed t-value, which is one possible situation) that were greater or equal to the observed t-value and express this as a percentage of the number of randomizations (plus one for the observed situation) performed.

```
> tval <- length(beetles.boot[beetles.boot$t >= abs(beetles.
+     boot$t0)]) + 1
> tval/(beetles.boot$R + 1)
[1] 0.00759848
```

Conclusions - Reject the null hypothesis that the difference in beetle consumption between small and large lizards is purely due to chance. It is likely that beetle consumption is significantly higher in large female eastern horned lizards than the smaller adult males and yearling females ($t = 2.191, R = 5000, P = 0.019$).

7

Introduction to Linear models

A **statistical model** is an expression that attempts to explain patterns in the observed values of a response variable by relating the response variable to a set of predictor variables and parameters. Consider the following familiar statistical model:

$$y = mx + c$$

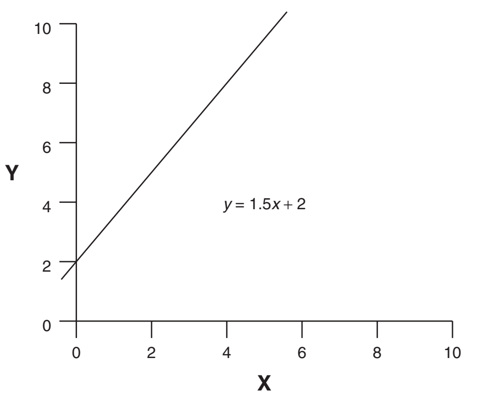
or equivalently:

$$y = bx + a$$

This simple statistical model relates a response variable (y) to a single predictor variable (x) as a straight line according to the values of two constant parameters:

b – the degree to which y changes per unit of change in x (gradient of line)

a – the value of y when $x = 0$ (y-intercept)



The above statistical model represents a perfect fit, that is, 100% of the change (variation) in y is explained by a change in x. However, rarely would this be the case when modeling biological variables. In complex biological systems, variables are typically the result of many influential and interacting factors and therefore simple models usually fail to fully explain a response variable. Consequently, the statistical model also has an *error* component that represents the portion of the response variable that the model fails to explain. Hence, statistical models are of the form:

$$\text{response variable} = \text{model} + \text{error}$$

where the model component comprises of one or more categorical and/or continuous predictor variable(s) and their parameter(s) that together represent the effect of the

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

predictors variable(s) on the mean the response variable. A parameter and its associated predictor variable(s) are referred to as a model *term*.

A statistical model is fitted to observed data so as to estimate the model parameters and test hypotheses about these parameters (coefficients).

7.1 Linear models

Linear models are those statistical models in which a series of parameters are arranged as a linear combination. That is, within the model, no parameter appears as either a multiplier, divisor or exponent to any other parameter. Importantly, the term 'linear' in this context does not pertain to the nature of the relationship between the response variable and the predictor variable(s), and thus linear models are not restricted to 'linear' (straight-line) relationships.

An example of a very simple linear model, is the model used to investigate the linear relationship between a continuous response variable (Y and a single continuous predictor variable, X):

$$\begin{array}{ccccccccc}
y_i & = & \beta_0 & + & \beta_1 & \times & x_i & + & \varepsilon_i \\
\text{response variable} & = & \text{population} & + & \text{population} & \times & \text{predictor} & + & \text{error} \\
 & = & \text{intercept} & & \text{slope} & & \text{variable} & &
\end{array}$$

$$\underbrace{\underbrace{\text{population intercept}}_{\text{intercept term}} + \underbrace{\text{population slope} \times \text{predictor variable}}_{\text{slope term}}}_{\text{model}}$$

The above notation is typical of that used to represent the elements of a linear model. y denotes the response variable and x represents the predictor variable. The subscript (i) is used to represent a set of observations (usually from 1 to n where n is the total sample size) and thus y_i and x_i represent respectively the i^{th} observation of the Y and X variables. ε_i represents the deviation of the i^{th} observed Y from the value of Y expected by the model component. The parameters β_0 and β_1 represent population intercept and population slope (effect of X on Y per unit of x) respectively. Population (effect) parameters are usually represented by Greek symbols[a]. The above linear model notation is therefore a condensed representation of a compilation of arithmetic relationships:

$$\begin{array}{ccccccccc}
y_1 & = & \beta_0 & + & \beta_1 & \times & x_1 & + & \varepsilon_1 \\
y_2 & = & \beta_0 & + & \beta_1 & \times & x_2 & + & \varepsilon_2 \\
y_3 & = & \beta_0 & + & \beta_1 & \times & x_3 & + & \varepsilon_3 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots
\end{array}$$

[a] Typically, effect parameters associated with continuous variables are represented by β and those associated with categorical variables are represented by the symbols $\alpha, \beta, \gamma, \dots$

the first y observation (y_1) is related to the first x observation (x_1) according to the values of the two constants (parameters β_0 and β_1) and ε_1 is the amount that the observed value of Y differs from the value expected according the model (the *residual*).

When there are multiple continuous predictor variables, in addition to the intercept parameter (β_0), the linear model includes a separate slope parameter for each of the predictor variables:

$$y_i = \beta_0 + \beta_1 x1_i + \beta_2 x2_i + ... + \varepsilon_i$$

The model structure for linear models containing a single categorical predictor variable (known as a factor) with two or more treatment levels (groups) is similar in form to the multiple linear regression model (listed immediately above) with the overall mean (μ) replacing the y-intercept (β_0). The factor levels (groups) are represented in the model by binary (contain only of 0s and 1s, see Table 7.1) *indicator* (or 'dummy') variables and associated estimable parameters ($\beta_1,\ \beta_2,\ ...$).

For a data set comprising of p groups and n replicates within each group, the linear model is:

$$y_{ij} = \mu + \beta_1(dummy_1)_{ij} + \beta_2(dummy_2)_{ij} + ... + \varepsilon_{ij}$$

where i represents the treatment levels (from 1 to p) and j represents the set of replicates (from 1 to n) within the i^{th} group. Hence, y_{ij} represents the j^{th} observation of the response variable within the i^{th} group and $(dummy_1)_{ij}$ represents the dummy code for the j^{th} replicate within the i^{th} group of the first dummy variable (first treatment level).

The dummy variable for a particular treatment level contains all 0s except in the rows that correspond to observations that received that treatment level. Table 7.1 illustrates

Table 7.1 Fictitious data set (consisting of three replicates for each of three groups:'G1','G1','G2') to illustrate the link between a) single factor dataset, and b) the indicator (dummy) variables.

a)

y	A
2	G1
3	G1
4	G1
6	G2
7	G2
8	G2
10	G3
11	G3
12	G3

b)

y	$dummy_1$	$dummy_2$	$dummy_3$
2	1	0	0
3	1	0	0
4	1	0	0
6	0	1	0
7	0	1	0
8	0	1	0
10	0	0	1
11	0	0	1
12	0	0	1

the dummy coding for a single factor within three levels (‘G1’, ‘G2’, ‘G3’) each with three replicates[b].

More typically however, statistical models that include one or more factors are expressed as *effects models* in which the individual treatment levels (and their parameters) are represented by a single term (e.g. α_i) that denotes the effect of each of the levels of the factor on the overall mean. For a data set comprised of p groups and n replicates within each group, the linear effects model is:

$$y_{ij} = \mu + \alpha_i + \varepsilon_{ij}$$

where i represents the set of treatments (from 1 to p) and j represents the set of replicates (from 1 to n) within the i^{th} group. Hence, y_{ij} represents the j^{th} observation of the response variable within the i^{th} group of the factor. μ is the overall population mean of the response variable (Y) and is equivalent to the intercept. α_i represents the effect of the i^{th} group calculated as the difference between each of the group means and the overall mean ($\alpha_i = \mu_i - \mu$).

7.2 Linear models in R

Statistical models in R are represented by a formula corresponding to the linear model (for continuous variables) or effects model (categorical variables):

```
> response~model
```

where the tilde (~) defines a model formula and `model` represents a set of terms to include in the model. Terms are included in a model via their variable names and terms preceded by the - (negative sign) operator are explicitly excluded. The intercept term (denoted by a 1) is implicit in the model and need not be specified. Hence the following model formulae all model the effect of the variable `X` on the `Y` variable with the inclusion of the intercept:

```
> Y~X
> Y~1+X
> Y~X+1
```

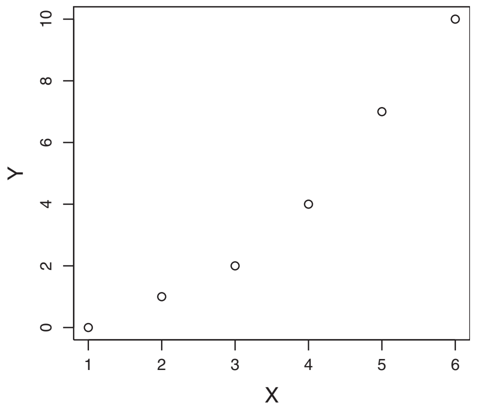
whereas the following exclude the intercept:

```
> Y~-1+X
> Y~X-1
```

Linear models are fitted by providing the model formula as an argument to the `lm()` *function*. To fit the simple linear regression model relating a fictitious response variable (`Y`) to fictitious continuous predictor variable (`X`):

[b] Note that linear model that this represents ($y_{ij} = \mu + \beta_1(dummy_1)_{ij} + \beta_2(dummy_2)_{ij} + \beta_3(dummy_3)_{ij} + \varepsilon_{ij}$) is over-parameterized, see section 7.3.

```
> Y<-c(0,1,2,4,7,10)
> X<-1:6
> plot(Y~X)
```




```
> Fictitious.lm <- lm(Y~X)
```

To examine the estimated parameters (and hypothesis tests) from the fitted model, provide the name of the fitted model as an argument to the `summary()` *function*[c].

```
> summary(Fictitious.lm)
Call:
lm(formula = Y ~ X)

Residuals:
         1          2          3          4          5          6
 1.000e+00  3.404e-16 -1.000e+00 -1.000e+00  6.280e-17  1.000e+00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -3.0000     0.9309  -3.223  0.03220 *
X             2.0000     0.2390   8.367  0.00112 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 4 degrees of freedom
Multiple R-squared: 0.9459,     Adjusted R-squared: 0.9324
F-statistic:    70 on 1 and 4 DF,  p-value: 0.001116
```

The summary output begins by specifying the nature of the call used to fit the model. Next is a summary of the residuals (differences between observed responses and

[c] Actually, the `summary()` function is an overloaded wrapper that invokes different specific functions depending on the class of object provided as the first argument. In the `summary()` *function* invokes the `summary.lm()` *function*.

expected responses for each value of the predictor variable). The estimated parameters are listed in the coefficients table. Each row of the table lists the value of an estimated parameter from the linear model along with the outcome of a hypothesis test for this parameter. The row labeled '`(Intercept)`' concerns the intercept (overall constant) and subsequent rows are labeled according to the model term that is associated with the estimated parameter. In this case, the row labeled '`X`' concerns the population slope (β_1). Finally a brief summary of the partitioning of total variation (ANOVA, see section 7.3.2) in the response variable is provided.

7.3 Estimating linear model parameters

During model fitting, parameters can be estimated using any of the estimation methods outlined in section 3.7, although ordinary least squares (OLS) and maximum likelihood (ML or REML) are most common. The OLS approach estimates the value of one or more parameters such that they minimize the sum of squared deviations between the observed values and the parameter (typically the values predicted by the model) and will be illustrated in detail in the following sections. Models that utilize OLS parameter estimates are referred to as 'general' linear models as they accommodate both continuous and categorical predictor variables. Broadly speaking, such models that incorporate purely continuous predictor variables are referred to as 'regression' models (see chapters 8 & 9) whereas models that purely incorporate categorical predictors are called 'ANOVA' models (see chapters 10 – 14). Analysis of covariance (ANCOVA) models incorporate both categorical and continuous predictor variables (see chapter 15).

ML estimators estimate one or more population parameters such that the (log) likelihood of obtaining the observed values from such populations is maximized and these models are useful when there is evidence of a relationship between mean and variance or for models involving correlated data structures. Maximum likelihood parameter estimation is also utilized by 'generalized' linear models, so called as they are not restricted to normally distributed response and residuals. Generalized linear models accommodate any exponential probability distribution (including normal, binomial, Poisson, gamma and negative binomial), see chapter 17.

The parameters estimated during simple linear and multiple linear regression analyses are relatively straightforward to interpret (they simply represent the rates of change in the response variable attributable to each individual predictor variable) and can be used to construct an algebraic representation of the relationship between a response variable and one or more predictor variables. However, this is generally not the case for linear models containing factorial variables.

7.3.1 Linear models with factorial variables

Recall from section 7.1 that linear models comprising of a single factor are expressed as an effects model:

$$y_{ij} = \mu + \alpha_i + \varepsilon_{ij}$$

where α_i estimates the effect of each treatment group on the overall mean of groups ($\alpha_i = \mu_i - \mu$). However, the effects model for a factor with p groups, will have $p+1$ parameters (the overall mean μ plus the p α parameters), and thus the linear effects model is considered to be 'over-parameterized'[d]. In order to obtain parameter estimates, the model must be reduced to a total of p parameters. Over-parameterization can be resolved either by removing one of the parameters from the effects model (either the overall mean (μ) or one of the treatment effects (α_i) parameters - a procedure rarely used in biology), or by generating a new set ($p-1$) of effects parameters (α^*_q, where q represents the set of orthogonal parameters from 1 to $p-1$) each of which represent a linear combination of groups rather than a single group effect. That is, each α^* can include varying contributions from any number of the groups and are not restricted to a single contrast of ($= \mu_i - \mu$). For example, one of the parameters might represent the difference in means between two groups or the difference in means between one group and the average of two other groups. The reduced number of effects parameters are defined through the use of a matrix of 'contrast coefficients'. Note, the new set of effects parameters should incorporate the overall relational effects of each of the groups equally such that each group maintains an equal contribution to the overall model fit.

A number of 'pre-fabricated', contrast matrices exist, each of which estimate a different set of specific comparisons between treatment combinations. The most common contrasts types include:

Treatment contrasts - in which each of the treatment groups means are compared to the mean of a 'control' group. This approach to over-parameterization is computationally identical to fitting $p-1$ dummy variables via multiple linear regression. However, due to the interpretation of the parameters (groups compared to a control) and the fact that treatment effects are not orthogonal to the intercept, the interpretation of treatment contrasts (and thus dummy regression) is really only meaningful for situations where there is clearly a single group (control) to which the other groups can be compared. For treatment contrasts, the intercept is replaced by α^*_1 and thus the remaining α^*_q parameters are numbered starting at 2.

Parameter	Estimates	Null hypothesis
Intercept	mean of 'control' group (μ_1)	H_0: $\mu = \mu_1 = 0$
α^*_2	mean of group 2 minus mean of 'control' group ($\mu_2 - \mu_1$)	H_0: $\alpha^*_2 = \mu_2 - \mu_1 = 0$
α^*_3	mean of group 3 minus mean of 'control' group ($\mu_3 - \mu_1$)	H_0: $\alpha^*_3 = \mu_3 - \mu_1 = 0$
...		

[d] Given that $\alpha_i = \mu_i - \mu$, it is only possible to estimate $p-1$ orthogonal (independent) parameters. For example, once μ and $p-1$ of the effects parameters have been estimated, the final effects parameter is no longer 'free to vary' and therefore cannot be independently estimated. Likewise, if the full linear model contains as many dummy variables as there are treatment groups, then it too is over-parameterized.

```
> Y <- c(2,3,4,6,7,8,10,11,12)
> A <- gl(3,3,9,lab=c("G1","G2","G3"))
> # specify that treatment contrasts should be used
> contrasts(A) <-contr.treatment
> summary(lm(Y~A))
Call:
lm(formula = Y ~ A)

Residuals:
       Min         1Q     Median         3Q        Max
-1.000e+00 -1.000e+00  6.939e-17  1.000e+00  1.000e+00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0000     0.5774   5.196  0.00202 **
A2            4.0000     0.8165   4.899  0.00271 **
A3            8.0000     0.8165   9.798  6.5e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 6 degrees of freedom
Multiple R-squared: 0.9412,     Adjusted R-squared: 0.9216
F-statistic:    48 on 2 and 6 DF,  p-value: 0.0002035
```

Sum to zero contrasts - this technique constrains the sum of the unconstrained treatment effects (α) to zero. In this model, the intercept estimates the average treatment effect and the remaining (α^*) estimate the differences between each of the treatment means and the average treatment mean.

Parameter	Estimates	Null hypothesis
Intercept	mean of group means (μ_{i^*}/p)	H_0: $\mu = \mu_q/p = 0$
α_1^*	mean of group 1 minus mean of group means ($\mu_1 - (\mu_q/p)$)	H_0: $\alpha_1 = \mu_1 - (\mu_q/p) = 0$
α_2^*	mean of group 2 minus mean of group means ($\mu_2 - (\mu_q/p)$)	H_0: $\alpha_2 = \mu_2 - (\mu_q/p) = 0$
...		

```
> # specify that sum-to-zero contrast should be used
> contrasts(A) <-contr.sum
> summary(lm(Y~A))
Call:
lm(formula = Y ~ A)

Residuals:
       Min         1Q     Median         3Q        Max
-1.000e+00 -1.000e+00  1.388e-17  1.000e+00  1.000e+00
```

```
Coefficients:
              Estimate Std. Error  t value Pr(>|t|)
(Intercept)  7.000e+00  3.333e-01   21.000  7.6e-07 ***
A1          -4.000e+00  4.714e-01   -8.485 0.000147 ***
A2           1.228e-16  4.714e-01 2.60e-16 1.000000
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 6 degrees of freedom
Multiple R-squared: 0.9412,     Adjusted R-squared: 0.9216
F-statistic:    48 on 2 and 6 DF,  p-value: 0.0002035
```

Helmert contrasts - the intercept estimates the average treatment effect and the remaining (α_q^*) estimate the differences between each of the treatment means and the mean of the group before it. In reality, parameter estimates from Helmert contrasts have little biological interpretability.

Parameter	Estimates	Null hypothesis
Intercept	mean of group means (μ_q/p)	H_0: $\mu = \mu_q/p = 0$
α_1^*	mean of group 2 minus mean of (group means and mean of group1) ($\mu_2 - (\mu_q/p + \mu_1)/2$)	H_0: $\alpha_1^* = \mu_2 - (\mu_q/p + \mu_1)/2 = 0$
α_2^*	mean of group 3 minus mean of (group means, mean of group1 and mean of group2) ($\mu_3 - (\mu_q/p + \mu_1 + \mu_2)/3$)	H_0: $\alpha_2^* = \mu_3 - (\mu_q/p + \mu_1 + \mu_2)/3 = 0$
...		

```
> # specify that Helmert contrasts should be used
> contrasts(A) <-contr.helmert
> summary(lm(Y~A))
Call:
lm(formula = Y ~ A)

Residuals:
       Min          1Q      Median          3Q         Max
-1.000e+00 -1.000e+00 -7.865e-17  1.000e+00  1.000e+00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.0000     0.3333  21.000  7.6e-07 ***
A1            2.0000     0.4082   4.899 0.002714 **
A2            2.0000     0.2357   8.485 0.000147 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1 on 6 degrees of freedom
Multiple R-squared: 0.9412,     Adjusted R-squared: 0.9216
F-statistic:    48 on 2 and 6 DF,  p-value: 0.0002035
```

Polynomial contrasts - generate orthogonal polynomial trends (such as linear, quadratic and cubic). This is equivalent to fitting a multiple linear regression (or polynomial regression) with orthogonal parameters.

Parameter	Estimates	Null hypothesis
Intercept	y-intercept	H_0: $\beta_0^* = 0$
β_1^*	partial slope for linear term	H_0: $\beta_1^* = 0$
β_2^*	partial slope for quadratic term	H_0: $\beta_2^* = 0$
...		

```
> # specify that orthogonal polynomial contrasts should be used
> contrasts(A) <-contr.poly
> summary(lm(Y~A))
Call:
lm(formula = Y ~ A)

Residuals:
       Min         1Q     Median         3Q        Max
-1.000e+00 -1.000e+00 -1.712e-16  1.000e+00  1.000e+00

Coefficients:
              Estimate Std. Error   t value Pr(>|t|)
(Intercept)  7.000e+00  3.333e-01    21.000  7.6e-07 ***
A.L          5.657e+00  5.774e-01     9.798  6.5e-05 ***
A.Q         -9.890e-16  5.774e-01 -1.71e-15        1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 6 degrees of freedom
Multiple R-squared: 0.9412,     Adjusted R-squared: 0.9216
F-statistic:    48 on 2 and 6 DF,  p-value: 0.0002035
```

User defined contrasts - In addition to the 'prefabricated' sets of comparisons illustrated above, it is possible to define other contrast combinations that are specifically suited to a particular experimental design and set of research questions. Contrasts are defined by constructing a contrast matrix according to the following rules:

(i) groups to be included and excluded in a specific contrasts (comparison) are represented by non-zero and zero coefficients respectively

(ii) groups to be apposed (contrasted) to one another should have apposing signs

(iii) the number of contrasts must not exceed $p - 1$[e], where p is the number of groups.
(iv) within a given contrast, the sum of positive coefficients (and negative coefficients) should sum to 1 to ensure that the resulting estimates can be sensibly interpreted
(v) all the contrasts must be orthogonal (independent of one another)

```
> # define potential contrast matrix for comparing group G1 with
> # the average of groups G2 and G3
> contrasts(A) <- cbind(c(1, -0.5, -0.5))
> contrasts(A)
   [,1]          [,2]
G1  1.0 -6.407635e-17
G2 -0.5 -7.071068e-01
G3 -0.5  7.071068e-01
> l <- lm(Y~A)
> # summarize the model fitting
> summary(l)
Call:
lm(formula = Y ~ A)

Residuals:
       Min          1Q     Median          3Q         Max
-1.000e+00 -1.000e+00 -4.163e-17  1.000e+00  1.000e+00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.0000     0.3333  21.000  7.6e-07 ***
A1           -4.0000     0.4714  -8.485 0.000147 ***
A2            2.8284     0.5774   4.899 0.002714 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1 on 6 degrees of freedom
Multiple R-squared: 0.9412,     Adjusted R-squared: 0.9216
F-statistic:    48 on 2 and 6 DF,  p-value: 0.0002035
```

By default, R[f] employs treatment contrasts for unordered factors[g] and orthogonal polynomial contrasts for ordered factors, although this behavior can be altered to an alternative (such as `contr.sum` for unordered factors) using the `options(contrasts =c("contr.sum", "contr.poly"))` *function*.

[e] Actually, it must equal $p - 1$ exactly. However, it is usually sufficient to define less than $p - 1$ contrasts and let R generate the remaining contrasts.
[f] Note that the default behaviour of S-PLUS is to employ sum to zero contrasts for unordered factors.
[g] Unordered factors are factors that have not specifically defined as 'ordered', see section 2.6.1. The order of groups in an ordered factor is usually important - for example when examining polynomial trends across groups.

Note that while the estimates and interpretations of individual model parameters differ between the alternative approaches, in all but the $\mu = 0$ (set-to-zero) case, the overall effects model is identical ($y_{qj} = \mu + \alpha^*_q + \varepsilon_{qj}$). Hence, the overall null hypothesis tested from the effects model (H_0: $\alpha^*_1 = \alpha^*_2 = ... = 0$) is the same irrespective of the contrasts chosen.

When the model contains more than one factor, a separate term is assigned for each factor and possibly the interactions between factors (e.g. $\alpha_i + \beta_j + \alpha\beta_{ij}$). Alternatively, statistical models containing factors can be expressed as *cell means models* in which the overall mean and treatment effects ($\mu + \alpha_i$) are replaced by the treatment (cell) means (μ_i). In the cell means model, there are as many cell means as there are unique treatment levels. These differences are thus summarized:

Linear model	$y_{ij} = \mu + \beta_1(dummy_1)_{ij} + \beta_2(dummy_2)_{ij} + ... + \varepsilon_{ij}$
Linear effects model	$y_{ij} = \mu + \alpha_i + \varepsilon_{ij}$
Orthogonal linear effects model	$y_{i^*j} = \mu + \alpha^*_{i^*} + \varepsilon_{i^*j}$
Cell means model	$y_{ij} = \mu_i + \varepsilon_{ij}$

For simple model fitting the choice of model type makes no difference, however for complex factorial models in which entire treatment levels (cells) are missing, full effects models cannot be fitted and therefore cell means models must be used.

7.3.2 Linear model hypothesis testing

Hypothesis testing is usually concerned with evaluating whether a population parameter is (or set of parameters are) equal to zero, as this signifies no 'relationship' or 'effect'.

Null hypotheses about individual model parameters

In a linear model, there is a null hypothesis associated with each of the individual model parameters (typically that the parameter is equal to zero), although not all the testable null hypotheses are necessarily biologically meaningful. Consider again the simple linear regression model:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

This linear model includes two parameters (β_0 and β_1), and thus there are two individual testable null hypotheses - that the population y-intercept is equal to zero (H_0: $\beta_0 = 0$) and the slope is equal to zero (H_0: $\beta_1 = 0$). While rejecting a null hypothesis that the slope parameter equals zero indicates the presence of a 'relationship', discovering that the value of the response variable when the predictor variable is equal to zero is usually of little biological relevance.

Null hypotheses about individual model parameters are usually tested using a *t*-test (see section 6.3), or equivalently via a single factor ANOVA (see chapter 10) with a single degree of freedom. The latter approach is often employed when user-defined contrasts are involved as it enables the results to be expressed in the context of the overall linear model (see below and section 10.6).

Null hypotheses about the fit of overall model

Recall that in hypothesis testing, a null hypothesis (H_0) is formulated to represent all possibilities except the hypothesized prediction and that disproving the null hypothesis provides evidence that some alternative hypothesis (H_A) is true. Consequently, there are typically at least two models fitted. The *reduced model*, in which the parameter of interest (and its associated predictor variable) is absent (or equivalently set to zero) represents the model predicted by null hypothesis. The *full model* represents the alternative hypothesis and includes the term of interest. For example, to test the null hypothesis that there is no relationship between populations x and y (and thus that the population slope (β_1)= 0):

$$\begin{aligned} \textit{full model } (H_A) \text{ -} \quad & y_i = \beta_0 + \beta_1 x_i + error_i \\ \textit{reduced model } (H_0) \text{ -} \quad & y_i = \beta_0 + 0x_i + error_i \\ & = \beta_0 + error_i \end{aligned}$$

The degree to which a model 'fits' the observed data is determined by the amount of variation that the model fails to explain, and is measured as the sum of the squared differences (termed *SS* or sums-of-squares) between the observed values of the response variable and the values predicted by the model. A model that fits the observed data perfectly will have a *SS* of 0.

The *reduced model* measures the amount of variation left unexplained by the statistical model when the contribution of the parameter and predictor variable (term) of interest is removed (SS_{Total}). The *full model* measures the amount of variation left unexplained by the statistical model when the contribution of the term is included ($SS_{Residual}$). The difference between the reduced and full models (SS_{Model}) is the amount of explained variation attributed to the term of interest. When the null hypothesis is true, the term of interest should not explain any of the variability in the observed data and thus the full model will not fit the observed data any better than the reduced model. That is, the proposed model would not be expected to explain any more of the total variation than it leaves unexplained. If however, the full model fits the data 'significantly' better (unexplained variability is substantially less in the full model compared to the reduced model) than the reduced model, there is evidence to reject the null hypothesis in favour of the alternative hypothesis.

Hypothesis testing formally evaluates this proposition by comparing the ratio of explained and unexplained variation to a probability distribution representing all possible ratios theoretically obtainable when the null hypothesis is true. The total variability in the observed data ($SS_{Residual}$ – *reduced model*) is partitioned into at least two sources.

(i) the variation that is explained by the model (SS_{Model})

$SS_{Model} = \quad SS_{Total}$ (*reduced model*) $-$ $SS_{Residual}$ (*full model*)

(ii) the variation that is unexplained by the model ($SS_{Residual}$)

$SS_{Residual}$ (*full model*)

The number of degrees of freedom (*d.f.*) associated with estimates of each source of variation reflect the number of observations involved in the estimate minus the

Table 7.2 Analysis of variance (ANOVA) table for a simple linear model. n is the number of observations, f_p is the number of parameters in the full model and r_p is the number of parameters in the reduced model.

Source of variation	SS	df	MS	F-ratio
Model	SS_{Model}	$f_p - 1$	$\frac{SS_{Model}}{df_{Model}}$	$\frac{MS_{Model}}{MS_{Residual}}$
Residual	$SS_{Residual}$	$n - f_p$	$\frac{SS_{Residual}}{df_{Residual}}$	
Total	SS_{Total}	$n - r_p$	$\frac{SS_{Residual}}{df_{Residual}}$	

number of other parameters that must have been estimated previously. Just like *SS*, *df* are additive and therefore:

$$df_{Model} = df_{Total}\ (reduced\ model) - df_{Residual}\ (full\ model)$$

Each of the sources of variation are based on a different number of contributing observations. Therefore more comparable, standardized versions are generated by dividing by the appropriate number of (degrees of freedom). These averaged measures of variation (known as mean squares or *MS*) are thus conservative mean measures of variation and importantly, they have known probability distributions (unlike the SS estimates).

The partitioned sources of variation are tabulated in the form of an analysis of variance (ANOVA) table (see Table 7.2), which also includes the ratio (F-ratio) of MS_{Model} to $MS_{Residual}$. When the null hypothesis is true MS_{Model} and $MS_{Residual}$ are expected to be the same, and thus their ratio (F-ratio) should be approximately 1. An F-ratio based on observed data is thus compared to an appropriate F-distribution (theoretical distribution of all possible F-ratios for the set of degrees of freedom) when the null hypothesis is true. If the probability of obtaining such an F-ratio (or one more extreme) is less than a critical value, the null hypothesis is rejected.

When there are multiple predictor variables, in addition to assessing the fit of the overall model, we usually want to determine the effect of individual factors. This is done by comparing the fit of models with and without the specific term(s) associated with that variable.

7.4 Comments about the importance of understanding the structure and parameterization of linear models

An understanding of how to formulate the correct statistical model from a design and set of null hypotheses is crucial to ensure that the correct R syntax (and thus

Table 7.3 Statistical models in R. Lower case letters denote continuous numeric variables and uppercase letters denote factors. Note that the error term is always implicit.

Effects model	R Model formula	Description
$y_i = \beta_0 + \beta_1 x_i$	`y ~ 1 + x` `y ~ x`	Simple linear regression model of `y` on `x` with intercept term included

$y_i = \beta_1 x_i$	`y ~ 0 + x`
	`y ~ -1 + x`
	`y ~ x - 1`
		Simple linear regression model of `y` on `x` with intercept term excluded

$y_i = \beta_0$	`y ~ 1` `y ~ 1 - x`	Simple linear regression model of `y` against the intercept term
$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}$	`y ~ x1 + x2`	Multiple linear regression model of `y` on `x1` and `x2` with the intercept term included implicitly
$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i1}^2$	`y ~ 1 + x + I(x^2)`	Second order polynomial regression of `y` on `x`
	`y ~ poly(x, 2)`	As above, but using orthogonal polynomials
$y_{ij} = \mu + \alpha_i$	`y ~ A`	Analysis of variance of `y` against a single factor `A`
$y_{ijk} = \mu + \alpha_i + \beta_j + \alpha\beta_{ij}$	`y ~ A + B + A:B` `y ~ A*B`	Fully factorial analysis of variance of `y` against `A` and `B`
$y_{ijk} = \mu + \alpha_i + \beta_j$	`y ~ A*B - A:B`	Fully factorial analysis of variance of `y` against `A` and `B` without the interaction term (equivalent to `A + B`)
$y_{ijk} = \mu + \alpha_i + \beta_{j(i)}$	`y ~ B %in% A` `y ~ A/B`	Nested analysis of variance of `y` against `A` and `B` nested within `A`
$y_{ij} = \mu + \alpha_i + \beta(x_{ij} - \overline{x})$	`y ~ A*x` `y ~ A/x`	Analysis of covariance of `y` on `x` at each level of `A`
$y_{ijkl} = \mu + \alpha_i + \beta_{j(i)} + \gamma_k + \alpha\gamma_{ik} + \beta\gamma_{j(i)k}$	`y ~ A + Error(B) + C + A:C + B:C`	Partly nested ANOVA of `y` against a single between block factor (`A`), a single within block factor (`C`) and a single random blocking factor (`B`).

analysis) is employed. This is particularly important for more complex designs which incorporate multiple error strata (such as partly nested ANOVA). Table 7.3 briefly illustrates the ways in which statistical models are represented in R. Moreover, in each of the remaining chapters, the statistical models as well as the appropriate R model formulae for each major form of modeling will be highlighted and demonstrated, thereby providing greater details about use of R in statistical modeling.

8

Correlation and simple linear regression

Correlation and regression are techniques used to examine associations and relationships between continuous variables collected on the same set of sampling or experimental units. Specifically, correlation is used to investigate the degree to which variables change or vary together (covary). In correlation, there is no distinction between dependent (response) and independent (predictor) variables and there is no attempt to prescribe or interpret the causality of the association. For example, there may be an association between arm and leg length in humans, whereby individuals with longer arms generally have longer legs. Neither variable directly causes the change in the other. Rather, they are both influenced by other variables to which they both have similar responses. Hence correlations apply mainly to survey designs where each variable is measured rather than specifically set or manipulated by the investigator.

Regression is used to investigate the nature of a relationship between variables in which the magnitude and changes in one variable (known as the independent or predictor variable) are assumed to be directly responsible for the magnitude and changes in the other variable (dependent or response variable). Regression analyses apply to both survey and experimental designs. Whilst for experimental designs, the direction of causality is established and dictated by the experiment, for surveys the direction of causality is somewhat discretionary and based on prior knowledge. For example, although it is possible that ambient temperature effects the growth rate of a species of plant, the reverse is not logical. As an example of regression, we could experimentally investigate the relationship between algal cover on rocks and molluscan grazer density by directly manipulating the density of snails in different specifically control plots and measuring the cover of algae therein. Any established relationship must be driven by snail density, as this was the controlled variable. Alternatively the relationship could be investigated via a field survey in which the density of snails and cover of algae could be measured from random locations across a rock platform. In this case, the direction of causality (or indeed the assumption of causality) may be more difficult to defend.

In addition to examining the strength and significance of a relationship (for which correlation and regression are equivalent), regression analysis also explores the functional nature of the relationship. In particular, it estimates the rate at which a change in an independent variable is reflected in a change in a dependent variable as

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

well as the expected value of the dependent variable when the independent variable is equal to zero. These estimates can be used to construct a predictive model (equation) that relates the magnitude of a dependent variable to the magnitude of an independent variable, and thus permit new responses to be predicted from new values of the independent variable.

8.1 Correlation

The simplest measure of association between two variables is the sum product of the deviations of each point from the mean center [*e.g.* $\sum(x - \bar{x})(y - \bar{y})$], see Figure. 8.1f. This method essentially partitions the cloud of points up into four quadrants and weighs up the amount in the positive and negative quadrants. The greater the degree to which points are unevenly distributed across the positive and negative quadrants, the greater the magnitude (either negative or positive) of the measure of association. Clearly however, the greater the number of points, the higher the measure of association. Covariance standardizes for sample size by dividing this measure by the degrees of freedom (number of observation pairs minus 1) and thus represents the average deviations from the mean center. Note that covariance is really the bivariate variance of two variables[a].

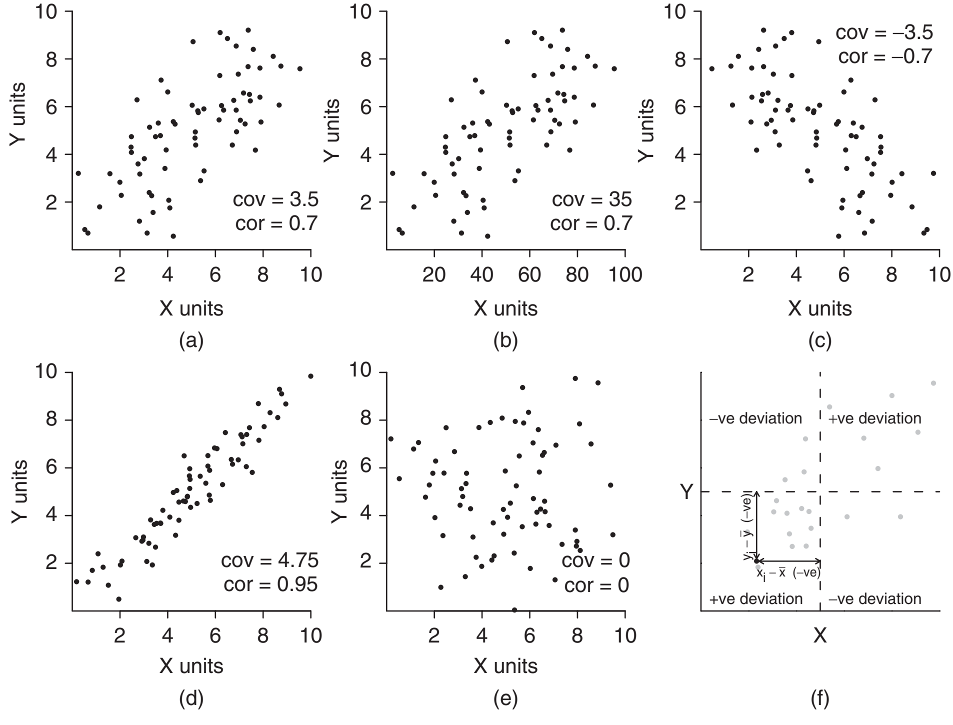


Fig 8.1 Fictitious data illustrating covariance, correlation, strength and polarity.

[a] Covariance of a single variable and itself is the variance of that variable.

8.1.1 Product moment correlation coefficient

Unfortunately, there are no limits on the range of covariance as its magnitude depends on the scale of the units of the variables (see Figure 8.1a-b). The Pearson's (product moment) correlation coefficient further standardizes covariance by dividing it by the standard deviations of x and y, thereby resulting in a standard coefficient (ranging from -1 to $+1$) that represents the strength and polarity of a linear association.

8.1.2 Null hypothesis

Correlation tests the H_0 that the population correlation coefficient (ρ, estimated by the sample correlation coefficient, r) equals zero:

$$H_0 : \rho = 0 \qquad \text{(the population correlation coefficient equals zero)}$$

This null hypothesis is tested using a t statistic ($t = \frac{r}{s_r}$), where s_r is the standard error of r. This t statistic is compared to a t distribution with $n - 2$ degrees of freedom.

8.1.3 Assumptions

In order that the calculated t-statistic should reliably represent the population trends, the following assumptions must be met:

(i) linearity - as the Pearson correlation coefficient measures the strength of a linear (straight-line) association, it is important to establish whether or not some other curved relationship represents the trends better. Scatterplots are useful for exploring linearity.
(ii) normality - the calculated t statistic will only reliably follow the theoretical t distribution when the joint XY population distribution is bivariate normal. This situation is only satisfied when both individual populations (X and Y) are themselves normally distributed. Boxplots should be used to explore normality of each variable.

Scale transformations are often useful to improve linearity and non-normality.

8.1.4 Robust correlation

For situations when one or both of the above assumptions are not met and transformations are either unsuccessful or not appropriate (particularly, proportions, indices and counts), monotonic associations (general positive or negative - not polynomial) can be investigated using non-parametric (rank-based) tests. The **Spearman's rank correlation coefficient** (r_s) calculates the product moment correlation coefficient on the ranks of the x and y variables and is suitable for samples with between 7 and 30 observations. For greater sample sizes, an alternative rank based coefficient **Kendall's** (τ) is more suitable. Note that non-parametric tests are more conservative (have less power) than parametric tests.

8.1.5 Confidence ellipses

Confidence ellipses are used to represent the region on a plot within which we have a certain degree of confidence (e.g 95%) the true population mean center is likely to occur. Such ellipses are centered at the sample mean center and oriented according to the covariance matrix[b] of x and y.

8.2 Simple linear regression

Simple linear regression is concerned with generating a mathematical equation (model) that relates the magnitude of dependent (response) variable to the magnitude of the independent (predictor) variable. The general equation for a straight line is $y = bx + a$, where a is the y-intercept (value of y when $x = 0$) and b is the gradient or slope (rate at which y changes per unit change in x).

Figure 8.2 illustrates sets of possible representatives of population trends between two variables. It should be apparent that if the population slope (β_1) is equal to zero there is no relationship between dependent (Y) and independent variables (X). Changes in the independent variable are not reflected by the dependent variable. Conversely, when the population slope is not equal to zero there is a relationship. Note that the population intercept (β_0) has less biological meaning.

The population parameters (β_0 and β_1) are estimated from a line of best fit through the cloud of sample data. There are a number of ways to determine the line of best fit, each of which represent different approach to regression analysis (see Figure 8.4, and section 8.2.5).

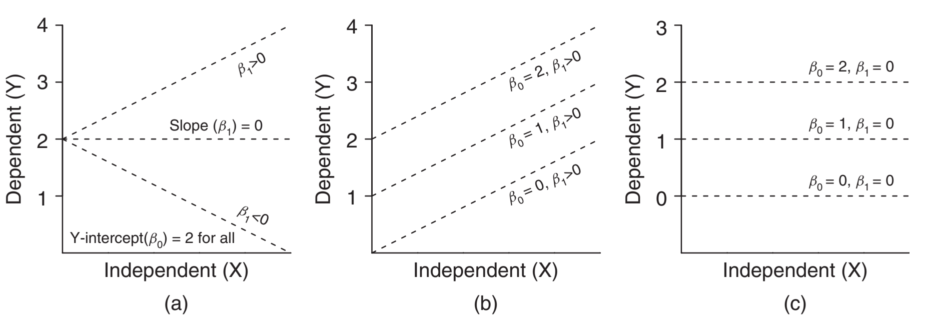


Fig 8.2 Fictitious data contrasting differences in interpretation between slope (β_1) and y-intercept (β_0) parameters.

[b] The covariance matrix of two variables has two rows and two columns. The upper left and lower right entries represent the variances of x and y respectively and the upper right and lower left entries represent the covariance of x and y.

8.2.1 Linear model

The linear model reflects the equation of the line of best fit:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

where β_0 is the population y-intercept, β_1 is the population slope and ε_i is the random unexplained error or residual component.

8.2.2 Null hypotheses

A separate H_0 is tested for each of the estimated model parameters:

$$H_0 : \beta_1 = 0 \qquad \text{(the population slope equals zero)}$$

This test examines whether or not there is likely to be a relationship between the dependent and independent variables in the population. In simple linear regression, this test is identical to the test that the population correlation coefficient equals zero ($\rho = 0$).

$$H_0 : \beta_0 = 0 \qquad \text{(the population y-intercept equals zero)}$$

This test is rarely of interest as it only tests the likelihood that the background level of the response variable is equal to zero (rarely a biologically meaningful comparison) and does not test whether or not there is a relationship (see Figure 8.4b-c).

These H_0's are tested using a t statistic (e.g. $t = \frac{b}{s_b}$), where s_b is the standard error of b. This t statistic is compared to a t distribution with $n - 2$ degrees of freedom.

Along with testing the individual parameters that make up the linear model via t-tests, linear regression typically also tests the $H_0 : \beta_1 = 0$ by partitioning the total variability in the response variable into a component that is explained by the β_1 term in the *full linear model* ($y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$) and a component of the variance that cannot be explained (residual), see Figure 8.3. As it is only possible to directly determine unexplained variation, the amount of variability explained by the full model (and therefore β_1) is calculated as the difference between the amount left unexplained by a *reduced model* ($y_i = \beta_0 + \varepsilon_i$, which represents the situation when $H_0 : \beta_1 = 0$ is true) and the amount left unexplained by the full model ($y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$).

When the null hypothesis is true (no relationship and therefore $\beta_1 = 0$) and the test assumptions are met, the ratio (F-ratio) of explained to unexplained variability follows a F-distribution. Likewise, full and reduced models respectively with and without the y-intercept could be used to test H_0: $\beta_1 = 0$. For simple linear regression, the t-tests and ANOVA's test equivalent null hypotheses[c], however this is not the case for more complex linear models.

[c] For simple linear regression the F-statistic is equal to the t-value squared ($F = t^2$).

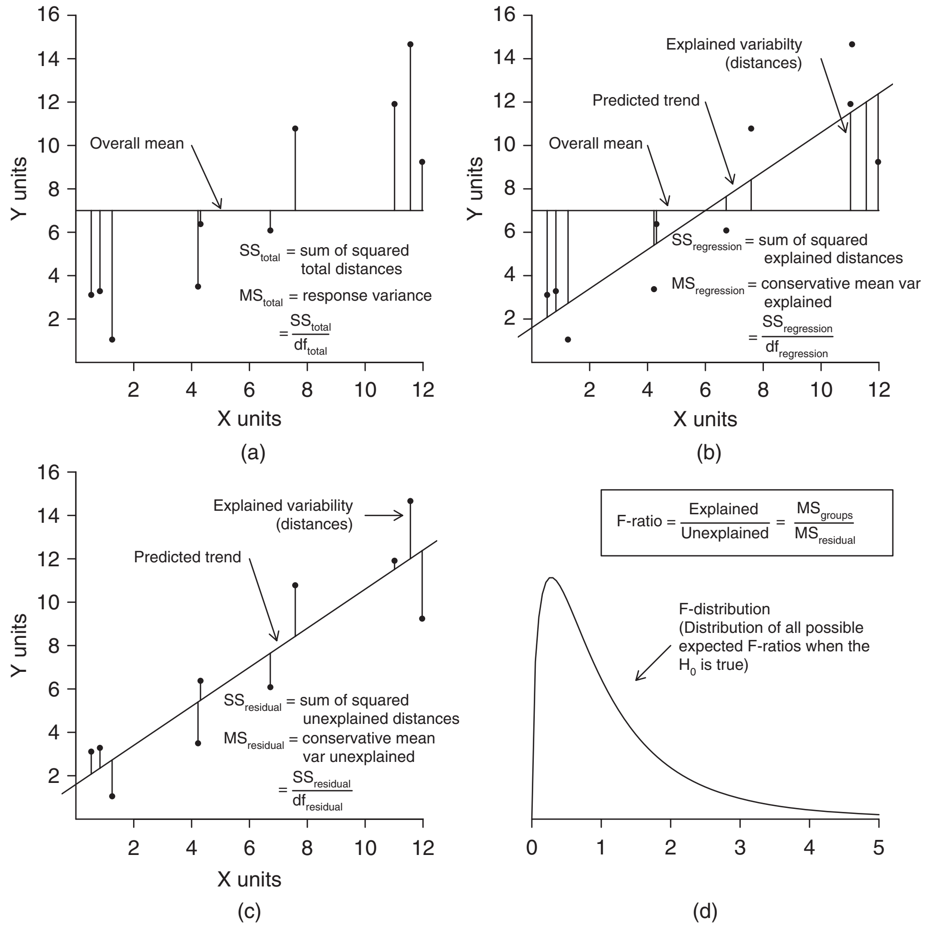


Fig 8.3 Fictitious data illustrating the partitioning of (a) total variation into components (b) explained ($MS_{regression}$) and (c) unexplained ($MS_{residual}$) by the linear trend. The probability of collecting our sample, and thus generating the sample ratio of explained to unexplained variation (or one more extreme), when the null hypothesis is true (and there is no relationship between X and Y) is the area under the F-distribution (d) beyond the sample F-ratio.

8.2.3 Assumptions

To maximize the reliability of null hypotheses tests, the following assumptions apply:

(i) linearity - simple linear regression models a linear (straight-line) relationship and thus it is important to establish whether or not some other curved relationship represents the trends better. Scatterplots are useful for exploring linearity.

(ii) normality - the populations from which the single responses were collected per level of the predictor variable are assumed to be normally distributed. Boxplots of the response variable (and predictor if it was measured rather than set) should be used to explore normality.

(iii) homogeneity of variance - the populations from which the single responses were collected per level of the predictor variable are assumed to be equally varied. With only a single representative of each population per level of the predictor variable, this can only be explored by examining the spread of responses around the fitted regression line. In particular, increasing spread along the regression line would suggest a relationship between population mean and variance (which must be independent to ensure unbiased parameter estimates). This can also be diagnosed with a residual plot.

8.2.4 Multiple responses for each level of the predictor

Simple linear regression assumes linearity and investigates whether there is a relationship between a response and predictor variable. In so doing, it is relying on single response values at each level of the predictor being good representatives of their respective populations. Having multiple independent replicates of each population from which a mean can be calculated thereby provides better data from which to investigate a relationship. Furthermore, the presence of replicates of the populations at each level of the predictor variable enables us to establish whether or not the observed responses differ significantly from their predicted values along a linear regression line and thus to investigate whether the population relationship is linear versus some other curvilinear relationship. Analysis of such data is equivalent to ANOVA with polynomial contrasts (see section 10.6).

8.2.5 Model I and II regression

The **ordinary least squares** (**OLS**, or **model I regression**) fits a line that minimizes the vertical spread of values around the line and is the standard regression procedure. Regression was originally devised to explore the nature of relationship between a measured dependent variable and an independent variable of which the levels where specifically set (and thus controlled) by the researcher to represent a uniform range of possibilities. As the independent variable is set (fixed) rather than measured, there is no uncertainty or error in the *y* values. The coordinates predicted (by the linear model) for any given observation must therefore lie in a vertical plane around the observed coordinates (see Figure 8.4a). The difference between an observed value and its predicted value is called a residual. Hence, OLS regression minimizes the sum of the squared[d] residuals.

Model II regression refers to a family of line fitting procedures that acknowledge and incorporate uncertainty in both response and predictor variables and primarily describe the first major axis through a bivariate normal distribution (see Table 8.1 and Figure 8.4). These techniques generate better parameter estimates (such as population slope) than model I regression when the levels of the predictor variable are measured, however, they are only necessary for situations in which the parameter estimates are the main interest of the analysis. For example, when performing regression analysis

[d] Residuals are squared to remove negatives. Since the regression line is fitted exactly through the middle of the cloud of points, some points will be above this line (+ve residuals) and some points will be below (-ve residuals) and therefore the sum of the residuals will equal exactly zero.

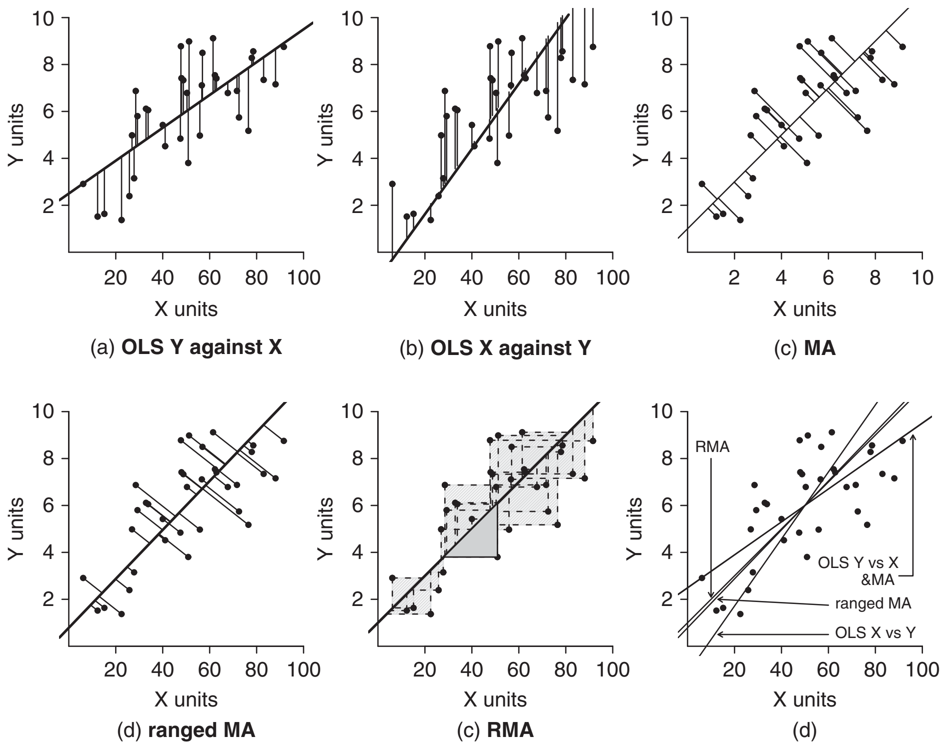


Fig 8.4 Fictitious data illustrating the differences between (a) ordinary least squares, (b) major axis and (c) reduced major axis regression. Each are also contrasted in (d) along with a depiction of ordinary least squares regression for X against Y. Note that the fitted line for all techniques passes through the center mean of the data cloud. When the X and Y are measured on the same scale, MA and RMA are the same.

to estimate the slope in allometric scaling relationships or to compare slopes between models.

Major axis (MA) minimizes the sum square of the perpendicular spread from the regression line (Figure 8.4c) and thus the predicted values line in a perpendicular planes from the regression line. Although this technique incorporates uncertainty in both response and predictor variable, it assumes that the degree of uncertainty is the same on both axes (1:1 ratio) and is therefore only appropriate when both variables are measured on the same scale and with the same units. **Ranged major axis (Ranged MA)** is a modification of major axis regression in which MA regression is performed on variables that are pre-standardized by their ranges (Figure 8.4d) and the resulting parameters are then returned to their original scales. Alternatively, **Reduced major axis (RMA)** minimizes the sum squared triangular areas bounded by the observations and the regression line (Figure 8.4e) thereby incorporating all possible ratios of uncertainty between the response and predictor variables. For this technique, the estimated slope is the average of the slope from a regression of y against x and the inverse of the slope of x against y.

Table 8.1 Comparison of the situations in which the different regression methods are suitable.

Method
Ordinary least squares (OLS) • When there is no uncertainty in *IV* (levels set not measured) or uncertainty in *IV* $\ll$ uncertainty in *DV* • When testing $H_0 : \beta_1 = 0$ (no linear relationship between *DV* and *IV*) • When generating predictive models from which new values of *DV* are predicted from given values of *IV*. Since we rarely have estimates of uncertainty in our new predictor values (and thus must assume there is no uncertainty), predictions likewise must be based on predictive models developed with the assumption of no uncertainty. Note, if there is uncertainty in *IV*, standard errors and confidence intervals inappropriate. • When distribution is not bivariate normal `> summary(lm(DV~IV, data))`

Major axis (MA)
• When a good estimate of the population parameters (slope) is required AND
• When distribution is bivariate normal (*IV* levels not set) AND
• When error variance (uncertainty) in *IV* and *DV* equal (typically because variables in same units or dimensionless)
`> library(biology)`
`> summary(lm.II(DV~IV, data, method='MA'))`

Ranged Major axis (Ranged MA)
• When a good estimate of the population parameters (slope) is required AND
• When distribution is bivariate normal (*IV* levels not set) AND
• When error variances are proportional to variable variances AND
• There are no outliers
`> library(biology)`
`> #For variables whose theoretical minimum is arbitrary`
`> summary(lm.II(DV~IV, data, method='rMA'))`
`> #OR for variables whose theoretical minimum must be zero`
`> #such as ratios, scaled variables & abundances`
`> summary(lm.II(DV~IV, data, method='rMA', zero=T))`

Reduced major axis (RMA) or Standard major axis (SMA)
• When a good estimate of the population parameters (slope) is required AND
• When distribution is bivariate normal (*IV* levels not set) AND
• When error variances are proportional to variable variances AND
• When there is a significant correlation *r* between *IV* and *DV*
`> library(biology)`
`> summary(lm.II(DV~IV, data, method='RMA'))`

Modified from Legendre (2001).

8.2.6 Regression diagnostics

As part of linear model fitting, a suite of diagnostic measures can be calculated each of which provide an indication of the appropriateness of the model for the data and the indication of each points influence (and outlyingness) of each point on resulting the model.

Leverage

Leverage is a measure of how much of an outlier each point is in x-space (on x-axis) and thus only applies to the predictor variable. Values greater than $2 * p/n$ (where p=number of model parameters ($p = 2$ for simple linear regression), and n is the number of observations) should be investigated as potential outliers.

Residuals

As the residuals are the differences between the observed and predicted values along a vertical plane, they provide a measure of how much of an outlier each point is in y-space (on y-axis). Outliers are identified by relatively large residual values. Residuals can also standardized and studentized, the latter of which can be compared across different models and follow a t distribution enabling the probability of obtaining a given residual can be determined. The patterns of residuals against predicted y values (residual plot) are also useful diagnostic tools for investigating linearity and homogeneity of variance assumptions (see Figure 8.5).

Cook's D

Cook's D statistic is a measure of the influence of each point on the fitted model (estimated slope) and incorporates both leverage and residuals. Values ≥ 1 (or even approaching 1) correspond to highly influential observations.

8.2.7 Robust regression

There are a range of model fitting procedures that are less sensitive to outliers and underlying error distributions. **Huber M-estimators** fit linear models by minimizing the sum of differentially weighted residuals. Small residuals (weakly influential) are squared and summed as for OLS, whereas residuals over a preselected critical size (more influential) are incorporated as the sum of the absolute residual values. A useful non-parametric test is the **Theil-Sen single median** (**Kendall's robust**) method which estimates the population slope (β_1) as the median of the $n(n-1)/2$ possible slopes (b_1) between each pair of observations and the population intercept (β_0) is estimated as the median of the n intercepts calculated by solving $y - b_1x$ for each observation. A more robust, yet complex procedure (**Siegel repeated medians**) estimates β_1 and β_0 as the median of the n median of the $n-1$ slopes and intercepts respectively between each point and all others. **Randomization tests** compare the statistic (b_1) to

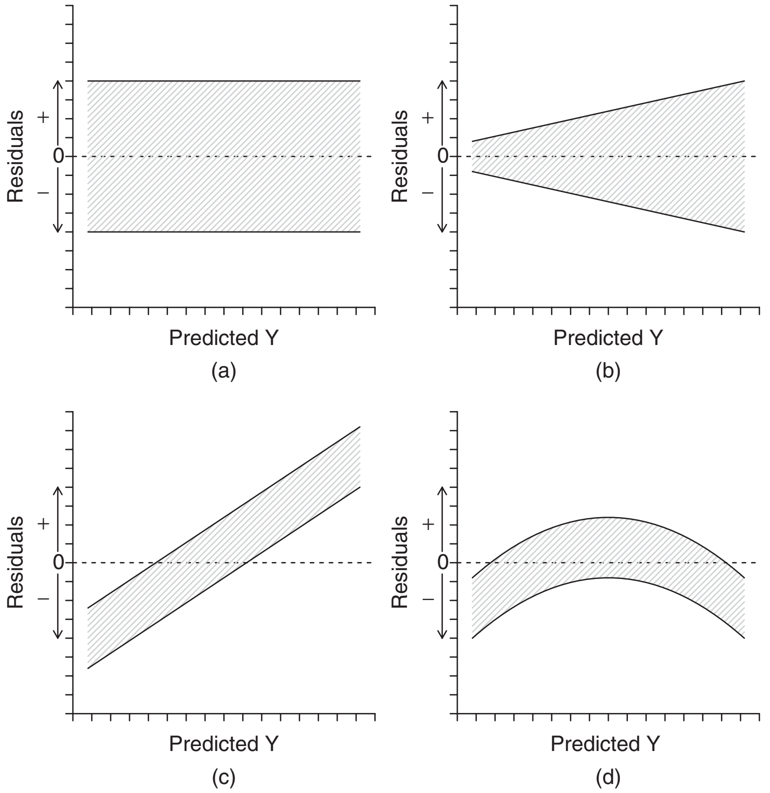


Fig 8.5 Stylised residual plots depicting characteristic patterns of residuals (a) random scatter of points - homogeneity of variance and linearity met (b) "wedge-shaped" - homogeneity of variance not met (c) linear pattern remaining - erroneously calculated residuals or additional variable(s) required and (d) curved pattern remaining - linear function applied to a curvilinear relationship. Modified from Zar (1999).

a unique probability distribution that is generated by repeatedly reshuffling one of the variables and recalculating the test statistic. As a result, they do not impose any distributional requirements on the data. Randomization tests are particularly useful for analysing data that could not be collected randomly or haphazardly as they test whether the patterns in the data could occur by chance rather than specifically testing hypotheses about populations. As a result, technically any conclusions pertain only to the collected observations and not to the populations from which the observations were collected.

8.2.8 Power and sample size determination

Although interpreted differently, the tests $H_0 : \rho = 0$ and $H_0 : \beta_1 = 0$ (population correlation and slope respectively equal zero) are statistically equivalent. Therefore power analyses to determine sample size required for null hypothesis rejection for both correlation and regression are identical and based on r (correlation coefficient), which from regression analyses, can be obtained from the coefficient of determination (r^2) or as $r = b\sqrt{\sum x^2 / \sum y^2}$.

8.3 Smoothers and local regression

Smoothers fit simple models (such as linear regression) through successive localized subsets of the data to describe the nature of relationships between a response variable and one or more predictor variables for each point in a data cloud. Importantly, these techniques do not require the data to conform to a particular global model structure (e.g. linear, exponential, etc). Essentially, smoothers generate a line (or surface) through the data cloud by replacing each observation with a new value that is predicted from the subset of observations immediately surrounding the original observation. The subset of neighbouring observations surrounding an observation is known as a *band* or *window* and the larger the *bandwidth*, the greater the degree of smoothing.

Smoothers can be used as graphical representations as well as to model (local regression) the nature of relationships between response and predictor variables in a manner analogous to linear regression. Different smoothers differ in the manner by which the predicted values are created.

- **running medians** (or less robust running means) generate predicted values that are the medians of the responses in the bands surrounding each observation.
- **loess** and **lowess**[e] (locally weighted scatterplot smoothing) - fit least squares regression lines to successive subsets of the observations weighted according to their distance from the target observation and thus depict changes in the trends throughout the data cloud.
- **kernel smoothers** - new smoothed y-values are computed as the weighted averages of points within a defined window (bandwidth) or neighbourhood of the original x-values. Hence the bandwidth depends on the scale of the x-axis. Weightings are determined by the type of kernel smoother specified, and for. Nevertheless, the larger the window, the greater the degree of smoothing.
- **splines** - join together a series of polynomial fits that have been generated after the entire data cloud is split up into a number of smaller windows, the widths of which determine the smoothness of the resulting piecewise polynomial.

Whilst the above smoothers provide valuable exploratory tools, they also form the basis of the formal model fitting procedures supported via generalized additive models (GAMs, see chapter 17).

8.4 Correlation and regression in R

Simple correlation and regression in R are performed using the `cor.test()` and `lm()` *functions*. The `mblm()` and `rlm()` *functions* offer a range of non-parametric regression

[e] Lowess and loess functions are similar in that they both fit linear models through localizations of the data. They differ in in that loess uses weighted quadratic least squares and lowess uses weighted linear least squares. They also differ in how they determine the data spanning (neighborhood of points regression model fitted to), and in that loess smoothers can fit surfaces and thus accomodate multivariate data.

Table 8.2 Smoothing function within R. For each of the following, `DV` is the response variable within the `data` dataset. Smoothers are plotted on scatterplots by using the smoother function as the response variable in the `points()` function (e.g. `points(runmed(DV)~IV, data, type='l')`).

Smoother[a]	Syntax
Running median	`> runmed(data$DV,k)` *where* `k` *is an odd number that defines the bandwidth of the window and if* `k` *omitted, defaults to either Turlach or Struetzle breaking algorithms depending on data size (Turlack for larger)*
Loess	`> loess(DV~IV1+IV2+..., data, span=0.75)` *where* `IV1, IV2` *represent one or more predictor variables and* `span` *controls the degree of smoothing*
Lowess	`> lowess(data$IV, data$DV, f=2/3)` *where* `IV` *represents the predictor variable and* `f` *controls the degree of smoothing*
Kernel	`> ksmooth(data$IV, data$DV, kernel="normal", bandwidth=0.5)` *where* `IV` *represents the predictor variable,* `kernel` *represents the smoothing kernel (*`box` *or* `normal`*) and* `bandwidth` *is the smoothing bandwidth*
	`> density(data$DV, bw="nrd0", adjust=1)` *where* `IV` *represents the predictor variable and* `bw` *and* `adjust` *"nrd0" the smoothing bandwidth and course bandwidth multiplier respectively. Information on the alternative smoothing bandwidth selectors for gaussian (normal) windows is obtained by typing* `?bw.nrd`

Splines	`> data.spl<-smooth.spline(data$IV, data$DV, spar)`
	 `> points(y~x, data.spl, type='l')`
	 where `IV` *represents the predictor variable and* `spar` *is the smoothing coeficient, typically between 0 and 1.*

[a]Note, there are many other functions and packages that facilitate alternatives to the smoothing functions listed here.

alternatives. Model II regressions are facilitated via the `lm.II()` *function* and the common smoothing functions available in R are described in Table 8.2.

8.5 Further reading

- Theory

Fowler, J., L. Cohen, and P. Jarvis. (1998). *Practical statistics for field biology*. John Wiley & Sons, England.

Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. John Wiley & Sons, New York.

Manly, B. F. J. (1991). *Randomization and Monte Carlo methods in biology*. Chapman & Hall, London.

Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

Dalgaard, P. (2002). *Introductory Statistics with R*. Springer-Verlag, New York.

Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

8.6 Key for correlation and regression

1 a. **Neither variable has been set (they are both measured) AND there is no implied causality between the variables (Correlation)** . . . . Go to 2

b. **Either one of the variables has been specifically set (not measured) OR there is an implied causality between the variables whereby one variable could influence the other but the reverse is unlikely (Regression)** . . . . Go to 4

2 a. **Check parametric assumptions for correlation analysis**

- **Bivariate normality of the response/predictor variables - marginal scatterplot boxplots**

```
> library(car)
> scatterplot(V1 ~ V2, dataset)
```

where `V1` *and* `V2` *are the continuous variables in the* `dataset` *data frame*

- **Linearity of data points on a scatterplot, trendline and lowess smoother useful**

```
> library(car)
> scatterplot(V1 ~ V2, dataset, reg.line = F)
```

where `V1` *and* `V2` *are the continuous variables in the* `dataset` *data frame and* `reg.line=F` *excludes the misleading regression line from the plot*

Parametric assumptions met (Pearson correlation) . . . . See Example 8A

```
> corr.test(~V1 + V2, data = dataset)
```

where `V1` *and* `V2` *are the continuous variables in the* `dataset` *data frame*

For a summary plot . . . . Go to 12

b. **Parametric assumptions NOT met or scale transformations (see Table 3.2) not successful or inappropriate** . . . . Go to 3

3 a. **Sample size between 7 and 30 (Spearman rank correlation)** . . . . . . See Example 8B

```
> cor.test(~V1 + V2, data = dataset, method = "spearman")
```

where `V1` *and* `V2` *are the continuous variables in the* `dataset` *data frame*

For a summary plot . . . . Go to 12

b. **Sample size > 30 (Kendall's tao correlation)**

```
> cor.test(~V1 + V2, data = dataset, method = "kendall")
```

where `V1` *and* `V2` *are the continuous variables in the* `dataset` *data frame*
For a summary plot . . . . Go to 12

4 a. Check parametric assumptions for regression analysis

- **Normality of the response variable (and predictor variable if measured) - marginal scatterplot boxplots**
- **Homogeneity of variance - spread of data around scatterplot trendline**
- **Linearity of data points on a scatterplot, trendline and lowess smoother useful**

```
> library(car)
> scatterplot(DV ~ IV, dataset)
```

where `DV` *and* `IV` *are response and predictor variables respectively in the* `dataset` *data frame*
Parametric assumptions met . . . . Go to 5

b. Parametric assumptions NOT met or scale transformations (see Table 3.2) not successful or inappropriate . . . . Go to 7

5 a. Levels of predictor variable set (not measured) - no uncertainty in predictor variable OR the primary aim of the analysis is:

- **hypothesis testing** ($H_0 : \beta_1 = 0$)
- **generating a predictive model** ($y = \beta_0 + \beta_1 x$)

(Ordinary least squares (OLS) regression) . . . . Go to 6

b. Levels of predictor variable NOT set (they are measured) AND the main aim of the analysis is to estimate the population slope of the relationship (Model II regression) . . . . See Example 8F

```
> library(biology)
> data.lm <- lm.II(DV ~ IV, christ, type = "RMA")
> summary(data.lm)
```

where `DV` *and* `IV` *are response and predictor variables respectively in the* `dataset` *data frame.* `type` *can be one of* `"MA"`, `"RMA"`, `"rMA"` *or* `"OLS"`*. For* `type="rMA"`, *it is also possible to force a minimum response of zero* (`zero=T`).
To produce a summary plot . . . . Go to 12

6 a. Single response value for each level of the predictor variable . . . . See Examples 8C&8D

```
> dataset.lm <- lm(IV ~ DV, dataset)
> plot(dataset.lm)
> influence.measures(dataset.lm)
> summary(dataset.lm)
```

where `DV` *and* `IV` *are response and predictor variables respectively in the* `dataset` *data frame.*
To get parameter confidence intervals[f] . . . . Go to 10
To predict new values of the response variable . . . . Go to 11
To produce a summary plot . . . . Go to 12

b. Multiple response values for each level of the predictor variable . . . . See Examples 8E

```
> anova(lm(DV ~ IV + as.factor(IV), dataset))
```

[f] If there is uncertainty in the predictor variable, parameter confidence intervals might be inappropriate.

- Pooled residual term

```
> dataset.lm <- lm(DV ~ IV, dataset)
> summary(dataset.lm)
```

- Non-pooled residual term

```
> dataset.lm <- aov(DV ~ IV + Error(as.factor(IV)), dataset)
> summary(dataset.lm)
> lm(DV ~ IV, dataset)
```

where `DV` *and* `IV` *are response and predictor variables respectively in the* `dataset` *data frame.*

7 a. **Observations collected randomly/haphazardly, no reason to suspect non-independence** .... Go to 8

b. **Random/haphazard sampling not possible, observations not necessarily independent (Randomization test)** .... See Example 8H

```
> stat <- function(data, index) {
+     summary(lm(DV ~ IV, data))$coef[2, 3]
+ }
> rand.gen <- function(data, mle) {
+     out <- data
+     out$IV <- sample(out$IV, replace = F)
+     out
+ }
> library(boot)
> dataset.boot <- boot(dataset, stat, R = 5000,
+     sim = "parametric", ran.gen = rand.gen)
> plot(dataset.boot)
> dataset.boot
```

where `DV` *and* `IV` *are response and predictor variables respectively in the* `dataset` *data frame.*

To get parameter confidence intervals[g] .... Go to 10
To predict new values of the response variable .... Go to 11
To produce a summary plot .... Go to 12

8 a. **Mild non-normality due mainly to outliers (influential obseravations), data linear (M-regression)**

```
> library(MASS)
> data.rlm <- rlm(DV ~ IV, dataset)
```

where `DV` *and* `IV` *are response and predictor variables respectively in the* `dataset` *data frame.*

To get parameter confidence intervals[h] .... Go to 12
To predict new values of the response variable .... Go to 11
To produce a summary plot .... Go to 10

[g] If there is uncertainty in the predictor variable, parameter confidence intervals might be inappropriate.
[h] If there is uncertainty in the predictor variable, parameter confidence intervals might be inappropriate.

b. Data non-normal and/or non-linear .... Go to 9

9 **a. Binary response (e.g. dead/alive, present/absent)** ...... Logistic Regression chapter 17

b. Underlying distribution of response variable and residuals is known ...... GLM chapter 17

c. Data curvilinear .... Non-linear regression chapter 9

d. Data monotonic non-linear (nonparametric regression) .... See Example 8G

- Theil-Sen single median (Kendall's) robust regression

```
> library(mblm)
> data.mblm <- mblm(DV ~ IV, dataset, repeated = F)
> summary(data.mblm)
```

- Siegel repeated medians regression

```
> library(mblm)
> data.mblm <- mblm(DV ~ IV, dataset, repeated = T)
> summary(data.mblm)
```

where `DV` *and* `IV` *are response and predictor variables respectively in the* `dataset` *data frame.*

To get parameter confidence intervals[i] .... Go to 12

To predict new values of the response variable .... Go to 11

To produce a summary plot .... Go to 10

10 **Generating parameter confidence intervals** .... See Example 8C&8G

```
> confint(model, level = 0.95)
```

where `model` *is a fitted model*

To get randomization parameter estimates and their confidence intervals .... See Example 8H

```
> par.boot <- function(dataset, index) {
+     x <- dataset$ALT[index]
+     y <- dataset$HK[index]
+     model <- lm(y ~ x)
+     coef(model)
+ }
> dataset.boot <- boot(dataset, par.boot, R = 5000)
> boot.ci(dataset.boot, index = 2)
```

where `dataset` *is the data.frame. The optional argument (*`R=5000`*) indicates 5000 randomizations and the optional argument (*`index=2`*) indicates which parameter to generate confidence intervals for (y-intercept=1, slope=2). Note the use of the* `lm()` *function for the parameter estimations and could be replaced by robust alternatives such as* `rlm()` *or* `mblm()`.

11 **Generating new response values (and corresponding prediction intervals)** ..... See Example 8C&8D

```
> predict(model, data.frame(IV = c()), interval = "p")
```

[i] If there is uncertainty in the predictor variable, parameter confidence intervals might be inappropriate.

where `model` *is a fitted model and* `IV` *is the predictor variable and* `c()` *is a vector of new predictor values (e.g.* `c(10,13.4)`*)*
To get randomization prediction intervals .... See Example 8H

```
> pred.boot <- function(dataset, index) {
+     dataset.rs <- dataset[index, ]
+     dataset.lm <- lm(HK ~ ALT, dataset.rs)
+     predict(dataset.lm, data.frame(ALT = 1))
+ }
> dataset.boot <- boot(dataset, pred.boot, R = 5000)
> boot.ci(dataset.boot)
```

where `dataset` *is the name of the data frame. Note the use of the* `lm()` *function for the parameter estimations. This could be replaced by robust alternatives such as* `rlm()` *or* `mblm()`.

12 **Base summary plot for correlation or regression** ...... See Example 8B&8C&8D&8F

```
> plot(V1 ~ V2, data, pch = 16, axes = F, xlab = "", ylab = "")
> axis(1, cex.axis = 0.8)
> mtext(text = "x-axis title", side = 1, line = 3)
> axis(2, las = 1)
> mtext(text = "y-axis title", side = 2, line = 3)
> box(bty = "l")
```

where `V1` *and* `V2` *are the continuous variables in the* `dataset` *data frame. For regression,* `V1` *represents the response variable and* `V2` *represents the predictor variable.*
Adding confidence ellipse .... See Example 8B

```
> data.ellipse(V2, V1, levels = 0.95, add = T)
```

Adding regression line .... See Example 8C

```
> abline(model)
```

where `model` *represents a fitted regression model*
Adding regression confidence intervals .... See Example 8C&8D

```
> x <- seq(min(IV), max(IV), l = 1000)
> y <- predict(object, data.frame(IV = x), interval = "c")
> matlines(x, y, lty = 1, col = 1)
```

where `IV` *is the name of the predictor variable (including the dataframe)* `model` *represents a fitted regression model*

8.7 Worked examples of real biological data sets

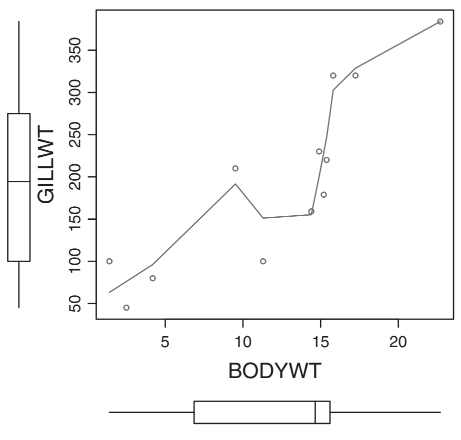
Example 8A: Pearson's product moment correlation
Sokal and Rohlf (1997) present an unpublished data set (L. Miller) in which the correlation between gill weight and body weight of the crab (*Pachygrapsus crassipes*) is investigated.

Step 1 - Import (section 2.3) the crabs data set

```
> crabs <- read.table("crabs.csv", header = T, sep = ",")
```

Step 2 (Key 8.2) - Assess linearity and bivariate normality using a scatterplot with marginal boxplots

```
> library(car)
> scatterplot(GILLWT ~ BODYWT, data = crabs, reg.line = F)
```



Conclusions - data not obviously nonlinear and no evidence of non-normality (boxplots not asymmetrical)

Step 3 (Key 8.2a) - Calculate the Pearson's correlation coefficient and test $H_0 : \rho = 0$ (that the population correlation coefficient equals zero).

```
> cor.test(~GILLWT + BODYWT, data = crabs)
        Pearson's product-moment correlation

data:  GILLWT and BODYWT
t = 5.4544, df = 10, p-value = 0.0002791
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.5783780 0.9615951
sample estimates:
      cor
0.8651189
```

Conclusions - reject H_0 that population correlation coefficient equals zero, there was a strong positive correlation between crab weight and gill weight ($r = 0.865, t_{10} = 5.45$, $P < 0.001$).

Example 8B: Spearman rank correlation

Green (1997) investigated the correlation between total biomass of red land crabs (*Gecarcoidea natalis* and the density of their burrows at a number of forested sites (Lower site: LS and Drumsite: DS) on Christmas Island.

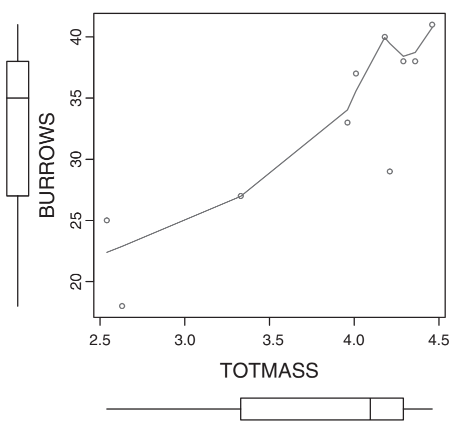
Step 1 - Import (section 2.3) the Green (1997) data set

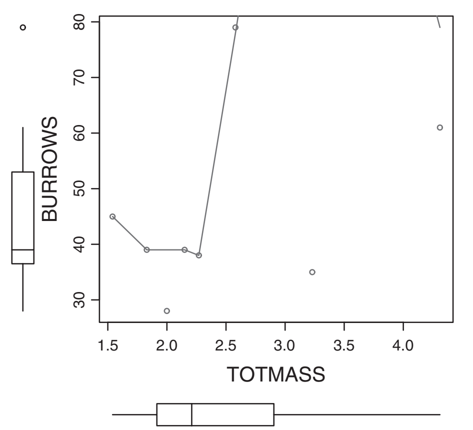
```
> green <- read.table("green.csv", header = T, sep = ",")
```

Step 2 (Key 8.2) - Assess linearity and bivariate normality for the two sites separately using a scatterplots with marginal boxplots

```
> library(car)
> scatterplot(BURROWS ~ TOTMASS,
+     data = green, subset =
+         SITE == "LS",
+         reg.line = F)
```

```
> library(car)
> scatterplot(BURROWS ~ TOTMASS,
+     data = green, subset =
+         SITE == "DS",
+         reg.line = F)
```





Conclusions - some evidence of non-normality (boxplots not asymmetrical)

Step 3 (Key 8.3a) - Calculate the Spearman's rank correlation coefficient and test $H_0 : \rho = 0$ (that the population correlation coefficient equals zero).

```
> cor.test(~BURROWS + TOTMASS, data = green, subset = SITE ==
+     "LS", method = "spearman")
        Spearman's rank correlation rho

data:  BURROWS and TOTMASS
S = 24.5738, p-value = 0.001791
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.8510678
```

Conclusions - reject H_0 that population correlation coefficient equals zero, there was a strong positive correlation between crab biomass and burrow density at Low site ($\rho = 0.851, S_{10} = 24.57, P = 0.0018$).

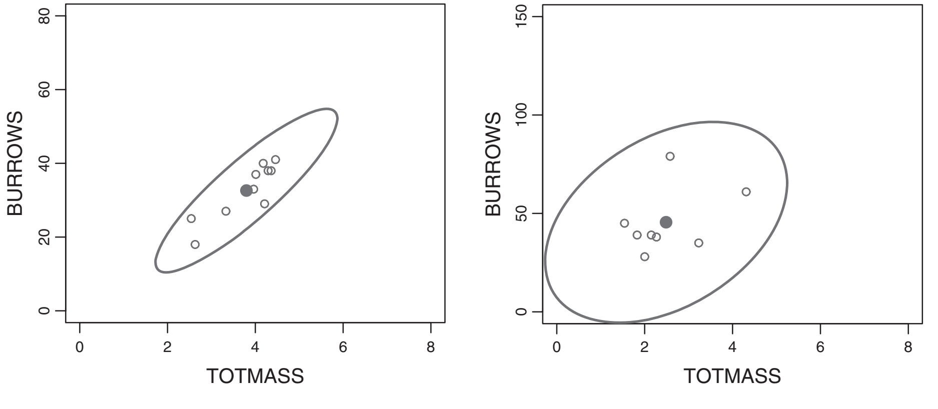
```
> cor.test(~BURROWS + TOTMASS, data = green, subset = SITE ==
+     "DS", method = "spearman")
         Spearman's rank correlation rho

data:  BURROWS and TOTMASS
S = 69.9159, p-value = 0.6915
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.1676677
```

Conclusions - do not reject H_0 that population correlation coefficient equals zero, there was no detectable correlation between crab weight and gill weight at Drumsite ($\rho = 0.168, S_{10} = 69.92, P = 0.692$).

Step 4 (Key 8.12) - Summarize findings with scatterplots (section 5.8.1), including 95% confidence ellipses for the population bivariate mean center. The following also indicate two alternative ways to specify a subset of a dataframe.

```
> plot(BURROWS ~ TOTMASS,
+     data = green, subset =
+         SITE == "LS",
+         xlim = c(0,
+         8), ylim = c(0,
+         80))
> with(subset(green, SITE ==
+     "LS"), data.ellipse
+     (TOTMASS,
+     BURROWS, levels = 0.95,
+     add = T))
```

```
> plot(BURROWS ~ TOTMASS,
+     data = green, subset =
+         SITE == "DS",
+          xlim = c(0,
+         8), ylim = c(0,
+         150))
> with(subset(green, SITE ==
+     "DS"), data.ellipse
+     (TOTMASS,
+     BURROWS, levels = 0.95,
+     add = T))
```



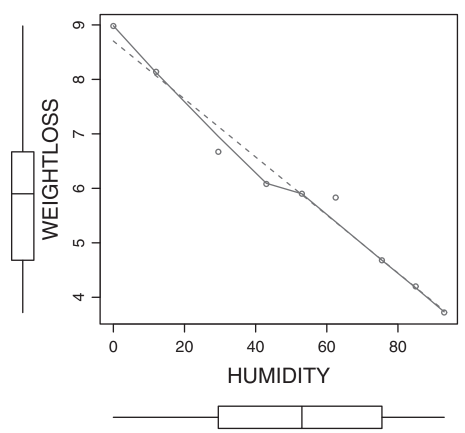
Example 8C: Simple linear regression - fixed X
As part of a Ph.D into the effects of starvation and humidity on water loss in the confused flour beetle (*Tribolium confusum*), Nelson (1964) investigated the linear relationship between humidity and water loss by measuring the amount of water loss (mg) by nine batches of beetles kept at different relative humidities (ranging from 0 to 93%) for a period of six days (Table 14.1 Sokal and Rohlf (1997)).

Step 1 - Import (section 2.3) the Nelson (1964) data set

```
> nelson <- read.table("nelson.csv", header = T, sep = ",")
```

Step 2 (Key 8.4) - Assess linearity, normality and homogeneity of variance using a scatterplot with marginal boxplots and a lowess smoother.

```
> library(car)
> scatterplot(WEIGHTLOSS ~ HUMIDITY, data = nelson)
```

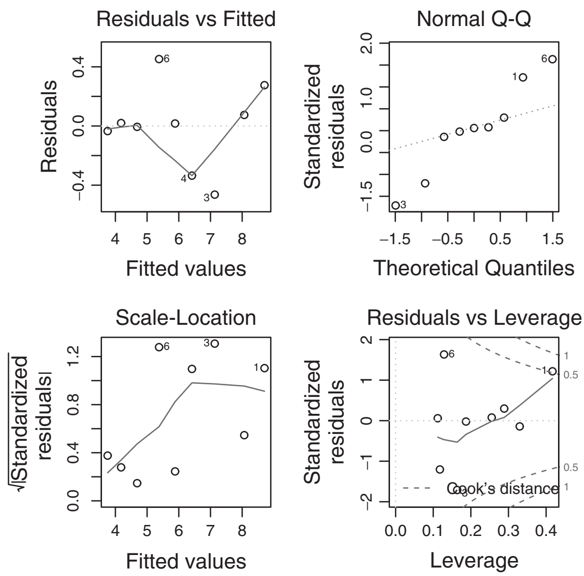


Conclusions - no evidence of non-normality (boxplots not overly asymmetrical), non homogeneity of variance (points do not become progressively more or less spread out along the regression line) or non-linearity.

Step 3 (Key 8.5a) - the ordinary least squares method is considered appropriate as the there is effectively no uncertainty (error) in the predictor variable (relative humidity).

Step 4 (Key 8.6a) - fit the simple linear regression model ($y_i = \beta_0 + \beta_1 x_i$) and examine the diagnostics.

```
> nelson.lm <- lm(WEIGHTLOSS ~ HUMIDITY, nelson)
> plot(nelson.lm)
```



Conclusions - There is no obvious "wedge" pattern evident in the residual plot (confirming that the assumption of homogeneity of variance is likely to be met). Although there is some deviation in the Q-Q normal plot (suggesting that the response variable does deviate from normal), the sample size is rather small and the test is reasonably robust to such deviations. Finally, none of the points approach the high Cook's D contours suggesting that none of the observations are overly influential on the final fitted model.

```
> influence.measures(nelson.lm)
Influence measures of
         lm(formula = WEIGHTLOSS ~ HUMIDITY, data = nelson) :

    dfb.1_ dfb.HUMI    dffit cov.r   cook.d   hat inf
1  1.07457 -0.92033  1.07457 1.449 5.31e-01 0.417   *
2  0.17562 -0.13885  0.17705 1.865 1.81e-02 0.289   *
3 -0.83600  0.52023 -0.91800 0.552 2.86e-01 0.164
4 -0.32184  0.10806 -0.45713 0.970 9.67e-02 0.118
5  0.00868  0.00169  0.01969 1.531 2.26e-04 0.112
6  0.11994  0.27382  0.73924 0.598 1.97e-01 0.129
7  0.00141 -0.00609 -0.00956 1.674 5.33e-05 0.187
8 -0.01276  0.03163  0.04208 1.825 1.03e-03 0.255
9  0.03662 -0.07495 -0.09204 2.019 4.93e-03 0.330   *
```

Conclusions - None of the leverage (hat) values are greater than $2 * p/n = 0.444$ and therefore (none are considered to be outliers in x-space). Furthermore, none of the Cook's D values are ≥ 1 (no point is overly influential). Hence there is no evidence that hypothesis tests will be unreliable.

Step 5 (Key 8.6a) - examine the parameter estimates and hypothesis tests (Boxes 14.1 & 14.3 of Sokal and Rohlf (1997)).

```
> summary(nelson.lm)
Call:
lm(formula = WEIGHTLOSS ~ HUMIDITY, data = nelson)

Residuals:
     Min       1Q   Median       3Q      Max
-0.46397 -0.03437  0.01675  0.07464  0.45236

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.704027   0.191565   45.44 6.54e-10 ***
HUMIDITY    -0.053222   0.003256  -16.35 7.82e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2967 on 7 degrees of freedom
Multiple R-squared: 0.9745,     Adjusted R-squared: 0.9708
F-statistic: 267.2 on 1 and 7 DF,  p-value: 7.816e-07
```

Conclusions - Reject H_0 that the population slope equals zero. An increase in relative humidity was found to be associated with a strong ($r^2 = 0.975$), significant decrease in weight loss ($b = -0.053, t_7 = -16.35, P < 0.001$) in confused flour beetles.

Step 6 (Key 8.10) - calculate the 95% confidence limits for the regression coefficients (Box 14.3 of Sokal and Rohlf (1997)).

```
> confint(nelson.lm)
                  2.5 %      97.5 %
(Intercept)  8.25104923  9.15700538
HUMIDITY    -0.06092143 -0.04552287
```

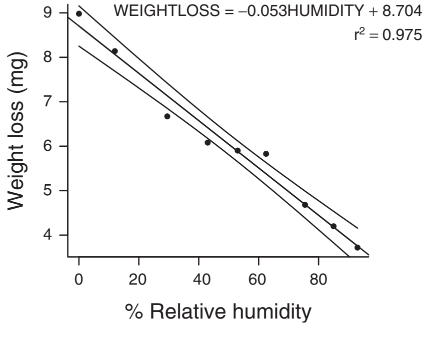
Step 7 (Key 8.11) - use the fitted linear model to predict the mean weight loss of flour beetles expected at 50 and 100% relative humidity (Box 14.3 of Sokal and Rohlf (1997)).

```
> predict(nelson.lm, data.frame(HUMIDITY = c(50, 100)),
+     interval = "prediction", se = T)
$fit
       fit      lwr      upr
1 6.042920 5.303471 6.782368
2 3.381812 2.549540 4.214084

$se.fit
        1         2
0.0988958 0.1894001
```

```
$df
[1] 7

$residual.scale
[1] 0.2966631
```

Step 8 (Key 8.12) - summarize the findings of the linear regression analysis with a scatterplot including the regression line, regression equation and r^2.

```
> #create a plot with solid dots (pch=16) and no axis or labels
> plot(WEIGHTLOSS~HUMIDITY, data=nelson, pch=16, axes=F, xlab="",
      ylab="")
> #put the x-axis (axis 1) with smaller label font size
> axis(1, cex.axis=.8)
> #put the x-axis label 3 lines down from the axis
> mtext(text="% Relative humidity", side=1, line=3)
> #put the y-axis (axis 2) with horizontal tick labels
> axis(2, las=1)
> #put the y-axis label 3 lines to the left of the axis
> mtext(text="Weight loss (mg)", side=2, line=3)
> #add the regression line from the fitted model
> abline(nelson.lm)
> #add the regression formula
> text(99,9,"WEIGHTLOSS = -0.053HUMIDITY + 8.704", pos=2)
> #add the r squared value
> text(99,8.6,expression(paste(r^2==0.975)), pos=2)
> #create a sequence of 1000 numbers spanning the range of
     humidities
> x <- seq(min(nelson$HUMIDITY), max(nelson$HUMIDITY),l=1000)
> #for each value of x, calculate the upper and lower 95%
     confidence
> y<-predict(nelson.lm, data.frame(HUMIDITY=x), interval="c")
> #plot the upper and lower 95% confidence limits
> matlines(x,y, lty=1, col=1)
> #put an L-shaped box to complete the axis
> box(bty="l")
```



Example 8D: Simple linear regression - random X

To investigated the nature of abundance-area relationships for invertebrates in intertidal mussel clumps, Peake and Quinn (1993) measured area (mm^2) (dependent variable: AREA) and number of non-mussel individuals supported (response variable: INDIV) from a total of 25 intertidal mussel clumps(from Box 5.4 of Quinn and Keough (2002)).

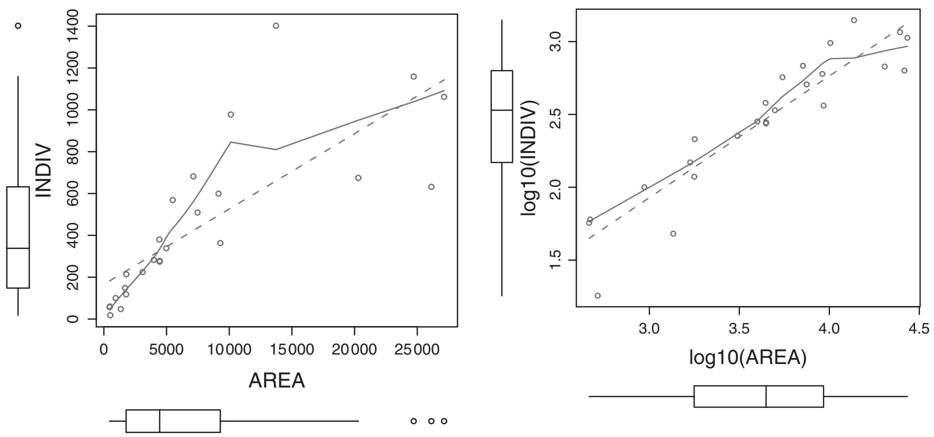
Step 1 - Import (section 2.3) the Peake and Quinn (1993) data set

```
> peake <- read.table("peake.csv", header = T, sep = ",")
```

Step 2 (Key 8.4) - Assess linearity, normality and homogeneity of variance using a scatterplot with marginal boxplots and a lowess smoother.

```
> library(car)
> scatterplot(INDIV ~ AREA,
+     data = peake)
```

```
> library(car)
> scatterplot(log10(INDIV) ~
+     log10(AREA), data = peake)
```

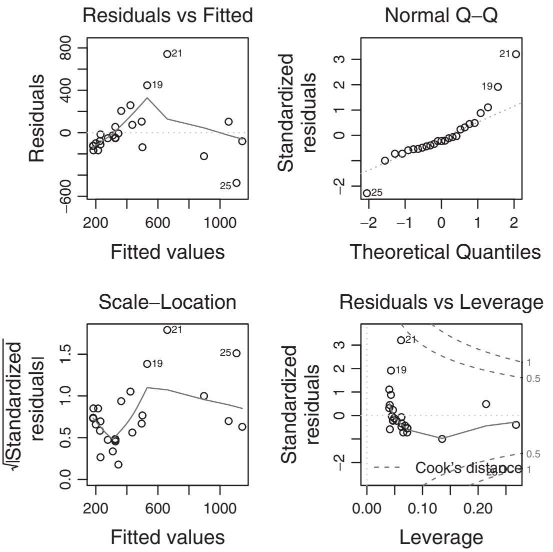


Conclusions - scatterplot of raw data (left figure) indicates evidence of non-normality (boxplots not symmetrical) and evidence that homogeneity of variance my also be violated (points become more spread along the line of the regression line). Data transformed to logarithms (base 10) appear to meet the assumptions of normality and homogeneity of variance better (right figure). Linearity of the log-log relationship also appears reasonable.

Step 3 (Key 8.5a) - the ordinary least squares method is considered appropriate as the main focus will be on hypothesis testing and generating a predictive model.

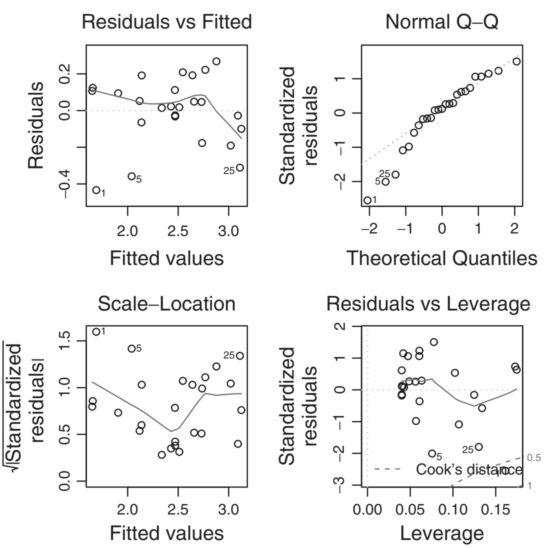
Step 4 (Key 8.6) - fit the simple linear regression model ($y_i = \beta_0 + \beta_1 x_i$) and examine the diagnostics.

```
> peake.lm <- lm(INDIV ~ AREA, data = peake)
> plot(peake.lm)
```



Conclusions - There is a definite "wedge" pattern evident in the residual plot which is indicative of a problem with homogeneity of variance. The Q-Q normal plot confirms that the response variable does deviate from normal. One of the points (observation 25, obscured by the legend) is close to the higher Cook's D contours suggesting that this observation may be overly influential on the final fitted model.

```
> peake.lm <- lm(log10(INDIV) ~ log10(AREA), data = peake)
> plot(peake.lm)
```



Conclusions - The residual plot resulting from a model based on log transformed data does not depict an obvious "wedge", the Q-Q normal plot indicates a greater degree of normality and non of the points are close to the higher Cook's D contours. This confirms that it is more appropriate to fit the linear model using the log transformed data.

```
> influence.measures(peake.lm)
Influence measures of
         lm(formula = log10(INDIV) ~ log10(AREA), data = peake) :

      dfb.1_ dfb.l10.   dffit cov.r   cook.d    hat inf
1  -1.202012  1.12137 -1.2929 0.670 0.626553 0.1615   *
2   0.310855 -0.29097  0.3319 1.260 0.056245 0.1727
3   0.269684 -0.25255  0.2877 1.278 0.042502 0.1745   *
4   0.153477 -0.13896  0.1781 1.187 0.016366 0.1023
5  -0.484207  0.42414 -0.6182 0.804 0.164749 0.0756
```

```
6  -0.062392  0.05251 -0.0897 1.151 0.004183 0.0608
7   0.052830 -0.04487  0.0739 1.158 0.002846 0.0633
8   0.187514 -0.15760  0.2707 1.052 0.036423 0.0605
9   0.006384 -0.00416  0.0164 1.141 0.000140 0.0428
10  0.004787 -0.00131  0.0244 1.137 0.000311 0.0401
11  0.013583  0.00419  0.1238 1.101 0.007882 0.0400
12 -0.003011 -0.00112 -0.0287 1.137 0.000432 0.0401
13  0.000247  0.00259  0.0198 1.138 0.000204 0.0407
14 -0.003734 -0.00138 -0.0356 1.135 0.000662 0.0401
15 -0.015811  0.05024  0.2419 1.013 0.028826 0.0418
16 -0.017200  0.02518  0.0595 1.142 0.001842 0.0487
17 -0.061445  0.09368  0.2375 1.038 0.028033 0.0474
18 -0.025317  0.03314  0.0619 1.151 0.001995 0.0561
19 -0.146377  0.18521  0.3173 1.015 0.049144 0.0607
20  0.100361 -0.13065 -0.2406 1.064 0.028981 0.0567
21 -0.263549  0.31302  0.4496 0.963 0.095261 0.0776
22  0.263206 -0.29948 -0.3786 1.101 0.071044 0.1069
23  0.043182 -0.04845 -0.0588 1.246 0.001804 0.1248
24  0.167829 -0.18726 -0.2236 1.226 0.025747 0.1341
25  0.545842 -0.61039 -0.7334 0.929 0.241660 0.1302
```

Conclusions - Whilst three leverage (hat) values are greater than $2 * p/n = 0.16$ (observations 1, 2 and 3) and therefore potentially outliers in x-space, none of the Cook's D values are ≥ 1 (no point is overly influential). No evidence that hypothesis tests will be unreliable.

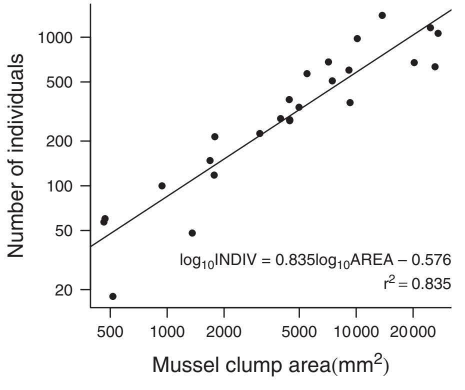
Step 5 (Key 8.6a) - examine the parameter estimates and hypothesis tests.

```
> summary(peake.lm)

Call:
lm(formula = log10(INDIV) ~ log10(AREA), data = peake)

Residuals:
     Min       1Q   Median       3Q      Max
-0.43355 -0.06464  0.02219  0.11178  0.26818

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.57601    0.25904  -2.224   0.0363 *
log10(AREA)  0.83492    0.07066  11.816 3.01e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1856 on 23 degrees of freedom
Multiple R-squared: 0.8586,     Adjusted R-squared: 0.8524
F-statistic: 139.6 on 1 and 23 DF,  p-value: 3.007e-11
```

Conclusions - Reject H_0 that the population slope equals zero. An increase in (log) mussel clump area was found to be associated with a strong ($r^2 = 0.859$), significant increase in the (log) number of supported invertebrate individuals ($b = 0.835, t_{23} = 11.816, P < 0.001$).

Step 6 (Key 8.12) - summarize the findings of the linear regression analysis with a scatterplot including the regression line, regression equation and r^2.

```
> #create a plot with solid dots (pch=16) and no axis or labels}
> plot(INDIV~AREA, data=peake, pch=16, axes=F, xlab="", ylab="",
      log="xy")
> #put the x-axis (axis 1) with smaller label font size
> axis(1, cex.axis=.8)
> #put the x-axis label 3 lines down from the axis
> mtext(text=expression(paste("Mussel clump area", (mm^2))),
       side=1, line=3)
> #put the y-axis (axis 2) with horizontal tick labels
> axis(2, las=1)
> #put the y-axis label 3 lines to the left of the axis
> mtext(text="Number of individuals", side=2, line=3)
> #add the regression line from the fitted model
> abline(peake.lm)
> #add the regression formula
> text(30000, 30, expression(paste(log[10], "INDIV = 0.835",
+ log[10], "AREA - 0.576")), pos=2)
> #add the r squared value
> text(30000, 22, expression(paste(r^2==0.835)), pos=2)
> #put an L-shaped box to complete the axis
> box(bty="l")
```



Step 7 (Key 8.11) - use the fitted linear model to predict the number of individuals that would be supported on two new mussel clumps with areas of 8000 and 10000 mm^2.

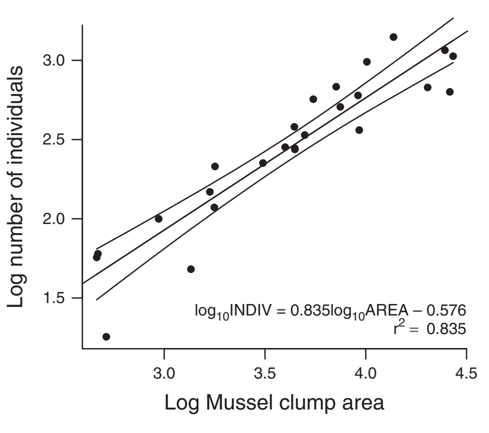
```
> 10^predict(peake.lm, data.frame(AREA = c(8000, 10000)))
       1        2
481.6561 580.2949
```

Since OLS was used to generate the predictive model, and yet there was likely to have been uncertainty in the original mussel clump area measurements, confidence intervals about these predictions are not valid. Nevertheless, the following illustrates how they would be obtained.

```
> 10^predict(peake.lm, data.frame(AREA = c(8000, 10000)),
            interval = "prediction")
       fit      lwr      upr
1 481.6561 194.5975 1192.167
2 580.2949 233.5345 1441.938
```

Similarly, confidence bands could be incorporated onto the plot to indicate confidence in the population regression line if there was no uncertainty in the predictor variable.

```
> plot(log10(INDIV) ~ log10(AREA), data = peake, pch = 16,
+     axes = F, xlab = "", ylab = "")
> axis(1, cex.axis = 0.8)
> mtext(text = "Log Mussel clump area", side = 1, line = 3)
> axis(2, las = 1)
> mtext(text = "Log number of individuals", side = 2, line = 3)
> abline(peake.lm)
> text(4.5, 1.4, expression(paste(log[10], "INDIV = 0.835",
+     log[10], "AREA - 0.576")), pos = 2)
> text(4.5, 1.3, expression(paste(r^2 == 0.835)), pos = 2)
> x <- seq(min(peake$AREA), max(peake$AREA), l = 1000)
> y <- predict(peake.lm, data.frame(AREA = x), interval = "c")
> matlines(log10(x), y, lty = 1, col = 1)
> box(bty = "l")
```



Example 8E: Linear regression - with multiple values of Y per value of X

Sokal and Rohlf (1997) presented data on the (arcsine transformed) percentage survival to adulthood of *Tibolium castaneum* beetles housed at four densities (5, 20, 50 & 100 eggs per gram of flour medium). Each level of the density treatment was replicated (albeit to varying degrees) in a manner similar to single factor classification (ANOVA, see chapter 10).

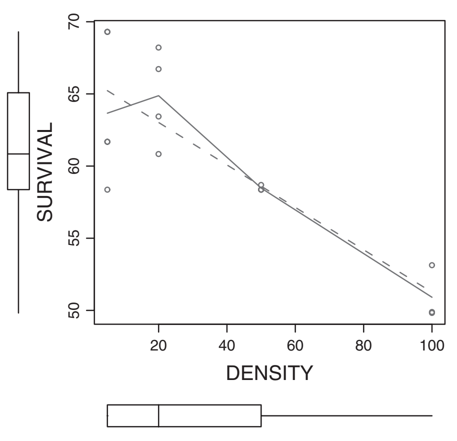
Step 1 - Import (section 2.3) the beetles data set

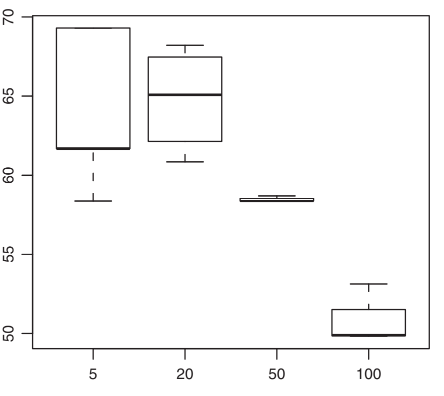
```
> beetles <- read.table("beetles.csv", header = T, sep = ",")
```

Step 2 (Key 8.4) - Assess linearity, normality and homogeneity of variance using a scatterplot with marginal boxplots and a lowess smoother. As there are replicates for each level of the predictor, normality and homogeneity of variance can also be assessed with boxplots of each population.

```
> library(car)
> scatterplot(SURVIVAL ~ DENSITY,
+     data = beetles)
```

```
> boxplot(SURVIVAL ~ DENSITY,
+      data = beetles)
```





Conclusions - the scatterplot indicates that the assumption of linearity is likely to be ok. Note that the boxplot on the x-margin of the scatterplot only reflects an imbalance in replication. Whilst there is some evidence of non-homogeniety of variance, a consistent relationship between mean and variance cannot be fully established, and thus the data are considered suitable.

Step 3 (Key 8.5a) - the ordinary least squares method is considered appropriate as the there is considered to be no uncertainty (error) in the predictor variable (relative density).

Step 4 (Key 8.5b) - determine the lack of fit to the regression line by comparing deviations of observations from the regression line to deviations of observations from their means per density.

```
> anova(lm(SURVIVAL ~ DENSITY + as.factor(DENSITY), beetles))
Analysis of Variance Table

Response: SURVIVAL
                   Df Sum Sq Mean Sq F value    Pr(>F)
DENSITY             1 403.93  403.93 32.0377 0.0001466 ***
as.factor(DENSITY)  2  19.77    9.89  0.7842 0.4804305
Residuals          11 138.69   12.61
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - deviations from linear not significantly different from zero ($F = 0.7842, P = 0.480$), hence there is no evidence that a straight line is not an adequate representation of these data.

Step 5 (Key 8.5b) - consider whether to pool deviations from the regression line and the deviations from the predictor level means

```
> #calculate critical F for alpha=0.25, df=2,11
> qf(0.25,2,11, lower=T)
[1] 0.2953387
```

Conclusions - Sokal and Rohlf (1997) suggest that while there is no difference between the deviations from the regression line and the deviations from the predictor level means, they should not be pooled because $F = 0.784 > F_{0.75[2,11]} = 0.295$.

Step 6 (Key 8.5b) - to test whether the regression is linear by comparing the fit of the linear regression with the deviations from linearity (non pooled).

```
> beetles.lm <- aov(SURVIVAL ~ DENSITY + Error(as.factor(DENSITY)),
+     beetles)
> summary(beetles.lm)
Error: as.factor(DENSITY)
          Df Sum Sq Mean Sq F value  Pr(>F)
DENSITY    1 403.93  403.93  40.855 0.02361 *
Residuals  2  19.77    9.89
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
          Df  Sum Sq Mean Sq F value Pr(>F)
Residuals 11 138.687  12.608
```

Conclusions - Reject H_0 that the population is not linear.

```
> #to get the regression coefficients
> lm(SURVIVAL~DENSITY, beetles)
Call:
lm(formula = SURVIVAL ~ DENSITY, data = beetles)

Coefficients:
(Intercept)      DENSITY
     65.960       -0.147
```

If we had decided to pool, the analysis could have been performed as follows:

```
> summary(lm(SURVIVAL ~ DENSITY, beetles))
Call:
lm(formula = SURVIVAL ~ DENSITY, data = beetles)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-6.8550 -1.8094 -0.2395  2.7856  5.1902

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 65.96004    1.30593  50.508 2.63e-16 ***
DENSITY     -0.14701    0.02554  -5.757 6.64e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.491 on 13 degrees of freedom
Multiple R-squared: 0.7182,     Adjusted R-squared: 0.6966
F-statistic: 33.14 on 1 and 13 DF,  p-value: 6.637e-05
```

Note that these data could also have been analysed as a single factor ANOVA with polynomial contrasts

```
> beetles$DENSITY <- as.factor(beetles$DENSITY)
> contrasts(beetles$DENSITY) <- contr.poly(4, c(5, 20, 50,
+     100))
> beetles.aov <- aov(SURVIVAL ~ DENSITY, beetles)
> summary(beetles.aov, split = list(DENSITY = list(1, c(2,
+     3))))
              Df Sum Sq Mean Sq F value    Pr(>F)
DENSITY         3 423.70  141.23 11.2020 0.0011367 **
  DENSITY: C1   1 403.93  403.93 32.0377 0.0001466 ***
  DENSITY: C2   2  19.77    9.89  0.7842 0.4804305
Residuals      11 138.69   12.61
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Example 8F: Model II regression

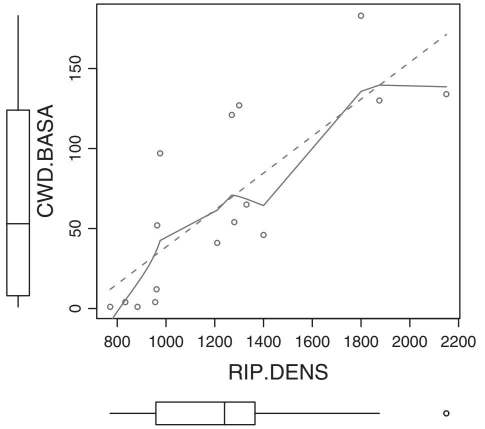
To contrast the parameter estimates resulting from model II regression, Quinn and Keough (2002) used a data set from Christensen et al. (1996) (Box 5.7 Quinn and Keough (2002)). Whilst model II regression is arguably unnecessary for these data (as it is hard to imagine why estimates of the regression parameters would be the sole interest of the Christensen et al. (1996) investigation), we will proceed with the aim of gaining a reliable estimate of the population slope is required.

Step 1 - Import (section 2.3) the Christensen et al. (1996) data set

```
> christ <- read.table("christ.csv", header = T, sep = ",")
```

Step 2 (Key 8.4) - Assess linearity, normality and homogeneity of variance using a scatterplot with marginal boxplots and a lowess smoother.

```
> library(car)
> scatterplot(CWD.BASA ~ RIP.DENS, data = christ)
```



Conclusions - no evidence of non-normality (boxplots not overly asymmetrical), non homogeneity of variance (points do not become progressively more or less spread out along the regression line) or non-linearity.

Step 3 (Key 8.5b) - as there is likely to be uncertainty in the measured levels of the predictor variable and the stated intention of the analysis is to obtain a reliable estimate of the population slope, model II regression is considered appropriate. Furthermore, as the basal area of course woody debris and the density of riparian vegetation are measured on different scales, the degrees of uncertainty in the variables are unlikely to be equal (yet may well be proportionaly to the respective variances of each variable), MA regression is not appropriate. Finally, as there is some evidence that there may be outliers present, RMA is considered the most appropriate method.

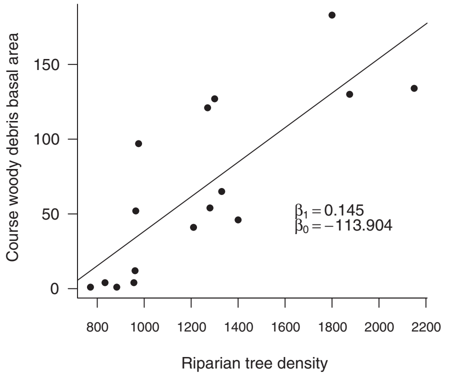
Step 4 (Key 8.5b) - fit the RMA linear regression model.

```
> library(biology)
> christ.lm <- lm.II(CWD.BASA ~ RIP.DENS, christ, type = "RMA")
> summary(christ.lm)
$Call
lm.II(formula = CWD.BASA ~ RIP.DENS, data = christ, type = "RMA")

$Coefficients
                Estimate Lower 95% CI Upper 95% CI
(Intercept) -113.9042556 -187.1524427  -61.7666149
RIP.DENS       0.1450207    0.1032249    0.2037396
```

Step 5 (Key 8.12) - summarize the findings of the linear regression analysis with a scatterplot including the regression line, regression equation and r^2.

```
> #create a plot with solid dots (pch=16) and no axis or labels
> plot(CWD.BASA~RIP.DENS, christ, pch=16, axes=F, xlab="",
      ylab="")
> #put the x-axis (axis 1) with smaller label font size
```

```
> axis(1, cex.axis=.8)
> #put the x-axis label 3 lines down from the axis
> mtext(text="Riparian tree density", side=1, line=3)
> #put the y-axis (axis 2) with horizontal tick labels
> axis(2, las=1)
> #put the y-axis label 3 lines to the left of the axis
> mtext(text="Course woody debris basal area", side=2, line=3)
> #add the regression line from the fitted model
> abline(christ.lm)
> #add the regression parameters
> text(1600,50,expression(paste(beta[1]==0.145)), pos=4)
> text(1600,40,expression(paste(beta[0]==-113.904)), pos=4)
> #put an L-shaped box to complete the axis
> box(bty="l")
```



Example 8G: Linear regression - non-parametric regression

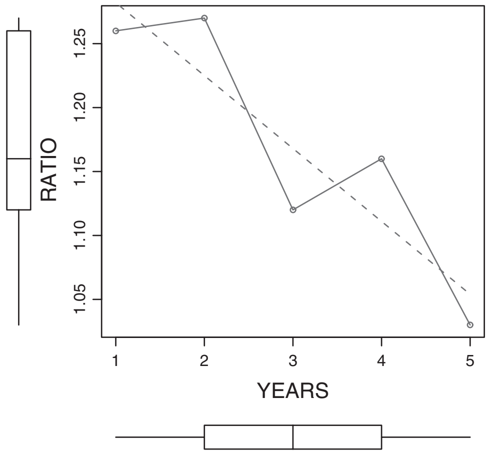
Smith (1967) investigated the effects of cloud seeding on rainfall in the Snowy Mountains, Australia. The experiment took place in two areas (the target and control). Within a year a number of periods were randomly allocated for seeding and additional periods for non-seeding. The total rainfall in the target and control areas during each of these periods were recorded. Within a single year, the impact of seeding was assessed via a double ratio (ratio of rainfall in target to control areas for seeding periods versus ratio of target to control areas during non-seeding times) and the experiment was repeated over 5 years (Example 9.2 Hollander and Wolfe (1999)).

Step 1 - Import (section 2.3) the Smith (1967) data set

```
> smith <- read.table("smith.csv", header = T, sep = ",")
```

Step 2 (Key 8.4) - Assess linearity, normality and homogeneity of variance using a scatterplot with marginal boxplots and a lowess smoother.

```
> scatterplot(RATIO ~ YEARS, smith)
```



Conclusions - whilst there may not appear to be any evidence of non-normality (boxplots not overly asymmetrical), non homogeneity of variance (points do not become progressively more or less spread out along the regression line) or non-linearity, it could be argued that there are too few observations on which to make meaningful decisions about normality and it might be safer to not make distributional assumptions.

Step 3 (Key 8.7) - as far as we know, there are no reasons to suspect that that observations wont be independent.

Step 4 (Key 8.8b) - it is difficult to assess normality, homogeneity of variance and linearity with such a small sample size. We will take the conservative approach and not make any such assumptions.

Step 5 (Key 8.9d) - perform non-parametric (Kendall's) robust regression to assess the $H_0 : \beta_1 = 0$.

```
> library(mblm)
> smith.mblm <- mblm(RATIO ~ YEARS, smith, repeated = F)
> summary(smith.mblm)
Call:
mblm(formula = RATIO ~ YEARS, dataframe = smith, repeated = F)

Residuals:
       1        2        3        4        5
 0.00000  0.06625 -0.02750  0.06875 -0.00500

Coefficients:
            Estimate      MAD V value Pr(>|V|)
(Intercept)  1.31625  0.04077      15   0.0625 .
YEARS       -0.05625  0.03459       4   0.0137 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.05744 on 3 degrees of freedom
```

Conclusions - reject H_0. The impact of cloud seeding significantly declines over time (b=-0.056, V=4, P=0.0137).

Step 6 (Key 8.10) - calculate 95% confidence intervals for the parameter estimates.

```
> confint.mblm(smith.mblm, level = 0.95)
                 0.025   0.975
(Intercept)  1.28875  1.385
YEARS       -0.10000 -0.015
```

Example 8H: Linear regression - randomization test

McKechnie et al. (1975) investigated the relationship between altitude and the frequency of hezokinase (HK) 1.00 mobility genes from colonies of *Euphydras editha* butterflies (Example 8.1 Manly (1991)).

Step 1 - Import (section 2.3) the McKechnie et al. (1975) data set

```
> mckechnie <- read.table("mckechnie.csv", header = T, sep = ",")
```

Step 2 (Key 8.4) - Assess linearity, normality and homogeneity of variance using a scatterplot with marginal boxplots and a lowess smoother. For the purpose of this demonstration, lets assume that the assumption of normality could not be met and more importantly, that the observations are not independent, thereby necessitating an alternative regression method.

Step 3 (Key 8.7b) - use randomization to test whether the observed trend could be due to chance.

1. define the statistic[j] to use in the randomization test - in this case the *t*-statistic

```
> stat <- function(data, index) {
+     summary(lm(HK ~ ALT, data))$coef[2, 3]
+ }
```

2. define how the data should be randomized - randomize the pairing of predictor and responses (shuffle without replacement the predictor values amongst observations)

```
> rand.gen <- function(data, mle) {
+     out <- data
+     out$ALT <- sample(out$ALT, replace = F)
+     out
+ }
```

3. call a bootstrapping procedure to randomize 5000 times (this can take some time)

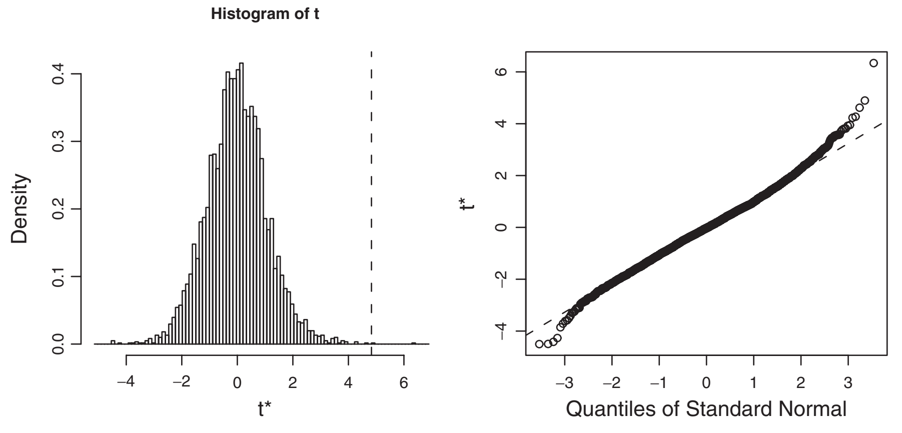
```
> library(boot)
```

[j] Consistent with Manly (1991), I have used OLS to estimate the regression parameters. However, these parameters could alternatively be RMA or non-parametric regression estimates.

```
> mckechnie.boot <- boot(mckechnie, stat, R = 5000,
+     sim = "parametric", ran.gen = rand.gen)
```

4. examine the distribution of *t*-values generated from the randomization procedure

```
> plot(mckechnie.boot)
```



5. examine the bootstrap statistics

```
> mckechnie.boot
PARAMETRIC BOOTSTRAP


Call:
boot(data = mckechnie, statistic = stat, R = 5000,
    sim = "parametric", ran.gen = rand.gen)


Bootstrap Statistics :
    original    bias    std. error
t1* 4.830571 -4.846745    1.084864
```

6. calculate the number of possible *t*-values (including the observed *t*-value, which is one possible outcome) that were greater or equal to the observed *t*-value and express this as a percentage of the number of randomizations (plus one for the observed outcome).

```
> t <- length(mckechnie.boot$t[mckechnie.boot$t >=
+      mckechnie.boot$t0]) + 1
> t/(mckechnie.boot$R + 1)
[1] 0.00059988
```

Conclusions - probability of obtaining a *t*-value of 4.83 or greater when H_0 is true is 0.0006 (0.06%). Note that as this is a randomization procedure, the p-value will vary slightly each time.

Step 4 (Key 8.10) - calculate 95% confidence intervals for the parameter estimates (example 8.2 Manly (1991))

1. define how the parameters (coefficients) are to be calculated (from OLS regression of a random resample with replacement of the observations).

```
> par.boot <- function(mckechnie, index) {
+     x <- mckechnie$ALT[index]

+     y <- mckechnie$HK[index]
+     model <- lm(y ~ x)
+     coef(model)
+ }
```

2. call a bootstrapping procedure to randomize 5000 times (this can take some time)

```
> mckechnie.boot <- boot(mckechnie, par.boot, R = 5000)

> mckechnie.boot
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = mckechnie, statistic = par.boot, R = 5000)


Bootstrap Statistics :
    original     bias    std. error
t1* 10.65409  0.2426368    4.853195
t2* 29.15347 -0.1309074    5.581786
```

3. examine the bootstrap 95% confidence intervals for the second (index=2) parameter (slope)

```
> boot.ci(mckechnie.boot, index = 2)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = mckechnie.boot, index = 2)

Intervals :
Level      Normal              Basic
95%   (18.34, 40.22 )   (18.38, 40.81 )

Level     Percentile            BCa
95%   (17.50, 39.92 )   (16.95, 39.52 )
Calculations and Intervals on Original Scale
```

Conclusions - 95% confidence interval for the true regression coefficients is 15.49 - 39.52

Step 5 (Key 8.11) - predict the percentage of HK genes at an altitude of 1.

1. define the function to predict new values.

```
> pred.boot <- function(mckechnie, index) {
+     mckechnie.rs <- mckechnie[index, ]
+     mckechnie.lm <- lm(HK ~ ALT, mckechnie.rs)
+     predict(mckechnie.lm, data.frame(ALT = 1))
+ }
```

2. call a bootstrapping procedure to randomize 5000 times (this can take some time)

```
> mckechnie.boot <- boot(mckechnie, pred.boot, R = 5000)

> mckechnie.boot
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = mckechnie, statistic = pred.boot, R = 5000)


Bootstrap Statistics :
    original    bias    std. error
t1* 39.80756 0.1235158    4.914043
```

3. examine the bootstrap 95% intervals for this prediction

```
> boot.ci(mckechnie.boot, index = 1)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = mckechnie.boot, index = 1)

Intervals :
Level      Normal              Basic
95%   (30.05, 49.32 )   (30.66, 49.80 )

Level     Percentile            BCa
95%   (29.82, 48.96 )   (27.68, 47.58 )
Calculations and Intervals on Original Scale
```

Conclusions - 95% confidence interval for the true regression coefficients is 27.59 - 47.81

Alternatively, if the levels of the predictor variable were specifically set, then it might be more appropriate to base hypothesis tests, predictions and confidence intervals on randomized residuals rather than randomizing the predictor variable.

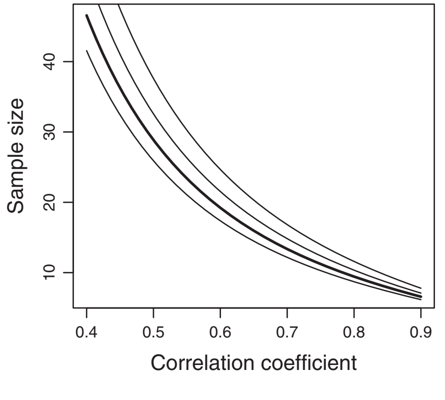
Example 8I: Power analysis - sample size determination in testing $H_0 : \rho = 0$
Zar (1999) provided a worked example in which the sample size required to reject the null hypothesis ($H_0 : \rho = 0$) 99% of the time when the correlation coefficient has an absolute magnitude (ignore sign) greater or equal to 0.5 ($|\rho| \geq 0.5$) (Example 19.5 Zar (1999)).

Step 1 - calculate the sample size required to detect a correlation of greater or equal to 0.5 with a power of 0.99

```
> library(pwr)
> pwr.r.test(r = 0.5, power = 0.99)
     approximate correlation power calculation (arctangh
     transformation)

              n = 63.50301
              r = 0.5
      sig.level = 0.05
          power = 0.99
    alternative = two.sided
```

Step 2 - generate a plot that illustrates the relationship between target correlation (from 0.4 to 0.9) and sample size for a range of levels of power (0.75,0.8,0.85,0.9).

```
> library(pwr)
> r <- seq(0.4, 0.9, l = 100)
> plot(sapply(r, function(x) pwr.r.test(r = x, power = 0.8)$n) ~
+     r, type = "l", lwd = 2, xlab = "Correlation coefficient",
+     ylab = "Sample size")
> points(sapply(r, function(x) pwr.r.test(r = x, power = 0.9)$n) ~
+     r, type = "l")
> points(sapply(r, function(x) pwr.r.test(r = x, power = 0.85)$n) ~
+     r, type = "l")
> points(sapply(r, function(x) pwr.r.test(r = x, power = 0.75)$n) ~
+     r, type = "l")
```



Conclusions - graph provides a means to evaluate the cost-benefit compromises between power and sample size for a range of possible correlations. Informed design decisions can result from such graphs. If the degree of correlation is expected to be high, approximately 10 replicates would be adequate. However, if the degree of correlation is expected to be lower, a greater number of replicates are required. Furthermore, as the degree of correlation declines, the difference in estimated required sample size for different levels of power becomes greater.

9

Multiple and curvilinear regression

Multiple and complex regression analyses can be useful for situations in which patterns in a response variable can not be adequately described by a single straight line resulting from a single predictor and/or a simple linear equation.

9.1 Multiple linear regression

Multiple regression is an extension of simple linear regression whereby a response variable is modeled against a linear combination of two or more simultaneously measured continuous predictor variables. There are two main purposes of multiple linear regression:

(i) To develop a better predictive model (equation) than is possible from models based on single independent variables.
(ii) To investigate the relative individual effects of each of the multiple independent variables above and beyond (standardized across) the effects of the other variables.

Although the relationship between response variable and the additive effect of all the predictor variables is represented overall by a single multidimensional plane (surface), the individual effects of each of the predictor variables on the response variable (standardized across the other variables) can be depicted by single *partial regression* lines. The slope of any single partial regression line (*partial regression slope*) thereby represents the rate of change or effect of that specific predictor variable (holding all the other predictor variables constant to their respective mean values) on the response variable. In essence, it is the effect of one predictor variable at one specific level (the means) of all the other predictor variables (i.e. when each of the other predictors are set to their averages).

Multiple regression models can be constructed additively (containing only the predictor variables themselves) or in a multiplicative design (which incorporate interactions between predictor variables in addition to the predictor variables themselves). Multiplicative models are used primarily for testing inferences about the effects of various predictor variables and their interactions on the response variable in much the same way as factorial ANOVA (see chapter 12). Additive models by contrast are used for generating predictive models and estimating the relative importance of individual predictor variables more so than hypothesis testing.

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

9.2 Linear models

Additive model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_j x_{ij} + \varepsilon_i$$

where β_0 is the population y-intercept (value of y when all partial slopes equal zero), β_1, β_2, etc are the partial population slopes of Y on X_1, X_2, etc respectively holding the other X constant. ε_i is the random unexplained error or residual component. The additive model assumes that the effect of one predictor variable (partial slope) is independent of the levels of the other predictor variables.

Multiplicative model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i1} x_{i2} + \ldots + \varepsilon_i$$

where $\beta_3 x_{i1} x_{i2}$ is the interactive effect of X_1 and X_2 on Y and it examines the degree to which the effect of one of the predictor variables depends on the levels of the other predictor variable(s).

9.3 Null hypotheses

A separate H_0 is tested for each of the estimated model parameters:

H_0: $\beta_0 = 0$ (the population y-intercept equals zero)

This test is rarely of interest as it only tests the likelihood that the background level of the response variable is equal to zero (rarely a biologically meaningful comparison) and does not test whether or not there is a relationship.

H_0: $\beta_1 = 0$ (the partial population slope of X_1 on Y equals zero)
H_0: $\beta_2 = 0$ (the partial population slope of X_2 on Y equals zero)
....

These tests examine respectively whether or not there is likely to be a relationship between the dependent and one of the independent variables (holding the other independent variables constant) in the population.

For an additive model

$H_0 : \beta_3 = 0$ (the partial population slope of the interactive effect of X_1 and X_2 on Y equals zero)

This test examines whether or not the effect of one dependent variable on the independent variable (holding others constant) is dependent on other independent variables.

As with simple linear regression, these individual parameter null hypothesis tests can all be tested using the t-statistic with $n - (p + 1)$ degrees of freedom (where p is the number of parameters in the linear model) or by comparing the lack of fit of a *full model* (model containing all predictor variables) to an appropriate *reduced model* (model containing all but the individual predictor variable or interacting variables) via analysis of variance. In addition, the overall analysis of variance (which tests the $H_0 : \beta_1 = \beta_2 = \ldots = \beta_j = 0$) investigates whether the response variable can be modeled by the particular linear combination of predictor variables.

Interactions

The nature of significant interactions (e.g. X_1 and X_2 on Y) can be further explored by re-fitting the multiple linear model to explore the partial effects of one of the predictor variables (e.g. X_1) for a specific set of levels of the other interacting predictor variable(s) (e.g. the mean of x_2 as well as this mean $\pm$ 1 and or 2 standard deviations). For such subsequent main effects tests, ignore the effect of the interaction, which will be identical to that previously tested, and focus purely on the individual partial slope (β_1).

9.4 Assumptions

To maximize the reliability of hypothesis tests, the following assumptions apply:

(i) linearity - no other curved relationship represents the relationships between each of the predictors and the response variable. Scatterplots and scatterplot matrices are useful for exploring linearity.

(ii) normality - the residuals, and therefore the populations from which each of the responses were collected, are normally distributed. Note that in the majority of multiple linear regression cases, the predictor variables are measured (not specifically set), and therefore the respective populations are also assumed to be normally distributed. Boxplots of each variable (particularly those incorporated within the diagonals of a scatterplot matrix) are useful diagnostics.

(iii) homogeneity of variance - the residuals (populations from which each of the responses were collected) are equally varied. Exploring the spread of points around individual scatterplot trendlines can be useful, as can residual plots. Plots of residuals against each of the predictor variables can also be useful for diagnostic spatial and temporal autocorrelation.

(iv) (multi)collinearity - a predictor variable must not be correlated to the combination of other predictor variables. Multicollinearity has major detrimental effects on model fitting:
- instability of the estimated partial regression slopes (small changes in the data or variable inclusion can cause dramatic changes in parameter estimates).
- inflated standard errors and confidence intervals of model parameters, thereby increasing the type II error rate (reducing power) of parameter hypothesis tests.

Multicollinearity can be diagnosed with the following:
- investigate pairwise correlations between all the predictor variables either by a correlation matrix or a scatterplot matrix.

- calculate **tolerance** ($1 - r^2$ of the relationship between a predictor variable and all the other predictor variables) for each of the predictor variables. Tolerance is a measure of the degree of collinearity and values less < 0.2 should be considered and values < 0.1 given series attention. Variance inflation factor (VIF) are the inverse of tolerance and thus values greater than 5, or worse, 10 indicate collinearity.
- PCA (principle components analysis) eigenvalues (from a correlation matrix for all the predictor variables) close to zero indicate collinearity and component loadings may be useful in determining which predictor variables cause collinearity.

There are several approaches to dealing with collinearity[a]:

- remove the highly correlated predictor variable(s), starting with the least most biologically interesting variable(s).
- PCA (principle components analysis) regression - regress the response variable against the principal components resulting from a correlation matrix for all the predictor variables. Each of these principal components by definition are completely independent, but the resulting parameter estimates must be back-calculated in order to have any biological meaning.

Interaction terms in multiplicative models are likely to be correlated to their constituent individual predictors, and thus the partial slopes of these individual predictors are likely to be unstable. However, this problem can be reduced by first centering (subtracting the mean from the predictor values) the individual predictor variables.

(v) the number of predictor variables must be less than the number of observations otherwise the linear model will be over-parameterized (more parameters to estimate than there are independent data from which estimations are calculated).

As with simple linear regression, regression diagnostics (residuals, leverage and Cook's D) should be examined following model fitting.

9.5 Curvilinear models

It is not always appropriate to attempt to model the relationship between a response and predictor variable with a straight line in which it is assumed that the rate of change (slope) remains constant throughout the range of the predictor variable. In such cases, scale transformations may not only be unable to correct linearity, they may be inappropriate when we are trying to describe a model that reflects the true nature of the relationship. To some degree, curvilinear models assume that there is a relationship between the variables and are themselves more concerned with exploring the nature of the relationship. Table 9.1 depicts the general nature and corresponding models and R syntax for some simple or useful non-linear models.

9.5.1 Polynomial regression

Polynomials are linear combinations of predictor variables (no predictor variable is the exponent, multiplier or deviser of any other) in which a predictor variable is represented

[a] Note that all of these are likely to result in biased parameter estimates.

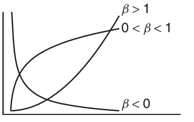
Table 9.1 Illustrative set of useful non-linear functions with corresponding R model fitting syntax. Some examples also illustrate corresponding self-starting functions. Note that this is a non-exhaustive set.

Function	Preview

Concave/convex functions

Power ($y = \alpha x^{\beta}$)

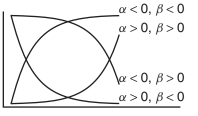
Used to describe a large range of physical and biological trends including allometric scaling relationships (e.g. Kleiber's law) and inverse square laws (e.g. Newtonian gravity). α defines the scale of the y-axis and β defines the magnitude and polatity of the rate of change and thus the degree of curvature




```
> nls(DV~a*IV^b, dataset, start=list(a=1,
    b=0.1))
```

Exponential ($y = \alpha e^{\beta x}$)

Models non-asymptotic growth and decay. α defines the scale of the y-axis and increasing magnitude of β increases the curvature of the curve.

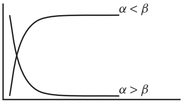



```
> nls(DV~a*exp(b*IV), dataset, start=list(a=1,
    b=0.1))
```

Aymptotic functions

Asymptotic exponential ($y = \alpha + (\beta - \alpha)e^{-e^{\gamma}x}$)

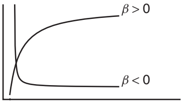
Used to describe general asymptotic relationships.
Equivelent to the more simple $y = a - be^{-cx}$ when $a = \alpha$, $b = \beta - \alpha$ and $c = e^{\gamma}$
α - y value of horizontal asymptote. β - value of y when $x = 0$.
γ - natural log of rate of curvature




```
> nls(DV~a+b*exp(c*x),dataset,start=list(a=1,
    b=-1,c=-1))
> nls(DV~SSasymp(IV,a,b,c), dataset)
```

Michaelis-Menten ($y = \frac{\alpha x}{\beta + x}$)

Used to relate rates of enzymatic reactions to substrate concentrations
$\alpha - y$ value of horizontal asymptote. β (*Mechaelis parameter*) - value of x at which half the asymptotic response is obtained.




```
> nls(DV~(a*IV)/(b*IV), dataset,
    start=list(a=1, b=1))
> nls(DV~SSmicmen(IV,a,b), dataset)
```

Table 9.1 (*continued*)

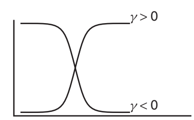
Function	Preview

Sigmoidal

Logistic ($y = \frac{\alpha}{1+e^{(\beta-x)/\gamma}}$)

Used to describe binary responses (presence/absence, alive/dead, etc) relationships.
α - horizontal asymptote (typically 1). β - value of x at which half the asymptotic response is obtained (inflection point).
γ - determines the steepness at inflection.

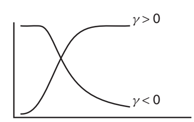
```
> nls(DV~a/(1+exp((b-IV)/c)), dataset,
    start=list(a=1,b=1,c=.1))
> nls(DV~SSlogis(IV,a,b,c), dataset)
```



Weibull ($y = \alpha - \beta e^{-(e^{\gamma} x^{\delta})}$)

Describes the kinetics of many enzymes. Used to relate rates of enzymatic reactions to substrate concentrations
α - right side horizontal asymptote. β - rate of vertical change.
γ - natural log of rate of curvature. δ - power to raise x.

```
> nls(DV~a - b*exp(-exp(c)*IV^d), dataset,
    start=list(a=1, b=1, c=1, d=1))
> nls(DV~SSweibull(IV,a,b,c,d), dataset)
```

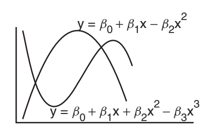


Peaks and/or valleys

Polynomials

Describes the kinetics of many enzymes. Used to relate rates of enzymatic reactions to substrate concentrations

```
> lm(DV~ IV + I(IV^2) + I(IV^3), dataset)
> lm(DV poly(IV, 3), dataset)
```



by multiple instances of itself (each of a successively higher order). These higher order terms are quadratic ($2nd$ order, x^2), cubic ($3rd$ order, x^3), etc terms and are interactions of the predictor variables with itself. The linear model for a second-order (quadratic) regression (parabola) is:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i1}^2 + \varepsilon_i$$

Parameters are estimated and tests of the H_0's that $\beta_0 = 0$, $\beta_1 = 0$, $\beta_2 = 0$ and $\beta_0 = \beta_1 = \beta_2 = 0$ are performed as per multiple linear regression. Note that the polynomial regression model contains multiple instances of a predictor variable (including interactions), and that each of these instances will be correlated to one another, thereby violating the assumption of collinearity. Centering the predictor variable first reduces this problem.

Arguably a more biologically meaningful test is whether a higher-order polynomial model (e.g. quadratic) fits the data better than a lower-order model (such as a simple linear regression) and this is tested with a *F-statistic* by comparing the fit of the model with the higher-order term versus a model without this term.

9.5.2 Nonlinear regression

Non-linear regression models enable us to investigate the fit of various predefined functions (such as power, exponential, logarithmic as well as any other non straight line functions) to our collected data. Non-linear model parameters are estimated by iteratively changing the values of the parameters so as to either minimize the sum of squared residuals (OLS) or to maximize the log-likelihood (ML). Starting values of the parameters must be provided, and should be realistic to maximize the chances of convergence (reaching stable parameter estimates). Furthermore, it is advisable that non-linear models be re-fitted with a range of starting values so as to reduce the risks of parameter estimates converging on a 'local minimum' (a set of parameters arrived on through the sequential iteration process that produce a better fit than slightly different values of the parameters, yet still not the estimates that produce the best fit). When using OLS, the typical regression assumptions of residual normality and equal variance apply, whereas, ML can be more robust to these assumptions.

9.5.3 Diagnostics

The same model fitting diagostic issues and measures that were highlighted in section 8.2.6 are relevant to multiple linear regression and non-linear regression.

9.6 Robust regression

The robust alternatives introduced for simple linear regression in section 8.2.7 can largely be extended to multiple linear regression applications.

9.7 Model selection

Not all the predictor variables in a multiple linear model necessarily contribute substantially to explaining variation in the response variable. Those that do not, are unlikely to have much biological impact on the response and therefore could be ommitted from the final regression equation (along with all the other unmeasured variables). Furthermore, we may wish to determine which of a range of linear and non-linear models best fits the collected data. For the purpose of explaining a response variable[b], the 'best' regression model is arguably the model that contains only a subset combination of important predictor variables and is therefore the model that explains the most amount of response variability with the fewest predictor terms[c] (parsimony).

---

[b] Likewise, for the pursuit of developing predictive multiple regression models, the 'best' regression model will contain the fewest predictor variables as greater numbers of predictor variables increases the model complexity and sources of uncertainty and thus decreases the precision of resulting predictions.
[c] Recall that in statistical models, a 'term' denotes an estimable parameter (such as partial slope) and its associated predictor or interaction of predictors.

There are several criteria that can be used to assess the efficiency or fit of a model that are penalized by the number of predictor terms. These criteria are calculated and compared for a set of competing models thereby providing an objective basis on which to select the 'best' regression model.

***$MS_{residuals}$* -** represents the mean amount of variation unexplained by the model, and therefore the lowest value indicates the best fit.

Adjusted r^2 - (the proportion of mean amount of variation in response variable explained by the model) is calculated as $adj.\ r^2 = \frac{MS_{regression}}{MS_{total}}$ and is therefore adjusted for both sample size and the number of terms. Larger values indicate better fit. Adjusted r^2 and $MS_{residuals}$ should not be used to compare between linear and non-linear models.

Mallow's C_p - is an index resulting from the comparison of the specific model to a model that contain all the possible terms. Models with the lowest value and/or values closest to their respective *p* (the number of model terms, including the y-intercept) indicate best fit.

Akaike Information Criteria (AIC) - there are several different versions of AIC, each of which adds a different constant (designed to penalize according to the number of parameters and sample size) to a likelihood function to produce a relative measure of the information content of a model. Smaller values indicate more parsimonious models. As a rule of thumb, if the difference between two AIC values (delta AIC) is greater than 2, the lower AIC is a significant improvement in parsimony.

Schwarz Bayesian Information Criteria (BIC or SIC) - is outwardly similar to AIC. The constant added to the likelihood function penalizes models with more predictor terms more heavily (and thus select more simple models) than AIC. It is for this reason that BIC is favored by many workers, however, others argue strongly in favor of AIC claiming that the theoretical basis for BIC may not be relevant for most biological applications[d].

Traditionally, the set of competing linear models were generated by stepwise procedures in which terms were progressively added or dropped from a model on the basis of importance (as assessed via p-values of partial slopes). Whilst such procedures reduce the number of models that are assessed and compared (it is for the associated reductions in computational intensity that such procedures where originally developed), it is possible that the 'best' model is never assessed. Modern computing now allows all combinations to be assessed rapidly thereby voiding the need for such selection procedures.

9.7.1 Model averaging

Typically, there are multiple plausible alternative models that incorporate different combinations of predictor variables and that yield similar degrees of fit (based on AIC, QAIC, BIC, etc). Each alternative model will result in different parameter estimates for the predictor variables. Furthermore, conclusions about the relative importance of each of the predictor variables is likely to be dependent on which model is selected. Model averaging is a technique that calculates weighted averages of the parameter estimates

[d] The original basis for BIC was for situations in which there were either no effects or else there were a mixture of major and no effects with no intermediate or tapering effects. Furthermore, it assumes that the true model (against which all others are compared) is among the set being assessed.

Table 9.2 Comparison of the different model selction criteria. Where n is the number of observations, p is the number of model terms (not including the y-intercept), k is the number of model parameters (including the y-intercept) and c is an additive constant. The value of the additive constant differs under different circumstances (and purposes) and therefore `extractAIC()` and `AIC()` methods can give different values (and do for parametric models). *Model* and *Full* refer to the specific model being assessed and the fully populated model respectively.

Selection criteria	**Form**	**R syntax**
$MS_{residuals}$		`> anova(model)["Residuals","Mean Sq"]`
Adjusted r^2	$1 - \frac{SS_{resid}/[n-(p+1)]}{SS_{total}/(n-1)}$	`> summary(model)$adj.r.squared`
Mallow's C_p	$\frac{ModelSS_{resid}}{FullMS_{resid}} - [n-2(p+1)]$	`> library(biology)`
	$n\left[\ln\left(\frac{SS_{resid}}{n}\right)\right] + 2(p+1) - n$	`> Cp(model, full)`
BIC (Bayesian Information Criterion)		
• for parametric models only	$n\left[\ln\left(\frac{SS_{resid}}{n}\right)\right] + \ln(p+1) + c$	`> extractAIC(model, k=log(nrow(dataset))`
• generic for any fitted model	$n\left[\ln\left(\frac{SS_{resid}}{n}\right)\right] + \ln(p+1) + c$	`> AIC(model, k=log(nrow(dataset))`

AIC (Akaike Information Criterion)		
• for parametric models only	$n\left[\ln\left(\frac{SS_{resid}}{n}\right)\right] + 2(p+1) + c$	`> extractAIC(model)`
• generic for any fitted model	$n\left[\ln\left(\frac{SS_{resid}}{n}\right)\right] + 2(p+1) + c$	`> AIC(model)`
AIC_c (second order correction)	$\mathrm{AIC} + \frac{2k(k+1)}{n-k-1}$	`> library(MuMIn)`
corrected for small sample sizes		`> AICc(model)`
QAIC (quasi-AIC)	$n\left[\ln\left(\frac{SS_{resid}}{n}\right)\right] /\hat{c} + 2(p+1) + c$	`> library(MuMIn)`

corrected for overdispersion		`> QAIC(model)`
		OR
		`> library(biology)`
		`> qAICc(model)`

[a] The `MuMIn` *package* is not yet part of the official comprehensive R archive network (CRAN). The package can be downloaded from http://mumin.r-forge.r-project.org/ or installed from within R: `> install.packages("MuMIn", repos="http://R-Forge.R-project.org")`.
[b] Only relevant for models in which overdispersion is likely to be an issue, see section 17.1. For such cases, $\hat{c}$ (the quasi-likelihood parameter), is a measure of the degree of overdispersion and can be estimated by dividing the model deviance by its degrees of freedom

for each predictor variable across all the possible models. In so doing, model selection uncertainty can be incorporated into estimates of parameter precision. Furthermore, through model averaging, we are able to obtain an estimate the relative importance of each of the predictor variables on the the response.

9.7.2 Hierarchical partitioning

For applications that are primarily focused on identifying the polarity and relative magnitudes of the effects (importance) of predictor variables, constructing a single 'best' predictive model may be of little value and indeed may not necessarily identify the important causal variables. Similar to model averaging, hierarchical partitioning assesses the independent, joint and total contribution (relative influence) of each predictor variable by averaging a measure of goodness-of-fit[e] over all possible models that include that predictor variable. In so doing, hierarchical partitioning is also less susceptible to multicollinearity problems than are the single-model approaches outlined above. Note that since hierarchical partitioning operates within an entire model set, it is not appropriate for comparing the fit of single models.

In order to evaluate whether the magnitude of a variable's contribution is great enough to warrant retention (or attributed as important), a randomization procedure can be used in which the independent contributions of each predictor variable are compared to distributions of such contributions generated by repeated (e.g. 1000 times) randomizations of the data matrix. Alternatively, the randomized outcomes can be used to calculate Z-scores[f] for each predictor variable, which in turn can be used to test significance ($Z \geq 1.65$ at the 95% level).

9.8 Regression trees

Regression trees are a robust[g] alternative to multiple regression for exploring and describing patterns between a response variable and multiple predictor variables as well as developing predictive models. In addition, as regression trees are rank-based, they accommodate a range and combination of response and predictor data types (including categorical, numerical and rankings) and do not depend on the nature of monotonic relationships (linearity not assumed nor is the arbitrary family of a curvilinear relationship required).

Regression trees are constructed via *binary recursive partitioning*, a process in which the data are progressively split into a dichotomously branching tree. Initially, for each predictor variable, the process iteratively determines the value of that predictor variable that results in the single dichotomous split that minimizes the sum of squared deviations from the split response means. The predictor variable (and split) with the smallest deviations is thereby installed as a *node* at the top of the tree and is interpreted as the most explanatory of the patterns in the response variable. Two

[e] r^2 in multiple linear regression, χ^2 in log-linear models.
[f] calculated as $Z = (I_{obs} - mean\{I_{rand}\})/sd\{I_{rand}\}$.
[g] They are invariant to underlying distributions.

branches descend from this top tree node. The left and right branches represent subsets of the entire dataset for which the values of the top predictor variable are respectively less than and greater than the splitting threshold value. This partitioning process then continues recursively down each branch until either a specific number of branches have been produced or a pre-defined minimum number of observations within the branch has been obtained. Graphical trees can be constructed to illustrate the hierarchy of importance of the predictor variables as well as the nature of interactions between predictor variables.

Each additional split increases the overall explanatory power of the tree (as measured by total deviance). However, greater numbers of branches also increase the degree of *over-fitting*[h] and complexity resulting in models with poor predictive performance. A *cost-complexity measure* can be used to visually assess the compromise between explanatory power and complexity (number of branches) and thus help identify how the tree could be *pruned*.

9.9 Further reading

- Theory

Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. John Wiley & Sons, New York.

Manly, B. F. J. (1991). *Randomization and Monte Carlo methods in biology*. Chapman & Hall, London.

Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

Faraway, J. J. (2006). *Extending Linear Models with R: generalized linear mixed effects and nonparametric regression models*. Chapman & Hall/CRC.

Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS, 4th edn*. Springer-Verlag, New York.

9.10 Key and analysis sequence for multiple and complex regression

1 a. **Investigating relationships between a single response variable and multiple predictor variables with the expectation that the predictor variables will be linearly related to the response (Multiple linear regression)** .... Go to 2

---

[h] Over-fitting is were additional branches have began to represent and "explain" random aspects of the dataset (such as individual variation) rather than genuine population patterns.

b. Investigating non-linear relationships between a single response variable and a single predictor variable (Non-linear regression) . . . . Go to 7

c. Develop descriptive and predictive models between a single response variable and multiple predictor variables with few distributional, curvilinear or data type restrictions (Regression trees) . . . . Go to 13

2 a. Check assumptions for multiple linear regression

Parametric assumptions

- **Normality of the response variable and predictor variables - scatterplot matrix with boxplots in diagonals**
- **Homogeneity of variance - spread of data around scatterplot matrix trendlines**
- **Linearity of data points on a scatterplot, trendline and lowess smoother useful**

```
> library(car)
> scatterplot.matrix(~DV+IV1+IV2+IV3, dataset,
+ diag="boxplot")
```

where `DV` *and* `IV1`, `IV2`,. . . *are the response and predictor variables respectively in the* `dataset` *data frame*

(Multi)collinearity assumption . . . . Go to 3

Parametric assumptions met . . . . Go to 4

b. Parametric assumptions NOT met or scale transformations (see tab. 3.2) not successful or inappropriate . . . . Go to 7

3 a. Check (multi)collinearity assumption

```
> cor(dataset[, cols])
```

where `cols` *is a set (vector) of numbers representing the column numbers for the predictor variables in the* `dataset` *data frame*

```
> vif(lm(DV ~ IV1 + IV2 + ..., dataset))
> 1/vif(lm(DV ~ IV1 + IV2 + ..., dataset))
```

where `DV` *and* `IV1`, `IV2`, ... *are the response and predictor variables respectively in the* `dataset` *data frame.*

(Multi)collinearity assumption met . . . . return to previous

b. (Multi)collinearity assumption not met - attempt one of the following:

- **Exclude one or more predictor variables - retain most biologically important on an priori theoretical basis** . . . . See Example 9A
- **(Multi)collinearity due to interactive/polynomial terms - center predictors** See Example 9B

```
> dataset$cIV1 <- scale(dataset$IV1, scale = F)
> dataset$cIV2 <- scale(dataset$IV2, scale = F)
> ...
```

where `IV1` *and* `IV2` *are two of the predictor variables in the* `dataset` *data frame.*

Return . . . . Return to previous

- **PCA regression** . . . . see Quinn and Keough (2002) chapter 17.

4 a. The effects of each predictor variable on the response variable are expected to be independent of other measured predictor variables (fit additive model) ..... See Example 9A

```
> data.lm <- lm(DV ~ IV1 + IV2 + .., dataset)
> plot(data.lm)
> summary(data.lm)
```

To summarize the partial relationships graphically .... Go to 12
To select the 'best' model or compare fit to other models .... Go to 8

b. The effects of one or more predictor variables are expected to depend on the level of other measured predictor variables and such interactions are of biological interest (fit multiplicative model) .... See Example 9B

```
> data.lm <- lm(DV ~ IV1 + IV2 + .. + IV1:IV2 + .., dataset)
> plot(data.lm)
> summary(data.lm)
```

where `DV` *and* `IV1`, `IV2`, . . . *are the response and predictor variables respectively in the* `dataset` *data frame.*
To summarize the partial relationships graphically .... Go to 12
Interaction(s) present .... Go to 6
To select the 'best' model or compare fit to other models .... Go to 8

5 a. Random/haphazard sampling not possible, observations not necessarily independent (randomization test) .... See Example 9E

```
> stat <- function(data, indices) {
+     summary(lm(DV ~ IV1 + IV2 + ..., data))$coef[,
+         3]
+ }
> rand.gen <- function(data, mle) {
+     out <- data
+     out$DV <- sample(out$DV, replace = F)
+     out
+ }
> library(boot)
> dataset.boot <- boot(dataset, stat, R = 1000,
+     sim = "parametric", ran.gen = rand.gen)
> t <- apply(apply(abs(dataset.boot$t), 1, ">=",
+     abs(dataset.boot$t0)) * 1, 1, "sum") + 1
> t/(dataset.boot$R + 1)
```

where `DV` *and* `IV1`, `IV2`, . . . *are the response and predictor variables respectively in the* `dataset` *data frame.*
Interaction(s) present .... Go to 6

b. Observations independent however data non-normal with few outliers (robust M-estimator test)

6 **Exploring interactions further** .... See Example 9B

```
> IV1_sd2 <- mean(IV1) - 2 * sd(IV1)
> data.lm2 <- lm(DV ~ IV2 * c(IV1 - IV1_sd2), data = dataset)
> summary(data.lm2)
```

where the effect of one of the predictor variables (`IV2`*) on the dependent variable (*`DV`*) is modeled for a value of another predictor variable (*`IV1`*) equal to its mean minus 1 standard deviation.*

Return .... Return to previous

7 a. **Relationship should theoretically asymptote (reach a plateau) (Nonlinear regression)** .... Go to 7

Power function

```
> dataset.nls <- nls(DV ~ alpha * IV^beta,
+       start = list(alpha = a, beta = b), dataset)
```

Logarithmic function

```
> dataset.nls <- nls(DV ~ alpha * log(IV),
+       start = list(alpha = a), dataset)
```

Exponential function

```
> dataset.nls <- nls(DV ~ alpha * exp(IV * beta),
+      start = list(alpha = a, beta = b), dataset)
```

where `DV` *and* `IV` *are the response and predictor variables respectively in the* `dataset` *data frame. The starting parameters* `a` *and* `b` *are numbers selected to represent the starting configuration (see Table 9.1).*

Examine the parameter estimates

```
> summary(dataset.nls)
```

b. **Relationship does not necessarily plateau (Polynomial regression)** .... see Example 9F

```
> data.lm3 <- lm(DV ~ IV + I(IV^2) + I(IV^3) + ..., dataset)
```

OR

```
> data.lm3 <- lm(DV ~ poly(IV, 3), dataset)
> plot(data.lm3)
```

Compare fit to that of a lower order polynomial

```
> data.lm2 <- lm(DV ~ IV + I(IV ~ 2) + ..., dataset)
> anova(data.lm2, data.lm3)
> summary(data.lm2)
```

To produce a summary plot .... Go to 11

8 Comparing the fit of two or more models (see table 9.2) .... See Example 9G
Additionally, to compare the fit of two or more parametric linear models via ANOVA

```
> anova(model.lm1, model.lm2, ...)
```

where `data.lm1` *and* `data.lm2`, ... *are two or more parametric linear models.*

9 Generating the 'best' predictive model (Model Selection)[i] ...... See Example 9C

```
> library(biology)
> Model.selection(data.lm)
> library(MuMIn)
> model.avg(get.models(dredge(data.glm)))
```

where `data.lm` *is the full fitted linear model containing all the predictor variable combinations.*

10 Determine the relative influence of each of the predictor variables (Hierarchical partitioning) .... See Example 9D

```
> library(hier.part)
> data.preds <- data.lm$model[, 1]
> hier.part(dataset$DV, data.preds, gof = "Rsqu")
> rand.hp(dataset$DV, data.preds, gof = "Rsqu",
+     num.reps = 100)$Iprobs
```

11 Base summary plot for curvilinear regression .... See Example 9F &9G

```
> plot(V1 ~ V2, data, pch = 16, axes = F, xlab = "", ylab = "")
> axis(1, cex.axis = 0.8)
> mtext(text = "x-axis title", side = 1, line = 3)
> axis(2, las = 1)
> mtext(text = "y-axis title", side = 2, line = 3)
> box(bty = "l")
```

where `V1` *and* `V2` *are the continuous variables in the* `dataset` *data frame. For regression,* `V1` *represents the response variable and* `V2` *represents the predictor variable.*

Adding fitted regression line .... See Example 9F&9G

```
> x <- seq(min(dataset$IV), max(dataset$IV), l = 1000)
> points(x, predict(model, data.frame(IV = x)), type = "l")
```

where `IV` *represents the predictor variable within the* `dataset` *data frame and* `model` *represents a fitted regression model.*

---

[i] The `MuMIn` *package* is not yet part of the official comprehensive R archive network (CRAN). The package can be downloaded from http://mumin.r-forge.r-project.org/ or installed from within R: `> install.packages("MuMIn", repos="http://R-Forge.R-project.org")`.

12 Exploring added variable plots to illustrate the relationships between the response variable and each of the predictor terms . . . . See Example 9A

```
> av.plots(data.lm, ask = F)
```

where `DV` *and* `IV1`, `IV2`, . . . *are the response and predictor variables respectively in the* `dataset` *data frame.*

13 Perform binary recursive partitioning (Regression tree) . . . . See Example 9H

```
> library(tree)
> data.tree <- tree(DV ~ IV1 + IV2 + ..., dataset,
+     mindev = 0)
```

where `DV` *and* `IV1`, `IV2`, . . . *are the response and predictor variables respectively in the* `dataset` *data frame.*

To examine a residual plot

```
> plot(residuals(data.tree) ~ predict(data.tree))
```

To construct the graphical tree

```
> plot(data.tree, type = "uniform")
> text(data.tree, cex = 0.5, all = T)
> text(data.tree, lab = paste("n"), cex = 0.5, adj = c(0,
+     2), splits = F)
```

For tree pruning . . . . Go to 14

14 Regression tree pruning . . . . See Example 9H

To investigate a const-complexity measure plot

```
> plot(prune.tree(data.tree))
```

To prune the tree to a specific number of branches (e.g. 3)

```
> data.tree.prune <- prune.tree(data.tree, best = 3)
```

9.11 Worked examples of real biological data sets

Example 9A: Multiple linear regression - additive model

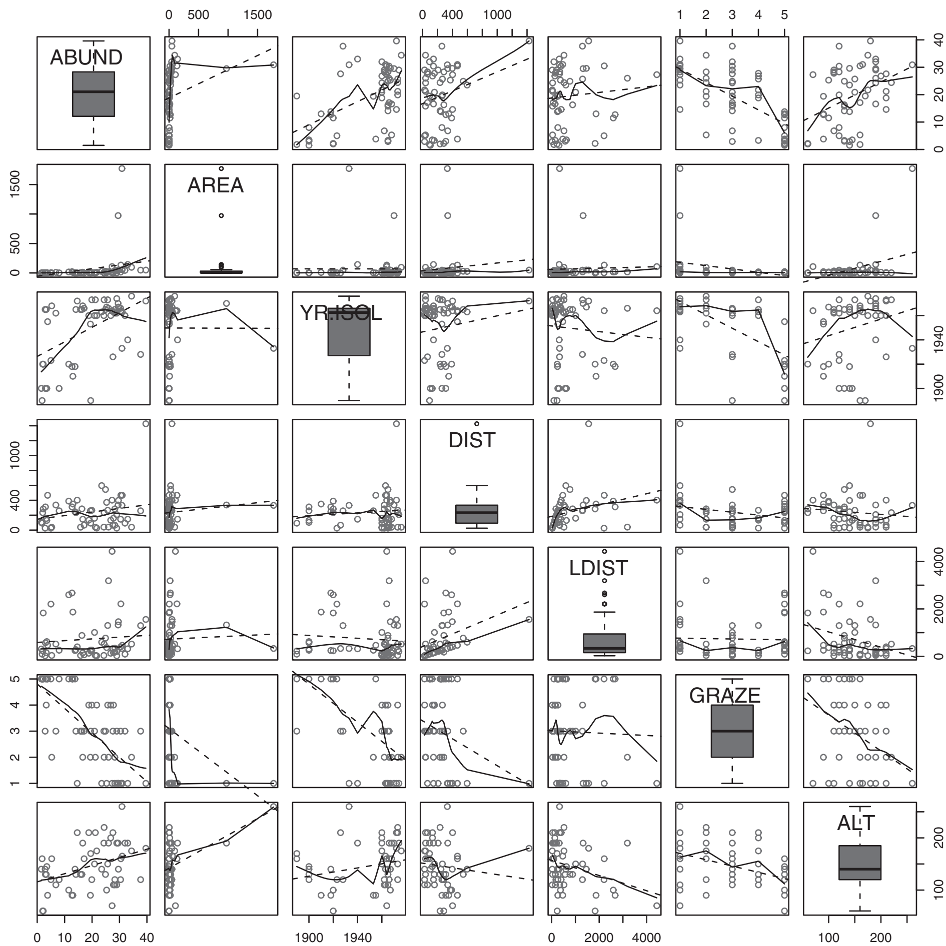
To investigate the effects of habitat fragmentation, Loyn (1987) related the abundance of forest birds to a range of variables (including patch area, number of years of isolation, distance to the nearest patch and larger patch, grazing intensity and altitude) collected from a total of 56 forest patches throughout Victoria (Box 6.2 Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Loyn (1987) data set

```
> loyn <- read.table("loyn.csv", header = T, sep = ",")
```

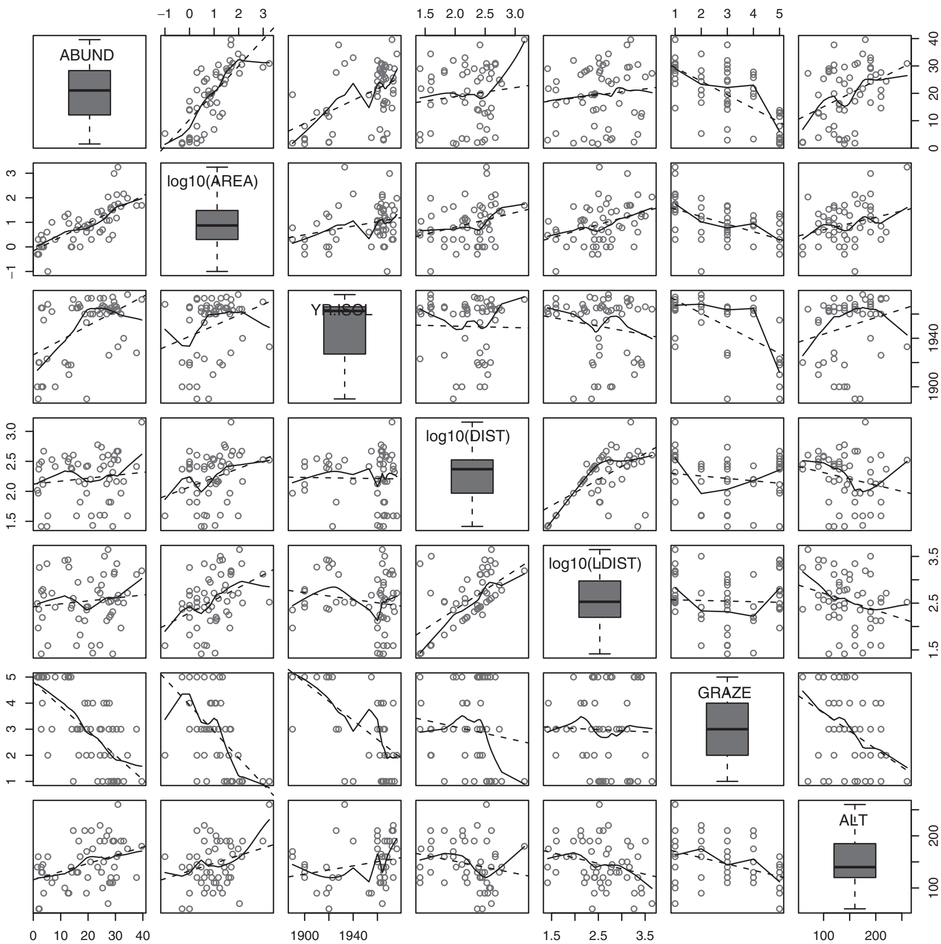
Step 2 (Key 9.2) - Assess assumptions of linearity, normality and homogeneity of variance.

```
> library(car)
> scatterplot.matrix(~ABUND + AREA + YR.ISOL + DIST +
+     LDIST + GRAZE + ALT, data = loyn, diag = "boxplot")
```



Conclusions - AREA, DIST and LDIST variables obviously non-normal (asymmetrical boxplots) and consequently the relationships between each of these variables and the response variable (ABUND) show non-linearity. In light of the normality problems, homogeneity of variance is difficult to assess. Scale transformations of the non-normal variables should be attempted.

```
> scatterplot.matrix(~ABUND + log10(AREA) + YR.ISOL +
+     log10(DIST) + log10(LDIST) + GRAZE + ALT, data = loyn,
+     diag = "boxplot")
```



Conclusions - $\log_{10}$ transformation appear successful, no evidence of non-normality (symmetrical boxplots), non-homogeneity of variance (even spread of points around each trend) or non-linearity.

Step 3 (Key 9.3) - Assess multicollinearity.

```
> cor(loyn[, 2:7])
                AREA      YR.ISOL       DIST       LDIST
AREA     1.000000000 -0.001494192  0.1083429  0.03458035
YR.ISOL -0.001494192  1.000000000  0.1132175 -0.08331686
DIST     0.108342870  0.113217524  1.0000000  0.31717234
LDIST    0.034580346 -0.083316857  0.3171723  1.00000000
GRAZE   -0.310402417 -0.635567104 -0.2558418 -0.02800944
ALT      0.387753885  0.232715406 -0.1101125 -0.30602220
              GRAZE        ALT
AREA    -0.31040242  0.3877539
YR.ISOL -0.63556710  0.2327154
DIST    -0.25584182 -0.1101125
```

```
LDIST   -0.02800944 -0.3060222
GRAZE    1.00000000 -0.4071671
ALT     -0.40716705  1.0000000
```

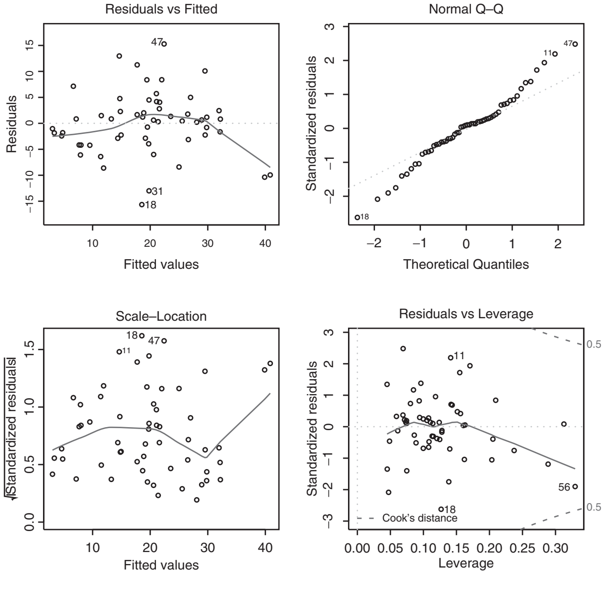
Conclusions - With the exception of GRAZE and YR.ISOL, none of the predictor variables are particularly correlated to one another.

```
> vif(lm(ABUND ~ log10(AREA) + YR.ISOL + log10(DIST) +
+     log10(LDIST) + GRAZE + ALT, data = loyn))
 log10(AREA)      YR.ISOL  log10(DIST) log10(LDIST)        GRAZE
    1.911514     1.804769     1.654553     2.009749     2.524814
         ALT
    1.467937
> 1/vif(lm(ABUND ~ log10(AREA) + YR.ISOL + log10(DIST) +
+     log10(LDIST) + GRAZE + ALT, data = loyn))
 log10(AREA)      YR.ISOL  log10(DIST) log10(LDIST)        GRAZE
   0.5231454    0.5540876    0.6043930    0.4975746    0.3960688
         ALT
   0.6812282
```

Conclusions - Variance inflation and their inverses (tolerances) are less than 5 and greater than 0.2 respectively suggesting that multicollinearity is unlikely to be a problem.

Step 4 (Key 9.4) - fit the additive multiple linear model relating bird abundance to the range of appropriately scaled patch characteristics.

```
> loyn.lm <- lm(ABUND ~ log10(AREA) + YR.ISOL + log10(DIST) +
+     log10(LDIST) + GRAZE + ALT, data = loyn)

> plot(loyn.lm)
```



Conclusions - There is no obvious "wedge" pattern evident in the residual plot (confirming that the assumption of homogeneity of variance is likely to be met). The Q-Q normal plot does not deviate greatly from normal. Finally, none of the points approach the high Cook's D contours suggesting that none of the observations are overly influential on the final fitted model.

```
> influence.measures(loyn.lm)

      dfb.1_   dfb.l10(A     dfb.YR.I   dfb.l10(D    dfb.l10(L
1 -0.02454653  0.32534847  0.008468066  0.08370776 -0.022663517
2 -0.01750873  0.01265303  0.016012689 -0.01656030  0.020997123
3 -0.05891170  0.04830884  0.060903999  0.01044557 -0.016320746
4 -0.02464857 -0.04735981  0.028326646 -0.01082504 -0.015503647
5  0.06451364 -0.09167341 -0.078406403  0.17235656 -0.075678399
6 -0.01395526 -0.02707540  0.014184325  0.01153817  0.003907139
     dfb.GRAZ       dfb.ALT        dffit    cov.r       cook.d
1  0.218999564 -0.0055469496 -0.42060699 1.394989 0.0254974592
2  0.003658088  0.0372465169 -0.06571529 1.319078 0.0006293951
3  0.012240659 -0.0219517552 -0.11033159 1.287647 0.0017717789
4 -0.005964993  0.0102469605  0.09983048 1.216839 0.0014493334
5  0.105181168  0.1013851217  0.35751545 1.035693 0.0181201227
6 -0.003666825  0.0009195532  0.03845593 1.243342 0.0002155830
         hat
1 0.23735383
2 0.12793356
3 0.11497013
4 0.06900608
5 0.08492694
6 0.07336138

...
```

Conclusions - Whilst a couple of the leverage (hat) values are greater than $2 * p/n = 0.286$ and therefore potentially outliers in x-space, none of the Cook's D values are ≥ 1. Hence the hypothesis tests are likely to be reliable.

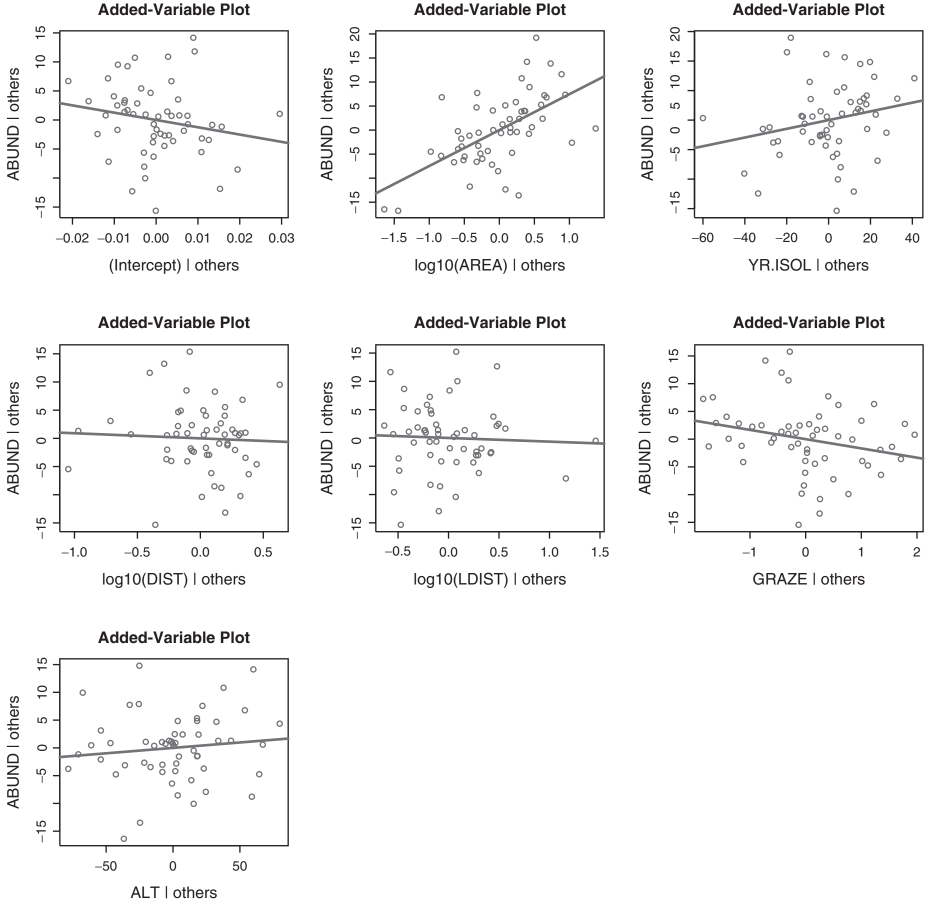
```
> summary(loyn.lm)

Call:
lm(formula = ABUND ~ log10(AREA) + YR.ISOL + log10(DIST) +
    log10(LDIST) + GRAZE + ALT, data = loyn)

Residuals:
     Min       1Q   Median       3Q      Max
-15.6506  -2.9390   0.5289   2.5353  15.2842

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)  -125.69725   91.69228  -1.371   0.1767
log10(AREA)     7.47023    1.46489   5.099 5.49e-06 ***
YR.ISOL         0.07387    0.04520   1.634   0.1086
log10(DIST)    -0.90696    2.67572  -0.339   0.7361
log10(LDIST)   -0.64842    2.12270  -0.305   0.7613
```

```
GRAZE          -1.66774    0.92993  -1.793   0.0791 .
ALT             0.01951    0.02396   0.814   0.4195
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.384 on 49 degrees of freedom
Multiple R-squared: 0.6849,     Adjusted R-squared: 0.6464
F-statistic: 17.75 on 6 and 49 DF,  p-value: 8.443e-11
```

Conclusions - there was a significant positive partial slope for bird abundance against $\log_{10}$ patch area. The overall model explained 69% of the variability in bird abundances across the 56 patches in Victoria. Bird abundances were found to increase with increasing patch area, but were not found to be significantly effected by grazing, altitude, years of isolation and distance to nearest patch or larger patch.

Step 5 (Key 9.12) - explore plots of the individual partial relationships between the response variable and each of the predictor variables (holding the other predictor variables constant).

```
> av.plots(loyn.lm, ask = F)
```



Example 9B: Multiple linear regression - multiplicative model

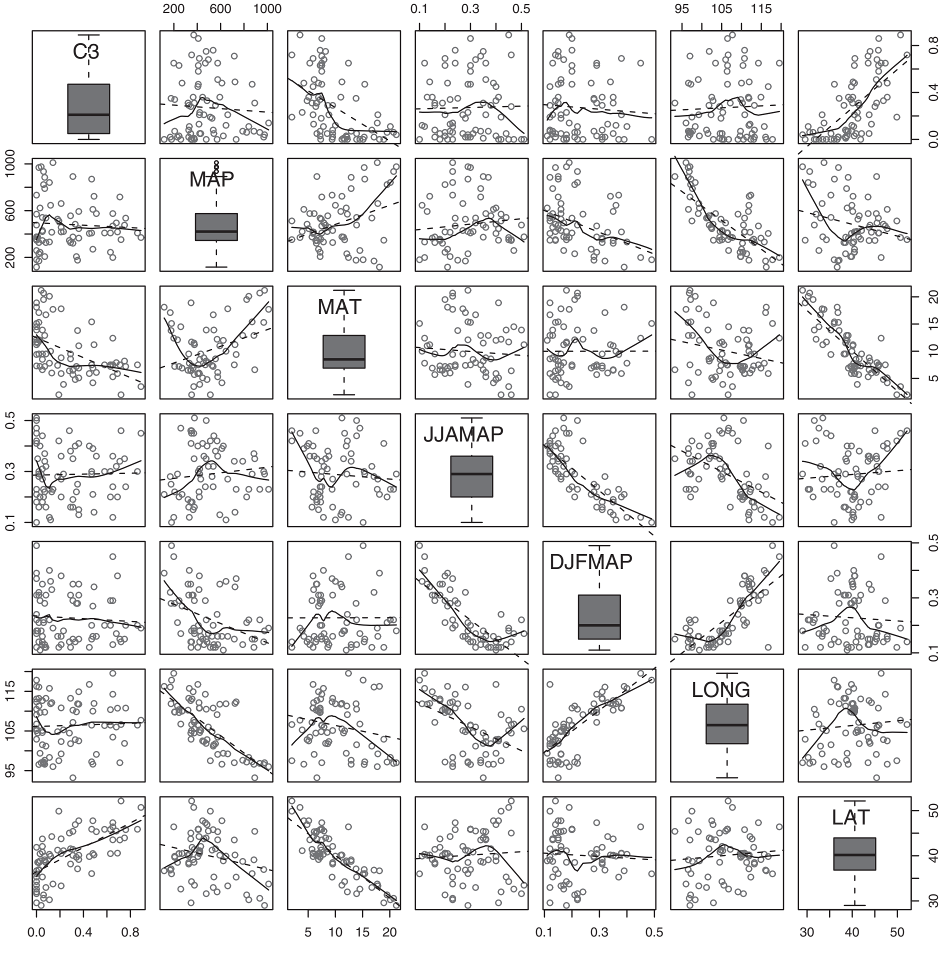
Paruelo and Lauenroth (1996) investigated the geographic (latitude and longitude) and climatic (mean annual temperature, means annual precipitation and the proportion of the mean annual precipitation that fall in the periods June-August and December-February) patterns in the relative abundance of C_3 plants throughout 73 sites across North America (Box 6.1 Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Paruelo and Lauenroth (1996) data set

```
> paruelo <- read.table("paruelo.csv", header = T,
+     sep = ",")
```

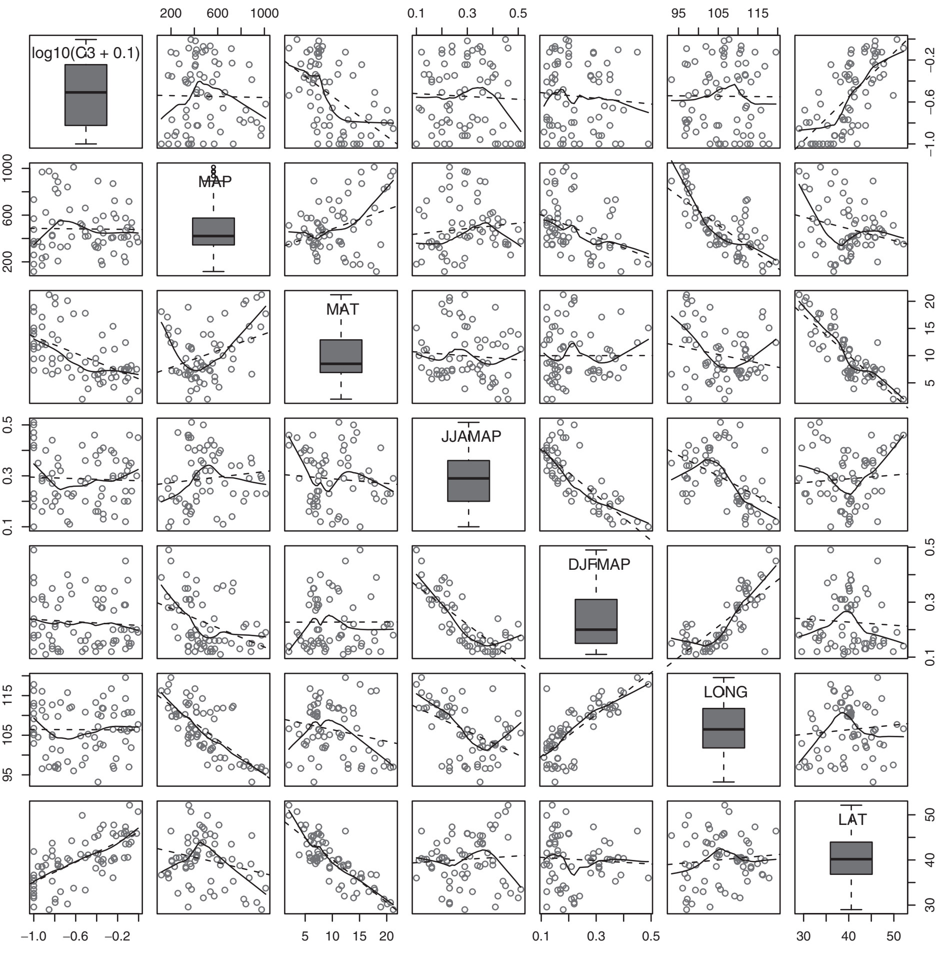
Step 2 (Key 9.2) - Assess assumptions of linearity, normality and homogeneity of variance.

```
> library(car)
> scatterplot.matrix(~C3 + MAP + MAT + JJAMAP + DJFMAP +
+     LONG + LAT, data = paruelo, diag = "boxplot")
```



Conclusions - whilst all the predictor variables appear normally distributed (symmetrical boxplots), the response variable (C3) appears to be positively skewed and thus a candidate for scale transformation (either a root transformation or a heavier log transformation). Paruelo and Lauenroth (1996) and therefore Quinn and Keough (2002) used a $\log_{10}(y+1)$. Note that as there are 0 values present and that log(0) cannot be evaluated, a small constant (such as 0.1[j]) must be added to each count in the response variable prior to the log transformation.

```
> scatterplot.matrix(~log10(C3 + 0.1) + MAP + MAT +
+      JJAMAP + DJFMAP + LONG + LAT, data = paruelo,
+      diag = "boxplot")
```



Conclusions - transformation appear successful, now no evidence of non-normality (symmetrical boxplots), non-homogeneity of variance (even spread of points around each trend) or

[j] This constant value should be small relative to the values in the variable so that it does not overshadow the existing values. However, if the value is more than two orders of magnitude smaller than the majority of the values, it will make the zero values outliers (influential points).

non-linearity. However there is some indication that multicollinearity could be an issue (there are some strong trends between pairs of predictor variables).

Step 3 (Key 9.3) - Assess multicollinearity.

```
> cor(paruelo[, 2:7])
              MAP          MAT      JJAMAP       DJFMAP
MAP     1.0000000  0.355090766  0.11225905 -0.404512409
MAT     0.3550908  1.000000000 -0.08077131  0.001478037
JJAMAP  0.1122590 -0.080771307  1.00000000 -0.791540381
DJFMAP -0.4045124  0.001478037 -0.79154038  1.000000000
LONG   -0.7336870 -0.213109100 -0.49155774  0.770743994
LAT    -0.2465058 -0.838590413  0.07417497 -0.065124848
              LONG         LAT
MAP    -0.73368703 -0.24650582
MAT    -0.21310910 -0.83859041
JJAMAP -0.49155774  0.07417497
DJFMAP  0.77074399 -0.06512485
LONG    1.00000000  0.09655281
LAT     0.09655281  1.00000000
```

Conclusions - as was expected, some pairs of predictor variables (`MAP` & `LONG`, `MAT` & `LAT` and `JJAMAP` & `DJFMAP`) are strongly correlated to one another suggesting multicollinearity could potentially be a problem.

```
> vif(lm(log10(C3 + 0.1) ~ MAP + MAT + JJAMAP + DJFMAP +
+     LONG + LAT, data = paruelo))
     MAP      MAT   JJAMAP   DJFMAP     LONG      LAT
2.799428 3.742780 3.163215 5.710315 5.267618 3.502732
> 1/vif(lm(log10(C3 + 0.1) ~ MAP + MAT + JJAMAP + DJFMAP +
+     LONG + LAT, data = paruelo))
      MAP       MAT    JJAMAP    DJFMAP      LONG       LAT
0.3572159 0.2671810 0.3161340 0.1751217 0.1898391 0.2854914
```

Conclusions - Some of the variance inflation and their inverses (tolerances) are approaching 5 and 0.2 respectively again suggesting that multicollinearity could be a problem. Paruelo and Lauenroth (1996) and Quinn and Keough (2002) decided to split the analysis up into two smaller analyses (**Key 9.3b**), one representing an investigation of geographic distribution and the other investigating the climatic factors. different aspects of the overall study.

Step 4 - The investigation of geographic patterns in C_3 plant abundances would model the log transformed abundance of C_3 plants against latitude and longitude. The extent of any latitudinal effects might be expected to depend on longitude and visa versa. For example, perhaps longitudinal effects are only important above or below a certain latitudes. Such possibilities suggest that fitting a more complicated multiplicative model (with interaction effects) might be more informative than an additive model.

Step 5 (Key 9.3) - check multicollinearity by assessing tolerances.

```
> 1/vif(lm(log10(C3 + 0.1) ~ LAT + LONG + LAT:LONG,
+     data = paruelo))
        LAT        LONG    LAT:LONG
0.003249445 0.014973575 0.002494144
```

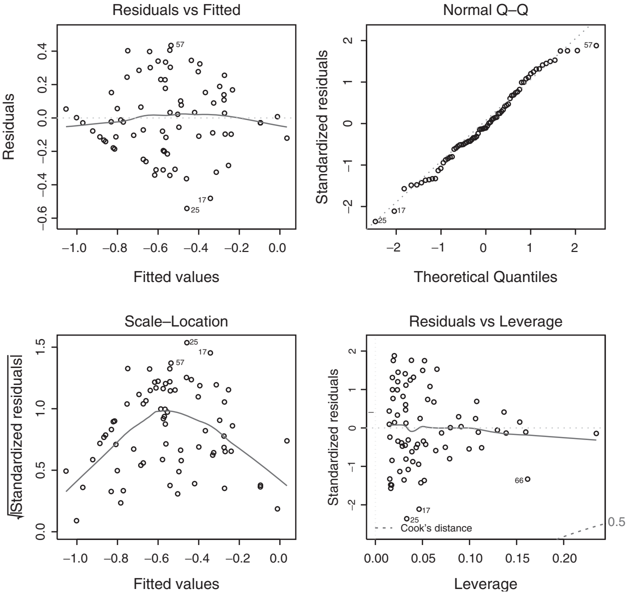
Conclusions - not surprisingly, there are very low tolerances since each of the individual predictors are going to be correlated to their interaction term. Centering (**Key 9.3b**) the predictor variables before re-fitting the model should address this.

```
> paruelo$cLAT <- paruelo$LAT-mean(paruelo$LAT)
> #OR
> paruelo$cLAT <- scale(paruelo$LAT, scale=F)
> paruelo$cLONG <- scale(paruelo$LONG, scale=F)
> 1/vif(lm(log10(C3+.1)~cLAT+cLONG+cLAT:cLONG, data=paruelo))
      cLAT      cLONG cLAT:cLONG
 0.8268942  0.9799097  0.8195915
```

Conclusions - multicollinearity is no longer likely to be a problem and the parameter estimates and hypothesis tests are likely to be reliable.

Step 6 (Key 9.4b) - fit the multiplicative linear model and test whether each of the partial population slopes are likely to equal zero.

```
> paruelo.lm <- lm(log10(C3 + 0.1) ~ cLAT + cLONG +
+     cLAT:cLONG, data = paruelo)
> plot(paruelo.lm)
```



Conclusions - There is no obvious "wedge" pattern evident in the residual plot (confirming that the assumption of homogeneity of variance is likely to be met). The Q-Q normal plot does not deviate greatly from normal. Finally, none of the points approach the high Cook's D contours suggesting that none of the observations are overly influential on the final fitted model.

```
> influence.measures(paruelo.lm)

       dfb.1_    dfb.cLAT    dfb.cLON   dfb.cLAT:       dffit
1 -0.01240897 -0.04291203 -0.04343888 -0.06275532 -0.07869325
2 -0.01232348 -0.03577596 -0.02255957 -0.04094363 -0.05303525
3  0.07696884  0.12765517  0.06321144  0.11087334  0.17912507
4  0.17518366  0.09561479 -0.13875996 -0.06937259  0.25698909
5 -0.05221407 -0.05487872  0.03652972  0.01850913 -0.09147598
6 -0.16175075  0.02214619  0.17475473  0.00759321 -0.24141744
     cov.r       cook.d         hat
1 1.383538 0.0015704573 0.23466106
2 1.229880 0.0007133425 0.13890169
3 1.087217 0.0080746585 0.05557079
4 0.974066 0.0162711273 0.03171179
5 1.098320 0.0021171765 0.04482606
6 0.981941 0.0143925390 0.03048383

...
```

Conclusions - few leverage (hat) values are greater than $2 * p/n = 0.082$, none of the Cook's D values are approaching 1. Hence the hypothesis tests are likely to be reliable.

```
> summary(paruelo.lm)
Call:
lm(formula = log10(C3 + 0.1) ~ cLAT + cLONG + cLAT:cLONG,
  data = paruelo)

Residuals:
     Min       1Q   Median       3Q      Max
-0.54185 -0.13298 -0.02287  0.16807  0.43410

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.5529416  0.0274679 -20.130  < 2e-16 ***
cLAT         0.0483954  0.0057047   8.483 2.61e-12 ***
cLONG       -0.0025787  0.0043182  -0.597   0.5523
cLAT:cLONG   0.0022522  0.0008757   2.572   0.0123 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2334 on 69 degrees of freedom
Multiple R-squared: 0.5137,     Adjusted R-squared: 0.4926
F-statistic:  24.3 on 3 and 69 DF,  p-value: 7.657e-11
```

Conclusions - reject the H_0 that there is no interactive effect of latitude and longitude on the ($\log_{10}$) abundance of C_3 plants.

Step 7 (Key 9.6) - to further investigate this interaction, calculate the simple slopes of C_3 plant abundance against longitude for a range of latitudes (e.g. mean $\pm$ 1 standard deviation and $\pm$ 2 standard deviations). Since the partial slopes in the multiplicative model are the simple slopes for the mean values of the other predictor (hence partial effect of one predictor holding the other predictor variables constant), the simple slope of longitude at the mean latitude has already been calculated (−0.0026) and can be extracted from the summarized multiplicative model.

$\overline{x_1} - 2\sigma$ (mean centered longitude - 2 standard deviations)

```
> LAT_sd1 <- mean(paruelo$cLAT) - 2 * sd(paruelo$cLAT)
> paruelo_LONG.lm1 <- lm(log10(C3 + 0.1) ~ cLONG *
+     c(cLAT - LAT_sd1), data = paruelo)
> summary(paruelo_LONG.lm1)
Call:
lm(formula = log10(C3 + 0.1) ~ cLONG * c(cLAT - LAT_sd1),
  data = paruelo)

Residuals:
     Min       1Q   Median       3Q      Max
-0.54185 -0.13298 -0.02287  0.16807  0.43410

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              -1.0662239  0.0674922 -15.798  < 2e-16
cLONG                    -0.0264657  0.0098255  -2.694  0.00887
c(cLAT - LAT_sd1)         0.0483954  0.0057047   8.483 2.61e-12
cLONG:c(cLAT - LAT_sd1)   0.0022522  0.0008757   2.572  0.01227

(Intercept)              ***
cLONG                    **
c(cLAT - LAT_sd1)        ***
cLONG:c(cLAT - LAT_sd1) *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2334 on 69 degrees of freedom
Multiple R-squared: 0.5137,     Adjusted R-squared: 0.4926
F-statistic:  24.3 on 3 and 69 DF,  p-value: 7.657e-11
```

$\overline{x_1} - 1\sigma$ (mean centered longitude - 1 standard deviation)

```
> LAT_sd2 <- mean(paruelo$cLAT) - 1 * sd(paruelo$cLAT)
> paruelo_LONG.lm2 <- lm(log10(C3 + 0.1) ~ cLONG *
+     c(cLAT - LAT_sd2), data = paruelo)
> summary(paruelo_LONG.lm2)
Call:
lm(formula = log10(C3 + 0.1) ~ cLONG * c(cLAT - LAT_sd2),
  data = paruelo)
```

```
Residuals:
     Min       1Q   Median       3Q      Max 
-0.54185 -0.13298 -0.02287  0.16807  0.43410 

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)             -0.8095827  0.0417093 -19.410  < 2e-16
cLONG                   -0.0145222  0.0060025  -2.419   0.0182
c(cLAT - LAT_sd2)        0.0483954  0.0057047   8.483 2.61e-12
cLONG:c(cLAT - LAT_sd2)  0.0022522  0.0008757   2.572   0.0123

(Intercept)             ***
cLONG                   *  
c(cLAT - LAT_sd2)       ***
cLONG:c(cLAT - LAT_sd2) *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2334 on 69 degrees of freedom
Multiple R-squared: 0.5137,     Adjusted R-squared: 0.4926 
F-statistic:  24.3 on 3 and 69 DF,  p-value: 7.657e-11
```

$\overline{x_1} + 1\sigma$ (mean centered longitude + 1 standard deviation)

```
> LAT_sd4 <- mean(paruelo$cLAT) - 1 * sd(paruelo$cLAT)
> paruelo_LONG.lm4 <- lm(log10(C3 + 0.1) ~ cLONG *
+     c(cLAT - LAT_sd4), data = paruelo)
> summary(paruelo_LONG.lm4)
Call:
lm(formula = log10(C3 + 0.1) ~ cLONG * c(cLAT - LAT_sd4),
  data = paruelo)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.54185 -0.13298 -0.02287  0.16807  0.43410 

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)             -0.8095827  0.0417093 -19.410  < 2e-16
cLONG                   -0.0145222  0.0060025  -2.419   0.0182
c(cLAT - LAT_sd4)        0.0483954  0.0057047   8.483 2.61e-12
cLONG:c(cLAT - LAT_sd4)  0.0022522  0.0008757   2.572   0.0123

(Intercept)             ***
cLONG                   *  
c(cLAT - LAT_sd4)       ***
cLONG:c(cLAT - LAT_sd4) *  
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2334 on 69 degrees of freedom
Multiple R-squared: 0.5137,     Adjusted R-squared: 0.4926
F-statistic:  24.3 on 3 and 69 DF,  p-value: 7.657e-11
```

$\overline{x_1} + 2\sigma$ (mean centered longitude + 2 standard deviation)

```
> LAT_sd5 <- mean(paruelo$cLAT) - 1 * sd(paruelo$cLAT)
> paruelo_LONG.lm5 <- lm(log10(C3 + 0.1) ~ cLONG *
+     c(cLAT - LAT_sd5), data = paruelo)
> summary(paruelo_LONG.lm5)
Call:
lm(formula = log10(C3 + 0.1) ~ cLONG * c(cLAT - LAT_sd5),
  data = paruelo)

Residuals:
     Min       1Q   Median       3Q      Max
-0.54185 -0.13298 -0.02287  0.16807  0.43410

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              -0.8095827  0.0417093 -19.410  < 2e-16
cLONG                    -0.0145222  0.0060025  -2.419   0.0182
c(cLAT - LAT_sd5)         0.0483954  0.0057047   8.483 2.61e-12
cLONG:c(cLAT - LAT_sd5)  0.0022522  0.0008757   2.572   0.0123

(Intercept)              ***
cLONG                    *
c(cLAT - LAT_sd5)        ***
cLONG:c(cLAT - LAT_sd5) *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2334 on 69 degrees of freedom
Multiple R-squared: 0.5137,     Adjusted R-squared: 0.4926
F-statistic:  24.3 on 3 and 69 DF,  p-value: 7.657e-11
```

Conclusions - the abundance of C_3 plants is negatively related to longitude at low latitudes however this longitudinal effect diminishes with increasing latitude and becomes a positive effect at very high latitudes. Additionally (or alternatively), latitudinal effects could be seen to become more positive with increasing longitude (from east to west).

Example 9C: Selecting the 'best' regression model

Quinn and Keough (2002) used the Loyn (1987) data set (analysed in Example 9A on page 224) demonstrated the use of various criteria as the basis of selecting the 'best' model

(Quinn and Keough (2002) Box 6.8). Continuing on from Example 9A, we will attempt to determine the 'best', most parsimonious regression model for the purpose of either generating a predictive model or simply to determine which predictor variables have the greatest relative influence on the response variable.

Step 1 (Key 9.9b) - Compare the fit of all additive combinations of predictor variables from the full fitted linear model of the Loyn (1987) data set via AIC, BIC, C_p and adjusted r^2.

```
> library(biology)
> Model.selection(loyn.lm)

                                                 Adj.r.sq      AIC     AICc   deltaAIC
1. log10(AREA)                                 0.53927618 224.3964 227.4602 14.2082619
2. YR.ISOL                                     0.23954252 252.4592 255.5230 42.2710623
3. log10(DIST)                                -0.00216233 267.9149 270.9788 57.7267862
4. log10(LDIST)                               -0.00430673 268.0346 271.0984 57.8464855
5. GRAZE                                       0.45592959 233.7081 236.7719 23.5199360
6. ALT                                         0.13310788 259.7949 262.8587 49.6067453
7. log10(AREA)+YR.ISOL                         0.63002728 213.0649 216.1288  2.8768100
8. log10(AREA)+log10(DIST)                     0.54130423 225.1026 228.1664 14.9144529
9. log10(AREA)+log10(LDIST)                    0.56364647 222.3063 225.3701 12.1181257
...
24. log10(AREA)+YR.ISOL+GRAZE                  0.65436215 210.1881 213.2520  0.0000000
25. log10(AREA)+YR.ISOL+ALT                    0.64340828 211.9353 214.9992  1.7471955
26. log10(AREA)+log10(DIST)+log10(LDIST)       0.55526607 224.3049 227.3687 14.1167393
27. log10(AREA)+log10(DIST)+GRAZE              0.64144047 212.2435 215.3073  2.0553756
28. log10(AREA)+log10(DIST)+ALT                0.56137177 223.5307 226.5946 13.3425950
29. log10(AREA)+log10(LDIST)+GRAZE             0.64443577 211.7737 214.8376  1.5856031
...
39. log10(DIST)+log10(LDIST)+ALT               0.16767219 259.4029 262.4667 49.2147489
40. log10(DIST)+GRAZE+ALT                      0.45484515 235.7061 238.7699 25.5179860
41. log10(LDIST)+GRAZE+ALT                     0.47031877 234.0936 237.1575 23.9054939
42. log10(AREA)+YR.ISOL+log10(DIST)+log10(LDIST)  0.62461805 215.7237 218.7875  5.5355253
43. log10(AREA)+YR.ISOL+log10(DIST)+GRAZE      0.65360148 211.2238 214.2877  1.0356946
44. log10(AREA)+YR.ISOL+log10(DIST)+ALT        0.63704328 213.8387 216.9025  3.6505413
...
                Estimate Unconditional_SE   Lower95CI
log10(AREA)   7.54126720       1.43013594  4.73820077
YR.ISOL       0.06204083       0.03729047 -0.01104849
log10(DIST)  -0.51987543       0.87724385 -2.23927338
log10(LDIST) -0.52400077       0.75025473 -1.99450004
GRAZE        -1.73681399       0.83173477 -3.36701413
ALT           0.01065631       0.01150212 -0.01188785
               Upper95CI
log10(AREA)  10.34433364
YR.ISOL       0.13513016
log10(DIST)   1.19952252
log10(LDIST)  0.94649850
GRAZE        -0.10661385
ALT           0.03320047
attr(,"heading")
[1] "Model averaging\n"  "Response: ABUND \n"
```

Note some of the rows and columns have been omitted from the above output to conserve space.
Alternatively, using the `MuMIn` *package*

```
> library(MuMIn)
> model.avg(get.models(dredge(loyn.lm, rank = "AIC")))
Model summary:
            Deviance AIC Delta Weight
2+3+6           2070 371 0.000 0.1330
1+2+3+6         2010 372 0.414 0.1080
2+3+5+6         2030 372 0.962 0.0820
2+3+4+6         2040 372 1.040 0.0790
2+3             2200 373 1.400 0.0657
2+3+5           2130 373 1.590 0.0600
```

```
1+3+6           2140 373 1.750 0.0554
1+2+3           2150 373 2.050 0.0477
2+3+4           2150 373 2.060 0.0475
1+2+3+4+6       2000 373 2.060 0.0473
1+2+3+5+6       2000 373 2.090 0.0467
2+3+4+5+6       2020 374 2.710 0.0343
3+6             2260 374 2.880 0.0315
1+2+3+5         2110 374 3.080 0.0285
2+3+4+5         2120 374 3.340 0.0250
1+2+3+4         2120 374 3.370 0.0246
1+3+5+6         2130 375 3.520 0.0228
3+5+6           2210 375 3.610 0.0218
1+3+4+6         2130 375 3.650 0.0214
1+2+3+4+5+6     2000 375 3.950 0.0184

Variables:
         1           2           3            4            5            6
       ALT       GRAZE  log10(AREA)  log10(DIST) log10(LDIST)      YR.ISOL

Averaged model parameters:
             Coefficient Variance       SE Unconditional SE  Lower CI Upper CI
ALT               0.0107 1.46e-07   0.0177           0.0178   -0.0243   0.0457
GRAZE            -1.7900 1.81e+00   1.1200           1.1300   -4.0000   0.4330
(Intercept)     -99.4000 1.66e+08 111.0000         112.0000 -320.0000 121.0000
log10(AREA)       7.5000 4.07e+00   1.4100           1.4400    4.6700  10.3000
log10(DIST)      -0.4930 5.39e+00   1.1400           1.1600   -2.7600   1.7800
log10(LDIST)     -0.5130 2.95e+00   1.0600           1.0700   -2.6200   1.5900
YR.ISOL           0.0606 9.85e-06   0.0550           0.0556   -0.0485   0.1700

Relative variable importance:
 log10(AREA)        GRAZE      YR.ISOL          ALT log10(LDIST)  log10(DIST)
        1.00         0.85         0.70         0.42         0.34         0.30
```

Conclusions - AIC and C_p (not shown) both select a model with three predictor variables ($\log_{10}$area, grazing intensity and years of isolation). However, it should be noted, that using the rule-of-thumb that delta AIC values less than 2 do not represent significant improvements in fit, it could be argued that the three variable model is not significantly better than the simpler two variable ($\log_{10}$area and grazing intensity) model. Hence $\log_{10}$ patch area and grazing intensity are the most important measured influences on bird abundances across the fragmented Victorian landscape.

Step 2 - construct the predictive model

```
> loyn.lm2 <- lm(ABUND ~ log10(AREA) + GRAZE, data = loyn)
> summary(loyn.lm2)
Call:
lm(formula = ABUND ~ log10(AREA) + GRAZE, data = loyn)

Residuals:
     Min       1Q   Median       3Q      Max
-13.4296  -4.3186  -0.6323   4.1273  13.0739

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  21.6029     3.0917   6.987 4.73e-09 ***
log10(AREA)   6.8901     1.2900   5.341 1.98e-06 ***
GRAZE        -2.8535     0.7125  -4.005 0.000195 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 6.444 on 53 degrees of freedom
Multiple R-squared: 0.6527,     Adjusted R-squared: 0.6396
F-statistic: 49.81 on 2 and 53 DF,  p-value: 6.723e-13
```

Conclusions - the predictive model (resulting from the 'best' regression model is $abund = 6.89log_{10}area - 2.85graze + 21.60$ and explains approximately 65% of the variation in bird abundance.

Example 9D: Hierarchical partitioning

Quinn and Keough (2002) also used the Loyn (1987) data set (analysed in Example 9A on page 224) to demonstrate the use of hierarchical partitioning to determine the relative contributions of each of the predictor variables (Quinn and Keough (2002) Box 6.8).

Step 1 (Key 9.10) - Perform a hierarchical partitioning on the multiple linear model fitted to the Loyn (1987) data set. As this is a linear model, the goodness-of-fit of the model should be assessed by the r^2 value.

1. determine independent and joint contribution of each predictor variable averaged across all possible model combinations.

```
> library(hier.part)
> loyn.preds <- with(loyn, data.frame(logAREA = log10(AREA),
+      YR.ISOL, logDIST = log10(DIST), logLDIST = log10(LDIST),
+      GRAZE, ALT))
> hier.part(loyn$ABUND, loyn.preds, gof = "Rsqu")

$gfs
 [1] 0.00000000 0.54765298 0.25336902 0.01605880 0.01395339
 [6] 0.46582178 0.14886955 0.64348084 0.55798408 0.57951387
[11] 0.65273437 0.58357693 0.27202894 0.29411677 0.47394321
[16] 0.32970100 0.01878268 0.46670232 0.19573296 0.47484303
[21] 0.20305219 0.47978826 0.64797136 0.65145633 0.67321512
[26] 0.66285874 0.57952428 0.66099826 0.58529695 0.66383018
[31] 0.59521919 0.66105930 0.29441552 0.47580294 0.37071613
[36] 0.48827761 0.40728610 0.48872839 0.47606705 0.21307189
[41] 0.48458087 0.49921047 0.65191856 0.67879410 0.66344013
[46] 0.67921724 0.66420358 0.68234183 0.66529515 0.59537174
[51] 0.66514424 0.66687281 0.48949273 0.40962297 0.49609855
[56] 0.51765498 0.49933677 0.68067311 0.66425545 0.68433597
[61] 0.68419720 0.66776512 0.51772763 0.68493595

$IJ
                   I            J      Total
logAREA  0.315204510 0.2324484698 0.54765298
YR.ISOL  0.101458466 0.1519105495 0.25336902
logDIST  0.007285099 0.0087737041 0.01605880
```

```
logLDIST 0.013677502 0.0002758905 0.01395339
GRAZE    0.190462561 0.2753592211 0.46582178
ALT      0.056847811 0.0920217408 0.14886955

$I.perc
                 I
logAREA  46.019560
YR.ISOL  14.812840
logDIST   1.063618
logLDIST  1.996902
GRAZE    27.807354
ALT       8.299727
```

Conclusions - $\log_{10}$area and grazing intensity contribute most to the explained variance in bird abundance (46.0 and 27.8% respectively), although years of isolation and to a lesser degree, altitude also make some contributions.

2. determine the likelihood that the independent contributions for each predictor variable could be due to change by performing a randomization test and assessing the significance of Z scores at the 95% level. Note that this procedure takes some time.

```
> r.HP <- rand.hp(loyn$ABUND, loyn.preds, gof = "Rsqu",
+     num.reps = 100)$Iprobs

          Obs Z.score sig95
logAREA  0.32   11.86     *
YR.ISOL  0.10    2.67     *
logDIST  0.01   -0.50
logLDIST 0.01   -0.12
GRAZE    0.19    8.99     *
ALT      0.06    1.09
```

Conclusions - the individual contributions of $\log_{10}$area, grazing, and years of isolation were all found to be significantly greater than would be expected by chance and therefore each has some influence on the abundance of forest birds within habitat patches across Victoria.

Example 9E: Randomization and multiple regression

McKechnie et al. (1975) investigated the relationship between the frequency of hezokinase (HK) 1.00 mobility genes and a range of climatic conditions (including altitude, temperature and precipitation) from colonies of *Euphydras editha* butterflies (example 8.3 Manly (1991)).

Step 1 - Import (section 2.3) the McKechnie et al. (1975) data set

```
> mckechnie2 <- read.table("mckechnie2.csv", header = T,
+     sep = ",")
```

Step 2 (Key 9.2) - Assess linearity, normality and homogeneity of variance using a scatterplot with marginal boxplots and a lowess smoother.

For the purpose of this demonstration, lets assume that the assumption of normality could not be met and more importantly, that the observations are not independent, thereby necessitating an alternative regression method.

Step 3 (Key 9.3) - assess (multi)collinearity.

```
> library(car)
> vif(lm(HK ~ PRECIP + MAXTEMP + MINTEMP + ALT, mckechnie2))
  PRECIP  MAXTEMP  MINTEMP      ALT
2.242274 3.375163 6.727932 1.921078
```

Conclusions - there is some indication of a collinearity issue concerning the minimum temperature variable (VIF greater than 5), however this will be overlooked for consistency with Manly (1991).

Step 4 (Key 9.5) - use a randomization test to test whether the observed trends could be due to chance.

1. use conventional multiple regression methods[k] to estimate the regression parameters.

```
> mckechnie2.lm <- lm(HK ~ PRECIP + MAXTEMP + MINTEMP +
+     ALT, mckechnie2)
> summary(mckechnie2.lm)
Call:
lm(formula = HK ~ PRECIP + MAXTEMP + MINTEMP + ALT,
  data = mckechnie2)

Residuals:
    Min      1Q  Median      3Q     Max
-50.995  -5.141   2.656  10.091  29.620

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -88.5645   101.1793  -0.875  0.39728
PRECIP        0.4720     0.4955   0.952  0.35823
MAXTEMP       0.8668     1.1725   0.739  0.47290
MINTEMP       0.2503     1.0195   0.246  0.80986
ALT          26.1237     8.6450   3.022  0.00982 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 20.95 on 13 degrees of freedom
Multiple R-squared: 0.647,      Adjusted R-squared: 0.5384
F-statistic: 5.957 on 4 and 13 DF,  p-value: 0.005984
```

[k] Consistent with Manly (1991), I have used OLS to estimate the regression parameters. However, these parameters could alternatively be RMA or non-parametric regression estimates.

2. define the statistic (again this example uses OLS) to use in the randomization test - in this case the *t*-statistics for each of the estimated parameters.

```
> stat <- function(data, indices) {
+     summary(lm(HK ~ PRECIP + MAXTEMP + MINTEMP +
+         ALT, data))$coef[, 3]
+ }
```

3. define how the data should be randomized - randomize the response-predictor pairings (shuffle the response variable without replacement).

```
> rand.gen <- function(data, mle) {
+     out <- data
+     out$HK <- sample(out$HK, replace = F)
+     out
+ }
```

4. call a bootstrapping procedure to randomize 1000 times (this can take some time)

```
> library(boot)
> mckechnie2.boot <- boot(mckechnie2, stat, R = 1000,
+     sim = "parametric", ran.gen = rand.gen)
```

5. calculate the number of possible *t*-values (including the observed *t*-value, which is one possible outcome) that were greater or equal to the observed *t*-value (for each parameter) and express these as a percentage of the number of randomizations (plus one for the observed outcomes).

```
> t <- apply(apply(abs(mckechnie2.boot$t), 1, ">=",
+     abs(mckechnie2.boot$t0)) * 1, 1, "sum") + 1
> t/(mckechnie2.boot$R + 1)
(Intercept)      PRECIP     MAXTEMP     MINTEMP         ALT
 0.39360639  0.36563437  0.48151848  0.79620380  0.00999001
```

6. perform a similar randomization to investigate the ANOVA *F*-ratio. This requires a couple of minor adjustments of the above procedures.

```
> stat <- function(data, indices) {
+     summary(lm(HK ~ PRECIP + MAXTEMP + MINTEMP +
+         ALT, data))$fstatistic
+ }
> rand.gen <- function(data, mle) {
+     out <- data
+     out$HK <- sample(out$HK, replace = F)
+     out
+ }
> mckechnie2.boot <- boot(mckechnie2, stat, R = 1000,
+     sim = "parametric", ran.gen = rand.gen)
> f <- apply(apply(abs(mckechnie2.boot$t), 1, ">=",
+     abs(mckechnie2.boot$t0)) * 1, 1, "sum") + 1
> f/(mckechnie2.boot$R + 1)
```

```
        value       numdf       dendf
  0.006993007 1.000000000 1.000000000
```

Conclusions - in this case, the p-values for both regression parameters and the overall ANOVA are almost identical to those produced via conventional regression analysis.

Example 9F: Polynomial regression

Sokal and Rohlf (1997) present an unpublished data set (R. K. Koehn) in which the nature of the relationship between *Lap*^{94} allele frequency in *Mytilus edulis* and distance (in miles) from Southport was investigated (Box 16.5, Sokal and Rohlf (1997)).

Step 1 - Import (section 2.3) the mytilus data set

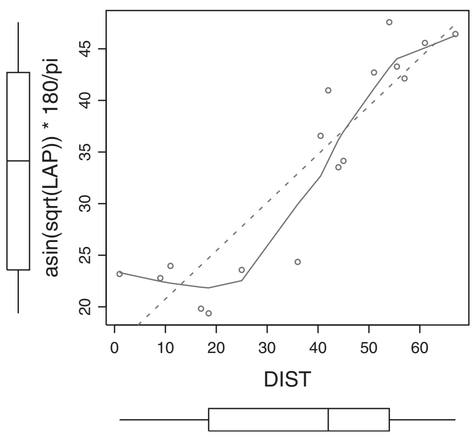
```
> mytilus <- read.table("mytilus.csv", header = T,
+     sep = ",")
```

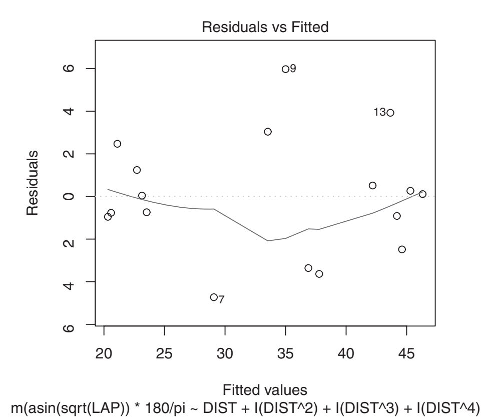
As a matter of course, Sokal and Rohlf (1997) transform frequencies using angular transformations (arcsin transformations) and henceforth *Lap*^{94} will be transformed in-line using `asin(sqrt(LAP))*180/pi`.

Step 2 (Key 8.2a) - confirm that simple linear regression does not adequately describe the relationship between *Lap*^{94} allele frequency and distance by examining a scatterplot and residual plot.

```
> library(car)
> scatterplot(asin(sqrt(LAP)) *
+     180/pi ~ DIST,
+     data = mytilus)
```

```
> plot(lm(asin(sqrt(LAP)) *
+     180/pi ~ DIST,
+     data = mytilus),
+     which = 1)
```





Conclusions - the scatterplot smoother suggests a potentially non-linear relationship and a persisting pattern in the residuals further suggests that the linear model is inadequate for explaining the response variable.

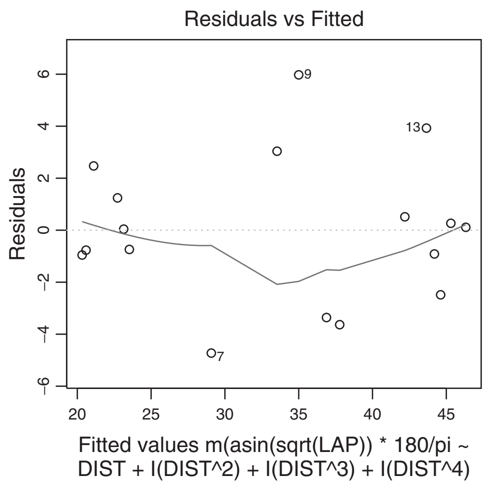
Step 3 (Key 9.7b) - fit a polynomial regression (additive multiple regression) model incorporating up to the fifth power (5^{th} order polynomial)[l].

1. Fit the quintic model

```
> mytilus.lm5 <- lm(asin(sqrt(LAP)) * 180/pi ~ DIST +
+     I(DIST^2) + I(DIST^3) + I(DIST^4) + I(DIST^5),
+     mytilus)
```

2. Examine the diagnostics

```
> plot(mytilus.lm5, which = 1)
```



Conclusions - no "wedge" pattern suggesting that homogeneity of variance and there is no persisting pattern suggesting that the fitted model is appropriate for modeling these data.

3. Examine the fit of the model including the contribution of different powers

```
> anova(mytilus.lm5)
Analysis of Variance Table

Response: asin(sqrt(LAP)) * 180/pi
          Df  Sum Sq Mean Sq  F value    Pr(>F)
DIST       1 1418.37 1418.37 125.5532 2.346e-07 ***
I(DIST^2)  1   57.28   57.28   5.0701   0.04575 *
I(DIST^3)  1   85.11   85.11   7.5336   0.01907 *
I(DIST^4)  1   11.85   11.85   1.0493   0.32767
I(DIST^5)  1   15.99   15.99   1.4158   0.25915
Residuals 11  124.27   11.30
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - powers of distance beyond a cubic (3) do not make significant contributions to explaining the variation in arcsin transformed *Lat*[94] allele frequency.

[l] Note that trends beyond a third order polynomial are unlikely to have much biological basis and are likely to be over-fit.

4. The improved fit (and significance) attributed to an additional power can be evaluated by comparing the fit of the higher order models against models one lower in order.

```
> mytilus.lm1 <- lm(asin(sqrt(LAP)) * 180/pi ~ DIST,
+     mytilus)
> mytilus.lm2 <- lm(asin(sqrt(LAP)) * 180/pi ~ DIST +
+     I(DIST^2), mytilus)
> anova(mytilus.lm2, mytilus.lm1)
Analysis of Variance Table

Model 1: asin(sqrt(LAP)) * 180/pi ~ DIST + I(DIST^2)
Model 2: asin(sqrt(LAP)) * 180/pi ~ DIST
  Res.Df     RSS Df Sum of Sq      F  Pr(>F)
1     14 237.222
2     15 294.500 -1   -57.277 3.3803 0.08729 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> mytilus.lm3 <- lm(asin(sqrt(LAP)) * 180/pi ~ DIST +
+     I(DIST^2) + I(DIST^3), mytilus)
> anova(mytilus.lm3, mytilus.lm2)
Analysis of Variance Table

Model 1: asin(sqrt(LAP)) * 180/pi ~ DIST + I(DIST^2) + I(DIST^3)
Model 2: asin(sqrt(LAP)) * 180/pi ~ DIST + I(DIST^2)
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     13 152.115
2     14 237.222 -1   -85.108 7.2734 0.0183 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - a cubic model fits the data significantly better than a quadratic model ($P = 0.018$), the latter of which does not fit significantly better than a linear model ($P = 0.09$).

5. Estimate the model parameters[m] for the cubic model so as to establish the descriptive or predictive model.

---

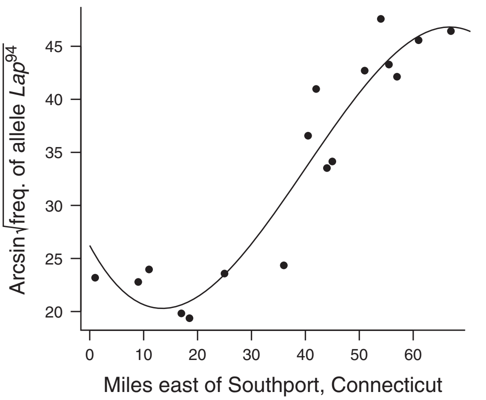
[m] Due to the extreme multicollinearity problems (*dist* must be correlated to $dist^2$ and $dist^3$ etc), the parameter estimates are not stable, the standard errors are inflated and the individual parameter hypothesis tests are non informative. As with multiplicative multiple regression, this problem can be greatly alleviated by first centering the predictor variable. However, the value in doing so is limited as the resulting parameters (and associated confidence intervals) would then have to be back transformed into the original scales in order to construct a descriptive or predictive model (main uses of polynomial regression). Since the values of the estimated polynomial parameters do not have any biological meaning, standard errors and hypothesis tests of the parameter estimates should be ignored.

```
> summary(mytilus.lm3)
Call:
lm(formula = asin(sqrt(LAP)) * 180/pi ~ DIST + I(DIST^2) +
+    I(DIST^3), data = mytilus)

Residuals:
    Min      1Q  Median      3Q     Max
-6.1661 -2.1360 -0.3908  1.9016  6.0079

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 26.2232524  3.4126910   7.684 3.47e-06 ***
DIST        -0.9440845  0.4220118  -2.237  0.04343 *
I(DIST^2)    0.0421452  0.0138001   3.054  0.00923 **
I(DIST^3)   -0.0003502  0.0001299  -2.697  0.01830 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.421 on 13 degrees of freedom
Multiple R-squared: 0.9112,     Adjusted R-squared: 0.8907
F-statistic: 44.46 on 3 and 13 DF,  p-value: 4.268e-07
```

Conclusions - there was a significant cubic relationship between the frequency of the Lat^{94} allele in *Mytilus edulis* and distance from Southport ($P < 0.001, r^2 = 0.911$: $arcsin\sqrt{Lat} = 26.2233 - 0.9441 dist + 0.0421 dist^2 - 0.0003 dist^3$).

Step 4 (Key 9.11) - construct a summary figure to summarize the illustrate the proposed nature of the relationship.

```
> plot(asin(sqrt(LAP)) * 180/pi ~ DIST, data = mytilus,
+     pch = 16, axes = F, xlab = "", ylab = "")
> axis(1, cex.axis = 0.8)
> mtext(text = expression(paste("Miles east of Southport,
+      Connecticut")), side = 1, line = 3)
> axis(2, las = 1)
> mtext(text = expression(paste("Arcsin ",
+      sqrt(paste("freq. of allele ", italic("Lap"))^{
+     94
+ }))), side = 2, line = 3)
> x <- seq(0, 80, l = 1000)
> points(x, predict(mytilus.lm3, data.frame(DIST = x)),
+     type = "l")
> box(bty = "l")
```



Example 9G: Nonlinear regression

Peake and Quinn (1993) investigated the nature of species-area relationships for invertebrates inhabiting inter-tidal mussel clumps (Box 6.11, Quinn and Keough (2002)).

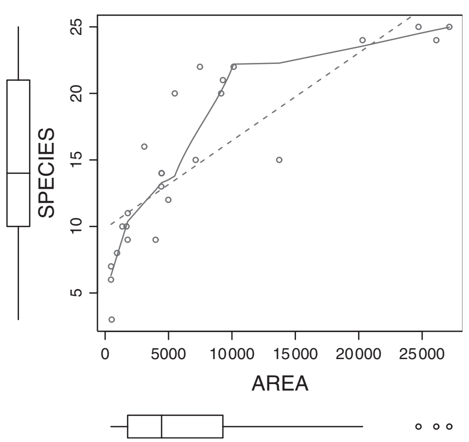
Step 1 - Import (section 2.3) the peake data set

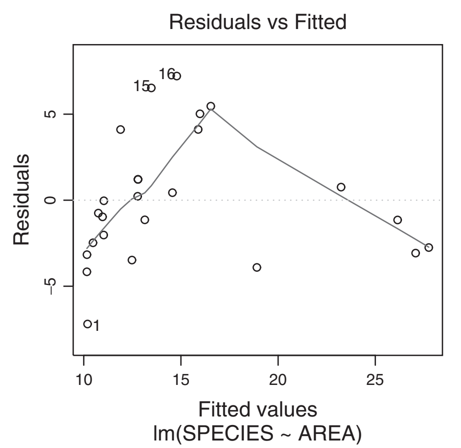
```
> peake <- read.table("peake.csv", header = T, sep = ",")
```

Step 2 (Key 8.2a) - confirm that simple linear regression does not adequately describe the relationship between the number of species and mussel clump area by examining a scatterplot and residual plot.

```
> library(car)
> scatterplot(SPECIES ~ AREA,
+     data = peake)
```

```
> plot(lm(SPECIES ~ AREA,
+     data = peake), which = 1)
```





Conclusions - the scatterplot smoother suggests a non-linear relationship and the persisting pattern in the residuals further suggests that the linear model is inadequate for explaining the response variable. Although this could probably be corrected by transforming the scale of the mussel clump area variable, in this case, theory suggests that species-area relationships might be more appropriately modeled with a power function.

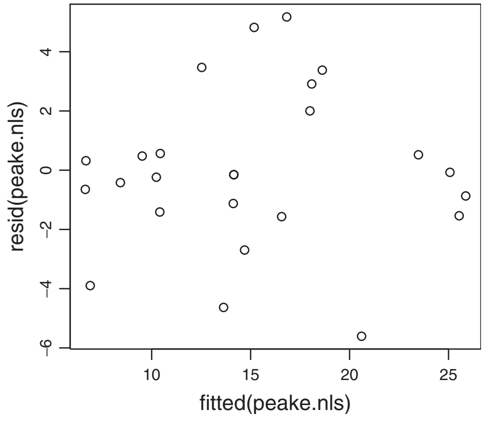
Step 3 (Key 9.7) - fit a nonlinear regression (power) model.

1. Fit the model (a power model would seem appropriate, see also Table 9.1)

```
> peake.nls <- nls(SPECIES ~ alpha * AREA^beta,
    start = list(alpha = 0.1,
+     beta = 1), peake)
```

2. Examine the diagnostics

```
> plot(resid(peake.nls) ~ fitted(peake.nls))
```



Conclusions - no persisting pattern suggesting that the fitted power model is appropriate for modeling these data. Additionally, there is no "wedge" pattern suggesting that the homogeneity of variance assumption is satisfied.

3. Examine the estimated nonlinear model parameters

```
> summary(peake.nls)
Formula: SPECIES ~ alpha * AREA^beta

Parameters:
      Estimate Std. Error t value Pr(>|t|)
alpha   0.8584     0.2769   3.100  0.00505 **
beta    0.3336     0.0350   9.532 1.87e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.733 on 23 degrees of freedom

Number of iterations to convergence: 17
Achieved convergence tolerance: 1.043e-06
```

Step 4 (Key 9.8a) - Examine the fit of the nonlinear model (compared to a linear model).

```
> AIC(peake.nls, k=log(nrow(peake))) #BIC
[1] 128.7878
> AIC(peake.nls) #AIC
[1] 125.1312
> deviance(peake.nls)/df.residual(peake.nls) #MSresid
[1] 7.468933
```

```
> peake.lm<-lm(SPECIES~AREA, data=peake) #linear fit
> AIC(peake.lm, k=log(nrow(peake))) #lm BIC
[1] 144.7322
> AIC(peake.lm) #lm AIC
[1] 141.0756
> deviance(peake.lm)/df.residual(peake.lm) #lm MSresid
[1] 14.13324
```

Conclusions - all fit criterion concur that the nonlinear power model is a better fit to the data than the linear model.

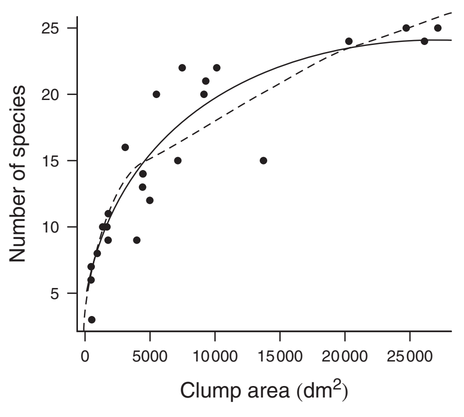
Step 5 (Key 9.8a) - Arguably, these data would be better modelled by a asymptotic relationship. Fit such a relationship.

```
> peake.nls1 <- nls(SPECIES~SSasymp(AREA,a,b,c),peake)
> summary(peake.nls1)
Formula: SPECIES ~ SSasymp(AREA, a, b, c)

Parameters:
  Estimate Std. Error t value Pr(>|t|)
a  24.4114     1.6644  14.667 7.71e-13 ***
b   4.9563     1.4244   3.479  0.00213 **
c  -8.8138     0.2482 -35.512  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.719 on 22 degrees of freedom

Number of iterations to convergence: 0
Achieved convergence tolerance: 7.128e-07
> AIC(peake.nls1) #AIC
[1] 125.7644
> deviance(peake.nls1)/df.residual(peake.nls1) #MSresid
[1] 7.393005
> anova(peake.nls,peake.nls1)

Analysis of Variance Table

Model 1: SPECIES ~ alpha * AREA^beta
Model 2: SPECIES ~ SSasymp(AREA, a, b, c)
  Res.Df Res.Sum Sq Df  Sum Sq F value Pr(>F)
1     23    171.785
2     22    162.646  1   9.139  1.2362 0.2782
```

Conclusions - the asymptotic trend does not fit the data significantly better than the exponential trend.

Step 6 (Key 9.11) - summarize the nonlinear species-area relationship with a scatterplot and exponential (dashed line) and asymptotic (solid) line trends.

```
> plot(SPECIES ~ AREA, peake, pch = 16, axes = F, xlab = "",
+     ylab = "")
> axis(1, cex.axis = 0.8)
> mtext(text = expression(paste("Clump area ", (dm^2))),
+     side = 1, line = 3)
> axis(2, las = 1)
> mtext(text = "Number of species", side = 2, line = 3)
> box(bty = "l")
> x <- seq(0, 30000, l = 1000)
> points(x, predict(peake.nls, data.frame(AREA = x)),
+     type = "l", lty = 2)
> points(x, predict(nls(SPECIES ~ SSasymp(AREA, a,
+     b, c), peake), data.frame(AREA = x)), type = "l",
+     lty = 1)
> box(bty = "l")
```



Example 9H: Regression trees

Quinn and Keough (2002) used the Loyn (1987) data set (analysed in Example 9A on page 224) to demonstrate the use of regression trees for producing descriptive and predictive models (Quinn and Keough (2002) Box 6.9). Using the same data from Example 9A, we will illustrate the use of R to produce regression trees.

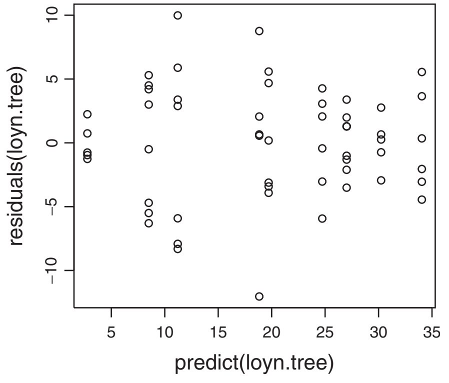
Step 1 (Key 9.13) - Perform binary recursive partitioning and construct the resulting regression tree.

```
> library(tree)
> loyn.tree <- tree(ABUND ~ AREA + YR.ISOL + DIST +
+     LDIST + GRAZE + ALT, data = loyn, mindev = 0)
```

Note that Quinn and Keough (2002) used log_{10} transformed data for some of the variables. Such transformations have no impact on the construction of the tree nodes or branches, however the split threshold values for transformed predictor variables will be on a log_{10} scale.

Step 2 (Key 9.13) - Examine the residuals for outlying, influential observations.

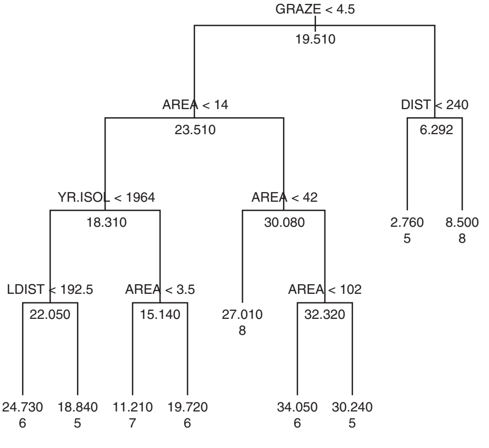
```
> plot(residuals(loyn.tree) ~ predict(loyn.tree))
```



Conclusions - There are an even spread of residuals with no obvious potentially influential observations (no outliers from the patterns within each branches predicted values).

Step 3 (Key 9.13) - Construct the regression tree.

```
> plot(loyn.tree, type = "uniform")
> text(loyn.tree, cex = 0.5, all = T)
> text(loyn.tree, lab = paste("n"), cex = 0.5, adj = c(0,
+      2), splits = F)
```

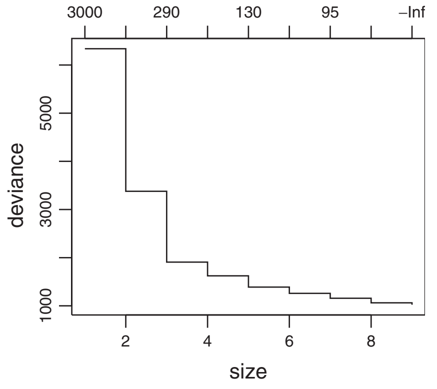


Conclusions - Grazing intensity was considered the most important single predictor of forest bird abundance. When grazing intensity was less than 4.5, patch area is important and when grazing intensity is greater than 4.5, the split in distance to nearest patch produced the greatest deviance (albeit very small suggesting that this entire branch is probably of little importance). Larger patch sizes continue to be split according to patch size suggesting that patch area is an important predictor of bird abundance. Smaller patches however are split by years since isolation and then by distance to the nearest patch and again patch area.

This is in broad agreement with the model selection outcomes demonstrated in examples 9C and 9D, although grazing intensity is of elevated importance in the regression tree. Patch area and years since isolation are considered important within the patches of lower grazing pressure.

Step 4 (Key 9.14) - Examine the cost-complexity measure.

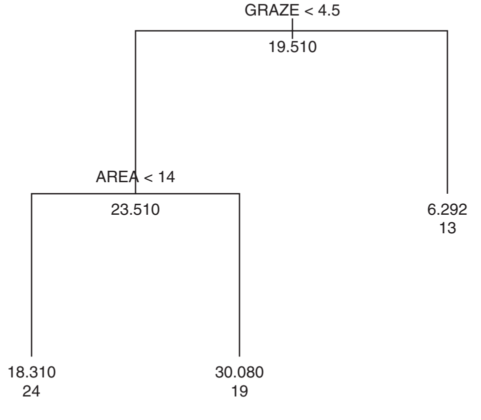
```
> plot(prune.tree(loyn.tree))
```



Conclusions - It is clear that the additional deviance (fit) achieved by adding more nodes beyond 3 is very marginal (cost-complexity curve begins to asymptote at this point). This suggests that the tree could potentially be pruned to just three terminal branches without great loss of predictive power too achieve a more genuine predictive model.

Step 5 (Key 9.14) - Prune the tree.

```
> loyn.tree.prune <- prune.tree(loyn.tree, best = 3)
> plot(loyn.tree.prune, type = "uniform")
> text(loyn.tree.prune, cex = 0.5, all = T)
> text(loyn.tree.prune, lab = paste("n"), cex = 0.5,
+      adj = c(0, 2), splits = F)
```



Conclusions - The pruned regression tree suggests a predictive model with two variables (grazing intensity and patch area).

10

Single factor classification (ANOVA)

Single factor classification (also known as analysis of variance of ANOVA) is used to investigate the effect of single factor comprising of two or more groups (treatment levels) from a completely randomized design (see Figure 10.1 & Figure 11.1a). Completely randomized refers to the absence of restrictions on the random allocation of experimental or sampling units to factor levels.

10.0.1 Fixed versus random factors

Fixed factors are factors whose levels represent the specific populations of interest. For example, a factor that comprises 'high', 'medium' and 'low' temperature treatments is a fixed factor – we are only interested in comparing those three populations. Conclusions about the effects of a fixed factor are restricted to the specific treatment levels investigated and for any subsequent experiments to be comparable, the same specific treatments of the factor would need to be used.

By contrast, **Random factors** are factors whose levels are randomly chosen from all the possible levels of populations and are used as random representatives of the populations. For example, five random temperature treatments could be used to represent a full spectrum of temperature treatments. In this case, conclusions are extrapolated to all the possible treatment (temperature) levels and for subsequent experiments, a new random set of treatments of the factor would be selected. Other

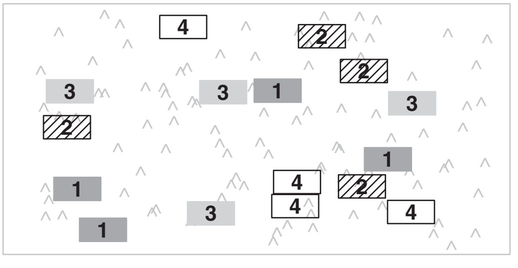


Fig 10.1 *A fictitious spatial depiction of sampling units arranged randomly and randomly assigned to one of four treatment levels (n = 4 for each treatment level).*

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

common examples of random factors include sites and subjects - factors for which we are attempting to generalize over. Furthermore, the nature of random factors means that we have no indication of how a new level of that factor (such as another subject or site) are likely to respond and thus it is not possible to predict new observations from random factors.

These differences between fixed and random factors are reflected in the way their respective null hypotheses are formulated and interpreted. Whilst fixed factors contrast the effects of the different levels of the factor, random factors are modelled as the amount of additional variability they introduce.

10.1 Null hypotheses

Fixed factor

A single fixed factor ANOVA tests the H_0 that there are no differences between the population group means

$$H_0 : \mu_1 = \mu_2 = \ldots = \mu_i = \mu \qquad \text{(the population group means are all equal)}$$

That is, that the mean of population 1 is equal to that of population 2 and so on, and thus all population means are equal to an overall mean. If the effect of the i^{th} group is the difference between the i^{th} group mean and the overall mean ($\alpha_i = \mu_i - \mu$) then the H_0 can alternatively be written as:

$$H_0 : \alpha_1 = \alpha_2 = \ldots = \alpha_i = 0 \qquad \text{(the effect of each group equals zero)}$$

If one or more of the α_i are different from zero (the response mean for this treatment differs from the overall response mean), the null hypothesis is not true indicating that the treatment does affect the response variable.

Random factor

The H_0 for a random factor is that the variance between all possible groups equals zero:

$$H_0 : \sigma^2_\alpha = 0 \qquad \text{(added variance due to this factor equals zero)}$$

10.2 Linear model

The linear model for single factor classification is similar to that of multiple linear regression[a]. There is a separate parameter for each level (group) of the factor and a constant parameter that estimates the overall mean of the response variable:

$$y_{ij} = \mu + \beta_1(level_1)_{ij} + \beta_2(level_2)_{ij} + \ldots + \varepsilon_{ij}$$

[a] Indeed, if the model is fitted with the `lm()` *function* rather than the more specific `aov()` *function*, parameters associated with each level of the treatment are estimated and tested.

where β_1 and β_2 respectively represent the effects of level 1 and 2 on the mean response. When these individual effects are combined into a single term, the linear effects model for single factor classification becomes:

$$y_{ij} = \mu + \alpha_i + \varepsilon_{ij}$$

Term	Fixed/random	Description	Null hypothesis
α_i	fixed	the effect of the i^{th} group	$\alpha_i = 0$ (no effect of factor A)
	random	random variable	$\sigma^2_\alpha = 0$ (variances between all possible levels of A are equal)

Note that whilst the null hypotheses for fixed and random factors are different (fixed: population group means all equal, random: variances between populations all equal zero, the linear model fitted for fixed and random factors in single factor ANOVA models is identical. For more complex multifactor ANOVA models however, the distinction between fixed and random factors has important consequences for statistical models and null hypotheses.

10.3 Analysis of variance

When the null hypothesis is true (and the populations are identical), the amount of variation among observations within groups should be similar to the amount of variation in observations between groups. However, when the null hypothesis is false, the amount of variation among observations might be expected to be less than the amount of variation within groups. Analysis of variance, or ANOVA, partitions the total variance in the response (dependent) variable into a component of the variance that is explained by combinations of one or more categorical predictor variables (called factors) and a component of the variance that cannot be explained (residual), see Figure 10.2. In effect, these are the variances among observations between and within groups respectively. The variance ratio (F-ratio) from this partitioning can then be used to test the null hypothesis (H_0) that the population group or treatment means are all equal.

When the null hypothesis is true (and the test assumptions have not been violated), the ratio (F-ratio) of explained to unexplained variance follows a theoretical probability distribution (F-distribution, see Figure 10.2d). When the null hypothesis is true, and there is no affect of the treatment on the response variable, the ratio of explained variability to unexplained variability is expected to be ≤ 1[b].

Importantly, the denominator in an F-ratio calculation essentially represents what we would expect the numerator to be in the absence of a treatment effect. For simple analyses, identifying the what these expected values are straight forward (equivalent to the degree of within group variability). However, in more complex designs (particularly involving random factors and hierarchical treatment levels), the logical "groups" can be more difficult (and in some cases impossible) to identify. In such cases, nominating

[b] Since the denominator should represent the expected numerator in the absence of an effect.

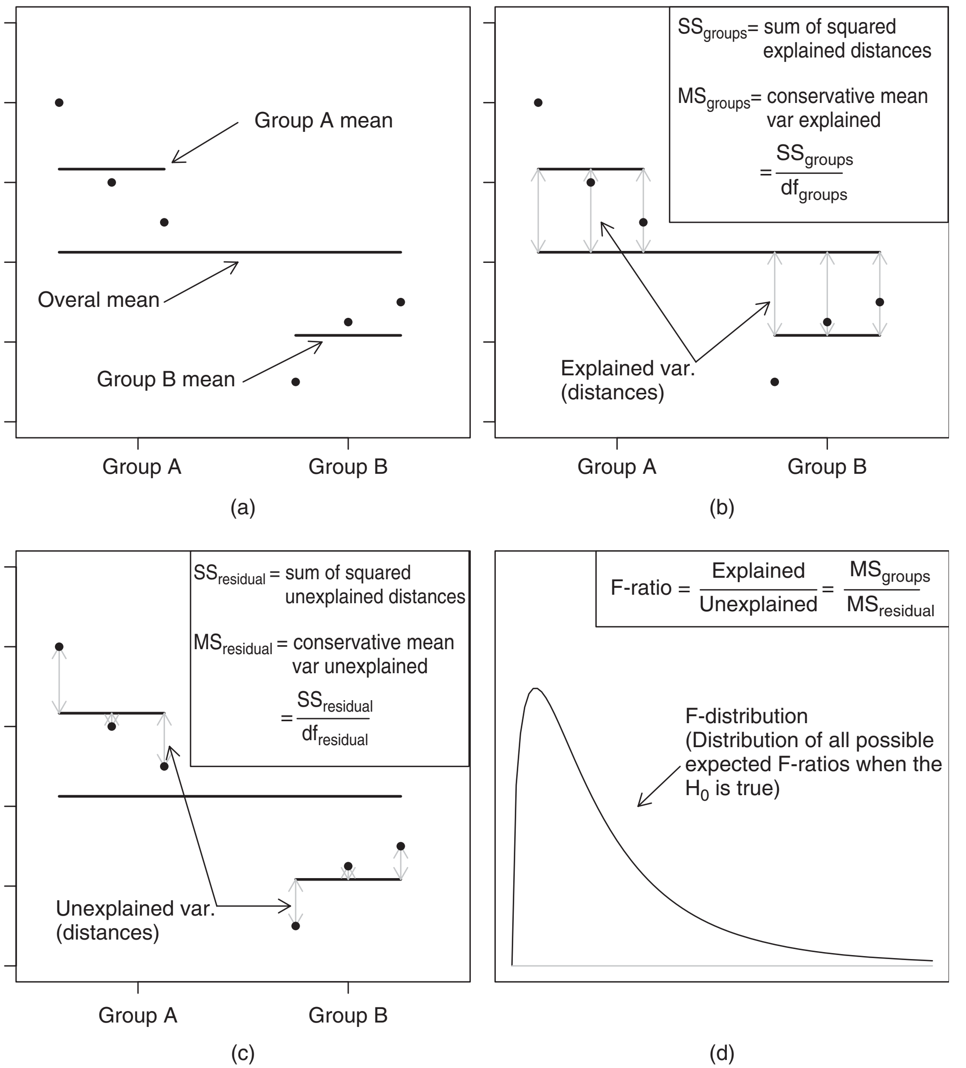


Fig 10.2 Fictitious data illustrating the partitioning of total variation into components explained by the groups (MS_{groups}) and unexplained ($MS_{residual}$) by the groups. The gray arrows in (b) depict the relative amounts explained by the groups. The proposed groupings generally explain why the first few points are higher on the y-axis than the last three points. The gray arrows in (c) depict the relative amounts unexplained (the residuals) by the groups. The proposed groupings fail to explain the differences within the first three points and within the last three points. The probability of collecting our sample, and thus generating the sample ratio of explained to unexplained variation (or one more extreme), when the null hypothesis is true (and population means are equal) is the area under the *F*-distribution (d) beyond our sample *F*-ratio.

Table 10.1 *F*-ratios and corresponding R syntax for single factor ANOVA designs (**A fixed or random**).

Factor	d.f.	MS	F-ratio
A	$a-1$	MS_A	$\frac{MS_A}{MS_{Resid}}$
Residual (=N′(A))	$(n-1)a$	MS_{Resid}	
	> anova(aov(DV A,dataset))		

the appropriate F-ratio deniminator for estimating an specific effect requires careful consideration (see chapters 11–14). Table 10.1 depicts the anatomy of the single factor ANOVA table and corresponding R syntax.

An F-ratio substantially greater than 1 suggests that the model relating the response variable to the categorical variable explains substantially more variability than is left unexplained. In turn, this implies that the linear model does represent the data well and that differences between observations can be explained largely by differences in treatment levels rather than purely the result of random variation. If the probability of getting the observed (sample) F-ratio or one more extreme is less than some predefined critical value (typically 5% or 0.05), we conclude that it is highly unlikely that the observed samples could have been collected from populations in which the treatment has no effect and therefore we would reject the null hypothesis.

10.4 Assumptions

An F-ratio from real data can only reliably relate to a theoretical F-distribution when the data conform to certain assumptions. Hypothesis testing for a single factor ANOVA model assumes that the residuals (and therefore the response variable for each of the treatment levels) are all:

(i) normally distributed - although ANOVA is robust to non-normality provided sample sizes and variances are equal. Boxplots should be used to explore normality, skewness, bimodality and outliers. Scale transformations are often useful.

(ii) equally varied - provided sample sizes are equal and the largest to smallest variance ratio does not exceed 3:1 (9:1 for sd), ANOVA is reasonably robust to this assumption, however, relationships between variance and mean and/or sample size are of particular concern as they elevate the Type I error rate. Boxplots and plots of means against variance should be used to explore the spread of values. Residual plots should reveal no patterns (see Figure 8.5). Since unequal variances are often the result of non-normality, transformations that improve normality will also improve variance homogeneity.

(iii) independent of one another - this assumption must be addressed at the design and collection stages and cannot be compensated for later[c].

Violations of these assumptions reduce the reliability of the analysis.

[c] Unless a model is used that specifically accounts for particular types of non-independent data, such as repeated measures ANOVA - see chapter 13.

10.5 Robust classification (ANOVA)

There are a number of alternatives to ANOVA that are more robust (less sensitive) to conditions of either non-normality or unequal variance. **Welch's test** adjusts the degrees of freedom to maintain test reliability in situations where populations are normally distributed but unequally varied. Alternatively, **Randomization tests** repeatedly shuffle the observations randomly, each time calculating a specific test statistic so as to build up a unique probability distribution for the test statistic for the collected data and thus make no assumptions about the distribution of the underlying population. Such tests do not assume observations were collected via random sampling, however they do assume that populations are equally varied.

Non-parametric (rank-based) tests such as the **Kruskal-Wallis test** use ranks of the observations to calculate test statistics rather than the actual observations and thus do not assume that the underlying populations are normally distributed. They test the null hypothesis that population medians are equal and are useful in situations where there are outliers. Although technically these tests still assume that the populations are equally varied, violations of this assumption apparently have little impact.

10.6 Tests of trends and means comparisons

Rejecting the null hypothesis that all of population group means are equal only indicates that at least one of the population group means differs from the others, it does not indicate which groups differ from which other groups. Consequently, researchers often wish to examine patterns of differences among groups. However, this requires multiple comparisons of group means and multiple comparisons lead to two statistical problems. First, multiple significance tests increase the probability of Type I errors (α, the probability of falsely rejecting H_0). If the decision criteria for any single hypothesis test is 0.05 (5%), then we are accepting that there is a 5% chance of committing a Type I error (falsely rejecting the null hypothesis). As a result, if many related hypothesis tests are conducted, then the overall Type I error rate (probability of making at least one Type I error) compounds to unacceptably high levels. For example, testing for differences between 5 groups requires ten pairwise comparisons. If the α for each test is 0.05 (5%), then the probability of at least one Type I error for the family of 10 tests is approximately 0.4 (40%). Second, the outcome of each test might not be independent (orthogonal). For example, if one test reveals that the population mean of group A is significantly different from the population mean of group B (A>B) and B>C then we already know the result of A vs. C.

Post-hoc unplanned pairwise comparisons compare all possible pairs of group means and are useful in an exploratory fashion to reveal differences between groups when it is not possible to justify any specific comparisons over other comparisons prior to the collection and analysis of data. There are a variety of procedures available to control the family-wise Type I error rate (e.g. Bonferroni and Tukey's test), thereby minimizing the probability of making Type I errors. However, these procedures reduce the power of each individual pairwise comparison (increase Type II error), and the reduction in power is directly related to the number of groups (and hence number of comparisons) being compared. For ordered factors (e.g. Temperature: 10, 15, 20, . . .), multiple pairwise comparisons are arguably less informative than an investigation of the overall trends across the set of factor levels.

Planned comparisons are specific comparisons that are usually planned during the design stage of the experiment. Most textbooks recommend that multiple comparisons can be made (each at $\alpha = 0.05$) provided each comparison is independent of (orthogonal to) other comparisons and that no more than $p - 1$ (where p is the number of groups) comparisons are made. Among all possible comparisons (both pairwise and combinational), only a select sub-set are performed, while other less meaningful (within the biological context of the investigation) combinations are ignored. Occasionally, the comparisons of greatest interest are not independent (non-orthogonal). In such circumstances, some statisticians recommend performing the each of the individual comparisons separately before applying a Dunn-Sidak p-value correction.

Specific comparisons are defined via a set of contrast coefficients associated with a linear combination of the treatment means (see section 7.3.1):

$$\overline{y}_1(C_1) + \overline{y}_2(C_2) + \ldots + \overline{y}_p(C_p)$$

where p is the number of groups in the factor. The contrast coefficients for a specific comparison must sum to zero and the groups being contrasted should have opposing signs. In addition to facilitating specific comparisons between individual groups, it is also possible to compare multiple groups to other groups or multiples and investigate polynomial trends. Table 10.2 provides example contrast coefficients for a number of commonly used planned comparison H_0 types. Note that polynomial trends assume that factor levels are ordered according to a natural gradient or progression (eg. low, medium, high) and that the factor levels are evenly spaced along this gradient. If you have reason to suspect that this is not the case, consider either weighting the

Table 10.2 Example contrast coefficients for specific comparisons and the first three order polynomials for a factor with four levels (groups).

H_0:	Group$_1$	Group$_2$	Group$_3$	Group$_4$
$\mu_1 = \mu_2$	1	−1	0	0
$(\mu_1 + \mu_2)/2 = \mu_3$[a]	.5	.5	−1	0
no linear trend	−3	−1	1	3
no quadratic trend	1	−1	−1	1
no cubic trend	−1	3	−3	1

[a]while alternatively, this planned contrast could have been defined as $1, 1, -2, 0$, yielding the same partitioning on $SS_{CONTRAST}$, its estimated parameter value would not reflect the value inferred by the null hypothesis.

contrast coefficients to better represent the increments between treatment levels[d], or else regression analysis (see chapter 8) as an alternative.

10.7 Power and sample size determination

Recall from section 6.5, that power (the probability of detecting an effect if an effect really exists) is proportional to the effect size, sample size and significance level (α) and inversely proportional to the background variability. It is convienient to think about the effect size as the absolute magnitude of the effect. When there are only two groups, the effect size is relatively straight forward to estimate (it is the expected difference between the means of two populations). However, when there are more than two groups, there are numerous ways in which this effect size can manifest. For example, in an investigation into the effect of temperature ('v.high', 'high', 'medium' and 'low') on the growth rate of seedlings, there are numerous ways that an effect size of (for example) 10 units above the expected background mean growth rate of 20 units could be distributed across the four groups (see Table 10.3). Consequently, effect size is expressed in terms of the expected variability both within and between the populations (groups). The smaller the degree of variability between groups, the more difficult it is to detect differences, or the greater the sample size required to detect differences. It is therefore important to anticipate the nature of between group patterns in conducting power analyses and sample size determinations.

Table 10.3 Fictitious illustration of the variety of ways that an effect size of 10 units could be distributed over four groups.

Possible trends		Between group variability
One group different	$\mu_V > \mu_H = \mu_M = \mu_L$	`var(c(30,20,20,20)) = 25.00`
Two groups different	$\mu_V = \mu_H > \mu_M = \mu_L$	`var(c(30,30,20,20)) = 33.33`
Equal increments	$\mu_V > \mu_H > \mu_M > \mu_L$	`var(seq(30,20,l=4)) = 18.52`
Other increments	$\mu_V > \mu_H = \mu_M > \mu_L$	`var(c(30,25,25,20)) = 16.67`

10.8 ANOVA in R

Single factor ANOVA models can be fitted with the either the `lm()` linear modelling *function* or the more specific `aov()` *function*, the latter of which provides a wrapper for the `lm()` *function* that redefines output for standard analysis of variance rather than

[d] For a linear trend, weighted coefficients can be calculated by providing numerical representations of each of the factor levels and then subtracting the mean of these levels from each numeric level.

parameter estimates. ANOVA tables for balanced, fixed factor designs can be viewed using either the `anova()` or `summary()`, the latter of which is used to accommodate planned contrasts with the `split=` *argument*.

10.9 Further reading

- Theory

  Doncaster, C. P., and A. J. H. Davey. (2007). *Analysis of Variance and Covariance. How to Choose and Construct Models for the Life Sciences*. Cambridge University Press, Cambridge.

  Fowler, J., L. Cohen, and P. Jarvis. (1998). *Practical statistics for field biology*. John Wiley & Sons, England.

  Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. 2 edition. John Wiley & Sons, New York.

  Manly, B. F. J. (1991). *Randomization and Monte Carlo methods in biology*. Chapman & Hall, London.

  Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

  Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

  Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

  Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

  Dalgaard, P. (2002). *Introductory Statistics with R*. Springer-Verlag, New York.

  Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

  Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

  Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS*, 4th edn. Springer-Verlag, New York.

  Wilcox, R. R. (2005). *Introduction to Robust Estimation and Hypothesis Testing*. Elsevier Academic Press.

10.10 Key for single factor classification (ANOVA)

1 a. Check parametric assumptions

- **Normality of the response variable at each level of the categorical variable - boxplots**

  ```
  > boxplot(DV ~ Factor, dataset)
  ```

  where `DV` *and* `Factor` *are response and factor variables respectively in the* `dataset` *data frame*

- **Homogeneity of variance - boxplots (as above) and scatterplot of mean vs variance**

```
> plot(tapply(dataset$DV, dataset$Factor, var),
+     tapply(dataset$DV, dataset$Factor, mean))
```

where `DV` *and* `Factor` *are response and factor variables respectively in the* `dataset` *data frame*

Parametric assumptions met . . . . Go to 2
b. Parametric assumptions NOT met . . . . Go to 5
2 a. ANOVA with specific comparisons or trends . . . . Go to 4
b. ANOVA without specific comparisons or trends . . . . Go to 3
3 a. Single fixed factor (model I) . . . . See Example 10A

```
> data.aov <- aov(DV ~ Factor, dataset)
> plot(data.aov)
> anova(data.aov)
```

if Reject H_0 - Significant difference between group means detected . . . . Go to 9
b. Single random factor (model II) . . . . See Example 10D

```
> anova(aov(DV ~ Factor, dataset))
```

if Reject H_0 - Significant difference between group means detected - calculate variance components

```
> library(nlme)
> data.lme <- lme(DV ~ 1, random = ~1 | Factor, data = dataset,
+     method = "ML")
> VarCorr(data.lme)
> data.lme <- lme(DV ~ 1, random = ~1 | Factor, data = dataset,
+     method = "REML")
> VarCorr(data.lme)
```

4 a. With planned comparisons of means . . . . See Example 10B

```
> contrasts(dataset$Factor) <- cbind(c(contrasts), c(contrasts),
+     ...)
> round(crossprod(contrasts(dataset$Factor)), 2)
> data.list <- list(Factor = list(lab = 1, ..), ..)
> data.aov <- aov(DV ~ Factor, data = dataset)
> plot(data.aov)
> summary(data.aov, split = data.list)
```

b. With planned polynomial trends . . . . See Example 10C

```
> contrasts(dataset$Factor) <- "contr.poly"
> data.list <- list(Factor = list(Linear = 1))
> data.aov <- aov(DV ~ Factor, data = dataset)
> plot(data.aov)
> summary(data.aov, split = data.list)
```

5 a. Attempt a scale transformation (see Table 3.2 for common transformation options) . . . . Go to 1
b. Transformations unsuccessful or inappropriate . . . . Go to 6

6 a. Underlying distribution of the response variable is normal but variances are unequal (Welch's test) . . . . See Example 10F

```
> oneway.test(DV ~ Factor, var.equal = F)
```

If Reject H_0 - Significant difference between group means detected . . . . Go to 9c
or consider GLM . . . . GLM chapter 17

b. Underlying distribution of the response variable is NOT normal . . . . Go to 7

7 a. Underlying distribution of the response variable and residuals is known . . . . GLM chapter 17

b. Underlying distribution of the response variable and residuals is NOT known . . . . Go to 8

8 a. Variances not wildly unequal, but outliers present (Kruskal-Wallis nonparametric test) . . . . See Example 10G

```
> kruskal.test(DV ~ Factor, var.equal = F)
```

If Reject H_0 - Significant difference between group means detected . . . . Go to 9cb/c

b. Variances not wildly unequal, random sampling not possible (Randomization test) . . . . See Example 10G

```
> library(boot)
> data.boot <- boot(dataset, stat, R = 999, sim = "parametric",
+     rand.gen = rand.gen)
> plot(data.boot)
> print(data.boot)
```

where `stat` *is the statistic to repeatedly calculate and* `rand.gen` *defines how the data are randomized.*

9 a. Parametric simultaneous multiple comparisons - Tukey's test . . See Example 10A

```
> library(multcomp)
> summary(glht(model, linfct = mcp(Factor = "Tukey")))
```

b. Non-parametric simultaneous multiple comparisons - Steel test . . . . See Example 10E

```
> library(npmc)
> data <- data.frame(var = dataset$DV, class = dataset$Factor)
> summary(npmc(data), type = "steel")
```

c. Multiple comparisons based on p-value adjustments . . . . See Example 10G

```
> library(multtest)
> mt.rawp2adjp(pvalues, proc = "SidakSD")
> p.adjust(pvalues, method = "holm")
```

where pvalues is a list of pvalues from each pairwise comparison and `'holm'` *and* `'SidakSD'` *are the names of the p-value adjustment procedures. For alternative procedures, see Table 10.4.*

The `p.adjust` function above can also be called from within other pairwise routines

Parametric pairwise tests

```
> pairwise.t.test(DV ~ Factor, pool.sd = F, p.adjust = "holm")
```

Non-parametric pairwise tests

```
> pairwise.wilcox.test(DV ~ Factor, p.adjust = "holm")
```

Table 10.4 Alternative p-value adjustments (`p.adjust`) for use with the `pairwise.wilcoxon.test` and `pairwise.t.test`.

Syntax	Correction	Description
`'bonferroni'`	Bonferroni single-step correction	p-values multiplied by number of comparisons to control the family-wise error rate
`'holm'`	sequential step-down Bonferroni correction	More powerful than Bonferroni to control the family-wise error rate
`'hochberg'`	Hochberg step-up correction	Reverse of Holm procedure and possibly more powerful to control the family-wise error rate
`'hommel'`	sequential Bonferroni correction	Reportedly more powerful than Hochberg procedure to control the family-wise error rate
`'BH'`	Benjamini & Hochberg step-up correction	Controls the false discovery rate
`'BY'`	Benjamini & Yekutieli step-up correction	Controls the false discovery rate
`'none'`	no correction	Uncorrected p-values
`'SidakSS'`[a]	Sidak single-step correction	More powerful modification of Bonferroni procedure
`'SidakSD'`[a]	Sidak step-down correction	More powerful modification of Bonferroni procedure

[a]only available via the `mt.rawp2adjp` function of the `multtest` package, see Example 10F.

10.11 Worked examples of real biological data sets

Example 10A: Single factor ANOVA with Tukey's test

Medley and Clements (1998) investigated the impact of zinc contamination (and other heavy metals) on the diversity of diatom species in the USA Rocky Mountains (from Box 8.1 of Quinn and Keough (2002)). The diversity of diatoms (number of species) and degree of zinc contamination (categorized as either of high, medium, low or natural background level) were recorded from between four and six sampling stations within each of six streams known to be polluted. These data were used to test the null hypothesis that there were no differences the diversity of diatoms between different zinc levels (H_0: $\mu_H = \mu_M = \mu_L = \mu_B = \mu$; $\alpha_i = 0$).

The linear effects model would be:

$$\begin{array}{ccccccc} y_{ij} & = & \mu & + & \alpha_i & + & \varepsilon_{ij} \\ \text{diatom species} & = & \text{overall} & + & \text{effect of zinc} & + & \text{error} \\ \text{diversity} & & \text{mean} & & \text{level} & & \end{array}$$

Step 1 - Import (section 2.3) the Medley and Clements (1998) data set

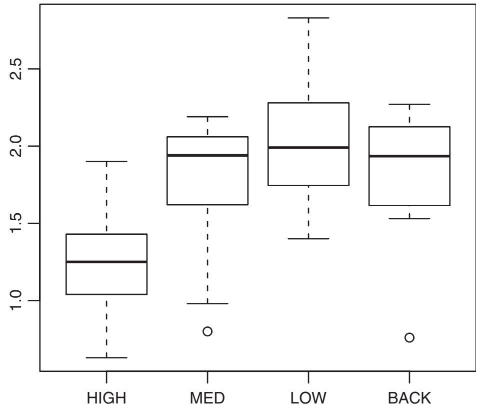
```
> medley <- read.table("medley.csv", header = T, sep = ",")
```

Step 2 - Reorganize the levels of the categorical factor into a more logical order (section 2.6.1)

```
> medley$ZINC <- factor(medley$ZINC, levels = c("HIGH", "MED",
+     "LOW", "BACK"), ordered = F)
```

Step 3 (Key 10.1) - Assess normality/homogeneity of variance using boxplot of species diversity against zinc group

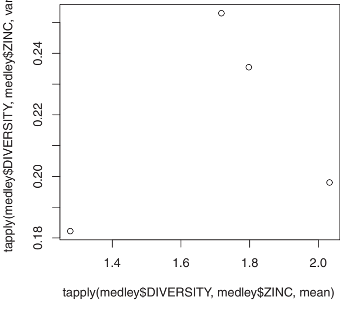
```
> boxplot(DIVERSITY ~ ZINC, medley)
```



Conclusions - no obvious violations of normality or homogeneity of variance (boxplots not asymmetrical and do not vary greatly in size)

Step 4 (Key 10.1) - Assess homogeneity of variance assumption with a table and/or plot of mean vs variance

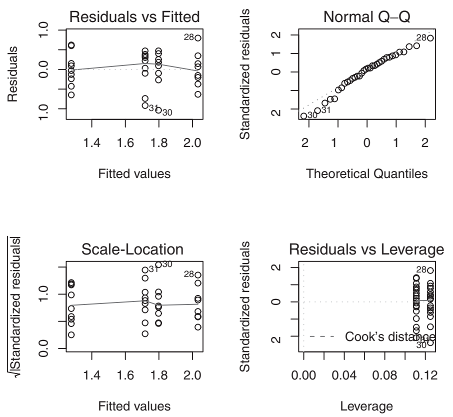
```
> plot(tapply(medley$DIVERSITY, medley$ZINC, mean),
+     tapply(medley$DIVERSITY, medley$ZINC, var))
```



Conclusions - no obvious relationship between group mean and variance

Step 5 (Key 10.3a) - Test H_0 that population group means are all equal - perform analysis of variance (fit the linear model) of species diversity versus zinc-level group and examine the diagnostics (residual plot)

```
> medley.aov <- aov(DIVERSITY ~ ZINC, medley)
> plot(medley.aov)
```



Conclusions - no obvious violations of normality or homogeneity of variance (no obvious wedge shape in residuals, normal Q-Q plot approximately linear). Note that Cook's D values meaningless in ANOVA.

Step 6 (Key 10.3a) - Examine the ANOVA table.

```
> anova(medley.aov)
Analysis of Variance Table

Response: DIVERSITY
          Df Sum Sq Mean Sq F value  Pr(>F)
ZINC       3 2.5666  0.8555  3.9387 0.01756 *
Residuals 30 6.5164  0.2172
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - reject H_0 that population group means are equal, ZINC was found to have a significant impact on the DIVERSITY of diatoms ($F_{3,30} = 3.939, P = 0.018$).

Step 7 (Key 10.9a) - Perform post-hoc Tukey's test to investigate pairwise mean differences between all groups

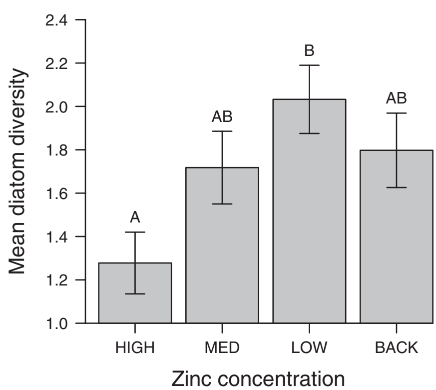
```
> library(multcomp)
> summary(glht(medley.aov, linfct = mcp(ZINC = "Tukey")))
         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = DIVERSITY ~ ZINC, data = medley)

Linear Hypotheses:
                 Estimate Std. Error t value Pr(>|t|)
MED - HIGH == 0   0.44000    0.21970   2.003   0.2093
LOW - HIGH == 0   0.75472    0.22647   3.333   0.0114 *
BACK - HIGH == 0  0.51972    0.22647   2.295   0.1219
LOW - MED == 0    0.31472    0.22647   1.390   0.5152
BACK - MED == 0   0.07972    0.22647   0.352   0.9847
BACK - LOW == 0  -0.23500    0.23303  -1.008   0.7457
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

Conclusions - diatom species diversity is significantly higher in low zinc sites than high zinc sites ($t_{15} = 3.333, P = 0.011$). No other H_0 rejected. Note, the Tukey's adjusted P-values are based on robust procedures that were not available to Quinn and Keough (2002). The more recent Tukey's test makes use of randomization procedures and thus the exact P-values differ from run to run.

Step 8 - Summarize findings of global ANOVA and post-hoc Tukey's test with a bargraph (see also section 5.9.4)

```
> library(biology)
> Mbargraph(medley$DIVERSITY, medley$ZINC, symbols = c("A", "AB",
+     "B", "AB"), ylab = "Mean diatom diversity",
+     xlab = "Zinc concentration")
```



Example 10B: Single factor ANOVA with planned comparisons

Keough and Raimondi (1995) examined the effects of four biofilm types (SL: sterile unfilmed substrate, NL: netted laboratory biofilms, UL: unnetted laboratory biofilms and F: netted field biofilms) on the recruitment of serpulid larvae (from Box8.2 and Box8.4 of Quinn and Keough, 2002). Substrates treated with one of the four biofilm types were left in shallow marine waters for one week after which the number of newly recruited serpulid worms were counted. These data were used to test the null hypothesis that there were no differences in serpulid numbers between the different biofilms (H_0: $\mu_{SL} = \mu_{NL} = \mu_{UL} = \mu_{SL} = \mu_F = \mu; \alpha_i = 0$). The linear effects model would be:

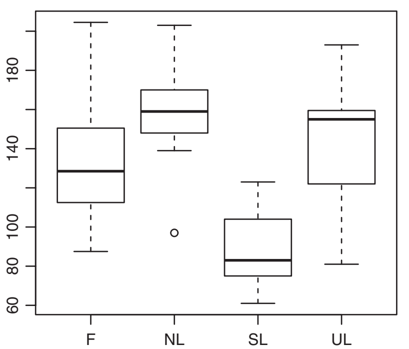
y_{ij}	$=$	μ	$+$	α_i	$+$	ε_{ij}
serpulid number	=	overall mean	+	effect of biofilm type	+	error

Step 1 - Import (section 2.3) the Keough and Raimondi (1995) data set

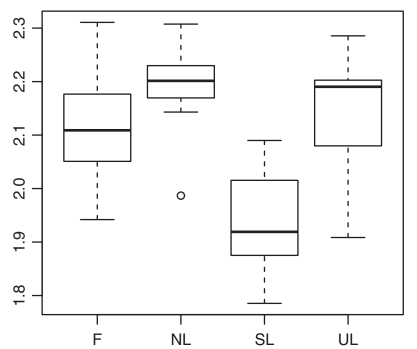
```
> keough <- read.table("keough.csv", header = T, sep = ",")
```

Step 2 (Keys 10.1 & 10.5) - Check the assumptions and scale data if appropriate

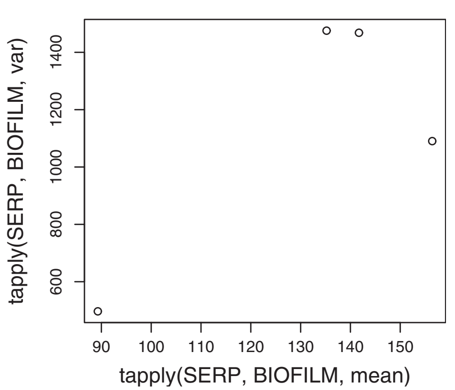
```
> boxplot(SERP ~ BIOFILM,
+     data = keough)
```



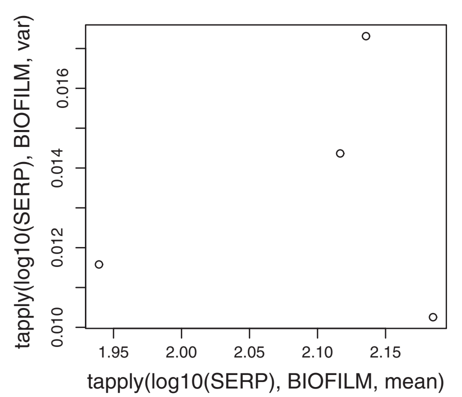

```
> boxplot(log10(SERP) ~ BIOFILM,
+     data = keough)
```




```
> with(keough, plot(tapply(SERP,
+     BIOFILM, mean),
+     tapply(SERP, BIOFILM,
+     var)))
```




```
> with(keough,
+     plot(tapply(log10(SERP),
+     BIOFILM, mean),
+     tapply(log10(SERP),
+     BIOFILM, var)))
```



Conclusions - some evidence of a relationship between population mean and population variance from untransformed data, $\log_{10}$ transformed data meets assumptions better, therefore transformation appropriate.

In addition to examining the overall effect of BIOFILM treatments on the number of newly recruited serpulid worms, Keough and Raimondi (1995) were interested in examining a number of other specific null hypotheses. In particular, whether recruitment was effected by the presence of netting in laboratory biofilms (NL vs UL), whether recruitment differed between field and laboratory biofilms (F vs (NL&UL) and finally whether recruitment differed between unfilmed and filmed treatments (SL vs (F&NL&UL)).

There specific null hypotheses and corresponding contrast coefficients are (Note, technically, we should not define contrasts with values greater than 1. However, in this case, as we are not going to examine the estimated regression parameters, the magnitude of the contrast coefficients will have no impact on the analyses.):

H_0:	F	NL	SL	UL
$\mu_{NL} = \mu_{UL}$	0	1	0	-1
$\mu_F = (\mu_{NL} + \mu_{UL})/2$	2	-1	0	-1
$\mu_{SL} = (\mu_F + \mu_{NL} + \mu_{UL})/3$	-1	-1	3	-1

Step 3 (Key 10.4a) - Define a list of contrasts for the following planned comparisons: NL vs UL, F vs the average of NL and UL, and SL vs the average of F, NL and UL.

```
> contrasts(keough$BIOFILM) <- cbind(c(0, 1, 0, -1), c(2, -1, 0,
+     -1), c(-1, -1, 3, -1))
```

Step 4 (Key 10.4a) - Confirm that defined contrasts are orthogonal.

```
> round(crossprod(contrasts(keough$BIOFILM)), 2)
     [,1] [,2] [,3]
[1,]    2    0    0
[2,]    0    6    0
[3,]    0    0   12
```

Conclusions - all defined planned contrasts are orthogonal (values above or below the cross-product matrix diagonal are all be zero).

Step 5 (Key 10.4a) - Define contrast labels. These are labels to represent each of the defined planned comparisons in the ANOVA table

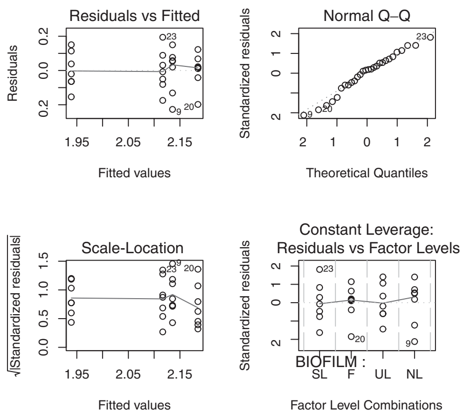
```
> keough.list <- list(BIOFILM = list('NL vs UL' = 1,
+     'F vs (NL&UL)' = 2, 'SL vs (F&NL&UL)' = 3))
```

Step 6 (Key 10.4a cont.) - Fit the linear model to test the null hypothesis that the population group means are all equal as well as the specific null hypotheses that the population means of treatments SL and F are equal, SL and the average of NL and F are equal, and UL and the average of SL, NL and F are equal.

```
> keough.aov <- aov(log10(SERP) ~ BIOFILM, data = keough)
```

Step 7 (Key 10.4a cont.) - Check the diagnostic plots to confirm assumptions are met

```
> plot(keough.aov)
```



Conclusions - no obvious violations of normality or homogeneity of variance (no obvious wedge shape in residuals, normal Q-Q plot approximately linear), Ignore Cook's D values for ANOVA.

Step 8 (Key 10.4a cont.) - Examine the ANOVA table

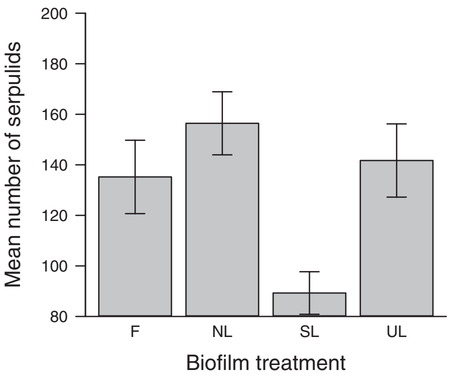
```
> summary(keough.aov, split = keough.list)
                           Df  Sum Sq Mean Sq F value    Pr(>F)
BIOFILM                     3 0.24103 0.08034  6.0058 0.0033386 **
  BIOFILM: NL vs UL         1 0.00850 0.00850  0.6352 0.4332635
  BIOFILM: F vs (NL&UL)     1 0.00888 0.00888  0.6635 0.4233267
  BIOFILM: SL vs (F&NL&UL)  1 0.22366 0.22366 16.7188 0.0004208 ***
Residuals                  24 0.32106 0.01338
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Biofilm treatments were found to have a significant affect on the mean $\log_{10}$ number of serpulid recruits ($F_{3,24} = 6.0058, P = 0.003$). The presence of a net (NL) over the substrate was not found to alter the mean $\log_{10}$ serpulid recruits compared to a surface without (UL) a net ($F_{1,24} = 0.6352, P = 0.4332$). Field biofilms (F) were not found to have different mean $\log_{10}$ serpulid recruits than the laboratory (NL, UL) biofilms ($F_{1,24} = 0.6635, P = 0.4233$). Unfilmed treatments were found to have significantly lower mean $\log_{10}$ serpulid recruits than treatments with biofilms ($F_{1,24} = 16.719, P < 0.001$).

Step 9 - Summarize findings with a bargraph (see section 5.9.4)

```
> means <- with(keough, tapply(SERP, BIOFILM, mean, na.rm = T))
> sds <- with(keough, tapply(SERP, BIOFILM, sd, na.rm = T))
> n <- with(keough, tapply(SERP, BIOFILM, length))
> ses <- sds/sqrt(n)
> ys <- pretty(c(means - ses, means + (2 * ses)))
> xs <- barplot(means, beside = T, axes = F, ann = F,
+      ylim = c(min(ys), max(ys)), xpd = F)
> arrows(xs, means + ses, xs, means - ses, ang = 90, length = 0.1,
+     code = 3)
```

```
> axis(2, las = 1)
> mtext(2, text = "Mean number of serpulids", line = 3, cex = 1.5)
> mtext(1, text = "Biofilm treatment", line = 3, cex = 1.5)
> box(bty = "l")
```



Example 10C: Single factor ANOVA with planned polynomial trends

As an illustration of polynomial trends, Quinn and Keough (2002) suggested a hypothetical situation in which Keough and Raimondi (1995) might have also included an examination of the linear change in settlement across the four treatments (SL, NL, UL & F).

Step 1 - Import the Keough and Raimondi (1995) data set, see Example 10B.

```
> keough <- read.table("keough.csv", header = T, sep = ",")
```

Step 2 (see section 2.6.1) - Reorder the factor levels into a logical order in preparation of the polynomial trends - so that not in alphabetical order

```
> keough$BIOFILM <- factor(keough$BIOFILM, levels = c("SL", "NL",
+     "UL", "F"))
```

Step 3 (Key 10.4b) - Define the polynomial contrast coefficients. These will be automatically generated and orthogonal.

```
> contrasts(keough$BIOFILM) <- "contr.poly"
```

Step 4 (Key 10.4b) - Define the polynomial contrast labels

```
> keough.list <- list(BIOFILM = list(Linear = 1, Quadratic = 2,
+     Cubic = 3))
```

Step 5 (Key 10.4b) - Fit the ANOVA model and the first, second and third order polynomial trends

```
> keough.aov <- aov(log10(SERP) ~ BIOFILM, data = keough)
```

Step 6 (Key 10.4b) - Examine the ANOVA table including the first three polynomial trends

```
> summary(keough.aov, split = keough.list)
                     Df  Sum Sq Mean Sq F value   Pr(>F)
BIOFILM               3 0.24103 0.08034  6.0058 0.003339 **
  BIOFILM: Linear     1 0.08155 0.08155  6.0961 0.021054 *
  BIOFILM: Quadratic  1 0.12248 0.12248  9.1555 0.005836 **
  BIOFILM: Cubic      1 0.03700 0.03700  2.7660 0.109294
Residuals            24 0.32106 0.01338
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - We would reject the null hypothesis of no quadratic trend over and above a linear trend ($F_{1,24} = 9.156, P = 0.006$), suggesting that there is a significant quadratic trend in mean $\log_{10}$ number of serpulid recruits across the ordered BIOFILM treatments (SL, NL, UL, F). Whilst this is a statistically significant outcome, it does not necessarily infer biological significance.

Example 10D: Single random factor ANOVA and variance components

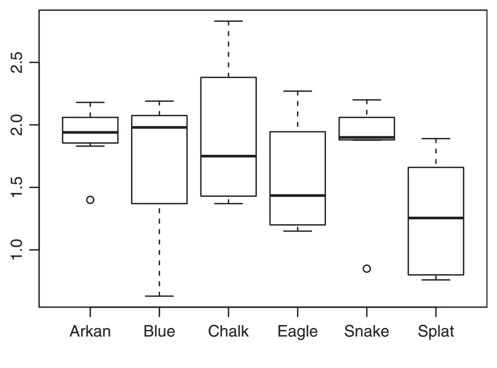
Following on from Example 10A, Medley and Clements (1998) may also have been interested in whether diatom diversity differed across Rocky Mountain streams (Box8.1 from Quinn and Keough, 2002). Hence, streams could be treated as a random factor in testing the null hypothesis that there was no added variance in diatom diversity due to streams.

Step 1 - Import (section 2.3) the Medley and Clements (1998) data set

```
> medley <- read.table("medley.csv", header = T, sep = ",")
```

Step 2 (Key 10.1a & 10.1b) - Assess normality/homogeneity of variance using boxplot of species diversity against stream

```
> boxplot(DIVERSITY ~ STREAM, medley)
```



Conclusions - although not ideal, there is no evidence that population diatom diversity is consistently non-normally distributed and drastically unequally varied. Note that small boxplots are accompanied by outliers suggestive of potentially greater variance. Consequently, perform ANOVA and rely on general robustness of the test.

Step 3 (Key 10.3a) - Test H_0 that there is no added variation in diatom diversity due to stream - perform analysis of variance (fit the linear model) of species diversity versus stream and examine the ANOVA table.

```
> medley.aov <- aov(DIVERSITY ~ STREAM, medley)
> anova(medley.aov)
Analysis of Variance Table

Response: DIVERSITY
          Df Sum Sq Mean Sq F value Pr(>F)
STREAM     5 1.8278  0.3656  1.4108 0.2508
Residuals 28 7.2552  0.2591
```

Conclusions - do not reject the null hypothesis that there is no added variance in diatom diversity due to streams.

Step 4 (Key 10.3a) - Calculate ML and REML estimates of variance components (random factor and residuals).

```
> library(nlme)
> print(VarCorr(lme(DIVERSITY ~ 1, random = ~1 | STREAM,
+     method = "ML", data = medley)))
STREAM = pdLogChol(1)
            Variance    StdDev
(Intercept) 0.009927963 0.09963916
Residual    0.257182572 0.50713171
> print(VarCorr(lme(DIVERSITY ~ 1, random = ~1 | STREAM,
+     method = "REML", data = medley)))
STREAM = pdLogChol(1)
            Variance   StdDev
(Intercept) 0.02053683 0.1433068
Residual    0.25755732 0.5075011
```

Conclusions - Most of the variance in diatom diversity is due to differences between sampling stations within the streams (ML: 0.2571, REML: 0.2576), very little variance is added due to differences between streams (ML: 0.0099, REML: 0.0205)

Example 10E: Kruskal-Wallis test with non-parametric post-hoc test

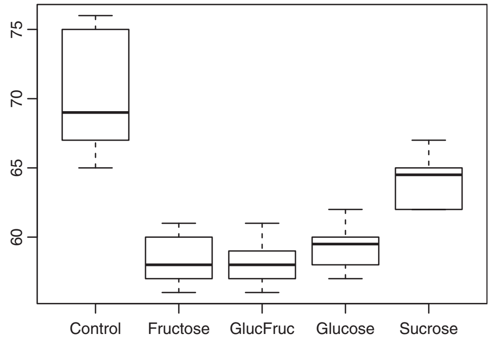
Sokal and Rohlf (1997) present an unpublished data set (W. Purves) in which the effect of different sugar treatments (Control, 2% glucose added, 2% fructose added, 1% glucose and 1% fructose added, and 2% sucrose added) on pea length was investigated (from Box 13.6 of Sokal and Rohlf, 1997).

Step 1 - Import the Purves (unpublished) data set

```
> purves <- read.table("purves.csv", header = T, sep = ",")
```

Step 2 (Keys 10.1a & 10.5) - Check the assumptions of normality and equal variance

```
> boxplot(LENGTH ~ TREAT, data = purves)
```



Conclusions - strong evidence of unequal variance. Note that this data set would probably be better suited to a Welch's test, however, for the purpose of providing worked examples that are consistent with popular biometry texts, a Kruskal-Wallis test will be demonstrated.

Step 3 (Key 10.8) - Perform non-parametric Kruskal-Wallis test.

```
> kruskal.test(LENGTH ~ TREAT, data = purves)
        Kruskal-Wallis rank sum test

data:  LENGTH by TREAT
Kruskal-Wallis chi-squared = 38.4368, df = 4, p-value = 9.105e-08
```

Conclusions - reject null hypothesis, sugar treatment has a significant affect on the growth of pea sections.

Step 4 (Key 10.8) - Perform non-parametric post-hoc test.

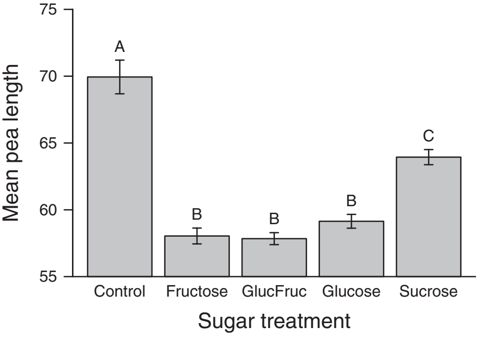
```
> library(npmc)
> dat <- data.frame(var = purves$LENGTH, class = purves$TREAT)
> summary(npmc(dat), type = "Steel")

$'Data-structure'
         group.index class.level nobs
Control            1     Control   10
Fructose           2    Fructose   10
GlucFruc           3    GlucFruc   10
Glucose            4     Glucose   10
Sucrose            5     Sucrose   10

$'Results of the multiple Steel-Test'
  cmp effect   lower.cl  upper.cl   p.value.1s  p.value.2s
1  1-2  0.000 -0.3599019 0.3599019 1.0000000000 0.001470977
2  1-3  0.000 -0.3596288 0.3596288 1.0000000000 0.001298745
3  1-4  0.000 -0.3600384 0.3600384 1.0000000000 0.001041309
4  1-5  0.050 -0.3081226 0.4081226 1.0000000000 0.005696086
5  2-3  0.495  0.1422692 0.8477308 0.9943192409 1.000000000
```

```
6  2-4  0.670  0.3133899 1.0266101 0.5005921659 0.713955365
7  2-5  1.000  0.6405079 1.3594921 0.0005691443 0.001327216
8  3-4  0.730  0.3746322 1.0853678 0.2525087694 0.407630138
9  3-5  1.000  0.6407814 1.3592186 0.0008494360 0.001372916
10 4-5  0.985  0.6261920 1.3438080 0.0010278350 0.001889472
```

Conclusions - The pea sections treated with sugar were significantly shorter than the controls and sections treated with sucrose were significantly longer than sections treated with either glucose, fructose or a mixture of glucose and fructose.

Step 5 - Summarize findings with a bargraph

```
> means <- with(purves, tapply(LENGTH, TREAT, mean, na.rm = T))
> sds <- with(purves, tapply(LENGTH, TREAT, sd, na.rm = T))
> n <- with(purves, tapply(LENGTH, TREAT, length))
> ses <- sds/sqrt(n)
> ys <- pretty(c(means - ses, means + (2 * ses)))
> xs <- barplot(means, beside = T, axes = F, ann = F,
+     ylim = c(min(ys), max(ys)), xpd = F)
> arrows(xs, means + ses, xs, means - ses, ang = 90, length = 0.05,
+     code = 3)
> axis(2, las = 1)
> mtext(2, text = "Mean pea length", line = 3, cex = 1.5)
> mtext(1, text = "Sugar treatment", line = 3, cex = 1.5)
> text(xs, means + ses, labels = c("A", "B", "B", "B", "C"),
      pos = 3)
> box(bty = "l")
```



Example 10F: Welch's test

Sánchez-Piñero and Polis (2000) studied the effects of sea birds on tenebrionid beetles on islands in the Gulf of California. These beetles are the dominant consumers on these islands and it was envisaged that sea birds leaving guano and carrion would increase beetle productivity. They had a sample of 25 islands and recorded the beetle density, the type of bird colony (roosting, breeding, no birds), % cover of guano and % plant cover of annuals and perennials.

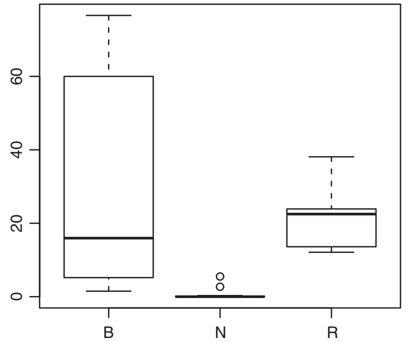
Step 1 - Import the Sánchez-Piñero and Polis (2000) data set

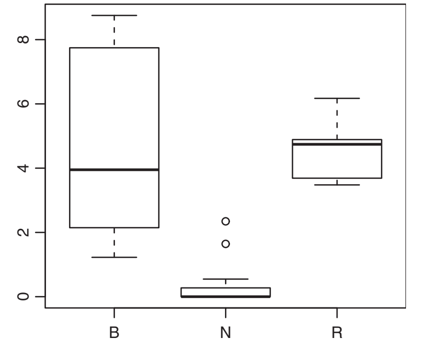
```
> sanchez <- read.table("sanchez.csv", header = T, sep = ",")
```

Step 2 (Keys 10.1a & 10.5) - Check the assumptions and scale data if necessary

```
> boxplot(GUANO ~ COLTYPE,
+         data = sanchez)
```

```
> boxplot(sqrt(GUANO) ~ COLTYPE,
+      data = sanchez)
```





Conclusions - clear evidence that normality and homogeneity of variance assumptions are likely to be violated, square-root transformation improves normality, however, there is still clear evidence that that homogeneity of variance assumption is likely to be violated. Consequently use a Welch's test.

Step 3 (Key 10.6a) - Perform the Welch's test.

```
> oneway.test(sqrt(GUANO) ~ COLTYPE, data = sanchez)
        One-way analysis of means (not assuming equal variances)

data:  sqrt(GUANO) and COLTYPE
F = 42.2862, num df = 2.000, denom df = 10.706, p-value = 8.282e-06
```

Conclusions - Reject the null hypothesis that population means are equal - percentage guano cover differs significantly in different colony types.

Step 4 (Key 10.9c) - Perform post-hoc test.

```
> pairwise.t.test(sqrt(sanchez$GUANO), sanchez$COLTYPE,
+     pool.sd = F, p.adj = "holm")
        Pairwise comparisons using t tests with non-pooled SD

data:  sqrt(sanchez$GUANO) and sanchez$COLTYPE

  B      N
N 0.0091 -
R 0.9390 2.7e-05

P value adjustment method: holm
```

Conclusions - Square root transformed guano cover was significantly higher in breeding colonies than roosting colonies and significantly lower in roosting colonies than the controls and sections treated with sucrose were significantly longer than sections treated with either glucose, fructose or a mixture of glucose and fructose.

Alternatively, the Dunn-Sidak procedure of p-value adjustments could be performed. First re-perform each of the pairwise comparisons but without any p-value corrections and keep a copy of the p-values. Examine these unadjusted p-values to determine which p-value is associated with which comparison. Then use the `mt.rawp2adjp` function of the `multtest` package to perform Dunn-Sidak step-down p-value corrections. Note that adjusted p-values are ordered from lowest to largest and labels are not supplied, so to determine which p-values are associated with which comparison, cross reference with the raw p-values or use the values of the index attribute.

```
> pvalues <- pairwise.t.test(sqrt(sanchez$GUANO), sanchez$COLTYPE,
+     pool.sd = F, p.adj = "none")$p.value
> pvalues
           B            N
N 0.00455275           NA
R 0.93900231 8.846058e-06

> library(multtest)
> mt.rawp2adjp(pvalues, proc = "SidakSD")
$adjp
             rawp      SidakSD
[1,] 8.846058e-06 3.538376e-05
[2,] 4.552750e-03 1.359616e-02
[3,] 9.390023e-01 9.962793e-01
[4,]           NA           NA

$index
[1] 4 1 2 3

$h0.ABH
NULL

$h0.TSBH
NULL
```

Conclusions - the square root transformed guano cover of sites without birds was found to be significantly lower than the cover in both breeding ($p < 0.001$) and roosting ($p = 0.0136$) colonies, however the square root transformed guano cover was not found to differ significantly between breeding and roosting colonies ($p = 0.996$).

Step 5 - Summarize findings with a bargraph

```
> library(biology)
> Mbargraph(sanchez$GUANO, sanchez$COLTYPE, symbols = c("A", "B",
+     "A"), ylab = "Mean percentage Guano cover",
+     xlab = "Bird colony type")
```

Example 10G: Randomization test

As part of a study into the diets of of eastern horned lizard (*Phrynosoma douglassi brevirostre*), Powell and Russell (1984, 1985) investigated whether the consumption of ants changed over time from June to September (Example 5.1 from Manly, 1991). They measured the dry biomass of ants collected from the stomachs of 24 adult male and yearling females in June, July, August and September of 1980.

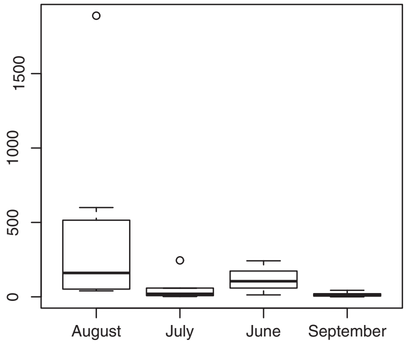
Step 1 - Import the Powell and Russell (1984, 1985) data set

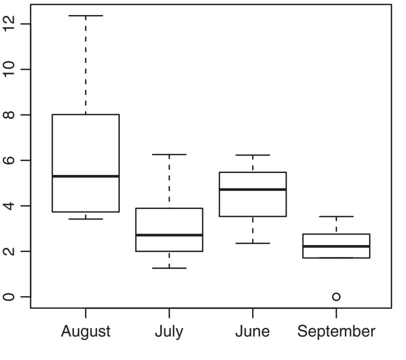
```
> ants <- read.table("ants.csv", header = T, sep = ",")
```

Step 2 (Key 10.1a) - Assess normality/homogeneity of variance using boxplot of ant biomass against month. Cube root transformation also assessed.

```
> boxplot(BIOMASS ~ MONTH, ants)
```

```
> boxplot(BIOMASS^(1/3) ~ MONTH,
        ants)
```





Conclusions - strong evidence of non-normality and unequal variance in raw data. Cube root transformation greatly improved homogeneity of variance, however there is evidence that the populations are not of the same distribution (August appears to be skewed). As a result a randomization test in which the the *F*-distribution is generated from the samples, might be more robust than an ANOVA that assumes each of the populations are normally distributed.

Step 3 (Key 10.8b) - define the statistic to use in the randomization test – in this case the *F*-ratio

```
> stat <- function(data, indices) {
+     f.ratio <- anova(aov(BIOMASS^(1/3) ~ MONTH, data))$"F
+     value"[1] f.ratio
+ }
```

Step 4 (Key 10.8b) - define how the data should be randomized – randomly reorder the which month each biomass observation was collected (without replacement)

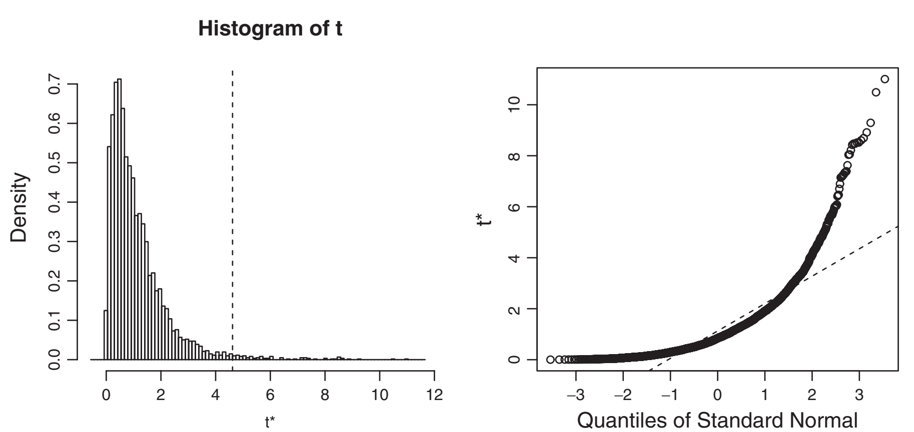
```
> rand.gen <- function(data, mle) {
+     out <- data
+     out$MONTH <- sample(out$MONTH, replace = F)
+     out
+ }
```

Step 5 (Key 10.8b) - call a bootstrapping procedure to randomize 5000 times (this can take some time).

```
> ants.boot <- boot(ants, stat, R = 5000, sim = "parametric",
      ran.gen = rand.gen)
```

Step 6 (Key 10.8b) - examine the distribution of *F*-ratios generated from the randomization procedure

```
> plot(ants.boot)
```



Step 7 (Key 10.8b) - examine the bootstrap statistics

```
> print(ants.boot)
PARAMETRIC BOOTSTRAP
```

```
Call:
boot(data = ants, statistic = stat, R = 5000, sim = "parametric",
    ran.gen = rand.gen)


Bootstrap Statistics :
    original    bias    std. error
t1* 4.618806 -3.491630    1.074420
```

Conclusions - The observed *F*-ratio was 4.619

Step 8 (Key 10.8b) - calculate the number of possible *F*-ratios (including the observed *F*-ratio, which is one possible situation) that were greater or equal to the observed *F*-ratio and express this as a percentage of the number of randomizations (plus one for the observed situation) performed.

```
> f <- length(ants.boot[ants.boot$t >= ants.boot$t0]) + 1
> print(f/(ants.boot$R + 1))
[1] 0.0159968
```

Conclusions - Reject the null hypothesis that the population cubed root ant biomass consumption was equal in each of the four months because the p-value was less than 0.05. The consumption of ants by eastern horned lizard different between the four months.

Step 9 - Perform post-hoc multiple comparisons via randomization and use the Holm correction procedure on the pairwise p-values. For each pairwise comparison, specify which levels of the categorical variable to include in the randomization (`boot`) function and calculate a p-value.

```
> ants.rand1 <- boot(ants[ants$MONTH == "September" | ants$MONTH ==
+     "August", ], stat, R = 1000, sim = "parametric", ran.gen =
+     rand.gen)
> ants.rand2 <- boot(ants[ants$MONTH == "September" | ants$MONTH ==
+     "July", ], stat, R = 1000, sim = "parametric", ran.gen =
+     rand.gen)

> p.S.A <- print(length(ants.rand1[ants.rand1$t >= ants.rand1$t0])/
+     (ants.rand1$R + 1))
[1] 0.000999001
> p.S.Jy <- print(length(ants.rand2[ants.rand2$t >= ants.rand2$t0])/
+     (ants.rand2$R + 1))
[1] 0.2677323
```

Step 10 - Compile a list of all the pairwise p-values and perform Holm correction.

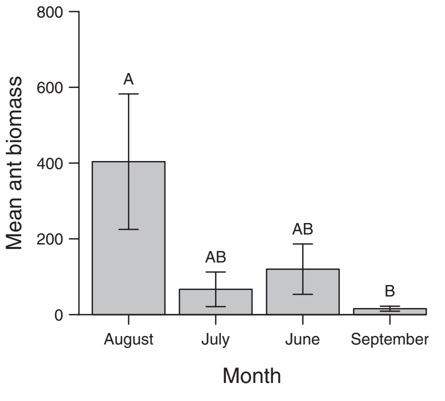
```
> p.values <- c('Sep vs Aug' = p.S.A, 'Sep vs Jul' = p.S.Jy,
+     'Sep vs Jun' = p.S.Jn, 'Aug vs Jul' = p.A.Jy,
+     'Aug vs Jun' = p.A.Jn, 'Jul vs Jun' = p.Jy.Jn)
```

```
> p.adjust(p.values, "holm")
 Sep vs Aug  Sep vs Jul  Sep vs Jun  Aug vs Jul  Aug vs Jun
     Jul vs Jun
0.005994006 0.803196803 0.264735265 0.264735265 0.803196803
     0.803196803
```

Conclusions - The cubed root ant biomass consumption by eastern horned lizards was found to be significantly different between September and August (p=0.006), but was not found to be significantly different between any other month pairs.

Step 11 - Summarize findings with a bargraph

```
> Mbargraph(ants$BIOMASS, ants$MONTH, symbols = c("A", "AB", "AB",
+     "B"), ylab = "Mean ant biomass", xlab = "Month")
```



11

Nested ANOVA

When single sampling units are selected amongst highly heterogeneous conditions (as represented in Figure 11.1a), it is unlikely that these single units will adequately represent the populations and repeated sampling is likely to yield very different outcomes. As a result, the amount of variation within the main treatment effect (unexplained variability) remains high, thereby potentially masking any detectable effects due to the measured treatments. Although this problem can be addressed by increased replication, this is not always practical or possible. For example, if we were investigating the impacts of fuel reduction burning across a highly heterogeneous landscape, our ability to replicate adequately might be limited by the number of burn sites available.

Alternatively, sub-replicates within each of the sampling units (e.g. sites) can be collected (and averaged) so as to provided better representatives for each of the units (see Figure 11.1b) and ultimately reduce the unexplained variability of the test of treatments. In essence, the sub-replicates are the replicates of an additional *nested* factor whose levels are nested within the main treatment factor. A nested factor refers to a factor whose levels are unique within each level of the factor it is nested within and each level is only represented once. For example, the fuel reduction burn study design could consist of three burnt sites and three un-burnt (control) sites each containing four quadrats (replicates of site and sub-replicates of the burn treatment). Each site represents a unique level of a random factor (any given site cannot be both burnt and un-burnt) that is nested within the fire treatment (burned or not).

A nested design can be thought of as a hierarchical arrangement of factors (hence the alternative name *hierarchical* designs) whereby a treatment is progressively sub-replicated. As an additional example, imagine an experiment designed to comparing the leaf toughness of a number of tree species. Working down the hierarchy, five individual trees were randomly selected within (nested within) each species, three branches were randomly selected within each tree, two leaves were randomly selected within each branch and the force required to shear the leaf material in half (transversely) was measured in four random locations along the leaf. Clearly any given leaf can only be from a single branch, tree and species. Each level of sub-replication is introduced to further reduce the amount of unexplained variation and thereby increasing the power of the test for the main treatment effect (the effect of species). Additionally, it is possible to investigate which scale of replication has the greatest (or least, etc) degree of variability - the level of the species, individual tree, branch, leaf, leaf region etc.

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

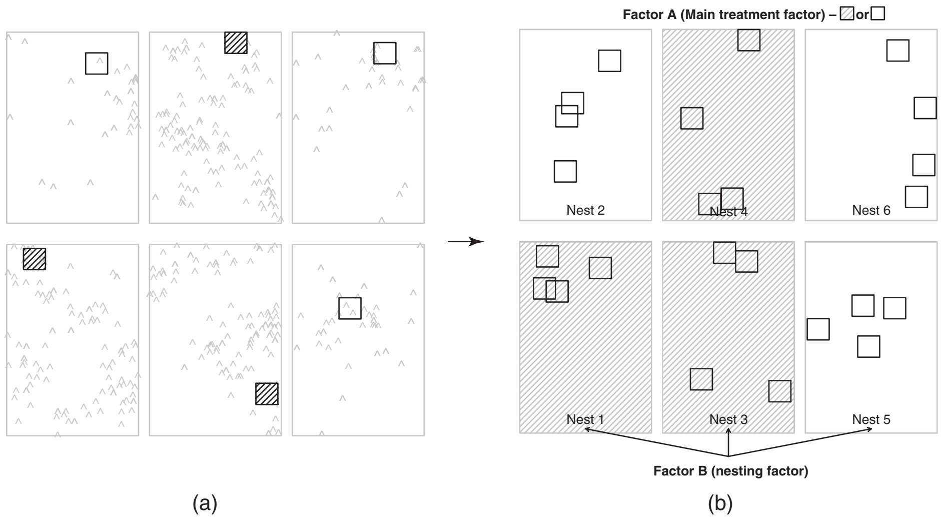


Fig 11.1 Fictitious spatial depictions contrasting (a) single factor and (b) nested ANOVA designs each with three replicate sampling units for each of two treatment levels (n = 3 for each treatment level). When single sampling units are selected amongst highly heterogeneous conditions (as represented in (a)), it is unlikely that these single units will adequately represent the populations and repeated sampling is likely to yield very different outcomes. For such situations, this heterogeneity increases the unexplained variation thereby potentially masking any detectable effects due to the measured treatments. Sub-replicates within each of the sampling units can be collected so as to provided a better representative for each unit.

Nested factors are typically random factors (see section 10.0.1), of which the levels are randomly selected to represent all possible levels (e.g. sites). When the main treatment effect (called Factor A) is a fixed factor, such designs are referred to as a *mixed model nested anova*, whereas when Factor A is random, the design is referred to as a *Model II nested anova*. Fixed nested factors are also possible. For example, specific dates (corresponding to particular times during a season) could be nested within season. When all factors are fixed, the design is referred to as a *Model I mixed model*.

Fully nested designs (the topic of this chapter) differ from other multi-factor designs in that all factors within (below) the main treatment factor are nested and thus interactions are un-replicated and cannot be tested[a]. Partly nested designs in which some of the factors within the main treatment effect are not nested (that is, their levels are repeated within each of the levels of the factor(s) above) are dealt with in chapter 14.

11.1 Linear models

The linear models for two and three factor nested design are:

$$\begin{aligned} y_{ijk} &= \mu + \alpha_i + \beta_{j(i)} + \varepsilon_{ijk} \\ y_{ijkl} &= \mu + \alpha_i + \beta_{j(i)} + \gamma_{k(j(i))} + \varepsilon_{ijkl} \end{aligned}$$

[a] Interaction effects are assumed to be zero.

where μ is the overall mean, α is the effect of Factor A, β is the effect of Factor B, γ is the effect of Factor C and ε is the random unexplained or residual component.

11.2 Null hypotheses

Separate null hypotheses are associated with each of the factors, however, nested factors are typically only added to absorb some of the unexplained variability and thus, specific hypotheses tests associated with nested factors are of lesser biological importance.

11.2.1 *Factor A* - the main treatment effect

Fixed

$$\mathrm{H}_0(A): \mu_1 = \mu_2 = \ldots = \mu_i = \mu \qquad \text{(the population group means are all equal)}$$

The mean of population 1 is equal to that of population 2 and so on, and thus all population means are equal to an overall mean. If the effect of the i^{th} group is the difference between the i^{th} group mean and the overall mean ($\alpha_i = \mu_i - \mu$) then the H_0 can alternatively be written as:

$$\mathrm{H}_0(A): \alpha_1 = \alpha_2 = \ldots = \alpha_i = 0 \qquad \text{(the effect of each group equals zero)}$$

If one or more of the α_i are different from zero (the response mean for this treatment differs from the overall response mean), the null hypothesis is not true indicating that the treatment does affect the response variable.

Random

$$\mathrm{H}_0(A): \sigma^2_\alpha = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of A.

11.2.2 *Factor B* - the nested factor

Random (typical case)

$$\mathrm{H}_0(B): \sigma^2_\beta = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of B within the (set or all possible) levels of A.

Fixed

$$\mathrm{H}_0(B): \mu_{1(1)} = \mu_{2(1)} = \ldots = \mu_{j(i)} = \mu \qquad \text{(the population group means of B (within A) are all equal)}$$

$$\mathrm{H}_0(B): \beta_{1(1)} = \beta_{2(1)} = \ldots = \beta_{j(i)} = 0 \qquad \text{(the effect of each chosen B group equals zero)}$$

Table 11.1 *F*-ratios, estimated variance components (for balanced ANOVA only) and corresponding R syntax for two factor nested designs.

			A fixed/random, B random		A fixed/random, B fixed	
Factor	**d.f.**	**MS**	***F*-ratio**	**Var. comp.**	***F*-ratio**	**Var. comp.**
A	$a-1$	MS_A	$\frac{MS_A}{MS_{B'(A)}}$	$\frac{MS_A - MS_{B'(A)}}{nb}$	$\frac{MS_A}{MS_{Resid}}$	$\frac{MS_A - MS_{Resid}}{nb}$
B′(A)	$(b-1)a$	$MS_{B'(A)}$	$\frac{MS_{B'(A)}}{MS_{Resid}}$	$\frac{MS_{B'(A)} - MS_{Resid}}{n}$	$\frac{MS_{B'(A)}}{MS_{Resid}}$	$\frac{MS_{B'(A)} - MS_{Resid}}{n}$
Residual (=N′(B′(A)))	$(n-1)ba$	MS_{Resid}				

A fixed/random, B random

```
> summary(aov(DV~A+Error(B), data))
> VarCorr(lme(DV~A,random=~1|B)
```

Unbalanced

```
> anova(lme(DV~A,random=~1|B), data)
```

A fixed/random, B fixed

```
> summary(aov(DV~A+B, data)))
```

Unbalanced

```
> Anova(aov(DV~A/B,data),type="III")
```
[a]

[a]To use Type III sums of squares, Factor B contrasts must first be defined as something other than 'treatment' (such as 'sum' or 'helmert') prior to fitting the model (`> contrasts(data$B)<-contr.helmert`).

The null hypotheses associated with additional factors, are treated similarly to Factor B above.

11.3 Analysis of variance

Analysis of variance sequentially partitions the total variability in the response variable into components explained by each of the factors[b] (starting with the factors lowest down in the hierarchy - the most deeply nested) and the components unexplained by each factor. When the null hypothesis for a factor is true (no effect or added variability), the ratio of explained and unexplained components for that factor (*F*-ratio) should follow a theoretical *F*-distribution with an expected value less than 1.

The appropriate unexplained residuals and therefore the appropriate *F*-ratios for each factor differ according to the different null hypotheses associated with different combinations of fixed and random factors in a nested linear model (see Tables 11.1 & 11.2).

11.4 Variance components

As previously alluded to, it can often be useful to determine the relative contribution (to explaining the unexplained variability) of each of the factors as this provides insights

[b] Explained variability is calculated by subtracting the amount unexplained by the factor from the amount unexplained by a reduced model that does not contain the factor.

Table 11.2 *F*-ratios, estimated variance components (for balanced ANOVA only) and corresponding R syntax for three factor nested designs.

		A fixed/random, B random		A fixed/random, B fixed	
Factor	**d.f.**	***F*-ratio**	**Var. comp.**	***F*-ratio**	**Var. comp.**
C′ random					
A	$a-1$	$\frac{MS_A}{MS_{B'(A)}}$	$\frac{MS_A - MS_{B'(A)}}{ncb}$	$\left[\frac{MS_A}{MS_{C'(B'(A))}}\right]$	$\frac{MS_A - MS_{C'(B'(A))}}{ncb}$
B′(A)	$(b-1)a$	$\frac{MS_{B'(A)}}{MS_{C'(B'(A))}}$	$\frac{MS_{B'(A)} - MS_{C'(B'(A))}}{nc}$	$\left[\frac{MS_{B'(A)}}{MS_{C'(B'(A))}}\right]$	$\frac{MS_{B'(A)} - MS_{C'(B'(A))}}{nc}$
C′(B′(A))	$(c-1)ba$	$\frac{MS_{C'(B'(A))}}{MS_{Resid}}$	$\frac{MS_{C'(B'(A))} - MS_{Resid}}{n}$	$\frac{MS_{C'(B'(A))}}{MS_{Resid}}$	$\frac{MS_{C'(B'(A))} - MS_{Resid}}{n}$
Residual (=N′(C′(B′(A))))	$(n-1)cba$		MS_{Resid}		MS_{Resid}

A fixed/random, B random, C random

```
> summary(aov(DV~A+Error(B/C), data))
> VarCorr(lme(DV~A,random=~1|B/C, data))
```

Unbalanced

```
> anova(lme(DV~A,random=~1|B/C, data))
```

A fixed/random, B fixed, C random

```
> summary(aov(DV~A+B+Error(C), data))
> VarCorr(lme(DV~A+B,random=~1|C, data))
```

Unbalanced

```
> anova(lme(DV~A+B,random=~1|C,data))
```

(*continued overleaf*)

Table 11.2 (*continued*)

		A fixed/random, B random		A fixed/random, B fixed	
Factor	**d.f.**	***F*-ratio**	**Var. comp.**	***F*-ratio**	**Var. comp.**
C fixed					
A	$a-1$	$\frac{MS_A}{MS_{B(A)}}$	$\frac{MS_A - MS_{B(A)}}{ncb}$	$\frac{MS_A}{MS_{Resid}}$	$\frac{MS_A - MS_{Resid}}{ncb}$
B′(A)	$(b-1)a$	$\frac{MS_{B'(A)}}{MS_{Resid}}$	$\frac{MS_{B'(A)} - MS_{Resid}}{nc}$	$\frac{MS_{B'(A)}}{MS_{Resid}}$	$\frac{MS_{B'(A)} - MS_{Resid}}{nc}$
C(B′(A))	$(c-1)ba$	$\frac{MS_{C(B'(A))}}{MS_{Resid}}$	$\frac{MS_{C(B'(A))} - MS_{Resid}}{n}$	$\frac{MS_{C(B'(A))}}{MS_{Resid}}$	$\frac{MS_{C(B'(A))} - MS_{Resid}}{n}$
Residual (=N′(C(B′(A))))	$(n-1)cba$		MS_{Resid}		MS_{Resid}

A fixed/random, B random, C fixed

```
> summary(aov(DV~A+Error(B), data))
> VarCorr(lme(DV~A,random=~1|B)
```

Unbalanced

```
> anova(lme(DV~A,random=~1|B), data
```

A fixed/random, B fixed, C fixed

```
> summary(aov(DV~A+B, data)))
```

Unbalanced[a]

```
> Anova(aov(DV~A/B,data),type="III")
```

[a]To use Type III sums of squares, Factor B contrasts must first be defined as something other than ‘treatment’ (such as ‘sum’ or ‘helmert’) prior to fitting the model (`> contrasts(data$B)<-contr.helmert`).

into the variability at each different scale. These contributions are known as *variance components* and are estimates of the added variances due to each of the factors. For consistency with other texts, I have included estimated variance components for various balanced nested ANOVA designs in Tables 11.1 & 11.2. However, variance components based on a modified version of the maximum likelihood iterative model fitting (see chapter 3.7.2) procedure (REML) is generally recommended as this accommodates both balanced and unbalanced designs.

While there are no numerical differences in the calculations of variance components for fixed and random factors, fixed factors are interpreted very differently and arguably have little biological meaning (other to infer relative contribution). For fixed factors, variance components estimate the variance between the means of the specific populations that are represented by the selected levels of the factor and therefore represent somewhat arbitrary and artificial populations. For random factors, variance components estimate the variance between means of all possible populations that could have been selected and thus represents the true population variance.

11.5 Assumptions

An F-distribution represents the relative frequencies of all the possible F-ratio's when a given null hypothesis is true and certain assumptions about the residuals (denominator in the F-ratio calculation) hold. Consequently, it is also important that diagnostics associated with a particular hypothesis test reflect the denominator for the appropriate F-ratio. For example, when testing the null hypothesis that there is no effect of Factor A ($H_0(A) : \alpha_i = 0$) in a mixed nested anova, the means of each level of Factor B are used as the replicates of Factor A. As with single factor anova, hypothesis testing for nested ANOVA assumes the residuals are (for greater explanation of each see chapter 10.4):

(i) normally distributed. Factors higher up in the hierarchy of a nested model are based on means (or means of means) of lower factors and thus the Central Limit Theory would predict that normality will usually be satisfied for the higher level factors. Nevertheless, boxplots using the appropriate scale of replication should be used to explore normality. Scale transformations are often useful.
(ii) equally varied. Boxplots and plots of means against variance (using the appropriate scale of replication) should be used to explore the spread of values. Residual plots should reveal no patterns (see Figure 8.5). Scale transformations are often useful.
(iii) independent of one another - this requires special consideration so as to ensure that the scale at which sub-replicates are measured is still great enough to enable observations to be independent.

11.6 Pooling denominator terms

Designs that incorporate fixed and random factors (either nested or factorial), involve F-ratio calculations in which the denominators that are themselves random factors other than the overall residuals. Many statisticians argue that when such denominators are themselves not statistically significant (at the 0.25 level), there are substantial power benefits from pooling together successive non-significant denominator terms. Thus an F-ratio for a particular factor might be recalculated after pooling together its original denominator with its denominators denominator and so on. The conservative 0.25 is used instead of the usual 0.05 to reduce further the likelihood of Type II errors (falsely concluding an effect is non-significant - that might result from insufficient power).

11.7 Unbalanced nested designs

Unbalanced designs are those designs in which sample (subsample) sizes for each level of one or more factors differ. These situations are relatively common in biological research, however such imbalance has some important implications for nested designs. Firstly, hypothesis tests are more robust to the assumptions of normality and equal variance when the design is balanced. Secondly (and arguably, more importantly), the model contrasts are not orthogonal (independent) and the sums of squares component attributed to each of the model terms cannot be calculated by simple additive partitioning of the total sums of squares (see section 12.6). In such situations, exact F-ratios cannot be constructed (at least in theory[c]), variance components calculations are more complicated and significance tests cannot be computed.

The severity of this issue depends on which scale of the sub-sampling hierarchy the unbalance(s) occurs as well whether the unbalance occurs in the replication of a fixed or random factor. For example, whilst unequal levels of the first nesting factor (e.g. unequal number of burn vs un-burnt sites) has no effect on F-ratio construction or hypothesis testing for the top level factor (irrespective of whether either of the factors are fixed or random), unequal sub-sampling (replication) at the level of a random (but not fixed) nesting factor will impact on the ability to construct F-ratios and variance components of all terms above it in the hierarchy.

There are a number of alternative ways of dealing with unbalanced nested designs[d]:

(i) split the analysis up into separate smaller simple ANOVA's each using the means of the nesting factor to reflect the appropriate scale of replication. As the resulting sums of squares components are thereby based on an aggregated dataset the analyses then inherit the procedures and requirements of single (chapter 10) or fully factorial (chapter 12) ANOVA.

(ii) adopt mixed-modelling techniques (see section 11.8)

11.8 Linear mixed effects models

Although the term 'mixed-effects' can be used to refer to any design that incorporates both *fixed* and *random* predictors, its use is more commonly restricted to designs in

[c] The denominator MS in an F-ratio is determined by examining the expected value of the mean squares of each term in a model. Unequal sample sizes result in expected means squares for which there are no obvious logical comparators that enable the impact of an individual model term to be isolated.

[d] All assume that the imbalance is not a direct result of the treatments themselves. Such outcomes are more appropriately analysed by modelling the counts of surviving observations via frequency analysis (see chapters 16&17).

which factors are nested or grouped within other factors. Typically examples include nested, longitudinal[e] data, repeated measures and blocking designs (see chapters 13 & 14). Furthermore, rather than basing parameter estimations on observed and expected mean squares or error strata (as outlined above), mixed-effects models estimate parameters via maximum likelihood (ML) or residual maximum likelihood (REML). In so doing, mixed-effects models more appropriately handle estimation of parameters, effects and variance components of unbalanced designs (particularly for random effects). Resulting fitted (or expected) values of each level of a factor (for example, the expected population site means) are referred to as Best Linear Unbiased Predictors (BLUP's). As an acknowledgement that most estimated site means will be more extreme than the underlying true population means they estimate[f], BLUP's are less spread from the overall mean than are simple site means. In addition, mixed-effects models naturally model the 'within-block' correlation structure that complicates many longitudinal designs (see section 13.4.1). Whilst the basic concepts of mixed-effects models have been around for a long time, recent computing advances and adoptions have greatly boosted the popularity of these procedures.

Linear mixed effects models are currently at the forefront of statistical development, and as such, are very much a work in progress - both in theory and in practice. Recent developments have seen a further shift away from the traditional practices associated with degrees of freedom, probability distribution and p-value calculations.

The traditional approach to inference testing is to compare the fit of an alternative (full) model to a null (reduced) model (via an *F*-ratio). When assumptions of normality and homogeneity of variance apply, the degrees of freedom are easily computed and the *F*-ratio has an exact *F*-distribution to which it can be compared. However, this approach introduces two additional problematic assumptions when estimating fixed effects in a mixed effects model.

Firstly, when estimating the effects of one factor, the parameter estimates associated with other factor(s) are assumed to be the true values of those parameters (not estimates). Whilst this assumption is reasonable when all factors are fixed, as random factors are selected such that they represent one possible set of levels drawn from an entire population of possible levels for the random factor, it is unlikely that the associated parameter estimates accurately reflect the true values. Consequently, there is not necessarily an appropriate *F*-distribution.

Furthermore, determining the appropriate degrees of freedom (nominally, the number of independent observations on which estimates are based) for models that incorporate a hierarchical structure is only possible under very specific circumstances (such as completely balanced designs). Degrees of freedom is a somewhat arbitrary defined concept used primarily to select a theoretical probability distribution on which a statistic can be compared. Arguably, however, it is a concept that is overly simplistic for complex hierarchical designs.

Most statistical applications continue to provide the 'approximate' solutions (as did earlier versions within R). However, R linear mixed effects development leaders argue

[e] measurements repeated over time.

[f] This is based on the principle that smaller sample sizes result in greater chances of more extreme observations and that nested sub-replicates are also likely to be highly intercorrelated).

strenuously that given the above shortcomings, such approximations are variably inappropriate and are thus omitted.

Markov chain Monte Carlo (MCMC) sampling methods provide a Bayesian-like alternative for inference testing. Markov chains use the mixed model parameter estimates to generate posterior probability distributions of each parameter from which Monte Carlo sampling methods draw a large set of parameter samples. These parameter samples can then be used to calculate highest posterior density (HPD) intervals[g]. Such intervals indicate the interval in which there is a specified probability (typically 95%) that the true population parameter lies. Furthermore, whilst technically against the spirit of the Bayesian philosophy, it is also possible to generate P values on which to base inferences.

11.9 Robust alternatives

There are no formal robust or non-parametric tests specifically formulated for nested analyses. However, since nested designs simply represent a hierarchical set of ANOVA's, it is possible to employ the techniques outlined in chapter 10.5 in a series of simple ANOVA's each using aggregated portions of the full data set (reflecting the appropriate scale of replication of each individual hypothesis test). Likewise, randomization tests (which are useful for situations in which observation independence could be questionable) can be performed by comparing the F-ratios to a large number of sets of F-ratios calculated from repeatedly shuffled data[h].

Note that nested designs are often incompatible with randomization procedures due to the low number of possible randomization combinations possible. For example, if the design consists of three locations nested within two treatments (e.g. burnt and unburnt), there are only $(kn)!/[(n!)^k k!] = 10$ (where n is the number of replicates within each of the k treatments) unique ways in which the sites can be randomized within the treatments, and thus the smallest possible p-value is 0.1 (1/10).

11.10 Power and optimisation of resource allocation

Since nested designs represent a hierarchical set of ANOVA's, it is possible to employ the power analysis techniques outlined in section 10.7 in a series of analyses using aggregated portions of the full data set (reflecting the appropriate scale of replication of each individual hypothesis test).

At the start of this chapter, an example of a leaf toughness investigation was introduced so as to demonstrate the nature of a nested design. In this example, the choice of sample size within each scale of sub-replication (individual tree, branch, leaf) was completely arbitrary, yet such choices are actually of great importance. Since the individual trees are the direct replicates of the species treatment, the power of the test

[g] HPD intervals are also known as Bayesian credible intervals.

[h] Various ways of shuffling the data have been suggested. These include:

(i) Complete shuffling of the data set

(ii) When testing a given factor, constrain (restrict) the shuffling to the scale of the replicates for that factor.

of species is directly affected by the number of replicate trees per species. However, the power of this test will also indirectly benefit from greater replication at the scale with the greatest degree of variability as this will further reduce the unexplained variability.

The optimal degree of replication at each levels of a nested design can be assessed by examining the ratio of the variance components of each of the nested effects with their respective residual variance components. Furthermore, such calculations can incorporate the costs (time and/or money) associated with each level of replication so as to estimate the optimal allocation of resources. For example, in a three factor mixed nested design (fixed A, random B and C), the optimum number of replicates within each level of the random nested factors B and C would be defined by:

$$r = \sqrt{\frac{C_{B(A)} s^2_{C(B(A))}}{C_{C(B(A))} s^2_{B(A)}}} \qquad n = \sqrt{\frac{C_{C(B(A))} s^2}{C_{Reps} s^2_{C(B(A))}}}$$

where C and s^2 are respectively the cost and estimated variances associated with the subscripted effects levels and r and n denote the number of replicates for B (levels of C) and C respectively. Note that for two factor mixed nested model, only the first of these are required (although it is now defining r) and $C(B(A))$ represents the lowest form of replication and therefore the overall residuals (s^2). Costs can be ignored by making them equal to 1. Similarly, for any mixed design with a fixed Factor A, the optimum number of replicates of factor A (levels of factor B) can be estimated by solving for q from either of the following:

$$s^2_A = \frac{n s^2_{B(A)} + s^2_{C(B(A))}}{nq}$$

$$C_A = qC_{B(A)} + nqC_{C(B(A))}$$

where s^2_A represents the expected (or desired) variance amongst group means for the fixed Factor A.

11.11 Nested ANOVA in R

11.11.1 Error strata (`aov`)

Nested ANOVA can be thought of as a series of ANOVA models, each with a different error (residual term). Each of the separate models and their corresponding error term are referred to as a *strata*. The first error strata corresponds to a linear model that incorporates factor(s) for which the levels first random nesting factor are the appropriate replicates. Likewise, the second error strata corresponds to the next level of error terms (residuals) and so on. For a two factor mixed nested ANOVA, the second error strata will be the overall measurements (residuals). Modelling ANOVA with multiple error strata is accommodated via the `aov` function. Note however, that this is really only appropriate for balanced designs - particularly if the source of imbalance is at the level of the nesting factor replication.

11.11.2 Linear mixed effects models (`lme` and `lmer`)

The `lme` (`nlme`) and more recent `lmer` (`lme4`) functions facilitate linear mixed-effects and generalized linear mixed-effects modelling respectively. As such these procedures are more suitable for unbalanced and longitudinal designs. Note that recent versions of `lmer` have omitted P value approximations and that inference testing is performed by the `pvals.fnc` (`languageR`) *function* via the presently inconsistent `mcmcsamp` (`lme4`) *function*.

11.12 Further reading

- Theory

  Doncaster, C. P., and A. J. H. Davey. (2007). *Analysis of Variance and Covariance. How to Choose and Construct Models for the Life Sciences*. Cambridge University Press, Cambridge.

  Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. 2 edition. John Wiley & Sons, New York.

  Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

  Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

  Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

  Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

  Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

  Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

  Pinheiro, J. C., and D. M. Bates. (2000). *Mixed effects models in S and S-PLUS*. Springer-Verlag, New York.

  Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS, 4th edn*. Springer-Verlag, New York.

  Zuur, A. F., E. N. Ieno, N. J. Walker, A. A. Saveliev, and G. M. Smith. (2009). *Mixed Effects Models and Extensions in Ecology with R*. Springer.

11.13 Key for nested ANOVA

1 **Determine the appropriate model design and hierarchy**

- Conceptualise the design into a hierarchy (ladder) of factors
  - Main factor(s) with levels that are applied to complete sets of other (nesting) factors at the top
  - Progressively deeper levels of sub-replication of these main factor(s) considered progressively lower in the hierarchy

- Label random nesting factor levels with unique names for each level across the entire design (within and between main factor(s)). Label fixed nesting factor levels according to the levels they represent (recycled label names within each level of the main factor(s))

Random B

Fact A	Fact B	DV
A1	B1	.
A1	B2	.
A2	B3	.
A2	B4	.

Fixed B

Fact A	Fact B	DV
A1	B1	.
A1	B2	.
A2	B1	.
A2	B2	.

- Identify the correct error (residual) term for each factor (see Tables 11.1 & 11.2).

. . . . Go to 2

2 a. Check assumptions for nested ANOVA
As the assumptions of any given hypothesis test relate to residuals, all diagnostics should reflect the appropriate error (residual) terms for the hypothesis. Typically this means generating temporary aggregated data sets.

- **Normality (symmetry) of the response variable at each level of the factor - boxplots of mean values for each level of the next random term in the hierarchy Factor A (with random factor B)**

```
> data.B.agg <- with(data, aggregate(data.frame(DV),
+     by = list(A = A, B = B), mean))
> #OR
> library(nlme)
> data.B.agg <- gsummary(data, data$B)
> boxplot(DV ~ A, data.B.agg)
```

where `DV` *is the response variable,* `A` *is the main fixed factor and* `B` *is a random factor nested within* `A` *within the* `data` *dataset.*
Factor B (random)
If Factor C exits and is random

```
> library(nlme)
> data.C.agg <- gsummary(data, data$C)
> boxplot(DV ~ A:B, data.C.agg)
```

If no random Factor C

```
> boxplot(DV ~ A:B, data)
```

where `DV` *is the response variable,* `A` *is the main fixed factor and* `B` *is a random factor nested within* `A` *within the* `data` *dataset.*

- **Homogeneity of variance (relationship between mean and variance) - boxplots (as above) and scatterplot of mean vs variance (fixed factors only)**

```
> with(data.B.agg, plot(tapply(DV, A, var),
      tapply(DV, A, mean)))
```

where `DV` *is the response variable,* `A` *is the main fixed factor and* `B` *is a random factor nested within* `A` *within the* `data.B.agg` *aggregated dataset.*

Parametric assumptions met . . . . Go to 4

b. Parametric assumptions not met . . . . Go to 3

3 a. Attempt a scale transformation (see Table 3.2 for transformation options) . . . . Go to 2

b. Transformations unsuccessful or inappropriate . . . . Go to 8

4 a. Determine whether the design is balanced and if not, at what scale of replication the imbalance occurs . . . . See Examples 11A,11C,11D

```
> library(biology)
> is.balanced(DV ~ A + b + C + .., data)
> #OR
> !is.list(replications(DV ~ A + b + C + .., data))
```

value of `TRUE` *indicates design is completely balanced*

```
> replications(DV ~ A + b + C + .., data)
```

where `DV` *is the response variable,* `A` *is the main fixed factor and* `B` *is a random factor nested within* `A` *within the* `data` *dataset.*

Design is balanced with respect to the appropriate sub-replicates of the term of interest . . . . Go to 5a-d

b. Design is NOT balanced with respect to the appropriate sub-replicates of the term of interest . . . . Go to 5b-d

5 a. Fit nested model using complete `aov` procedure (for balanced designs only) . See Example 11A

Define planned contrasts if required . . . . Refer back to Key 10.4

```
> data.aov <- aov(DV ~ A + Error(B), data)
> summary(data.aov)
```

For additional combinations of fixed and random factors see Tables 11.1 & 11.2

Examine residuals . . . . Go to 6

For variance components. . . . . Go to 7

b. Fit nested model using simple ANOVA of aggregated dataset . . . . See Example 11C,11D

Factor A (with random factor B)

```
> library(nlme)
> data.B.agg <- gsummary(data, data$B)
```

Define planned contrasts if required . . . . Refer back to Key 10.4

```
> anova(aov(DV ~ A, data.B.agg))
```

Factor B (with random factor C or no C)

```
> library(nlme)
> data.C.agg <- gsummary(data, data$B)
```

Define planned contrasts if required . . . . Refer back to Key 10.4

```
> anova(aov(DV ~ A + B, data.C.agg))
```

where `DV` *is the response variable,* `A` *is the main fixed factor,* `B` *is a random factor nested within* `A` *and* `C` *is a random factor nested within* `B(A)` *within the* `data` *dataset. If there is no random Factor C, substitute* `data` *for* `data.C.agg` *in the* `aov()` *function above.*

For additional combinations of fixed and random factors see Table. 11.2

For variance components. . . . . Go to 7

c. Fit nested model using `lme` procedure . . . . See Example 11D

Define planned contrasts if required .... Refer back to Key 10.4

```
> library(nlme)
> data.lme <- lme(DV ~ A, random = ~1 | B, data)
> summary(data.lme)
> anova(data.lme)
```

OR if three factor mixed-effects (A fixed, B & C random)

```
> data.lme <- lme(DV ~ A, random = ~1 | B/C, data)
> summary(data.lme)
> anova(data.lme)
```

where `DV` *is the response variable,* `A` *is the main fixed factor and* `B` *is a random factor nested within* `A` *and, if present,* `C` *is a random factor nested within* `B(A)` *within the* `data` *dataset. Note that the summary includes variance components for the random factors.*

For additional combinations of fixed and random factors see Table 11.1 & 11.2

Examine residuals .... Go to 6

For variance components.. Go to 7

d. Fit nested model using `lmer` procedure .... See Example 11C,11D

Define planned contrasts if required .... Refer back to Key 10.4

```
> library(lme4)
> data.lmer <- lmer(DV ~ A + (1 | B), data)
> summary(data.lmer)
> anova(data.lmer)
```

OR if three factor mixed-effects (A fixed, B & C random)

```
> data.lmer <- lmer(DV ~ A + (1 | B/C), data)
> summary(data.lmer)
> anova(data.lmer)
```

where `DV` *is the response variable,* `A` *is the main fixed factor and* `B` *is a random factor nested within* `A` *and, if present,*`C` *is a random factor nested within* `B(A)` *within the* `data` *dataset. Note that the summary includes variance components for the random factors.*

Examine residuals .... Go to 6

For model parameter and fixed factor effects confidence intervals via Markov chain Monte Carlo sampling

```
> library(languageR)
> pvals.fnc(data.lmer)
```

For model parameter and fixed factor effects (if more than two groups) significance via Markov chain Monte Carlo sampling

```
> library(languageR)
> pvals <- pvals.fnc(data.lmer, nsim = 10000, withMCMC = T)
> library(biology)
> mcmcpvalue(as.matrix(pvals$mcmc), "A")
```

where `"A"` *is string to indicate the name of the fixed factor (A in this case) to test.*

6 a. Examining a residual plot of the nested models fitted with `aov` . See Example 11A

```
> plot(resid(model[[2]]) ~ fitted(model[[2]]))
```

where `model` *is the name of a model fitted via* `aov` *and* `[[2]]` *refers to the second object in the fitted model (which is the first strata).*

b. Examining a residual plot of the mixed-effects models fitted with `lme` or `lmer` . . . . See Example 11D

```
> plot(resid(model) ~ fitted(model))
```

where `model` *is the name of a model fitted via* `lme` *or* `lmer`.

7 Calculate variance components of random factors . . . . See Example 11A

```
> library(nlme)
> VarCorr(lme(lme(DV ~ A, random = ~1 | B, data)))
```

For additional combinations of fixed and random factors see Table. 11.1 & 11.2

8 a. Underlying distribution of the response variable is normal for each level of the main fixed factor, but the variances are unequal (Welch's test from aggregated data) . . . . See Example 11B

```
> data.B.agg <- gsummary(data, data$B)
> oneway.test(DV ~ A, data.B.agg, var.equal = F)
```

or consider GLM . . . . GLM chapter 17

b. Underlying distributions not normally distributed . . . . Go to 9

9 a. Underlying distribution of the response variable and residuals is known . . . . GLM chapter 17

b. Underlying distributions of the response variable and residuals is not known . . . . Go to 10

10 a. Variances not wildly unequal, outliers present, but data independent (Kruskal-Wallis non-parametric test on aggregated data)

```
> data.B.agg <- gsummary(data, data$B)
> kruskal.test(DV ~ A, data.B.agg, var.equal = F)
```

b. Variances not wildly unequal, random sampling not possible - data might not be independent (Randomization test on aggregated data

```
> data.B.agg <- gsummary(data, data$B)
```

Use this aggregated data set and follow the instructions in Key 10. 8b. **Warning, randomization procedures are only useful when there are a large number of possible randomization combinations (rarely the case in nested designs)**

11.14 Worked examples of real biological data sets

Example 11A: Two factor mixed nested ANOVA

To investigate density-dependent grazing effects of sea urchin Andrew and Underwood (1993) on filamentous algae measured the percentage of filamentous algae within five quadrats randomly positioned within each of four random patches of reef that were in turn nested within four sea urchin density treatments (no urchins, 33% of natural density, 66% natural density and 100% natural density). The sea urchin density treatment was considered a fixed factor and patch within density treatment as well as the individual quadrats were treated as random factors.

Step 1 - Import (section 2.3) the Andrew and Underwood (1993) data set

```
> andrew <- read.table("andrew.csv", header = T, sep = ",")
```

Step 2 - The patch vector (variable) contains numerical representations of the patch identifications, therefore by default R considers this to be a *integer* vector rather than a categorical *factor*. In order to ensure that this variable is treated as a factor we need to redefine its class

```
> class(andrew$PATCH)
[1] "integer"

> andrew$PATCH <- factor(andrew$PATCH)
> class(andrew$PATCH)
[1] "factor"
```

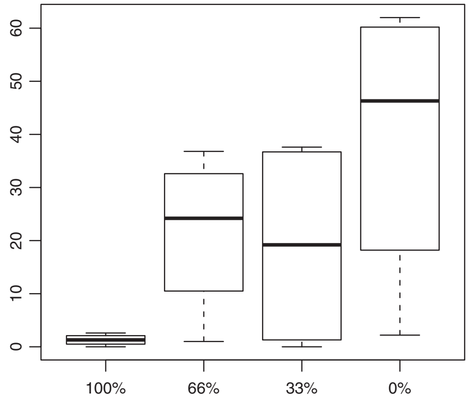
Additionally, all variables that contain strings (alphanumeric characters) are automatically defined as *factor* variables during the data importation stage. In doing so, R by default, orders the levels of all factors in alphabetical order. Consequently, the levels of the density treatment factor are ordered as 0%, 100%, 33%, 66%. Whilst the order of these levels has no impact on the outcome of statistical analyses, defining a more logical order of factor levels can improve graphical summaries and simplify defining contrast matrices. Since 100% density represents the natural density (and thus the control), logically we would order our treatments from 100% down to 0%.

```
> levels(andrew$TREAT)
[1] "0%"   "100%" "33%"  "66%"

> andrew$TREAT <- factor(andrew$TREAT, levels = c("100%", "66%",
+     "33%", "0%"))
```

Step 3 (Key 11.2) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 11.1).

1. Factor A (density treatment - fixed factor). The patch means are the replicates for the density treatment, and thus an aggregated dataset needs to be created from which the boxplots can be based.

```
> andrew.agg <- with(andrew, aggregate(data.frame(ALGAE),
+     by = list(TREAT = TREAT, PATCH = PATCH), mean))
> library(nlme)
> andrew.agg <- gsummary(andrew, groups = andrew$PATCH)

> boxplot(ALGAE ~ TREAT, andrew.agg)
```



Conclusions - Although there is no evidence of non-normality (boxplots not wildly asymmetrical), there is strong evidence of unequal variance. Of particular concern is the apparent relationship between mean and variance (heights of boxplots increase up the y-axis). Transformations (*arcsin*$\sqrt{}$ and log) are ineffectual. Andrew and Underwood (1993) and therefore Quinn and Keough (2002) decided to proceed and rely on the robustness of the parametric test for balanced designs.

2. Factor B (patches - random factor). As this factor is of little biological interest, checking the assumptions associated with its hypothesis tests are of little value.

Conclusions - For the purpose of demonstrating how to use R to perform the worked examples that appear in the popular biostatistics reference literature, we will proceed with raw data (following Quinn and Keough (2002)). Note, however, as a demonstration of non-parametric or robust alternatives in nested designs, we will reanalyse these data in example 11B.
Although Quinn and Keough (2002) did not include either planned or post-hoc comparisons, in this case, the former would seem appropriate. We will compare each of the reduced urchin density treatments to the control – these are known as treatment contrasts[i].

Step 4 (Key 11.4) - Determine whether or not the design is balanced (at least with respect to sub-replication).

```
> replications(ALGAE ~ TREAT + PATCH, andrew)
TREAT PATCH
   20     5

> library(biology)
> is.balanced(ALGAE ~ TREAT + PATCH, andrew)
[1] TRUE
```

Conclusions - The design is completely balanced. There are two replicate patches within each of the four treatments and there are five replicate quadrats within each patch.

Step 5 - Define treatment contrasts (see sections 10.6 and 7.3.1 for more information on setting contrasts).

```
> contrasts(andrew$TREAT) <- contr.treatment
```

Note that there is no need to check the orthogonality of these contrasts, when using one of the contrasts functions, they will always be constructed correctly in accordance with the relevant contrast definition.

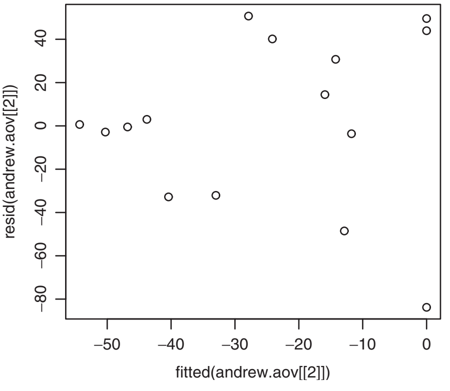
[i] Alternatively, as the levels of the main treatment factor are naturally ordered (according to urchin density), polynomial contrasts might be desirable.

Step 6 (Key 11.5a) - As the design is completely balanced, there are a number of ways to fit the linear model to test the null hypotheses that there is no effect of urchin treatment and no added variance due to patches[j]. The complete `aov()` procedure is the traditional method and arguably the simplest.

```
> andrew.aov <- aov(ALGAE ~ TREAT + Error(PATCH), andrew)
```

Step 7 (Key 11.6a) - Examine the fitted model diagnostics[k]. Note that it is only the first error strata that we are interested in and this is the second object within the `aov` object (hence the `[[2]]`)

```
> plot(resid(andrew.aov[[2]]) ~ fitted(andrew.aov[[2]]))
```



Conclusions - As anticipated, there is an indication of a 'wedge' pattern in the residuals indicative of unequal variance.

Step 8 (Key 11.5a) - Examine the anova tables[l], including the set of defined planned treatment contrasts.

```
> summary(andrew.aov, split = list(TREAT = list('cont vs 66' = 1,
+     'cont vs 33' = 2, 'cont vs 0' = 3)))
Error: PATCH
                    Df  Sum Sq Mean Sq F value  Pr(>F)
TREAT                3 14429.1  4809.7  2.7171 0.09126 .
  TREAT: cont vs 66  1    44.2    44.2  0.0250 0.87707
  TREAT: cont vs 33  1    20.8    20.8  0.0118 0.91540
  TREAT: cont vs 0   1 14364.1 14364.1  8.1146 0.01466 *
Residuals           12 21242.0  1770.2
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[j] Note that if we were also intending to investigate a set of planned comparisons/contrasts (see chapter 10.6), these should be defined prior to fitting the linear model. In this case, treatment contrasts (with the 100% urchin density as the 'control') would probably be the most logical.

[k] Recall that leverage, and thus Cook's D are not informative for categorical predictor variables.

[l] R does not provide the hypothesis tests associated with the random nesting factors as these are rarely of interest. In order to obtain such tests, re-fit the linear model treating the random nesting factor as a fixed factor. All hypothesis tests in the output above this term in the hierarchy should be ignored as they will not be tested against the incorrect error (residual) terms. E.g. `> andrew.aov1<-aov(ALGAE TREAT+PATCH, andrew)`.

```
Error: Within
           Df  Sum Sq Mean Sq F value Pr(>F)
Residuals 64 19110.4   298.6
```

Conclusions - Note that the output has been split into two error strata each reflecting the appropriate error (residual) term to test the corresponding hypothesis against. Do not reject the null hypothesis of no effect of urchin density treatment. Sea urchin density was not found to have an impact on the percentage of filamentous algae. As no overall difference was observed, neither planned or unplanned comparisons are appropriate and therefore ignored.

Step 9 (Key 11.7) - Examine the variance components to determine the relative contribution of each of the random factors. This must be done via a linear mixed effects model. Note further, that to get an estimate of the variance component for a fixed factor (purely for the purpose of comparison to other components, as the actual estimates of variance components for fixed factors are illogical), it must be modelled as a random factor.

```
> library(nlme)
> VarCorr(lme(ALGAE ~ 1, random = ~1 | TREAT/PATCH, andrew))
            Variance     StdDev
TREAT =     pdLogChol(1)
(Intercept) 151.9443     12.32657
PATCH =     pdLogChol(1)
(Intercept) 294.3209     17.15578
Residual    298.6005     17.28006
```

Conclusions - There was a high level of variance between patches within treatment $((294.32 \times 100)/(151.94 + 294.32 + 298.60) = 39.51\%)$ compared to between treatments (20.40%).

Example 11B: Two factor non-parametric mixed nested ANOVA

To demonstrate the hierarchical nature of nested ANOVA designs and how alternative model fitting procedures can be fitted to such designs in R, we will re-analyse the Andrew and Underwood (1993) data (which you may recall from example 11A, did not really satisfy the assumption on equal variance).

Step 1 - Import and prepare the Andrew and Underwood (1993) data set as in Steps 1-2 of example 11A

Step 2- Generate a separate data set for each of the appropriate error strata (consult Table 11.1)

Urchin treatment – for testing the effect of urchin treatment (fixed factor) the patch means are the appropriate replicates. Generate a dataset that is aggregated according to the patch means.

```
> andrew.patch <- with(andrew, aggregate(data.frame(ALGAE),
+     by = list(TREAT = TREAT, PATCH = PATCH), mean))
> library(nlme)
> andrew.patch <- gsummary(andrew, groups = andrew$PATCH)
```

Patch treatment – for testing whether there is any added variance due to patches (random factor) the replicates are the values of the quadrats within the patches that are the appropriate replicates. As the values of the quadrats within each patch are the lowest level of sub-replication represented by the original dataset, the original dataset is appropriate for the error strata for testing the hypothesis about patches.

Step 3 (Key 11.8) - Perform a non-parametric ANOVA on each strata (see also Key 10. 6). Note, it is rarely of interest to test hypotheses about nested factors and thus only the main effect of treatment is tested.

Urchin treatment

```
> oneway.test(ALGAE ~ TREAT, andrew.patch, var.equal = F)
        One-way analysis of means (not assuming equal variances)

data:  ALGAE and TREAT
F = 4.5792, num df = 3.000, denom df = 5.031, p-value = 0.06687
```

Alternatively, we could convert the response variable to ranks and perform the parametric nested ANOVA on these ranks. It should be acknowledged that these methods are not ideal in this example. This approach can be a useful alternative when normality is suspect, yet still assumes similar variances.

```
> summary(aov(rank(ALGAE) ~ TREAT + Error(PATCH), andrew))
Error: PATCH
          Df  Sum Sq Mean Sq F value  Pr(>F)
TREAT      3 10761.7  3587.2  2.8916 0.07929 .
Residuals 12 14886.8  1240.6
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
          Df  Sum Sq Mean Sq F value Pr(>F)
Residuals 64 13432.9   209.9
```

Conclusions - The conclusions are much the same as they were based on the parametric nested ANOVA, thereby confirming the general robustness of balanced ANOVA.

Example 11C: Two factor model II nested ANOVA with unequal sample sizes

Sokal and Rohlf (1997) present a dataset containing single blood pH readings from the female offspring of 15 dams (females). Each of the offspring were nested within different litters resulting from either two or three sires (males) which were in turn nested within the 15 dams. The dams represent a random factor at the top of the hierarchy (Factor A), sire represents the first random nesting factor (Factor B(A)), and the individual offspring within each litter represent the replicates of the sires.

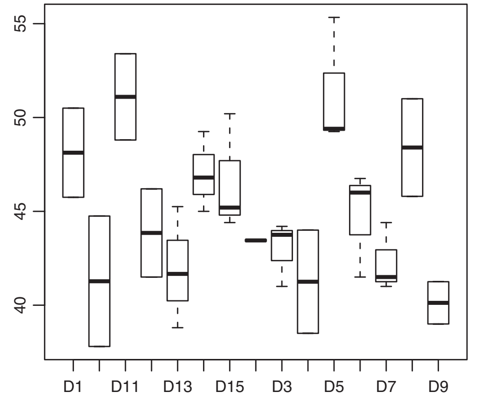
Step 1 - Import (section 2.3) the blood pH data set

```
> ph <- read.table("ph.csv", header = T, sep = ",")
```

Step 2 (Key 11.2) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 11.1).

1. Factor A (dams - random factor). The means of mice within each sire litter are the replicates for the dams, and thus an aggregated dataset needs to be created from which the boxplots can be based.

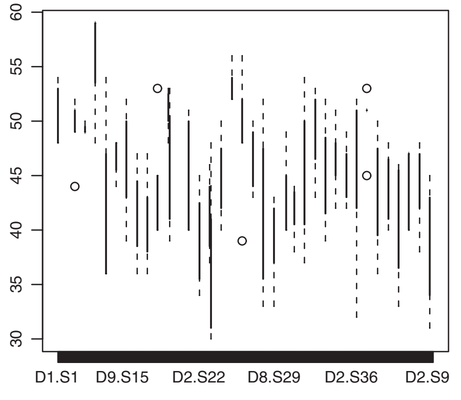
```
> library(nlme)
> ph.agg <- gsummary(ph, groups = ph$SIRE)
> boxplot(PH ~ DAM, ph.agg)
```



Conclusions - no evidence of consistent non-normality and no evidence of a relationship between mean and variance.

2. Factor B (sires - random factor). The blood pH readings from each mice are the replicates of the sires, therefore boxplots should be based on the entire data set.

```
> boxplot(PH ~ DAM:SIRE, ph)
```



Conclusions - no evidence of consistent non-normality and no evidence of a relationship between mean and variance.

Step 3 (Key 11.4) - Assess whether the design is balanced (are there equal sample sizes in each treatment).

```
> replications(PH ~ DAM + SIRE, data = ph)
$DAM
DAM
 D1 D10 D11 D12 D13 D14 D15  D2  D3  D4  D5  D6  D7  D8  D9
  8   9  10   9  12  13  15   9  13   7  12  13  14   8   8

$SIRE
SIRE
 S1 S10 S11 S12 S13 S14 S15 S16 S17 S18 S19  S2 S20 S21 S22 S23 S24
  4   5   4   3   4   4   5   4   5   5   3   4   5   4   4   5   4
S25 S26 S27 S28 S29  S3 S30 S31 S32 S33 S34 S35 S36 S37  S4  S5  S6
  5   5   5   4   5   5   3   4   4   4   5   5   5   5   4   4   4
 S7  S8  S9
  5   3   4
> library(biology)
> is.balanced(PH ~ DAM + SIRE, data = ph)
[1] FALSE
```

Conclusions - the design is not balanced (there are a different number of sired litters and offspring per dam). The `FALSE` indicates that the design is **not** balanced. This design is therefore best modelled using linear mixed effects (REML) procedures. Note that Sokal and Rohlf (1997) employ an older procedure (which some argue is now outdated and potentially inappropriate) in which the *F*-ratio and variance components calculations are adjusted to account for the degree of imbalance.

Step 4 (Key 11.5b) - fit one or more linear models to test the null hypotheses that there is no added variation due to dams and no added variation due to sires within dams. Note, as this is an unbalanced design, we cannot rely on the usual additive partitioning of SS_{Total}. There are two options (both of which will result in slightly different estimates - yet the conclusions are consistent):

1. (Key 11.5b) use a single factor ANOVA to model the effects of dam against the mean pH values for each sire (use the aggregated dataset from Step 2 above).

```
> ph.aov <- aov(PH ~ DAM, ph.agg)
> anova(ph.aov)
Analysis of Variance Table

Response: PH
          Df Sum Sq Mean Sq F value   Pr(>F)
DAM       14 430.90   30.78  3.5464 0.003963 **
Residuals 22 190.93    8.68
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There are maternal influences on the blood pH of female offspring in mice ($F_{14,22} = 3.546, P = 0.003$).

Perform simple ANOVA to investigate the effects of sire using the individual pH readings from each of the offspring as the replicates. Note that the hypothesis test for dam that is included in this modelling should be ignored.

```
> ph.aov1 <- aov(PH ~ DAM + SIRE, data = ph)
> anova(ph.aov1)
Analysis of Variance Table

Response: PH
           Df  Sum Sq Mean Sq F value    Pr(>F)
DAM        14 1780.17  127.16  5.1405 1.563e-07 ***
SIRE       22  800.24   36.37  1.4705   0.09662 .
Residuals 123 3042.53   24.74
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Paternity was not found to have a significant impact on the blood pH of female offspring in mice ($F_{22,123} = 1.470, P = 0.097$).

2. (Key 11.5d) fit the linear mixed effects model using `lmer`.

```
> library(lme4)
> ph.lmer <- lmer(PH ~ 1 + (1 | DAM/SIRE), ph)
> summary(ph.lmer)
Linear mixed model fit by REML
Formula: PH ~ 1 + (1 | DAM/SIRE)
   Data: ph
  AIC  BIC logLik deviance REMLdev
 1006 1019 -499.1    999.9   998.2
Random effects:
 Groups   Name        Variance Std.Dev.
 SIRE:DAM (Intercept)  2.6456  1.6265
 DAM      (Intercept)  8.8957  2.9826
 Residual             24.8079  4.9807
Number of obs: 160, groups: SIRE:DAM, 37; DAM, 15

Fixed effects:
            Estimate Std. Error t value
(Intercept)  44.9179     0.9104   49.34
```

Conclusions - the main interest in this output is the variance components for each of the random effects. It is clear that there is more variation between dams than there is between sires within dams (8.90 *cf* 2.64) suggesting that maternal impacts on female blood pH are stronger than paternal influences. There is however, a large amount of variation between offspring (within sires: 24.81 *cf* 2.64) indicating that blood pH is probably influenced by a number of other factors, some of which may even be more important than the measured maternal and paternal associations.

Step 5 (Key 11.5d) - Calculate the 95% confidence intervals of the random effects (based on Markov chain Monte Carlo sampling).

```
> library(languageR)
> pvals.fnc(ph.lmer)

$fixed
  Estimate MCMCmean HPD95lower HPD95upper  pMCMC Pr(>|t|)
1    44.92    44.91      43.35      46.56 0.0001        0

$random
    Groups        Name Std.Dev. MCMCmedian MCMCmean HPD95lower HPD95upper
1 SIRE:DAM (Intercept)   1.6265     0.6168   0.7046     0.0000     1.8502
2      DAM (Intercept)   2.9826     2.4766   2.5250     1.3511     3.8754
3 Residual               4.9807     5.2150   5.2319     4.6293     5.8855
```

Conclusions - The 95% confidence interval for the random effect of dam (no added variance due to dams) does not include 0, and therefore we would reject the modified null hypothesis and conclude that there is a maternal influence on offspring blood pH. On the other hand, the interval for the effect of sires does appear to include 0 and thus we would conclude that there is no significant paternal influence on blood pH. It is also evident that the maternal influence on female offspring blood pH is stronger than the paternal influence.

Example 11D: Three factor mixed model nested ANOVA

Sokal and Rohlf (1997) demonstrate the analysis of a balanced three factor nested ANOVA design in which the glycogen levels had been measured from two separate readings from each of three liver preparations from each of two individual rats per one of three different treatments (which they did not elaborate on). In this case, the treatments represent the fixed Factor A, the individual rats represent the first random nesting factor (Factor B and therefore the replicates of the treatment effects) and liver preparations represent an additional random nesting factor (Factor C). The duplicate readings from each liver, are the units of replication for the preparations.

Presumably, the researchers would have been primarily interested in whether there was an effect of treatment on liver glycogen content. The design acknowledges that individual liver preparations and glycogen readings as well as the individual rats are themselves likely to be of substantially great enough variability with respect to glycogen measurements that they could potentially mask the ability to detect an impact of treatment – hence the use of a nested design[m].

Step 1 - Import (section 2.3) the liver glycogen data set

```
> glyco <- read.table("glyco.csv", header = T, sep = ",")
```

Recall that `read.table()` automatically alphabetises the order of factor levels (hence in this case: `Compound217, Compound217Sugar, Control`) and defines treatment contrasts. For treatment contrasts to be meaningful in this case, the order of factor levels should be `Control, Compound217, Compound217Sugar`.

```
> glyco$TREAT <- factor(glyco$TREAT, levels = c("Control",
+     "Compound217", "Compound217Sugar"))
```

[m] Additionally, a nested design substantially reduces the number of rats required for the experiment.

Step 2 (Key 11.2) - Assess assumptions of normality and homogeneity of variance for each null hypotheses ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 11.1). Note that for each hypothesis test there are only either two or three replicates, and thus it is virtually impossible to confidently examine the assumptions. Instead, we must rely on the robustness of the test for a balanced design. As a result, I will only illustrate the process of producing the appropriate aggregated data sets for each hypothesis test.

1. Factor A (treatment - fixed factor). The mean glycogen levels per rat are the replicates for the treatment effects, and thus an aggregated dataset needs to be created from which the boxplots can be based.

```
> library(nlme)
> glyco.treat.agg <- gsummary(glyco, groups = glyco$RAT)
```

2. Factor B (rats - random factor). The mean glycogen levels per liver preparation are the replicates for the contributions of rats to added variation.

```
> glyco.rat.agg <- gsummary(glyco, groups = glyco$PREP)
```

3. Factor C (preparations - random factor). The mean glycogen levels per duplicate reading are the replicates for the contributions of the preparations to added variation. Note that in this case, since the individual readings are the lowest level of sub-replication, the aggregated dataset is the same as the original.

```
> glyco.prep.agg <- gsummary(glyco, groups = glyco$READ)
```

Step 3 (Key 11.4) - Assess whether the design is balanced (are there equal sample sizes in each treatment).

```
> library(biology)
> is.balanced(GLYCO ~ TREAT + RAT + PREP, data = glyco)
[1] TRUE
```

Conclusions - the design is balanced.

Step 4 (Key 11.5a) - fit one or more linear models to test the null hypotheses that there are no effects of treatment and no added variation due to rats within treatments and preparations within rats within treatments. As this is a balanced design, all three parametric model fitting procedures (`aov`, ANOVA from aggregated data sets and linear mixed effects models) will yield equivalent outcomes.

1. Factor A (treatment - fixed factor)

```
> glyco.aov <- aov(GLYCO ~ TREAT + Error(RAT/PREP), glyco)
> summary(glyco.aov)
Error: RAT
          Df  Sum Sq Mean Sq F value Pr(>F)
TREAT      2 1557.56  778.78   2.929 0.1971
Residuals  3  797.67  265.89
```

```
Error: RAT:PREP
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 12  594.0    49.5

Error: Within
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 18 381.00   21.17
```

2. Factor B (rats - random factor). Ignore the test of treatment from this output.

```
> glyco.rat.aov <- aov(GLYCO ~ TREAT + RAT + Error(PREP),
    glyco.rat.agg)
> summary(glyco.rat.aov)
Error: PREP
          Df Sum Sq Mean Sq F value    Pr(>F)
TREAT      2 778.78  389.39 15.7329 0.0004428 ***
RAT        3 398.83  132.94  5.3715 0.0141091 *
Residuals 12 297.00   24.75
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3. Factor C (preparations - random factor). Ignore the tests of treatment and rat from this output.

```
> glyco.prep.aov <- aov(GLYCO ~ TREAT + RAT + PREP,
    glyco.prep.agg)
> summary(glyco.prep.aov)
            Df  Sum Sq Mean Sq F value    Pr(>F)
TREAT        2 1557.56  778.78 36.7927 4.375e-07 ***
RAT          3  797.67  265.89 12.5617 0.0001143 ***
PREP        12  594.00   49.50  2.3386 0.0502907 .
Residuals   18  381.00   21.17
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Treatments were not found to have an impact on the glycogen content of rat livers ($F_{2,3} = 2.929$, $P = 0.197$). Liver glycogen content varies significantly between rats ($F_{3,12} = 5.372$, $P = 0.014$), but only marginally between liver preparations $F_{12,18} = 2.339$, $P = 0.050$). Alternatively, we could use a linear mixed effects model to investigate the effect of treatment and examine the variance components. As the design is balanced, the `lme()` function is perhaps more preferable to many workers (than the `lmer()` function) as it provides an F-ratio and P-value **(Key 11.5c)**

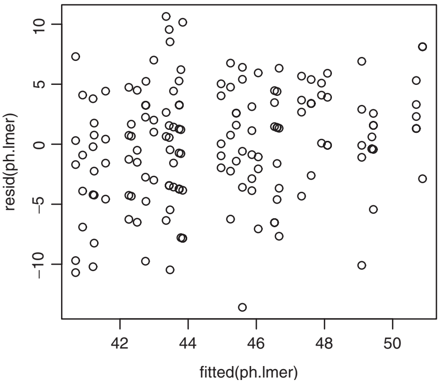
```
> library(nlme)
> glyco.lme <- lme(GLYCO ~ TREAT, random = ~1 | RAT/PREP, glyco)
> summary(glyco.lme)
Linear mixed-effects model fit by REML
 Data: glyco
```

```
       AIC      BIC    logLik
  231.6213 240.6003 -109.8106

Random effects:
 Formula: ~1 | RAT
        (Intercept)
StdDev:    6.005399

 Formula: ~1 | PREP %in% RAT
        (Intercept) Residual
StdDev:    3.763863 4.600725

Fixed effects: GLYCO ~ TREAT
                          Value Std.Error DF   t-value p-value
(Intercept)           140.50000  4.707166 18 29.848111  0.0000
TREATCompound217       10.50000  6.656937  3  1.577302  0.2128
TREATCompound217Sugar  -5.33333  6.656937  3 -0.801169  0.4816
 Correlation:
                      (Intr) TREATCm217
TREATCompound217      -0.707
TREATCompound217Sugar -0.707  0.500

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.48211987 -0.47263005  0.03061539  0.42934293  1.82934636

Number of Observations: 36
Number of Groups:
          RAT PREP %in% RAT
            6            18

> anova(glyco.lme)
            numDF denDF  F-value p-value
(Intercept)     1    18 2738.654  <.0001
TREAT           2     3    2.929  0.1971

> library(nlme)
> VarCorr(glyco.lme)

            Variance     StdDev
RAT =       pdLogChol(1)
(Intercept) 36.06482     6.005399
PREP =      pdLogChol(1)
(Intercept) 14.16667     3.763863
Residual    21.16667     4.600725
```

Conclusions - Again, treatments were not found to have an impact on the glycogen content of rat livers ($F_{2,3} = 2.929, P = 0.197$). The variability in liver glycogen content is greater between the individual rats than it is between preparations within the rats.
Yet another alternative is to employ the newer generalized mixed effects modelling procedure (`lmer`) **(Key 11.5d)**. Although this will not produce *F*-ratios, *P*-values for fixed effects can be determined from a sampling distribution generated via Markov Chain Monte Carlo techniques[n].

```
> library(lme4)
> glyco.lmer <- lmer(GLYCO ~ TREAT + (1 | RAT/PREP), glyco)
```

```
> plot(resid(ph.lmer) ~ fitted(ph.lmer))
```



Conclusions - no evidence of a wedge or other pattern in the residuals.

```
> glyco.lmer
Linear mixed model fit by REML
Formula: GLYCO ~ TREAT + (1 | RAT/PREP)
   Data: glyco
   AIC   BIC logLik deviance REMLdev
 231.6 241.1 -109.8    234.3   219.6
Random effects:
 Groups   Name        Variance Std.Dev.
 PREP:RAT (Intercept) 14.167   3.7639
 RAT      (Intercept) 36.065   6.0054
 Residual             21.167   4.6007
Number of obs: 36, groups: PREP:RAT, 18; RAT, 6

Fixed effects:
                      Estimate Std. Error t value
(Intercept)            140.500      4.707  29.850
TREATCompound217        10.500      6.656   1.577
TREATCompound217Sugar   -5.333      6.656  -0.801
```

[n] Markov chain Monte Carlo procedures in this context generate samples of model parameters via randomizations of Markov chains. which themselves represent states or estimates by incorporating previous states or estimates.

```
Correlation of Fixed Effects:
            (Intr) TREATCm217
TREATCmp217 -0.707
TREATCm217S -0.707  0.500
```

Conclusions - The conclusions about the sources of variability are the same as previous (greater variability between rats than between preparations). Note that degrees of freedom and P values are intentionally omitted from the output since (arguably) sensible values are not identifiable by traditional techniques.

Employ Markov chain Monte Carlo (MCMC) sampling methods to generate distributions of each of the parameter estimates from which confidence intervals and P values[o] can be calculated. Markov chain Monte Carlo sampling is performed using the recently updated `mcmcsamp` *function*. These techniques are at the bleeding edge of theoretical and practical statistics and the author of this function stresses that it is currently displaying some peculiar behaviour and should not yet be trusted. Nevertheless, I will include it as these teething issues are likely to be rectified in the near future.

```
> library(languageR)
> glyco.pval <- pvals.fnc(glyco.lmer, nsim = 10000, withMCMC = T)
```

Examine the fixed effects

```
> glyco.pval$fixed
                      Estimate MCMCmean HPD95lower HPD95upper  pMCMC Pr(>|t|)
(Intercept)            140.500  140.501   133.4425     147.54 0.0001   0.0000
TREATCompound217        10.500   10.507     0.3542      20.20 0.0398   0.1242
TREATCompound217Sugar   -5.333   -5.392   -15.2432       4.74 0.2386   0.4287
```

Examine the random effects

```
> glyco.pval$random
    Groups        Name Std.Dev. MCMCmedian MCMCmean HPD95lower HPD95upper
1 PREP:RAT (Intercept)   3.7639     0.8526   1.0771     0.0000     3.1076
2      RAT (Intercept)   6.0054     3.7633   3.9243     0.0000     6.9293
3 Residual               4.6007     6.0172   6.1119     4.4933     7.8493
```

Conclusions - The output would suggest that (based on MCMC P values) whilst there was no evidence that liver glycogen levels associated with the Compound217sugar treatment are not different to those of the control, there is some evidence that the levels are higher when associated with the Compound217 treatment. Note that the significant P value (0.0398) resulting from the MCMC sampling is suspiciously low, particularly when we consider that it is lower than the included anti-conservative P value (0.1242).

Examine the null hypothesis that there is no overall treatment effect (via MCMC sampling).

```
> glyco.mcmc <- glyco.pval$mcmc
> library(biology)
> mcmcpvalue(as.matrix(glyco.mcmc), "TREAT")
[1] 0.017
```

Conclusions - This *P*-value is based on the current implementation of MCMC sampling and thus is presently suspect.

[o] Note that the calculation of P values is contrary to the general Bayesian philosophy on which these methods are based and it is therefore an unsupported pursuit.

12

Factorial ANOVA

Factorial designs are an extension of single factor ANOVA designs in which additional factors are added such that each level of one factor is applied to all levels of the other factor(s) and all combinations are replicated (see Figure 12.1). For example, we might design an experiment in which the effects of temperature (high vs low) and fertilizer (added vs not added) on the growth rate of seedlings are investigated by growing seedlings under the different temperature and fertilizer combinations. In addition to investigating the impacts of the main factors, factorial designs allow us to investigate whether the effects of one factor are consistent across levels of another factor. For example, is the effect of temperature on growth rate the same for both fertilized and unfertilized seedlings and similarly, does the impact of fertilizer treatment depend on the temperature under which the seedlings are grown?

To appreciate the interpretation of interactions, consider the following figures that depict fictitious two factor (temperature and fertilizer) designs. For Figure 12.2a, it is clear that whether or not there is an observed effect of adding fertilizer or not depends on whether we are focused on seedlings growth under high or low temperatures. Fertilizer is only important for seedlings grown under high temperatures. In this case it is not possible to simply state that there is an effect of fertilizer, as it depends on the level of temperature. Similarly, the magnitude of the effect of temperature depends on whether fertilizer has been added or not. Such interactions are represented by plots in which lines either intersect or converge. Figure 12.2b-c both depict parallel lines which are indicative of no interaction. That is, the effects of temperature are similar for both fertilizer added and controls and vice versa. Whilst the former displays an effect of both fertilizer and temperature, in the latter, only fertilizer is important. Finally, Figure 12.2d represents a strong interaction that would mask the main effects of temperature and fertilizer (since the nature of the effect of temperature is very different for the different fertilizer treatments and visa versa).

Factorial designs can consist entirely of fixed (see section 10.0.1) factors (**Model I ANOVA**) in which conclusions are restricted to the specific combinations of levels selected for the experiment, entirely of random factors (**Model II ANOVA**) or a mixture of fixed and random factors (**Model III ANOVA**). The latter are useful for investigating the generality of a main treatment effect (fixed) over broad spatial, temporal or biological levels of organization. That is, whether the observed effects of

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

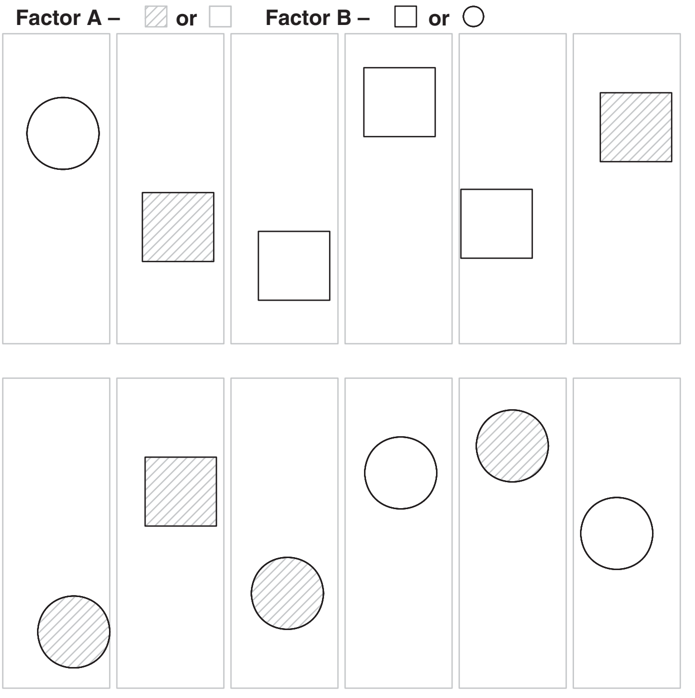


Fig 12.1 Fictitious spatial depictions of a multi (two) factor ANOVA design. There are two levels of factor A (shaded or not) and two levels of factor B (square or circle) and three replicates of each shape/fill combination.

temperature and/or fertilizer (for example) are observed across the entire genera or country.

12.1 Linear models

The linear models for two and three factor designs are:

$$y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \varepsilon_{ijk}$$
$$y_{ijkl} = \mu + \alpha_i + \beta_j + \gamma_k + (\alpha\beta)_{ij} + (\alpha\gamma)_{ik} + (\beta\gamma)_{jk} + (\alpha\beta\gamma)_{ijk} + \varepsilon_{ijkl}$$

where μ is the overall mean, α is the effect of Factor A, β is the effect of Factor B, γ is the effect of Factor C and ε is the random unexplained or residual component. Note that although the linear models for Model I, Model II and Model III designs are identical, the interpretation of terms (and thus null hypothesis) differ.

12.2 Null hypotheses

There are separate null hypothesis associated with each of the main effects and the interaction terms.

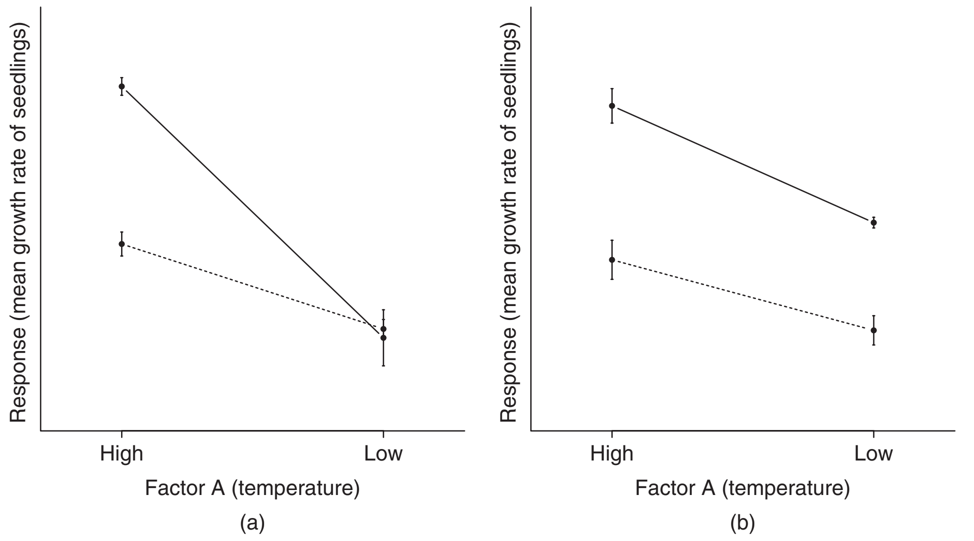


Fig 12.2 Fictitious depictions of two factor ANOVA design. There are two levels of factor A (temperature: High and Low) and two levels of factor B (fertilizer: Added or not added).

12.2.1 Model 1 - fixed effects

Factor A

$$H_0(A): \mu_1 = \mu_2 = \cdots = \mu_i = \mu \qquad \text{(the population group means are all equal)}$$

The mean of population 1 is equal to that of population 2 and so on, and thus all population means are equal to an overall mean. If the effect of the i^{th} group is the difference between the i^{th} group mean and the overall mean ($\alpha_i = \mu_i - \mu$) then the

H_0 can alternatively be written as:

$$H_0(A): \alpha_1 = \alpha_2 = \cdots = \alpha_i = 0 \qquad \text{(the effect of each group equals zero)}$$

If one or more of the α_i are different from zero (the response mean for this treatment differs from the overall response mean), the null hypothesis is not true, indicating that the treatment does affect the response variable.

Factor B

$$H_0(B): \mu_1 = \mu_2 = \cdots = \mu_i = \mu \qquad \text{(the population group means are all equal)}$$

Equivalent interpretation to Factor A above.

A:B Interaction

$$H_0(AB): \mu_{ij} = \mu_i + \mu_j - \mu \qquad \text{(the population group means are all equal)}$$

For any given combination of factor levels, the population group mean will be equal to the difference between the overall population mean and the simple additive effects of the individual factor group means. That is, the effects of the main treatment factors are purely additive and independent of one another. This is equivalent to $H_0(AB)$: $\alpha\beta_{ij} = 0$, no interaction between Factor A and Factor B.

12.2.2 Model 2 - random effects

Factor A

$$H_0(A): \sigma_\alpha^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of A.

Factor B

$$H_0(B): \sigma_\alpha^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of B.

A:B Interaction

$$H_0(AB): \sigma_{\alpha\beta}^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible interactions between all possible levels of A and B.

12.2.3 Model 3 - mixed effects

Fixed factor - e.g. A

$$H_0(A): \mu_1 = \mu_2 = \cdots = \mu_i = \mu \qquad \text{(the population group means are all equal)}$$

The mean of population 1 (pooled over all levels of the random factor) is equal to that of population 2 and so on, and thus all population means are equal to an overall mean pooling over all possible levels of the random factor. If the effect of the i^{th} group is the difference between the i^{th} group mean and the overall mean ($\alpha_i = \mu_i - \mu$) then the H_0 can alternatively be written as:

$$H_0(A): \alpha_1 = \alpha_2 = \cdots = \alpha_i = 0 \qquad \text{(no effect of any level of this factor pooled over all possible levels of the random factor)}$$

Random factor - e.g. B

$$H_0(B): \sigma^2_\alpha = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of B.

A:B Interaction

The interaction of a fixed and random factor is always considered a random factor.

$$H_0(AB): \sigma^2_{\alpha\beta} = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible interactions between all possible levels of A and B.

12.3 Analysis of variance

When fixed factorial designs are balanced, the total variance in the response variable can be sequentially partitioned into what is explained by each of the model terms (factors and their interactions) and what is left unexplained. For each of the specific null hypotheses, the overall unexplained variability is used as the denominator in F-ratio calculations (see Tables 12.1 & 12.2), and when a null hypothesis is true, an F-ratio should follow an F distribution with an expected value less than 1.

Random factors are added to provide greater generality of conclusions. That is, to enable us to make conclusions about the effect of one factor (such as whether or not fertilizer is added) over all possible levels (not just those sampled) of a random factor (such as all possible locations, seasons, varieties etc). In order to expand our conclusions beyond the specific levels used in the design, the hypothesis tests (and thus F-ratios)

Table 12.1 *F*-ratio determination and general R syntax two factor factorial designs.

	Factor	d.f	A&B fixed	A&B random	A fixed, B random	
					Restricted[a]	Unrestricted
1	A	$a-1$	$\frac{MS_A}{MS_{Resid}}$	$\frac{MS_{A'}}{MS_{B'\times A'}}$	$\frac{MS_A}{MS_{B'} \times A}$	$\frac{MS_A}{MS_{B'\times A}}$
2	B	$b-1$	$\frac{MS_B}{MS_{Resid}}$	$\frac{MS_{B'}}{MS_{B'\times A'}}$	$\frac{MS_{B'}}{MS_{Resid}}$	$\frac{MS_{B'}}{MS_{AB'}}$
3	B×A	$(b-1)(a-1)$	$\frac{MS_{B\times A}}{MS_{Resid}}$	$\frac{MS_{B'\times A'}}{MS_{Resid}}$	$\frac{MS_{B'\times A}}{MS_{Resid}}$	$\frac{MS_{B'\times A}}{MS_{Resid}}$
4	Residual (=N′(B×A))	$(n-1)ba$				

	R syntax[b]
Type I SS (Balanced)	`> anova(aov(DV~A*B, data))`
Type II SS (Unbalanced)	`> Anova(aov(DV~A*B, data), type="II"))`
Type III SS (Unbalanced)[c]	`> Anova(aov(DV~A*B, data), type="III"))`
Variance components[d]	`> lmer(DV~1+(1|B)+(1|A)+(1|A:B), data)`

[a]Typically only for balanced designs.
[b]Mixed models require manual *F*-ratio and *P*-value calculations.
[c]To use Type III sums of squares, random factors need to be defined as something other than 'treatment' (e.g. 'helmert' or 'sum') contrasts prior to fitting the model.

```
> contrasts(data$B)<-contr.helmert
```

[d]Note this uses the REML method and is therefore valid for balanced and unbalanced designs, but will yield slightly different estimates than simple formulae used for purely balanced designs.

Table 12.2 *F*-ratio determination and general R syntax two and three factor factorial designs.

			A,B&C	A,B&C	A&C fixed, B random		A fixed, B&C random	
	Factor	d.f	fixed	random	Restricted[a]	Unrestricted	Restricted[a]	Unrestricted
1	A	$a-1$	1/8	1/(3+5-7)	1/3	1/3	1/(3+5-7)	1/(3+5-7)
2	B	$b-1$	2/8	2/(3+6-7)	2/8	?	2/6	?
3	B×A	$(b-1)(a-1)$	3/8	3/7	3/8	3/7	3/7	3/7
4	C	$(c-1)$	4/8	4/(5+6-7)	4/6	4/6	4/6	?
5	C×A	$(c-1)(a-1)$	5/8	5/7	5/7	5/7	5/7	5/7
6	C×B	$(c-1)(b-1)$	6/8	6/7	6/8	6/7	6/8	6/7
7	C×B×A	$(c-1)(b-1)(a-1)$	7/8	7/8	7/8	7/8	7/8	7/7
8	Residual (=N′(C×B×A))	$(n-1)cba$						

	***R* syntax**[b]
Type I SS (Balanced)	`> anova(aov(DV~A*B*C, data))`
Type II SS (Unbalanced)	`> Anova(aov(DV~A*B*C, data), type="II")`
Type III SS (Unbalanced)[c]	`> Anova(aov(DV~A*B*C, data), type="III")`
Variance components[d]	`> lmer(DV~1+(1|C)+(1|B)+(1|A)+(1|A:B), data)`

[a]Typically only for balanced designs.
[b]Mixed models require manual *F*-ratio and *P*-value calculations.
[c]To use Type III sums of squares, random factors need to be defined as something other than 'treatment' (e.g. 'helmert' or 'sum') contrasts prior to fitting the model.

```
> contrasts(data$B)<-contr.helmert
```

[d]Note this uses the REML method and is therefore valid for balanced and unbalanced designs, but will yield slightly different estimates than simple formulae used for purely balanced designs.

must reflect this extra generality by being more conservative. The appropriate[a] F-ratios for fixed, random and mixed factorial designs are presented in Tables 12.1 & 12.2. Generally, once the terms (factors and interactions) have been ordered into a hierarchy (single factors at the top, highest level interactions at the bottom and terms of same order given equivalent positions in the hierarchy), the denominator for any term is selected as the next appropriate random term (an interaction that includes the term to be tested) encountered lower in the hierarchy. Interaction terms that contain one or more random factors are considered themselves to be random terms, as is the overall residual term (as all observations are assumed to be random representations of the entire population(s)).

Pooling of non-significant F-ratio denominator terms (see section 11.6), in which lower random terms are added to the denominator (provided $\alpha > 0.25$), may also be useful.

For random factors within mixed models, selecting F-ratio denominators that are appropriate for the intended hypothesis tests is a particularly complex and controversial issue. Traditionally, there are two alternative approaches and whilst the statistical resumes of each are complicated, essentially they differ in whether or not the interaction term is constrained for the test of the random factor. The constrained or restricted method (Model I), stipulates that for the calculation of a random factor F-ratio (which investigates the added variance added due to the random factor), the overall effect of the interaction is treated as zero. Consequently, the random factor is tested against the residual term (see Tables 12.1 & 12.2). The unconstrained or unrestrained method (Model II) however, does not set the interaction effect to zero and therefore the interaction term is used as the random factor F-ratio denominator (see Tables 12.1 & 12.2). This method assumes that the interaction terms for each level of the random factor are completely independent (correlations between the fixed factor must be consistent across all levels of the random factor). Some statisticians maintain that the independence of the interaction term is difficult to assess for biological data and therefore, the restricted approach is more appropriate. However, others have suggested that the restricted method is only appropriate for balanced designs.

12.3.1 Quasi F-ratios

An additional complication for three or more factor models that contain two or more random factors, is that there may not be a single appropriate interaction term to use as the denominator for many of the main effects F-ratios. For example, if Factors A and B are random and C is fixed, then there are two random interaction terms of equivalent level under Factor C ($A' \times C$ and $B' \times C$). As a result, the value of the of the Mean Squares expected when the nul hypothesis is true cannot be easily defined. The solutions for dealing with such situations (quasi F-ratios[b]) involve adding (and subtracting) terms together to create approximate estimates of F-ratio denominators. These solutions are

[a] When designs include a mixture of fixed and random crossed effects, exact demoninators for certain F-ratios are undefined and traditional approaches adopt rather inexact estimated approximate or "Quasi" F-ratios.

[b] Alternatively, for random factors, variance components with confidence intervals can be used.

sufficiently unsatisfying as to lead many biostatisticians to recommend that factorial designs with two or more random factors should avoided if possible. Arguably however, **linear mixed effects models** (see section 11.8) offer more appropriate solutions to the above issues as they are more robust for unbalanced designs, accommodate covariates and provide a more comprehensive treatment and overview of all the underlying data structures.

12.3.2 Interactions and main effects tests

Note that for fixed factor models, when null hypotheses of interactions are rejected, the null hypothesis of the individual constituent factors are unlikely to represent the true nature of the effects and thus are of little value. The nature of such interactions are further explored by fitting simpler linear models (containing at least one less factor) separately for each of the levels of the other removed factor(s). Such **Main effects tests** are based on a subset of the data, and therefore estimates of the overall residual (unexplained) variabilty are unlikely to be as precise as the estimates based on the global model. Consequently, F-ratios involving MS_{Resid} should use the estimate of MS_{Resid} from the global model rather than that based on the smaller, theoretically less precise subset of data. For random and mixed models, since the objective is to generalize the effect of one factor *over and above* any interactions with other factors, the main factor effects can be interpreted even in the presence of significant interactions[c].

12.4 Assumptions

Hypothesis tests assume that the residuals are:

(i) normally distributed. Boxplots using the appropriate scale of replication (reflecting the appropriate residuals/F-ratio denominator (see Tables 12.1 & 12.2)) should be used to explore normality. Scale transformations are often useful.
(ii) equally varied. Boxplots and plots of means against variance (using the appropriate scale of replication) should be used to explore the spread of values. Residual plots should reveal no patterns (see Figure 8.5). Scale transformations are often useful.
(iii) independent of one another.

12.5 Planned and unplanned comparisons

As with single factor analysis of variance, planned[d] and unplanned multiple comparisons (such as Tukey's test) can be incorporated into or follow the linear model

[c] Although it should be noted that when a significant interaction is present in a mixed model, the power of the main fixed effects will be reduced (since the amount of variability explained by the interaction term will be relatively high, and this term is used as the denominator for the F-ratio calculation, see Table 12.1).
[d] As with single factor analysis of variance, the contrasts must be defined prior to fitting the linear model, and no more than $p - 1$ (where p is the number of levels of the factor) contrasts can be defined for a factor.

respectively so as to further investigate any patterns or trends within the main factors and/or the interactions (see section 10.6).

12.6 Unbalanced designs

A factorial design can be thought of as a table made up of rows (representing the levels of one factor), columns (levels of another factor) and cells (the individual combinations of the set of factors), see Table 12.3(a). Table 12.3(b) depicts a balanced two factor (3x3) design in which each cell (combination of factor levels) has three replicate observations. Whilst Table 12.3(c) does not have equal sample sizes in each cell, the sample sizes are in proportion and as such, does not present the issues discussed below for unbalanced designs. Tables 12.3(d) & (e), are considered unbalanced.

12.6.1 Missing observations

In addition to impacting on normality and homogeneity of variance, unequal sample sizes in factorial designs have major implications for the partitioning of the total sums of squares into each of the model components.

For balanced designs, the total sums of squares (SS_{Total}) is equal to the additive sums of squares of each of the components (including the residual). For example, in a two factor balanced design, $SS_{Total} = SS_A + SS_B + SS_{AB} + SS_{Resid}$. This can be represented diagrammatically by a Venn Diagram (see Figure 12.3) in which each of the *SS* for the term components butt against one another and are surrounded by the SS_{Resid} (see Figure 12.2a). However, in unbalanced designs, the sums of squares will be nonorthogonal and the sum of the individual components does not add up to the total sums of squares. Diagrammatically, the SS of the terms intersect or are separated (see Figure 12.2b and 12.2g respectively). In regular **sequential sums of squares (Type I SS)**, the sum of the individual sums of squares must be equal to the total sums of squares, the sums of squares of the last factor to be estimated will be calculated as the difference between the total sums of squares and what has already been accounted for by other components. Consequently, the order in which factors are specified in the model (and thus estimated) will alter their sums of squares and therefore their *F*-ratios (see Figure 12.2c-d).

To overcome this problem, traditionally there are two other alternative methods of calculating sums of squares. **Type II (hierarchical) SS** estimate the sums of squares of each term as the improvement it contributes upon the addition of that term to a model of greater complexity and lower in the hierarchy (recall that the hierarchical structure descends from the simplest model down to the fully populated model). The SS for the interaction as well as the first factor to be estimated are the same as for Type I SS. Type II SS estimate the contribution of a factor over and above the contributions of other factors of equal or lower complexity but not above the contributions of the interaction terms or terms nested within the factor (see Figure 12.3e & 12.3k). However, these sums of squares are weighted by the sample sizes of each level and

Table 12.3 Factorial cell means structure (a) for a fictitious two factor design (effect of Temperature: high, medium or low, and Shading: full, partial or control on seedling growth) illustrating (b) balanced, (c) proportionally balanced, (d-e) unbalanced and (f) missing cells designs. For the missing cell example, in which one combination or cell is missing (perhaps seedlings grown under these conditions all died), three alternative sets of that can be used to estimate individual factor effects for factor A and B are listed in subfigures (g) and (h) respectively. Gray coefficients indicate coefficients to be omitted when cell FL is missing (as an example) and coefficients in brackets are replacement coefficients relevant for the missing cell example. Similarly, interaction effects are estimated from one of four alternative contrast sets (i). Note that cell means contrasts are not orthogonal and therefore the individual hypotheses tests should be ignored (SS will differ substantially according to the order in which the contrasts are defined). They are used purely to establish the overall factor and interaction effects.

(a) Cell means structure

	High	Medium	Low
Full shade	μ_{FH}	μ_{FM}	μ_{FL}
Partial shade	μ_{PH}	μ_{PM}	μ_{PL}
Control	μ_{CH}	μ_{CM}	μ_{CL}

(b) Balanced design (3 replicates)

	High	Medium	Low
Full shade	XXX	XXX	XXX
Partial shade	XXX	XXX	XXX
Control	XXX	XXX	XXX

(c) Proportionally balanced design (2-3 replicates)

	High	Medium	Low
Full shade	XXX	XXX	XXX
Partial shade	XX	XX	XX
Control	XXX	XXX	XXX

(d) Unbalanced design (2-3 replicates)

	High	Medium	Low
Full shade	XX	XXX	XXX
Partial shade	XXX	XXX	XXX
Control	XXX	XXX	XXX

(e) Unbalanced design (1-3 replicates)

	High	Medium	Low
Full shade	XX	XXX	XXX
Partial shade	XXX	X	XX
Control	XXX	XXX	XXX

(f) Missing cells design (3 replicates)

	High	Medium	Low
Full shade	XXX	XXX	
Partial shade	XXX	XXX	XXX
Control	XXX	XXX	XXX

(g) Factor A (Shade) contrasts

	FH	FM	FL	PH	PM	PL	CH	CM	CL
Set 1									
$H_0: \mu_F = \mu_P$	1	1	1	−1	−1	−1 (0)	0	0	0
$H_0: \mu_P = \mu_C$	0	0	0	1	1	1	−1	−1	−1
Set 2									
$H_0: \mu_F = \mu_P$	1	1	1	−1	−1	−1 (0)	0	0	0
$H_0: \mu_F = \mu_C$	1	1	1	0	0	0	−1	−1	−1 (0)
Set 3									
$H_0: \mu_F = \mu_C$	1	1	1	0	0	0	−1	−1	−1 (0)
$H_0: \mu_P = \mu_C$	0	0	0	1	1	1	−1	−1	−1

(h) Factor B (Temperature) contrasts

	FH	**FM**	FL	**PH**	**PM**	**PL**	**CH**	**CM**	**CL**
Set 1									
$H_0: \mu_H = \mu_M$	**1**	**−1**	0	**1**	**−1**	**0**	**1**	**−1**	**0**
$H_0: \mu_M = \mu_L$	**0**	1 **(0)**	**−1**	**0**	**1**	**−1**	**0**	**1**	**−1**
Set 2									
$H_0: \mu_M = \mu_L$	**0**	1 **(0)**	**−1**	**0**	**1**	**−1**	**0**	**1**	**−1**
$H_0: \mu_H = \mu_L$	1 **(0)**	**0**	**−1**	**1**	**0**	**−1**	**1**	**0**	**−1**
Set 3									
$H_0: \mu_H = \mu_L$	1 **(0)**	**0**	**−1**	**1**	**0**	**−1**	**1**	**0**	**−1**
$H_0: \mu_M = \mu_L$	**0**	1 **(0)**	**−1**	**0**	**1**	**−1**	**0**	**1**	**−1**

Table 12.3 (*continued*)

(i) AB interaction contrasts

Effects of A at each level of B

	FH	FM	FL	PH	PM	PL	CH	CM	CL
Set 1									
$H_0: \mu_{FH}+\mu_{PM}=\mu_{PH}+\mu_{FM}$	1	−1	0	−1	1	0	0	0	0
$H_0: \mu_{FM}+\mu_{PL}=\mu_{PM}+\mu_{FL}$	0	1	−1	0	−1	1	0	0	0
$H_0: \mu_{PH}+\mu_{CM}=\mu_{CH}+\mu_{PM}$	0	0	0	1	−1	0	−1	1	0
$H_0: \mu_{PM}+\mu_{CL}=\mu_{CM}+\mu_{PL}$	0	0	0	0	1	−1	0	−1	1
Set 2									
$H_0: \mu_{FH}+\mu_{PM}=\mu_{PH}+\mu_{FM}$	1	−1	0	−1	1	0	0	0	0
$H_0: \mu_{FH}+\mu_{PL}=\mu_{PH}+\mu_{FL}$	1	0	−1	−1	0	1	0	0	0
$H_0: \mu_{FH}+\mu_{CM}=\mu_{CH}+\mu_{FM}$	1	−1	0	0	0	0	−1	1	0
$H_0: \mu_{FH}+\mu_{CL}=\mu_{CH}+\mu_{FL}$	1	0	−1	0	0	0	−1	0	1

Effects of B at each level of A

	FH	FM	FL	PH	PM	PL	CH	CM	CL
Set 3									
$H_0: \mu_{FH}+\mu_{PM}=\mu_{PH}+\mu_{FM}$	1	−1	0	−1	1	0	0	0	0
$H_0: \mu_{PH}+\mu_{CM}=\mu_{PM}+\mu_{CH}$	0	0	0	1	−1	0	1	−1	0
$H_0: \mu_{FM}+\mu_{PL}=\mu_{FL}+\mu_{PM}$	0	1	−1	0	−1	1	0	0	0
$H_0: \mu_{PM}+\mu_{CL}=\mu_{PL}+\mu_{CM}$	0	0	0	0	1	−1	0	−1	1
Set 4									
$H_0: \mu_{FH}+\mu_{PM}=\mu_{FM}+\mu_{PH}$	1	−1	0	−1	1	0	0	0	0
$H_0: \mu_{FH}+\mu_{CM}=\mu_{FM}+\mu_{CH}$	1	−1	0	0	0	0	−1	1	0
$H_0: \mu_{FH}+\mu_{PL}=\mu_{FL}+\mu_{PH}$	1	0	−1	−1	0	1	0	0	0
$H_0: \mu_{FH}+\mu_{CL}=\mu_{FL}+\mu_{CH}$	1	0	−1	0	0	0	−1	0	1

therefore are biased towards the trends produced by the groups (levels) that have higher sample sizes[e].

Type III (marginal or orthogonal) SS estimate the sums of squares of each term as the improvement based on a comparison of models with and without the term and are unweighted by sample sizes. Type III SS essentially measure just the unique contribution of each factor over and above the contributions of the other factors and interactions (see Figure 12.3f & 12.3l). For unbalanced designs,Type III SS essentially test equivalent hypotheses to balanced Type I SS and are therefore arguably more appropriate for unbalanced factorial designs than Type II SS. Importantly, Type III SS are only interpretable if they are based on orthogonal contrasts (such as sum or helmert contrasts and not treatment contrasts).

The choice between Type II and III SS clearly depends on the nature of the question. For example, if we had measured the growth rate of seedlings subjected to two factors (temperature and fertilizer), Type II SS could address whether there was an effect of temperature across the level of fertilizer treatment, whereas Type III SS could assess whether there was an effect of temperature within each level of the fertilizer treatment.

12.6.2 Missing combinations - missing cells

When an entire combination, or cell, is missing (perhaps due to unforeseen circumstances) it is not possible to test all the main effects and/or interactions. Table 12.3(f) depicts such as situation. One solution is to fit a large single factor ANOVA with as many levels as there are cells (this is known as a **cell means model**) and investigate various factor and interaction effects via specific contrasts (see Tables 12.3(g)-(j) and 12.4). Difficulties in establishing appropriate error terms, makes missing cells in random and mixed factor designs substantially more complex.

[e] As a result of the weightings, Type II SS actually test hypotheses about really quite complex combinations of factor levels. Rather than test a hypothesis that $\mu_{High} = \mu_{Medium} = \mu Low$, Type II SS might be testing that $4\mu_{High} = 1\mu_{Medium} = 0.25\mu Low$.

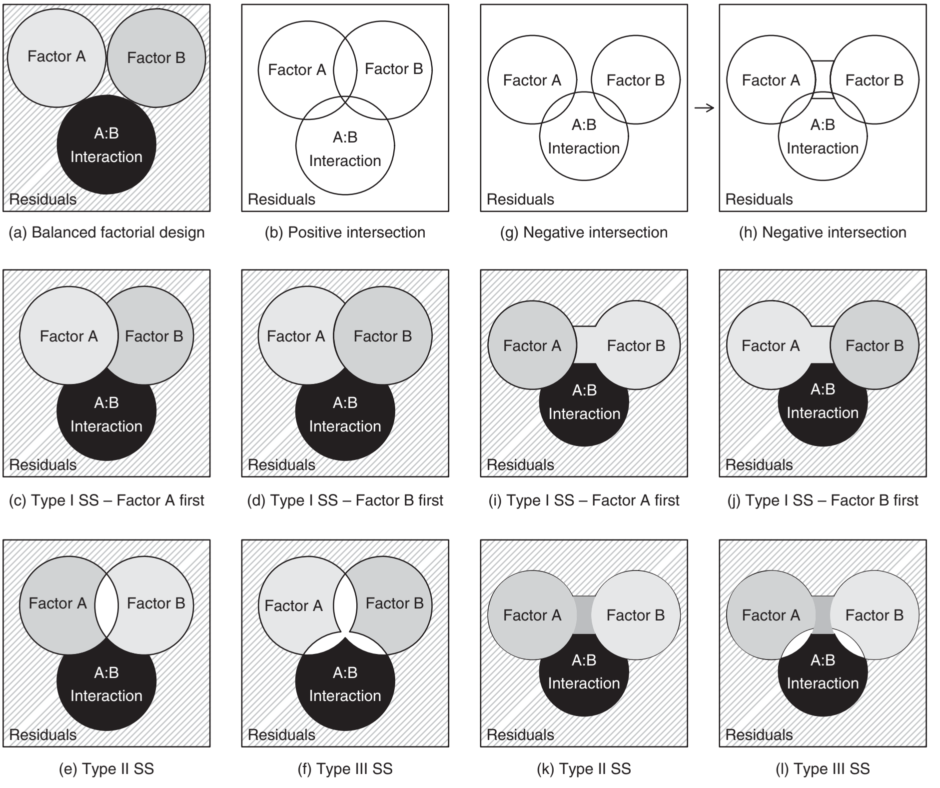


Fig 12.3 Fictitious representations of Type I, II and III Sums of Squares (SS) calculations for balanced and unbalanced two factor designs with positive (b-f) and negative (g-l) intersections. Striped pattern represents SS*resid*, shaded patterns represent SS for the respective terms and the white fill represent ignored areas. For completely balanced designs (a), the terms are all completely orthogonal or independent (no intersections) and thus Type I, II and III SS are identical. The Type I, II and III sums of squares for the interaction term for unbalanced two-factor designs are also identical. Type II SS for the main factors are the same as the Type I SS for the second factor calculated. When there are positive intersections between factors (factors are positively dependent), Type I SS for the first factor will be greater than its Type II estimate which in turn will be greater than its Type III estimate. For negative intersections (in which factors are negatively dependent), Type I SS for the first factor will be less than its Type II and III estimates. For such intersections, factors are joined by a *bridge* which is included in the SS calculations for each of the factors it joins. It is also possible to have bridges between factors and interaction terms, in which case Type III SS estimates can be substantially larger than Type I and II estimates. Note that intersections are not the same as interactions and the two issues are completely separate.

12.7 Robust factorial ANOVA

Factorial designs can be analysed as large single factor designs that incorporate specific sets of contrasts. Therefore, many of the robust or non-parametric techniques outlined in chapter 10.5 can be used to analyze factorial designs. Alternatively, standard factorial ANOVA can be performed on rank transformed data. This approach can also

Table 12.4 Cell means structure and associated contrasts for fictitious factorial designs (a) 3x2 one cell missing, (b) 3x3 two cells missing and (c) 3x3 three cells missing. Note that contrasts are not orthogonal and that in some cases the set of contrasts presented represents one of multiple possible combinations.

(a)

A \ B	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	μ_{22}	μ_{23}

Factor A

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	μ_{22}	μ_{23}

Factor B

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	μ_{22}	μ_{23}

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	μ_{22}	μ_{23}

AB interaction

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	μ_{22}	μ_{23}

	μ_{11}	μ_{12}	μ_{21}	μ_{22}	μ_{23}
Factor A					
(H_0: $\mu_{A1} = \mu_{A2}$)	**1**	**1**	**–1**	**–1**	**0**
Factor B					
(H_0: $\mu_{B1} = \mu_{B2}$)	**1**	**–1**	**1**	**–1**	**0**
(H_0: $\mu_{B1} = \mu_{B3}$)	**0**	**0**	**1**	**0**	**–1**
AB					
(H_0: $\mu_{A11} + \mu_{22} = \mu_{A21} + \mu_{21}$)	**1**	**–1**	**–1**	**1**	**0**

(b)

A \ B	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	-	μ_{23}
3	μ_{31}	μ_{32}	μ_{33}

Factor A

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	-	μ_{23}
3	μ_{31}	μ_{32}	μ_{33}

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	-	μ_{23}
3	μ_{31}	μ_{32}	μ_{33}

Factor B

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	-	μ_{23}
3	μ_{31}	μ_{32}	μ_{33}

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	-	μ_{23}
3	μ_{31}	μ_{32}	μ_{33}

AB interaction

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	-	μ_{23}
3	μ_{31}	μ_{32}	μ_{33}

	1	2	3
1	μ_{11}	μ_{12}	-
2	μ_{21}	-	μ_{23}
3	μ_{31}	μ_{32}	μ_{33}

	μ_{11}	μ_{12}	μ_{21}	μ_{23}	μ_{31}	μ_{32}	μ_{33}
Factor A							
(H_0: $\mu_{A1} = \mu_{A3}$)	**1**	**1**	**0**	**0**	**–1**	**–1**	**0**
(H_0: $\mu_{A2} = \mu_{A3}$)	**0**	**0**	**1**	**1**	**–1**	**0**	**–1**
Factor B							
(H_0: $\mu_{B2} = \mu_{B3}$)	**0**	**0**	**1**	**–1**	**1**	**0**	**–1**
(H_0: $\mu_{B1} = \mu_{B2}$)	**1**	**–1**	**0**	**0**	**1**	**–1**	**0**
AB interaction							
(H_0: $\mu_{A21} + \mu_{33} = \mu_{A23} + \mu_{31}$)	**1**	**–1**	**0**	**0**	**–1**	**1**	**0**

(c)

A \ B	1	2	3
1	μ_{11}	μ_{12}	μ_{13}
2	μ_{21}	μ_{22}	-
3	-	-	μ_{33}

Factor A

	1	2	3
1	μ_{11}	μ_{12}	μ_{13}
2	μ_{21}	μ_{22}	-
3	-	-	μ_{33}

	1	2	3
1	μ_{11}	μ_{12}	μ_{13}
2	μ_{21}	μ_{22}	-
3	-	-	μ_{33}

Factor B

	1	2	3
1	μ_{11}	μ_{12}	μ_{13}
2	μ_{21}	μ_{22}	-
3	-	-	μ_{33}

	1	2	3
1	μ_{11}	μ_{12}	μ_{13}
2	μ_{21}	μ_{22}	-
3	-	-	μ_{33}

AB interaction

	1	2	3
1	μ_{11}	μ_{12}	μ_{13}
2	μ_{21}	μ_{22}	-
3	-	-	μ_{33}

	μ_{11}	μ_{12}	μ_{13}	μ_{21}	μ_{22}	μ_{33}
Factor A						
(H_0: $\mu_{A1} = \mu_{A3}$)	**1**	**1**	**0**	**–1**	**–1**	**0**
(H_0: $\mu_{A2} = \mu_{A3}$)	**0**	**0**	**1**	**0**	**0**	**–1**
Factor B						
(H_0: $\mu_{B2} = \mu_{B3}$)	**1**	**–1**	**0**	**1**	**–1**	**0**
(H_0: $\mu_{B1} = \mu_{B2}$)	**1**	**0**	**–1**	**0**	**0**	**0**
AB interaction						
(H_0: $\mu_{A21} + \mu_{33} = \mu_{A23} + \mu_{31}$)	**1**	**1**	**0**	**–1**	**–1**	**0**

be extended to more complex designs, thereby providing a way to analyse unbalanced and mixed effects designs that display evidence of non-normality. Unfortunately, there is some evidence to suggest that testing interactions on rank transformed data can increase the Type I error rate. Furthermore, in the presence of significant main effects, the power to detect interaction effects is low.

Randomization tests (which are useful for situations in which observation independence could be questionable) can be performed by comparing the *F*-ratios (or mean squares) to a large number of *F*-ratios calculated from repeatedly shuffled data[f]. In so doing, randomization tests can accomodate random, fixed and mixed models as well as Type I, II and III SS and cell means models (for missing cells).

12.8 Power and sample sizes

Although power analyses for main effects within factorial designs follow the same principles as single factor designs, for interactions, it is very difficult to estimate the meaningful effect sizes due to the large number of factor level combinations. That said, the tests of interactions are typically more powerful than main effects (due to greater available degrees of freedom) and for fixed models, efforts to improve the power of any of the main effects will also benefit the corresponding interactions. Power analyses for mixed and random factorial designs should reflect the appropriate residuals (see Tables 12.1 & 12.2).

12.9 Factorial ANOVA in R

Fully factorial linear models are predominantly fitted using the `aov()` *function*. Anova tables for balanced, fixed factor designs can be viewed using either the `anova()` or `summary()`, the latter of which is used to accommodate planned contrasts with the `split=` *argument*. Type II and III sums of squares are estimated for unbalanced designs using either the `Anova()`[g] or `AnovaM()`[h] *functions*, the latter of which also accommodates planned contrasts (with the `split=` *argument*) as well as random and mixed models by enabling the appropriate *F*-ratio denominators to be defined via the `denoms=` *argument*.

12.10 Further reading

- Theory

  Doncaster, C. P., and A. J. H. Davey. (2007). *Analysis of Variance and Covariance. How to Choose and Construct Models for the Life Sciences*. Cambridge University Press, Cambridge.

---

[f] Although there are various ways in which the data or residuals could be shuffled, simulations suggest that they all yield very similar results.
[g] From the `car` *package*.
[h] From the `biology` *package*.

Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. 2 edition. John Wiley & Sons, New York.

Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

Sokal, R., and F. J. Rohlf. (1997). *Biometry*, 3rd edition. W. H. Freeman, San Francisco.

Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS*, 4th edn. Springer-Verlag, New York.

Wilcox, R. R. (2005). *Introduction to Robust Estimation and Hypothesis Testing*. Elsevier Academic Press.

12.11 Key for factorial ANOVA

1 **For each factor (categorical variable), establish whether it is to be considered a fixed or random factor**
- **Conclusions about the factor are restricted to the specific levels selected in the design.** Levels of the factor selected to represent the specific levels of interest (**fixed factor**)
- **Conclusions about the factor to be generalized across all possible levels of the factor.** Levels of the factor used represent a random selection of all the possible levels (**random factor**)

. . . . Go to 2

2 **Establish what sort of model it is and therefore what the appropriate *F*-ratio denominators apply (see Tables 12.1 & 12.2)**
- **All factors fixed (Model I)**
- **All factors random (Model II)**
- **Mixture of fixed and random factors (Model III)**

. . . . Go to 3

3 a. **Check assumptions for factorial ANOVA**
As the assumptions of any given hypothesis test relate to residuals, all diagnostics should reflect the appropriate error (residual) terms for the hypothesis. This is particularly important for random and mixed models where interaction terms might be the appropriate denominators (residuals).

- **Normality (symmetry) of the response variable (residuals) at each level of each factor or combination of factors - boxplots of mean values**

Fixed factor model (Model I) - using MS_{Resid} as denominator in each case

```
> boxplot(DV ~ A, data) #factor A
> boxplot(DV ~ B, data) #factor B
> boxplot(DV ~ A * B, data) #A:B interaction
```

Random or mixed model (Model II or III - factor B random) - using MS_{AB} as denominator as example

```
> library(nlme)
> data.AB.agg <- gsummary(data, groups = data$A:data$B)
> boxplot(DV ~ A, data.AB.agg) #factor A
```

where `DV` *is the response variable,* `A` *is a main fixed or random factor within the* `data` *dataset.*

- **Homogeneity (equality) of variance of the response variable (residuals) at each level of each factor or combination of factors - boxplots of mean values**
  As for Normality.

Parametric assumptions met . . . . Go to 5
b. Parametric assumptions not met . . . . Go to 4
4 a. Attempt a scale transformation (see Table 3.2 for transformation options) . . . . Go to 3
b. Transformations unsuccessful or inappropriate . . . . Go to 15
5 a. All factor combinations (cells) have at least one observation (no missing cells) . . . . Go to 6
b. One or more factor combinations without any observations (missing cells). Analyze as single factor cell means model. . . . . Go to 10
6 If incorporating planned contrasts (comparisons) . . . . . . See Examples 12A,12B,12C

```
> contrasts(data$A) <- cbind(c(contrasts), ...)
> round(crossprod(contrasts(data$A)), 2)
```

. . . . Go to 7
7 a. Determine whether the design is balanced

```
> replications(DV ~ A * b * C + .., data)
> library(biology)
> is.balanced(DV ~ A * b * C + .., data)
```

Design is balanced - sample sizes of all cells are equal (Type I SS) . . . . Go to 8
b. Design is NOT balanced - sample sizes of cells differ (Type III SS) . . . . Go to 9
8 a. Balanced Model I (Fixed factors) . . . . See Examples 12A,12B

```
> data.aov <- aov(DV ~ A * B, data)
```

To check residual plot . . . . Go to 21

- **With planned contrasts**

```
> library(biology)
> AnovaM(data.aov, split = list(A = list(Name1 = 1, Name2 = 2,
+     ...), B = list()))
> #OR
> summary(data.aov, split = list(A = list(Name1 = 1,
+     Name2 = 2, ...), B = list()))
```

where `DV` *is the response variable,* `A` *and* `B` *are the main fixed factors within the* `data` *dataset.*

- **Without planned contrasts**

```
> AnovaM(data.aov)
> #OR
> summary(data.aov)
> #OR
> anova(data.aov)
```

For post-hoc multiple comparisons . . . . Go to 20

If significant interation . . . . Go to 14
For summary plot . . . . Go to 18

b. Balanced Model II (random factors) or Model III (mixed factors) . . . . See Example 12C

```
> data.aov <- aov(DV ~ A * B, data)
```

To check residual plot . . . . Go to 21

- **With planned contrasts**

```
> AnovaM(data.aov, denoms = c("A:B", "Resid", "Resid"),
+     split = list(A = list(Name1 = 1, Name2 = 2, ...),
+         B = list()))
```

This example is a restricted model III where `DV` *is the response variable,* `A` *is a fixed factor and* `B` *is a random factor within the* `data` *dataset.* `denoms=c()` *is used to specify the denominators for each term in the model according to table 12.1*

- **Without planned contrasts**

```
> AnovaM(data.aov, denoms = c("A:B", "Resid", "Resid"))
```

For post-hoc multiple comparisons . . . . Go to 20

For variance components . . . . Go to 19
If significant interation . . . . Go to 14
For summary plot . . . . Go to 18

9 a. Unbalanced Model I (Fixed factors) . . . . See Example 12D

```
> data.aov <- aov(DV ~ A * B, data)
```

To check residual plot . . . . Go to 21

- **With planned contrasts**

```
> AnovaM(data.aov, type = "III", split = list(A = list
+     (Name1 = 1, Name2 = 2, ...), B = list()))
```

where `DV` *is the response variable,* `A` *and* `B` *are the main fixed factors within the* `data` *dataset.*

- **Without planned contrasts** - must define contrasts other than the default (treatment contratsts)

```
> contrasts(data$A) <- contr.helmert
> contrasts(data$B) <- contr.helmert
> data.aov <- aov(DV ~ A * B, data)
> AnovaM(data.aov, type = "III", data)
```

For post-hoc multiple comparisons . . . . Go to 20

If significant interation . . . . Go to 14

For summary plot . . . . Go to 18

b. Unbalanced Model II (random factors) or Model III (mixed factors)

```
> data.aov <- aov(DV ~ A * B, data)
```

To check residual plot . . . . Go to 21

- **With planned contrasts**

```
> AnovaM(data.aov, denoms = c("A:B", "Resid", "Resid"),
+      type = "III", split = list(A = list(Name1 = 1,
+      Name2 = 2, ...), B = list()))
```

example is a restricted model III where `DV` is the response variable, `A` is a fixed factor and `B` is a random factor within the `data` dataset. `denoms=c()` is used to specify the denominators for each term in the model according to table 12.1

- **Without planned contrasts**

```
> AnovaM(data.aov, denoms = c("A:B", "Resid", "Resid"),
+       type = "III")
```

For post-hoc multiple comparisons . . . . Go to 20

For variance components . . . . Go to 19
If significant interation . . . . Go to 14
For summary plot . . . . Go to 18

10 **Generate a new factorial variable to represent the combinations of factor levels and define sets of contrasts to represent each of the terms (main factors and interactions) in the design** . . . . See Examples 12E,13

```
> data$AB <- factor(paste(data$A, data$B, sep = "A:B"))
> contrasts(data$AB) <- cbind(c(contrasts), c(contrasts), ...)
```

. . . . Go to 12

11 **a. Determine whether the design is otherwise balanced (all present cells have equal sample sizes)**

```
> replications(DV ~ A * b * C + .., data)
> library(biology)
> is.balanced(DV ~ A * b * C + .., data)
```

Design is balanced - sample sizes of all cells are equal (Type I SS) . . . . Go to 12

b. Design is NOT balanced - sample sizes of cells differ (Type III SS) . . . . . . Go to 13

12 **a. Balanced missing cells Model I (Fixed factors)** . . . . See Example 12E

```
> data.aov <- aov(DV ~ AB, data)
```

To check residual plot . . . . Go to 21

```
> AnovaM(data.aov, split = list(AB = list('Factor A' = 1:2)))
```

where in this case, `DV` is the response variable and `AB` is the combined factors (`A` and `B`) within the `data` dataset. In this case, the ANOVA table will also include a line titled "Factor A" which represents the combination of the first two contrasts.

For post-hoc multiple comparisons . . . . Go to 20
If significant interation . . . . Go to 14
For summary plot . . . . Go to 18

b. Balanced missing cells Model II (random factors) or Model III (mixed factors)

```
> data.aov <- aov(DV ~ AB, data)
```

To check residual plot .... Go to 21

```
> AnovaM(data.aov, denoms = c(object), split = list(AB = list
+     ('Factor A' = 1:2)))
```

example is a restricted model III where `DV` *is the response variable, and* `AB` *is a random factor representing the combination of factors* `A` *and* `B` *within the* `data` *dataset.* `denoms=c(object)` *is used to specify the denominators for each term in the model according to table 12.1. The* `object` *can be either a list of labels that refer to terms in the current model, a single alternative* `aov` *model from which to extract the* `Residual` *term, or a list of alternative model terms. Note, interaction terms should be derived prior to main factors.*

For post-hoc multiple comparisons .... Go to 20
For variance components .... Go to 19
If significant interation .... Go to 14
For summary plot .... Go to 18

13 a. **Unbalanced missing cells Model I (Fixed factors)** .... See Example 12F

```
> data.aov <- aov(DV ~ AB, data)
```

To check residual plot .... Go to 21

```
> AnovaM(data.aov, type = "III", split = list(AB = list
+     ('Factor A' = 1:2)))
```

where `DV` *is the response variable,* `A` *and* `B` *are the main fixed factors within the* `data` *dataset.*

For post-hoc multiple comparisons .... Go to 20
If significant interation .... Go to 14
For summary plot .... Go to 18

b. **Unbalanced missing cells Model II (random factors) or Model III (mixed factors)**

```
> data.aov <- aov(DV ~ AB, data)
```

To check residual plot .... Go to 21

```
> AnovaM(data.aov, denoms = c(c(object)), type = "III",
+     split = list(AB = list('Factor A' = 1:2)))
```

example is a restricted model III where `DV` *is the response variable,* `A` *is a fixed factor and* `B` *is a random factor within the* `data` *dataset.* `denoms=c(object)` *is used to specify the denominators for each term in the model according to table 12.1. The* `object` *can be either a list of labels that refer to terms in the current model, a single alternative* `aov` *model from which to extract the* `Residual` *term, or a list of alternative model terms.*

For post-hoc multiple comparisons .... Go to 20
For variance components .... Go to 19
If significant interation .... Go to 14
For summary plot .... Go to 18

14 **Main effects tests** .... See Examples 12B,12D,12E,12F

- **Repeat analysis steps above with on a subset of the data (just one levels of one of the factors) and use the MS_{Resid} from the global model.**

```
> AnovaM(mainEffects(data.aov, at = B == "B1"), split = list
+     (A = list(Name1 = 1, Name2 = 2, ...)))
```

where in this case, `DV` *is the response variable and* `A` *is a fixed factor (*`A` *from a Model I factorial design within the* `data` *dataset.* `denoms=c()` *is used to specify the denominators for each term in the model according to table 12.1*

15 a. Underlying distribution of the response variable is normal for each level of the interaction, but the variances are unequal (Welch's test on combined factors)
Generate a new factorial variable to represent the combinations of factor levels and analyse as a single factor ANOVA using a Welch's test (see Key 10.6)

```
> data$AB <- factor(paste(data$A, data$B, sep = "A:B"))
> oneway.test(DV ~ AB, data, var.equal = F)
```

b. Underlying distributions not normally distributed . . . . Go to 16
or consider GLM . . . . GLM chapter 17

c. Underlying distributions not normally distributed . . . . Go to 16

16 a. Underlying distribution of the response variable and residuals is known . . . GLM chapter 17

b. Underlying distributions of the response variable and residuals is not known . . . . Go to 17

17 a. Variances not wildly unequal, outliers present, but data independent (Kruskal-Wallis non-parametric test on combined factors)

```
> data$AB <- factor(paste(data$A, data$B, sep = "A:B"))
> kruskal.test(DV ~ AB, data, var.equal = F)
```

b. Variances not wildly unequal, random sampling not possible - data might not be independent (Randomization test)
Follow the instructions in Key 10.8b to randomize the *F*-ratios or *MS* values from ANOVA tables produced using the parametric steps above. **Warning, randomization procedures are only useful when there are a large number of possible randomization combinations (rarely the case in factorial designs)**

18 a. Interaction plot to summarize an ordered trend (line graph) . . . . See Examples 12A,12B,12E

```
> library(gmodels)
> data.means <- with(data, tapply(DV, list(FACTA, FACTB), mean))
> data.se <- with(data, tapply(DV, list(FACTA, FACTB),
+     function(x) ci(x)[4]))
> with(data, interaction.plot(FACTA, FACTB, DV, las = 1,
+     lwd = 2, ylim = range(pretty(data$DV), na.rm = T),
+     axes = F, xlab = "", ylab = "", pch = c(16, 17),
+     type = "b", legend = F))
> arrows(1:3, data.means - data.se, 1:3, data.means + data.se,
+     code = 3, angle = 90, len = 0.05)
> axis(2, cex.axis = 0.8, las = 1, mgp = c(3, 0.5, 0),
+     tcl = -0.2)
> mtext(text = "Y-label", side = 2, line = 3, cex = 1)
> axis(1, cex.axis = 0.8, at = 1:3, lab = c("Lab1",
+     "Lab2", ...))
> mtext(text = "X-label", 1, line = 3, cex = 1)
> box(bty = "l")
```

```
> legend("topright", leg = c("Lab1", "Lab2", ...), lwd = 2,
+     lty = c(2, 1), bty = "n", pch = c(16, 17), cex = 1)
```

where `FACTA` *is the factor to placed on the x-axis.*

b. Interaction plot to summarize an unordered categories (bargraph) .... See Examples 12C,12D,12F

```
> library(gmodels)
> data.means <- t(tapply(data$DV, list(data$FACTA, data$FACTB),
+     mean, na.rm = T))
> data.se <- t(tapply(data$DV, list(data$FACTA, data$FACTB),
+     function(x) ci(x, na.rm = T)[4]))
> xs <- barplot(data.means, ylim = range(pretty(data$DV),
+     na.rm = T), beside = T, axes = F, xpd = F, axisnames = F,
+     axis.lty = 2, legend.text = F, col = c(0, 1))
> arrows(xs, data.means, xs, data.means + data.se, code = 2,
+     angle = 90, len = 0.05)
> axis(2, las = 1)
> axis(1, at = apply(xs, 2, median), lab = c("Lab1",
+     "Lab2", ...), padj = 1, mgp = c(0, 0, 0))
> mtext(2, text = "Y-label", line = 3, cex = 1)
> mtext(1, text = "X-label", line = 3, cex = 1)
> box(bty = "l")
> legend("topright", leg = c("Lab1", "Lab2", ...), fill = c(0,
+     1), col = c(0, 1), bty = "n", cex = 1)
```

where `FACTA` *is the factor to placed on the x-axis.*

19 Estimate variance components .... See Example 12C

```
> library(lme4)
> lmer(DV ~ 1 + (1 | A) + (1 | B) + (1 | A:B) + ..., data)
```

20 a. Perform Tukey's post-hoc multiple comparisons .... See Example 12D

```
> TukeyHSD(mod, which = "Factor")
> library(multcomp)
> summary(glht(mod, linfct = mcp(Factor = "Tukey")))
> confint(glht(mod, linfct = mcp(Factor = "Tukey")))
```

where mod is the name of an `aov` *model and* `,Factor,` *is the name of a factor.*

b. Perform other form of post-hoc multiple comparisons .... Go to Key 10.9

21 Examine a residual plot .... See Examples 12A-12D

```
> plot(data.aov, which = 1)
```

12.12 Worked examples of real biological data sets

Example 12A: Two factor fixed (Model I) ANOVA

Quinn (1988) manipulated the density of adults limpets within enclosures (8, 15, 30 and 45 individuals per enclosure) during two seasons (winter-spring and summer-autumn) so as to investigate the effects of adult density and season on egg mass production by intertidal limpets. Three replicate enclosures per density/season combination were used, and both density and season were considered fixed factors (from Box 9.4 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Quinn (1988) data set

```
> quinn <- read.table("quinn.csv", header = T, sep = ",")
```

Step 2 - The density vector (variable) contains numerical representations of the adult limpet densities, and R will consider this to be a *integer* vector rather than a categorical *factor*. In order to ensure that this variable is treated as a factor we need to redefine its class

```
> class(quinn$DENSITY)
[1] "integer"

> quinn$DENSITY <- factor(quinn$DENSITY)
> class(quinn$DENSITY)
[1] "factor"
```

Step 3 (Key 12.2) Quinn (1988) considered both factors to be fixed factors and thus the data represent a Model I design

Step 4 (Key 12.3) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 12.1).
According to Table 12.1, the MS_{Resid} (individual enclosures) should be used as the replicates for all hypothesis tests for Model I designs.

Factor A (Fixed)

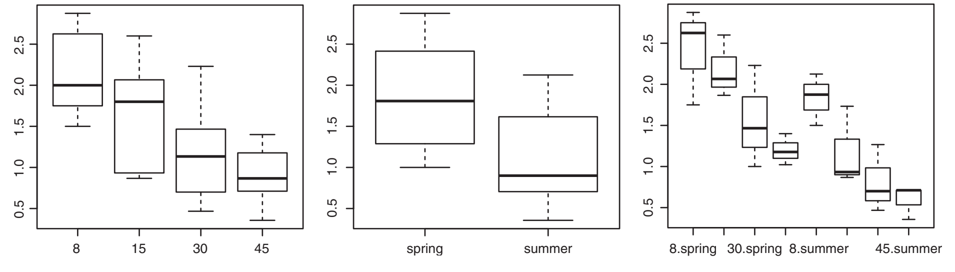
```
> boxplot(EGGS ~
+      DENSITY, quinn)
```

Factor B (Fixed)

```
> boxplot(EGGS ~
+      SEASON, quinn)
```

A:B interaction (Fixed)

```
> boxplot(EGGS ~
+      DENSITY * SEASON,
+      quinn)
```



Conclusions - No evidence of non-normality (boxplots not wildly asymmetrical) and no apparent relationship between mean and variance (heights of boxplots increase up the y-axis). No evidence that any of the hypothesis tests will be unreliable.

Step 5 (Key 12.5 & 12.7) - Determine whether or not the design is missing any factor combinations (cells) or is unbalanced (unequal sample sizes).

```
> replications(EGGS ~ DENSITY * SEASON, quinn)
       DENSITY          SEASON DENSITY:SEASON
             6              12              3
```

```
> library(biology)
> is.balanced(EGGS ~ DENSITY * SEASON, quinn)
[1] TRUE
```

Conclusions - The design is completely balanced. There are three replicate enclosures for each of the four densities and two seasons.

Step 6 - (Key 12.6) - Define polynomial contrasts (see sections 10.6 and 7.3.1 for more information on setting contrasts) to further investigate the nature of the effects of density on egg mass production.

```
> contrasts(quinn$DENSITY) <- contr.poly(4, scores = c(8, 15, 30,
+     45))
```

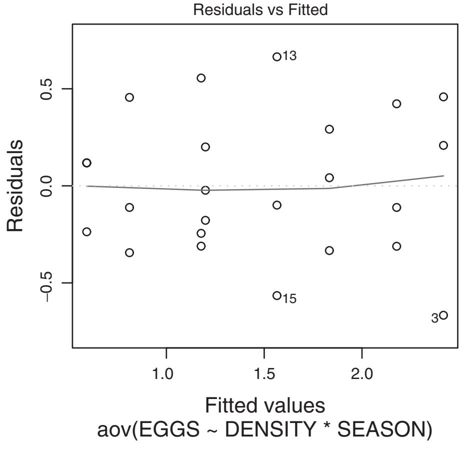
Note that there is no need to check the orthogonality of these contrasts, they will always be constructed to be orthogonal.

Step 7 (Key 12.8) - Fit the factorial linear model[i].

```
> quinn.aov <- aov(EGGS ~ DENSITY + SEASON + DENSITY:SEASON,
+     data = quinn)
> #OR
> quinn.aov <- aov(EGGS ~ DENSITY * SEASON, data = quinn)
```

Step 8 (Key 12.21) - Examine the fitted model diagnostics[j]. Note that this is evaluating the overall residuals and predicted values for the interaction effect.)

```
> plot(quinn.aov, which = 1)
```



Conclusions - As anticipated, there is no indication of a 'wedge' pattern in the residuals suggesting that the assumption of unequal variance is likely to be satisfied.

Step 9 (Key 12.8) - Examine the balanced model I ANOVA table, including the set of defined planned polynomial contrasts.

```
> summary(quinn.aov, split = list(DENSITY = list(Linear = 1,
+     Quadratic = 2)))
```

[i] Note that if we were also intending to investigate a set of planned comparisons/contrasts (see chapter 10.6), these should be defined prior to fitting the linear model.

[j] Recall that leverage, and thus Cook's D are not informative for categorical predictor variables.

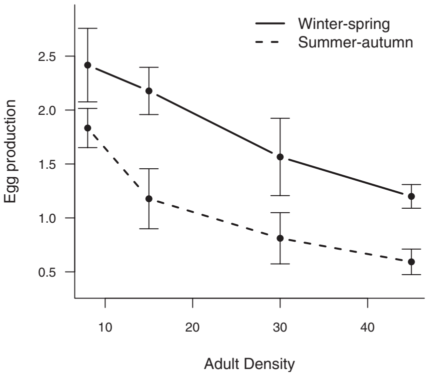
OR

```
> library(biology)
> AnovaM(quinn.aov, type = "I", split = list(DENSITY =
+     list(Linear = 1, Quadratic = 2)))
                            Df Sum Sq Mean Sq F value    Pr(>F)
DENSITY                      3 5.2841  1.7614  9.6691 0.0007041 ***
  DENSITY: Linear            1 5.0241  5.0241 27.5799 7.907e-05 ***
  DENSITY: Quadratic         1 0.2358  0.2358  1.2946 0.2719497
SEASON                       1 3.2502  3.2502 17.8419 0.0006453 ***
DENSITY:SEASON               3 0.1647  0.0549  0.3014 0.8239545
  DENSITY:SEASON: Linear     1 0.0118  0.0118  0.0649 0.8021605
  DENSITY:SEASON: Quadratic  1 0.0691  0.0691  0.3796 0.5464978
Residuals                   16 2.9146  0.1822
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There was no evidence of an interaction between density and season (suggesting that the effect of density was consistent across both seasons). Egg production was significantly greater in winter-spring than summer-autumn and declined linearly with increasing adult density.

Step 10 (Key 12.18) - Summarize the trends in a interaction plot.

```
> library(gmodels)
> quinn.means <- tapply(quinn$EGGS, list(quinn$DENSITY,
+     quinn$SEASON), mean)
> quinn.se <- tapply(quinn$EGGS, list(quinn$DENSITY, quinn$SEASON),
+     function(x) ci(x)[4])
> quinn$DENS <- as.numeric(as.character(quinn$DENSITY))
> plot(EGGS ~ DENS, quinn, type = "n", axes = F, xlab = "",
+     ylab = "")
> points(quinn.means[, 1] ~ unique(quinn$DENS), pch = 16,
+     type = "b", lwd = 2)
> arrows(unique(quinn$DENS), quinn.means[, 1] - quinn.se[, 1],
+     unique(quinn$DENS), quinn.means[, 1] + quinn.se[, 1],
+     code = 3, angle = 90, len = 0.1)
> points(quinn.means[, 2] ~ unique(quinn$DENS), pch = 16,
+     type = "b", lwd = 2, lty = 2)
> arrows(unique(quinn$DENS), quinn.means[, 2] - quinn.se[, 2],
+     unique(quinn$DENS), quinn.means[, 2] + quinn.se[, 2],
+     code = 3, angle = 90, len = 0.1)
> axis(1, cex.axis = 0.8)
> mtext(text = "Adult Density", 1, line = 3)
> axis(2, cex.axis = 0.8, las = 1)
> mtext(text = "Egg production", side = 2, line = 3)
> legend("topright", leg = c("Winter-spring", "Summer-autumn"),
+     lwd = 2, lty = c(1, 2), bty = "n")
> box(bty = "l")
```



Example 12B: Two factor fixed (Model I) ANOVA

In a similar experiment to that illustrated in Example 12A, Quinn (1988) also manipulated the density of larger adults limpets further down the shoreline within enclosures (6, 12 and 24 individuals per enclosure) during the two seasons (winter-spring and summer-autumn) so as to investigate their effects on egg mass production. Again, three replicate enclosures per density/season combination were used, and both density and season were considered fixed factors (from Box 9.4 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Quinn (1988) data set

```
> quinn1 <- read.table("quinn1.csv", header = T, sep = ",")
```

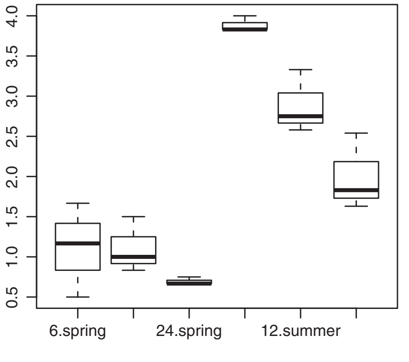
Step 2 - redefine the density vector as a factor

```
> quinn1$DENSITY <- factor(quinn1$DENSITY)
```

Step 3 (Key 12.2) Quinn (1988) considered both factors to be fixed factors and thus the data represent a Model I design

Step 4 (Key 12.3) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 12.1).
According to Table 12.1, the MS_{Resid} (individual enclosures) should be used as the replicates for all hypothesis tests for Model I designs.

```
> boxplot(EGGS ~ DENSITY * SEASON, quinn1)
```



Conclusions - No evidence of non-normality (boxplots not wildly asymmetrical) and no apparent relationship between mean and variance (heights of boxplots increase up the y-axis). No evidence that any of the hypothesis tests will be unreliable.

Step 5 (Key 12.5 & 12.7) - Determine whether or not the design is missing any factor combinations (cells) or is unbalanced (unequal sample sizes).

```
> replications(EGGS ~ DENSITY * SEASON, quinn1)
       DENSITY         SEASON DENSITY:SEASON
             6              9              3
> library(biology)
> is.balanced(EGGS ~ DENSITY * SEASON, quinn1)
[1] TRUE
```

Conclusions - The design is completely balanced. There are three replicate enclosures for each of the three densities and two seasons.

Step 6 - (Key 12.6) - Quinn and Keough (2002) illustrated treatment contrasts to compare the control adult density (6) to the increased densities (12 and 24) and whether this differed between the seasons[k]. To do this we define our own contrasts (see sections 10.6 and 7.3.1 for more information on setting contrasts).

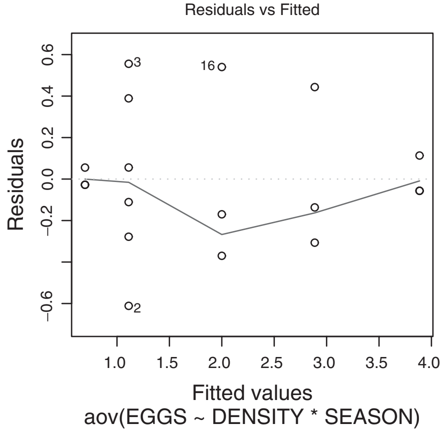
```
> contrasts(quinn1$DENSITY) <- cbind(c(1, -0.5, -0.5))
```

Step 7 (Key 12.8) - Fit the factorial linear model[l].

```
> quinn1.aov <- aov(EGGS ~ DENSITY * SEASON, data = quinn1)
```

Step 8 (Key 12.21) - Examine the fitted model diagnostics[m]. Note that is evaluating the overall residuals and predicted values for the interaction effect.)

```
> plot(quinn1.aov, which = 1)
```



Conclusions - As anticipated, there is no indication of a 'wedge' pattern in the residuals suggesting that the assumption of unequal variance is likely to be satisfied.

Step 9 (Key 12.8) - Examine the model I, balanced anova table, including the set of defined planned contrasts. Store the resulting ANOVA table with a name so that the data therein can later be accessed.

[k] Note that Quinn and Keough (2002) also defined a linear polynomial contrast. However, as this contrast is not orthogonal (independent) of the treatment contrast, it cannot be included in the one linear model.

[l] Note that if we were also intending to investigate a set of planned comparisons/contrasts (see chapter 10.6), these should be defined prior to fitting the linear model.

[m] Recall that leverage, and thus Cook's D are not informative for categorical predictor variables.

```
> library(biology)
> quinn1.anova<-AnovaM(quinn1.aov, type="I", split=list(DENSITY=
+ list('6 vs 12&24'=1)))
> quinn1.anova
                             Df  Sum Sq Mean Sq F value    Pr(>F)
DENSITY                       2  4.0019  2.0010  13.984 0.0007325 ***
  DENSITY: 6 vs 12&24         1  2.7286  2.7286  19.069 0.0009173 ***
SEASON                        1 17.1483 17.1483 119.845 1.336e-07 ***
DENSITY:SEASON                2  1.6907  0.8454   5.908 0.0163632 *
  DENSITY:SEASON: 6 vs 12&24  1  1.5248  1.5248  10.656 0.0067727 **
Residuals                    12  1.7170  0.1431
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There is strong evidence of a interaction between density and season. Whether or not there is a difference between the egg production of control vs high adult density depends on the season.

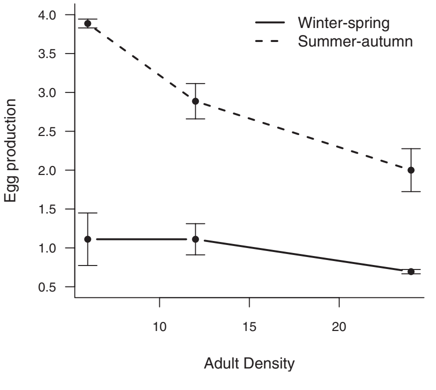
Step 10 (Key 12.14) - To further explore the interaction between density and season, Quinn and Keough (2002) investigated the effects of adult density separately for each season using two single factor ANOVA's. For each ANOVA, the MS_{Resid} from the global (overall) model was used as the denominator in *F*-ratio calculations.

```
> # effect of density in spring
> library(biology)
> AnovaM(mainEffects(quinn1.aov, at=SEASON=="spring"),
+ split=list(DENSITY=list('6 vs 12&24'=1)))

                      Df  Sum Sq Mean Sq F value    Pr(>F)
INT                    3 22.4940  7.4980 52.4017 3.616e-07 ***
DENSITY                2  0.3469  0.1735  1.2124    0.3315
  DENSITY: 6 vs 12&24  1  0.0869  0.0869  0.6076    0.4508
Residuals             12  1.7170  0.1431
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> # effect of density in summer
> AnovaM(mainEffects(quinn1.aov, at=SEASON=="summer"),
+ split=list(DENSITY=list('6 vs 12&24'=1)))

                      Df  Sum Sq Mean Sq F value    Pr(>F)
INT                    3 17.4953  5.8318  40.757 1.436e-06 ***
DENSITY                2  5.3457  2.6728  18.680 0.0002065 ***
  DENSITY: 6 vs 12&24  1  4.1664  4.1664  29.118 0.0001611 ***
Residuals             12  1.7170  0.1431
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Whilst egg production was found to be significantly lower in higher densities of adult limpets compared to natural densities during the summer-autumn season, such as trend was not observed during the spring-winter season.

Step 11 (Key 12.18) - Summarize the trends in a interaction plot.

```
> library(gmodels)
> quinn1.means <- tapply(quinn1$EGGS, list(quinn1$DENSITY,
+     quinn1$SEASON), mean)
> quinn1.se <- tapply(quinn1$EGGS, list(quinn1$DENSITY,
+     quinn1$SEASON), function(x) ci(x)[4])
> quinn1$DENS <- as.numeric(as.character(quinn1$DENSITY))
> plot(EGGS ~ DENS, quinn1, type = "n", axes = F, xlab = "",
+     ylab = "")
> points(quinn1.means[, 1] ~ unique(quinn1$DENS), pch = 16,
+     type = "b", lwd = 2)
> arrows(unique(quinn1$DENS), quinn1.means[, 1] - quinn1.se[, 1],
+     unique(quinn1$DENS), quinn1.means[, 1] + quinn1.se[, 1],
+     code = 3, angle = 90, len = 0.1)
> points(quinn1.means[, 2] ~ unique(quinn1$DENS), pch = 16,
+     type = "b", lwd = 2, lty = 2)
> arrows(unique(quinn1$DENS), quinn1.means[, 2] - quinn1.se[, 2],
+     unique(quinn1$DENS), quinn1.means[, 2] + quinn1.se[, 2],
+     code = 3, angle = 90, len = 0.1)
> axis(1, cex.axis = 0.8)
> mtext(text = "Adult Density", 1, line = 3)
> axis(2, cex.axis = 0.8, las = 1)
> mtext(text = "Egg production", side = 2, line = 3)
> legend("topright", leg = c("Winter-spring", "Summer-autumn"),
+     lwd = 2, lty = c(1, 2), bty = "n")
> box(bty = "l")
```



Example 12C: Two factor mixed (Model III) ANOVA

Minchinton and Ross (1999) investigated the distribution of oyster substrates for limpets in four zones alone the shore (the landward zone high on the shore, the mid zone with mangrove trees, the seaward zone with mangrove trees and the seaward zone without trees) by measuring the number of limpets per oyster shell (expressed as the number of limpets per 100 oysters) in five quadrats per zone. Data were collected from two sites (considered a random factor) so as to provide some estimates of the spatial generality of the observed trends (from Box 9.4 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Minchinton and Ross (1999) data set

```
> minch <- read.table("minch.csv", header = T, sep = ",")
```

Step 2 (Key 12.2) Minchinton and Ross (1999) considered the zone factor to be fixed and the site factor to be a random factor and thus the data represent a Model III design

Step 3 (Key 12.3) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate F-ratio denominators see Table 12.1).

According to Table 12.1, the effect of zone should be tested against the zone by site interaction whereas the effect of site and the interaction should be tested against the overall residual term (MS_{Resid}). As boxplots are

Factor A (Fixed)

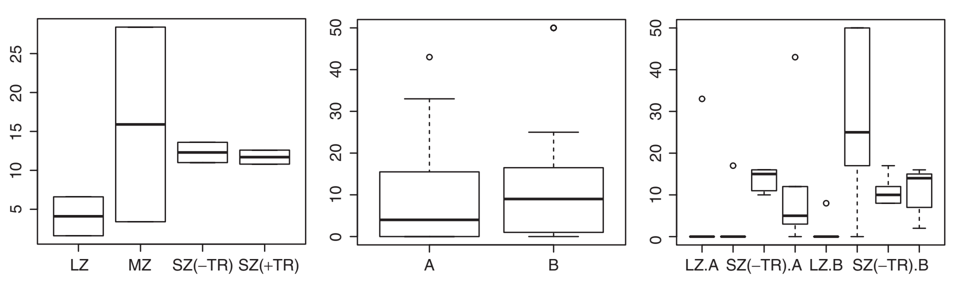
```
> library(nlme)
> minch.agg<-gsummary
+ (minch, groups=
+ minch$ZONE:minch$SITE)
> boxplot(LIMPT100~ZONE,
+ minch.agg)
```

Factor B (Random)

```
> boxplot(LIMPT100 ~
+      SITE, minch)
```

A:B interaction (Random)

```
> boxplot(LIMPT100 ~
+      ZONE * SITE,
+      minch)
```

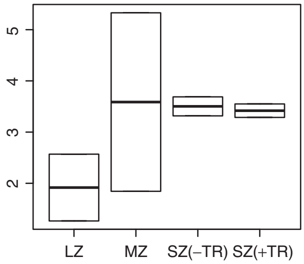


Conclusions - strong evidence to suggest both non-normality (boxplots asymmetrical where enough data) and the existence of a relationship between mean and variance (heights of boxplots increase up the y-axis). Hypothesis tests may well be unreliable.

Step 4 (Key 12.4) - Assess square-root transformed data (square root appropriate given the number of 0 counts).

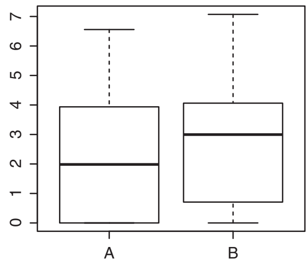
Factor A (Fixed)[n]

```
> boxplot(sqrt
+     (LIMPT100) ~
+     ZONE, minch.agg)
```



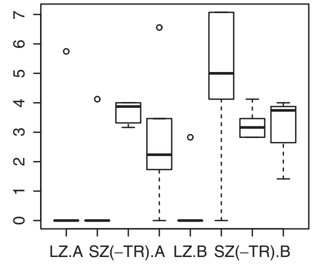
Factor B (Random)

```
> boxplot(sqrt
+     (LIMPT100) ~
+     SITE, minch)
```



A:B interaction (Random)

```
> boxplot(sqrt
+     (LIMPT100) ~
+     ZONE * SITE, minch)
```



Conclusions - although not ideal, the transformation is an improvment and thus hypothesis tests based on the square root transformed data are likely to be more reliable.

Step 5 (Key 12.5 & 12.7) - Determine whether or not the design is missing any factor combinations (cells) or is unbalanced (unequal sample sizes).

```
> replications(sqrt(LIMPT100) ~ ZONE * SITE, minch)
     ZONE      SITE ZONE:SITE
       10        20         5
> library(biology)
> is.balanced(sqrt(LIMPT100) ~ ZONE * SITE, minch)
[1] TRUE
```

Conclusions - The design is completely balanced. There are five replicate quadrats for each of the four zones and two sites.

Step 6 - (Key 12.6) - Quinn and Keough (2002) did not illustrate the use of planned contrasts in Box 9.5 (presumably due to the lack of any main effects). However, prior to analysing these data, a number of sensible planned contrasts are identifiable in the context of investigating the distribution of suitable limpet substrates. We will further propose contrasting the treed zones to the treeless seaward zone by defining our own contrasts (see sections 10.6 and 7.3.1 for more information on setting contrasts).

```
> contrasts(minch$ZONE) <- cbind(c(1/3, 1/3, -1, 1/3))
```

Step 7 (Key 12.8b) - Fit the factorial linear model [o].

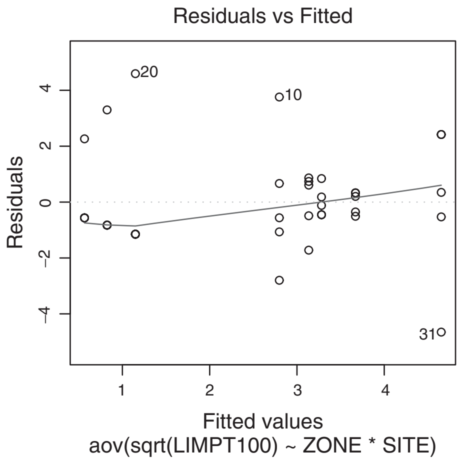
```
> minch.aov <- aov(sqrt(LIMPT100) ~ ZONE * SITE, data = minch)
```

[n] Note that the following procedure is mimicking a square root transformation. Ideally, these data should be transformed prior to aggregation rather than transforming the aggregated data (as demonstrated), but for the purpose of assumption checking it is acceptable.

[o] Note that if we were also intending to investigate a set of planned comparisons/contrasts (see chapter 10.6), these should be defined prior to fitting the linear model.

Step 8 (Key 12.21) - Examine the fitted model diagnostics[p]. Note that this is evaluating the overall residuals and predicted values for the interaction effect.

```
> plot(minch.aov, which = 1)
```



Conclusions - there is no indication of a 'wedge' pattern in the residuals suggesting that the assumption of unequal variance is likely to be satisfied.

Step 9 (Key 12.8b) - Examine the balanced model III ANOVA table, including the set of defined planned contrasts. Store the resulting ANOVA table with a name so that the data therein can later be accessed.

```
> library(biology)
> (minch.anova<-AnovaM(minch.aov, split = list(ZONE =
+ list('Treed vs No trees' = 1)), denoms = c("ZONE:SITE","Resid",
+ "Resid")))
Anova Table (Type III tests)

Response: sqrt(LIMPT100)
                            Df  Sum Sq Mean Sq F value  Pr(>F)
ZONE                         3  39.249  13.083  1.2349 0.43320
  ZONE: Treed vs No trees    1  12.448  12.448  1.1750 0.35772
SITE                         1   6.372   6.372  1.8425 0.18415
ZONE:SITE                    3  31.783  10.594  3.0632 0.04205 *
  ZONE:SITE: Treed vs No trees 1   4.700   4.700  1.3588 0.25236
Residuals                   32 110.673   3.459
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There is evidence of a interaction between zone and site suggesting that any patterns in limpet numbers between zones are not consistent across sites.

[p] Recall that leverage, and thus Cook's D are not informative for categorical predictor variables.

Step 10 (Key 12.19) - Estimate the variance components of the random (and fixed) terms[q] via the restricted maximum likelihood (REML) method.

```
> library(lme4)
> lmer(sqrt(LIMPT100) ~ 1 + (1 | ZONE) + (1 | SITE) +
+     (1 | ZONE:SITE), minch)
Linear mixed model fit by REML
Formula: sqrt(LIMPT100) ~ 1 + (1 | ZONE) + (1 | SITE) +
             (1 | ZONE:SITE)
   Data: minch
   AIC   BIC logLik deviance REMLdev
 180.8 189.3  -85.4    171.5   170.8
Random effects:
 Groups    Name        Variance   Std.Dev.
 ZONE:SITE (Intercept) 1.2160e+00 1.1027e+00
 ZONE      (Intercept) 3.5443e-01 5.9534e-01
 SITE      (Intercept) 5.0652e-16 2.2506e-08
 Residual              3.4585e+00 1.8597e+00
Number of obs: 40, groups: ZONE:SITE, 8; ZONE, 4; SITE, 2

Fixed effects:
            Estimate Std. Error t value
(Intercept)   2.5096     0.5719   4.388
```

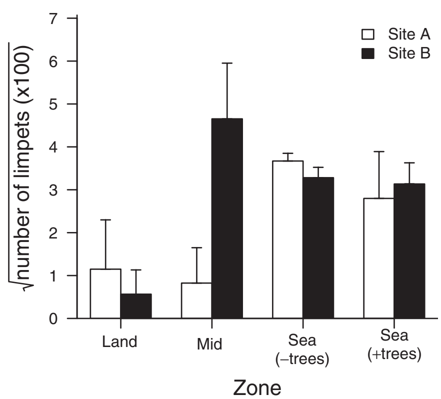
Conclusions - Although the interaction term explained approximately 26% ($1.216/(1.216 + 0 + 3.455)$), most of the variance was unexplained ($(3.455/(1.216 + 0 + 3.455) = 74\%$). Note that these values differ slightly from those presented by Quinn and Keough (2002) in Box 9.5, because they are estimated by the REML method rather than the ANOVA method which is restricted to balanced designs.

Step 11 (Key 12.18b) - Summarize the trends in a bargraph (from Quinn and Keough (2002)).

```
> library(gmodels)
> minch.means <- t(tapply(sqrt(minch$LIMPT100), list(minch$ZONE,
+     minch$SITE), mean))
> minch.se <- t(tapply(sqrt(minch$LIMPT100), list(minch$ZONE,
+     minch$SITE), function(x) ci(x)[4]))
> xs <- barplot(minch.means, ylim = range(sqrt(minch$LIMPT100)),
+     beside = T, axes = F, xpd = F, axisnames = F, axis.lty = 2,
+     legend.text = F, col = c(0, 1))
```

[q] Note that variance components for fixed terms are interpreted differnently to those of random terms. Whereas for random terms, variance components estimate the variance between all possible population means, for fixed factors they only estimate the variance between the specific populations used.

```
> arrows(xs, minch.means, xs, minch.means + minch.se, code = 3,
+     angle = 90, len = 0.05)
> axis(2, las = 1)
> axis(1, at = apply(xs, 2, median), lab = c("Land", "Mid",
+     "Sea\n(-trees)", "Sea\n(+trees)"), padj = 1,
+     mgp = c(0, 0, 0))
> mtext(2, text = expression(paste(sqrt("number of limpets
+     (x100)"))), line = 3, cex = 1)
> mtext(1, text = "Zone", line = 3, cex = 1)
> legend("topright", leg = c("Site A", "Site B"), fill = c(0, 1),
+     col = c(0, 1), bty = "n", cex = 1)
> box(bty = "l")
```



Example 12D: Two factor unbalanced fixed (Model I) ANOVA

Quinn and Keough (2002) present a two factor analysis of variance (Quinn and Keough, 2002; Table 9.15b) of a subset of a dataset by Reich et al. (1999) in which the specific leaf area of a number of plant species were compared from four different biomes (New Mexico woodlands, South Carolina temperate/sub-tropical forests, Venezuela tropical rain forests and Wisconsin temperate forests) and two different functional groups (shrubs and trees). Sample sizes varied for each combination of factors (cells).

Step 1 - Import (section 2.3) the modified Reich et al. (1999) data set

```
> reich <- read.table("reich.csv", header = T, sep = ",")
```

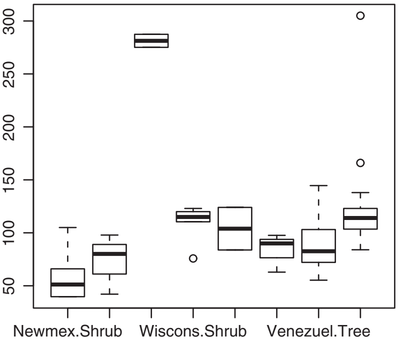
Step 2 (Key 12.2) Reich et al. (1999) considered both location and functional group to be fixed factors and thus the data represent a Model I design

Step 3 (Key 12.3) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate F-ratio denominators see Table 12.1).

According to Table 12.1, the effect of location, functional group as well as their interaction should all be tested against the overall residual term (MS_{Resid}).

A:B interaction (Fixed)[r]

```
> boxplot(LEAFAREA ~ LOCATION * FUNCTION, na.omit(reich))
```



Conclusions - no strong evidence to suggest either consistent non-normality or the of a relationship between mean and variance (heights of boxplots increase up the y-axis). Hypothesis tests likely to be reliable.

Step 4 (Key 12.5 & 12.7) - Determine whether or not the design is missing any factor combinations (cells) or is unbalanced (unequal sample sizes).

```
> replications(LEAFAREA ~ LOCATION * FUNCTION, reich)
$LOCATION
LOCATION
  Newmex Scarolin Venezuel  Wiscons
       7        6       23       21

$FUNCTION
FUNCTION
Shrub  Tree
   16    41

$'LOCATION:FUNCTION'
          FUNCTION
LOCATION   Shrub Tree
  Newmex       5    2
  Scarolin     3    3
  Venezuel     2   21
  Wiscons      6   15
> library(biology)
> is.balanced(LEAFAREA ~ LOCATION * FUNCTION, reich)
[1] FALSE
```

[r] Note that there is a missing case (denoted "NA" in the dataset). There are many functions that by default return an error when there are missing cases (so as to reduce the risks that potentially unrepresentative outcomes being blindly accepted by the user). Such functions need to be informed to ignore missing cases. This can be done either with the `na.rm=T` argument or by using the `na.omit()` *function* to create a temporary copy of the original dataset with the entire row of the missing case removed.

Conclusions - The design is unbalanced. The number of samples per location and function combination varies from 2 to 21. Therefore Type II or III sums of squares are appropriate. In this case, as we potentially wish to make conclusions about each of the main effects that are over and above the other main effects and their interaction, Type III sums of squares will be demonstrated.

Step 5 - (Key 12.6) - By default, all unordered factors are coded as treatment (compare to control) contrasts which are not appropriate for Type III sums of squares. Therefore, although we have no planned contrasts to perform in association with fitting the linear model, we do need to code the contrasts of the factors as helmert contrasts[s].

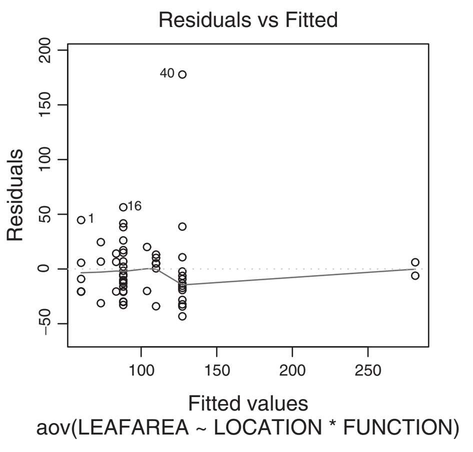
```
> contrasts(reich$LOCATION) <- contr.helmert
> contrasts(reich$FUNCTION) <- contr.helmert
```

Step 6 (Key 12.9) - Fit the factorial linear model.

```
> reich.aov <- aov(LEAFAREA ~ LOCATION * FUNCTION, data = reich)
```

Step 7 (Key 12.21) - Examine the fitted model diagnostics[t]. Note that is evaluating the overall residuals and predicted values for the interaction effect.)

```
> plot(reich.aov, which = 1)
```



Conclusions - Although there is no indication of a 'wedge' pattern in the residuals, observation 40 has a very large residual (considered an extreme outlier) and is potentially very influential. Caution should be excised for any hypothesis test close to the critical α value (0.05).

Step 8 (Key 12.9) - Examine the unbalanced model I ANOVA table. Store the resulting ANOVA table with a name so that the data therein can later be accessed.

```
> library(biology)
> (reich.anova <- AnovaM(reich.aov, type = "III"))
                  Df Sum Sq Mean Sq F value    Pr(>F)
LOCATION           3  49202   16401 13.6005 1.401e-06 ***
FUNCTION           1   6919    6919  5.7378   0.02047 *
```

[s] Other contrasts (such as polynomial or user defined orthogonal contrasts) would also be equally as valid - just not treatment contrasts.
[t] Recall that leverage, and thus Cook's D are not informative for categorical predictor variables.

```
LOCATION:FUNCTION  3  67783   22594 18.7367 3.120e-08 ***
Residuals         49  59088    1206
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
1 observation deleted due to missingness
```

Conclusions - There is strong evidence of a interaction between location and functional group suggesting that the patterns between different ecosystems differ according to the functional type of the plants and visa versa.

Step 9 (Key 12.14 & 12.20) - To better appreciate the patterns in specific leaf area between the different ecosystems, simple main effects tests can be performed to investigate the effects of location separately for each functional group. When so doing, recall that it is necessary to use the MS_{Resid} from the original (global) analysis of variance as the residual term. Tukey's post hoc honestly significant difference tests have also been included to investigate the pairwise differences between locations.

Effect of location for the shrub functional group

```
> AnovaM(reich.aov.shrub <- mainEffects(reich.aov, at =
+     FUNCTION == "Shrub"), type = "III")
            Df Sum Sq Mean Sq F value    Pr(>F)
INT          4  14994    3749  3.1086   0.02338 *
LOCATION     3  75012   25004 20.7351 8.199e-09 ***
Residuals   49  59088    1206
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
1 observation deleted due to missingness
> library(multcomp)
> summary(glht(reich.aov.shrub, linfct = mcp(LOCATION = "Tukey")))
         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = update(object, ~INT + .), data = dn)

Linear Hypotheses:
                         Estimate Std. Error t value Pr(>|t|)
Scarolin - Newmex == 0      13.07      25.36   0.515   0.9542
Venezuel - Newmex == 0     220.95      29.05   7.605   <0.001 ***
Wiscons - Newmex == 0       49.55      21.03   2.356   0.0973 .
Venezuel - Scarolin == 0   207.88      31.70   6.558   <0.001 ***
Wiscons - Scarolin == 0     36.48      24.55   1.486   0.4485
Wiscons - Venezuel == 0   -171.40      28.35  -6.045   <0.001 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

```
> confint(glht(reich.aov.shrub, linfct = mcp(LOCATION = "Tukey")))
         Simultaneous Confidence Intervals

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = update(object, ~INT + .), data = dn)

Estimated Quantile = 2.6496
95% family-wise confidence level


Linear Hypotheses:
                         Estimate  lwr       upr
Scarolin - Newmex == 0     13.0667  -54.1263   80.2596
Venezuel - Newmex == 0    220.9500  143.9708  297.9292
Wiscons - Newmex == 0      49.5500   -6.1634  105.2634
Venezuel - Scarolin == 0  207.8833  123.8922  291.8745
Wiscons - Scarolin == 0    36.4833  -28.5760  101.5426
Wiscons - Venezuel == 0  -171.4000 -246.5240  -96.2760
```

Effect of location for the tree functional group

```
> AnovaM(reich.aov.tree <- mainEffects(reich.aov, at = FUNCTION ==
+      "Tree"), type = "III")
            Df Sum Sq Mean Sq F value  Pr(>F)
INT          4  75431   18858 15.6382 2.6e-08 ***
LOCATION     3  14575    4858  4.0289 0.01222 *
Residuals   49  59088    1206
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
1 observation deleted due to missingness
> library(multcomp)
> summary(glht(reich.aov.tree, linfct = mcp(LOCATION = "Tukey")))
         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = update(object, ~INT + .), data = dn)

Linear Hypotheses:
                         Estimate Std. Error t value Pr(>|t|)
Scarolin - Newmex == 0     -20.40      31.70  -0.644   0.9108
Venezuel - Newmex == 0     -15.70      25.70  -0.611   0.9224
Wiscons - Newmex == 0       23.37      26.14   0.894   0.7950
Venezuel - Scarolin == 0     4.70      21.43   0.219   0.9959
```

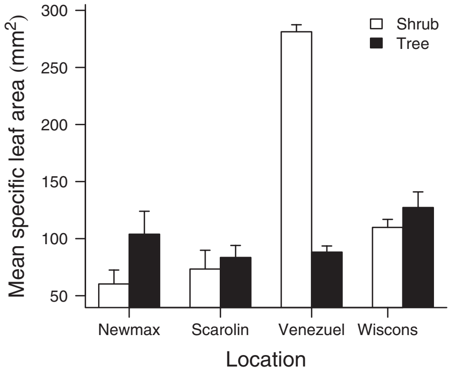
```
Wiscons - Scarolin == 0     43.77      21.96   1.993   0.1895
Wiscons - Venezuel == 0     39.07      11.74   3.328   0.0079 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
> confint(glht(reich.aov.tree, linfct = mcp(LOCATION = "Tukey")))
         Simultaneous Confidence Intervals

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = update(object, ~INT + .), data = dn)

Estimated Quantile = 2.6156
95% family-wise confidence level


Linear Hypotheses:
                         Estimate  lwr       upr
Scarolin - Newmex == 0    -20.4000 -103.3134   62.5134
Venezuel - Newmex == 0    -15.7000  -82.9132   51.5132
Wiscons - Newmex == 0      23.3667  -45.0055   91.7388
Venezuel - Scarolin == 0    4.7000  -51.3597   60.7597
Wiscons - Scarolin == 0    43.7667  -13.6774  101.2108
Wiscons - Venezuel == 0    39.0667    8.3615   69.7718
```

Conclusions - Specific leaf area differs significantly between locations for both shrub and tree functional groups. However, whilst specific leaf area of trees was only found to differ significantly between Wisconsin cold temperate forests and Venezuela topical forests (the former having greater area), for shrubs, the Venezuela topical forests were found to have significantly greater leaf areas than shrubs in the other ecosystems.

Step 10 (Key 12.18b) - Summarize the trends in a bargraph (from Quinn and Keough (2002)).

```
> library(gmodels)
> reich.means <- t(tapply(reich$LEAFAREA, list(reich$LOCATION,
+     reich$FUNCTION), mean, na.rm = T))
> reich.se <- t(tapply(reich$LEAFAREA, list(reich$LOCATION,
+     reich$FUNCTION), function(x) ci(x, na.rm = T)[4]))
> xs <- barplot(reich.means, ylim = range(reich$LEAFAREA,
+     na.rm = T), beside = T, axes = F, xpd = F, axisnames = F,
+     axis.lty = 2, legend.text = F, col = c(0, 1))
> arrows(xs, reich.means, xs, reich.means + reich.se, code = 2,
+     angle = 90, len = 0.05)
> axis(2, las = 1)
> axis(1, at = apply(xs, 2, median), lab = c("Newmax", "Scarolin",
+     "Venezuel", "Wiscons"), padj = 1, mgp = c(0, 0, 0))
```

```
> mtext(2, text = expression(paste("Mean specific leaf area ",
+     (mm^2))), line = 3, cex = 1)
> mtext(1, text = "Location", line = 3, cex = 1)
> box(bty = "l")
> legend("topright", leg = c("Shrub", "Tree"), fill = c(0, 1),
+     col = c(0, 1), bty = "n", cex = 1)
```



Example 12E: Two factor fixed (Model I) ANOVA with missing cells

Hall et al. (2000) measured the number of macroinvertebrate individuals colonizing small sheets of submerged cloth subjected to one of two treatments (nitrogen and phosphorus nutrients added or control) for either two, four or six months (time factor). Quinn and Keough (2002) present an analysis of a modification of these data in which the control treatments (no nutrients added) for the six month duration are all missing (from Table 9.16 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Hall et al. (2000) data set

```
> hall1 <- read.table("hall1.csv", header = T, sep = ",")
```

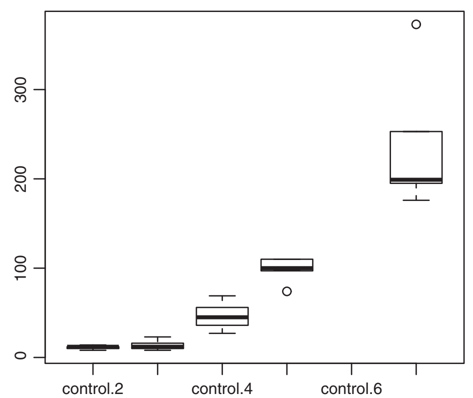
Step 2 (Key 12.2) Since the levels of the time factor are purely numbers, R considers this vector as a numeric variable rather than as a factorial variable. In order for the effect of time to be modeled appropriately, the time vector needs to be explicitly defined as a factor.

```
> hall1$TIME <- as.factor(hall1$TIME)
```

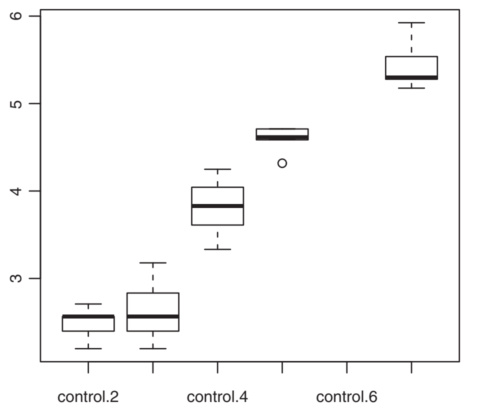
Step 3 (Key 12.2) Hall et al. (2000) considered both treatment and time to be fixed factors and thus the data represent a Model I design

Step 4 (Key 12.3) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate F-ratio denominators see Table 12.1).
According to Table 12.1, the effect of treatment and time as well as their interaction should all be tested against the overall residual term (MS_{Resid}).

```
> boxplot(IND ~ TREAT * TIME,
+     hall1)
```




```
> boxplot(log(IND + 1) ~
+     TREAT * TIME, hall1)
```



Conclusions - boxplots of the raw data (plot on left) show strong evidence of a relationship between mean and variance (height of boxplots related to their positions on the y-axis). The plot on the right illustrates boxplots of the data transformed to logs[u] and indicates that transforming the data to logs improves its suitability to parametric analysis.

Step 5 (Key 12.5b & 12.11) - Determine whether or not the design is missing any factor combinations (cells) or is unbalanced (unequal sample sizes).

```
> replications(log(IND + 1) ~ TREAT * TIME, hall1)
$TREAT
TREAT
 control nutrient
      10       15

$TIME
TIME
 2  4  6
10 10  5

$'TREAT:TIME'
          TIME
TREAT       2 4 6
  control   5 5 0
  nutrient 5 5 5
> library(biology)
> is.balanced(log(IND + 1) ~ TREAT * TIME, hall1)
[1] FALSE
```

[u] In order to accommodate zero values in the data, a small number (1) is added to each count prior to logarithmic transformation. This is referred to as a log plus one transformation.

Conclusions - The design has a missing cell - there are no replicates of the control treatment at 6 months. Quinn and Keough (2002) analysed this two factor ANOVA using a cell means model in which all replicated factor level combinations are treated as levels of a single factor in a single factor ANOVA. The main treatment effects are estimated by defining planned contrasts that are carefully selected to model the 'estimatable' comparisons.

Step 6 - (Key 12.10) - Convert the factor combinations into a single factor design.

```
> hall1$TREATTIME <- as.factor(paste(hall1$TREAT, hall1$TIME,
+     sep = ""))
```

Step 7 - (Key 12.10) - For each of the main terms in the original multifactor model (the main effects and interactions), define appropriate contrasts to estimate the effects of each term (see Tables 12.3 & 12.4), fit the cell means linear model and partition the sums of squares accordingly. Note that as missing cells are an extreme form of unbalance, they too can result in non-orthogonality of contrasts and therefore each of the main effects should be estimated separately.

Effect of nutrient treatment

```
> contrasts(hall1$TREATTIME) <- cbind(c(1, 1, -1,
+ -1, 0))
> AnovaM(aov(log(IND + 1) ~ TREATTIME, hall1),
+ split = list(TREATTIME = list("treatment" = 1)))
                       Df Sum Sq Mean Sq F value    Pr(>F)
TREATTIME               4 32.013   8.003  93.169 1.232e-12 ***
  TREATTIME: treatment  1  1.063   1.063  12.379  0.002161 **
Residuals              20  1.718   0.086
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Effect of time

```
> contrasts(hall1$TREATTIME) <- cbind(c(1, -1, 1,
+ -1, 0), c(0, 0, 1, 0, -1))
> AnovaM(aov(log(IND + 1) ~ TREATTIME, hall1),
+ split = list(TREATTIME = list("time" = 1:2,
+ " time 2 vs 4" = 1, " time 2 vs 6" = 2)))
                          Df Sum Sq Mean Sq F value    Pr(>F)
TREATTIME                  4 32.013   8.003  93.169 1.232e-12 ***
  TREATTIME: time          2 24.742  12.371 144.013 1.332e-12 ***
  TREATTIME:  time 2 vs 4  1 13.441  13.441 156.468 6.505e-11 ***
  TREATTIME:  time 2 vs 6  1 11.301  11.301 131.557 3.008e-10 ***
Residuals                 20  1.718   0.086
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Nutrient treatment by time interaction

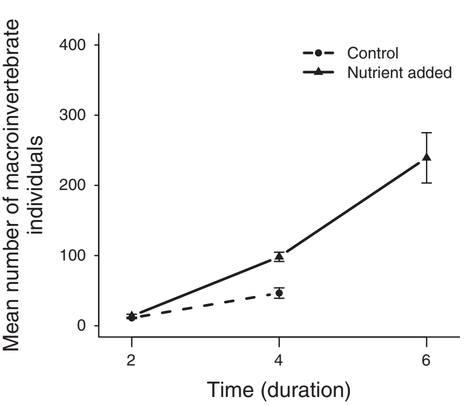
```
> contrasts(hall1$TREATTIME) <- cbind(c(1, -1, -1,
+ 1, 0))
```

```
> AnovaM(aov(log(IND + 1) ~ TREATTIME, hall1),
+ split = list(TREATTIME = list("treatment:time" = 1)))
                             Df Sum Sq Mean Sq F value    Pr(>F)
TREATTIME                     4 32.013   8.003 93.1689 1.232e-12 ***
  TREATTIME: treatment:time  1  0.491   0.491  5.7209   0.02670 *
Residuals                    20  1.718   0.086
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There is strong evidence of a significant interaction between the nutrient treatment and time. The effect of the nutrient treatment on the number of macroinvertebrate individuals colonizing the artificial substrates differs according to the duration for which the substrates have been available. The nature of the interaction could be explored by splitting the data up and analysing the effects of the nutrient treatment separately for each time. Additionally, given the sequential nature of time, polynomial trends could be explored for the nutrient added treatments.

Step 8 (Key 12.18a) - Summarize the trends with an interaction plot.

```
> library(gmodels)
> hall1.means <- with(hall1, tapply(IND, list(TIME, TREAT), mean))
> hall1.se <- with(hall1, tapply(IND, list(TIME, TREAT),
+     function(x) ci(x)[4]))
> with(hall1, interaction.plot(TIME, TREAT, IND, las = 1, lwd = 2,
+     ylim = range(pretty(hall1$IND)), axes = F, xlab = "",
+     ylab = "", pch = c(16, 17), type = "b", legend = F))
> arrows(1:3, hall1.means - hall1.se, 1:3, hall1.means + hall1.se,
+     code = 3, angle = 90, len = 0.05)
> axis(2, cex.axis = 0.8, las = 1, mgp = c(3, 0.5, 0), tcl = -0.2)
> mtext(text = expression(paste("Mean number of macroinvertebrate")),
+     side = 2, line = 3, cex = 1)
> mtext(text = expression(paste("individuals")), side = 2, line = 2,
+     cex = 1)
> axis(1, cex.axis = 0.8, at = 1:3, lab = c("2", "4", "6"))
> mtext(text = "Time (duration)", 1, line = 3, cex = 1)
> box(bty = "l")
> legend("topright", leg = c("Control", "Nutrient added"), lwd = 2,
+     lty = c(2, 1), bty = "n", pch = c(16, 17), cex = 1)
```



Example 12F: Two factor fixed (Model I) ANOVA with missing cells and unbalanced replication

Milliken and Johnson (1984) present a data set from a fictitious investigation into the effects of different fats and surfactants on the specific volume of baked bread. The 3x3 design was to include four replicates of each of the three fat types and three surfactant types (nine combinations). Unfortunately, many of the replicates were lost due to a defective batch of yeast. The structure of the data a represented below.

	Surf.1	Surf.2	Surf.3
Fat 1	XXX	XXX	
Fat 2	XXX		XXXX
Fat 3	XX	XXXX	XX

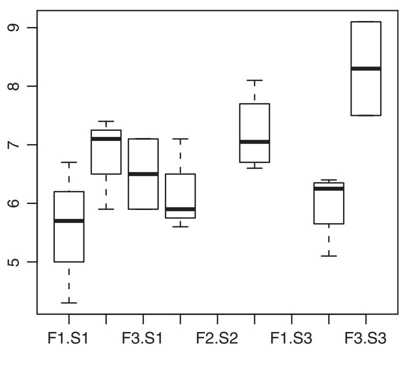
Step 1 - Import (section 2.3) the Milliken and Johnson (1984) data set

```
> milliken <- read.table("milliken.csv", header = T, sep = ",")
```

Step 2 (Key 12.2) Milliken and Johnson (1984) considered both treatment and time to be fixed factors and thus the data represent a Model I design

Step 3 (Key 12.3) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 12.1). According to Table 12.1, the effect of fat and surfactant type as well as their interaction should all be tested against the overall residual term (MS_{Resid}).

```
> boxplot(VOL ~ FAT * SURF, milliken)
```



Conclusions - no evidence of either non-normality (boxplots not consistently asymmetrical) or a relationship between mean and variance (height of boxplots related to their positions on the y-axis).

Step 4 (Key 12.5 & 12.11) - Determine whether or not the design is missing any factor combinations (cells) or is unbalanced (unequal sample sizes).

```
> replications(VOL ~ FAT * SURF, milliken)
$FAT
FAT
F1 F2 F3
 6  7  8
```

```
$SURF
SURF
S1 S2 S3
 8  7  6

$'FAT:SURF'
    SURF
FAT  S1 S2 S3
  F1  3  3  0
  F2  3  0  4
  F3  2  4  2
> library(biology)
> is.balanced(VOL ~ FAT * SURF, milliken)
[1] FALSE
```

Conclusions - The design is not balanced - the number of replicates in each fat/surfactant combination differs. Furthermore, there are two missing cells. As with example 12E, this can be analysed with a cell means model in which all replicated factor level combinations are treated as levels of a single factor in a single factor ANOVA. The main treatment effects are estimated by defining planned contrasts that are carefully selected to model the 'estimatable' comparisons.

Step 5 - (Key 12.10) - Convert the factor combinations into a single factor design.

```
> milliken$FS <- as.factor(paste(milliken$FAT, milliken$SURF,
+     sep = ""))
```

Step 6 - (Key 12.12F) - For each of the main terms in the original multifactor model (the main effects and interactions), define appropriate contrasts to estimate the effects of each term (see Tables 12.3 & 12.4), fit the cell means linear model and partition the sums of squares accordingly. Note that Type III sums of squares are used due to unbalanced data. Note also, that additional planned contrasts will also be included to potentially explore any main effects further.

Effect of the fat type

```
> contrasts(milliken$FS) <- cbind(c(1, 1, 0, 0, -1, -1, 0), c(0,
+     0, 1, 1, -1, 0, -1))
> AnovaM(aov(VOL ~ FS, milliken), split = list(FS = list
+     (fat = 1:2, ' fat: 1 vs 3' = 1, ' fat 2 vs 3' = 2)),
+     type = "III")
                    Df  Sum Sq Mean Sq F value  Pr(>F)
FS                   6 12.4714  2.0786  2.9493 0.04473 *
  FS: fat            2  3.8725  1.9363  2.7474 0.09851 .
  FS:  fat: 1 vs 3   1  1.6233  1.6233  2.3033 0.15135
  FS:  fat 2 vs 3    1  1.6178  1.6178  2.2955 0.15200
Residuals           14  9.8667  0.7048
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Effect of surfactant type

```
> contrasts(milliken$FS) <- cbind(c(0, 0, 0, 0, 0, 1, -1), c(0,
+     0, 1, -1, 1, 0, -1))
> AnovaM(aov(VOL ~ FS, milliken), split = list(FS = list
+     (surf = 1:2, ' surf: 2 vs 3' = 1, ' surf: 1 vs 3' = 2)),
+     type = "III")
                    Df  Sum Sq Mean Sq F value  Pr(>F)
FS                   6 12.4714  2.0786  2.9493 0.04473 *
  FS: surf           2  1.6702  0.8351  1.1850 0.33464
  FS:  surf: 2 vs 3  1  1.2063  1.2063  1.7116 0.21185
  FS:  surf: 1 vs 3  1  0.1593  0.1593  0.2261 0.64177
Residuals           14  9.8667  0.7048
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Fat type by surfactant type interaction

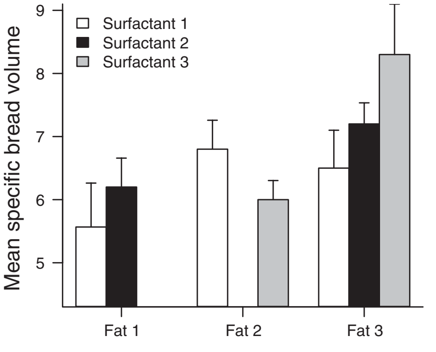
```
> contrasts(milliken$FS) <- cbind(c(1, -1, 0, 0, -1, 1, 0), c(0,
+     0, 1, -1, -1, 0, 1))
> AnovaM(aov(VOL ~ FS, milliken), split = list(FS = list
+     ('fat:surf' = 1:2, ' fat:surf1' = 1, ' fat:surf2' = 2)),
+     type = "III")
                 Df  Sum Sq Mean Sq F value  Pr(>F)
FS                6 12.4714  2.0786  2.9493 0.04473 *
  FS: fat:surf    2  4.7216  2.3608  3.3498 0.06474 .
  FS:  fat:surf1  1  0.2689  0.2689  0.3815 0.54672
  FS:  fat:surf2  1  4.6935  4.6935  6.6597 0.02178 *
Residuals        14  9.8667  0.7048
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Neither fat type nor surfactant type were found to significantly effect the specific volume of baked bread and nor was the impact of either found to be dependent on the other.

Step 7 (Key 12.18b) - Summarize the trends with an interaction plot.

```
> library(gmodels)
> milliken.means <- with(milliken, tapply(VOL, list(SURF, FAT),
+     mean, na.rm = T))
> milliken.se <- with(milliken, tapply(VOL, list(SURF, FAT),
+     function(x) ci(x, na.rm = T)[4]))
> xs <- barplot(milliken.means, ylim = range(milliken$VOL,
+     na.rm = T), beside = T, axes = F, xpd = F, axisnames = F,
+     axis.lty = 2, legend.text = F, col = c(0, 1, "gray"))
> axis(2, las = 1)
> axis(1, at = apply(xs, 2, median), lab = c("Fat 1", "Fat 2",
+     "Fat 3"), padj = 1, mgp = c(0, 0, 0))
```

```
> mtext(2, text = expression(paste("Mean specific bread volume ")),
+     line = 3, cex = 1)
> box(bty = "l")
> arrows(xs, milliken.means, xs, milliken.means + milliken.se,
+     code = 2, angle = 90, len = 0.05)
> legend("topleft", leg = c("Surfactant 1", "Surfactant 2",
+     "Surfactant 3"), fill = c(0, 1, "gray"), col = c(0, 1,
+     "gray"), bty = "n", cex = 1)
```



13

Unreplicated factorial designs – randomized block and simple repeated measures

Chapter 11 introduced the concept of employing sub-replicates that are nested within the main treatment levels as a means of absorbing some of the unexplained variability that would otherwise arise from designs in which sampling units are selected from amongst highly heterogeneous conditions. Such (nested) designs are useful in circumstances where the levels of the main treatment (such as burnt and un-burnt sites) occur at a much larger temporal or spatial scale than the experimental/sampling units (e.g. vegetation monitoring quadrats). For circumstances in which the main treatments can be applied (or naturally occur) at the same scale as the sampling units (such as whether a stream rock is enclosed by a fish proof fence or not), an alternative design is available. In this design (*randomized complete block design*), each of the levels of the main treatment factor are grouped (blocked) together (in space and/or time) and therefore, whilst the conditions between the groups (referred to as 'blocks') might vary substantially, the conditions under which each of the levels of the treatment are tested within any given block are far more homogeneous (see Figure 13.1b). If any differences between blocks (due to the heterogeneity) can account for some of the total variability between the sampling units (thereby reducing the amount of variability that the main treatment(s) failed to explain), then the main test of treatment effects will be more powerful/sensitive.

As an simple example of a randomized block, consider an investigation into the roles of different organism scales (microbial, macro invertebrate and vertebrate) on the breakdown of leaf debris packs within streams. An experiment could consist of four treatment levels - leaf packs protected by fish-proof mesh, leaf packs protected by fine macro invertebrate exclusion mesh, leaf packs protected by dissolving antibacterial tablets, and leaf packs relatively unprotected as controls. As an acknowledgement that there are many other unmeasured factors that could influence leaf pack breakdown (such as flow velocity, light levels, etc) and that these are likely to vary substantially throughout a stream, the treatments are to be arranged into groups or 'blocks' (each containing a single control, microbial, macro invertebrate and fish protected leaf pack).

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

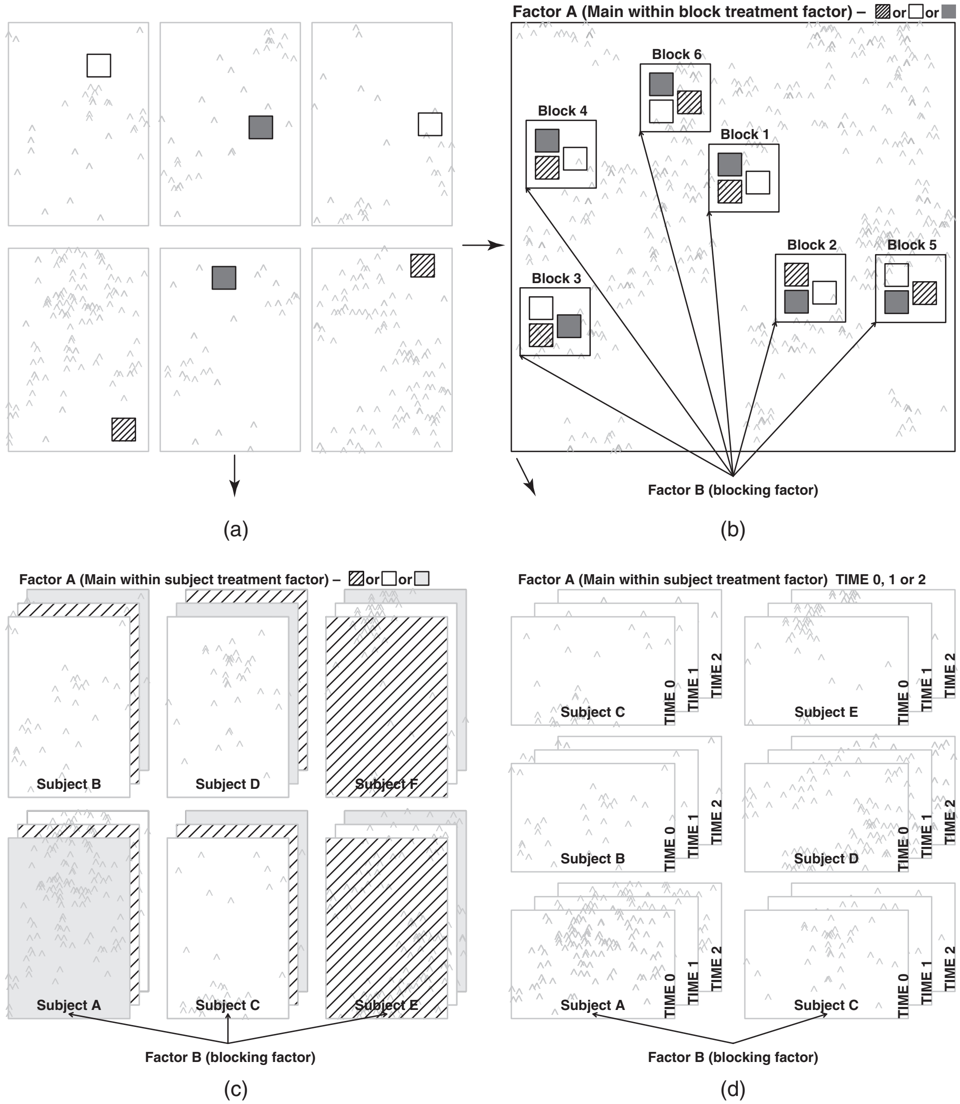


Fig 13.1 Fictitious spatial depictions contrasting (a) single factor (n = 2), (b) randomized complete block (n = 6) and (c-d) repeated measures (n = 6) ANOVA designs each with three treatment levels. When single sampling units are selected amongst highly heterogeneous conditions (as represented in (a)), it is unlikely that these single units will adequately represent the populations and repeated sampling is likely to yield very different outcomes. In such cases, this heterogeneity increases the unexplained variation thereby potentially masking any detectable effects due to the measured treatments. If however, it is possible to group each of the main treatment levels together within a small spatial or temporal scale (in which the conditions are likely to be more homogeneous), the groups (or 'blocks') should account for some of the unexplained variability between replicates thereby reducing the unexplained variability (and thus increasing the power of the main test of treatments).

Blocks of treatment sets are then secured in locations haphazardly selected throughout a particular reach of stream.

Blocking does however come at a cost. The blocks absorb both unexplained variability as well as degrees of freedom from the residuals. Consequently, if the amount of the total unexplained variation that is absorbed by the blocks is not sufficiently large enough to offset the reduction in degrees of freedom (which may result from either less than expected heterogeneity, or due to the scale at which the blocks are established being inappropriate to explain much of the variation), for a given number of sampling units (leaf packs), the tests of main treatment effects will suffer power reductions.

Treatments can also be applied sequentially or repeatedly at the scale of the entire block, such that at any single time, only a single treatment level is being applied (see Figure 13.1c-d). Such designs are called *repeated measures*. A repeated measures ANOVA is to an single factor ANOVA as a paired t-test is to a independent samples t-test. One example of a repeated measures analysis might be an investigation into the effects of five different diet drugs (four doses and a placebo) on the food intake of lab rats. Each of the rats ('subjects') is subject to each of the four drugs (within subject effects) which are administered in a random order. In another example, temporal recovery responses of sharks to bi-catch entanglement stresses might be simulated by analysing blood samples collected from captive sharks (subjects) every half hour for three hours following a stress inducing restraint.

This repeated measures design allows the anticipated variability in stress tolerances between individual sharks to be accounted for in the analysis (so as to permit more powerful test of the main treatments). Furthermore, by performing repeated measures on the same subjects, repeated measures designs reduce the number of subjects required for the investigation. Essentially, this is a randomized complete block design except that the within subject (block) effect (e.g. time since stress exposure) cannot be randomized (the consequences of which are discussed in section 13.4.1).

To suppress contamination effects resulting from the proximity of treatment sampling units within a block, units should be adequately spaced in time and space. For example, the leaf packs should not be so close to one another that the control packs are effected by the antibacterial tablets and there should be sufficient recovery time between subsequent drug administrations. In addition, the order or arrangement of treatments within the blocks must be randomized so as to prevent both confounding as well as computational complications (see section 13.4.1). Whilst this is relatively straight forward for the classic randomized complete block design (such as the leaf packs in streams), it is logically not possible for repeated measures designs.

Blocking factors are typically random factors (see section 10.0.1) that represent all the possible blocks that could be selected. As such, no individual block can truly be replicated. Randomized complete block and repeated measures designs can therefore also be thought of as un-replicated factorial designs in which there are two or more factors but that the interactions between the blocks and all the within block factors are not replicated.

13.1 Linear models

The linear models[a] for two and three factor un-replicated factorial design are:

$$y_{ij} = \mu + \beta_i + \alpha_j + \varepsilon_{ij} \qquad \text{(Model 1 or 2)}$$

$$y_{ijk} = \mu + \beta_i + \alpha_j + \gamma_k + \beta\alpha_{ij} + \beta\gamma_{ik} + \alpha\gamma_{jk} + \gamma\alpha\beta_{ijk} + \varepsilon_{ijk} \qquad \text{(Model 1)}$$

$$y_{ijk} = \mu + \beta_i + \alpha_j + \gamma_k + \alpha\gamma_{jk} + \varepsilon_{ijk} \qquad \text{(Model 2)}$$

where μ is the overall mean, β is the effect of the Blocking Factor B, α and γ are the effects of withing block Factor A and Factor C respectively and ε is the random unexplained or residual component.

Tests for the effects of blocks as well as effects within blocks assume that there are no interactions between blocks and the within block effects. That is, it is assumed that any effects are of similar nature within each of the blocks. Whilst this assumption may well hold for experiments that are able to consciously set the scale over which the blocking units are arranged, when designs utilize arbitrary or naturally occurring blocking units, the magnitude and even polarity of the main effects are likely to vary substantially between the blocks. The preferred (non-additive or 'Model 1') approach to un-replicated factorial analysis of some bio-statisticians is to include the block by within subject effect interactions (e.g. $\beta\alpha$). Whilst these interaction effects cannot be formally tested, they can be used as the denominators in F-ratio calculations of their respective main effects tests (see Tables 13.1 & 13.2). Proponents argue that since these blocking interactions cannot be formally tested, there is no sound inferential basis for using these error terms separately. Alternatively, models can be fitted additively ('Model 2') whereby all the block by within subject effect interactions are pooled into a single residual term (ε). Although the latter approach is simpler, each of the within subject effects tests do assume that there are no interactions involving the blocks[b] and that perhaps even more restrictively, that sphericity (see section 13.4.1) holds across the entire design.

13.2 Null hypotheses

Separate null hypotheses are associated with each of the factors, however, blocking factors are typically only added to absorb some of the unexplained variability and therefore specific hypothesis tests associated with blocking factors are of lesser biological importance.

---

[a] Note that whilst the order of the linear model terms is not important as far as software is concerned, the order presented above reflects (most closely) the hierarchy of the design structure. That is, the main factor effect (α) occurs within the blocking factor effect (β) and is thus placed after the blocking effect in the linear model. I say most closely since some of the terms are at the same hierarchical level (e.g. α and γ) and thus their orders are interchangeable.

[b] The presence of such interactions increase the residual variability and thus reduce the power of tests.

13.2.1 *Factor A* - the main within block treatment effect

Fixed (typical case)

$$H_0(A): \mu_1 = \mu_2 = ... = \mu_i = \mu \qquad \text{(the population group means of A (pooling B) are all equal)}$$

The mean of population 1 (pooling blocks) is equal to that of population 2 and so on, and thus all population means are equal to an overall mean. No effect of A within each block (Model 2) or over and above the effect of blocks. If the effect of the i^{th} group is the difference between the i^{th} group mean and the overall mean ($\alpha_i = \mu_i - \mu$) then the H_0 can alternatively be written as:

$$H_0(A): \alpha_1 = \alpha_2 = ... = \alpha_i = 0 \qquad \text{(the effect of each group equals zero)}$$

If one or more of the α_i are different from zero (the response mean for this treatment differs from the overall response mean), the null hypothesis is not true indicating that the treatment does affect the response variable.

Random

$$H_0(A): \sigma_\alpha^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of A (pooling B).

13.2.2 *Factor B* - the blocking factor

Random (typical case)

$$H_0(B): \sigma_\beta^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of B.

Fixed

$$H_0(B): \mu_1 = \mu_2 = \ldots = \mu_i = \mu \quad \text{(the population group means of B are all equal)}$$
$$H_0(B): \beta_1 = \beta_2 = \ldots = \beta_i = 0 \quad \text{(the effect of each chosen B group equals zero)}$$

The null hypotheses associated with additional within block factors, are treated similarly to Factor A above.

13.3 Analysis of variance

Partitioning of the total variance sequentially into explained and unexplained components and *F*-ratio calculations predominantly follows the rules established in

Table 13.1 *F*-ratios and corresponding R syntax for a range of two un-replicated factorial (randomized complete block and repeated measures) designs.

			***F*-ratio**	
Factor	**d.f.**	**MS**	**Model 1 (non-additive)**	**Model 2 (additive)**
B′ (block)	$b-1$	$MS_{B'}$	No test[a]	$\frac{MS_{B'}}{MS_{Resid}}$
A	$a-1$	MS_A	$\frac{MS_A}{MS_{Resid}}$	$\frac{MS_A}{MS_{Resid}}$
Residual (=B′A)	$(b-1)(a-1)$	MS_{Resid}		
		`> summary(aov(DV~Error(B)+A))`		
	Unbalanced	`> anova(lme(DV~A, random=~1|B))`		

[a]If A is random (or an unrestricted model), then *F*-ratio is $MS_{B'}/MS_{Resid}$.

chapters 11 and 12. Randomized block and repeated measures designs can essentially be analysed as Model III ANOVAs. The appropriate unexplained residuals and therefore the appropriate *F*-ratios for each factor differ according to the different null hypotheses associated with different combinations of fixed and random factors and what analysis approach (Model 1 or 2) is adopted for the randomized block linear model (see Tables 13.1 & 13.2).

In additively (Model 2) fitted models (in which block interactions are assumed not to exist and are thus pooled into a single residual term), hypothesis tests of the effect of B (blocking factor) are possible. However, since blocking designs are usually employed out of expectation for substantial variability between blocks, such tests are rarely of much biological interest.

13.4 Assumptions

As with other ANOVA designs, the reliability of hypothesis tests is dependent on the residuals being:

(i) normally distributed. Boxplots using the appropriate scale of replication (reflecting the appropriate residuals/*F*-ratio denominator (see Tables 13.1 & 13.2) should be used to explore normality. Scale transformations are often useful.

(ii) equally varied. Boxplots and plots of means against variance (using the appropriate scale of replication) should be used to explore the spread of values. Residual plots should reveal no patterns. Scale transformations are often useful.

(iii) independent of one another. Although the observations within a block may not strictly be independent, provided the treatments are applied or ordered randomly within each block or subject, within block proximity effects on the residuals should be random across all blocks and thus the residuals should still be independent of one another. Nevertheless, it is important that experimental units within blocks are adequately spaced in space and time so as to suppress contamination or carryover effects.

Table 13.2 *F*-ratios and corresponding R syntax for a range of un-replicated three-factor (randomized complete block and repeated measures) designs. *F*-ratio numerators and demoninators are represented by numbers that correspond to the rows from which the appropriate mean square values would be associated.

			***F*-ratio**					
			A&C, B random		**A fixed,B&C random**		**A,B&C random**	
	Factor	**d.f.**	**Model 1**	**Model 2**	**Model 1**	**Model 2**	**Model 1**	**Model 2**
1	B′	$b-1$	No test[a]	1/7	$1/6^b$	1/7	$1/(5+6-7)^{b\ c}$	1/7
2	A	$a-1$	2/5	2/7	$2/(4+5-7)^{b\ d}$	2/4	$2/(4+5-7)^{b\ c}$	2/4
3	C	$c-1$	3/6	3/7	$3/6^b$	$3/7^e$	$3/(4+6-7)^{b\ c}$	3/4
4	A×C	$(a-1)(c-1)$	4/7	4/7	4/7	4/7	4/7	4/7
5	B′×A	$(b-1)(a-1)$	No test		No test		5/7	
6	B′×C	$(b-1)(a-1)$	No test		No test		6/7	
7	Residuals (=B′×A×C)	$(b-1)(a-1)(c-1)$						

B random, A&C fixed

Model 1
```
> summary(aov(DV~+Error(B/(A*C)+A*C)))
```
Model 2
```
> summary(aov(DV~Error(B)+A*C))
```
Unbalanced
```
#sphericity met
> anova(lme(DV~A*C, random=~1|B), type='marginal')
#sphericity not met
> anova(lme(DV~A*C,random=~1|B,corr=corAR1(form=~1|B),
    type='marginal')
```

Other models
```
#F-ratios and P-values must be calculated individually
> AnovaM(aov(DV~B*A*C))
```

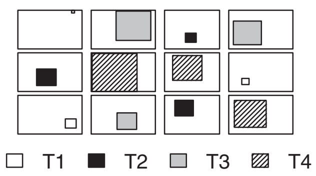
[a]If A is random (or an unrestricted model), then *F*-ratio is 1/7 ($MS_{B'}/MS_{Resid}$).
[b]Inexact *F*-ratio for restricted model.
[c]Pooling: higher order interactions with P> 0.25 can be removed to produce more exact denominators.
[d]Pooling: If P>0.25 for AC′ and P<0.25 for B′A, *F*-ratio denominator is $MS_{B'A}$. If P>0.25 for B′A and P<0.25 for AC′, *F*-ratio denominator is $MS_{AC'}$. If P>0.25 for both B′A and AC′, *F*-ratio denominator is $(SS_{AC'}+SS_{B'A}+SS_{B'AC'})/((a-1)(c-1)+(a-1)(c-1)+(a-1)(b-1)(c-1))$.
[e]For unrestricted model *F*-ratio denominator is $MS_{AC'}$.

13.4.1 Sphericity

Un-replicated factorial designs extend the usual equal variance (no relationship between mean and variance) assumption to further assume that the differences between each pair of within block treatments are equally varied across the blocks (see Figure 13.2). To meet this assumption, a matrix of variances (between pairs of observations within treatments) and covariances (between treatment pairs within each block) must display a pattern known as sphericity[c]

Typically, un-replicated factorial designs in which the treatment levels have been randomly arranged (temporally and spatially) within each block (randomized complete

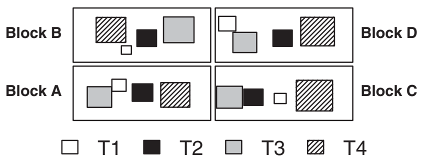
[c] Strickly, the variance-covariance matrix must display a very specific pattern of sphericity in which both variances and covariances are equal (compound symmetry), however an *F*-ratio will still reliably follow an *F* distribution provided basic sphericity holds.



Variance-covariance structure

	T1	T2	T3	T4
T1	0.15	0.00	0.00	0.00
T2	0.00	0.15	0.00	0.00
T3	0.00	0.00	0.15	0.00
T4	0.00	0.00	0.00	0.15

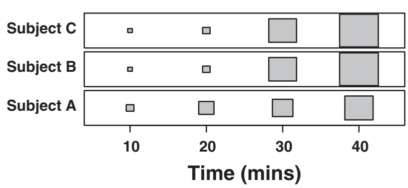
(a) *Single factor ANOVA*



Variance-covariance structure

	T1	T2	T3	T4
T1	0.15	0.05	0.05	0.05
T2	0.05	0.15	0.05	0.05
T3	0.05	0.05	0.15	0.05
T4	0.05	0.05	0.05	0.15

(b) *Randomized complete block design*



Variance-covariance structure

	Time1	Time2	Time3	Time4
Time1	0.50	0.60	0.30	0.10
Time2	0.60	0.50	0.60	0.30
Time3	0.30	0.60	0.50	0.60
Time4	0.10	0.30	0.60	0.50

(c) *Repeated measures design*

Fig 13.2 Fictitious representations of variance-covariance structures associated with examples of (a) Single factor ANOVA, (b) Randomized complete block and (c) Repeated measures designs. The matrix diagonals represent within group variances and the off-diagonals represent the covariances between each group pair. In each of the example designs, homogeneity of variance (between treatment groups) is met. The variance-covariance structure associated with single factor ANOVA designs typically have either zero covariance or at least no pattern in the covariances. Randomized complete block designs (in which the treatment levels are arranged randomly within each block) usually display compound symmetry (equal covariances). By contrast, repeated measures designs often violate this assumption (sphericity) and display a covariance structure that reflects a particular pattern in which progressively closer (temporally or spatially) observations collected from the same sampling units are progressively more similar (autocorrelated).

block) should meet this sphericity assumption. Conversely, repeated measures designs that incorporate factors whose levels cannot be randomized within each block (such as distances from a source or time), are likely to violate this assumption. In such designs, the differences between treatments that are arranged closer together (in either space or time) are likely to be less variable (greater paired covariances) than the differences between treatments that are further apart.

Hypothesis tests are not very robust to substantial deviations from sphericity and consequently would tend to have inflated type I errors. There are three broad techniques for compensating or tackling the issues of sphericity:

(i) reducing the degrees of freedom for *F*-tests according to the degree of departure from sphericity (measured by epsilon (ε)). The two main estimates of epsilon are

Greenhouse-Geisser and Huynh-Feldt, the former of which is preferred (as it provides more liberal protection) unless its value is less than 0.5.

(ii) perform a multivariate ANOVA (MANOVA). Although the sphericity assumption does not apply to such procedures, MANOVA's essentially test null hypotheses about the differences between multiple treatment pairs (and thus test whether an array of population means equals zero), and therefore assume multivariate normality - a difficult assumption to explore.

(iii) fit a linear mixed effects (lme) model (see section 11.8). The approximate form of the correlation structure can be specified up-front when fitting linear mixed effects models and thus correlated data are more appropriately handled. A selection of variance-covariance structures appropriate for biological data are listed in Table 13.3. It is generally recommended that linear mixed effects models be fitted with a range of covariance structures. The "best" covariance structure is that the results in a better fit (as measured by either AIC, BIC or ANOVA) than a model fitted with a compound symmetry structure.

13.4.2 Block by treatment interactions

The presence of block by treatment interactions have important implications for models that incorporate a single within block factor as well as additive models involving two or more within block factors. In both cases, the blocking interactions and overall random errors are pooled into a residual term that is used as the denominator in F-ratio calculations (see Table 13.1). Consequently, block by treatment interactions increase the denominator (MS_{Resid}) resulting in lower F-ratios (lower power). Moreover, the presence of strong blocking interactions would imply that any effects of the main factor are not consistent. Drawing conclusions from such an analysis (particularly in light of non-significant main effects) is difficult. Unless we assume that there are no block by within block interactions, non-significant within block effects could be due to either an absence of a treatment effect, or as a result of opposing effects within different blocks. As these block by within block interactions are unreplicated, they can neither be formally tested nor is it possible to perform main effects tests to diagnose non-significant within block effects.

Block by treatment interactions can be diagnosed by examining;

(i) interaction (cell means) plot. The mean ($n = 1$) response for each level of the main factor is plotted against the block number. Parallel lines infer no block by treatment interaction.

(ii) residual plot. A curvilinear pattern in which the residual values switch from positive to negative and back again (or visa versa) over the range of predicted values implies that the scale (magnitude but not polarity) of the main treatment effects differs substantially across the range of blocks. Scale transformations can be useful in removing such interactions.

(iii) Tukey's test for non-additivity evaluated at $\alpha = 0.10$ or even $\alpha = 0.25$. This (curvilinear test) formally tests for the presence of a quadratic trend in the relationship between residuals and predicted values. As such, it too is only appropriate for simple interactions of scale.

Table 13.3 Standard variance-covariance structures used in `lme`. It is generally recommended that the appropriateness of the various covariance structures be assessed in the order presented (top ones first). The moving average (φ) and autoregressive parameters (ρ) can range from -1 to $+1$ (0 to $+1$ for Continuous time). Note that by default, these parameter estimates vary during model optimization. Alternatively, parameters can be fixed, by providing values for the parameters with the `values=` argument and specifying `fixed=TRUE`.

Description	Structure	R syntax
General (unstructured) structure • Most complex (and least precise) • Separate variance and covariance estimates for all combinations	$\begin{matrix} \sigma_1^2 & \sigma_{21} & \sigma_{31} & \cdots \\ \sigma_{12} & \sigma_2^2 & \sigma_{32} & \cdots \\ \sigma_{13} & \sigma_{23} & \sigma_3^2 & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{matrix}$	`corSymm(form=~ 1|Block)`
Compound symmetry structure • Diagonals equal variance • Off-diagonals equal covariance • Simplest structure	$\begin{matrix} \sigma^2+\sigma_1^2 & \sigma_1^2 & \sigma_1^2 & \cdots \\ \sigma_1^2 & \sigma^2+\sigma_1^2 & \sigma_1^2 & \cdots \\ \sigma_1^2 & \sigma_1^2 & \sigma^2+\sigma_1^2 & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{matrix}$	`corCompSymm(form=~ 1|Block)`
First order autoregressive structure • Diagonals equal variance • Off-diagonal variance multiplied by the autoregressive coefficient (ρ) raised to increasing power • Covariance decreases with increased separation • Assumes equal separation spacing	$\begin{matrix} \sigma^2 & \sigma^2\rho & \sigma_1^2\rho & \cdots \\ \sigma_1^2\rho & \sigma^2 & \sigma_1^2\rho & \cdots \\ \sigma_1^2\rho & \sigma_1^2\rho & \sigma^2 & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{matrix}$	`corAR1(form=~ 1|Block)`
Moving average autoregressive structure • A more general autoregressive structure • Off-diagonal variance multiplied by the moving average (φ) parameter as well as the autoregressive coefficient (ρ) raised to increasing power	$\begin{matrix} \sigma^2 & \sigma^2\varphi\rho & \sigma_1^2\varphi\rho & \cdots \\ \sigma_1^2\varphi\rho & \sigma^2 & \sigma_1^2\varphi\rho & \cdots \\ \sigma_1^2\varphi\rho & \sigma_1^2\varphi\rho & \sigma^2 & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{matrix}$	`corARMA(form=~ 1|Block)`
Continuous time autoregressive structure • Accommodates unequal separation spacing		`corCAR1(form=~ 1|Block)`
Heterogenous variances structures • Each of the above can also be modified to accommodate unequal variances		`weights=varIdent(form=~ 1|Block)`

There are no corrections for other more severe interactions (such as cross-over) - effected conclusions must therefore be made cautiously.

13.5 Specific comparisons

For randomized complete block designs in which the levels of within block factors can be randomly arranged, both planned and unplanned multiple comparisons tests can be performed as per single factor or fully factorial linear models (see chapters 10&12). However, when the assumption of sphericity is likely to be violated (as is typically the case for repeated measures designs), the appropriate compensatory adjustments for each specific comparison are not clearly defined. Therefore, each specific planned comparison should be performed using separately generated denominators (error terms). Unplanned multiple comparisons should be performed as a series of paired *t* tests, subsequently corrected for inflated type I error rates (e.g. Bonferroni corrections) if necessary (see section 10.6).

13.6 Unbalanced un-replicated factorial designs

Since these designs are un-replicated, any missing observation equates to an entire missing combination (cell) and thus an unbalanced design. Unbalanced designs (to reiterate) are less robust to deviations from the assumptions (particularly sphericity) and therefore require special attention. There are a number of approaches for dealing with unbalanced un-replicated designs, the pros and cons of which are described below:

(i) Omit the entire block/subject from which the observation is missing. Clearly, such an approach is only acceptable for designs that have a large number of blocks in the first place as it involves disregarding otherwise good data.

(ii) Fit a cell means model with appropriate contrasts (see section 12.6.2). Defining the appropriate contrasts can be a very difficult process.

(iii) If block interactions are assumed not to exist (additivity)

  (a) perform regular analysis with missing values have been replaced by values predicted by solving equations such as (predicted value = treatment mean + block mean - overall mean) and subtract one degree of freedom for each substituted value.

  (b) compare the fit (residual sums of squares) of appropriate full and reduced models (e.g. full: $y_{ij} = \mu + \beta_i + \alpha_j + \varepsilon_{ij}$ versus reduced: $y_{ij} = \mu + \beta_i + \varepsilon_{ij}$) using ANOVA. Importantly, sphericity corrections should also be incorporated into this approach - a task that is difficult to achieve.

(iv) Fit a linear mixed effects (lme) model (see section 11.8). In contrast to ANOVA, which only produces optimal estimators (estimators that minimize variance) for balanced designs, maximum likelihood (ML and REML) and thus linear mixed effects estimators yield estimators that are 'asymptotically efficient' for both balanced and unbalanced designs. The ability of linear mixed effects models to accommodate balanced and unbalanced, correlated and hierarchical (nested) data makes them the preferred approach to analyzing unbalanced un-replicated factorial designs.

13.7 Robust alternatives

When the data are non-normal (or infected with outliers), rank-based analysis can be useful. Of particular note is the **Friedman test** which generates a test statistic after ranking the observations within each block and compares this statistic to a chi-square distribution. As is the case for other rank based alternatives, this approach is less powerful than the parametric equivalents and is less capable of handling blocking interactions. Moreover, rank based tests do not directly address the issues of sphericity and are therefore inappporpriate for repeated measures designs.

Randomization tests, in which observations are repeatedly shuffled amongst the treatments within each block, are useful (particularly when observational independence is violated).

13.8 Power and blocking efficiency

Power analyses follow single factor and fully factorial power analyses, except that with respect to sample sizes, the blocks become the replicates. The decision of whether or not to block is often a comprimise between reducing unexplained variation and retaining maximum degrees of freedom. For the benifit of future investigations on similar systems, it is often desirable to determine what benifit incorporating a blocking factor offered over a regular completely randomized design. An estimate of the relative effeciency of the blocking can be obtained from:

$$\text{Estimated blocking efficiency} = \frac{(q-1)MS_{Block} + q(p-1)MS_{Resid}}{(pq-1)MS_{Resid}}$$

13.9 Unreplicated factorial ANOVA in R

Randomized complete block and repeated measures designs can be analysed using the **`aov()`** function with blocking factors defined with the **`Error=`** argument. Anova tables for balanced designs that meet the assumption of sphericity can be viewed using the **`summary()`** function which can also accommodate planned contrasts with the **`split= argument`**. Alternatively, **`lme`** (**`nlme`**) and the more recent **`lmer`** (**`lme4`**) functions facilitate the argubly more appropriate linear mixed effects modelling approach to analysing unreplicated factorial designs. Associated planned comparisons are performed as **`estimable()`** functions.

13.10 Further reading

- Theory

Doncaster, C. P., and A. J. H. Davey. (2007). *Analysis of Variance and Covariance. How to Choose and Construct Models for the Life Sciences*. Cambridge University Press, Cambridge.

Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. 2 edition. John Wiley & Sons, New York.

Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

Pinheiro, J. C., and D. M. Bates. (2000). *Mixed effects models in S and S-PLUS*. Springer-Verlag, New York.

Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS, 4th edn*. Springer-Verlag, New York.

Zuur, A. F., E. N. Ieno, N. J. Walker, A. A. Saveliev, and G. M. Smith. (2009). *Mixed Effects Models and Extensions in Ecology with R*. Springer.

13.11 Key for randomized block and simple repeated measures ANOVA

1 a. Determine the appropriate model design and hierarchy

- **Conceptualise the design into a hierarchy (ladder) of factors**
  - Blocking factor (factor to which all levels (complete sets) of other factors are applied) at the top
  - Each of the main treatment factors (that are applied within each block) are considered lower in the hierarchy
  - The Block by treatment interactions (which are unreplicated) are next on the heirarchy
  - If there are two or more fixed within block treatment factors, then there are also interactions between these factors to consider
- Label random blocking factor levels (blocks or subjects) with a unique name

Block	Fact A	DV
B1	A1	.
B1	A2	.
B1	A3	.
B2	A1	.

- Identify the correct error (residual) term and thus F-ratio denominator for each factor (see Tables 13.1 & 13.2)

. . . . Go to 2

2 a. **Check assumptions for unreplicated factorial ANOVA**
As the assumptions of any given hypothesis test relate to residuals, all diagnostics should reflect the appropriate error (residual) terms for the hypothesis. This is particularly important for Model 1 (non-additive) models where interaction terms are used as the appropriate denominators (residuals).

- **No block by within block treatment interactions**

```
> with(data, interaction.plot(B, A, DV))
```

Residual curvature plot and Tukey's test for nonadditivity

```
> library(alr3)
> residual.plots(lm(DV ~ BLOCK + A, data))
> tukey.nonadd.test(lm(DV ~ BLOCK + A, data))
```

- **Normality (symmetry) of the response variable (residuals) at each level of each factor or combination of factors - boxplots of mean values**
Single within block factor or additive model (no interactions - Model 2) using MS_{Resid} as denominator in each case

```
> boxplot(DV ~ A, data)
> boxplot(DV ~ C, data)
> boxplot(DV ~ A * C, data)
```

Two or more within block factor non-additive (Model 1) model using interactions (such as MS_{BA}) as denominator as example

```
> library(lme4)
> data.BA.agg <- gsummary(data, groups = data$B:data$A)
> boxplot(DV ~ A, data.BA.agg)
```

where `DV` *is the response variable*, `A` *is a main fixed or random factor within the* `data` *dataset.*

- **Homogeneity (equality) of variance of the response variable (residuals) at each level of each factor or combination of factors - boxplots of mean values**
As for Normality.
Parametric assumptions (Normality/Homogeneity of variance) met . . Go to 4

b. **Parametric assumptions not met** . . . . Go to 3

3 a. **Attempt a scale transformation (see Table 3.2 for transformation options)** Go to 2
b. **Transformations unsuccessful or inappropriate** . . . . Go to 9

4 a. **If incorporating planned contrasts (comparisons)** . . . . . . See Examples 13A&13B

```
> contrasts(data$A) <- cbind(c(contrasts), ...)
> round(crossprod(contrasts(data$A)), 2)
```

. . . . Go to 5

5 a. **Determine whether the design is balanced**

```
> replications(DV ~ Error(Block) + A * C.., data)
> is.balanced(DV ~ Error(Block) + A * C.., data)
```

Design is balanced - sample sizes of all cells are equal . . . . Go to 6
b. **Design is NOT balanced - one or more cells (combinations) missing (0 replicates)** . . . . Go to 7

c. Design is NOT balanced - sample sizes of cells differ, but all combinations have at least one replicate . . . . Go to 8

6 a. Balanced single within block factor or additive (no interactions - Model 2) . . . . See Examples 13A,13B

```
> data.aov <- aov(DV ~ A + Error(Block), data)
> data.aov <- aov(DV ~ A * C + Error(Block), data)
```

Alternatively, consider linear mixed effects (lme) model . . . . Go to 13
Check for sphericity . . . . Go to 12

- **Sphericity met**

```
> summary(data.aov)
```

OR

```
> library(biology)
> AnovaM(data.aov)
```

- **Sphericity NOT met**

```
> library(biology)
> AnovaM(data.aov, RM = T)
```

To incorporate planned comparisons, utilize the `split=` argument, see Key 12.6
For post-hoc multiple comparisons . . . . Go to 12.20a

b. Balanced two or more within block factor non-additive (Model 1) . . . . See Examples 13A,13B&13D)

```
> data.aov <- aov(DV ~ Error(Block/A + Block/C) + A * C, data)
```

Alternatively, consider linear mixed effects (lme) model . . . . Go to 13
Check for sphericity . . . . Go to 12

- **Sphericity met**

```
> summary(data.aov)
```

OR

```
> library(biology)
> AnovaM(data.aov)
```

- **Sphericity NOT met**

```
> library(biology)
> AnovaM(data.aov, RM = T)
```

To incorporate planned comparisons, utilize the `split=` argument, see Key 12.6
For post-hoc multiple comparisons . . . . Go to Key 12.20a

7 a. Unbalanced (missing cells) single within block or additive (Model 2)

```
> data.lme <- lme(DV ~ A, random = ~1 | Block, data)
> data.lme <- lme(Y ~ A * C, random = ~1 | Block, data)
> anova(data.lme)
```

b. Unbalanced (missing cells) two or more within block factor non-additive (Model 1)

```
> data.lme <- lme(Y ~ A * C, random = ~1 | Block/A + 1 |
+     Block/C, data)
> anova(data.lme)
```

8 a. Unbalanced (unequal sample sizes n > 0) additive (Model 2)

```
> contrasts(data$A) <- contr.helmert
> contrasts(data$C) <- contr.helmert
> data.aov <- aov(DV ~ Error(Block) + A * C, data)
> AnovaM(data.aov, type = "III")
```

OR

```
> data.lme <- lme(DV ~ A * C, random = ~1 | Block, data)
```

b. Unbalanced (unequal sample sizes n > 0) non-additive (Model 1)

```
> data.aov <- aov(DV ~ Error(Block/A + Block/C) + A * C, data)
```

OR

```
> data.lme <- lme(DV ~ A, random = ~1 | Block, data)
> data.lme <- lme(Y ~ A * C, random = ~1 | Block, data)
> anova(data.lme)
```

9 a. Underlying distributions not normally distributed . . . . Go to 10
or consider GLM . . . . GLM chapter 17
b. Underlying distributions not normally distributed . . . . Go to 10
10 a. Underlying distribution of the response variable and residuals is known . . . . GLM chapter 17
b. Underlying distributions of the response variable and residuals is not known . . . . Go to 11
11 a. Variances not wildly unequal, outliers present, but data independent (Friedman non-parametric test) . . . . See Examples 13E

```
> friedman.test(DV ~ A | Block, data)
```

b. Variances not wildly unequal, random sampling not possible - data might not be independent (Randomization test
Follow the instructions in Key 10.8b to randomize the *F*-ratios or *MS* values from ANOVA tables produced using the parametric steps above. **Warning, randomization procedures are only useful when there are a large number of possible randomization combinations (rarely the case in blocking designs)**

12 a. Checking sphericity

```
> library(biology)
> epsi.GG.HF(data.aov)
```

13 a. Fitting linear mixed effects models

- Fit a range of models with alternative covariance structures

```
> library(nlme)
> #General (unstructured)
> data.lme <- lme(DV ~ A, random = ~1 | Block, data, corr =
+ corSymm(form = ~1 | Block))
> #Compound symmetry
> data.lme1 <- lme(DV ~ A, random = ~1 | Block, data, corr =
```

```
+ corrCompSymm(form = ~1 | Block))
> #Compound symmetry with heterogenous variances
> data.lme2 <- lme(DV ~ A, random = ~1 | Block, data, corr =
+ corrCompSymm(form = ~1 | Block), weights = varIdent(form =
+ ~1 | Block))
> #First order autoregressive
> data.lme3 <- lme(DV ~ A, random = ~1 | Block, data, corr =
+ corrAR1(form = ~1 | Block))
```

- Compare the fit of each to the model incorporating compound symmetry

```
> anova(data.lme1, data.lme)
```

- Examine the anova table (for fixed effects) for the fitted model with the "best" covariance structure

```
> summary(data.lme)
```

- Examine the parameter estimates for the fitted model with the "best" covariance structure

```
> summary(data.lme)
```

13.12 Worked examples of real biological data sets

Example 13A: Two factor fixed (Model I) ANOVA

To investigate the importance of leaf domatia on the abundance of mites, Walter and O'Dowd (1992) shaved the domatia off the surface of one random leaf from each of 14 leaf pairs. Leaves where blocked into pairs of neighboring leaves in anticipation that different parts of a plant might have different numbers of mites. Their design represents a randomized complete block with leaf pairs as random blocks and the treatment (shaved or not) as the within block effect (from Box 10.1 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Walter and O'Dowd (1992) data set

```
> walter <- read.table("walter.csv", header = T, sep = ",")
```

Step 2 - The block vector (variable) contains a unique identifier of each leaf pair. However, R will consider this to be a *integer* vector rather than a categorical *factor*. In order to ensure that this variable is treated as a factor we need to redefine its class

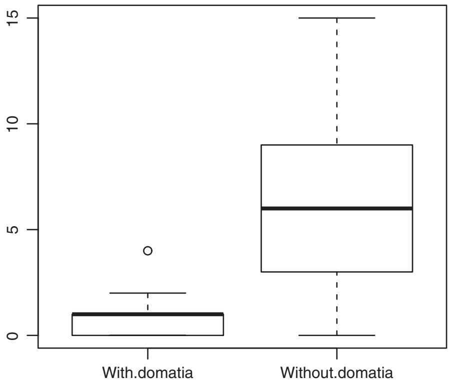
```
> walter$BLOCK <- factor(walter$BLOCK)
> class(walter$BLOCK)
[1] "factor"
```

Step 3 (Key 13.2) - Assess assumptions of normality and homogeneity of variance for the main null hypothesis that there is no effect of shaving domatia on the number of mites found on leaves.

According to Table 13.1, the MS_{Resid} (individual leaves within leaf pairs) should be used as the replicates for this hypothesis irrespective of whether a blocking interaction (the consistency of the effect of shaving is across leaf pairs) is likely to be present or not.

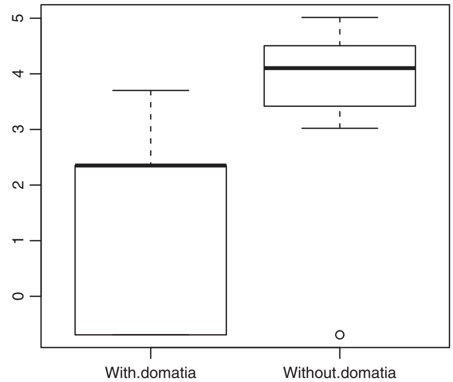
MITE

```
> boxplot(MITE ~ TREAT, walter)
```



log transformed MITE[d]

```
> boxplot(log(0.5 + (MITE *
      10)) ~ TREAT, walter)
```

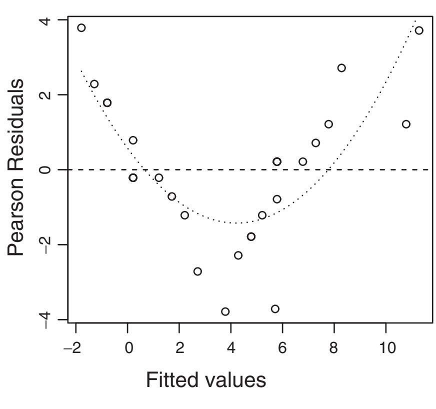


Conclusions - Strong evidence of unequal variance, potentially due to non-normality. Logarithmic transformation to normalize is an improvement.

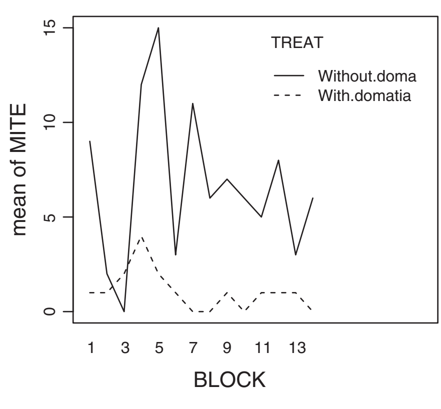
Step 4 (Key 13.2) - Investigate whether or not there is any evidence of a block by treatment interaction.

Response variable: MITE

```
> library(alr3)
> resplot(lm(MITE ~ BLOCK +
    TREAT, walter))
    t value     Pr(>|t|)
6.452132e+00 1.102875e-10
```




```
> with(walter, interaction.plot
      (BLOCK, TREAT, MITE))
```

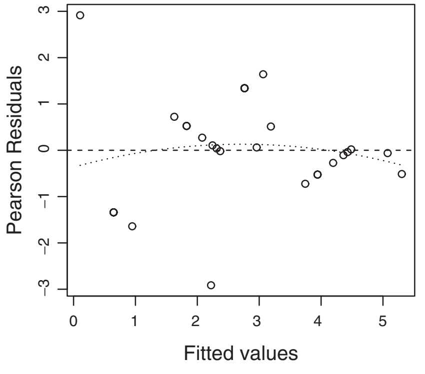


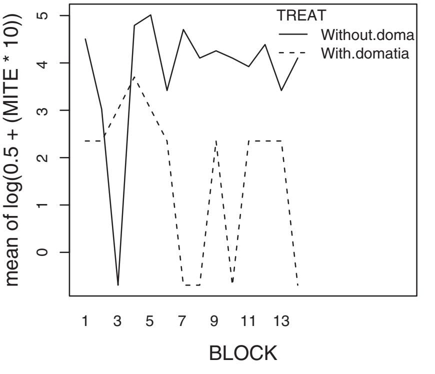
[d] Note that due to the presence of zero values Walter and O'Dowd (1992) added a small constant (0.5) to each of the mite counts prior to logarithmic transformation. They also multiplied the number of mites by 10, although it is not clear why.

Response variable: log transformed MITE

```
> library(alr3)
> resplot(lm(log(0.5 + (MITE *
+      10)) ~ BLOCK + TREAT,
+      walter))
   t value    Pr(>|t|)
-0.4644124  0.6423523
```

```
> with(walter, interaction.plot
+      (BLOCK, TREAT, log(0.5 +
+      (MITE * 10))))
```





Conclusions - Strong evidence of a blocking interaction with the raw data (curvature pattern in the residuals and a significant Tukey's non-additivity statistic), yet no evidence with the log transformed data.

Step 5 (Key 13.5) - Determine whether or not the design is balanced (equal sample sizes).

```
> replications(log(0.5 + (MITE * 10)) ~ Error(BLOCK) + TREAT,
              data = walter)
TREAT
   14
> library(biology)
> is.balanced(log(0.5 + (MITE * 10)) ~ Error(BLOCK) + TREAT,
             data = walter)
[1] TRUE
```

Conclusions - The design is completely balanced. Each of the 14 leaf pairs have exactly one leaf for each treatment (shaved or not).

Step 6 (Key 13.6) - Fit the randomized complete block linear model (additive or non-additive).

```
> walter.aov <- aov(log(0.5 + (MITE * 10)) ~ Error(BLOCK) + TREAT,
+      data = walter)
```

Step 7 (Key 13.6) - Examine the anova table.

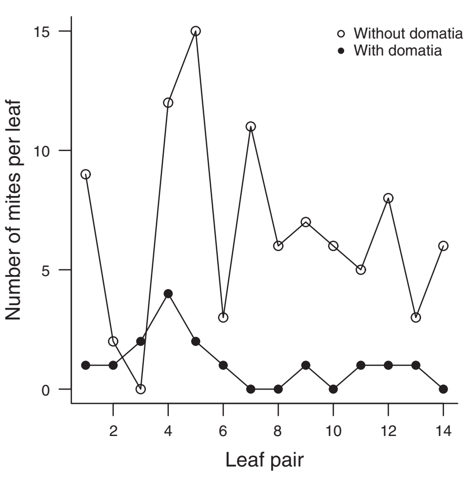
```
> summary(walter.aov)
Error: BLOCK
         Df  Sum Sq Mean Sq F value Pr(>F)
Residuals 13 23.0576  1.7737
```

```
Error: Within
          Df Sum Sq Mean Sq F value   Pr(>F)
TREAT      1 31.341  31.341  11.315 0.005084 **
Residuals 13 36.007   2.770
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - the number of mites were found to be significantly lower on shaved leaves (those without domatia) than unshaved leaves.

Step 8 (Key 12.18) - Summarize the trends in a plot.

```
> op <- par(mar = c(4, 4, 0.1, 0.1))
> plot(MITE ~ as.numeric(BLOCK), data = walter, type = "n",
+     axes = F, xlab = "", ylab = "")
> with(subset(walter, TREAT == "Without.domatia"), points(MITE ~
+     as.numeric(BLOCK), pch = 21, type = "o", lwd = 1))
> with(subset(walter, TREAT == "With.domatia"), points(MITE ~
+     as.numeric(BLOCK), pch = 16, type = "o", lwd = 1, lty = 1))
> axis(1, cex.axis = 0.8)
> mtext(text = "Leaf pair", side = 1, line = 3)
> axis(2, cex.axis = 0.8, las = 1)
> mtext(text = "Number of mites per leaf", side = 2, line = 3)
> legend("topright", leg = c("Without domatia", "With domatia"),
+     lty = 0, pch = c(21, 16), bty = "n", cex = 0.7)
> box(bty = "l")
> par(op)
```



Example 13B: Simple repeated measures ANOVA

Driscoll and Roberts (1997) investigated the impact of fuel-reduction burning on the number of individual male frogs calling. Matched burnt and unburnt sites were blocked within six drainages, and the difference in number of calling male frogs between the sites was recorded for each drainage on three occasions (a 1992 pre-burn and two post burns in 1993 and

1994). They were primarily interested in investigating whether the mean difference in number of calling frogs between burn and control sites differed between years (from Box 10.2 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Driscoll and Roberts (1997) data set

```
> driscoll <- read.table("driscoll.csv", header = T, sep = ",")
```

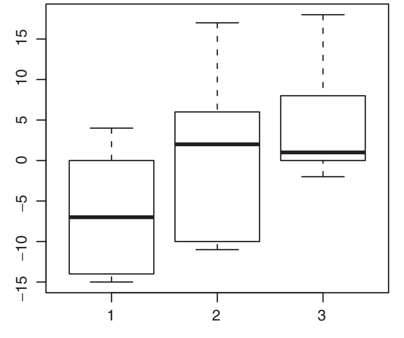
Step 2 - The year vector is represented by single integer entries, and therefore to ensure that it is treated as a factor, we need to manually define it as such.

```
> driscoll$YEAR <- factor(driscoll$YEAR)
```

Step 3 (Key 13.2) - Assess assumptions of normality and homogeneity of variance for the main null hypothesis that there is no effect of year on the difference in male frogs calling between burnt and unburnt sites (within blocks).
According to Table 13.1, the MS_{Resid} (individual frog call differences) should be used as the replicates for this hypothesis irrespective of whether a blocking interaction is likely to be present or not.

```
> boxplot(CALLS ~ YEAR, driscoll)
```

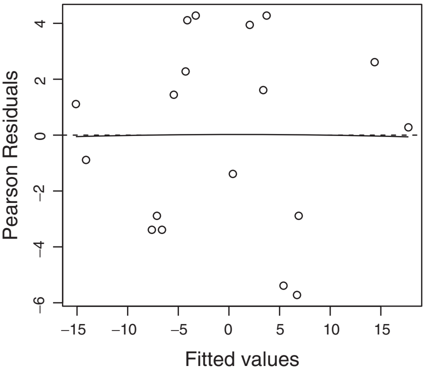
Conclusions - No evidence of unequal variance, and the hypothesis test should be robust enough to account for any potential non-normality.

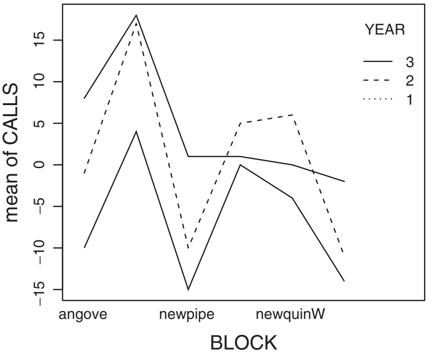


Step 4 (Key 13.2) - Investigate whether or not there is any evidence of a block by year interaction.

```
> library(alr3)
> resplot(lm(CALLS ~ BLOCK +
+     YEAR, driscoll))
    t value    Pr(>|t|)
-0.03404365  0.97284234
```

```
> with(driscoll, interaction.plot
+     (BLOCK, YEAR, CALLS))
```





Conclusions - No strong evidence of a blocking interaction.

Step 5 (Key 13.5) - Determine whether or not the design is balanced (equal sample sizes).

```
> replications(CALLS ~ Error(BLOCK) + YEAR, data = driscoll)
YEAR
   6
> library(biology)
> is.balanced(CALLS ~ Error(BLOCK) + YEAR, data = driscoll)
[1] TRUE
```

Conclusions - The design is completely balanced. Each of the three years were represented within each of the 6 drainages (blocks).

Step 6 (Key 13.6) - Fit the repeated measures linear model (additive or non-additive).

```
> driscoll.aov <- aov(CALLS ~ Error(BLOCK) + YEAR, data = driscoll)
```

Step 7 (Key 13.6) - Examine the anova table. Since, the levels of year cannot be randomized within each block (the order must always be 1, 2, 3), we might suspect that sphericity will be an issue. Consequently, we will calculate Greenhouse-Geisser and Huynh-Feldt epsilon values and adjust the hypothesis tests accordingly.

```
> library(biology)
> AnovaM(driscoll.aov, RM = T)
   Sphericity Epsilon Values
-------------------------------
 Greenhouse.Geisser Huynh.Feldt
          0.7121834   0.9153904


Anova Table (Type I tests)
Response: CALLS
Error: BLOCK
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  5 955.61  191.12

Error: Within
          Df Sum Sq Mean Sq F value   Pr(>F)
YEAR       2 369.44  184.72  9.6601 0.004615 **
Residuals 10 191.22   19.12
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Greenhouse-Geisser corrected ANOVA table
Response: CALLS
Error: BLOCK
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  5 955.61  191.12
```

```
Error: Within
               Df Sum Sq Mean Sq F value  Pr(>F)
YEAR       1.4244 369.44  184.72  9.6601 0.00722 **
Residuals 10.0000 191.22   19.12
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Huynh-Feldt corrected ANOVA table
Response: CALLS
Error: BLOCK
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  5 955.61  191.12

Error: Within
               Df Sum Sq Mean Sq F value   Pr(>F)
YEAR       1.8308 369.44  184.72  9.6601 0.005196 **
Residuals 10.0000 191.22   19.12
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - The Greenhouse-Geisser epsilon (0.712) confirmed a deviation from sphericity and thus the Greenhouse-Geisser adjusted P-value (0.013) should be used. Analysis indicates that there was a significant effect of year (time prior or post fuel reduction burn) on the difference in number of males calling between burnt and unburnt sites.

Step 8 (Key 12.8) - Quinn and Keough (2002) also presented the output of a multivariate analysis of variance (MANOVA) as an alternative.

```
> #convert the data to wide format
> dris.rm <- reshape(driscoll, timevar = "YEAR", v.names = "CALLS",
+ idvar = "BLOCK", direction = "wide")
> #fit the simple MANOVA
> dris.lm <- lm(cbind(CALLS.1, CALLS.2, CALLS.3) ~ 1, dris.rm)
> #create a data frame that defines the intra-block design
> idata <- data.frame(YEAR = as.factor(c(1, 2, 3)))
> #use the Anova (car) function to estimate the MANOVA test
  statistics
> (av.ok <- Anova(dris.lm, idata = idata, idesign = ~YEAR))
Type III Repeated Measures MANOVA Tests: Pillai test statistic
            Df test stat approx F num Df den Df  Pr(>F)
(Intercept)  1    0.0028   0.0142      1      5 0.90965
YEAR         1    0.8725  13.6913      2      4 0.01625 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> summary(av.ok)     #NOTE the output has been truntcated
Type III Repeated Measures MANOVA Tests:

------------------------------------------
```

```
Term: (Intercept)

 Response transformation matrix:
        (Intercept)
CALLS.1           1
CALLS.2           1
CALLS.3           1

Sum of squares and products for the hypothesis:
            (Intercept)
(Intercept)    8.166667

Sum of squares and products for error:
            (Intercept)
(Intercept)    2866.833

Multivariate Tests: (Intercept)
                 Df test stat  approx F num Df den Df  Pr(>F)
Pillai            1 0.0028406 0.0142434      1      5 0.90965
Wilks             1 0.9971594 0.0142434      1      5 0.90965
Hotelling-Lawley  1 0.0028487 0.0142434      1      5 0.90965
Roy               1 0.0028487 0.0142434      1      5 0.90965

------------------------------------------

Term: YEAR

 Response transformation matrix:
        YEAR1 YEAR2
CALLS.1     1     0
CALLS.2     0     1
CALLS.3    -1    -1

Sum of squares and products for the hypothesis:
         YEAR1     YEAR2
YEAR1 704.1667 216.66667
YEAR2 216.6667  66.66667

Sum of squares and products for error:
         YEAR1    YEAR2
YEAR1 232.8333 215.3333
YEAR2 215.3333 269.3333

Multivariate Tests: YEAR
                 Df test stat  approx F num Df den Df   Pr(>F)
Pillai            1  0.872540 13.691253      2      4 0.016246 *
```

```
Wilks             1  0.127460 13.691253      2      4 0.016246 *
Hotelling-Lawley  1  6.845627 13.691253      2      4 0.016246 *
Roy               1  6.845627 13.691253      2      4 0.016246 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Univariate Type III Repeated-Measures ANOVA Assuming Sphericity

                SS num Df Error SS den Df      F   Pr(>F)
(Intercept)   2.72      1   955.61      5 0.0142 0.909649
YEAR        369.44      2   191.22     10 9.6601 0.004615 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Mauchly Tests for Sphericity

     Test statistic p-value
YEAR        0.59587 0.35506


Greenhouse-Geisser and Huynh-Feldt Corrections
 for Departure from Sphericity

      GG eps Pr(>F[GG])
YEAR 0.71218    0.01252 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

      HF eps Pr(>F[HF])
YEAR 0.91539   0.006175 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Multivariate tests for the within block effects, all concur that there is an effect of YEAR on the differences in number of male frogs calling. Whilst the Mauchly test for sphericity does not indicate a problem with sphericity (*P*=0.355), Greenhouse-Geisser epsilon suggest substantial departures from sphericity (0.712). Univariate repeated measures ANOVA corrected for sphericity yield the same outcomes as Step 13B above.

Step 9 - Quinn and Keough (2002) suggested a logical planned contrast of year 1 (pre burn) with the year 2 and 3 (post burn). Note that as sphericity was clearly violated, this comparison must be performed using a separately calculated error term.

```
> driscoll.aov2 <- aov(CALLS ~ C(YEAR, c(1, -0.5, -0.5), 1) +
+     BLOCK + Error(BLOCK/C(YEAR, c(1, -0.5, -0.5), 1)),
+     data = driscoll)
> summary(driscoll.aov2)
```

```
Error: BLOCK
      Df Sum Sq Mean Sq
BLOCK  5 955.61  191.12

Error: BLOCK:C(YEAR, c(1, -0.5, -0.5), 1)
                             Df Sum Sq Mean Sq F value   Pr(>F)
C(YEAR, c(1, -0.5, -0.5), 1)  1 336.11  336.11  29.715 0.002823 **
Residuals                     5  56.56   11.31
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  6    168      28
```

Conclusions - the burnt-unburnt differences in number of frogs calling was significantly lower prior to the burn than after.
Purely for illustrative purposes, Quinn and Keough (2002) also highlighted the exploration of polynomial trends[e] (specifically a linear trend) across years.

```
> driscoll.aov3 <- aov(CALLS ~ C(YEAR, poly, 1) + BLOCK +
+     Error(BLOCK/C(YEAR, poly, 1)), data = driscoll)
> summary(driscoll.aov3)
Error: BLOCK
      Df Sum Sq Mean Sq
BLOCK  5 955.61  191.12

Error: BLOCK:C(YEAR, poly, 1)
                 Df Sum Sq Mean Sq F value  Pr(>F)
C(YEAR, poly, 1)  1 352.08  352.08  15.122 0.01154 *
Residuals         5 116.42   23.28
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  6 92.167  15.361
```

Conclusions - there was a significant linear trend in burnt-unburnt differences in number of frogs calling across the years.

Step 10 (Key 13.13) - Finally, rather than attempting a *post-hoc* correct for the *estimated* departures from compound symmetry (sphericity), we could instead fit a linear mixed effects model (lme) in which the within block correlation structure is specified and incorporated.

[e] Note that this contrast is not independent of the previous contrast and is perhaps not of great biological meaning given that the impact occurred mid-way through the years rather than at the start.

- Fit the linear mixed effects model with a range of covariance structures

```
> library(nlme)
> #fit the lme with unstructured covariance structure
> driscoll.lme <- lme(CALLS ~ YEAR, random =~1 | BLOCK,
+ data = driscoll, correlation = corSymm(form = ~1 | BLOCK))
> #fit the lme assuming compound symmetry (sphericity)
> driscoll.lme1 <- update(driscoll.lme, correlation =
+ corCompSymm(form = ~1 | BLOCK))
> #compare the fit of the models
> anova(driscoll.lme, driscoll.lme1)
              Model df      AIC      BIC    logLik   Test  L.Ratio p-value
driscoll.lme      1  8 114.3804 120.0448 -49.19019
driscoll.lme1     2  6 115.7165 119.9648 -51.85826 1 vs 2 5.336127  0.0694

> #fit the lme with a first order autoregressive covariance structure
> driscoll.lme2 <- update(driscoll.lme, correlation = corAR1(form = ~1 |
+ BLOCK))
> driscoll.lme2
Linear mixed-effects model fit by REML
  Data: driscoll
  Log-restricted-likelihood: -51.31218
  Fixed: CALLS ~ YEAR
(Intercept)       YEAR2       YEAR3
   -6.50000     7.50000    10.83333

Random effects:
 Formula: ~1 | BLOCK
        (Intercept) Residual
StdDev: 0.002177376 8.230684

Correlation Structure: AR(1)
 Formula: ~1 | BLOCK
 Parameter estimate(s):
      Phi
0.758245
Number of Observations: 18
Number of Groups: 6
```

Conclusions - ρ (autocorrelation parameter) estimated to be 0.758245.

```
> #compare the fit of the models
> anova(driscoll.lme2, driscoll.lme1)
              Model df      AIC      BIC    logLik
driscoll.lme2     1  6 114.6244 118.8727 -51.31218
driscoll.lme1     2  6 115.7165 119.9648 -51.85826

> #fit the lme with a first order autoregressive covariance and heterogenous
> #variances structure
> driscoll.lme3 <- update(driscoll.lme, correlation = corAR1(form = ~1 |
+ BLOCK), weights = varIdent(form = ~1 | BLOCK))
> #compare the fit of the models
> anova(driscoll.lme3, driscoll.lme1)
              Model df      AIC      BIC    logLik   Test  L.Ratio p-value
driscoll.lme3     1 11 120.1991 127.9876 -49.09953
driscoll.lme1     2  6 115.7165 119.9648 -51.85826 1 vs 2 5.517442   0.356
```

Conclusions - Inferential evidence for a deviation from compound symmetry is not significant (AIC and BIC differentials less than 2 and logLikelihood statistic not significantly for any alternative models).

- Examine the anova table for the "best" lme

```
> anova(driscoll.lme1)
            numDF denDF  F-value p-value
(Intercept)     1    10 0.014243  0.9074
YEAR            2    10 9.660081  0.0046
```

Conclusions - There is a significant effect of year on the difference in number of males calling between burnt and unburnt sites.

- Fit the planned contrast of year 1 (pre burn) versus year 2 and 4 (post burns).

```
> library(gmodels)
> fit.contrast(driscoll.lme1, "YEAR", c(1, -0.5, -0.5))
                        Estimate Std. Error   t-value    Pr(>|t|)
YEAR c=( 1 -0.5 -0.5 ) -9.166667   2.186448 -4.192492 0.001850619
```

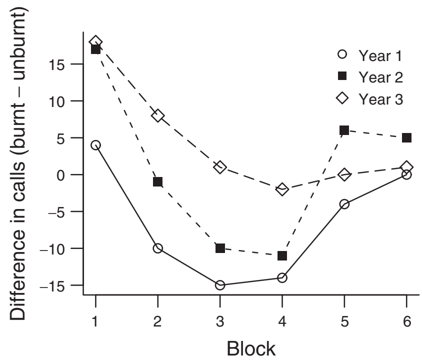
- Examine the polynomial trends.

```
> library(gmodels)
> fit.contrast(driscoll.lme1, "YEAR", t(contr.poly(3, c(1, 2, 3))))
        Estimate Std. Error    t-value    Pr(>|t|)
YEAR.L  7.660323   1.785227  4.2909509 0.001583836
YEAR.Q -1.701035   1.785227 -0.9528391 0.363134925
```

Step 11 - Summarize the trends in a plot.

```
> # create a blocking variable (called BLCK) that represents the
> #order of data in rows
> driscoll$BLCK <- as.numeric(factor(driscoll$BLOCK, levels =
+ unique(driscoll$BLOCK)))
> # construct the base plot with different point types for each
> # treatment
> plot(CALLS ~ BLCK, data = driscoll, type = "n", axes = F, xlab = "",
+ ylab = "")
> with(subset(driscoll, YEAR == "1"), points(CALLS ~ BLCK, pch = 21,
+ type = "o", lwd = 1))
> with(subset(driscoll, YEAR == "2"), points(CALLS ~ BLCK, pch = 15,
+ type = "o", lwd = 1, lty = 2))
> with(subset(driscoll, YEAR == "3"), points(CALLS ~ BLCK, pch = 5,
+ type = "o", lwd = 1, lty = 5))
> # create the axes and their labels
> axis(1, cex.axis = 0.8)
> mtext(text = "Block", side = 1, line = 3)
> axis(2, cex.axis = 0.8, las = 1)
> mtext(text = "Difference in calls (burnt - unburnt)", side = 2,
+ line = 3)
> # include a legend
```

```
> legend("topright",leg = c("Year 1", "Year 2", "Year 3"), lty = 0,
+ pch = c(21, 15, 5), bty = "n",
+ cex=0.9)
> box(bty="l")
```



Example 13C: Unreplicated ANOVA with missing observations

Quinn and Keough (2002) presented a modification of the Driscoll and Roberts (1997) data set in which one of the observations (newpipe year 2) was removed - so as to demonstrate and contrast the options for dealing with missing observations (=cells) in unreplicated designs (see Box 10.8 Quinn and Keough (2002)).

Step 1 - Prepare the data (from example 13B).

```
> driscoll1 <- driscoll
> driscoll1[9, 4] <- NA
```

Step 2 - As we have already examined the assumptions associated with the relevant design, we will skip straight to the analysis options

Option 1- Omit the newpipe block

```
> driscoll1.aov <- aov(CALLS ~ Error(BLOCK) + YEAR,
+     data = driscoll1, subset = BLOCK != "newpipe")
> summary(driscoll1.aov)
Error: BLOCK
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  4 747.07  186.77

Error: Within
          Df  Sum Sq Mean Sq F value  Pr(>F)
YEAR       2 272.133 136.067   7.044 0.01721 *
Residuals  8 154.533  19.317
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Option 2- Substitute a new value (by solving the equation $\hat{y}_{ij} = \overline{y}_i + \overline{y}_j - \overline{y}$ - that is, the expected value of any given observation within a specific year/block is equal to the sum of the mean of the block, the mean of the year and the negative of the overall mean).

```
> #calculate the mean of the newpipe block
> BM<-with(driscoll1, tapply(CALLS, BLOCK, mean, na.rm=T))
+ ["newpipe"]
> #calculate the mean of year 2
> YM<-with(driscoll1, tapply(CALLS, YEAR, mean, na.rm=T))["2"]
> #calculate the overall mean
> M<-mean(driscoll1$CALLS,na.rm=T)
> #duplicate the data set and work on the duplicate
> driscoll2 <- driscoll1
> #substitute the new value into the data frame
> driscoll2[9,3]<-YM+BM-M
> #fit the linear model
> driscoll2.aov <- aov(CALLS~Error(BLOCK)+YEAR, data=driscoll2)
> summary(driscoll2.aov)
Error: BLOCK
          Df Sum Sq Mean Sq F value Pr(>F)
YEAR       1 116.74  116.74   0.625 0.4734
Residuals  4 747.07  186.77

Error: Within
          Df Sum Sq Mean Sq F value   Pr(>F)
YEAR       2 384.12  192.06  10.135 0.004957 **
Residuals  9 170.55   18.95
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> #then make adjustments to the F-ratio and Pvalue (to reflect a
> #reduction) in residual degrees of freedom by one for each
> #substituted value)
> (MSresid <- summary(driscoll2.aov)[[2]][[1]]["Residuals",
  "Sum Sq"]/9)
[1] 18.95

> (Fyear <- summary(driscoll2.aov)[[2]][[1]]["YEAR","Mean Sq"]/
  MSresid)
[1] 10.13500

> (Pvalue <- 1-pf(Fyear, 2,8))
[1] 0.006412925
```

Option 3- Compare appropriate full and reduced models

```
> driscoll1.aovF <- aov(CALLS ~ BLOCK + YEAR, data = driscoll1)
> driscoll1.aovR <- aov(CALLS ~ BLOCK, data = driscoll1)
> anova(driscoll1.aovF, driscoll1.aovR)
Analysis of Variance Table

Model 1: CALLS ~ BLOCK + YEAR
Model 2: CALLS ~ BLOCK
```

```
  Res.Df     RSS Df Sum of Sq      F   Pr(>F)
1      9  170.55
2     11  554.67 -2   -384.12 10.135 0.004957 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

OR

```
> anova(driscoll1.aovF)
Analysis of Variance Table

Response: CALLS
          Df Sum Sq Mean Sq F value   Pr(>F)
BLOCK      5 863.80  172.76  9.1167 0.002500 **
YEAR       2 384.12  192.06 10.1350 0.004957 **
Residuals  9 170.55   18.95
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that options 2 and 3 are only valid if we assume that there are no block by year interactions and are both difficult to make reasonable sphericity deviation estimates and corrections.

Option 4- fit some alternative linear mixed effects models (with different covariance structures).

```
> library(nlme)
> #No structure
> driscoll1.lme1 <- lme(CALLS ~ YEAR, random = ~1 | BLOCK, data = driscoll1,
+ subset = !is.na(CALLS))
> #Unstructured
> driscoll1.lme2 <- lme(CALLS ~ YEAR, random = ~1 | BLOCK, data = driscoll1,
+ subset = !is.na(CALLS), correlation = corSymm(form = ~1 | BLOCK))
> #Compound symmetry
> driscoll1.lme3 <- update(driscoll1.lme1, correlation =
+ corCompSymm(form = ~1 | BLOCK))
> #First order autoregressive
> driscoll1.lme4 <- lme(CALLS ~ YEAR, random = ~1 | BLOCK, data = driscoll1,
+ subset = !is.na(CALLS), correlation = corAR1(form = ~1 | BLOCK))
> driscoll1.lme4 <- update(driscoll1.lme1, correlation =
+ corAR1(form = ~1 | BLOCK))
> #Compare each to compound symmetry
> anova(driscoll1.lme3, driscoll1.lme1, driscoll1.lme2, driscoll1.lme4)
               Model df      AIC      BIC    logLik   Test  L.Ratio p-value
driscoll1.lme3     1  6 108.8226 112.6570 -48.41133
driscoll1.lme1     2  5 106.8226 110.0179 -48.41133 1 vs 2 0.000000  1.0000
driscoll1.lme2     3  8 109.2339 114.3464 -46.61695 2 vs 3 3.588753  0.3094
driscoll1.lme4     4  6 107.7288 111.5632 -47.86441 3 vs 4 2.494909  0.2872
> anova(driscoll1.lme3)
            numDF denDF   F-value p-value
(Intercept)     1     9  0.002742  0.9594
YEAR            2     9 10.264400  0.0048
```

Note that the lme method also implicitly incorporates the correlation structure of the data and therefore arguably handles the issues of sphericity (which are exacerbated with missing observations) more appropriately than ANOVA. Nevertheless, none of the alternative covariance structures resulted in significantly better fits (based on AIC values) than a model incorporating compound symmetry (`driscoll1.lme3`). Consistent with other analyses, the impact of burning was found to differ significantly over time.

Example 13D: Two factor randomized block design

To illustrate two factor randomized blocking designs, Doncaster and Davey (2007)[f] introduced a fictitious data set in which all the levels of sewing density (factor A) and fertilizer treatments (Factor B) were randomly allocated within blocks (Factor S) which in turn where arranged across a heterogeneous landscape. The response variable was the yield of crop (Y).

Step 1 - Import (section 2.3) the crop yield data set

```
> crop <- read.table("crop.csv", header = T, sep = ",")
```

Step 2 - Each of the categorical variables are listed as *integer* vectors rather than a categorical *factors*. In order to ensure that this variable is treated as a factor we need to redefine them.

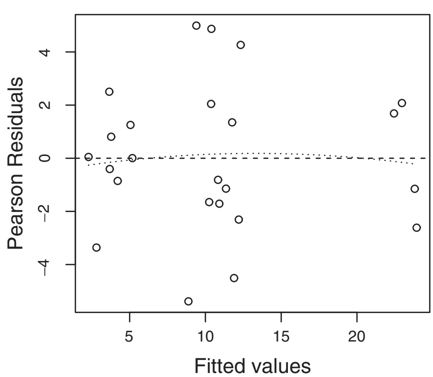
```
> crop$A <- factor(crop$A)
> crop$B <- factor(crop$B)
> crop$S <- factor(crop$S)
```

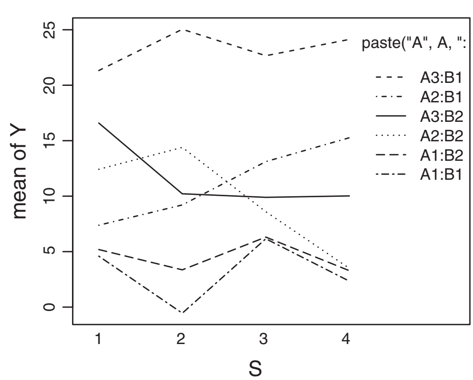
Step 3 (Key 13.2) - Assess whether there is any evidence of treatment by block interactions

Response variable: MITE

```
> library(alr3)
> resplot(lm(Y ~ S + A * B,
         crop))
   t value   Pr(>|t|)
-1.3756093  0.1689426
```

```
> with(crop, interaction.plot(S,
+     paste("A", A, ":B", B,
+     sep = ""), Y))
```



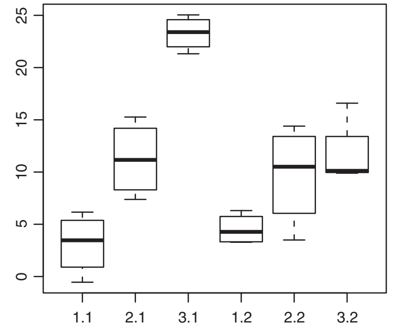


Conclusions - No clear evidence of a blocking interaction (no obvious curvature pattern in the residuals and non-significant Tukey's non-additivity statistic). Hence according to Table 13.2, the MS_{Resid} (individual treatment plots within the blocks) should be used as the replicates for each of the hypotheses.

[f] The data and example output can be found on the book's web page http://www.southampton.ac.uk/ cpd/anovas/datasets/.

Step 4 (Key 13.2) - Assess assumptions of normality and homogeneity of variance for the main null hypotheses that there are no effects of sewing density, fertilizer treatment or no interaction between the two on the yield of crop.

```
> boxplot(Y ~ A * B, crop)
```



Conclusions - No evidence of unequal variance or non-normality.

Step 5 (Key 13.5) - Determine whether or not the design is balanced (equal sample sizes).

```
> replications(Y ~ Error(S) + A * B, data = crop)
  A   B A:B
  8  12   4

> library(biology)
> is.balanced(Y ~ Error(S) + A * B, data = crop)
[1] TRUE
```

Conclusions - The design is completely balanced. Each of the four field blocks have exactly one replicate of each combination of the levels of A and B.

Step 6 (Key 13.6) - Fit the randomized complete block linear model (additive).

```
> crop.aov <- aov(Y ~ Error(S) + A * B, data = crop)
```

Note, a non-additive model would be fit as:

```
> crop.aov <- aov(Y ~ A * B + Error(S/A + S/B), data = crop)
```

Step 7 (Key 13.6) - Examine the anova table.

```
> summary(crop.aov)
Error: S
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  3 9.0746  3.0249
```

[a] Note that due to the presence of zero values Walter and O'Dowd (1992) added a small constant (0.5) to each of the mite counts prior to logarithmic transformation. They also multiplied the number of mites by 10, although it is not clear why.

```
Error: Within
          Df Sum Sq Mean Sq F value    Pr(>F)
A          2 745.36  372.68 32.6710 3.417e-06 ***
B          1  91.65   91.65  8.0346  0.012553 *
A:B        2 186.37   93.18  8.1690  0.003983 **
Residuals 15 171.11   11.41
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - there is a significant sewing density by fertilizer treatment interaction.

Step 8 - Examine the main effects

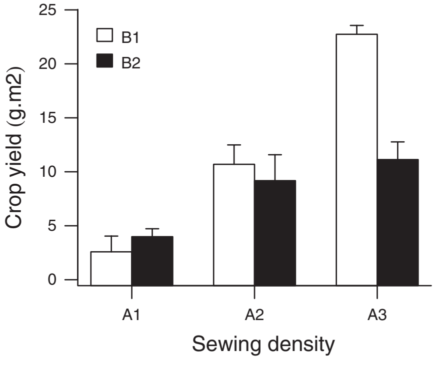
```
> #Examine the effects of B at A=1
> summary(mainEffects(crop.aov, at = A == "1"))
Error: S
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  3 9.0746  3.0249

Error: Within
          Df  Sum Sq Mean Sq F value    Pr(>F)
INT        4 1019.46  254.87 22.3429 3.563e-06 ***
B          1    3.91    3.91  0.3432    0.5667
Residuals 15  171.11   11.41
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> #Examine the effects of B at A=2
> summary(mainEffects(crop.aov, at = A == "2"))
Error: S
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  3 9.0746  3.0249

Error: Within
          Df  Sum Sq Mean Sq F value    Pr(>F)
INT        4 1018.78  254.70 22.3280 3.578e-06 ***
B          1    4.59    4.59  0.4028    0.5352
Residuals 15  171.11   11.41
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> #Examine the effects of B at A=3
> summary(mainEffects(crop.aov, at = A == "3"))
Error: S
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  3 9.0746  3.0249
```

```
Error: Within
          Df Sum Sq Mean Sq F value    Pr(>F)
INT        4 753.87  188.47  16.522 2.267e-05 ***
B          1 269.51  269.51  23.627 0.0002077 ***
Residuals 15 171.11   11.41
---
Signi f. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - there is significant effect of fertilizer treatment (B1 vs B2) on crop yield, but only at a sewing density of A3.

Step 9 - Summarize the trends in a plot.

```
> crop.means <- with(crop, t(tapply(Y, list(A, B), mean)))
> library(gmodels)
> crop.se <- with(crop, t(tapply(Y, list(A, B), function(x) ci(x,
+     na.rm = T)[4])))
> ofst <- min(crop$Y)
> xs <- barplot(crop.means, ylim = range(crop$Y, na.rm = T),
+     beside = T, axes = F, xpd = T, axisnames = F,
+     axis.lty = 2, legend.text = F, col = c(0, 1), offset = ofst)
> arrows(xs, crop.means + ofst, xs, crop.means + crop.se + ofst,
+     code = 2, angle = 90, len = 0.05)
> axis(2, las = 1)
> axis(1, at = apply(xs, 2, median), lab = c("A1", "A2", "A3"),
+     padj = 1, mgp = c(0, 0, 0))
> mtext(2, text = expression(paste("Crop yield ", (g.m^2))), line = 3,
+     cex = 1)
> mtext(1, text = "Sewing density", line = 3, cex = 1)
> box(bty = "l", xpd = 1)
> legend("topleft", leg = c("B1", "B2"), fill = c(0, 1), col = c(0,
+     1), bty = "n", cex = 1)
```



Example 13E: Non-parametric randomized block

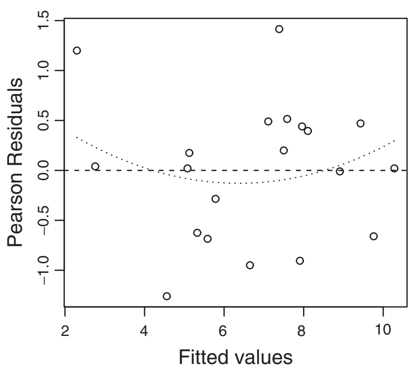
Zar (1999) illustrated two approaches (Example 12.6 and Example 12.7) to non-parametric unreplicated factorial designs. Both approaches made use of data collected on the weight gained by guinea pigs maintained on one of four diets. Each guinea pig was individually caged and in an attempt to account for any variability in weight gain resulting from differences in cage position (the holding facility was potentially not homogeneous with respect to lighting, noise, temperature etc), guinea pigs were also blocked into sets of four individuals (one on each diet) whose cages were in close proximity. Within a block, individuals were randomly assigned to one of the four treatment diets. Whilst the data do not show concerning deviations from the parametric assumptions of normality and equal variance, for the purpose of illustration we will assume these assumptions have been violated.

Step 1 - Import (section 2.3) the Zar (1999) guinea pig data set

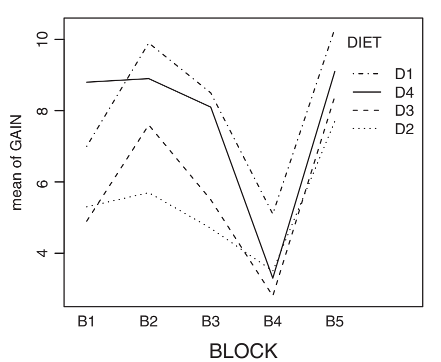
```
> gp <- read.table("gp.csv", header = T, sep = ",")
```

Step 2 (Key 13.2) - Assess whether there is any evidence of treatment by block interactions
Response variable: GAIN

```
> library(alr3)
> resplot(lm(GAIN ~ BLOCK +
        DIET, gp))
  t value  Pr(>|t|)
0.9315358 0.3515765
```



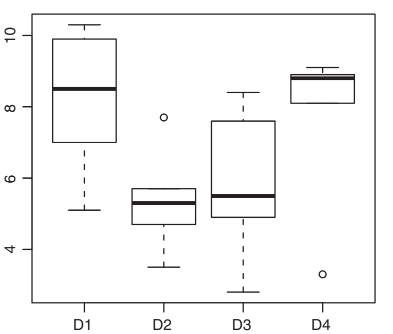

```
> with(gp, interaction.plot
      (BLOCK, DIET, GAIN))
```



Conclusions - No clear evidence of a blocking interaction (only very slight curvature pattern in the residuals and non-significant Tukey's non-additivity statistic).

Step 3 (Key 13.2) - Assess assumptions of normality and homogeneity of variance for the main null hypotheses that there are no effects of diet within Block on the weight gain of guinea pigs.

```
> boxplot(GAIN ~ DIET, gp)
```



Conclusions - Although the evidence of unequal variance and non-normality is not substantial, outliers are present and some skewness is suggested. **Note, that for the purpose of reproducing the output of one of the major texts in biostatisics, we will proceed as if the parametric assumptions had been violated**.

Step 4 (Key 13.5) - Determine whether or not the design is balanced (equal sample sizes).

```
> replications(GAIN ~ Error(BLOCK) + DIET, data = gp)
DIET
   5
> library(biology)
> is.balanced(GAIN ~ Error(BLOCK) + DIET, data = gp)
[1] TRUE
```

Conclusions - The design is completely balanced. There are exactly one of each diet treatment per block.

Step 5 (Key 13.11) - Perform a Friedman's test (Zar (1999), Example 12.6)

```
> library(pgirmess)
> friedman.test(GAIN ~ DIET | BLOCK, data = gp)
        Friedman rank sum test

data:  GAIN and DIET and BLOCK
Friedman chi-squared = 10.68, df = 3, p-value = 0.01359
```

Conclusions - there is a significant effect of diet on the weight gain of guinea pigs.

Step 6 - Perform a multiple comparisons test following a Friedman's test

```
> library(pgirmess)
> friedmanmc(gp$GAIN, gp$DIET, gp$BLOCK, p = 0.05)
Multiple comparisons between groups after Friedman test
p.value: 0.05
Comparisons
      obs.dif critical.dif difference
D1-D2       0     10.77064      FALSE
D1-D3       5     10.77064      FALSE
D1-D4       1     10.77064      FALSE
D2-D3       5     10.77064      FALSE
D2-D4       1     10.77064      FALSE
D3-D4       4     10.77064      FALSE
```

Conclusions - None of the diet types were found to be significantly different from each other, however, this is a very conservative test.

Step 7 - **Alternatively**, we could perform a randomized complete block on rank transformed data (Zar (1999)–example 12.7)

```
> summary(aov(rank(GAIN) ~ Error(BLOCK) + DIET, gp))
Error: BLOCK
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  4    400     100

Error: Within
          Df  Sum Sq Mean Sq F value    Pr(>F)
DIET       3 195.400  65.133   11.23 0.0008471 ***
Residuals 12  69.600   5.800
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - there is a significant effect of diet on the weight gain of guinea pigs.

Step 8 - Perform a multiple comparisons test rank based randomized block analysis

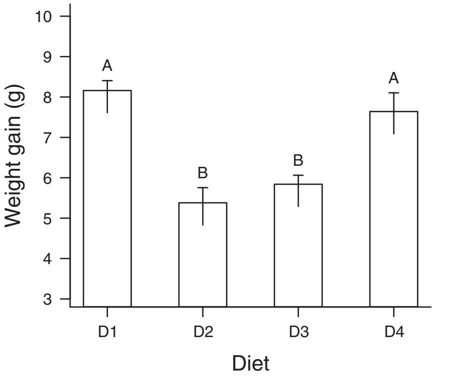
```
> library(multcomp)
> summary(glht(aov(rank(GAIN) ~ BLOCK + DIET, gp),
         linfct = mcp(DIET = "Tukey")))
         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = rank(GAIN) ~ BLOCK + DIET, data = gp)

Linear Hypotheses:
             Estimate Std. Error t value Pr(>|t|)
D2 - D1 == 0   -7.000      1.523  -4.596  0.00279 **
D3 - D1 == 0   -6.200      1.523  -4.070  0.00740 **
D4 - D1 == 0   -0.800      1.523  -0.525  0.95130
D3 - D2 == 0    0.800      1.523   0.525  0.95135
D4 - D2 == 0    6.200      1.523   4.070  0.00712 **
D4 - D3 == 0    5.400      1.523   3.545  0.01827 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

Conclusions - The weight gain of guinea pigs on diets one and four was significantly greater than that on either diet two or three.

Step 9 - Summarize the trends in a plot.

```
> gp.means <- with(gp, t(tapply(GAIN, DIET, mean)))
> library(gmodels)
> gp.res <- resid(aov(GAIN ~ BLOCK + DIET, gp))
> gp.se <- with(gp, t(tapply(gp.res, DIET, function(x) ci(x,
+ na.rm = T)[4])))
```

```
> xs <- barplot(gp.means, ylim = range(gp$GAIN), beside = T,
+     axes = F, xpd = F, axisnames = F, axis.lty = 2,
+     legend.text = F, col = c(0,0))
> arrows(xs, gp.means + ofst, xs, gp.means + gp.se, code = 2,
+     angle = 90, len = 0.05)
> axis(2, las = 1)
> axis(1, at = apply(xs, 2, median), lab = c("D1", "D2", "D3",
+     "D4"), padj = 1, mgp = c(0, 0, 0))
> mtext(2, text = expression(paste("Weight gain ", (g))), line = 3,
+     cex = 1)
> mtext(1, text = "Diet", line = 3, cex = 1)
> box(bty = "l", xpd = 1)
> text(gp.means + gp.se + 0.5 ~ xs, lab = c("A", "B", "B", "A"))
```



14

Partly nested designs: split plot and complex repeated measures

Split-plot[a] designs extend unreplicated factorial (randomized complete block and simple repeated measures) designs by incorporating an additional factor whose levels are applied to entire blocks. Similarly, complex repeated measures designs are repeated measures designs in which there are different types of subjects. Split-plot and complex repeated measures designs are depicted diagrammatically in Figure 14.1 .

Such designs are often referred to as partly nested designs which reflects the fact that blocks are (partly[b]) nested within the main between blocking factor. These designs include both within and between block (subject) effects and as a result, they are subject to the considerations of both nested (Chapter 11) and unreplicated factorial designs (Chapter 13). Whilst most of the issues have therefore already been covered separately in previous chapters, the popularity and additional accumulated complexity of these designs warrants special treatment.

Consider the example of a randomized complete block presented at the start of Chapter 13. Blocks of four treatments (representing leaf packs subject to different aquatic taxa) were secured in numerous locations throughout a potentially heterogeneous stream. If some of those blocks had been placed in riffles, some in runs and some in pool habitats of the stream, the design becomes a split-plot design incorporating a between block factor (stream region: runs, riffles or pools) and a within block factor (leaf pack exposure type: microbial, macro invertebrate or vertebrate). Furthermore, the design would enable us to investigate whether the roles that different organism scales play on the breakdown of leaf material in stream are consistent across each of the major regions of a stream (interaction between region and exposure type). Alternatively (or in addition), shading could be artificially applied to half of the blocks, thereby introducing a between block effect (whether the block is shaded or not).

Extending the repeated measures examples from Chapter 13, there might have been different populations (such as different species or histories) of rats or sharks. Any single

[a] The term "split-plot" refers to the agricultural field plots for which these designs were originally devised.
[b] It is only partly, since there is only a single block within each level of the main factor.

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

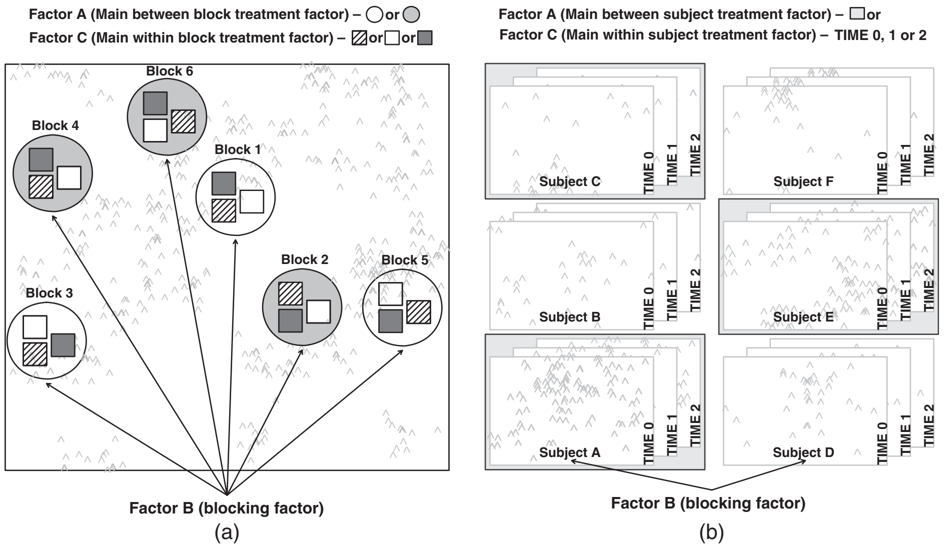


Fig 14.1 Fictitious spatial depictions of (a) split-plot and (b) complex repeated measures designs. The levels of the between block (or subject) effect (Factor A) are applied to the entire block. Note that the appropriate replicates for the effects of the between block effects are the block means. Therefore, for the effect of Factor A, n = 3 and for the effect of Factor C (within block or subject effect), n = 6.

subject (such as an individual shark or rat) can only be of one of the populations types and thus this additional factor represents a between subject effect.

14.1 Null hypotheses

There are separate null hypotheses associated with each of the main factors (and interactions), although typically, null hypotheses associated with the random blocking factors are of little interest.

14.1.1 *Factor A* - the main between block treatment effect

Fixed (typical case)

$$H_0(A): \mu_1 = \mu_2 = \ldots = \mu_i = \mu \qquad \text{(the population group means of A are all equal)}$$

The mean of population 1 is equal to that of population 2 and so on, and thus all population means are equal to an overall mean. No effect of A. If the effect of the i^{th} group is the difference between the i^{th} group mean and the overall mean ($\alpha_i = \mu_i - \mu$) then the H_0 can alternatively be written as:

$$H_0(A): \alpha_1 = \alpha_2 = \ldots = \alpha_i = 0 \qquad \text{(the effect of each group equals zero)}$$

If one or more of the α_i are different from zero (the response mean for this treatment differs from the overall response mean), the null hypothesis is not true indicating that the treatment does affect the response variable.

Random

$$H_0(A) : \sigma_\alpha^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of A.

14.1.2 *Factor B* - the blocking factor

Random (typical case)

$$H_0(B) : \sigma_\beta^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of B.

Fixed

$$H_0(B) : \mu_1 = \mu_2 = \ldots = \mu_i = \mu \qquad \text{(the population group means of B are all equal)}$$
$$H_0(B) : \beta_1 = \beta_2 = \ldots = \beta_i = 0 \qquad \text{(the effect of each chosen B group equals zero)}$$

14.1.3 *Factor C* - the main within block treatment effect

Fixed (typical case)

$$H_0(C) : \mu_1 = \mu_2 = \ldots = \mu_k = \mu \qquad \text{(the population group means of C (pooling B) are all equal)}$$

The mean of population 1 (pooling blocks) is equal to that of population 2 and so on, and thus all population means are equal to an overall mean. No effect of C within each block (Model 2) or over and above the effect of blocks. If the effect of the k^{th} group is the difference between the k^{th} group mean and the overall mean ($\gamma_k = \mu_k - \mu$) then the H_0 can alternatively be written as:

$$H_0(C) : \gamma_1 = \gamma_2 = \ldots = \gamma_k = 0 \qquad \text{(the effect of each group equals zero)}$$

If one or more of the γ_k are different from zero (the response mean for this treatment differs from the overall response mean), the null hypothesis is not true indicating that the treatment does affect the response variable.

Random

$$H_0(C) : \sigma_\gamma^2 = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to all possible levels of C (pooling B).

14.1.4 *AC interaction* - the within block interaction effect

Fixed (typical case)

$$H_0(A \times C) : \mu_{ijk} - \mu_i - \mu_k + \mu = 0 \qquad \text{(the population group means of AC combinations (pooling B) are all equal)}$$

There are no effects in addition to the main effects and the overall mean. If the effect of the ik^{th} group is the difference between the ik^{th} group mean and the overall mean ($\gamma_{ik} = \mu_i - \mu$) then the H_0 can alternatively be written as:

$$\mathrm{H}_0(AC) : \alpha\gamma_{11} = \alpha\gamma_{12} = \ldots = \alpha\gamma_{ik} = 0 \qquad \text{(the interaction is equal to zero)}$$

Random

$$\mathrm{H}_0(AC) : \sigma^2_{\alpha\gamma} = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to any interaction effects (pooling B).

14.1.5 *BC interaction* - the within block interaction effect

Typically random

$$\mathrm{H}_0(BC) : \sigma^2_{\beta\gamma} = 0 \qquad \text{(population variance equals zero)}$$

There is no added variance due to any block by within block interaction effects. That is, the patterns amongst the levels of C are consistent across all the blocks. Unless each of the levels of Factor C are replicated (occur more than once) within each block, this null hypotheses about this effect cannot be tested.

14.2 Linear models

The linear models for three and four factor partly nested designs are:

14.2.1 One between (α), one within (γ) block effect

$$y_{ijkl} = \mu + \alpha_i + \beta_j + \gamma_k + \alpha\gamma_{ij} + \beta\gamma_{jk} + \varepsilon_{ijkl}$$

14.2.2 Two between (α, γ), one within (δ) block effect

$$y_{ijklm} = \mu + \alpha_i + \gamma_j + \alpha\gamma_{ij} + \beta_k + \delta_l + \alpha\delta_{il} + \gamma\delta_{jl} + \alpha\gamma\delta_{ijl} + \varepsilon_{ijklm} \qquad \text{(Model 2 - Additive)}$$

$$y_{ijklm} = \mu + \alpha_i + \gamma_j + \alpha\gamma_{ij} + \beta_k + \delta_l + \alpha\delta_{il} + \gamma\delta_{jl} + \alpha\gamma\delta_{ijl} + \beta\delta_{kl} + \beta\alpha\delta_{kil} + \beta\gamma\delta_{kjl} + \beta\alpha\gamma\delta_{kijl} + \varepsilon_{ijklm} \qquad \text{(Model 1 - Non-additive)}$$

14.2.3 One between (α), two within (γ, δ) block effects

$$y_{ijklm} = \mu + \alpha_i + \beta_j + \gamma_k + \delta_l + \gamma\delta_{kl} + \alpha\gamma_{ik} + \alpha\delta_{il} + \alpha\gamma\delta_{ikl} + \varepsilon_{ijk} \qquad \text{(Model 2- Additive)}$$

$$y_{ijklm} = \mu + \alpha_i + \beta_j + \gamma_k + \beta\gamma_{jk} + \delta_l + \beta\delta_{jl} + \gamma\delta_{kl} + \beta\gamma\delta_{jkl} + \alpha\gamma_{ik} + \alpha\delta_{il} + \alpha\gamma\delta_{ikl} + \varepsilon_{ijk} \qquad \text{(Model 1 - Non-additive)}$$

where μ is the overall mean, β is the effect of the Blocking Factor B and ε is the random unexplained or residual component.

14.3 Analysis of variance

The construction of appropriate F-ratios generally follow the rules and conventions established in Chapters 10-13, albeit with additional complexity. Tables 14.1-14.3 document the appropriate numerator and denominator mean squares and degrees of freedom for each null hypothesis for a range of two and three factor partly nested designs.

14.4 Assumptions

As partly nested designs share elements in common with each of nested, factorial and unreplicated factorial designs, they also share similar assumptions and implications to these other designs. Readers should also consult sections 11.5, 12.4 and 14.4. Specifically, hypothesis tests assume that:

(i) the appropriate residuals are normally distributed. Boxplots using the appropriate scale of replication (reflecting the appropriate residuals/F-ratio denominator (see Tables 14.1-14.3) should be used to explore normality. Scale transformations are often useful.
(ii) the appropriate residuals are equally varied. Boxplots and plots of means against variance (using the appropriate scale of replication) should be used to explore the spread of values. Residual plots should reveal no patterns (see Figure 8.5). Scale transformations are often useful.
(iii) the appropriate residuals are independent of one another. Critically, experimental units within blocks/subjects should be adequately spaced temporally and spatially to restrict contamination or carryover effects.
(iv) that the variance/covariance matrix displays **sphericity**[c] (see section 13.4.1). This assumption is likely to be met only if the treatment levels within each block can be randomly ordered. This assumption can be managed by either adjusting the sensitivity of the affected F-ratios or employing linear mixed effects (see section 11.8) modelling to the design.

[c] Strickly, the variance-covariance matrix must display a very specific pattern of sphericity in which both variances and covariances are equal (compound symmetry), however an F-ratio will still reliably follow an F distribution provided basic sphericity holds.

Table 14.1 *F*-ratios and corresponding R syntax for partly additive nested designs with one between block and one within block effect. *F*-ratio numerators and demoninators are represented by numbers that correspond to the rows from which the appropriate mean square values would be associated. *F*-ratios denoted '?' indicate inexact denominators.

			F-ratio						
			A&C fixed, B random		A fixed, B&C random		C fixed, A&B random		A,B&C random
	Factor	**d.f.**	**Restricted**	**Unrestricted**	**Restricted**	**Unrestricted**	**Restricted**	**Unrestricted**	
1	A	$a-1$	1/2	1/2	$1/(2+4-5)^a$	$1/(2+4-5)$	1/2	?	$1/(2+4-5)^b$
2	B′(A)	$(b-1)a$	No test	2/5	2/5	2/5	No test	2/5	2/5
3	C	$(c-1)$	3/5	3/5	3/5	3/4	$3/4^a$	3/4	$3/4^b$
4	A×C	$(c-1)(a-1)$	4/5	4/5	4/5	4/5	4/5	4/5	4/5
5	Residuals (=C×B′(A))	$(c-1)(b-1)a$	No test	No test	No test	No test	No test		

R syntax [c]

A&C fixed, B random

```
> summary(aov(DV~A*C+Error(B)))
```

Unbalanced

```
#sphericity met
> anova(lme(DV~A*C, random=~1|B, correlation=corCompSymm(form=~1|B))
#sphericity not met
> anova(lme(DV~A*C, random=~1|B, correlation=corAR1(form=~1|B))
> anova(lme(DV~A*C, random=~1|B, correlation=...)
```

[a] Pooling: higher order interactions with $P > 0.25$ can be removed to produce more exact denominators.
[b] Inexact *F*-ratio for restricted model.
[c] Mixed models with non-hierarchical random factors require manual *F*-ratio and *P*-value calculations.

Table 14.2 *F*-ratios and corresponding R syntax for partly additive nested designs with one between block and two within block effect. *F*-ratio numerators and demoninators are represented by numbers that correspond to the rows from which the appropriate mean square values would be associated.

			F-ratio						
			A,C&D fixed, B random		**A&D fixed**	**A fixed**	**C&D fixed**	**D fixed**	**A,B,C&D**
	Factor	**d.f.**	**Model 1**	**Model 2**	**B&C random**	**B,C&D random**	**A&B random**	**A,B&C random**	**random**
1	A	$a-1$	1/2	1/2	$1/(2+4-5)^a$	$1/(2+4-5+7-8-10+11)$	$1/2^b$	$1/(2+4-5)^a$	$1/(2+4-5+7-8-10+11)$
2	B′(A)	$(b-1)a$	No test[b c]	No test	$2/(5)^b$	$2/(5+8-11)$	No test[b]	$2/(5)^b$	$2/(5+8-11)^a$
3	C	$(c-1)$	3/5	3/11	$3/5^b$	$3/(5+9-11)$	3/4	$3/4^b$	$3/(4+9-10)^a$
4	C×A	$(c-1)(a-1)$	4/5	4/11	$4/5^b$	$4/(5+10-11)$	$4/5^b$	$4/5^b$	$4/(4+10-11)^a$
5	C×B′(A)	$(c-1)(b-1)a$	No test[b]		No test	5/11	No test	No test	5/11
6	D	$(d-1)$	6/8	6/11	$6/(8+9-11)^a$	$6/(8+9-11)$	6/7	$6/(7+9-10)^b$	$6/(7+9-10)^a$
7	D × A	$(d-1)(a-1)$	7/8	7/11	$7/(8+10-11)^a$	$7/(8+10-11)$	$7/8^b$	$7/(8+10-11)^b$	$7/(8+10-11)^a$
8	D ×B′(A)	$(d-1)(b-1)a$	No test[b]		$8/11^a$	8/11	No test	8/11	8/11
9	D×C	$(d-1)(c-1)$	9/11	9/11	$9/11^b$	9/11	9/10	9/10	$9/10^a$
10	D × C × A	$(d-1)(c-1)(a-1)$	10/11	10/11	10/11	10/11	10/10	10/10	10/10
11	Residuals (=D×C×B′(A))	$(d-1)(c-1)(b-1)a$	No test		No test	No test	No test	No test	No test

R syntax

A,C&D fixed, B random

Model 1
```
> summary(aov(DV~A*C*D+Error(B/(C*D))))
```
Unbalanced
```
#sphericity met
> anova(lmer(DV~A*C*D*B+(1|B))) #note, only MS produced
#sphericity not met
> anova(lmer(DV~-1+A*C*D*B+(-1+A|B)+(-1+C|B)+(-1+A:C|B))) #note, only MS produced
```

(*continued overleaf*)

Table 14.2 (*continued*)

<table>
<tr><th rowspan="3">Factor</th><th rowspan="3">d.f.</th><th></th><th colspan="7">F-ratio</th></tr>
<tr><th></th><th colspan="2">A,C&D fixed, B random</th><th rowspan="2">A&D fixed
B&C random</th><th rowspan="2">A fixed
B,C&D random</th><th rowspan="2">C&D fixed
A&B random</th><th rowspan="2">D fixed
A,B&C random</th><th rowspan="2">A,B,C&D
random</th></tr>
<tr><th></th><th>Model 1</th><th>Model 2</th></tr>
<tr><td></td><td></td><td>Model 2</td><td colspan="7"><code>> summary(aov(DV~A*C*D+Error(B)))</code></td></tr>
<tr><td></td><td></td><td>Unbalanced</td><td colspan="7"><code>#sphericity met</code></td></tr>
<tr><td></td><td></td><td></td><td colspan="7"><code>> anova(lme(DV~A*C*D, random=~1|B))</code></td></tr>
<tr><td></td><td></td><td></td><td colspan="7"><code>#sphericity not met</code></td></tr>
<tr><td></td><td></td><td></td><td colspan="7"><code>> anova(lme(DV~A*C*D, random=~1|B, correlation=corAR1(form~1|B))</code></td></tr>
<tr><td></td><td></td><td></td><td colspan="7">Other models</td></tr>
<tr><td></td><td></td><td>Balanced</td><td colspan="7"><code>> AnovaM(aov(DV~A*B*C*D)) #F-ratios and P-values must be calculated individually</code></td></tr>
<tr><td></td><td></td><td>Unbalanced</td><td colspan="7"><code>#sphericity met</code></td></tr>
<tr><td></td><td></td><td></td><td colspan="7"><code>> anova(lme(DV~A*C*D, random=~1|B, , correlation=corCompSymm(form~1|B)))</code></td></tr>
<tr><td></td><td></td><td></td><td colspan="7"><code>#sphericity not met</code></td></tr>
<tr><td></td><td></td><td></td><td colspan="7"><code>> anova(lme(DV~A*C*D, random=~1|B, correlation=corAR1(form~1|B)))</code></td></tr>
</table>

[a] Pooling: higher order interactions with $P > 0.25$ can be removed to produce more exact denominators.
[b] Inexact *F*-ratio for restricted model.
[c] Inexact *F*-ratio for restricted model.

Table 14.3 *F*-ratios and corresponding R syntax for partly additive nested designs with two between block and one within block effect. *F*-ratio numerators and demoninators are represented by numbers that correspond to the rows from which the appropriate mean square values would be associated.

			F-ratio					
	Factor	d.f.	A,C&D fixed B random	A&C fixed B&D random	A&D fixed B&C random	A fixed C,B&D random	D fixed A,C&B random	A,B,C&D random
1	A	$a-1$	1/4	$1/(4+6-9)^a$	$1/3^a$	$1/(3+6-8)^a$	$1/3^{a\ b}$	$1/(3+6-8)^a$
2	C	$c-1$	2/4	$2/(4+7-9)^a$	$2/4^b$	$2/(4+7-9)^b$	$2/3^a$	$2/(3+7-8)^{a\ b}$
3	C×A	$(c-1)(a-1)$	3/4	$3/(4+8-9)^a$	$3/4^b$	$3/(4+8-9)^a$	$3/4^b$	$3/(3+8-9)^a$
4	B′(C×A)	$(b-1)ca$	No test	4/9	No test	4/9	No test	4/9
5	D	$(d-1)$	5/9	$5/9^b$	$5/7^a$	$5/7^{ab}$	$5/(6+7-8)^a$	$5/(6+7-8)$
6	D×A	$(d-1)(a-1)$	6/9	6/9	$6/8^a$	$6/8^a$	$6/8^a$	$6/8^a$
7	D×C	$(d-1)(c-1)$	7/9	7/9	7/9	7/9	$7/8^a$	$7/8^a$
8	D×C×A	$(d-1)(c-1)(a-1)$	8/9	8/9	8/9	8/9	8/9	8/0
11	Residuals (=D×B′(C×A))	$(d-1)(b-1)ca$	No test	No test	No test	No test	No test	

R syntax[c]

A,C&D fixed, B random

Balanced
```
> summary(aov(DV~A*C*D+Error(B/(C*D))))
```
Unbalanced
```
#sphericity met
> anova(lme(DV~A*C*D*B+(1|B)))
#sphericity not met
> anova(lme(DV~A*C*D*B+(1|B)), type='marginal')
```

Other models
```
> AnovaM(aov(DV~(A*C)/B+(D*C*A)/B)) #F-ratios and P-values must be calculated individually
```

[a]Pooling: higher order interactions with P > 0.25 can be removed to produce more exact denominators.
[b]Inexact *F*-ratio for restricted model.
[c]Mixed models with non-hierarchical random factors require manual *F*-ratio and *P*-value calculations.

(v) there are no block by within block interactions. Such interactions render non-significant within block effects difficult to interpret[d].

14.5 Other issues

Issues of *post hoc* and specific comparisons as well design balance and power follow the discussions in sections 10.6, 13.5, 13.6, 11.7, 11.10 and 13.8.

14.5.1 Robust alternatives

As designs increase in complexity, so too do the options for robust alternatives. In particular, rank based procedures can yield highly misleading outcomes. Generalized linear models (GLM: chpt 17) can be useful for modelling alternative (non-normal) residual distributions provided pairs of full and reduced models are chosen carefully and sensibly. Finally, randomizations can also be of use (particularly when observational independence is violated). However, care must be exercised in determining the appropriate scale at which to randomize.

Partly nested designs consist of multiple error or residual terms arranged in hierarchical strata and can therefore be thought of as a series of linear models (one for each strata). For example, a repeated measures design might consist of a linear model representing the between subject effects and one or more linear models representing the within subject effects. As a result, partly nested designs can also be broken down into the individual linear models onto which more the simplified robust alternatives highlighted in previous chapters can be applied.

14.6 Further reading

- Theory

Doncaster, C. P., and A. J. H. Davey. (2007). *Analysis of Variance and Covariance. How to Choose and Construct Models for the Life Sciences*. Cambridge University Press, Cambridge.

Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

[d] Unless we assume that there are no block by within block interactions, non-significant within block effects could be due to either an absence of a treatment effect, or as a result of opposing effects within different blocks. As these block by within block interactions are unreplicated, they can neither be formally tested nor is it possible to perform main effects tests to diagnose non-significant within block effects.

Faraway, J. J. (2006). *Extending Linear Models with R: generalized linear mixed effects and nonparametric regression models*. Chapman & Hall/CRC.

Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

Pinheiro, J. C., and D. M. Bates. (2000). *Mixed effects models in S and S-PLUS*. Springer-Verlag, New York.

Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS, 4th edn*. Springer-Verlag, New York.

Zuur, A. F., E. N. Ieno, N. J. Walker, A. A. Saveliev, and G. M. Smith. (2009). *Mixed Effects Models and Extensions in Ecology with R*. Springer.

14.7 Key for partly nested ANOVA

1 **Determine the appropriate model design and hierarchy**

- **Conceptualise the design into a hierarchy (ladder) of factors**
  - Between block effects (factors whose levels differ between different blocks)
  - Between block interactions (if there are multiple between block effects)
  - Blocking factor (typically a random factor in which each level of other factors are applied)
  - Within block effects (factors that have all levels applied within each block).
  - Between block by within block interactions
  - Block by within block interactions
  - Within block interactions (if there are multiple within block effects)
- Identify the correct error (residual) term and thus F-ratio denominator for each factor (see Tables 14.1-14.3)

. . . . Go to 2

2 a. **Check assumptions for split-plot and complex randomized block ANOVA** . . . . see Examples 14A–14C&14E

As the assumptions of any given hypothesis test relate to residuals, all diagnostics should reflect the appropriate error (residual) terms for the hypothesis. This is particularly important for Model 1 (non-additive) models where interaction terms are used as the appropriate denominators (residuals).

- **No block by within block treatment interactions**

```
> with(data, interaction.plot(B, C, DV))
> library(lattice)
> bwplot(DV ~ C, groups = BLOCK, data)
```

Residual curvature plot and Tukey's test for nonadditivity

```
> library(alr3)
> residual.plots(lm(DV ~ B + C, data))
> tukey.nonadd.test(lm(DV ~ B + C, data))
```

- **Normality (symmetry) of the response variable (residuals) at each level of each factor or combination of factors - boxplots of mean values**
  Between block factor using MS_B as denominator in each case

```
> library(nlme)
> data.B.agg <- gsummary(data, groups = data$B)
> boxplot(DV ~ A, data.B.agg)
```

  Single within block factor or additive model (no interactions - Model 2) using MS_{Resid} as denominator in each case

```
> boxplot(DV ~ A, data) #factor A
> boxplot(DV ~ C, data) #factor C
> boxplot(DV ~ A * C, data) #A:C interaction
```

  Two or more within block factor non-additive (Model 1) model using interactions (such as MS_{BC}) as denominator as example

```
> library(nlme)
> data.BC.agg <- gsummary(data, groups = data$B : data$C)
> boxplot(DV ~ C, data.BC.agg) #factor C
```

  where `DV` *is the response variable*, `A` *is a main fixed or random factor within the* `data` *dataset.*

- **Homogeneity (equality) of variance of the response variable (residuals) at each level of each factor or combination of factors - boxplots of mean values**
  As for Normality.

Parametric assumptions (Normality/Homogeneity of variance) met ...... Go to 4
b. Parametric assumptions not met .... Go to 3
3 a. Attempt a scale transformation (see Table 3.2 for transformation options) Go to 2
b. Transformations unsuccessful or inappropriate .... Go to Key 13.9
4 a. If incorporating planned contrasts (comparisons) .... See Examples 14C,14D

```
> contrasts(data$A) <- cbind(c(contrasts), ...)
> round(crossprod(contrasts(data$A)), 2)
```

.... Go to 5
5 a. Determine whether the design is balanced

```
> replications(DV ~ Error(B) + A * C.., data)
> library(biology)
> is.balanced(DV ~ Error(B) + A * C.., data)
```

Design is balanced - sample sizes of all cells are equal .... Go to 6
b. Design is NOT balanced - one or more cells (combinations) missing (0 replicates) .... Go to 7
c. Design is NOT balanced - sample sizes of cells differ, but all combinations have at least one replicate .... Go to 8b
6 a. Balanced single between and single within block factor or additive (no interactions - Model 2) .... See Examples 14A,14B

```
> #Single within block factor
> data.aov <- aov(DV ~ A * C + Error(B), data)
> #Multiple within/between block factors
> data.aov <- aov(DV ~ A * C * D + Error(B), data)
```

Alternatively, consider linear mixed effects (lme) model . . . . See Key 13.13
Check for sphericity . . . . See Key 13.12

- **Sphericity met**

```
> summary(data.aov)
> library(biology)
> AnovaM(data.aov)
```

- **Sphericity NOT met**

```
> library(biology)
> AnovaM(data.aov, RM = T)
```

To incorporate planned comparisons, utilize the `split=` argument, see Key 12.8
For post-hoc multiple comparisons . . . . Go to 12.20a
If significant interation . . . . Go to 12.14

b. Balanced two or more within block factor non-additive (Model 1)

```
> data.aov <- aov(DV ~ A + Error(B/C + B/D) + C * D, data)
```

Alternatively, consider linear mixed effects (lme) model . . . . See Key 13.13
Check for sphericity . . . . Go to Key 13.12

- **Sphericity met**

```
> summary(data.aov)
> library(biology)
> AnovaM(data.aov)
```

- **Sphericity NOT met**

```
> library(biology)
> AnovaM(data.aov, RM = T)
```

To incorporate planned comparisons, utilize the `split=` argument, see Key 12.8
For post-hoc multiple comparisons . . . . Go to Key 12.20a
If significant interation . . . . Go to Key 12.14

7 a. Unbalanced (missing cells) single within block or additive (Model 2)

- No within block correlation structure

```
> #single within block factor
> data.lme <- lme(DV ~ A, random = ~1 | Block, data)
> #multiple within block factors
> data.lme <- lme(Y ~ A * C, random = ~1 | Block, data)
```

- Compound symmetry within block correlation structure

```
> #single within block factor
> data.lme <- lme(DV ~ A * C, random = ~1 | B, data,
+ correlation = corCompSymm(form = ~1 | B))
> #multiple within block factor
> data.lme <- lme(DV ~ A * C * D, random = ~1 | B, data,
+ correlation = corCompSymm(form = ~1 | B))
```

- General (unstructured) within block correlation structure

```
> #single within block factor
> data.lme <- lme(DV ~ A * C, random = ~1 | B, data,
```

```
+ correlation = corSymm(form = ~1 | B))
> #multiple within block factor
> data.lme <- lme(DV ~ A * C * D, random = ~1 | B, data,
+ correlation = corSymm(form = ~1 | B))
```

- First order autoregressive within block correlation structure

```
> #single within block factor
> data.lme <- lme(DV ~ A * C, random = ~1 | B, data,
+ correlation = corAR1(form = ~1 | B))
> #multiple within block factor
> data.lme <- lme(DV ~ A * C * D, random = ~1 | B, data,
+ correlation = corAR1(form = ~1 | B))
```

Comparing two models with differing correlation structures

```
> anova(data.lme, data.lme1)
> anova(data.lme)
```

b. Unbalanced (missing cells) two or more within block factor non-additive (Model 1)

```
> data.lme <- lme(Y ~ A * C, random = ~1 | Block/A + 1 |
+      Block/C, data)
> anova(data.lme)
```

8 a. Unbalanced (unequal sample sizes n > 0) additive (Model 2) . . . . See Examples 14C,14D

```
> data.aov <- aov(DV ~ A * C + Error(B), data)
> AnovaM(data.aov, type = "II")
```

OR

```
> contrasts(data$A) <- contr.helmert
> contrasts(data$C) <- contr.helmert
> data.aov <- aov(DV ~ A * C + Error(B), data)
> AnovaM(data.aov, type = "III")
```

OR

```
> data.lme <- lme(DV ~ A * C, random = ~1 | Block, data)
> summary(data.lme, type = "marginal")
```

b. Unbalanced (unequal sample sizes n > 0) non-additive (Model 1)

```
> data.aov <- aov(DV ~ A * C * D + Error(Block/C + Block/D),
      data)
> AnovaM(data.aov, type = "II")
```

OR

```
> contrasts(data$A) <- contr.helmert
> contrasts(data$C) <- contr.helmert
> contrasts(data$D) <- contr.helmert
> data.aov <- aov(DV ~ A * C * D + Error(Block/C + Block/D),
      data)
> AnovaM(data.aov, type = "III")
```

OR

```
> data.lme <- lme(Y ~ A * C, random = ~1 | Block, data)
> anova(data.lme, type = "marginal")
```

14.8 Worked examples of real biological data sets

Example 14A: Split-plot ANOVA

Kirk (1968) fabricated an experiment in which the effects of mode of signal presentation (Treatment A: 1 = auditory signal, 2 = visual signal), monitoring period throughout experiment (Treatment C[e]: 1 = one hour, 2 = two hours, 3 = three hours and 4 = four hours) on the degree of vigilance displayed (Y: measured as response latency) by a number of subjects was measured. Four of the subjects were randomly assigned to the auditory signal treatment and the another four subjects to the visual signal treatment and the response latency of each subject were repeated every hour for four hours. These data can be analysed as a split-plot or repeated measures design with subjects as the plots, signal type (Treatment A) as the between plot effect and monitoring period (Treatment C) as the within plot effect (from chapter 8 of Kirk (1968)).

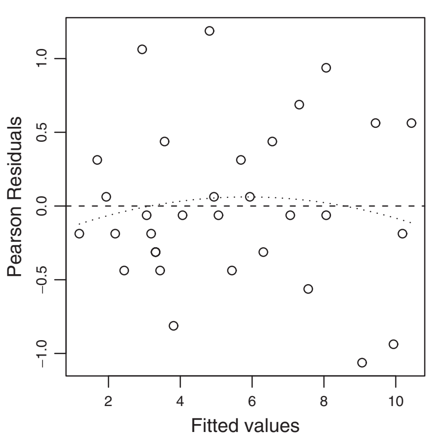
Step 1 - Import (section 2.3) the Kirk (1968) spf (split-plot factorial) data set.

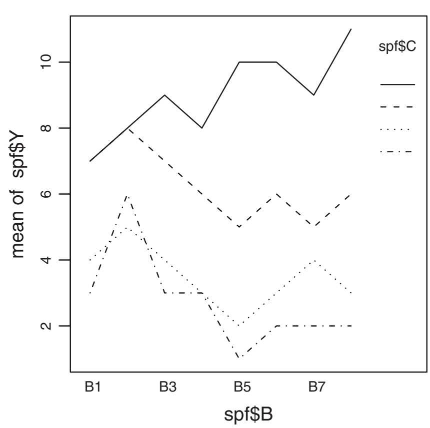
```
> spf <- read.table("spf.csv", header = T, sep = ",")
```

Step 2 (Key 14.2) - Assess whether there are likely to be any plot by within plot interactions.

```
> library(alr3)
> resplot(lm(Y ~ A * C + B,
        data = spf))
   t value   Pr(>|t|)
-1.2186824  0.2229648
```

```
> with(spf, interaction.plot
      (spf$B, spf$C, spf$Y))
```





[e] Note, to maintain consistency with the conventions adopted by Quinn and Keough (2002) as well as this book, I have altered Kirk (1968)'s Factor B into Factor C and subjects S into B.

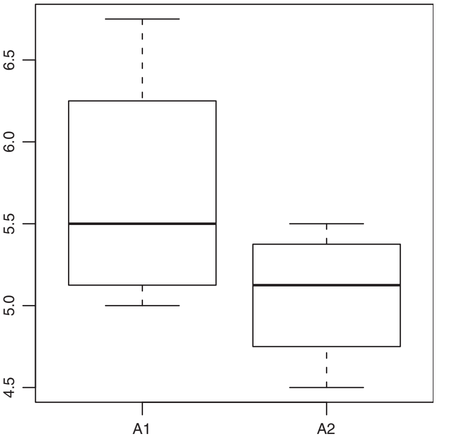
Conclusions - No strong evidence of a blocking interaction, therefore an additive model is appropriate.

Step 3 (Key 14.2) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators).

1. Factor *A* (signal type treatment - fixed effect). The subject means are the replicates for the signal treatment effect and thus an aggregated dataset needs to be created from which the boxplots can be based.

   ```
   > library(nlme)
   > spf.agg <- gsummary(spf,
         groups = spf$B)
   > boxplot(Y ~ A, spf.agg)
   ```

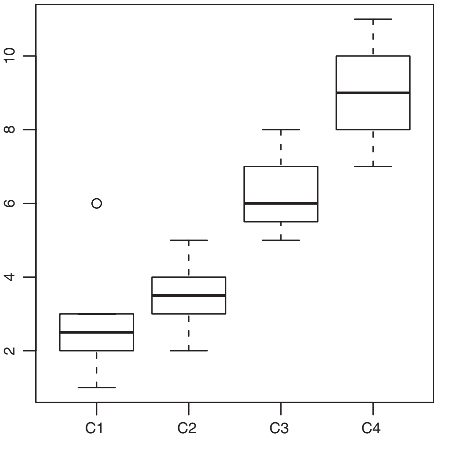
   Conclusions - There is no conclusive evidence of non-normality or unequal variance.



2. Factor C (monitoring period - fixed factor). The individual vigilance measurements within each subject are the replicates for the effect of monitoring period.

   ```
   > boxplot(Y ~ C, spf)
   ```

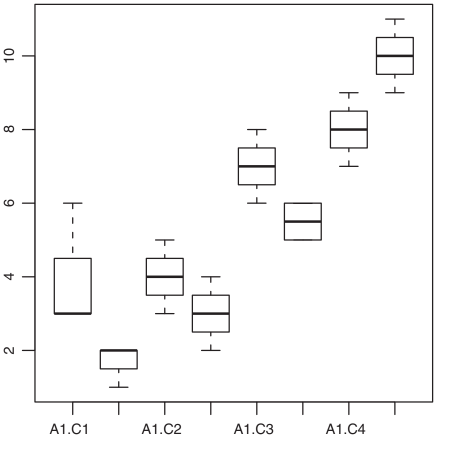
   Conclusions - There is no conclusive evidence of non-normality or unequal variance.



3. *A*:C interaction (fixed factor). The individual vigilance measurements within each subject are the replicates for the interaction effect.

   ```
   > boxplot(Y ~ A * C, spf)
   ```

   Conclusions - There is no conclusive evidence of non-normality or unequal variance.



Step 4 (Key 14.5) - Determine whether or not the design is balanced (at least with respect to sub-replication).

```
> replications(Y ~ A * C + Error(B), data = spf)
  A   C A:C
 16   8   4
> library(biology)
> is.balanced(Y ~ A * C + Error(B), data = spf)
[1] TRUE
```

Conclusions - The design is completely balanced. There are exactly one of each of the four monitoring periods per subject within each signal type.

Step 5 (Key 14.6) - fit the linear model and produce an ANOVA table to test the null hypotheses that there no effects of signal type, monitoring time or interaction on vigilance (Table 8.2-2 of Kirk (1968)).

```
> spf.aov <- aov(Y ~ A * C + Error(B), spf)
> summary(spf.aov)
Error: B
          Df Sum Sq Mean Sq F value Pr(>F)
A          1 3.1250  3.1250       2 0.2070
Residuals  6 9.3750  1.5625

Error: Within
          Df  Sum Sq Mean Sq F value    Pr(>F)
C          3 194.500  64.833 127.890 2.516e-12 ***
A:C        3  19.375   6.458  12.740 0.0001051 ***
Residuals 18   9.125   0.507
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There is a significant signal type by monitoring period interaction ($F_{3,18} = 12.740, P < 0.001$). Note that whilst sphericity might be expected to be an issue for these data (since the order of the monitoring periods could not be randomized), Kirk (1968) concluded that the variance-covariance matrix did not deviate substantially from symmetry and thus considered corrections unnecessary.

Step 6 - Explore the nature of the interaction further by evaluating the simple main effects (Table 8.6-2 of Kirk (1968)).

- Effect of monitoring period (C) at A1 (auditory signal). Note, to reduce inflated family-wise Type I errors, Kirk (1968) advocated testing each of the simple main effects tests at α/p where p is the number of simple main effects tests within a global linear model term such that the main effects family-wise α is the same as the α used to assess the global hypothesis in the original model. In this example, we will perform four (4) simple main effects tests, and thus $\alpha = 0.05/4 = 0.0125$ for each.

```
> library(biology)
> summary(mainEffects(spf.aov, at = A == "A1"))
Error: B
```

```
          Df Sum Sq Mean Sq F value Pr(>F)
INT        1 3.1250  3.1250       2 0.2070
Residuals  6 9.3750  1.5625

Error: Within
          Df  Sum Sq Mean Sq F value    Pr(>F)
INT        3 159.187  53.062 104.671 1.391e-11 ***
C          3  54.688  18.229  35.959 8.223e-08 ***
Residuals 18   9.125   0.507
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Effect of monitoring period (C) at *A*2 (visual signal)

```
> library(biology)
> summary(mainEffects(spf.aov, at = A == "A2"))
Error: B
          Df Sum Sq Mean Sq F value Pr(>F)
INT        1 3.1250  3.1250       2 0.2070
Residuals  6 9.3750  1.5625

Error: Within
          Df  Sum Sq Mean Sq F value    Pr(>F)
INT        3  54.688  18.229  35.959 8.223e-08 ***
C          3 159.188  53.062 104.671 1.391e-11 ***
Residuals 18   9.125   0.507
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Effect of signal type (*A*) at C1 (first hour). Note, when a main effect and an interaction to which it contributes do not have the same error term, main effects should be calculated using a pooled error term so fitting a fully factorial anova will pool error terms

```
> spf.aovA <- aov(Y ~ A * C, spf)
> summary(mainEffects(spf.aovA, at = C == "C1"))
            Df  Sum Sq Mean Sq F value   Pr(>F)
INT          6 209.000  34.833  45.189 6.51e-12 ***
A            1   8.000   8.000  10.378 0.003645 **
Residuals   24  18.500   0.771
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Effect of signal type (*A*) at C2 (second hour).

```
> summary(mainEffects(spf.aovA, at = C == "C2"))
            Df  Sum Sq Mean Sq F value    Pr(>F)
INT          6 215.000  35.833 46.4865 4.783e-12 ***
```

```
  A             1   2.000   2.000  2.5946    0.1203
  Residuals   24  18.500   0.771
  ---
  Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
  ...
```

Conclusions - Whilst the visual signal was more effective (lower response latency) than the auditory signal during the first hour of the experiment ($P < 0.001$), its superiority was not significant during hour two ($P > 0.0125$) and three ($P > 0.0125$) and it was significantly less effective than the auditory signal during the forth hour ($P = 0.004$).

Step 7 - Since factor C (monitoring period) represents a quantitative sequence of the duration of the experiment, we might also be interested in exploring the nature of trends (linear, quadratic, etc) in vigilance over time and whether these trends are consistent for both signal types. Trends should be compared using a separately calculated error term, each of which estimates a different source of variation. Furthermore, family-wise α values should be maintained by dividing the α by three (one for each polynomial trend $\alpha/3 = 0.017$), (Table 8.8-4 of Kirk (1968)).

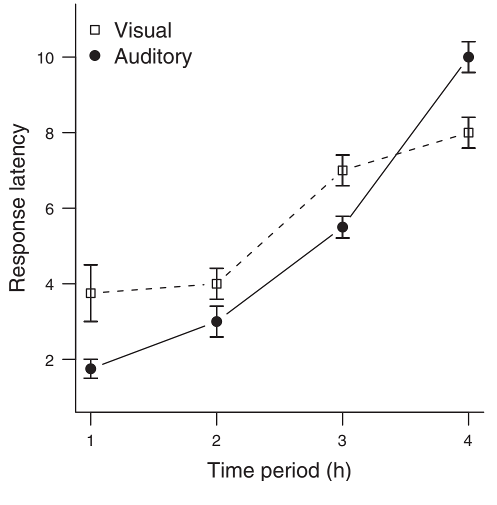
```
> p1 <- C(spf$C, poly, 1)
> p2 <- C(spf$C, poly, 2)
> p3 <- C(spf$C, poly, 3)
> spf.aov <- aov(Y ~ A * (p1 + p2 + p3) + Error(B/(p1 + p2 + p3)),
+     spf)
> summary(spf.aov)
Error: B
          Df Sum Sq Mean Sq F value Pr(>F)
A          1 3.1250  3.1250       2 0.2070
Residuals  6 9.3750  1.5625

Error: B:p1
          Df  Sum Sq Mean Sq F value    Pr(>F)
p1         1 184.900 184.900 182.617 1.018e-05 ***
A:p1       1  13.225  13.225  13.062   0.01118 *
Residuals  6   6.075   1.012
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: B:p2
          Df Sum Sq Mean Sq F value   Pr(>F)
p2         1 8.0000  8.0000    25.6 0.002311 **
A:p2       1 3.1250  3.1250    10.0 0.019509 *
Residuals  6 1.8750  0.3125
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: B:p3
          Df  Sum Sq Mean Sq F value  Pr(>F)
```

```
p3          1 1.60000 1.60000  8.1702 0.02886 *
A:p3        1 3.02500 3.02500 15.4468 0.00771 **
Residuals   6 1.17500 0.19583
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - At $\alpha = 0.017$, it is evident that a substantial component of the global A:C interaction (13.225/19.375 = 63.5%) is due to differences in the nature of the linear decline in vigilance through time between the two signal types ($P = 0.011$).

Step 8 (Key 12.18) - Summarize the trends in a plot.

```
> spf.means <- with(spf, tapply(Y, list(A, C), mean))
> library(gmodels)
> spf.se <- with(spf, tapply(Y, list(A, C), function(x) ci(x)[4]))
> plot(Y ~ as.numeric(C), data = spf, type = "n", axes = F,
+     xlab = "", ylab = "")
> xval <- as.numeric(spf$C)
> points(spf.means["A1", ], pch = 22, type = "b", lwd = 1, lty = 2)
> arrows(xval, spf.means["A1", ] - spf.se["A1", ], xval,
+     spf.means["A1", ] + spf.se["A1", ], code = 3, angle = 90,
+     len = 0.05)
> points(spf.means["A2", ], pch = 19, type = "b", lwd = 1, lty = 1)
> arrows(xval, spf.means["A2", ] - spf.se["A2", ], xval,
+     spf.means["A2", ] + spf.se["A2", ], code = 3, angle = 90,
+     len = 0.05)
> axis(1, at = 1:4, cex.axis = 0.8)
> mtext(text = "Time period (h)", side = 1, line = 3)
> axis(2, cex.axis = 0.8, las = 1)
> mtext(text = "Response latency", side = 2, line = 3)
> legend("topleft", leg = c("Visual", "Auditory"), lty = 0,
+     pch = c(22, 19), bty = "n", cex = 1)
> box(bty = "l")
```



Example 14B: Linear mixed effects split-plot

Alternatively, linear mixed effects modeling could be used to analyze the Kirk (1968) spf (split-plot factorial) data used in Example 14A. Notably, such an approach permits us to attempt to incorporate the nature of the variance-covariance matrix rather than wish it away post-hoc with estimated adjustments.

Step 1 (14.2&14.5) - Refer to Example 14A for importing the data and performing exploratory data analysis.

Step 2 (Key 14.6) - Fit a series of lme models (with and without random slope components as well as alternative correlation structures) and compare them to evaluate the appropriateness of each.

```
> library(nlme)
> spf.lme.1 <- lme(Y ~ A * C, random = ~C | B, spf)
> spf.lme.2 <- update(spf.lme.1, random = ~1 | B)
> anova(spf.lme.1, spf.lme.2)
          Model df      AIC      BIC    logLik   Test  L.Ratio p-value
spf.lme.1     1 19 97.63924 120.0223 -29.81962
spf.lme.2     2 10 89.64876 101.4293 -34.82438 1 vs 2 10.00952  0.3497
```

Conclusions - Random slope not required as the model incorporating the random slope is not a significantly better fit (likelihood ratio not significant).

```
> spf.lme.3 <- update(spf.lme.2, correlation = corAR1(form = ~1 |
+     B))
> anova(spf.lme.2, spf.lme.3)
          Model df      AIC      BIC    logLik   Test  L.Ratio p-value
spf.lme.2     1 10 89.64876 101.4293 -34.82438
spf.lme.3     2 11 88.44767 101.4063 -33.22384 1 vs 2 3.201085  0.0736
> spf.lme.4 <- update(spf.lme.2, correlation = corCompSymm(form = ~1 |
+     B))
> anova(spf.lme.2, spf.lme.4)
          Model df      AIC      BIC    logLik   Test      L.Ratio p-value
spf.lme.2     1 10 89.64876 101.4293 -34.82438
spf.lme.4     2 11 91.64876 104.6073 -34.82438 1 vs 2 1.421086e-14       1
> spf.lme.5 <- update(spf.lme.2, correlation = corSymm(form = ~1 |
+     B))
> anova(spf.lme.2, spf.lme.5)
          Model df      AIC      BIC    logLik   Test  L.Ratio p-value
spf.lme.2     1 10 89.64876 101.4293 -34.82438
spf.lme.5     2 16 94.52970 113.3786 -31.26485 1 vs 2 7.119057    0.31
```

Conclusions - neither first order continuous-time autoregressive, compound symmetry or a general correlation structure yield better fits than a no within-group correlation structure. Examine the fit of the linear mixed effects model.

```
> anova(spf.lme.2)
            numDF denDF  F-value p-value
(Intercept)     1    18 591.6800  <.0001
A               1     6   2.0000  0.2070
C               3    18 127.8904  <.0001
A:C             3    18  12.7397  0.0001
```

Conclusions - There is a significant signal type by monitoring period interaction ($F_{3,18} = 12.740$, $P < 0.001$).

Step 3 - Investigate the simple main effects.

- Effect of monitoring period (C) at A1.

```
> library(biology)
> anova(mainEffects(spf.lme.2, at = A == "A1"))
             numDF denDF  F-value p-value
(Intercept)      1    17 591.6800  <.0001
M1               4    17  79.0034  <.0001
M3               3    17  35.9589  <.0001
```

- Effect of monitoring period (C) at A2.

```
> anova(mainEffects(spf.lme.2, at = A == "A2"))
             numDF denDF  F-value p-value
(Intercept)      1    17 591.6800  <.0001
M1               4    17  27.4692  <.0001
M3               3    17 104.6712  <.0001
```

- Effect of monitoring period (A) at C1.

```
> anova(mainEffects(spf.lme.2, at = C == "C1"))
             numDF denDF  F-value p-value
(Intercept)      1    18 591.6800  <.0001
M1               6    18  68.9187  <.0001
M2               1     6  10.3784  0.0181
```

- Effect of monitoring period (A) at C2.

```
> anova(mainEffects(spf.lme.2, at = C == "C2"))
             numDF denDF  F-value p-value
(Intercept)      1    18 591.6800  <.0001
M1               6    18  70.2160  <.0001
M2               1     6   2.5946  0.1584
```

Step 4 - Similar with `lmer` (`lme4`).

```
> library(lme4)
> spf.lmer <- lmer(Y ~ A * C + (1 | B), spf)
> library(languageR)
> aovlmer.fnc(spf.lmer, noMCMC = T)
Analysis of Variance Table
    Df  Sum Sq Mean Sq  F value        F     Df2         p
A    1   1.014   1.014   1.9997   1.9997  24.000     0.170
C    3 194.500  64.833 127.8904 127.8904  24.000 6.772e-15
A:C  3  19.375   6.458  12.7397  12.7397  24.000 3.508e-05
```

Example 14C: Repeated measures ANOVA

Mullens (1993) investigated the impact of hypoxia (oxygen stress) on the ventilation patterns of cane toads (*Bufo marinus*). In anticipation of variability in ventilation patterns between individual toads, each oxygen concentration level (O2LEVEL: 0, 5, 10, 15, 20, 30, 40 and 50%) was measured from each individual. Hence the individual toads represent the blocks (TOADS) and the oxygen levels represent a within block treatment. Individual toads also categorized according to their typical predominant mode of breathing (BRTH.TYP: buccal or lung) and therefore breathing type represents a between block treatment. Ventilation patterns were measured as the frequency of buccal breathing (Box 11.2 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Mullens (1993) data set.

```
> mullens <- read.table("mullens.csv", header = T, sep = ",")
```

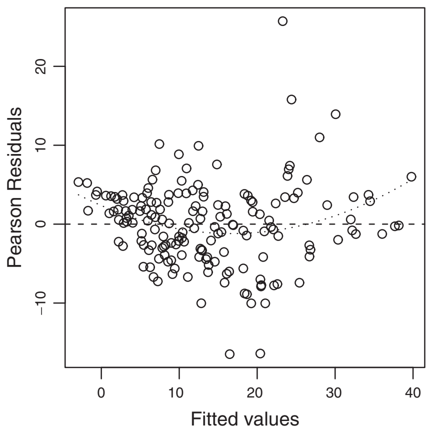
Step 2 - In order to ensure that the oxygen concentration variable is treated as a factor we need to redefine its class

```
> mullens$O2LEVEL <- factor(mullens$O2LEVEL)
```

Step 3 (Key 14.2) - Assess whether there are likely to be any plot by within plot interactions.

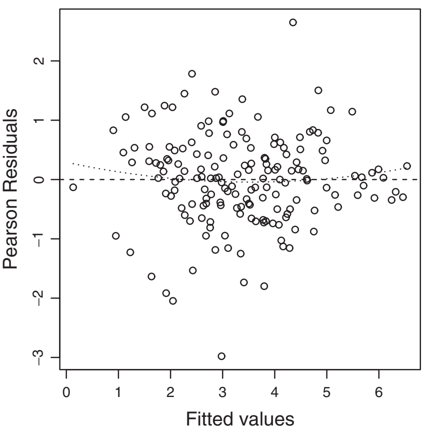
Raw data

```
> library(alr3)
> resplot(lm(FREQBUC ~
+      BRTH.TYP * O2LEVEL +
+      TOAD, data = mullens))
     t value     Pr(>|t|)
3.926581e+00 8.616205e-05
```



Square root transformed data

```
> resplot(lm(sqrt(FREQBUC) ~
+      BRTH.TYP * O2LEVEL + TOAD,
+      data = mullens))
  t value  Pr(>|t|)
1.2616950 0.2070586
```

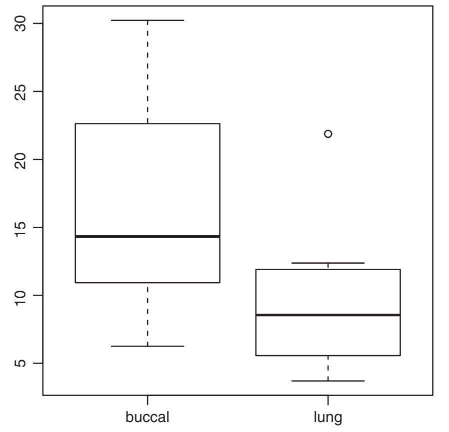


Conclusions - If raw data is appropriate, then there is some evidence of a blocking interaction (thus non-additive model). However, there is no strong evidence of a blocking interaction for square root transformed data (thus additive model).

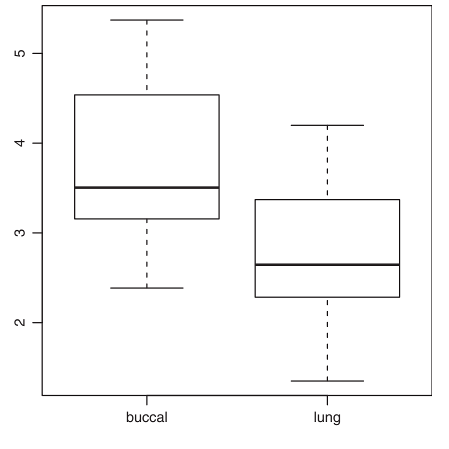
Step 4 (Key 14.2) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 14.1).

1. Between plot effect (Factor A: breathing type treatment - fixed effect). The means of each toad are the replicates for the breathing type effect and thus an aggregated dataset needs to be created from which the boxplots can be based.

```
> library(nlme)
> mullens.agg <- gsummary
+     (mullens, groups =
+     mullens$TOAD)
> boxplot(FREQBUC ~ BRTH.TYP,
+     mullens.agg)
```



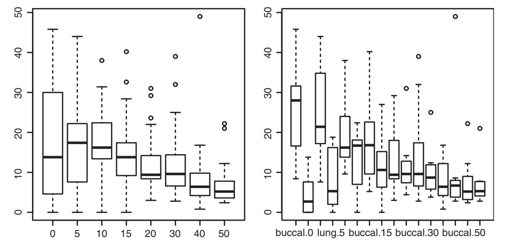

```
> mullens$SFREQBUC <- sqrt
+      (mullens$FREQBUC)
> mullens.agg <- gsummary
+      (mullens, groups =
+      mullens$TOAD)
> boxplot(SFREQBUC ~ BRTH.TYP,
+     mullens.agg)
```



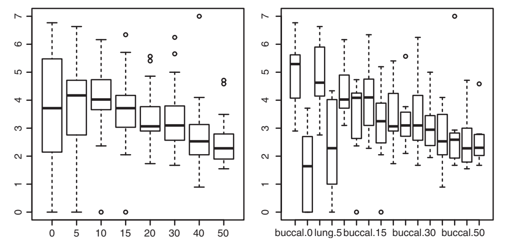
Conclusions - Square root transformed data appears to confirm to the parametric assumptions better than the raw data.

2. Within plot effects (Factor C: percentage oxygen treatment - fixed effect, A:C interaction - fixed factor). The individual frequency of buccal breathing measurements within each toad are the replicates for the effect of oxygen level.

```
> boxplot(FREQBUC ~
+      O2LEVEL, mullens)

> boxplot(FREQBUC ~
+      BRTH.TYP * O2LEVEL,
+      mullens)
```




```
> boxplot(sqrt(FREQBUC) ~
+      O2LEVEL, mullens)

> boxplot(sqrt(FREQBUC) ~
+      BRTH.TYP * O2LEVEL,
+      mullens)
```



Conclusions - Square root transformed data appears to confirm to the parametric assumptions better than the raw data.

Step 5 (Key 14.5) - Determine whether or not the design is balanced (at least with respect to sub-replication).

```
> replications(sqrt(FREQBUC) ~ BRTH.TYP * O2LEVEL + Error(TOAD),
+      mullens)
$BRTH.TYP
BRTH.TYP
buccal   lung
   104     64

$O2LEVEL
[1] 21

$'BRTH.TYP:O2LEVEL'
        O2LEVEL
BRTH.TYP  0  5 10 15 20 30 40 50
  buccal 13 13 13 13 13 13 13 13
  lung    8  8  8  8  8  8  8  8
> library(biology)
> is.balanced(sqrt(FREQBUC) ~ BRTH.TYP * O2LEVEL + Error(TOAD),
+      mullens)
[1] FALSE
```

Conclusions - The design is not balanced. Of the 21 toads, 13 where buccal breathing and only 8 where lung breathing. Consequently, type I Sums of Squares are not appropriate.

Step 6 (Key 14.8) - fit the linear model and produce an ANOVA table to test the null hypotheses that there no effects of breathing type, oxygen concentration or interaction on the pattern of ventilation (frequency of buccal breathing). Furthermore, we will treat the design as a repeated measures analysis to correct for deviations from sphericity that may result due to the ordered nature of the between plot effect (oxygen concentration).

```
> contrasts(mullens$TOAD) <- contr.helmert
> contrasts(mullens$BRTH.TYP) <- contr.helmert
> # define polynomial contrasts with a particular pattern of
  spacing between levels
> contrasts(mullens$O2LEVEL) <- contr.poly(8, c(0, 5, 10, 15, 20,
+     30, 40, 50))
> # create a new variable to represent the transformed response
> mullens$SFREQBUC <- sqrt(mullens$FREQBUC)
> mullens.aov <- aov(SFREQBUC ~ BRTH.TYP * O2LEVEL + Error(TOAD),
+     data = mullens)
> library(biology)
> AnovaM(mullens.aov, type = "III", RM = T)
   Sphericity Epsilon Values
-------------------------------
 Greenhouse.Geisser Huynh.Feldt
          0.4281775   0.5172755


Anova Table (Type III tests)
Response: SFREQBUC
Error: TOAD
          Df  Sum Sq Mean Sq F value  Pr(>F)
BRTH.TYP   1  39.921  39.921  5.7622 0.02678 *
Residuals 19 131.634   6.928
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
                  Df  Sum Sq Mean Sq F value    Pr(>F)
O2LEVEL            7  25.748   3.678  4.8841 6.258e-05 ***
BRTH.TYP:O2LEVEL   7  56.372   8.053 10.6928 1.228e-10 ***
Residuals        133 100.166   0.753
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Greenhouse-Geisser corrected ANOVA table
Response: SFREQBUC
Error: TOAD
                Df  Sum Sq Mean Sq F value  Pr(>F)
BRTH.TYP   0.42818  39.921  39.921  5.7622 0.04785 *
Residuals 19.00000 131.634   6.928
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Error: Within
                       Df  Sum Sq Mean Sq F value    Pr(>F)
O2LEVEL            2.9972  25.748   3.678  4.8841  0.002981 **
BRTH.TYP:O2LEVEL   2.9972  56.372   8.053 10.6928 2.435e-06 ***
Residuals        133.0000 100.166   0.753
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Huynh-Feldt corrected ANOVA table
Response: SFREQBUC
Error: TOAD
                Df  Sum Sq Mean Sq F value  Pr(>F)
BRTH.TYP   0.51728  39.921  39.921  5.7622 0.04393 *
Residuals 19.00000 131.634   6.928
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
                       Df  Sum Sq Mean Sq F value    Pr(>F)
O2LEVEL            3.6209  25.748   3.678  4.8841  0.001545 **
BRTH.TYP:O2LEVEL   3.6209  56.372   8.053 10.6928 4.223e-07 ***
Residuals        133.0000 100.166   0.753
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Both Greenhouse-Geisser and Huynh-Feldt epsilon estimates suggest that sphericity was not met (Greenhouse-Geisser preferred as they are both less than 0.75). There is a significant breathing type by oxygen level interaction ($P < 0.001$).

Step 7 - Explore the nature of the interaction further by evaluating the simple main effects.

- Effect of oxygen concentration for buccal breathing toads.

```
> library(biology)
> summary(mainEffects(mullens.aov, at = BRTH.TYP == "buccal"))
Error: TOAD
          Df  Sum Sq Mean Sq F value  Pr(>F)
INT        1  39.921  39.921  5.7622 0.02678 *
Residuals 19 131.634   6.928
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
           Df  Sum Sq Mean Sq F value    Pr(>F)
INT         7  19.907   2.844  3.7761 0.0009103 ***
O2LEVEL     7  75.433  10.776 14.3085 9.013e-14 ***
Residuals 133 100.166   0.753
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Effect of oxygen concentration for lung breathing toads.

```
> summary(mainEffects(mullens.aov, at = BRTH.TYP == "lung"))
Error: TOAD
          Df  Sum Sq Mean Sq F value  Pr(>F)
INT        1  39.921  39.921  5.7622 0.02678 *
Residuals 19 131.634   6.928
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
           Df  Sum Sq Mean Sq F value    Pr(>F)
INT         7  75.433  10.776 14.3085 9.013e-14 ***
O2LEVEL     7  19.907   2.844  3.7761 0.0009103 ***
Residuals 133 100.166   0.753
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Step 8 - Quinn and Keough (2002) also illustrated polynomial trends which can be useful for exploring the nature of the within plot treatments(s) in repeated measures designs (when the treatment has an ordered[f] set of levels). Such trends should be compared using a separately calculated error term (to reduce the impacts of deviations from sphericity), each of which estimates a different source of variation.

```
> # begin by defining the appropriate linear, quadratic and cubic terms
> p1 <- C(mullens$O2LEVEL, poly, 1, c(0, 5, 10, 15, 20, 30, 40, 50))
> p2 <- C(mullens$O2LEVEL, poly, 2, c(0, 5, 10, 15, 20, 30, 40, 50))
> p3 <- C(mullens$O2LEVEL, poly, 3, c(0, 5, 10, 15, 20, 30, 40, 50))
> # calculate the Linear trend
> # Note the use of Type III Sums of Squares due to design imbalance
> mullens.aovP1 <- aov(SFREQBUC ~ BRTH.TYP * p1 + Error(TOAD/(p1)),
> # data = mullens)
> AnovaM(mullens.aovP1, type = "III")[[2]]
            Df Sum Sq Mean Sq F value    Pr(>F)
p1           1 17.010  17.010  8.2555 0.0097341 **
BRTH.TYP:p1  1 40.065  40.065 19.4441 0.0003011 ***
Residuals   19 39.149   2.060
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> # calculate the Quadratic trend
> mullens.aovP2 <- aov(SFREQBUC ~ BRTH.TYP * (p1 + p2) + Error(TOAD/(p1 +
+ p2)), data = mullens)
> AnovaM(mullens.aovP2, type = "III")[[3]]
            Df  Sum Sq Mean Sq F value   Pr(>F)
p2           1  5.0069  5.0069  6.9667 0.016162 *
BRTH.TYP:p2  1 12.3256 12.3256 17.1498 0.000555 ***
Residuals   19 13.6553  0.7187
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[f] by ordered, I refer to a set of factor levels that have a natural order such as time, distance, concentration etc.

```
> # calculate the Cubic trend
> mullens.aovP3 <- aov(SFREQBUC ~ BRTH.TYP * (p1 + p2 + p3) +
+ Error(TOAD/(p1 + p2 + p3)), data = mullens)
> AnovaM(mullens.aovP3, type = "III")[[4]]
            Df  Sum Sq Mean Sq F value  Pr(>F)
p3           1  1.7470  1.7470  3.2625 0.08675 .
BRTH.TYP:p3  1  1.7839  1.7839  3.3314 0.08373 .
Residuals   19 10.1742  0.5355
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There are significant breathing type by oxygen concentration linear and quadratic interactions, suggesting that the nature of the trends in ventilation performance depend upon on the natural breathing type of the toads. We shall therefore explore the nature of the trends separately for each breathing type.

```
> # explore the trends for buccal breathing toads
> library(biology)
> mullens.aovB <- aov(SFREQBUC ~ p1 + p2 + p3 + Error(TOAD/(p1 +
+     p2 + p3)), data = mullens, subset = BRTH.TYP == "buccal")
> AnovaM(mullens.aovB)
Error: TOAD
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 12 88.286   7.357

Error: TOAD:p1
          Df Sum Sq Mean Sq F value    Pr(>F)
p1         1 71.719  71.719  178.87 1.432e-08 ***
Residuals 12  4.811   0.401
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: TOAD:p2
          Df  Sum Sq Mean Sq F value  Pr(>F)
p2         1 1.06373 1.06373  5.4981 0.03706 *
Residuals 12 2.32167 0.19347
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: TOAD:p3
          Df Sum Sq Mean Sq F value Pr(>F)
p3         1 0.0001  0.0001   2e-04 0.9877
Residuals 12 6.1020  0.5085

Error: Within
          Df  Sum Sq Mean Sq F value Pr(>F)
Residuals 52 22.6251  0.4351
> # explore the trends for lung breathing toads
> mullens.aovL <- aov(SFREQBUC ~ p1 + p2 + p3 + Error(TOAD/(p1 +
```

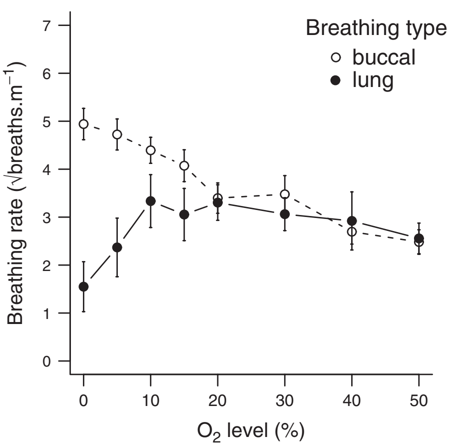
```
+     p2 + p3)), data = mullens, subset = BRTH.TYP == "lung")
> AnovaM(mullens.aovL)
Error: TOAD
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  7 43.349   6.193

Error: TOAD:p1
          Df Sum Sq Mean Sq F value Pr(>F)
p1         1  1.964   1.964  0.4004  0.547
Residuals  7 34.338   4.905

Error: TOAD:p2
          Df  Sum Sq Mean Sq F value  Pr(>F)
p2         1 13.3447 13.3447  8.2421 0.02396 *
Residuals  7 11.3336  1.6191
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: TOAD:p3
          Df Sum Sq Mean Sq F value  Pr(>F)
p3         1 2.8518  2.8518  4.9023 0.06242 .
Residuals  7 4.0722  0.5817
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
          Df  Sum Sq Mean Sq F value Pr(>F)
Residuals 32 18.9595  0.5925
```

Conclusions - Whilst buccal breathing toads respond to increasing hypoxia (reducing oxygen levels) with a significant linear increase in buccal breathing rate ($P < 0.001$), the breathing rates of lung breathing toads displays a quadratic trend ($P = 0.024$), initially rising before declining sharply at oxygen concentrations lower than 10 percent (see figure).

Step 9 - Summarize the trends in a plot.

```
> # calculate the mean and standard error of each group
> mullens.means <- with(mullens, tapply(SFREQBUC, list(BRTH.TYP,
+     O2LEVEL), mean))
> mullens.se <- with(mullens, tapply(SFREQBUC, list(BRTH.TYP,
+     O2LEVEL), function(x) ci(x)[4]))
> mullens$O2 <- as.numeric(as.character(mullens$O2LEVEL))
> # create a numeric version of the oxygen level variable
> xval <- unique(mullens$O2)
> # construct the base plot
> plot(SFREQBUC ~ O2, data = mullens, type = "n", axes = F,
+     xlab = "", ylab = "")
> # create some shortcuts objects
> mB <- mullens.means["buccal", ]
> seB <- mullens.se["buccal", ]
```

```
> # plot the error bars with open circle symbols for buccal breathing toads
> arrows(xval, mB - seB, xval, mB + seB, code = 3, angle = 90,
+     len = 0.01)
> points(mB ~ xval, pch = 16, col = "white", type = "b", lwd = 1,
+     lty = 2)
> points(mB ~ xval, pch = 1, col = "black", type = "b", lwd = 1,
+     lty = 2)
> mL <- mullens.means["lung", ]
> seL <- mullens.se["lung", ]
> points(mL ~ xval, pch = 19, type = "b", lwd = 1, lty = 1)
> arrows(xval, mL - seL, xval, mL + seL, code = 3, angle = 90,
+     len = 0.01)
> axis(1, cex.axis = 0.8)
> mtext(text = expression(paste(O[2], " level (%)")), side = 1,
+     line = 3)
> axis(2, cex.axis = 0.8, las = 1)
> mtext(text = expression(paste("Breathing rate ",
+     (sqrt(breaths.m^{-1
+ })))), side = 2, line = 3)
> legend("topright", leg = c("buccal", "lung"),
+     title = "Breathing type", lty = 0, pch = c(22, 19),
+     bty = "n", cex = 1)
> box(bty = "l")
```



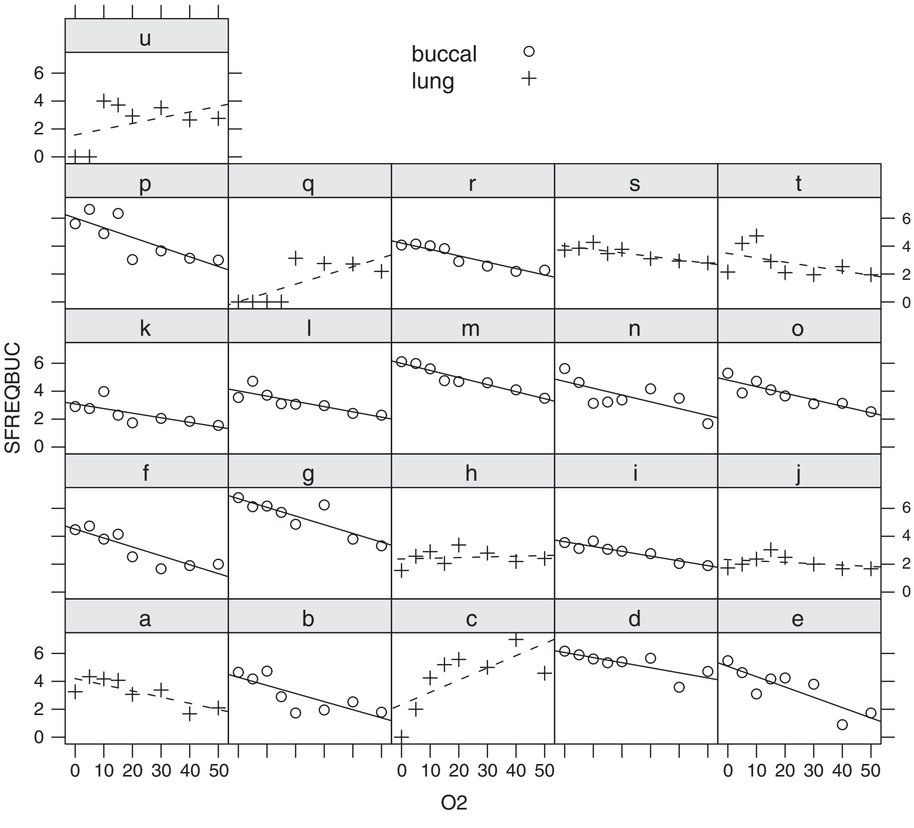
Example 14D: Linear mixed effects - unbalanced and compound symmetry violated

Alternatively, linear mixed effects modeling could be used to analyze the data introduced in Example 14C (Box 11.2 of Quinn and Keough (2002)). Notably, such an approach permits us to attempt to incorporate the nature of the variance-covariance matrix rather than wish it away post-hoc with estimated adjustments.

Step 1 (Key 14.2&14.5) - Refer to Example 14C for importing the data and performing exploratory data analysis.

Step 2 - Examine a lattice (trellis) plot in which the patterns of buccal breathing frequency against oxygen percentage are displayed for each individual toad.

```
> library(lattice)
> mullens$O2 <- as.numeric(as.character(mullens$O2LEVEL))
> xyplot(SFREQBUC ~ O2 | TOAD, groups = BRTH.TYP, mullens,
+      type = c("p", "r"), auto.key = T)
```



Conclusions - individual toads clearly display different propensities for buccal breathing. Thus a random intercepts model (essentially allowing each toad to have its own intercept) could be very useful. The slopes are fairly similar to one another (at least within a breathing type), and thus, models that incorporate random slopes in addition to random intercepts are perhaps overly complex.

Step 3 (Key 14.8) - Fit a series of lme models (with and without random slope components as well as alternative correlation structures) and compare them to evaluate the appropriateness of each.

```
> library(nlme)
> # model with random intercept and slope
> contrasts(mullens$BRTH.TYP) <- "contr.helmert"
> contrasts(mullens$O2LEVEL) <- contr.poly(8, c(0, 5, 10, 15, 20,
+ 30, 40, 50))
> # fit a model without correlation structure
> mullens.lme.1 <- lme(SFREQBUC ~ BRTH.TYP * O2LEVEL, random = ~1 |
+ TOAD, data = mullens)
> # fit a model with a compound symmetry correlation structure
```

```
> mullens.lme.2 <- lme(SFREQBUC ~ BRTH.TYP * O2LEVEL, random = ~1 |
+ TOAD, data = mullens, corr = corCompSymm(form = ~1 | TOAD))
> # compare the fit of models
> anova(mullens.lme.1, mullens.lme.2)
              Model df      AIC      BIC    logLik   Test
mullens.lme.1     1 18 518.8302 573.2601 -241.4151
mullens.lme.2     2 19 520.8302 578.2839 -241.4151 1 vs 2
                   L.Ratio p-value
mullens.lme.1
mullens.lme.2 1.136868e-13       1
```

Conclusions - Models incorporating either no correlation structure or compound symmetry are essentially equivalent (on the basis of AIC and log-likelihood ratio) to one another.

```
> # fit a model with a continuous first order autoregressive structure
> mullens.lme.3 <- lme(SFREQBUC ~ BRTH.TYP * O2LEVEL, random = ~1 | TOAD,
+ data = mullens, corr = corAR1(form = ~1 | TOAD))
> # compare the fit of models
> anova(mullens.lme.1, mullens.lme.3)
              Model df      AIC      BIC    logLik   Test  L.Ratio p-value
mullens.lme.1     1 18 518.8302 573.2601 -241.4151
mullens.lme.3     2 19 487.8255 545.2793 -224.9128 1 vs 2 33.00467  <.0001
```

Conclusions - A model that incorporates a continuous first order autoregressive correlation structure is a significantly better model that one without any correlation structure. Therefore use the autoregressive model to test the hypotheses about the fixed factors in the model (breathing rate, oxygen concentration and their interaction). Note a continuous time autoregressive structure is more appropriate than a regular first order autoregressive structure as the oxygen levels were not of equal spacing.
Examine the anova table for the model fit. As the design is not balanced, use marginal (Type III) sums of squares.

```
> anova(mullens.lme.3, type = "marginal")
                 numDF denDF   F-value p-value
(Intercept)          1   133 254.69303  <.0001
BRTH.TYP             1    19   5.99813  0.0242
O2LEVEL              7   133   2.96663  0.0064
BRTH.TYP:O2LEVEL     7   133   6.27943  <.0001
```

Conclusions - There is a significant breathing type by oxygen level interaction ($P < 0.001$)[g].

Step 4 - Explore the nature of the interaction further by evaluating the simple main effects.

- Effect of oxygen concentration for buccal breathing toads.

```
> library(biology)
> anova(mainEffects(mullens.lme.3, at = BRTH.TYP == "buccal"))
```

[g] Note, had we used the fitted linear effects model that assumed incorporated compound symmetry (mullens.lme.2), we would have produced the same *F*-ratios and *P*-values to a traditional split-plot ANOVA model that assumed compound symmetry (Step 5 of Example 14C).

```
               numDF denDF   F-value p-value
(Intercept)        1   132 283.35264  <.0001
M1                 8   132   3.79161   5e-04
M3                 7   132   7.02263  <.0001
```

- Effect of oxygen concentration for lung breathing toads.

```
> anova(mainEffects(mullens.lme.3, at = BRTH.TYP == "lung"))
               numDF denDF   F-value p-value
(Intercept)        1   132 283.35264  <.0001
M1                 8   132   7.18335  <.0001
M3                 7   132   3.14635  0.0042
```

Conclusions - There is a significant effect of oxygen concentration on the rate of breathing in both buccal ($P < 0.001$) and lung breathing toads, although the effect is perhaps stronger in the former ($P = 0.004$).

Step 5 - Quinn and Keough (2002) also illustrated polynomial trends which can be useful for exploring the nature of the within plot treatments(s) in repeated measures designs. Since the oxygen level contrasts were defined as polynomial contrasts prior to fitting the linear mixed effects model, the polynomial trends can be explored by examining their respective contrast estimates.

```
> summary(mullens.lme.3)$tTable
                          Value Std.Error  DF    t-value      p-value
(Intercept)          3.27055552 0.2049335 133 15.9591048 1.072635e-32
BRTH.TYP1           -0.50190423 0.2049335  19 -2.4491075 2.419395e-02
O2LEVEL.L           -0.92665742 0.2960086 133 -3.1305084 2.145655e-03
O2LEVEL.Q           -0.50274699 0.2350434 133 -2.1389536 3.426644e-02
O2LEVEL.C            0.29696969 0.1902856 133  1.5606525 1.209821e-01
O2LEVEL^4           -0.16531509 0.1656535 133 -0.9979572 3.201123e-01
O2LEVEL^5            0.12862277 0.1455696 133  0.8835824 3.785161e-01
O2LEVEL^6            0.21789466 0.1377385 133  1.5819442 1.160375e-01
O2LEVEL^7           -0.09384956 0.1248054 133 -0.7519672 4.533995e-01
BRTH.TYP1:O2LEVEL.L  1.42214172 0.2960086 133  4.8043931 4.125417e-06
BRTH.TYP1:O2LEVEL.Q -0.78879876 0.2350434 133 -3.3559702 1.031193e-03
BRTH.TYP1:O2LEVEL.C  0.30008904 0.1902856 133  1.5770455 1.171607e-01
BRTH.TYP1:O2LEVEL^4 -0.10039069 0.1656535 133 -0.6060282 5.455289e-01
BRTH.TYP1:O2LEVEL^5 -0.17042006 0.1455696 133 -1.1707115 2.438081e-01
BRTH.TYP1:O2LEVEL^6 -0.07034671 0.1377385 133 -0.5107264 6.103893e-01
BRTH.TYP1:O2LEVEL^7 -0.25859982 0.1248054 133 -2.0720245 4.019500e-02
```

Conclusions - There are significant breathing type by linear and quadratic interactions, suggesting that the nature of the trends in ventilation performance depend upon on the natural breathing type of the toads. We shall therefore explore the nature of the trends separately for each breathing type.

- Explore the polynomial trends for buccal breathing toads. nly terms beginning with `M3` are relevant to the trends of interest.

```
> summary(mainEffects(mullens.lme.3, at = BRTH.TYP == "buccal"))$tTable
                     Value Std.Error  DF     t-value      p-value
(Intercept)    3.772459750 0.2529754 132 14.91235638 4.296929e-30
M1INTlung.0   -3.391410652 0.5469173 132 -6.20095742 6.696311e-09
M1INTlung.5   -2.355189353 0.5469173 132 -4.30629917 3.213505e-05
M1INTlung.10  -1.058672073 0.5469173 132 -1.93570792 5.504132e-02
```

```
M1INTlung.15 -1.018480912 0.5469173 132 -1.86222119 6.479525e-02
M1INTlung.20 -0.092952990 0.5469173 132 -0.16995805 8.653033e-01
M1INTlung.30 -0.414694389 0.5469173 132 -0.75823972 4.496591e-01
M1INTlung.40  0.225424650 0.5469173 132  0.41217322 6.808811e-01
M1INTlung.50  0.075508077 0.5469173 132  0.13806124 8.904024e-01
M3O2LEVEL.L  -2.348799134 0.3654010 132 -6.42800489 2.169510e-09
M3O2LEVEL.Q   0.286051766 0.2901439 132  0.98589614 3.259878e-01
M3O2LEVEL.C  -0.003119349 0.2348936 132 -0.01327984 9.894246e-01
M3O2LEVEL^4  -0.064924399 0.2044871 132 -0.31749874 7.513669e-01
M3O2LEVEL^5   0.299042834 0.1796951 132  1.66416827 9.845096e-02
M3O2LEVEL^6   0.288241366 0.1700281 132  1.69525698 9.238432e-02
M3O2LEVEL^7   0.164750259 0.1540631 132  1.06936865 2.868552e-01
```

- Explore the polynomial trends for lung breathing toads. Only terms beginning with `M3` are relevant to the trends of interest.

```
> summary(mainEffects(mullens.lme.3, at = BRTH.TYP == "lung"))$tTable
                      Value Std.Error  DF    t-value      p-value
(Intercept)      6.16006195 0.4625892 133 13.3164859 3.078255e-26
M1INTlung.0     -3.39141065 0.5469173  19 -6.2009574 5.876085e-06
M1INTbuccal.5   -1.03622130 0.4195306 133 -2.4699543 1.477925e-02
M1INTbuccal.10 -2.33273858 0.5260936 133 -4.4340748 1.918234e-05
M1INTbuccal.15 -2.37292974 0.5783317 133 -4.1030600 7.066792e-05
M1INTbuccal.20 -3.29845766 0.6062166 133 -5.4410546 2.462483e-07
M1INTbuccal.30 -2.97671626 0.6216187 133 -4.7886530 4.410966e-06
M1INTbuccal.40 -3.61683530 0.6302675 133 -5.7385722 6.153507e-08
M1INTbuccal.50 -3.46691873 0.6351661 133 -5.4582867 2.275045e-07
M3O2LEVEL.L      0.49548430 0.4657967 133  1.0637352 2.893762e-01
M3O2LEVEL.Q     -1.29154575 0.3698624 133 -3.4919631 6.512915e-04
M3O2LEVEL.C      0.59705874 0.2994318 133  1.9939723 4.820021e-02
M3O2LEVEL^4     -0.26570578 0.2606709 133 -1.0193149 3.099041e-01
M3O2LEVEL^5     -0.04179729 0.2290672 133 -0.1824674 8.554938e-01
M3O2LEVEL^6      0.14754795 0.2167442 133  0.6807470 4.972150e-01
M3O2LEVEL^7     -0.35244937 0.1963927 133 -1.7946154 7.498631e-02
```

Conclusions - The rows in the above tables that are of interest are those labled `M302LEVEL.L`, `M302LEVEL.Q` and `M302LEVEL.C` representing linear, quadratic and cubic effects respectively. Whilst an increase in hypoxia (reduction in oxygen levels) was associated with a significant linear increase in buccal breathing rate by buccal breathing toads ($P < 0.001$), such as linear trend was not observed in lung breathing toads ($P = 0.289$). Instead, for lung breathing toads, increasing hypoxia was associated with a significant quadratic breathing rate ($P < 0.001$). The breathing rate of lung breathing toads initially increased as the oxygen concentration decreased before desplaying a sharp decline after oxygen concentrations lower than 10 percent (see Figure produced in Step 9 of Example 14C).

Example 14E: Repeated measures ANOVA

McGoldrick and Mac Nally (1998) investigated temporal changes in bird abundances in two different eucalypt habitat types (HABITAT: Ironbark and Stringybark) from two different regions (REGION: north and south) across south east Australia. Two sites (random plots) of each habitat/region combination were surveyed once a month for twelve months (MONTH: fixed within plot effect) and thus a partly nested design was employed (Box 11.4 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the McGoldrick and Mac Nally (1998) data set.

```
> mcgold <- read.table("mcgold.csv", header = T, sep = ",")
```

Step 2 - To preserve the natural chronological order of the month data (by default R orders all factors alphabetically), specify the logical sequence of months and define the factor as ordered[h]. In so doing, R will also define the contrasts for this factor as polynomials. Note the procedure below relies on the order of data in the data file reflecting the preferred order.

```
> # examine the first six entries in the MONTH vector
> head(mcgold$MONTH)
[1] MAY MAY MAY MAY MAY MAY
12 Levels: APRIL AUGUST DECEMBER FEBRUARY JANUARY JULY JUNE MARCH
    ... SEPTEMBER

> mcgold$MONTH <- ordered(mcgold$MONTH,
      levels = unique(mcgold$MONTH))
> # examine the first six entries in the MONTH vector again
> # note the format of the levels attribute
> head(mcgold$MONTH)
[1] MAY MAY MAY MAY MAY MAY
12 Levels: MAY < JUNE < JULY < AUGUST < SEPTEMBER < OCTOBER < ... <
    APRIL
```

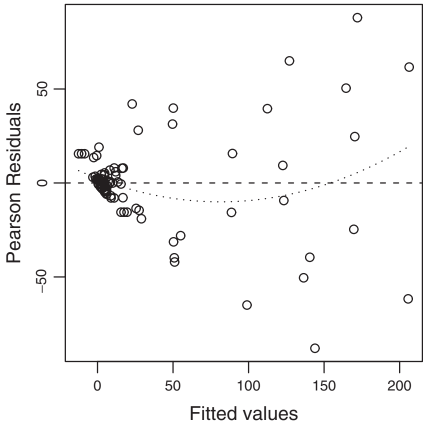
Step 3 (14.2) - Assess whether there are likely to be any plot by within plot interactions.

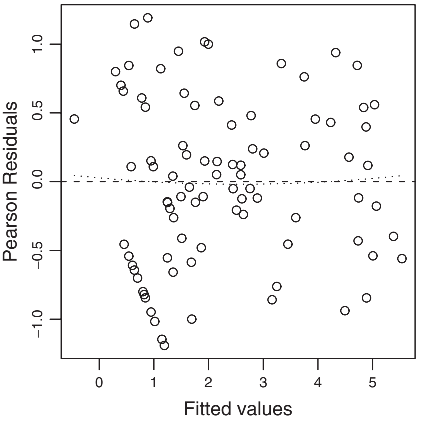
Raw data

```
> library(alr3)
> resplot(lm(BIRDS ~ HABITAT *
+     REGION * MONTH + SITE,
+     data = mcgold))
     t value     Pr(>|t|)
4.987846e+00 6.105625e-07
```

$Log_e + 1$ transformed data

```
> resplot(lm(log(BIRDS + 1) ~
+     HABITAT * REGION * MONTH +
+     SITE, data = mcgold))
  t value  Pr(>|t|)
0.7961976 0.4259172
```





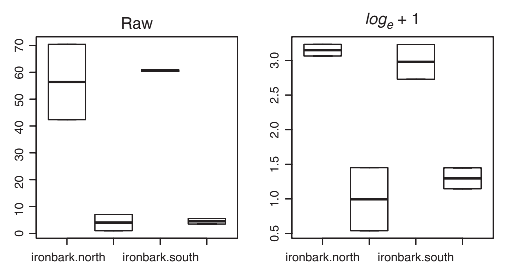
[h] Ordered factors are those in which the trends along the entire sequence of the levels are more interesting than the individual pairwise differences between levels.

Conclusions - The raw data shows a curvilinear trend implying that site by month interactions may be present. Moreover, the plot depicts a definite 'wedge' shape indicating that the assumption of homogeneity of variance is likely to be violated. Model fitting based on $log_e + 1$ transformed data shows no evidence of blocking interactions or non-homogeneity of variance.

Step 4 (14.2) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate F-ratio denominators see Table 14.3).

1. Between plot effects (Factor A: HABITAT - fixed effect, FACTOR C: REGION - fixed effect, A:C interaction - fixed). The mean bird abundances within each site (pooled over months) are the replicates for the between plot effects and thus an aggregated dataset needs to be created on which exploratory data analysis plots should be based. Prior to aggregating, we need to make a new variable to represent transformed data.

```
> library(nlme)
> mcgold$LBIRDS <- log
+     (mcgold$BIRDS + 1)
> mcgold.agg <- gsummary
+     (mcgold, groups =
+     mcgold$SITE)
> boxplot(BIRDS ~ HABITAT *
+     REGION, mcgold.agg)
```

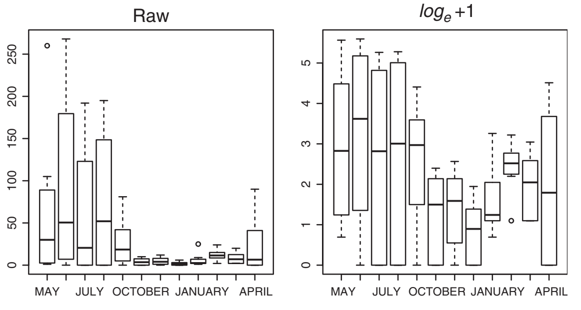



```
> boxplot(LBIRDS ~
+     HABITAT * REGION,
+     mcgold.agg)
```

Conclusions - $log_e + 1$ transformed data appears to confirm to the parametric assumptions better than the raw data.

2. Within plot effects (Factor D: MONTH - fixed effect, interactions - fixed factor, interactions involving month). The individual bird abundances within each month of each site are the replicates for the within site effects[i].

```
> boxplot(BIRDS ~
+     MONTH, mcgold)

> boxplot(LBIRDS ~
+     MONTH, mcgold)
```



[i] For the transformed data, there was no evidence of interactions involving the sites (blocks) and thus we can use this single pooled residual term. Had there have been strong evidence of blocking interactions, it would be appropriate to generate further appropriately aggregated datasets on which to perform exploratory data analysis.

Conclusions - $log_e + 1$ transformed data appears to confirm to the parametric assumptions better than the raw data.

Step 5 (Key 14.splitPlot-key-notMissing) - Determine whether or not the design is balanced (at least with respect to sub-replication).

```
> replications(log(BIRDS + 1) ~ HABITAT * REGION * MONTH +
+     Error(SITE), mcgold)
              HABITAT                REGION                 MONTH
                   48                    48                     8
       HABITAT:REGION         HABITAT:MONTH          REGION:MONTH
                   24                     4                     4
HABITAT:REGION:MONTH
                    2
> library(biology)
> is.balanced(log(BIRDS + 1) ~ HABITAT * REGION * MONTH +
+     Error(SITE), mcgold)
[1] TRUE
```

Conclusions - The design is balanced. There were exactly two sites per habitat/region combination and each site was surveyed every month.

Step 6 (Key 14.6) - fit the linear model and produce an ANOVA table to test the null hypotheses that there are no effects of habitat, region and month on the (log transformed) abundance of forest birds. Treat the design as a repeated measures analysis to correct for deviations from sphericity that may result due to the ordered nature of the between plot effect (month).

```
> mcgold.aov <- aov(LBIRDS ~ HABITAT * REGION * MONTH +
+     Error(SITE), data = mcgold)
> library(biology)
> AnovaM(mcgold.aov, RM = T)
   Sphericity Epsilon Values
-------------------------------
 Greenhouse.Geisser Huynh.Feldt
          0.2103946   0.5162103


Anova Table (Type I tests)
Response: LBIRDS
Error: SITE
               Df Sum Sq Mean Sq F value   Pr(>F)
HABITAT         1 88.313  88.313 48.9753 0.002194 **
REGION          1  0.106   0.106  0.0586 0.820678
HABITAT:REGION  1  1.334   1.334  0.7398 0.438236
Residuals       4  7.213   1.803
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
```

```
                    Df Sum Sq Mean Sq F value    Pr(>F)
MONTH               11 48.676   4.425  5.9408 8.029e-06 ***
HABITAT:MONTH       11 72.152   6.559  8.8061 5.488e-08 ***
REGION:MONTH        11 11.436   1.040  1.3957    0.2089
HABITAT:REGION:MONTH 11  3.858   0.351  0.4709    0.9113
Residuals           44 32.774   0.745
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Greenhouse-Geisser corrected ANOVA table
Response: LBIRDS
Error: SITE
                    Df Sum Sq Mean Sq F value   Pr(>F)
HABITAT        0.21039 88.313  88.313 48.9753 0.005497 **
REGION         0.21039  0.106   0.106  0.0586 0.398851
HABITAT:REGION 0.21039  1.334   1.334  0.7398 0.220487
Residuals      4.00000  7.213   1.803
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
                          Df Sum Sq Mean Sq F value    Pr(>F)
MONTH                 2.3143 48.676   4.425  5.9408 0.0036017 **
HABITAT:MONTH         2.3143 72.152   6.559  8.8061 0.0003426 ***
REGION:MONTH          2.3143 11.436   1.040  1.3957 0.2586627
HABITAT:REGION:MONTH  2.3143  3.858   0.351  0.4709 0.6552869
Residuals            44.0000 32.774   0.745
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Huynh-Feldt corrected ANOVA table
Response: LBIRDS
Error: SITE
                    Df Sum Sq Mean Sq F value   Pr(>F)
HABITAT        0.51621 88.313  88.313 48.9753 0.003255 **
REGION         0.51621  0.106   0.106  0.0586 0.644687
HABITAT:REGION 0.51621  1.334   1.334  0.7398 0.341802
Residuals      4.00000  7.213   1.803
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
                          Df Sum Sq Mean Sq F value    Pr(>F)
MONTH                 5.6783 48.676   4.425  5.9408 0.0001662 ***
HABITAT:MONTH         5.6783 72.152   6.559  8.8061 3.572e-06 ***
```

```
REGION:MONTH          5.6783 11.436   1.040  1.3957 0.2399414
HABITAT:REGION:MONTH  5.6783  3.858   0.351  0.4709 0.8171740
Residuals            44.0000 32.774   0.745
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Both Greenhouse-Geisser and Huynh-Feldt epsilon estimates suggest that sphericity was not met. There is a significant habitat by month interaction suggesting that the nature of the temporal patterns in bird abundances differ between the two habitats ($P < 0.001$). Similarly, whether or not there are differences in bird abundances in different habitats depends on the focal month.

Step 7 - Quinn and Keough (2002) presented the polynomial output that typically accompanies repeated measures analysis. Such trends should be compared using a separately calculated error term (to reduce the impacts of deviations from sphericity), each of which estimates a different source of variation.

```
> # begin by defining the appropriate linear, quadratic and cubic
> # terms
> MONTH.L <- C(mcgold$MONTH, poly, 1) # linear trend
> MONTH.Q <- C(mcgold$MONTH, poly, 2) # quadratic trend
> MONTH.C <- C(mcgold$MONTH, poly, 3) # cubic trend
> mcgold.aov <- aov(LBIRDS ~ HABITAT * REGION * MONTH + Error(SITE/
+ (MONTH.L + MONTH.Q + MONTH.C)), data = mcgold)
> summary(mcgold.aov)
Error: SITE
               Df Sum Sq Mean Sq F value   Pr(>F)
HABITAT         1 88.313  88.313 48.9753 0.002194 **
REGION          1  0.106   0.106  0.0586 0.820678
HABITAT:REGION  1  1.334   1.334  0.7398 0.438236
Residuals       4  7.213   1.803
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.L
                     Df  Sum Sq Mean Sq F value  Pr(>F)
MONTH                 1 16.0556 16.0556 12.2311 0.02496 *
HABITAT:MONTH         1 24.5324 24.5324 18.6887 0.01242 *
REGION:MONTH          1  3.0278  3.0278  2.3065 0.20345
HABITAT:REGION:MONTH  1  0.7168  0.7168  0.5460 0.50095
Residuals             4  5.2507  1.3127
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.Q
                     Df  Sum Sq Mean Sq F value  Pr(>F)
MONTH                 1 13.0991 13.0991  8.8971 0.04063 *
HABITAT:MONTH         1 17.9351 17.9351 12.1818 0.02512 *
```

```
REGION:MONTH          1  1.5739  1.5739  1.0690 0.35958
HABITAT:REGION:MONTH  1  0.8219  0.8219  0.5583 0.49648
Residuals             4  5.8891  1.4723
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.C
                     Df  Sum Sq Mean Sq F value   Pr(>F)
MONTH                 1  1.6960  1.6960  2.9426 0.161419
HABITAT:MONTH         1 22.6950 22.6950 39.3754 0.003293 **
REGION:MONTH          1  1.4015  1.4015  2.4316 0.193923
HABITAT:REGION:MONTH  1  0.1671  0.1671  0.2900 0.618808
Residuals             4  2.3055  0.5764
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
                     Df  Sum Sq Mean Sq F value   Pr(>F)
MONTH                 8 17.8249  2.2281  3.6889 0.003745 **
HABITAT:MONTH         8  6.9895  0.8737  1.4465 0.215806
REGION:MONTH          8  5.4324  0.6791  1.1242 0.373928
HABITAT:REGION:MONTH  8  2.1525  0.2691  0.4455 0.884351
Residuals            32 19.3284  0.6040
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - The nature of temporal trends in bird abundances differ between the two habitats.

Step 8 - Although Quinn and Keough (2002) did not show simple main effects tests, in this case, such tests would be useful to formally explore the nature of the habitat by trend interactions further.

- Effects of month in the ironbark region

```
> library(biology)
> summary(mainEffects(mcgold.aov, at = HABITAT == "ironbark"))
Error: SITE
          Df Sum Sq Mean Sq F value   Pr(>F)
INT        2 89.408  44.704  24.791 0.005573 **
REGION     1  0.344   0.344   0.191 0.684633
Residuals  4  7.213   1.803
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.L
             Df Sum Sq Mean Sq F value   Pr(>F)
INT           2  3.793   1.896  1.4447 0.337090
```

```
MONTH          1 40.140  40.140 30.5789 0.005225 **
REGION:MONTH  1  0.399   0.399  0.3040 0.610719
Residuals     4  5.251   1.313
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.Q
             Df  Sum Sq Mean Sq F value  Pr(>F)
INT           2  2.5249  1.2624  0.8575 0.48989
MONTH         1 30.8446 30.8446 20.9502 0.01021 *
REGION:MONTH  1  0.0605  0.0605  0.0411 0.84921
Residuals     4  5.8891  1.4723
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.C
             Df  Sum Sq Mean Sq F value   Pr(>F)
INT           2  7.2597  3.6298  6.2977 0.058096 .
MONTH         1 18.3996 18.3996 31.9230 0.004834 **
REGION:MONTH  1  0.3003  0.3003  0.5211 0.510325
Residuals     4  2.3055  0.5764
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
             Df  Sum Sq Mean Sq F value   Pr(>F)
INT          16 12.2722  0.7670  1.2699 0.273976
MONTH         8 16.7784  2.0973  3.4723 0.005442 **
REGION:MONTH  8  3.3488  0.4186  0.6930 0.694765
Residuals    32 19.3284  0.6040
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Effects of month in the stringybark region

```
> library(biology)
> summary(mainEffects(mcgold.aov, at = HABITAT == "stringybark"))
Error: SITE
          Df Sum Sq Mean Sq F value  Pr(>F)
INT        2 88.657  44.329 24.5832 0.00566 **
REGION     1  1.095   1.095  0.6073 0.47934
Residuals  4  7.213   1.803
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.L
             Df Sum Sq Mean Sq F value  Pr(>F)
```

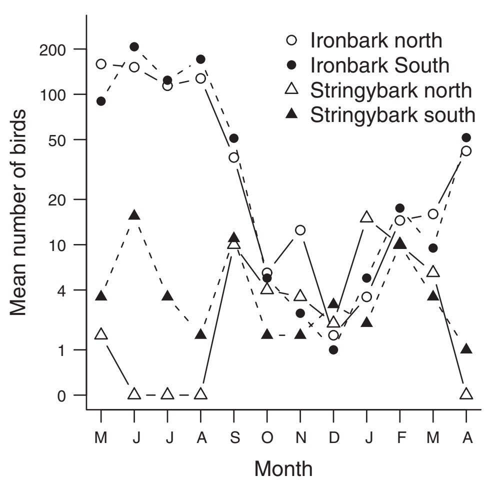
```
INT           2 40.540  20.270 15.4415 0.01315 *
MONTH         1  0.448   0.448  0.3409 0.59064
REGION:MONTH  1  3.345   3.345  2.5486 0.18563
Residuals     4  5.251   1.313
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.Q
             Df  Sum Sq Mean Sq F value  Pr(>F)
INT           2 30.9051 15.4526 10.4956 0.02562 *
MONTH         1  0.1896  0.1896  0.1288 0.73787
REGION:MONTH  1  2.3353  2.3353  1.5862 0.27635
Residuals     4  5.8891  1.4723
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: SITE:MONTH.C
             Df  Sum Sq Mean Sq F value  Pr(>F)
INT           2 18.7000  9.3500 16.2220 0.01205 *
MONTH         1  5.9914  5.9914 10.3949 0.03215 *
REGION:MONTH  1  1.2683  1.2683  2.2005 0.21212
Residuals     4  2.3055  0.5764
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
             Df  Sum Sq Mean Sq F value  Pr(>F)
INT          16 20.1272  1.2579  2.0827 0.03784 *
MONTH         8  8.0361  1.0045  1.6631 0.14622
REGION:MONTH  8  4.2361  0.5295  0.8767 0.54617
Residuals    32 19.3284  0.6040
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - Whereas bird abundances in the ironbark habitat increased substantially during the period from May-August (displaying a significant quadratic or even cubic trend through time), no real temporal trend was observed within the stringybark habitat. Bird abundances were not found to differ between the two regions and nor did the nature of the temporal trends.

Step 9 - Summarize the trends in a plot. Note that Quinn and Keough (2002, Fig. 11.5) plotted mean $log_e + 1$ number of birds on a linear scale. The following plot will illustrate plotting the mean number of birds on a log_e scale so as to depict the actual trends analysed, yet allow the actual bird abundances to be appreciated. To do so, slight modifications of the y-axis scale tick marks are necessary.

```
> mcgold.means <- with(mcgold, tapply(BIRDS + 1, list(interaction
+     (HABITAT, REGION), MONTH), mean))
```

```
> library(gmodels)
> mcgold.se <- with(mcgold, tapply(BIRDS + 1, list(interaction
+     (HABITAT, REGION), MONTH), function(x) ci(x)[4]))
> xval <- as.numeric(mcgold$MONTH)
> plot(BIRDS ~ xval, data = mcgold, type = "n", axes = F, xlab = "",
+     ylab = "", log = "y")
> xval <- unique(xval)
> points(mcgold.means["ironbark.north", ] ~ xval, pch = 1,
+     col = "black", type = "b", lwd = 1, lty = 1)
> points(mcgold.means["ironbark.south", ] ~ xval, pch = 16,
+     col = "black", type = "b", lwd = 1, lty = 2)
> points(mcgold.means["stringybark.north", ] ~ xval, pch = 2,
+     col = "black", type = "b", lwd = 1, lty = 1)
> points(mcgold.means["stringybark.south", ] ~ xval, pch = 17,
+     col = "black", type = "b", lwd = 1, lty = 2)
> axis(1, cex.axis = 0.8, at = xval, lab = substr(levels
+     (mcgold$MONTH), 1, 1))
> mtext(text = "Month", side = 1, line = 3)
> yticks <- ifelse(axTicks(2) > 9, axTicks(2), axTicks(2) - 1)
> axis(2, cex.axis = 0.8, las = 1, at = axTicks(2), lab = yticks)
> mtext(text = "Mean number of birds", side = 2, line = 3)
> legend("topright", leg = c("Ironbark north", "Ironbark South",
+     "Stringybark north", "Stringybark south"), lty = 0,
+     pch = c(1, 16, 2, 17), bty = "n", cex = 1)
> box(bty = "l")
```



Example 14F: Linear mixed effects - multiple between plot factors

Alternatively, linear mixed effects modeling could be used to analyze the data introduced in Example 14E (Box 11.4 of Quinn and Keough (2002)). Notably, such an approach permits us to attempt to incorporate the nature of the variance-covariance matrix rather than wish it away post-hoc with estimated adjustments.

Step 1 (Key 14.2) - Refer to Example 14E for importing the data and performing exploratory data analysis.

Step 2 (Key 14.6) - Fit a series of lme models (with alternative correlation structures) and compare them to evaluate the appropriateness of each.

```
> library(nlme)
> # fit a model without correlation structure
> mcgold.lme.1 <- lme(LBIRDS ~ HABITAT * REGION * MONTH, random=~1 |
+ SITE, data=mcgold)
> # fit a model with a first order autoregressive correlation structure
> mcgold.lme.2 <- lme(LBIRDS ~ HABITAT * REGION * MONTH, random = ~1 |
+ SITE, data = mcgold, correlation = corAR1(form = ~1 | SITE))
> # compare the fit of models
> anova(mcgold.lme.1, mcgold.lme.2)
             Model df      AIC      BIC    logLik   Test  L.Ratio p-value
mcgold.lme.1     1 50 268.8264 362.3865 -84.41320
mcgold.lme.2     2 51 263.7146 359.1458 -80.85729 1 vs 2 7.111819  0.0077
> # fit a model with compound symmetry structure
> mcgold.lme.3 <- update(mcgold.lme.1, correlation = corCompSymm(form = ~1 |
+ SITE))
> anova(mcgold.lme.2, mcgold.lme.3)
             Model df      AIC      BIC    logLik
mcgold.lme.2     1 51 263.7146 359.1458 -80.85729
mcgold.lme.3     2 51 270.8264 366.2577 -84.41320
```

A model that incorporates a first order autoregressive correlation structure is a significantly better model than either no structure or a compound symmetry model. Therefore use the autoregressive model to test the hypotheses about the fixed factors in the model (habitat type, region, month and their interaction).

```
> anova(mcgold.lme.2)
                     numDF denDF   F-value p-value
(Intercept)              1    44 246.45562  <.0001
HABITAT                  1     4  57.87532  0.0016
REGION                   1     4   0.07888  0.7927
MONTH                   11    44   4.30356  0.0002
HABITAT:REGION           1     4   0.87245  0.4032
HABITAT:MONTH           11    44   5.55797  <.0001
REGION:MONTH            11    44   1.31472  0.2486
HABITAT:REGION:MONTH    11    44   0.48120  0.9050
```

Conclusions - There is a significant habitat type by month interaction ($P < 0.001$). Note that the model assuming compound symmetry (mcgold.lme.3) would have yielded the same F-ratios and P-values to a traditional split-plot ANOVA model that assumed compound symmetry (Step 5 of Example 14E).

Step 3 - Explore the nature of the interaction further by evaluating the simple main effects.

- Effect of month in the ironbark habitat.

```
> library(biology)
> anova(mainEffects(mcgold.lme.2, at = HABITAT == "ironbark"))
```

```
            numDF denDF   F-value p-value
(Intercept)     1    42 246.45562  <.0001
M1             24    42   3.96420  <.0001
M3              1     6   0.21333  0.6604
M4             11    42   7.81231  <.0001
M7             11    42   0.52446  0.8757
```

- Effect of month in the stringybark habitat.

```
> anova(mainEffects(mcgold.lme.2, at = HABITAT == "stringybark"))
            numDF denDF   F-value p-value
(Intercept)     1    42 246.45562  <.0001
M1             24    42   6.24138  <.0001
M3              1     6   0.73800  0.4233
M4             11    42   2.04922  0.0472
M7             11    42   1.27146  0.2739
```

Conclusions - There is a significant effect of month on the abundance of birds rate in both ironbark ($P < 0.001$) and stringybark habitats ($P = 0.0472$), although the effect is perhaps stronger in the former.

Step 4 - Quinn and Keough (2002) also illustrated polynomial trends which can be useful for exploring the nature of the within plot treatments(s) in repeated measures designs.

```
> op <- options(width = 200)
> summary(mcgold.lme.2)$tTable
                                      Value Std.Error DF     t-value      p-value
(Intercept)                      3.14903875 0.2759018 44 11.41362002 9.701113e-15
HABITATstringybark              -2.15401165 0.3901841  4 -5.52050052 5.257047e-03
REGIONsouth                     -0.16942080 0.3901841  4 -0.43420733 6.865331e-01
MONTH.L                         -2.85195444 0.8688652 44 -3.28238987 2.021045e-03
MONTH.Q                          2.65387603 0.7858802 44  3.37694715 1.541953e-03
MONTH.C                          1.87072523 0.7114088 44  2.62960646 1.173306e-02
MONTH^4                         -0.80528671 0.6473149 44 -1.24404167 2.200710e-01
MONTH^5                         -0.57180974 0.5935441 44 -0.96338205 3.406208e-01
MONTH^6                          0.29126582 0.5490521 44  0.53048849 5.984407e-01
MONTH^7                          0.23388559 0.5124500 44  0.45640666 6.503426e-01
MONTH^8                          0.29995662 0.4823553 44  0.62185818 5.372442e-01
MONTH^9                          1.05595693 0.4575438 44  2.30788155 2.576776e-02
MONTH^10                        -0.61026362 0.4369923 44 -1.39650874 1.695666e-01
MONTH^11                        -0.79906329 0.4198714 44 -1.90311424 6.358096e-02
HABITATstringybark:REGIONsouth   0.47151096 0.5518037  4  0.85449042 4.409904e-01
HABITATstringybark:MONTH.L       4.10097201 1.2287610 44  3.33748549 1.727032e-03
HABITATstringybark:MONTH.Q      -3.63565503 1.1114025 44 -3.27123167 2.086112e-03
HABITATstringybark:MONTH.C      -3.65768273 1.0060840 44 -3.63556395 7.228149e-04
HABITATstringybark:MONTH^4       0.69302331 0.9154415 44  0.75703723 4.530626e-01
HABITATstringybark:MONTH^5      -0.32033115 0.8393981 44 -0.38162005 7.045800e-01
HABITATstringybark:MONTH^6      -0.35115544 0.7764769 44 -0.45224196 6.533165e-01
HABITATstringybark:MONTH^7       0.24974620 0.7247138 44  0.34461358 7.320267e-01
HABITATstringybark:MONTH^8      -0.37318162 0.6821535 44 -0.54706402 5.870988e-01
HABITATstringybark:MONTH^9      -1.63946191 0.6470647 44 -2.53369098 1.492385e-02
HABITATstringybark:MONTH^10      0.60736034 0.6180005 44  0.98278294 3.310877e-01
HABITATstringybark:MONTH^11      0.08769159 0.5937879 44  0.14768168 8.832687e-01
REGIONsouth:MONTH.L             -0.63173897 1.2287610 44 -0.51412680 6.097360e-01
REGIONsouth:MONTH.Q              0.24603964 1.1114025 44  0.22137762 8.258226e-01
REGIONsouth:MONTH.C              0.54802968 1.0060840 44  0.54471562 5.886995e-01
REGIONsouth:MONTH^4             -0.96233481 0.9154415 44 -1.05122478 2.988948e-01
REGIONsouth:MONTH^5              0.18744949 0.8393981 44  0.22331416 8.243246e-01
REGIONsouth:MONTH^6              1.02414643 0.7764769 44  1.31896571 1.940041e-01
```

```
REGIONsouth:MONTH^7                      0.69912331 0.7247138 44  0.96468890 3.399730e-01
REGIONsouth:MONTH^8                     -0.49437049 0.6821535 44 -0.72472034 4.724601e-01
REGIONsouth:MONTH^9                     -0.26957160 0.6470647 44 -0.41660690 6.789914e-01
REGIONsouth:MONTH^10                     0.64105415 0.6180005 44  1.03730361 3.052616e-01
REGIONsouth:MONTH^11                     0.34908521 0.5937879 44  0.58789548 5.596079e-01
HABITATstringybark:REGIONsouth:MONTH.L  -1.19732182 1.7377305 44 -0.68901469 4.944315e-01
HABITATstringybark:REGIONsouth:MONTH.Q   1.28213649 1.5717605 44  0.81573274 4.190471e-01
HABITATstringybark:REGIONsouth:MONTH.C   0.57815559 1.4228176 44  0.40634553 6.864584e-01
HABITATstringybark:REGIONsouth:MONTH^4  -0.11272067 1.2946298 44 -0.08706788 9.310126e-01
HABITATstringybark:REGIONsouth:MONTH^5   0.62160040 1.1870882 44  0.52363455 6.031603e-01
HABITATstringybark:REGIONsouth:MONTH^6  -1.59393894 1.0981042 44 -1.45153707 1.537236e-01
HABITATstringybark:REGIONsouth:MONTH^7   0.07801131 1.0249000 44  0.07611602 9.396718e-01
HABITATstringybark:REGIONsouth:MONTH^8   0.31864555 0.9647107 44  0.33030167 7.427396e-01
HABITATstringybark:REGIONsouth:MONTH^9   1.11230331 0.9150876 44  1.21551562 2.306513e-01
HABITATstringybark:REGIONsouth:MONTH^10  0.13738776 0.8739847 44  0.15719699 8.758087e-01
HABITATstringybark:REGIONsouth:MONTH^11  0.03842338 0.8397429 44  0.04575612 9.637117e-01
```

Conclusions - Whereas bird abundances in the ironbark habitat increased substantially during the period from May-August (displaying a significant quadratic or even cubic trend through time), no real temporal trend was observed within the stringybark habitat. Bird abundances were not found to differ between the two regions and nor did the nature of the temporal trends.

Step 5 - Although Quinn and Keough (2002) did not show simple main effects tests, in this case, such tests would be useful to formally explore the nature of the habitat by trend interactions further.

- Explore the polynomial trends for the ironbark habitat. Only terms beginning with `M4` are relevant to the trends of interest.

```
> summary(mainEffects(mcgold.lme.2, at = HABITAT == "ironbark"))$tTable
                                      Value Std.Error DF    t-value      p-value
(Intercept)                       3.1490387 0.2759018 42 11.4136200 1.875333e-14
M1INTstringybark.north.MAY       -3.9079061 0.9070778 42 -4.3082370 9.696800e-05
M1INTstringybark.south.MAY       -2.9858859 0.9070778 42 -3.2917639 2.023752e-03
M1INTstringybark.north.JUNE      -4.9205932 0.9070778 42 -5.4246651 2.656305e-06
M1INTstringybark.south.JUNE      -2.5454031 0.9070778 42 -2.8061575 7.567240e-03
M1INTstringybark.north.JULY      -4.4090191 0.9070778 42 -4.8606847 1.671342e-05
M1INTstringybark.south.JULY      -3.7779198 0.9070778 42 -4.1649348 1.514844e-04
M1INTstringybark.north.AUGUST    -4.8277053 0.9070778 42 -5.3222617 3.718097e-06
M1INTstringybark.south.AUGUST    -4.4377135 0.9070778 42 -4.8923185 1.508977e-05
M1INTstringybark.north.SEPTEMBER -1.7739004 0.9070778 42 -1.9556210 5.718397e-02
M1INTstringybark.south.SEPTEMBER -1.8567860 0.9070778 42 -2.0469975 4.694857e-02
M1INTstringybark.north.OCTOBER   -0.7458274 0.9070778 42 -0.8222309 4.155892e-01
M1INTstringybark.south.OCTOBER   -0.5058004 0.9070778 42 -0.5576153 5.800672e-01
M1INTstringybark.north.NOVEMBER  -1.0797421 0.9070778 42 -1.1903523 2.405928e-01
M1INTstringybark.south.NOVEMBER  -0.2027326 0.9070778 42 -0.2235007 8.242294e-01
M1INTstringybark.north.DECEMBER   0.3465736 0.9070778 42  0.3820770 7.043307e-01
M1INTstringybark.south.DECEMBER   0.4236489 0.9070778 42  0.4670480 6.428795e-01
M1INTstringybark.north.JANUARY    0.8770096 0.9070778 42  0.9668515 3.391529e-01
M1INTstringybark.south.JANUARY   -0.3992538 0.9070778 42 -0.4401539 6.620826e-01
M1INTstringybark.north.FEBRUARY  -0.7076410 0.9070778 42 -0.7801326 4.396874e-01
M1INTstringybark.south.FEBRUARY  -0.4631705 0.9070778 42 -0.5106183 6.122919e-01
M1INTstringybark.north.MARCH     -1.0206102 0.9070778 42 -1.1251628 2.669091e-01
M1INTstringybark.south.MARCH     -0.4904146 0.9070778 42 -0.5406533 5.916025e-01
M1INTstringybark.north.APRIL     -3.6787781 0.9070778 42 -4.0556368 2.121728e-04
M1INTstringybark.south.APRIL     -2.9485769 0.9070778 42 -3.2506329 2.271796e-03
M3                               -0.1694208 0.3901841  6 -0.4342073 6.793175e-01
M4MONTH.L                        -2.8519544 0.8688652 42 -3.2823899 2.077910e-03
M4MONTH.Q                         2.6538760 0.7858802 42  3.3769471 1.589463e-03
M4MONTH.C                         1.8707252 0.7114088 42  2.6296065 1.189480e-02
M4MONTH^4                        -0.8052867 0.6473149 42 -1.2440417 2.203827e-01
M4MONTH^5                        -0.5718097 0.5935441 42 -0.9633821 3.408701e-01
M4MONTH^6                         0.2912658 0.5490521 42  0.5304885 5.985672e-01
M4MONTH^7                         0.2338856 0.5124500 42  0.4564067 6.504491e-01
M4MONTH^8                         0.2999566 0.4823553 42  0.6218582 5.373964e-01
M4MONTH^9                         1.0559569 0.4575438 42  2.3078816 2.600141e-02
```

```
M4MONTH^10                    -0.6102636 0.4369923 42 -1.3965087 1.698983e-01
M4MONTH^11                    -0.7990633 0.4198714 42 -1.9031142 6.389469e-02
M7REGIONsouth:MONTH.L         -0.6317390 1.2287610 42 -0.5141268 6.098580e-01
M7REGIONsouth:MONTH.Q          0.2460396 1.1114025 42  0.2213776 8.258712e-01
M7REGIONsouth:MONTH.C          0.5480297 1.0060840 42  0.5447156 5.888299e-01
M7REGIONsouth:MONTH^4         -0.9623348 0.9154415 42 -1.0512248 2.991664e-01
M7REGIONsouth:MONTH^5          0.1874495 0.8393981 42  0.2233142 8.243736e-01
M7REGIONsouth:MONTH^6          1.0241464 0.7764769 42  1.3189657 1.943270e-01
M7REGIONsouth:MONTH^7          0.6991233 0.7247138 42  0.9646889 3.402226e-01
M7REGIONsouth:MONTH^8         -0.4943705 0.6821535 42 -0.7247203 4.726419e-01
M7REGIONsouth:MONTH^9         -0.2695716 0.6470647 42 -0.4166069 6.790875e-01
M7REGIONsouth:MONTH^10         0.6410541 0.6180005 42  1.0373036 3.055298e-01
M7REGIONsouth:MONTH^11         0.3490852 0.5937879 42  0.5878955 5.597504e-01
```

- Explore the polynomial trends for the stringybark habitat. Only terms beginning with `M4` are relevant to the trends of interest.

```
> summary(mainEffects(mcgold.lme.2, at = HABITAT == "stringybark"))$tTable
                                   Value Std.Error DF      t-value      p-value
(Intercept)                   4.90293320 0.8840529 43  5.545972514 1.666066e-06
M1INTstringybark.north.MAY   -3.90790610 0.9070778  5 -4.308237006 7.655410e-03
M1INTironbark.south.MAY      -0.92202015 1.2828017 43 -0.718755004 4.761797e-01
M1INTironbark.north.JUNE      1.01268715 0.9804901 43  1.032837723 3.074541e-01
M1INTironbark.south.JUNE     -1.36250300 1.2828017 43 -1.062130637 2.941054e-01
M1INTironbark.north.JULY      0.50111305 1.1666564 43  0.429529255 6.696828e-01
M1INTironbark.south.JULY     -0.12998625 1.2828017 43 -0.101329963 9.197596e-01
M1INTironbark.north.AUGUST    0.91979925 1.2358358 43  0.744273003 4.607595e-01
M1INTironbark.south.AUGUST    0.52980735 1.2828017 43  0.413007985 6.816531e-01
M1INTironbark.north.SEPTEMBER -2.13400570 1.2634857 43 -1.688982842 9.846263e-02
M1INTironbark.south.SEPTEMBER -2.05112010 1.2828017 43 -1.598937763 1.171575e-01
M1INTironbark.north.OCTOBER  -3.16207870 1.2748058 43 -2.480439496 1.711415e-02
M1INTironbark.south.OCTOBER  -3.40210565 1.2828017 43 -2.652090045 1.115844e-02
M1INTironbark.north.NOVEMBER -2.82816400 1.2794831 43 -2.210395662 3.244846e-02
M1INTironbark.south.NOVEMBER -3.70517355 1.2828017 43 -2.888344719 6.041419e-03
M1INTironbark.north.DECEMBER -4.25447970 1.2814229 43 -3.320121431 1.840760e-03
M1INTironbark.south.DECEMBER -4.33155500 1.2828017 43 -3.376636435 1.566445e-03
M1INTironbark.north.JANUARY  -4.78491565 1.2822286 43 -3.731718085 5.531570e-04
M1INTironbark.south.JANUARY  -3.50865225 1.2828017 43 -2.735147776 9.022136e-03
M1INTironbark.north.FEBRUARY -3.20026515 1.2825634 43 -2.495210014 1.650603e-02
M1INTironbark.south.FEBRUARY -3.44473560 1.2828017 43 -2.685321954 1.025323e-02
M1INTironbark.north.MARCH    -2.88729595 1.2827026 43 -2.250947218 2.955563e-02
M1INTironbark.south.MARCH    -3.41749150 1.2828017 43 -2.664083987 1.082358e-02
M1INTironbark.north.APRIL    -0.22912800 1.2827605 43 -0.178621026 8.590742e-01
M1INTironbark.south.APRIL    -0.95932920 1.2828017 43 -0.747839039 4.586278e-01
M3                            0.30209015 0.3901841  5  0.774224621 4.737980e-01
M4MONTH.L                     1.24901756 0.8688652 43  1.437527379 1.578060e-01
M4MONTH.Q                    -0.98177900 0.7858802 43 -1.249273045 2.183231e-01
M4MONTH.C                    -1.78695751 0.7114088 43 -2.511857394 1.584424e-02
M4MONTH^4                    -0.11226340 0.6473149 43 -0.173429348 8.631278e-01
M4MONTH^5                    -0.89214089 0.5935441 43 -1.503074297 1.401293e-01
M4MONTH^6                    -0.05988963 0.5490521 43 -0.109078228 9.136479e-01
M4MONTH^7                     0.48363180 0.5124500 43  0.943763856 3.505631e-01
M4MONTH^8                    -0.07322501 0.4823553 43 -0.151807182 8.800490e-01
M4MONTH^9                    -0.58350498 0.4575438 43 -1.275298586 2.090508e-01
M4MONTH^10                   -0.00290328 0.4369923 43 -0.006643778 9.947298e-01
M4MONTH^11                   -0.71137170 0.4198714 43 -1.694260795 9.744800e-02
M7REGIONsouth:MONTH.L        -1.82906078 1.2287610 43 -1.488540721 1.439063e-01
M7REGIONsouth:MONTH.Q         1.52817613 1.1114025 43  1.374997929 1.762546e-01
M7REGIONsouth:MONTH.C         1.12618526 1.0060840 43  1.119374982 2.691941e-01
M7REGIONsouth:MONTH^4        -1.07505548 0.9154415 43 -1.174357359 2.467148e-01
M7REGIONsouth:MONTH^5         0.80904989 0.8393981 43  0.963845241 3.405129e-01
M7REGIONsouth:MONTH^6        -0.56979251 0.7764769 43 -0.733817709 4.670423e-01
M7REGIONsouth:MONTH^7         0.77713462 0.7247138 43  1.072333205 2.895520e-01
M7REGIONsouth:MONTH^8        -0.17572494 0.6821535 43 -0.257603236 7.979420e-01
M7REGIONsouth:MONTH^9         0.84273171 0.6470647 43  1.302391783 1.997153e-01
M7REGIONsouth:MONTH^10        0.77844190 0.6180005 43  1.259613725 2.146029e-01
M7REGIONsouth:MONTH^11        0.38750858 0.5937879 43  0.652604416 5.174852e-01
```

Conclusions - Whereas bird abundances in the ironbark habitat increased substantially during the period from May-August (displaying a significant quadratic or even cubic trend through time), no real temporal trend was observed within the stringybark habitat. Bird abundances were not found to differ between the two regions and nor did the nature of the temporal trends (see figure in Example 14E Step 10).

15

Analysis of covariance (ANCOVA)

Previous chapters have concentrated on designs for either continuous (Regression) or categorical (ANOVA) predictor variables. Analysis of covariance (ANCOVA) models are essentially ANOVA models that incorporate one or more continuous variables (**covariates**). Although the relationship between a response variable and a covariate may itself be of substantial biological interest, typically covariate(s) are incorporated to reduce the amount of unexplained variability in the model (analogous to blocking -see Chapter 13) and thereby increase the power of any treatment effects.

In ANCOVA, a reduction in unexplained variability is achieved by adjusting the response (to each treatment) according to slight differences in the covariate means as well as accounting for any underlying trends between the response and covariate(s), see Figure 15.1. To do so, the extent to which the within treatment group small differences in covariate means between groups and treatment groups are essentially compared via differences in their y-intercepts. The total variation is thereafter partitioned into explained (using the deviations between the overall trend and trends approximated for each of the treatment groups) and unexplained components (using the deviations between the observations and the approximated within group trends). In this way, ANCOVA can be visualized as a regular ANOVA in which the group and overall means are replaced by group and overall trendlines. Importantly, it should be apparent that ANCOVA is only appropriate when each of the within group trends have the same slope and are thus parallel to one another and the overall trend (see Figures 15.1e-f to visualize a situation in which slopes are not parallel). Furthermore, ANCOVA is not appropriate when the resulting adjustments must be extrapolated from a linear relationship outside the measured range of the covariate (see Figures 15.1g-h).

As an example, an experiment might be set up to investigate the energetic impacts of sexual vs parthenogenetic (egg development without fertilization) reproduction on leaf insect food consumption. To do so, researchers could measure the daily food intake of individual adult female leaf insects from female only (parthenogenetic) and mixed (sexual) populations. Unfortunately, the available individual leaf insects vary substantially in body size as this is expected to increase the variability of daily food intake of treatment groups. Consequently, the researchers will also measure the body mass of the individuals as a covariate, thereby providing a means by which daily food consumption can be standardized for body mass.

Although ANCOVA and blocking designs both aim to reduce the sources of unexplained variation by incorporating additional variables, blocking designs do so by

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

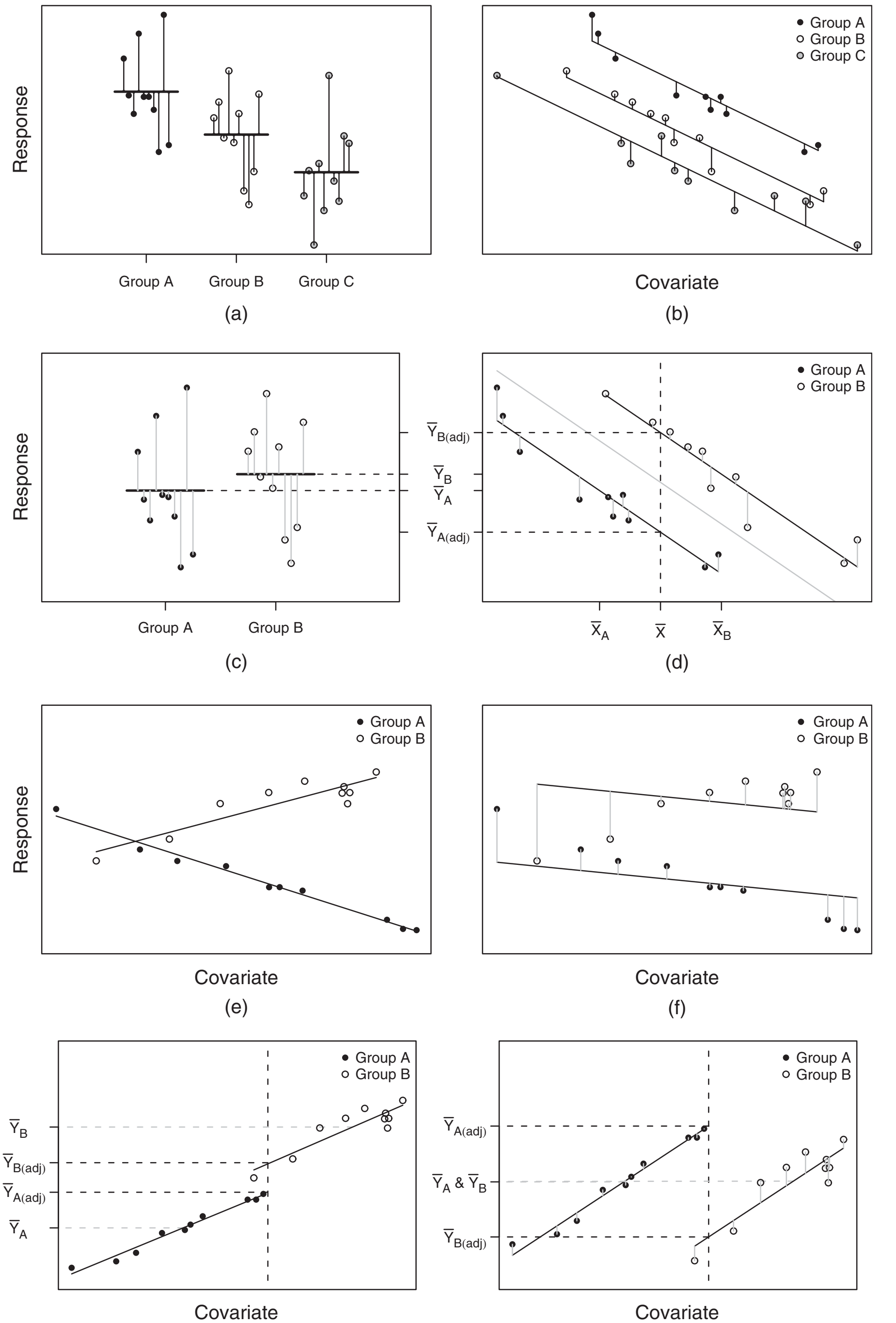


Fig 15.1 Fictitious data to illustrates the principles of analysis of covariance. The degree of unexplained variability (residuals) from single factor ANOVA and ANCOVA are represented in (a) and (b) respectively. (c) and (d) illustrate the use of the covariate in calculating adjusted group means (effects). The consequences of heterogeneous slopes are illustrated in (e) and (f) and the consequences of disparate covariate ranges on adjusted group means are illustrated in (g) and (h).

measuring a response to each level of a treatment factor under a similar (standardized) set of unmeasured conditions. By contrast, ANCOVA attempts to reduce unexplained variability by standardizing the response to the treatment by the effects of the specific covariate condition. Thus ANCOVA provides a means of exercising some statistical control over the variability when it is either not possible or not desirable to exercise experimental control (such as blocking or using otherwise homogeneous observations). From the example above, the researchers decided that the experiment would be too drawn out if each individual were to be measured under both sexual and parthenogenetic situations (due to the need to establish the new populations and allow enough time to minimize the risks of carryover effects).

15.1 Null hypotheses

15.1.1 *Factor A* - the main treatment effect

$H_0(A)\colon \mu_{1(adj)} = \mu_{2(adj)} = \ldots = \mu_{i(adj)} = \mu_{(adj)}$ (the adjusted population group means are all equal)

The mean of population 1 adjusted for the covariate is equal to that of population 2 adjusted for the covariate and so on, and thus all population means adjusted for the covariate are equal to an overall adjusted mean. If the effect of the i^{th} group is the difference between the i^{th} group adjusted mean and the overall adjusted mean ($\alpha_{i(adj)} = \mu_{i(adj)} - \mu_{(adj)}$) then the H_0 can alternatively be written as:

$H_0(A)\colon \alpha_{1(adj)} = \alpha_{2(adj)} = \ldots = \alpha_{i(adj)} = 0$ (the effect of each group equals zero)

If one or more of the $\alpha_{i(adj)}$ are different from zero (the response mean for this treatment differs from the overall response mean), the null hypothesis is not true indicating that the treatment does affect the response variable.

15.1.2 *Factor B* - the covariate effect

$H_0(B)\colon \beta_{1(pooled)} = 0$ (the pooled population slope equals zero)

Note, that this null hypothesis is rarely of much interest. It is precisely because of this nuisance relationship that ANCOVA designs are applied.

15.2 Linear models

One or more covariates can be incorporated into single factor, nested, factorial and partly nested designs in order to reduce the unexplained variation. Fundamentally, the covariate(s) are purely used to adjust the response values prior to the regular analysis. The difficulty is in determining the appropriate adjustments. Following is a list of the appropriate linear models and adjusted response calculations for a range of

ANCOVA designs. Note that these linear models do not include interactions involving the covariates as these are assumed to be zero. The inclusion of these interaction terms is however, a useful means of testing the homogeneity of slopes assumption (see section 15.4.1).

Single categorical and single covariate

$$\text{Linear model: } y_{ij} = \mu + \alpha_i + \beta(x_{ij} - \overline{x}) + \varepsilon_{ij}$$
$$\text{Adjustments: } y_{ij(adj)} = y_{ij} - b(x_{ij} - \overline{x})$$

Single categorical and two covariates (X&Z)

$$\text{Linear model: } y_{ij} = \mu + \alpha_i + \beta_{YX}(x_{ij} - \overline{x}) + \beta_{YZ}(z_{ij} - \overline{z}) + \varepsilon_{ij}$$
$$\text{Adjustments: } y_{ij(adj)} = y_{ij} - b_{YX}(x_{ij} - \overline{x}) - b_{YZ}(z_{ij} - \overline{z})$$

Special attention must be paid to the issues raised for multiple linear regression (see chapter 9).

Factorial designs (A&C categorical) with a single covariate)

$$\text{Linear model: } y_{ijk} = \mu + \alpha_i + \gamma_j + (\alpha\gamma)_{ij} + \beta(x_{ijk} - \overline{x}) + \varepsilon_{ijkl}$$
$$\text{Adjustments: } y_{ijk(adj)} = y_{ijk} - b(x_{ijk} - \overline{x})$$

where β is the population slope between the response and the covariate.

Nested designs (A&C categorical) with a single covariate)

$$\text{Linear model: } y_{ijk} = \mu + \alpha_i + \gamma_{j(i)} + \beta(x_{ijk} - \overline{x}) + \varepsilon_{ijk}$$
$$\text{Adjustments: } y_{ijk(adj)} = y_{ijk} - b(x_{ijk} - \overline{x})$$

Partly nested designs (A&C categorical) with a single covariate)

$$\text{Linear model: } y_{ijkl} = \mu + \alpha_i + \gamma_{j(i)} + \delta_k + (\alpha\delta)_{ik} + \gamma\delta_{j(i)k} + \beta(x_{ijk} - \overline{x}) + \varepsilon_{ijkl}$$
$$\text{Adjustments: } y_{ijk(adj)} = y_{ijk} - b_{between}(x_i - \overline{x}) - b_{within}(x_{ijk} - \overline{x}_i)$$

where $b_{between}$ and b_{within} refer to the between and within block/plot/subject regression slopes respectively.

15.3 Analysis of variance

In ANCOVA, the total variability of the response variable is sequentially partitioned into components explained by each of the model terms, starting with the covariate and is therefore equivalent to performing a regular analysis of variance on the response variables that have been adjusted for the covariate. The appropriate unexplained

Table 15.1 *F*-ratios and corresponding R syntax for simple ANCOVA (B is a covariate).

			***F*-ratio**	
Factor	**d.f.**	**MS**	**A&B fixed**	**A random, B fixed**
A	$a-1$	MS_A	$\frac{MS_A}{MS_{Resid}}$	$\frac{MS_A}{MS_{Resid}}$
B	1	MS_B	$\frac{MS_B}{MS_{Resid}}$	$\left[\frac{MS_A}{MS_{B\times A'}}\right]^a$
B×A	$a-1$	$MS_{B\times A}$	$\frac{MS_{B\times A}}{MS_{Resid}}$	$\frac{MS_{B\times A'}}{MS_{Resid}}$
Residual (=N′(B×A))	$(n-2)a$	MS_{Resid}		
	A&B fixed[b]			
	`> Anova(aov(DV~A*B, data), type="III")`[c]			

[a]If $P > 0.25$ for B×A′, pooled denominator for B could be $(SS_{B\times A'} + SS_{Resid})/((a-1)+(n-2)a)$.
[b]***For mixed models, it is necessary to manually calculate the correct F-ratios and P values.***
[c]To use type III sums of squares, Factor A contrasts must first be defined as something other than 'treatment' (such as 'sum' or 'helmert') prior to fitting the model (`> contrasts(data$A)<-contr.helmert`).

residuals and therefore the appropriate *F*-ratios for each factor differ according to the different null hypotheses associated with different linear models as well as combinations of fixed and random factors in the model (see Tables 15.1 & 15.2). Note that since the covariate levels measured are typically different for each group, ANCOVA designs are inherently non-orthogonal (unbalanced). Consequently, sequential (Type I sums of squares) should not be used[a].

15.4 Assumptions

As ANCOVA designs are essentially regular ANOVA designs that are first adjusted (centered) for the covariate(s), ANCOVA designs inherit all of the underlying assumptions of the appropriate ANOVA design. Readers should also consult sections 11.5, 12.4, 13.4 and 14.4. Specifically, hypothesis tests assume that:

(i) the appropriate residuals are normally distributed. Boxplots using the appropriate scale of replication (reflecting the appropriate residuals/*F*-ratio denominator, see Tables 15.1-15.2) should be used to explore normality. Scale transformations are often useful.
(ii) the appropriate residuals are equally varied. Boxplots and plots of means against variance (using the appropriate scale of replication) should be used to explore the spread of values. Residual plots should reveal no patterns (see Figure 8.5). Scale transformations are often useful.
(iii) the appropriate residuals are independent of one another.

[a] For very simple Ancova designs that incorporate a single categorical and single covariate, Type I sums of squares can be used provided the covariate appears in the linear model first (and thus is partitioned out last).

Table 15.2 *F*-ratios and corresponding R syntax for factorial ANCOVA (C is a covariate).

Factor	d.f.	*F*-ratio: A & B fixed	A fixed, B random	A & B random
A	$a-1$	$\frac{MS_A}{MS_{Resid}}$	$\left[\frac{MS_A}{MS_{B'\times A}}\right]^a$	$\left[\frac{MS'_A}{MS_{B'\times A'}}\right]^a$
B	$b-1$	$\frac{MS_B}{MS_{Resid}}$	$\left[\frac{MS_{B'}}{MS_{Resid}}\right]$	$\left[\frac{MS_{B'}}{MS_{B'\times A'}}\right]^a$
B×A	$(b-1)(a-1)$	$\frac{MS_{B\times A}}{MS_{Resid}}$	$\frac{MS_{B'\times A}}{MS_{Resid}}$	$\frac{MS_{B'\times A'}}{MS_{Resid}}$
C	1	$\frac{MS_C}{MS_{Resid}}$	$\left[\frac{MS_C}{MS_{C\times A'}}\right]^a$	$\left[\frac{MS_C}{MS_{C\times A'}+MS_{C\times B'}+MS_{C\times B'\times A'}}\right]^a$
C×A	$(a-1)$	$\frac{MS_{C\times A}}{MS_{Resid}}$	$\frac{MS_{C\times A}}{MS_{C\times B'\times A}}^a$	$\frac{MS_{C\times A}}{MS_{C\times B'\times A'}}^a$
C×B	$(b-1)$	$\frac{MS_{C\times B}}{MS_{Resid}}$	$\frac{MS_{C\times B'}}{MS_{Resid}}$	$\frac{MS_{C\times B'}}{MS_{C\times B'\times A'}}^a$
C×B×A	$(b-1)(a-1)$	$\frac{MS_{C\times B\times A}}{MS_{Resid}}$	$\frac{MS_{C\times B'\times A}}{MS_{Resid}}$	$\frac{MS_{C\times B'\times A'}}{MS_{Resid}}$
Residual (=N′(C×B×A))	$(n-2)ba$	MS_{Resid}		

R syntax

A & B fixed[b]

```
> Anova(aov(DV~A*B*C, data), type="III")
```
[c]

[a]Pooling: higher order interactions with $P > 0.25$ can be removed to produce more exact denominators.
[b]***For mixed models, it is necessary to manually calculate the correct F-ratios and P values.***
[c]To use type III sums of squares, Factor A contrasts must first be defined as something other than 'treatment' (such as 'sum' or 'helmert') prior to fitting the model (`contrasts(data$A)<-contr.helmert`).

(iv) the relationship between the response variable and the covariate should be linear. Linearity can be explored using scatterplots and residual plots should reveal no patterns (see fig 8.5).
(v) for repeated measures and other designs in which treatment levels within blocks can not be be randomly ordered, the variance/covariance matrix is assumed to display **sphericity** (see section 13.4.1).
(vi) for designs that utilize blocking, it is assumed that there are no block by within block interactions.

15.4.1 Homogeneity of slopes

In addition to the above assumptions, ANCOVA designs also assume that slopes of relationships between the response variable and the covariate(s) are the same for each treatment level (group). That is, all the trends are parallel. If the individual slopes deviate substantially from each other (and thus the overall slope), then adjustments made to each of the observations are nonsensical (see Figures 15.1e-f). This situation is analogous to an interaction between two or more factors. In ANCOVA, interactions involving the covariate suggest that the nature of the relationship between the response and the covariate differs between the levels of the categorical treatment. More importantly, they also indicate that the strength or presence of an effect of the treatment depends on what range of the covariate you are focussed on. Clearly then, it is not possible to make conclusions about the main effects of treatments in the presence of such interactions. The assumption of homogeneity of slopes can be examined via interaction plots or more formally, by testing hypotheses about the interactions between categorical variables and the covariate(s).

There are three broad approaches for dealing with ANCOVA designs with heterogeneous slopes and selection depends on the primary focus of the study.

(i) When the primary objective of the analysis is to investigate the effects of categorical treatments, it is possible to adopt an approach similar to that taken when exploring interactions in multiple regression. The effect of treatments can be examined at specific values of the covariate (such as the mean and $\pm$ one standard deviation). This approach is really only useful at revealing broad shifts in patterns over the range of the covariate and if the selected values of the covariate do not have some inherent biological meaning (selected arbitrarily), then the outcomes can be of only limited biological interest.

(ii) Alternatively, the Johnson-Neyman technique (or Wilxon modification thereof) procedure indicates the ranges of the covariate over which the individual regression lines of pairs of treatment groups overlap or cross. Although less powerful than the previous approach, the Wilcox(J-N) procedure has the advantage of revealing the important range (ranges for which the groups are different and not different) of the covariate rather than being constrained by specific levels selected.

(iii) Use contrast treatments to split up the interaction term into its constituent contrasts for each level of the treatment. Essentially this compares each of the treatment level slopes to the slope from the "control" group and is useful if the primary focus is on the relationships between the response and the covariate

15.4.2 Similar covariate ranges

Adjustments made to the response means (in an attempt to statistically account for differences in the covariate) involve predicting mean response values along displaced[b] linear relationships between the overall response and covariate variables (see Figure 15.1d). However, when the ranges of the covariate within each of the groups differ substantially from one another, these adjustments are effectively extrapolations (see Figures 15.1g-h) and therefore of unknown reliability. If a simple ANOVA of the covariate modelled against the categorical factor indicates that the covariate means differ significantly between groups, it may be necessary to either remove extreme observations or reconsider the analysis.

[b] The degree of trend displacement for any given group is essentially calculated by multiplying the overall regression slope by the degree of difference between the overall covariate mean and the mean of the covariate for that group.

15.5 Robust ANCOVA

ANCOVA based on rank transformed data can be useful for accommodating data with numerous problematic outliers. Nevertheless, the problems highlighted in section 12.7 about the difficulties of detecting interactions from rank transformed data obviously have implications for inferential tests of homogeneity of slopes. Randomization tests that maintain response-covariate pairs and repeatedly randomize these observations amongst the levels of the treatments can also be useful, particularly when there is doubt over the independence of observations.

15.6 Specific comparisons

Both planned and unplanned comparisons follow those of other ANOVA chapters without any real additional complications. Notably, recent implementations of the Tukey's test (within R) accommodate unbalanced designs and thus negate the need for some of the more complicated and specialized techniques that have been highlighted in past texts.

15.7 Further reading

- Theory

  Doncaster, C. P., and A. J. H. Davey. (2007). *Analysis of Variance and Covariance. How to Choose and Construct Models for the Life Sciences*. Cambridge University Press, Cambridge.

  Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

  Sokal, R., and F. J. Rohlf. (1997). *Biometry*, 3rd edition. W. H. Freeman, San Francisco.

  Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

  Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

  Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books.

  Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R - An Example-based Approach*. Cambridge University Press, London.

  Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS*, 4th edn. Springer-Verlag, New York.

15.8 Key for ANCOVA

Note, analysis of covariance (ANCOVA) design and analysis elements can be incorporated into more complex regression and ANOVA designs. The key presented here is for simple ANCOVA designs comprising a single categorical and a single covariate. For more complex designs, use the following key in combination with other appropriate keys from their respective chapters.

1 a. **Check parametric assumptions**

- **Normality of the response variable at each level of the categorical variable - boxplots**

```
> boxplot(DV ~ Factor, dataset)
```

where `DV` *and* `Factor` *are response and factor variables respectively in the* `dataset` *data frame*

- **Homogeneity of variance - residual plots**

```
> plot(aov(DV ~ CV + Factor, dataset), which = 1)
```

where `DV`, `CV` *and* `Factor` *are response, covariate and factor variables respectively in the* `dataset` *data frame*

Parametric assumptions met . . . . Go to 2

b. **Parametric assumptions NOT met** . . . . Go to 6

2 a. **Check assumptions of linearity and homogeneity of slopes** See Examples 15A & 15B

```
> library(lattice)
> xyplot(DV ~ CV | FACTOR, dataset, type = c("r", "p"))
> # OR
> library(car)
> scatterplot(DV ~ CV | FACTOR, dataset)
> # inference test for interaction (non-homogenous slopes)
> anova(aov(DV ~ CV * FACTOR, dataset))
```

Homogeneity of slopes assumption met . . . . Go to 3

b. **Homogeneity of slopes assumption NOT met** . . . . Go to 4

3 a. **Perform analysis of covariance** . . . . See Example 15A

```
> data.aov <- aov(DV ~ CV + FACTOR, dataset)
> anova(data.aov)
```

if Reject H_0 - Significant difference between group means detected, consider planned comparisons or post-hoc multiple pairwise comparisons tests . . . . . Go to Key 10.9a

4 a. **Primarily interested in the effect of the categorical variable** . . . . Go to 5

b. **Primarily interested in the effect of the continuous covariate**

```
> summary(lm(DV ~ CV, dataset, subset = FACTOR == "A"))
```

5 a. **Able to sensibly divide the range of the covariate into a small set of meaningful intervals (investigate the effect of the factor in each covariate interval separately using factorial analysis of variance (see chapter 12)**

```
> dataset$CV_F <- cut(dataset$CV, 4)
> data.aov1 <- aov(DV ~ CV_F * FACTOR, data = dataset)
```

. . . . Goto Key 12.11

b. **Investigate the effect of the factor at different values of the covariate (nominally, ± 1 and ± 2 standard deviations around the mean)**

```
> CV_sd2 <- mean(CV) - 2 * sd(CV)
> data.aov1 <- aov(DV ~ FACTOR + c(CV - CV_sd2), data = dataset)
> anova(data.lm2)
```

c. Use the Johnson-Neyman procedure to investigate the range(s) of the covariate for which the Factor levels are not significantly different . . . . See Example 15B

```
> data.lm <- lm(DV ~ CV * FACTOR, dataset)
> library(biology)
> wilcox.JN(data.lm, type = "H")
```

6 a. Attempt a scale transformation (see Table 3.2 for common transformation options) Go to 1

b. Transformations unsuccessful or inappropriate - see the range of options available for other analyses.

15.9 Worked examples of real biological data sets

Example 15A: Single factor ANCOVA

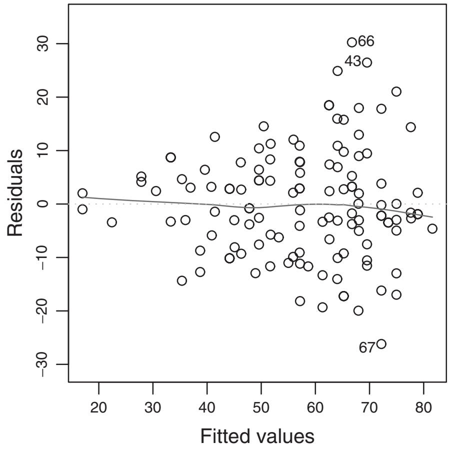
To investigate the impacts of sexual activity on male fruitfly longevity, Partridge and Farquhar (1981), measured the longevity of male fruitflies with access to either one virgin female (potential mate), eight virgin females, one pregnant female (not a potential mate), eight pregnant females or no females. The available male fruitflies varied in size and since size is known to impact longevity, the researchers randomly allocated each individual fruitfly to one of the five treatments and also measured thorax length as a covariate (from Box 12.1 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Partridge and Farquhar (1981) data set.

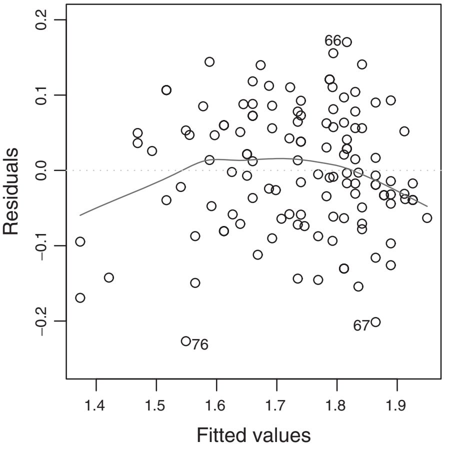
```
> partridge <- read.table("partridge.csv", header = T, sep = ",")
```

Step 2(Key 15.1) - Assess assumptions of normality and homogeneity of variance for each null hypothesis ensuring that the correct scale of replicates are represented for each (they should reflect the appropriate *F*-ratio denominators see Table 15.1).

```
> plot(aov(LONGEV ~
+      THORAX + TREATMENT,
+      partridge), which = 1)
```




```
> plot(aov(log10(LONGEV) ~
+      THORAX + TREATMENT,
+      partridge), which = 1)
```

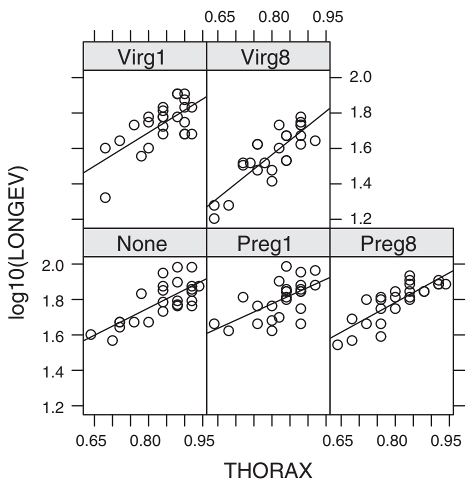


Conclusions - A distinct wedge shape is apparent in the residuals from the model fitted with the raw longevity measurements suggesting homogeneity of variance issues. This issue is less obvious in the residual plot based upon log_{10} transformed data, and thus analyses should be based on the transformed data.

Step 3 (Key 15.2) - Assess assumptions of linearity, homogeneity of slopes and covariate range equality (using log transformed data).

- Plot the relationship between male longevity and thorax length separately for each of the treatment groups in a lattice

```
> library(lattice)
> print(xyplot(log10(LONGEV) ~ THORAX | TREATMENT, partridge,
+      type = c("r","p")))
```



Conclusions - The slopes of each of the relationships between the response (longevity) and the covariate (thorax length) appear similar and there is no evidence of non-linearity.

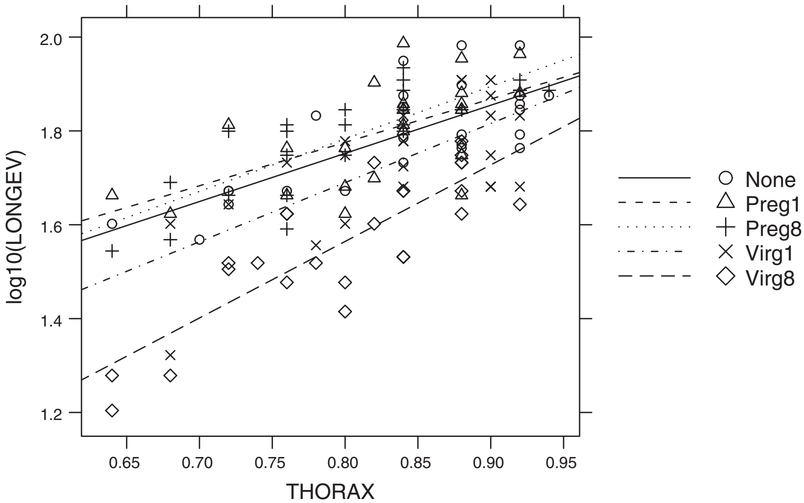
- The homogeneity of slopes assumption can also be formally tested by fitting the full multiplicative Anova model and examining the interaction term.

```
> anova(aov(log10(LONGEV) ~ THORAX * TREATMENT, partridge))
Analysis of Variance Table

Response: log10(LONGEV)
                  Df  Sum Sq Mean Sq  F value Pr(>F)
THORAX             1 1.21194 1.21194 176.4955 <2e-16 ***
TREATMENT          4 0.78272 0.19568  28.4970 <2e-16 ***
THORAX:TREATMENT   4 0.04288 0.01072   1.5611 0.1894
Residuals        115 0.78967 0.00687
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There is no evidence that the slopes are not parallel ($F_{4,115} = 1.56, P = 0.182$).

- Additionally, each of the relationships between longevity and the covariate (thorax length), could be placed on the same graph to enable simple comparisons of slopes and covariate ranges.

```
> library(lattice)
> print(with(partridge, xyplot(log10(LONGEV) ~
+     THORAX, groups = TREATMENT, type = c("p",
+     "r"), col = 1, par.settings = list(superpose.symbol =
+     list(pch = 1:5, col = 1), superpose.line = list(lty =
+     1:6)), key = list(space = "right", lty = 1:5,
+         lines = T, points = T, pch = 1:5,
+         col = 1, text = list(levels(TREATMENT))))))
```



- Formally, the covariate range disparity can be tested by modelling the effect of the treatments on the covariate (thorax length)

```
> anova(aov(THORAX ~ TREATMENT, partridge))
Analysis of Variance Table

Response: THORAX
           Df  Sum Sq Mean Sq F value Pr(>F)
TREATMENT   4 0.03000 0.00750  1.2606 0.2893
Residuals 120 0.71389 0.00595
```

Conclusions - There is no evidence that the treatments affect male fruitfly longevity and thus that the covariate ranges are not substantially different ($F_{4,120} = 1.26$, $P = 0.289$).

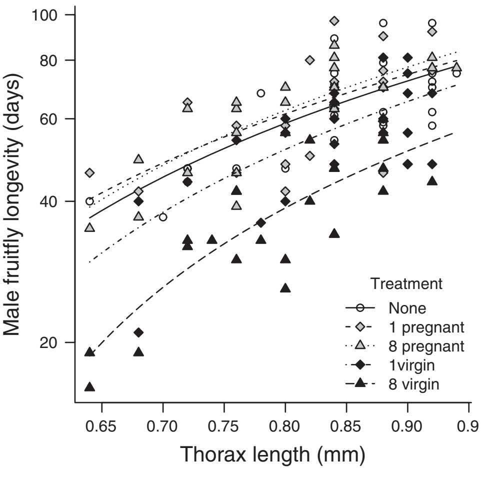
Step 4 (Key 15.3) - fit the linear model and produce an ANOVA table to test the null hypotheses that there are no effects of treatment (female type) on the (log transformed) longevity of male fruitflies adjusted for thorax length. Note that as the design is inherently imbalanced (since there is a different series of thorax lengths within each treatment type), Type I sums of squares are inappropriate. To be consistent with Quinn and Keough (2002) Box 12.1, Type III sums of squares will be used. In addition to the global ANCOVA, the researchers are likely to have been interested in examining a set of specific planned comparisons. Two such contrasts could be pregnant versus virgin partners (to investigate the impacts of any sexual activity) and one virgin versus eight virgin partners (to investigate the impacts of sexual frequency).

```
> # define contrasts
> contrasts(partridge$TREATMENT) <- cbind(c(0, 0.5, 0.5, -0.5,
+ -0.5), c(0, 0, 0, 1, -1))
> # confirm that contrasts orthogonal
> round(crossprod(contrasts(partridge$TREATMENT)), 1)
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    2    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1

> partridge.aov <- aov(log10(LONGEV) ~ THORAX +
+     TREATMENT, partridge)
> library(biology)
> AnovaM(partridge.aov, type = "III", split = list(TREATMENT =
+     list('Preg vs Virg' = 1, '1 Virg vs 8 Virg' = 2)))
                               Df  Sum Sq Mean Sq
THORAX                          1 1.01749 1.01749
TREATMENT                       4 0.78272 0.19568
  TREATMENT: Preg vs Virg       1 0.54203 0.54203
  TREATMENT: 1 Virg vs 8 Virg   1 0.19934 0.19934
Residuals                     119 0.83255 0.00700
                              F value    Pr(>F)
THORAX                        145.435 < 2.2e-16
TREATMENT                      27.970 2.231e-16
  TREATMENT: Preg vs Virg      77.474 1.269e-14
  TREATMENT: 1 Virg vs 8 Virg  28.492 4.567e-07
Residuals

THORAX                        ***
TREATMENT                     ***
  TREATMENT: Preg vs Virg     ***
  TREATMENT: 1 Virg vs 8 Virg ***
Residuals
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - The quantity ($F_{1,119} = 77.47, P < 0.001$) and reproductive state ($F_{1,119} = 28.49$, $P < 0.001$) of female partners that a male fruitfly has access to has a significant affect on male longevity.

Step 5 - Summarize the trends in a plot. Note this is not the same as the plot produced by Quinn and Keough (2002) (Figure 12.3). Whilst Quinn and Keough (2002) logged $\log_{10}$ transformed data on the y-axis, I have elected to plot the raw data on a log-scale y-axis.

```
> # create the base blank plot
> plot(LONGEV ~ THORAX, partridge, type = "n", axes = F, xlab = "",
+ ylab = "", log = "y")
> xs <- seq(min(partridge$THORAX), max(partridge$THORAX), l = 1000)
> # plot the None series
> part.lm <- lm(LONGEV ~ THORAX, partridge, subset = TREATMENT ==
     "None")
> lines(xs, predict(part.lm, data.frame(THORAX = xs)), lty = 1)
> points(LONGEV ~ THORAX, partridge, subset = TREATMENT == "None",
+ type = "p", pch = 1)
> # plot the Preg1 series
> part.lm <- lm(LONGEV ~ THORAX, partridge, subset = TREATMENT ==
     "Preg1")
> lines(xs, predict(part.lm, data.frame(THORAX = xs)), lty = 2)
> points(LONGEV ~ THORAX, partridge, subset = TREATMENT == "Preg1",
+ type = "p", pch = 23, bg = "gray")
> # plot the Preg8 series
> part.lm <- lm(LONGEV ~ THORAX, partridge, subset = TREATMENT ==
     "Preg8")
> lines(xs, predict(part.lm, data.frame(THORAX = xs)), lty = 3)
> points(LONGEV ~ THORAX, partridge, subset = TREATMENT == "Preg8",
+ type = "p", pch = 24, bg = "gray")
> # plot the Virg1 series
> part.lm <- lm(LONGEV ~ THORAX, partridge, subset = TREATMENT ==
     "Virg1")
> lines(xs, predict(part.lm, data.frame(THORAX = xs)), lty = 4)
> points(LONGEV ~ THORAX, partridge, subset = TREATMENT == "Virg1",
+ type = "p", pch = 23, bg = "black")
> # plot the Virg8 series
> part.lm <- lm(LONGEV ~ THORAX, partridge, subset = TREATMENT ==
     "Virg8")
> lines(xs, predict(part.lm, data.frame(THORAX = xs)), lty = 5)
> points(LONGEV ~ THORAX, partridge, subset = TREATMENT == "Virg8",
+ type = "p", pch = 24, bg = "black")
> axis(1)
> mtext("Thorax length (mm)", 1, line = 3)
> axis(2, las = 1)
> mtext(expression(paste("Male fruitfly longevity (days)")), 2,
     line = 3)
> legend("bottomright", legend = c("None", "1 pregnant",
+ "8 pregnant", "1virgin", "8 virgin"), bty = "n", title =
+ "Treatment", lty = 1:6, pch = c(1, 23, 24, 23, 24),
+ pt.bg = c(1, "gray", "gray", 1, 1))
> box(bty = "l")
```



Example 15B: Single factor ANCOVA - nonparallel slopes

Constable (1993) compared the inter-radial suture widths of urchins maintained on one of three food regimes (Initial: no additional food supplied above what was in the initial sample, low: food supplied periodically and high: food supplied *ad libitum*). In an attempt to control for substantial variability in urchin sizes, the initial body volume of each urchin was measured as a covariate (from Box12.2 of Quinn and Keough (2002)).

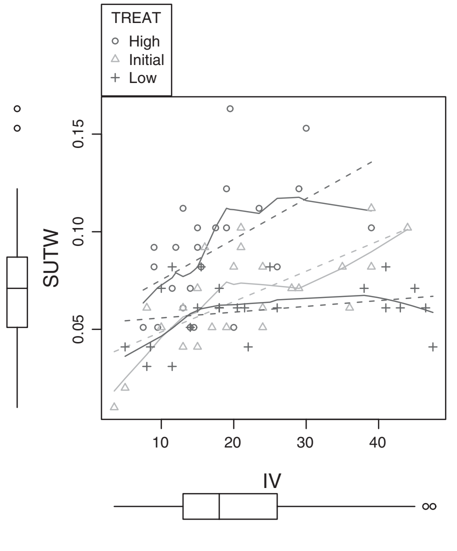
Step 1 - Import (section 2.3) the Constable (1993) data set.

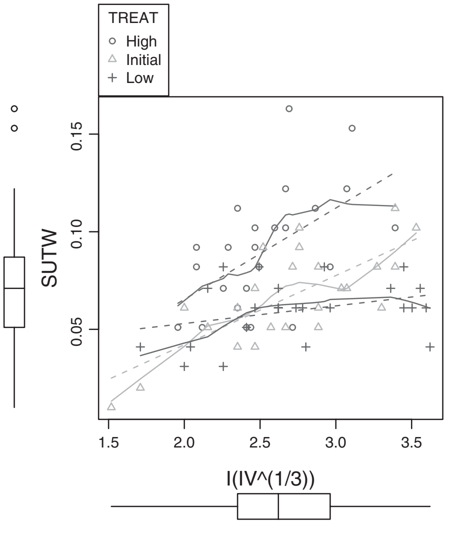
```
> constable <- read.table("constable.csv", header = T, sep = ",")
```

Step 2 (Key 15.1) - Assess assumptions of linearity and homogeneity of slopes.

```
> library(car)
> scatterplot(SUTW ~
+      IV | TREAT, constable)
```

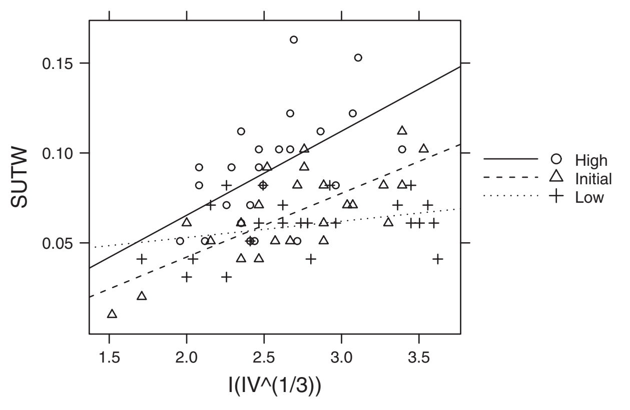
```
> library(car)
> scatterplot(SUTW ~
+      I(IV^(1/3)) | TREAT,
+      constable)
```





Conclusions - The relationship between suture width and initial volume shows some evidence of being non-linear. Linearity appears to be improved by a cube-root ($\sqrt[3]{}$) transformation, as is initial volume normality.

```
> library(lattice)
> print(with(constable, xyplot(SUTW ~ I(IV^(1/3)),
+     groups = TREAT, type = c("p", "r"), col = 1,
+     par.settings = list(superpose.symbol = list(pch = 1:3,
+         col = 1), superpose.line = list(lty = 1:3)),
+     key = list(space = "right", lty = 1:3, lines = T,
+         points = T, pch = 1:3, col = 1,
+         text = list(levels(TREAT))))))
```




```
> anova(aov(SUTW ~ I(IV^(1/3)) * TREAT, constable))
Analysis of Variance Table

Response: SUTW
                  Df    Sum Sq   Mean Sq F value    Pr(>F)
I(IV^(1/3))        1 0.0065364 0.0065364 15.5799 0.0001945 ***
TREAT              2 0.0167503 0.0083752 19.9626 1.660e-07 ***
I(IV^(1/3)):TREAT  2 0.0039443 0.0019721  4.7007 0.0123436 *
Residuals         66 0.0276899 0.0004195
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - There is clear evidence that the relationships between suture width and initial volume differ between the three food regimes (slopes are not parallel and a significant interaction between food treatment and initial volume). Regular ANCOVA is not appropriate.

Step 3 (Key 12.5cc) - Determine the regions of difference between each of the food regimes pairwise using the Wilcox modification of the Johnson-Newman procedure (with Games-Howell critical value approximation).

```
> library(biology)
> constable.lm <- lm(SUTW ~ I(IV^(1/3)) * TREAT, constable)
> wilcox.JN(constable.lm, type = "H")
```

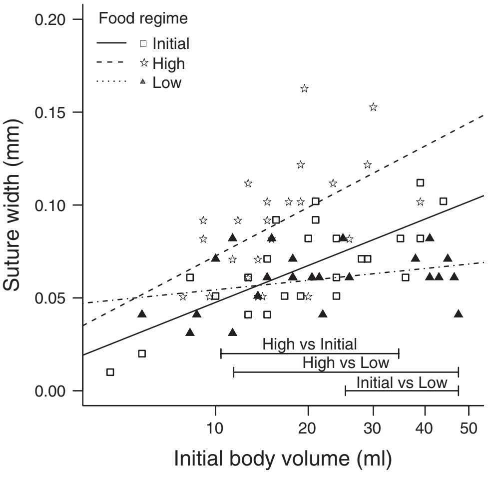
```
                df critical value     lower    upper
High vs Initial 37       3.867619  3.260903 2.187197
High vs Low     34       3.885401  6.595600 2.263724
Initial vs Low  43       3.839446 -1.547142 2.938749
```

Conclusions - Suture widths on a high food diet were greater than on initial diet for body volumes greater than 10.5 (2.19^3) ml and greater than a low food diet for body volumes greater than 11.6 (2.26^3), the latter of which was also lower than on initial diet for body volumes greater than 25.4 (2.94^3).

Step 4 (Key 12.18) - Summarize the trends in a plot. Bars at the bottom of the plot indicate Wilcox pairwise simultaneous regions of differences. Regions are capped to the data range. Note this plot also illustrates the use of Hershey fonts for special symbols (in this case a star).

```
> # fit the model and Wilcox modification of the Johnson-Newman
> constable.lm <- lm(SUTW ~ I(IV^(1/3)) * TREAT, constable)
> WJN <- wilcox.JN(constable.lm, type = "H")
> # create base plot
> plot(SUTW ~ I(IV^(1/3)), constable, type = "n", ylim = c(0,
+     0.2), xlim = c(3, 50)^(1/3), axes = F, xlab = "",
+     ylab = "")
> points(SUTW ~ I(IV^(1/3)), constable[constable$TREAT ==
+     "Initial", ], col = "black", pch = 22)
> lm1 <- lm(SUTW ~ I(IV^(1/3)), constable, subset = TREAT ==
+     "Initial")
> abline(lm1, col = "black", lty = 1)
> points(SUTW ~ I(IV^(1/3)), constable[constable$TREAT ==
+     "Low", ], col = "black", pch = 17)
> lm2 <- lm(SUTW ~ I(IV^(1/3)), constable, subset = TREAT ==
+     "Low")
> abline(lm2, col = "black", lty = 4)
> with(constable[constable$TREAT == "High", ], text(SUTW ~
+     I(IV^(1/3)), "\\#H0844", vfont = c("serif", "plain")))
> lm3 <- lm(SUTW ~ I(IV^(1/3)), constable, subset = TREAT ==
+     "High")
> abline(lm3, col = "black", lty = 2)
> axis(1, lab = c(10, 20, 30, 40, 50), at = c(10, 20,
+     30, 40, 50)^(1/3))
> axis(2, las = 1)
> mtext("Initial body volume (ml)", 1, line = 3)
> mtext("Suture width (mm)", 2, line = 3)
> Mpar <- par(family = "HersheySans", font = 2)
> library(biology)
> # the legend.vfont function facilitates Hershey fonts
> legend.vfont("topleft", c("\\#H0841 Initial", "\\#H0844 High",
+     "\\#H0852 Low"), bty = "n", lty = c(1, 2, 3),
+     merge = F, title = "Food regime", vfont = c("serif",
```

```
+         "plain"))
> par(Mpar)
> box(bty = "l")
> mn <- min(constable$IV^(1/3))
> mx <- max(constable$IV^(1/3))
> # since lower<upper (lines cross within the range - two regions
> # of significance (although one is outside data range))
> # region capped to the data range
> arrows(WJN[3, 4], 0, mx, 0, ang = 90, length = 0.05,
+     code = 3)
> text(mean(c(WJN[3, 4], mx)), 0.003, rownames(WJN)[3])
> # since lower>upper (lines cross outside data range
> # region capped to the data range if necessary
> arrows(min(WJN[2, 3], mx), 0.01, max(WJN[2, 4], mn),
+     0.01, ang = 90, length = 0.05, code = 3)
> text(mean(c(min(WJN[2, 3], mx), max(WJN[2, 4], mn))),
+     0.013, rownames(WJN)[2])
> # since lower>upper (lines cross outside data range
> # region capped to the data range if necessary
> arrows(min(WJN[1, 3], mx), 0.02, max(WJN[1, 4], mn),
+     0.02, ang = 90, length = 0.05, code = 3)
> text(mean(c(min(WJN[1, 3], mx), max(WJN[1, 4], mn))),
+     0.023, rownames(WJN)[1])
```



16

Simple Frequency Analysis

The analyses described in previous chapters have all involved response variables that implicitly represent normally distributed and continuous population responses. In this context, continuous indicates that (at least in theory), any value of measurement[a] down to an infinite number of decimal places is possible. Population responses can also be categorical such that the values could be logically or experimentally constrained to a set number of discrete possibilities. For example, individuals in a population can be categorized as either male or female, reaches in a stream could be classified as either riffles, runs or pools and salinity levels of sites might be categorized as either high, medium or low.

Typically, categorical response variables are tallied up to generate the frequency of replicates in each of the possible categories. From above, we would tally up the frequency of males and females, the number of riffles, runs and pools and the high, medium and low salinity sites. Hence, rather than model data in which a response was measured from each replicate in the sample (as was the case for previous analyses in this book), frequency analyses model data on the frequency of replicates in each possible category. Furthermore, frequency data follow a Poisson distribution rather than a normal distribution. The Poisson distribution is a symmetrical distribution in which only discrete integer values are possible and whose variance is equal to its mean (see Figure 16.1).

Frequency analysis essentially involves comparing the frequency of each category observed in a sample to the frequencies that might have been expected according to a particular scenario[b]. More specifically, it involves comparing the observed and expected frequency ratios. For example, if we are investigating population gender parity, the observed frequency of males and females could be compared to the frequency expected if the ratio of males to females was 1:1.

The frequencies expected for each category are determined by the size of the sample and the nature of the (null) hypothesis. For example, if the null hypothesis is that there are three times as many females as males in a population (ratio of 3:1), then a

[a] The term measurement is being used to refer to the characteristic of individual observations or replicates. Therefore a measurement could be a linear measure, a density, a count, etc.
[b] Dictated by the null hypothesis - see sections 16.2.2 and 16.2.2.

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

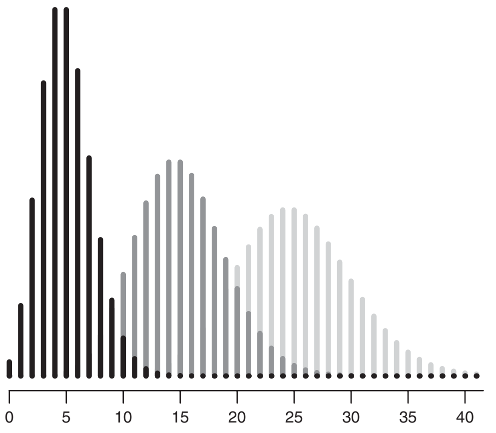


Fig 16.1 Poisson sampling distributions. The mean and variance of a Poisson distribution are equal and thus distributions with higher expected values are shorter and wider than those with smaller means. Note that a Poisson distribution with an expected less than less than 5 will be obviously asymmetrical as a Poisson distribution is bounded to the left by zero. This has important implications for the reliability of frequency analyses when sample sizes are low.

sample of 110 individuals would be expected to yield $0.75 * 110 = 82.5$ females and $0.25 * 110 = 27.5$ males.

16.1 The chi-square statistic

The degree of difference between the observed (o) and expected (e) sample category frequencies is represented by the chi-square (χ^2) statistic.

$$\chi^2 = \sum \frac{(o - e)^2}{e}$$

This is a relative measure that is standardized by the magnitude of the expected frequencies. When the null hypothesis is true (typically this represents the situation when there are no effects or patterns of interest in the population response category frequencies), and we have sampled in an unbiased manner, we might expect the observed category frequencies in the sample to be very similar (if not equal) to the expected frequencies and thus, the chi-square value should be close to zero. Likewise, repeated sampling from such a population is likely to yield chi-square values close to zero and large chi-square values should be relatively rare. As such, the chi-square statistic approximately follows a χ^2 distribution (see Figure 16.2), a mathematical probability distribution representing the frequency (and thus probability) of all possible ranges of chi-square statistics that could result when the null hypothesis is true.

The χ^2 distribution is an asymmetrical distribution bounded by zero and infinity and whose exact shape is determined by the degrees of freedom (calculated as the total

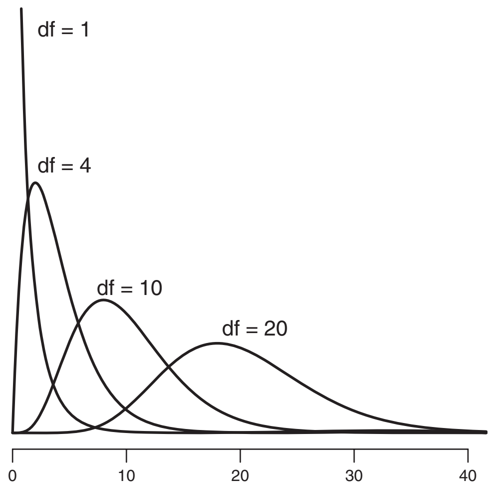


Fig 16.2 χ^2 probability distributions for a range of degrees of freedom. The expected value of the distribution is equal to the degrees of freedom. At low degrees of freedom, the χ^2 distribution is highly asymmetrical and approaches a more symmetrical shape with increasing degrees of freedom.

number of categories minus 1[c]). Note also that the peak of a chi-square distribution is not actually at zero (although it does approach it when the degrees of freedom is equal to zero). Initially, this might seem counter intuitive. We might expect that when a null hypothesis is true, the most common chi-square value will be zero. However, the χ^2 distribution takes into account the expected natural variability in a population as well as the nature of sampling (in which multiple samples should yield slightly different results). The more categories there are, the more likely that the observed and expected values will differ. It could be argued that when there are a large number of categories, samples in which all the observed frequencies are very close to the expected frequencies are a little suspicious and may represent dishonesty on the part of the researcher[d].

By comparing any given sample chi-square statistic to its appropriate χ^2 distribution, the probability that the observed category frequencies could have be collected from a population with a specific ratio of frequencies (for example 3:1) can be estimated. As is the case for most hypothesis tests, probabilities lower than 0.05 (5%) are considered unlikely and suggest that the sample is unlikely to have come from a population characterized by the null hypothesis. Chi-squared tests are typically one-tailed tests focussing on the right-hand tail as we are primarily interested in the probability of obtaining large chi-square values. Nevertheless, it is also possible to focus on the left-hand tail so as to investigate whether the observed values are "too good to be true".

[c] Recall that degrees of freedom is a measure of how many values are free to vary when determining independent estimates of parameters. Since estimations of the expected frequencies require multiplication by the total frequencies (which thereby include each of the category frequencies), not all of the frequencies are free to vary.
[d] Indeed the extraordinary conformity of Gregor Mendel's pea experiments have been subjected to such skepticism.

16.1.1 Assumptions

A chi-square statistic will follow a χ^2 distribution approximately provided;

(i) All observations are classified independently of one another. The classification of one replicate should not be influenced by or related to the classification of other replicates. Random sampling should address this.
(ii) No more than 20% of the expected frequencies are less than five. χ^2 distributions do not reliably approximate the distribution of all possible chi-square values under those circumstances[e]. Since the expected values are a function of sample sizes, meeting this assumption is a matter of ensuring sufficient replication. When sample sizes or other circumstances beyond control lead to a violation of this assumption, numerous options are available (see section 16.5)

16.2 Goodness of fit tests

16.2.1 Homogeneous frequencies tests

Homogeneous frequencies tests (often referred to as goodness of fit tests) are used to test null hypotheses that the category frequencies observed within a single variable could arise from a population displaying a specific ratio of frequencies. The null hypothesis ($\mathbf{H}_0$) is that the observed frequencies come from a population with a specific ratio of frequencies.

16.2.2 Distributional conformity - Kolmogorov-Smirnov tests

Strictly, goodness of fit tests are used to examine whether a frequency/sampling distribution is homogeneous with some declared distribution. For example, we might use a goodness of fit test to formally investigate whether the distribution of a response variable deviates substantially from a normal distribution. In this case, frequencies of responses in a set of pre-defined bin ranges are compared to those frequencies expected according to the mathematical model of a normal distribution. Since calculations of these expected frequencies also involve estimates of population mean and variance (both required to determine the mathematical formula), a two degree of freedom loss is incurred (hence $df = n - 2$).

16.3 Contingency tables

Contingency tables are used to investigate the associations between two or more categorical variables. That is, they test whether the patterns of frequencies in one categorical variable differ between different levels of other categorical variable(s) or

[e] Expected frequencies less than five result in asymmetrical sampling distributions (since they must be truncated at zero) and thus potentially unrepresentative χ^2 distributions.

could the variables be independent of another. In this way, they are analogous to interactions in factorial linear models (such as factorial ANOVA).

Contingency tables test the null hypothesis ($\mathbf{H}_0$) that the categorical variables are independent of (not associated with) one another. Note that analyses of contingency tables do not empirically distinguish between response and predictor variables (analogous to correlation), yet causality can be implied when logical and justified by interpretation. As an example, contingency tables could be used to investigate whether incidences of hair and eye color in a population are associated with one another (is one hair color type more commonly observed with a certain eye color). In this case, neither hair color nor eye color influence one another, their incidences are both controlled by a separate set of unmeasured factors. By contrast, an association between the presence or absence of a species of frog and the level of salinity (high, medium or low) could imply that salinity effects the distribution of that species of frog - but not vice versa.

Sample replicates are cross-classified according to the levels (categories) of multiple categorical variables. The data are conceptualized as a table (hence the name) with the rows representing the levels of one variable and the column the levels of the other variable(s) such that the cells represent the category combinations. The expected frequency of any given cell is calculated as:

$$\frac{(row\ total) \times (column\ total)}{grand\ total}$$

Thereafter, the chi-square calculations are calculated as described above and the chi-square value is compared to a χ^2 distribution with $(r-1)(c-1)$ degrees of freedom.

Contingency tables involving more than two variables have multiple interaction levels and thus multiple potential sources of independence. For example, in a three-way contingency table between variables A, B and C, there are four interactions (A:B, A:C, B:C and A:B:C). Such designs are arguably more appropriately analysed using log-linear models (see section 17.3.2).

16.3.1 Odds ratios

The chi-square test provides an indication of whether or not the occurrences in one set of categories are likely to be associated with other sets of categories (an interaction between two or more categorical variables), yet does not provide any indication of how strongly the variables are associated (magnitude of the effect). Furthermore, for variables with more than two categories (e.g. high, medium, low), there is no indication of which category combinations contribute most to the associations. This role is provided by odds ratios which are essentially a measure of effect size.

Odds refer the likelihood of a specific event or outcome occurring (such as the odds of a species being present) versus the it not occurring (and thus the occurrence of an alternative outcome) and are calculated as $\pi_j/(1-\pi_j)$ where π_j refers to the probability of the event occurring. For example we could calculate the odds of frogs being present in highly saline habitats as the probability of frogs being present divided by the probability of them being absent. Similarly, we could calculate the likelihood of frog presence (odds) within low salinity habitats.

The ratio of two of these likelihoods (odds ratio) can then be used to compare whether the likelihood of one outcome (frog presence) is the same for both categories (salinity levels). For example, is the likelihood of frogs being present in highly saline habitats the same as the probability of them being present in habitats with low levels of salinity. Although odds and thus odds ratios (θ) are technically derived from probabilities, they can also be estimated using cell frequencies (n).

$$\theta = \frac{n_{11}n_{22}}{n_{12}n_{21}} \quad \text{or alternatively}$$

$$\theta = \frac{(n_{11}+0.5)(n_{22}+0.5)}{(n_{12}+0.5)(n_{21}+0.5)}$$

where 0.5 is a small constant added to prevent division by zero. An odds ratio of one indicates that the event or occurance (presence of frogs) is equally likely in both categories (high and low salinity habitats). Odds ratios greater than one signify that the event or occurance is more likely in the first than second category and *vice versa* for odds ratios less than one. For example, when comparing the presence/absence of frogs in low versus high salinity habitats, a odds ratio of 5.8 would suggest that frogs are 5.8 times more likely to be present in low salinity habitats than those that highly saline.

The distribution of odds ratios (which range from 0 to ∞) is not symmetrical around the null possition (1) thereby precluding confidence interval and standard error calculations. Instead, these measures are calculated from log transformed (natural log) odds ratios (the distribution of which is a standard normal distribution centered around 0) and then converted back into a linear scale by anti-logging.

Odds ratios can only be calculated between category pairs from two variables and therefore 2×2 contingency tables (tables with only two rows and two columns). However, tables with more rows and columns can be accommodate by splitting the table up into **partial tables** of specific category pair combinations. Odds ratios (and confidence intervals) are then calculated from each pairing, nothwithstanding their lack of independence. For example, if there were three levels of salinity (high, medium and low), the odds ratios from three partial tables (high vs medium, high vs low, medium vs low) could be calculated.

Multi-way tables

Since odds ratios only explore pairwise patterns within two-way interactions, odds ratios for multi-way (three or more variables) tables are considerably more complex to calculate and interpret. Partial tables between two of the variables (e.g frog presence/absence and high/low salinity) are constructed for each level of a third (season: summer/winter). This essentially removes the effect of the third variable by holding it constant. Associations in partial tables are therefore referred to as *conditional associations*-since the outcomes (associated or independent) from each partial table are explicitly conditional on the level of the third variable at which they were tested.

Interpretation of odds ratios from three-way tables are summarised as:

- The odds ratios of partial tables (between X and Y) are the same for each level of Z and implies that the degree of association between X and Y (or effect of X on Y) is the same at all levels of Z. This is referred to as *homogeneous association* and is indicative of an absence of a three-way interaction.
- The odds ratios of partial tables (between X and Y) are all equal to 1 for each level of Z. This is a special case of homogeneous association referred to as *conditionally independence*. It implies that X and Y are not associated (independent) at all levels of Z.
- The odds ratios of partial tables (between X and Y) differ between the levels of Z implying that the degree of association between X and Y is not consistent across the levels of Z. This is equivalent to a three-way interaction between X, Y and Z.

16.3.2 Residuals

Specific contributions to a lack of independence (significant associations) can also be investigated by exploring the residuals. Recall that residuals are the difference between the observed values (frequencies) and those predicted or expected when the null hypothesis is true (no association between variables). Hence the magnitude of each residual indicates how much each of the cross classification combinations differs from what is expected. The residuals are typically standardized (by dividing by the square of the expected frequencies)[f] to enable individual residuals to be compared relative to one another. Large residuals (in magnitude) indicate large deviations from what is expected when the null hypothesis is true and thus also indicate large influences (contributions) to the overall association. The sign (+ or −) of the residual indicates whether the frequencies were higher or lower than expected.

16.4 G-tests

An alternative to the chi-square test for goodness of fit and contingency table analyses is the G-test. The G-test is based on a log likelihood-ratio test. A log likelihood ratio is a ratio of maximum likelihoods[g] of the alternative and null hypotheses. More simply, a log likelihood ratio test essentially examines how likely (the probability) the alternative hypothesis (representing an effect) is compared to how likely the null hypothesis (no effect) is given the collected data.

The G^2 statistic is calculated as:

$$G^2 = 2\sum o.ln\left(\frac{o}{e}\right)$$

where o and e are the observed and expected sample category frequencies respectively and ln denotes the natural logarithm (base e).

[f] Residuals can also be adjusted by dividing each residual by the square roots of the expected frequency as well as the observed frequency expressed as proportions of row and column totals.

[g] Recall that maximum likelihood refers to the maximum probability of obtaining a particular outcome given the observed data (see section 3.7.2).

When the null hypothesis is true, the G^2 statistic approximately follows a theoretical χ^2 distribution with the same degrees of freedom as the corresponding chi-square statistic. The G^2 statistic (which is twice the value of the log-likelihood ratio) is arguably more appropriate than the chi-square statistic as it is closely aligned with the theoretical basis of the χ^2 distribution (for which the chi-squared statistic is a convenient approximation). For large sample sizes, G^2 and χ^2 statistics are equivalent, however the former is a better approximation of the theoretical χ^2 distribution when the difference between the observed and expected is less than the expected frequencies (ie $|o - e| < e$). Nevertheless, G-tests operate under the same assumptions are the chi-square statistic and thus very small sample sizes (expected values less than 5) are still problematic. G-tests have the additional advantage that they can be used additively with more complex designs and a thus more extensible than the chi-squared statistic.

16.5 Small sample sizes

As discussed previously, both the χ^2 and G^2 statistics are poor approximations of theoretical χ^2 distributions when sample sizes are very small. Under these circumstances a number of alternative options are available:

(i) If the issue has arisen due to a large number of category levels in one or more of the variables, some categories could be combined together.

(ii) Fishers exact test[h] which essentially calculates the probability of obtaining the cell frequencies given the observed marginal totals in 2×2 tables. The calculations involved in such tests are extremely tedious as they involve calculating probabilities from hypergeometric distributions (discrete distributions describing the number of successes from sequences of samples drawn with out replacement) for all combinations of cell values that result in the given marginal totals.

(iii) Yates' continuity correction calculates the test statistic after adding and subtracting 0.5 from observed values less than and greater than expected values respectively. Yates' correction can only be applied to designs with a single degree of freedom (goodness-of-fit designs with two categories or 2×2 tables) and for goodness-of-fit tests provide p-values that are closer to those of an exact binomial. However, they typically yield over inflated p-values in contingency tables.

(iv) Williams' correction is applied by dividing the test statistic by

$$1 + (p^2 - 1)6n\upsilon$$

where p is the number of categories, n is the total sample size (total of observed frequencies) and υ is the number of degrees of freedom ($p - 1$). Williams' corrections can be applied to designs with greater than one degree of freedom, and are considered marginally more appropriate than Yates' corrections if corrections are insisted.

(v) Randomization tests in which the sample test statistic (either χ^2 or G^2) is compared to a probability distribution generated by repeatedly calculating the test statistic from an

[h] So called because as resulting p-values and assumptions are exact rather than approximated.

equivalent number of observations drawn from a population (sampling with replacement) with the specific ratio of category frequencies defined by the null hypothesis. Significance is thereafter determined by the proportion of the randomized test statistic values that are greater than or equal to the value of the statistic that is based on observed data.

(vi) Log-linear modelling (see section 16.6)

16.6 Alternatives

The χ^2 statistic has many limitations when applied to contingency table analyses (particularly concerning the testing and interpretation of interactions) and these issues are exacerbated with increasing numbers of categories and variables. Log-linear models are considered more appropriate than traditional chi-square statistics for analyzing contingency tables (particularly for multiway tables). Briefly, log-linear models (see section 17.3.2 for more complete treatment) are a form of generalized linear model in which the (natural log) expected frequencies of the category combinations (cells of the contingency table) are modelled against a combination of categorical variables around a Poisson distribution of residuals. This approach is analogous to analysis of variance, and thus, both individual and interaction effects can be estimated[i].

16.7 Power analysis

Power analyses are most usefully performed to provide an estimate of the sample size required to pick up a particular pattern (significant departure of category frequencies from the null hypothesis). Hence, in order to perform a power analysis, it is necessary to first define one or more possible patterns (effect sizes). To do so, we consider what percentage deviation from the null pattern would be considered biologically important and use these deviations to generate possible data sets that represent alternative hypotheses and thus effect sizes.

The overall effect size (w_s) is expressed as a standardized difference between the hypothetical proportions reflecting alternate and null hypotheses:

$$w_s = \sqrt{\sum \frac{(P_A - P_0)^2}{P_0}}$$

where P_A and P_0 represent the proportions expected according to the alternate and null hypotheses respectively. Note that this is just the square root of the χ^2 statistic comparing the alternate hypothesis frequencies to the null hypothesis frequencies. Note also that since mean and variance are related, power analysis calculations do not require estimates of population variation. Although power analysis is only available for χ^2 tests, since χ^2 and G-tests essentially approximate the same thing, the estimates based on χ^2 test should be equally appropriate for G-tests.

[i] Parameters are estimated by maximum likelihood and hypothesis tests are performed by comparing the fit (as measured by log-likelihood) of appropriate sets of full and reduced models.

16.8 Simple frequency analysis in R

Chi-square analysis for both goodness-of-fit and contingency analyses are accommodated by the `chisq.test()` *function*. Kolmogorov-Smirnov tests of distributional conformity are accomodated via the `ks.test()` *function*. G-tests are performed using the `g.test()`[j] *function*.

16.9 Further reading

- Theory

  Fowler, J., L. Cohen, and P. Jarvis. (1998). *Practical statistics for field biology*. John Wiley & Sons, England.

  Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

  Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

  Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

- Practical - R

  Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

  Dalgaard, P. (2002). *Introductory Statistics with R*. Springer-Verlag, New York.

  Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS*, 4th edn. Springer-Verlag, New York.

16.10 Key for Analysing frequencies

1 a. **Sampling units classified by a single category** .... Go to 2

b. **Sampling units cross classified according to multiple categories - tests of association (Contingency tables)**.. Go to 3

2 a. **Expected frequencies calculated from sample data according to a theoretical ratio (Homogeneous frequencies, chi-square test)** .... See Example 16A

```
> chisq.test(c(C1, C2, ..))
> # OR
> chisq.test(data.xtab)
```

where `C1, C2,..` *are the tabulated counts (frequencies) of each classification and* `data.xtab` *is a table of observed values.*

- To check assumption that no more than 20% of expected frequencies are less than 5, append the above function with `$res`, e.g. `chisq.test(data.xtab)$res`
- To specify an alternative ratio of expected values, use the `p=c()` argument
- To perform G-tests, use the `g.test()` function in the `biology` package ..... See Example 16B

[j] Pete Hurd provides R syntax for a version of a `g.test` on his web page http://wwwych.ualberta.ca/phurd/cruft/. I have included his function within the `biology` *package*.

b. Expected frequencies calculated from a mathematical model representing a distribution (Goodness of fit test - Kolmogorov-Smirnov test)

```
> ks.test(DV, DIST, ...)
> # OR
> ks.test(DV, "dist", ...)
> # For example
> ks.test(DV, "pnorm", mean(DV), sd(DV))
```

where DV is the name of the dependent variable. The second argument is either a numeric vector (`DIST`) representing the distribution to compare the dependent variable to, or else a character string (`"dist"`) representing the cummulative distribution function (as illustrated for a normal distribution above). The third and forth arguments in the above example provide parameters to the cummulative distribution function.

3 a. Two way contingency table . . . . Go to 4

b. Three or more way contingency table (consider GLM as an alternative) . . . . Go to Chapter 17

4 a. Check the assumption that no more than 20% of expected frequencies are less than 5. . . . . See Example 16C

```
> chisq.test(data.xtab, corr = F)$exp
```

Assumption met . . . . Go to 5a

b. Assumption not met . . . . Go to 5b

5 a. Analyse contingency table using chi-square test - all expected values greater than 5 . . . . See Example 16C

```
> chisq.test(data.xtab, corr = F)
```

- To perform G-tests, use the `g.test()` function in the `biology` package . . . . . See Example 16C

If null hypothesis is rejected

- Examine the residuals . . . . See Example 16C
  Append the above function with `$res`,
  e.g. `chisq.test(data.xtab,corr=F)$res`
- Examine odds ratios . . . . Go to 6
- To construct a summary figure . . . . Go to 7

b. Analyse contingency table using Fishers exact test

```
> fisher.test(data.xtab)
```

If null hypothesis is rejected

- Examine odds ratios . . . . Go to 6
- To construct a summary figure . . . . Go to 7

6 Calculate odds ratios . . . . See Example 16C

```
> library(biology)
> oddsratios(data.xtab)
```

7 a. Structure plot - summary figure . . . . See Example 16C

```
> library(vcd)
> strucplot(data.xtab, shade = T)
```

16.11 Worked examples of real biological data sets

Example 16A: Goodness of fit test - homogeneous frequencies test

Zar (1999) presented a dataset that depicted the classification of 250 plants into one of four categories on the basis of seed type (yellow smooth, yellow wrinkled, green smooth and green wrinkled). Zar (1999) used these data to test the null hypothesis that the sample came from a population that had a 9:3:3:1 ratio of these seed types (Example 22.2).

Step 1 - Create a dataframe with the Zar (1999) seeds data

```
> COUNT <- c(152, 39, 53, 6)
> TYPE <- gl(4, 1, 4, c("YellowSmooth", "YellowWrinkled",
+     "GreenSmooth", "GreenWrinkled"))
> seeds <- data.frame(TYPE, COUNT)
```

Step 2 - Convert the seeds dataframe into a table. Whilst this step is not strictly necessary, it does ensure that columns in various tabular outputs have meaningful names.

```
> seeds.xtab <- xtabs(COUNT ~ TYPE, seeds)
```

Step 3 (Key 16.2) - Assess the assumption of sufficient sample size ($\leq$ 20% of expected values $<$ 5) for the specified null hypothesis.

```
> chisq.test(seeds.xtab, p = c(9/16, 3/16, 3/16, 1/16),
       correct = F)$exp
  YellowSmooth YellowWrinkled    GreenSmooth  GreenWrinkled
       140.625         46.875         46.875         15.625
```

Conclusions - all expected values are greater than 5, therefore the chi-squared statistic is likely to be a reliable approximation of the χ^2 distribution.

Step 4 (Key 16.2) - Test the null hypothesis that the sample could have come from a population with a 9:3:3:1 seed type ratio. Yates' continuity correction is not required (`correct=F`).

```
> chisq.test(seeds.xtab, p = c(9/16, 3/16, 3/16, 1/16),
       correct = F)
        Chi-squared test for given probabilities


data:  seeds.xtab
X-squared = 8.9724, df = 3, p-value = 0.02966
```

Conclusions - reject the H_0. The samples are unlikely to have come from a population with a 9:3:3:1 ratio

Example 16B: G-test for goodness of fit test - homogeneous frequencies test

Smith (1939) crossed a complex combination of two varieties of beans yielding a total of 241 progeny across eight phenotypes. Mendelian theory should have resulted in phenotypic ratios of 18:6:6:2:12:4:12:4. Sokal and Rohlf (1997) used these data to test the null hypothesis that the observed frequencies could have come from a population with a 18:6:6:2:12:4:12:4 phenotypic ratio (Box 11.1).

Step 1 - Create a dataframe with the Smith (1939) beans data

```
> COUNT <- c(63, 31, 28, 12, 39, 16, 40, 12)
> PHENOTYPE <- gl(8, 1, 8, c("Pt", "Pt", "Rb", "Rt", "P", "O",
+     "B", "T"))
> beans <- data.frame(PHENOTYPE, COUNT)
```

Step 2 - Convert the beens dataframe into a table so as to allow for more meaningful output.

```
> beans.xtab <- xtabs(COUNT ~ PHENOTYPE, beans)
```

Step 3 - Define the expected probabilities based on the null hypothesis

```
> H0 <- c(18, 6, 6, 2, 12, 4, 12, 4)
> H0.prob <- H0/sum(H0)
```

Step 4 (Key 16.2) - Assess the assumption of sufficient sample size ($\leq$ 20% of expected values $<$ 5) for the specified null hypothesis.

```
> library(biology)
> g.test(beans.xtab, p = H0.prob)$exp
      Pt       Pt       Rb       Rt        P        O        B        T
67.78125 22.59375 22.59375  7.53125 45.18750 15.06250 45.18750 15.06250
```

Conclusions - all expected values are greater than 5, therefore the chi-squared and G-statistics are likely to be a reliable approximation of the χ^2 distribution. As one of the expected frequencies is close to 5 it could be argued that the G-statistic will more closely approximate the χ^2 distribution.

Step 5 (Key 16.2) - Test the null hypothesis that the sample could have come from a population with a 18:6:6:2:12:4:12:4 seed type ratio. As one of the expected values is close to 5, we will apply a Williams' correction - although this is unlikely to make much of a difference.

```
> g.test(beans.xtab, p = H0.prob, correct = "williams")
        Log likelihood ratio (G-test) goodness of fit test

data:  beans.xtab
Log likelihood ratio statistic (G) = 8.7694, X-squared df = 7,
p-value = 0.2696
```

Conclusions - do not reject the H_0. There is no evidence to suggest that the samples didn't come from a population with a 18:6:6:2:12:4:12:4 phenotypic ratio.

Example 16C: Two-way contingency table

In order to investigate the mortality of coolibah (*Eucalyptus coolibah*) trees across riparian dunes, Roberts (1993) counted the number of quadrats in which dead trees were present and the number in which they were absent in three positions (top, middle and bottom) along transects from the lakeshore up to the top of dunes. In this case, the classification of quadrats according to the presence/absence of dead coolibah trees will be interpreted as a response variable and the position along transect as a predictor variable (see Box 14.3 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Roberts (1993) data set[k].

```
> roberts <- read.table("roberts.csv", header = T, sep = ",")
```

Note that this data set contains the uncollated raw data (cross-classification of each quadrat).

Step 2 - Convert the dataframe into a collated table in preparation for contingency table analysis

```
> roberts.xtab <- table(roberts$POSITION, roberts$DEAD)
> roberts.xtab <- with(roberts, table(POSITION, DEAD))
> roberts.xtab
        DEAD
POSITION With Without
  Bottom   15      13
  Middle    4       8
  Top       0      17
```

Step 3 (Key 16.4b) - Assess the assumption of sufficient sample size ($\leq$ 20% of expected values $<$ 5) for the specified null hypothesis.

```
> chisq.test(roberts.xtab, corr = F)$exp
        DEAD
POSITION     With  Without
  Bottom 9.333333 18.66667
  Middle 4.000000  8.00000
  Top    5.666667 11.33333
```

Conclusions - only one ($1/6 = 16.67\%$) of the expected values are less than 5, therefore the χ^2 statistic should be a reasonably reliable approximation of the χ^2 distribution. Nevertheless, G-test will also be performed to confirm the outcome.

Step 4 (Key 16.5) - Test the null hypothesis that there is no association between the presence/absence of coolibah trees and position along transect.

```
> chisq.test(roberts.xtab, corr = F)
        Pearson's Chi-squared test

data:  roberts.xtab
X-squared = 13.6607, df = 2, p-value = 0.001080
> library(biology)
> g.test(roberts.xtab, corr = "williams")
```

[k] Note that for such a small dataset, it is also possible to tally the data up and enter it directly into a dataframe, however, in the interests of illustrating computer tallying, we will import the full data set containing the classification of each replicate.

```
        Log likelihood ratio (G-test) test of independence with
        Williams' correction

data:  roberts.xtab
Log likelihood ratio statistic (G) = 17.7815, X-squared df = 2,
p-value = 0.0001377
```

Conclusions - the null hypothesis of no association would be rejected via both the χ^2 test and the G-test. The mortality of coolibah trees was found to be significantly associated to position along lakeside-dune transects ($\chi^2 = 13.67$, $df = 2$, $P = 0.001$).

Step 5 (Key 16.5) - Explore the pattern of standardized residuals to reveal which cross classifications deviate greatest from the expected values and thus contribute greatest to the lack of independence between coolibah mortality and transect position.

```
> chisq.test(roberts.xtab, corr = F)$res
        DEAD
POSITION      With   Without
  Bottom  1.854852 -1.311578
  Middle  0.000000  0.000000
  Top    -2.380476  1.683251
```

Conclusions - clearly there were fewer quadrats at the bottom of the transects with dead coolibah trees (and more at the top of the transects) than would be expected if there was no association. This implies that coolibah mortality is greatest further up the dunes.

Step 6 (Key 16.6) - Explore the odds ratios to statistically compare the mortality of coolibah trees between each pairing of the transect positions. Note, we will use the modified Wald's odds ratio calculations that correct (by adding 0.5) for the impacts of observed frequencies of zero. Note also that since odds ratios can only be calculated for 2×2 tables, odds ratios must be calculated in a number of steps.

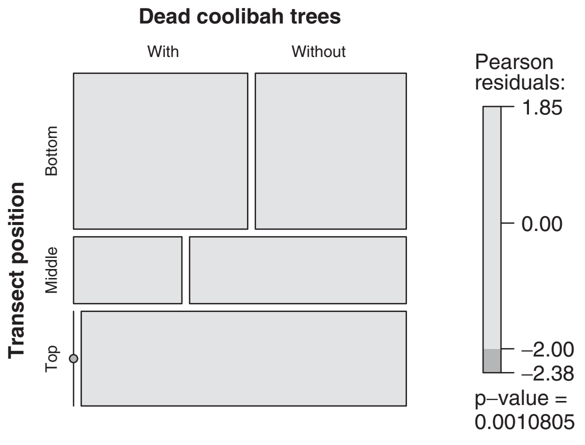
```
> library(biology)
> oddsratios(roberts.xtab + 0.5)
        Comparison  estimate     lower      upper   midp.exact
1 Bottom vs Middle  2.168724 0.5590128   8.413699 2.675536e-01
2    Bottom vs Top 40.185185 2.2016841 733.460860 6.449331e-05
3    Middle vs Top 18.529412 0.8912652 385.226628 1.806240e-02
  fisher.exact   chi.square
1 0.3147544186 0.2585776014
2 0.0000805218 0.0003656938
3 0.0180623974 0.0173950255
```

Conclusions - the odds of having dead coolibah trees is significantly higher at the top of the transect than the bottom (95% CI 2.2-733.5) or to a lesser degree, the middle (95% CI 0.9-385.2) of the transect.

Step 7 (Key 16.7) - Summarize the findings with a mosaic plot[l].

[l] Note an association plot can be produced with the `assoc()` *function* using similar syntax usin.

```
> library(vcd)
> strucplot(roberts.xtab, shade=T, labeling_args=list(
+ set_varnames=c(POSITION="Transect position",
+ DEAD="Dead coolibah trees"), offset_varnames = c(left = 1.5,
+ top=1.5)), margins=c(5,2,2,5))
```



Example 16D: Power analysis for contingency tables

In the absence of a good biological example of a power analysis for contingency tables in any of the main biostatistics texts, a fictitious example will be presented. A marine ecologist was interested in investigating whether North Stradbroke Island hermit crabs were selective in the shells they occupied (what a wet ecologist does on holidays I guess!). He intended to conduct a survey in which shells were cross-classified according to whether or not they were occupied and what type of gastropod they were from (*Austrocochlea* or *Bembicium*). Shells with living gastropods were to be ignored. Essentially, the nerd wanted to know whether or not hermit crabs occupy shells in the proportions that they are available (null hypothesis). A quick count of shells on the rocky shore revealed that approximately 30% of available gastropod shells were occupied and that there were less *Austrocochlea* shells available than *Bembicium* shells (40:60%). The ecologist scratches his sparsely haired scalp, raises one eyebrow and contemplates performing a quick power analysis to determine how many observation would be required to have an 80% chance of detecting a 20% preference for *Austrocochlea* shells.

Step 1 - Using the marginal proportions (0.7 and 0.3 for absent and occupied; 0.4 and 0.6 for *Austrocochlea* and *Bembicium*), calculate the proportions of each cross-classification for the null hypothesis (no association or selection).

```
> H0.tab <- matrix(c(0.7 * 0.4, 0.7 * 0.6, 0.3 * 0.4, 0.3 * 0.6),
+     nrow = 2)
> rownames(H0.tab) <- c("Aust", "Bemb")
> colnames(H0.tab) <- c("Empty", "Occupied")
> library(epitools)
```

```
> table.margins(H0.tab)
      Empty Occupied Total
Aust   0.28     0.12   0.4
Bemb   0.42     0.18   0.6
Total  0.70     0.30   1.0
```

Step 2 - Create proportions to represent the alternative hypothesis (20% more selective for *Austrocochlea*). Note that this does not mean that the hermit crabs are necessarily expected to occupy *Austrocochlea* 20% more than *Bembecium*, but rather that they are more selective for them.

```
> HA.tab <- matrix(c(0.7 * 0.4, 0.7 * 0.6, 0.3 * 0.5, 0.3 * 0.5),
+     nrow = 2)
> rownames(HA.tab) <- c("Aust", "Bemb")
> colnames(HA.tab) <- c("Empty", "Occupied")
> table.margins(HA.tab)
      Empty Occupied Total
Aust   0.28     0.15  0.43
Bemb   0.42     0.15  0.57
Total  0.70     0.30  1.00
```

Note from this alternate hypothesis, we expect to see hermit crabs occupying the different shells in equal proportion, despite *Austrocochlea* shells being less available.

Step 3 - Calculate the effect size corresponding to hermit crabs being 20% more selective for *Austrocochlea* shells.

```
> ws <- sqrt(chisq.test(as.vector(HA.tab),
  p = as.vector(H0.tab))$stat[[1]])
```

Step 4 - Calculate the approximate sample size required to have an 80% change of detecting such an association between shell type and occupancy.

```
> library(pwr)
> pwr.chisq.test(df = 1, w = ws, power = 0.8)
     Chi squared power calculation

              w = 0.1118034
              N = 627.9088
             df = 1
      sig.level = 0.05
          power = 0.8

 NOTE: N is the number of observations
```

Conclusions - The ecologist would need to contemplate sampling at least 628 shells in order to be confident of detecting a 20% greater selectivity of hermit crabs for *Austrocochlea* shells. Holidays don't get any better than that!

17

Generalized linear models (GLM)

General linear models (Chapters 8-15) provide a set of well adopted and recognised procedures for relating response variables to a linear combination of one or more predictors. Nevertheless, the reliability and applicability of such models are restricted by the degree to which the residuals conform to normality and the mean and variance are independent of one another. There are many real situations for which those assumptions are unlikely to be satisfied. For example, if the measured response to a predictor treatment (such as nest parasite load) can only be binary (such as abandoned or not), then the differences between the observed and expected values (residuals) are unlikely to follow a normal distribution. Instead, in this case, they will follow a binomial distribution. Furthermore, the variance will likely be tied to the mean in that the higher the expected probability of an event, the greater the variability in this probability.

Transformations to normalize the residuals and stabilize variances are useful in many instances (as demonstrated in numerous examples in previous chapters). However, the biological interpretations of models and parameters can be greatly complicated by scale alterations and scale transformations are not always successful. For example, response variables that represent counts (e.g. the number of individuals of a species per quadrat), are often highly skewed and contain an abundance of zeros. Thus, linear models based on transformed data in such situations can be unsuitable.

Generalized linear models (GLM's) extend the application range of linear modelling by accommodating non-stable variances as well as alternative exponential[a] residual distributions (such as the binomial and Poisson distributions). Generalized linear models have three components:

(i) The random component that specifies the conditional distribution of the response variable. Such distributions are characterised by some function of the mean (canonical or location parameter) and a function of the variance (dispersion parameter). Note that for binomial and Poisson distributions, the dispersion parameter is 1, whereas for the

[a] The exponential distributions are a class of continuous distribution which can be characterized by two parameters. One of these parameters (the location parameter) is a function of the mean and the other (the *dispersion* parameter) is a function of the variance of the distribution. Note that recent developments have further extended generalized linear models to accommodate other non-exponential residual distributions.

Biostatistical Design and Analysis Using R: a Practical Guide, 1st edition. By M. Logan.
Published 2010 by Blackwell Publishing.

Table 17.1 Common generalized linear models and associated canonical link-distribution pairs.

Model	Response variable	Predictor variable(s)	Residual distribution	Link
Linear regression[a]	Continuous	Continuous/ Categorical	Gaussian (normal)	Identity $g(\mu) = \mu$
Logistic regression	Binary	Continuous/ Categorical	Binomial	Logit $g(\mu) = log_e \frac{\mu}{1-\mu}$
Log-linear models	Counts	Categorical	Poisson	Log $g(\mu) = log_e \mu$

[a]Includes the standard ANOVA and ANCOVA designs.

Guassian (normal) distribution the dispersion parameter is the error variance and is assumed to be independent of the mean.

(ii) The systematic component that represents the linear combination of predictors (which can be categorical, continuous, polynomial or other contrasts). This is identical to that of general linear models.

(iii) The link function which links the expected values of the response (random component) to the linear combination of predictors (systematic component). The generalized linear model can thus be represented as:

$$g(\mu) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots$$

where $g(\mu)$ represents the link function and β_0, β_1 and β_2 represent parameters broadly analogous to those of general linear models. Although there are many commonly employed link functions, typically the exact form of the link function depends on the nature of the random response distribution. Some of the canonical (natural) link function and distribution pairings that are suitable for different forms of generalized linear models are listed in Table 17.1.

The *generalized* nature of GLM's makes them incompatible with ordinary least squares model fitting procedures. Instead, parameter estimates and model fitting are typically achieved by maximum likelihood[b] methods based on an iterative re-weighting algorithm (such as the Newton-Raphson algorithm). Essentially, the Newton-Raphson algorithm (also known as a scoring algorithm) fits a linear model to an adjusted response variable (transformed via the link function) using a set of weights and then iteratively re-fits the model with new sets of weights recalculated according to the fit of the previous iteration. For canonical link-distribution pairs (see Table 17.1), the Newton-Raphson algorithm usually converges (arrives at a common outcome or equilibrium) very efficiently and reliably.

The Newton-Raphson algorithm facilitates a unifying model fitting procedure across the family of exponential probability distributions thereby providing a means by which binary and count data can be incorporated into the suit of linear model designs

[b] Recall that maximum likelihood estimates are those maximize the likelihood of obtaining the actual observations for the chosen model.

described in chapters 8-15. In fact, linear regression (including ANOVA, ANCOVA and other general linear models) can be considered a special form of GLM that features a normal distribution and identity link function and for which the maximum likelihood procedure has an exact solution. Notably, when variance is stable, both maximum likelihood and ordinary least squares yield very similar parameter estimates.

17.1 Dispersion (over or under)

The variance of binomial or Poisson distributions is assumed to be related to the sample size and mean respectively, and thus, there is not a variance parameter in their definitions. In fact, the variance (or dispersion) parameter is fixed to 1. As a result, logistic regression and log-linear modelling assume that sample variances conform to the respective distribution definitions. However, it is common for individual sampling units (e.g. individuals) to co-vary such that other, unmeasured influences, increase (or less commonly, decrease) variability. For example, although a population sex ratio might be 1:1, male to female ratios within a clutch might be highly skewed towards one or other sex. Positive correlations cause greater variance (overdispersion) and result in deflated standard errors (and thus exaggerated levels of precision and higher Type I errors). Methods of diagnosing and modelling over-dispersed data are described in section 17.4.

17.2 Binary data - logistic (logit) regression

Logistic regression is a form of GLM that employs the logit-binomial link distribution canonical pairing to model the effects of one or more continuous or categorical (with dummy coding) predictor variables on a binary (dead/alive, presence/absence, etc) response variable. For example, we could investigate the relationship between salinity levels (salt concentration) and mortality of frogs. Similarly, we could model the presence of a species of bird as a function of habitat patch size, or nest predation (predated or not) as a function of the distance from vegetative cover.

17.2.1 Logistic model

Consider the fictitious data presented in Figure 17.1a&b. Clearly, a regular simple linear model (straight line, Figure 17.1a) is inappropriate for modelling the probability of presence. Note that at very low and high levels of X, the predicted probabilities (probabilities or proportions of the population) are less than zero and greater than one respectively - logically impossible outcomes.

The logistic model (Figure 17.1c) relating the probability ($\pi(x)$) that the response (y_i) equals one (present) for a given level of x_i (patch size) is defined as:

$$\pi(x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}$$

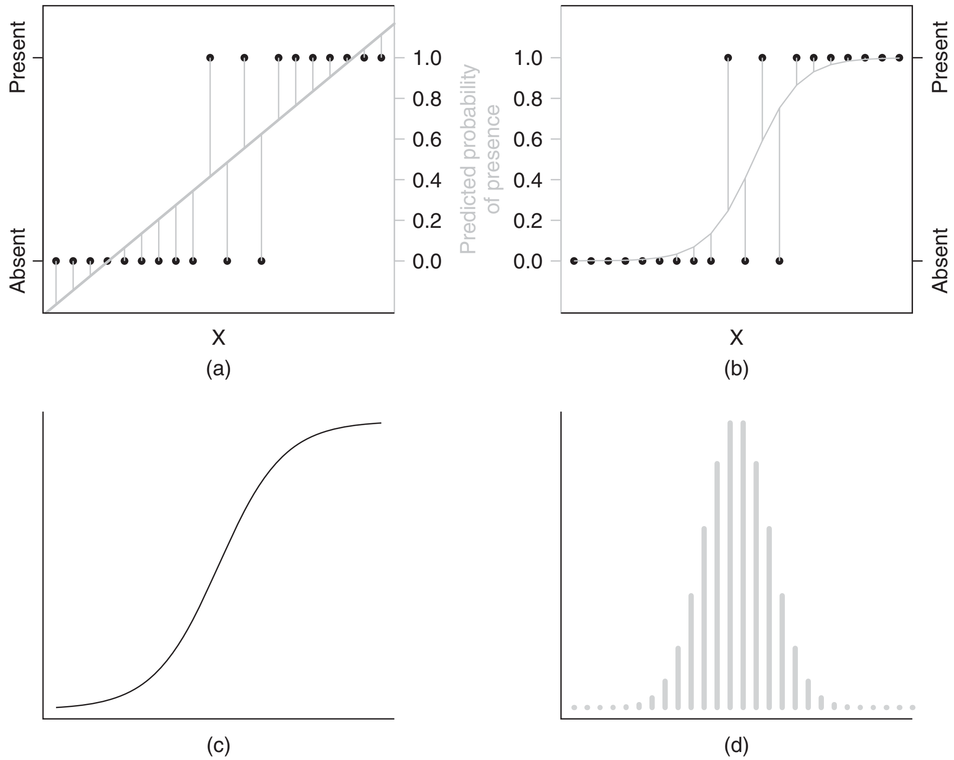


Fig 17.1 Fictitious data illustrating a binary response variable modelled with (a) a linear model and (b) an equivalent logistic regression model. Not only does the linear model violate linearity and normality, the predicted values are not bounded by the logical probability limits of 0 and 1. Accordingly, the inappropriately fitted linear model (a) implies that at very low levels of X, individuals are expected to be less than absent! Subfigures (c) and (d) represent the general logistic model and binomial probability distribution respectively.

Appropriately, since $e^{\beta_0+\beta_1 x}$ (the "natural constant" raised to a simple linear model) must evaluate to between 0 and infinity, the logistic model must asymptote towards (and is thus bounded by) zero and one. Alternatively, the logit link function:

$$\ln\left(\frac{\pi(x)}{1-\pi(x)}\right)$$

can be used to transform $\pi(x)$ such that the logistic model is expressed as the log odds (probability of one state relative to the alternative) against a familiar linear combination of the explanatory variables (as is linear regression).

$$\ln\left(\frac{\pi(x)}{1-\pi(x)}\right) = \beta_0 + \beta_1 x_i$$

Although the β_0 (y-intercept) parameter is interpreted similar to that of linear regression (albeit of little biological interest), this is not the case for the slope parameter (β_1).

Rather than representing the rate of change in the response for a given change in the predictor, in logistic regression, β_1 represents the rate of change in the odds ratio (ratio of odds of an event at two different levels of a predictor) for a given unit change in the predictor. The exponentiated slope represents the odds ratio, the proportional rate at which the predicted odds change for a given unit change of the predictor.

$$odds\ ratio = e^{\beta_1}$$

17.2.2 Null hypotheses

As with linear regression, a separate H_0 is tested for each of the estimated model parameters:

$$H_0: \beta_1 = 0 \qquad \text{(the population slope equals zero)}$$

This test examines whether the log odds of an occurrence are independent of the predictor variable and thus whether or not there is likely to be a relationship between the response and predictor.

$$H_0: \beta_0 = 0 \qquad \text{(the population y-intercept equals zero)}$$

As stated previously, this is typically of little biological interest.

Similar to linear regression, there are two ways of testing the main null hypotheses[c]

(i) Parameter estimation approach. Maximum likelihood estimates of the parameters and their asymptotic[d] standard errors (S_{b_1}) are used to calculate the Wald t (or t ratio) statistic:

$$W = \frac{b_1}{S_{b_1}}$$

which approximately follows a standard z distribution when the null hypothesis is true. The reliability of Wald tests deminishes substantially with small sample sizes. For such cases, the second option is therefore more appropriate.

(ii) (log)-likelihood ratio tests approach. This approach essentially involves comparing the fit of models with (full) and without (reduced) the term of interest:

$$logit(\pi) = \beta_0 + \beta_1 X_1 \quad \text{(Full model)}$$
$$logit(\pi) = \beta_0 \quad \text{(Reduced model)}$$

The fit of any given model is measured via log-likelihood and the differences between the fit of two models is described by a likelihood ratio statistic ($G^2 = 2$(log-likelihood reduced

---

[c] Note, that whilst in simple regression, the parameter and model comparison approaches yield identical outcomes, this is not the case and that the degree of correspondence depends on sample sizes. For small sample sizes, the model comparisons approach is considered more reliable.

[d] A parameter is refered to as an assymptotic estimate if their reliability is sample size dependent - they become progressively more accurate with increasing sample size, albeit with diminishing returns.

model - log-likelihood full model)). The G^2 quantity is also known as deviance and is analygous to the residual sums of squares in a linear model. When the null hypothesis is true, the G^2 statistic approximately follows a χ^2 distribution with one degree of freedom. An analogue of the linear model r^2 measure can be calculated as:

$$r^2 = 1 - \frac{G_0^2}{G_1^2}$$

where G_0^2 and G_1^2 represent the deviances due to the intercept and slope terms respectively.

17.2.3 Analysis of deviance

Analogous to the ANOVA table that partitions the total variation into components explained by each of the model terms (and the unexplained error), it is possible to construct a analysis of deviance table that partitions the deviance into components explained by each of the model terms.

17.2.4 Multiple logistic regression

Multiple logistic regression is an extension of logistic regression in the same way that multiple linear regression is an extension of simple linear regression.

$$logit(\pi) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots$$

Each of the parameters can be estimated either via Wald statistics or via a sequence of log-likelihood (G^2) tests comparing models with and without each predictor term. These estimatated parameters are partial logistic regression parameters. That is, they are the effect of one predictor on the odds of an occurance holding all other predictors constant.

Since the systematic component of GLM's is identical to that of linear models, multiple logistic regression shares the issues and diagnoses concerning (multi)collinearity[e] with multiple linear regression.

Model selection and model averaging

Selecting the best (most parsimonious) model as well as assessing the relative importance of each of the predictor variables follows similar procedures to those outlined in sections 9.7 & 9.7.1 and can be based on the following measures (see Table 9.2 for more formula and R syntax):

- Differences in deviance (G^2) between model pairs
- r^2 - analogous to multiple linear regression
- *AIC* (preferred). The Akaike Information Criterion (AIC) for generalized linear models is the deviance (G^2) penalized for the number of predictors (p) and either the number of

[e] Recall from chapter 9 that the assumption of multicollinearity concerns the issues that arise when two or more of the predictor variables are correlated to one another.

observations (n) or unique category combinations (D):

$$AIC = G^2 - n + 2p$$
$$AIC = G^2 - D + 2p$$

When comparing two possible models from a family of models, this is reduced to:

$$AIC = G^2 - 2df$$

where df is the difference in degrees of freedom of the two models. Models with the smallest AIC are the most parsimonious.

- $QAIC$. The Quasi Akaike Information Criterion is adjusted for the degree of overdispersion of lack of fit
- AIC_C and $QAIC_C$. Both AIC and $QAIC$ also have versions that correct for small ($n < 30$) sample sizes. Model selection should be based upon models fitted using maximum likelihood (ML) rather than restricted maximum likelihood (REML) as the former is more appropriate for comparing models with different fixed and random effects structures. The resulting 'best' model should then be refit using REML.

17.3 Count data - Poisson generalized linear models

Another form of data for which scale transformations are often unsuitable or unsuccessful are count data. Count data tend to follow a Poisson distribution (see Figure 16.1) and consequently, the mean and variance are usually related. Generalized linear models provide appropriate means to model count data according to two design contexts:

(i) as an alternative to linear regression for modeling count data against a linear combination of continuous and/or categorical predictor variables (**Poisson regresssion**)
(ii) as an alternative to contingency tables in which the associations between categorical variables are explored (**log-linear modelling**)

17.3.1 Poisson regression

The Poisson regression model is:

$$log(\mu) = \beta_0 + \beta_1 x_1$$

where $log(\mu)$ is the link function used to link the mean of the Poisson response variable to the linear combination of predictor variables. Poisson regression otherwise shares null hypotheses, parameter estimation, model fitting and selection with logistic regression (see section 17.2).

17.3.2 Log-linear Modelling

Contingency tables were introduced in section 16.3 along with caveats regarding the reliability and interoperability of such analyses (particularly when expected proportions are small or for multiway tables). In contrast to logistic and Poisson regression, all variables in a log-linear model do not empirically distinguish between response and predictor variables. Nevertheless, as in contingency tables, causality can be implied when logical and justified by interpretation.

Log-linear models

The saturated (or full) log-linear model resembles a multiway ANOVA model (see chapter 12). The full and reduced log-linear models for a two factor design are:

$$log(f_{ij}) = \mu + \lambda_i^A + \lambda_j^B + \lambda_{ij}^{AB} \text{ (full)}$$

$$log(f_{ij}) = \mu + \lambda_i^A + \lambda_j^B \text{ (reduced)}$$

where $log(f_{ij})$ is the log link function, μ is the mean of the (log) of expected frequencies (f_{ij}) and λ_i^A is the effect of the ith category of the variable (A), λ_j^B is the effect of the jth category of B and λ_{ij}^{AB} is the interactive effect of each category combination on the (log) expected frequencies.

Reduced models differ from full models in the absence of all higher order[f] interaction terms. Comparing the fit of full and reduced models therefore provides a means of assessing the effect of the interaction. Whilst two-way tables contain only a single interaction term (and thus a single full and reduced model), multiway tables have multiple interactions. For example, a three-way table has a three way interaction (ABC) as well as three two-way interactions (AB, AC, BC). Consequently, there are numerous full and reduced models, each appropriate for different interaction terms (see Table 17.2).

Null hypotheses

Consistent with contingency table analysis, log-linear models test the null hypothesis ($\mathbf{H}_0$) that the categorical variables are independent of (not associated with) one another. Such null hypotheses are tested by comparing the fit (deviance, G^2, see section 17.2.2) of full and reduced models. The G^2 is compared to a χ^2 distribution with degrees of freedom equal to the difference in degrees of freedom of the full and reduced models. Thereafter, odds ratios are useful for interpreting any lack of independence.

For multi-way tables, there are multiple full and reduced models.

Complete dependence:

H_0: $ABC = 0$. No three way interaction. Either no association (conditional independence) between each pair of variables, or else the patterns of associations (conditional dependencies) are the same for each level of the third. If this null hypothesis is rejected ($ABC \neq 0$), the causes of lack of independence can be explored by examining the residuals or odds ratios. Alternatively, main effects tests (testing the effects of two-way interactions separately

[f] In this context, higher order refers to interaction terms containing the term of interest as well as other factors/interactions.

Table 17.2 Full and reduced log-linear models for three-way tables in hierarchical order.

	H_0	Log-linear model	df	G^2 (reduced-full)
Saturated model				
1		$A+B+C+AB+$ $AC+BC+ABC$	0	
Complete dependence				
2	$ABC=0$	$A+B+C+$ $AB+AC+BC$	$(I-1)(J-1)$ $(K-1)$	2-1
Conditional independence				
3	$AB=0$	$A+B+C+AC+BC$	$K(I-1)(J-1)$	3-2
4	$AC=0$	$A+B+C+AB+BC$	$J(I-1)(K-1)$	4-2
5	$BC=0$	$A+B+C+AB+AC$	$I(J-1)(K-1)$	5-2
Conditional independence				
6	$AB=0$	$A+B$	(I-1)(J-1)	6-$(A+B+AB)$
7	$AC=0$	$A+C$	(I-1)(K-1)	7-$(A+C+AC)$
8	$BC=0$	$B+C$	(J-1)(K-1)	8-$(B+C+BC)$
Complete independence				
9	$AB=AC=BC=0$	$A+B+C$		9-2

at each level of the third) can be performed. If the three-way interaction is not rejected (no three-way association), lower order interactions can be explored.

Conditional independence/dependence:

If the three-way interaction is not rejected (no three-way association), lower order interactions can be explored.

H_0: $AB=0$. A and B conditionally independent (not associated) within each level of C.

H_0: $AC=0$. A and C conditionally independent (not associated) within each level of B.

H_0: $BC=0$. B and C conditionally independent (not associated) within each level of A.

Marginal independence:

H_0: $AB=0$. No association between A and B pooling over C

H_0: $AC=0$. No association between A and C pooling over B

H_0: $BC=0$. No association between B and C pooling over A

Complete independence:

If none of the two-way interactions are rejected (no two-way associations), complete independence (all two-way interactions equal zero) can be explored.

H_0: $AB=AC=BC=0$. Each of the variables are completely independent of all the other variables.

Analysis of designs with more than three factors proceed similarly, starting with tests of higher order interactions and progressing to lower order interactions only in the absence of higher order interactions. Selection of the “best” (most parsimonious) model is on the basis of the smallest G^2 or AIC where:

$$AIC = G^2 - 2df$$

17.4 Assumptions

Compared to general linear models, the requirements of generalized linear models are less stringent. In particular, neither normality nor homoscedasticity are assumed. Nevertheless, to maximize the reliability of null hypotheses tests, the following assumptions do apply:

(i) all observations should be **independent** to ensure that the samples provide an unbiased estimate of the intended population.

(ii) it is important to establish that no observations are overly influential. Most linear model **influence** (and outlier) diagnostics extend to generalized linear models and are taken from the final iteration of the weighted least squares algorithm. Useful diagnoses include:

- (a) Residuals - there are numerous forms of residuals that have been defined for generalized linear models, each essentially being a variant on the difference between observed and predicted (influence in y-space) theme. Note that the residuals from logistic regression are difficult to interpret.
- (b) Leverage - a measure of outlyingness and influence in x-space.
- (c) *Dfbeta* - an analogue of Cook’s D statisic which provides a standardized measure of the overall influence of observations on the parameter estimates and model fit.

(iii) although **linearity** between the response and predictors is not assumed, the relationship between each of the predictors and the link function is assumed to be linear. This linearity can be examined via the following:

- (a) goodness-of-fit. For log-linear models, χ^2 contingency tables (see chapter 16) can be performed[g], however due to the low reliability of such tests with small sample sizes, this is not an option for logistic regression with continuous predictor(s) (since each combination is typically unique and thus the expected values are always 1).
- (b) Hosmer-Lemeshow ($\hat{C}$). Data are aggregated into 10 groups or bins (either by cutting the data according to the predictor range or equal frequencies in each group) such that goodness-of-fit test is more reliable. Nevertheless, the Hosmer-Lemeshow statistic has low power and relies on the somewhat arbitrary bin sizes.
- (c) le Cessie-van Houwelingen-Copas omnibus test. This is a goodness-of-fit test for binary data based on the smoothing of residuals.
- (d) component+residual (partial residual) plots. Non-linearity is diagnosed as a substantial deviation from a linear trend.

Non-linearity can be dealt with either by transformation or generalized additive modelling (GAM, see section 17.5) depending on the degree and nature of the non-linearity.

(iv) **(over or under) dispersion (see section 17.1)**. The dispersion parameter (degree of variance inflation or over-dispersion) can be estimated[h] by dividing the Pearsons

[g] This is really examining whether the data could have come from a population that displays that specific fitted logistic model

[h] Overdispersion can also be diagnosed graphically from deviations on a q-q plot.

χ^2 by the degrees of freedom ($n - p$, where n is the number of observations in p parameters). As a general rule, dispersion parameters approaching 2 (or 0.5) indicate possible violations of this assumption. Where over (or under) dispersion is suspected to be an issue, **quasibinomial** and **quasipoisson** families can be used as alternatives to model the dispersion. These *quasi-likelihood* models derive the dispersion parameter (function of the variance) from the observed data. Test statistics from such models should be based on *F*-tests rather than chi-squared tests.

17.5 Generalized additive models (GAM's) - non-parametric GLM

Generalized additive models[i] are non-parametric alternatives to generalized linear models and are useful when the relationships are expected to be complex (not simple linear trends). In generalized additive models, the slope coefficients are replaced by smoothing functions;

$$g(\mu) = \beta_0 + f_1 x_{i1} + f_2 x_{i2} + \ldots$$

where f_1 and f_2 are non-parametric smoothing functions. The weighted smoothing functions permit trends to deviate at critical regions throughout the data cloud (see Figure 17.2) and thus the resulting smoother estimates tend to be less variable (or smoother) than the corresponding regression coefficients. Generalized additive models are fitted using a modification of the Newton-Raphson scoring algorithm in which the partial residuals are iteratively smoothed in a process known as backfitting.

Common smoothers include cubic splines and Loess smoothers (as well as running means, running medians, running lines, and kernel smoothers). Selection of the appropriate smoother(s) and smoothing coefficients usually follows scatterplot examination. Note that it is possible to apply different smoothers and smoothing coefficients for each of the predictor variables.

GAM's potentially model the nature of the data trends more "truly" and yield better fits in the presence of non-linear trends. However, they are substantially more complex to fit than GLM's, requiring consideration of not only the appropriate distribution and linkage function, but also the appropriate smoothers and smoothing coefficients. GAM's must also be fitted judiciously to avoid over-fitting[j]. GAM's can also be more difficult to interpret than GLM's, particularly with respect predictions. The principles of parsimony should be applied by verifying the fit of the GAM against the equivalent GLM.

Early methodologies extended the application of smoothing and local regression (as described in section 8.3) to generalized linear models. More recent developments in

[i] GAM's are a form of additive model. Additive models fit each of the model terms additively. That is, there are no interactions in the model.

[j] Over-fitting occurs when overly complex models are fitted to data. This is analogous to fitting a very high order polynomial to a data cloud. Whilst the model fits the observed data well, it does not reflect the true nature of the relationship.

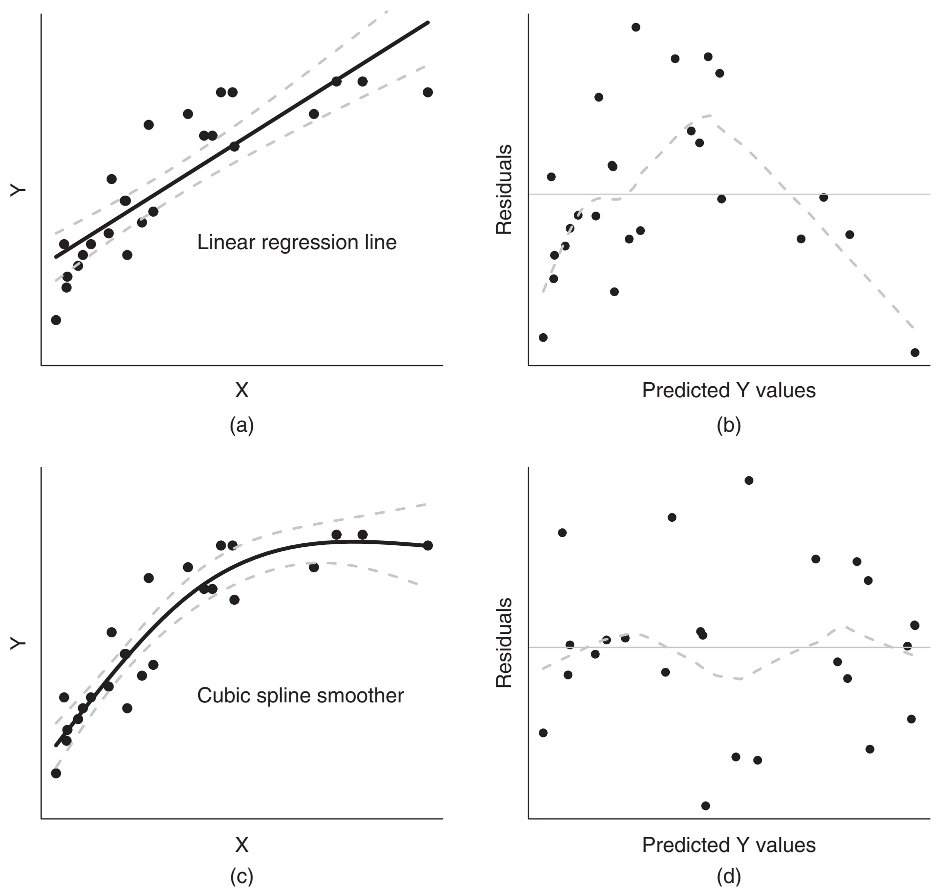


Fig 17.2 Fictitious relationship between Y and X contrasting the fit of (a) linear and (c) loess smoothers as well as the corresponding residual plots (b) and (d) respectively. The cubic spline fits the data substantially better than the fitted linear regression line.

the field of GAM's have expanded their capabilities to provide more sophisticated optimization of smoothing as well as accommodating mixed effects modelling approaches to hierarchical designs and correlations structures.

Clearly this has been a very brief and non-technical description of GAM's and is intended as an introduction to the existence of additional non-parametric alternatives.

17.6 GLM and R

Generalized linear models can be fit using the `glm()` *function* with the *family* parameter to specify the random component. The optional `link` parameter can be used to specify non-canonical link functions, otherwise the link function will be determined as appropriate for the specified `family`. Full and reduced models can be compared using the `anova`[k]. GAM's are supported by two packages, `gam` and `mgcv`, reflecting the simple and more modern generalized additive modelling techniques respectively.

17.7 Further reading

- Theory

  Hastie, T. J., and R. J. Tibshirani. (1990). *Generalized Additive Models.* Chapman & Hall.

  Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

  Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC.

- Practical - R

  Crawley, M. J. (2007). *The R Book*. John Wiley, New York.

  Faraway, J. J. (2006). *Extending Linear Models with R: generalized linear mixed effects and nonparametric regression models.* Chapman & Hall/CRC.

  Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS*, 4th edn. Springer-Verlag, New York.

  Zuur, A. F., E. N. Ieno, N. J. Walker, A. A. Saveliev, and G. M. Smith. (2009). *Mixed Effects Models and Extensions in Ecology with R*. Springer.

17.8 Key for GLM

1 a. Binary response variable (logistic regression) . . . . Go to 2

b. Count (frequency) data (Poisson generalized linear models) . . . . Go to 5

2 a. Logistic regression - single predictor variable . . . . See Example 17A

```
> data.glm <- glm(DV ~ IV, dataset, family = "Poisson")
```

- **Check that the model adheres to the assumptions** . . . . Go to 3
- **To examine (over) dispersion** . . . . Go to 4
- **To get the model parameter estimates**

```
> summary(data.glm)
```

- **To get the deviance table**

```
> anova(data.glm, test = "Chisq")
```

- **Examine the odds ratios** . . . . Go to 6

[k] Alternatively, the `Anova` *function* can be used to support Type II and Type III analogues when designs are not balanced.

b. Multiple predictor variables - multiple logistic regression .... See Examples 17B & 17C

```
> data.glm <- glm(DV ~ IV1 + IV2 + ..., dataset,
+     family = "Poisson")
```

- **Check for issues with (multi) collinearity** .... See Chapter 9
- **Check that the model adheres to the assumptions** .... Go to 3
- **To examine (over) dispersion** .... Go to 4
- **To get the model parameter estimates**

```
> summary(data.glm)
```

OR

```
> anova(data.glm, data.glmR, test = "Chisq")
```

where `data.glmR` *is a reduced model containing constructed by ommitting the term of interest.*
- **Examine the odds ratios** .... Go to 6
- **Perform model selection and model averaging** .... Go to 8

3 a. **Check the assumptions** .... See Examples 17A–17C. *In the following,* `data.glm` *is the fitted generalized linear model.*

- **Lack of fit**
  - le Cessie-van Houwelingen normal test statistic

```
> library(Design)
> data.lrm <- lrm(formula, dataset, y = T, x = T)
> resid(data.lrm)
```

  where `formula` *is a formula relating the response variable to the linear combination of predictor variables*
  - *Pearson* χ^2

```
> pp <- sum(resid(data.lrm, type = "pearson")^2)
> 1 - pchisq(pp, data.glm$df.resid)
```

  - *Deviance* (G^2)

```
> 1 - pchisq(data.glm, data.glm$df.resid)
```

- **Linear relationship between predictors and link function (component+residual plot)**

```
> library(car)
> cr.plots(data.glm, ask = F)
```

- **Influence**

```
> influence.measures(data.glm)
```

Assumptions met .... Go back

b. Assumptions not met - Transformations of the predictor variable scale can be useful in improving linearity, otherwise consider GAM's (Go to 7)

4 a. Examine (over) dispersion . . . . See Examples 17A – 17C

- **Pearson's residuals**

```
> sum(resid(data.glm, type = "pearson")^2)/data.glm$df.resid
```

- **Deviance**

```
> data.glm$deviance/data.glm$df.resid
```

Dispersion does not deviate substantially from 1 . . . . Go back

b. Model is over dispersed

- **Refit model with "quasi" distribution**

```
> data.glm <- glm(DV ~ IV, dataset, family = "quasibinomial")
> anova(data.glm, test = "F")
```

- **Consider a negative binomial**

```
> data.glm <- glm.nb(DV ~ IV, dataset)
> anova(data.glm, test = "F")
```

5 a. Continuous predictor variable(s) (Poisson regression)

```
> data.glm <- glm(DV ~ IV1 + ..., dataset, family = "poisson")
```

- **Check that the model adheres to the assumptions** . . . . Go to 3
- **To examine (over) dispersion** . . . . Go to 4
- **To get the model parameter estimates**

```
> summary(data.glm)
```

OR

```
> anova(data.glm, data.glmR, test = "Chisq")
```

where `data.glmR` *is a reduced model containing constructed by ommitting the term of interest.*

- ***To calculate the odds ratios*** . . . . *Go to 6*
- ***To perform model selection and model averaging*** . . . . *Go to 8*

b. Categorical variables only (log-linear modelling) . . . . See Examples 17D & 17E

```
> data.glm <- glm(DV ~ CAT1 * CAT2 * ..., dataset,
+     family = "poisson")
```

- **To examine conditional independence**

```
> data.glm1 <- update(data.glm, ~. - CAT1:CAT2, dataset)
> anova(data.glm, data.glm1, test = "Chisq")
```

See Table 17.2 for appropriate full and reduced log-linear models for examining complete and conditional dependence and independence

- **To calculate odds ratios** . . . . Go to 6
- **To perform model selection** . . . . Go to 8

6 a. Calculate odds ratios . . . . See Examples 17A – 17E

```
> library(biology)
> odds.ratio(data.glm)
```

7 a. Generalized additive models . . . . See Example 17F

```
> library(gam)
> data.gam <- gam(DV ~ lo(CAT1) + lo(CAT2) + ...,
+     family = "gaussian", dataset)
```

- **The `family=` parameter can be used to specify the appropriate error distribution**
- **To check that the model adheres to the assumptions**
- **To examine the parameter estimates**

```
> sumamry(data.gam)
```

- **To perform model selection** . . . . Go to 8

8 a. Perform model selection . . . . See Examples 17B, 17C & 17F

In the following model *is the fitted model from either glm or gam*

```
> library(MuMIn)
> dredge(data.glm)
> model.avg(get.models(dredge(model)))
```

OR

```
> library(biology)
> Model.selection.glm(model)
```

17.9 Worked examples of real biological data sets

Example 17A: Logistic regression

As part of an investigation into the factors controlling island spider populations, Polis et al. (1998) recorded the physical and biotic characteristics of the islands in the Gulf of California. Quinn and Keough (2002) subsequently modelled the presence/absence (PA) of a potential spider predator (*Uta* lizards) against the perimeter to area ratio (RATIO) of the islands to illustrate logistic regression (from Box 13.1 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Polis et al. (1998) data set

```
> polis <- read.table("polis.csv", header = T, sep = ",")
```

Step 2 (Key 17.2) - Fit the logistic regression model relating the log odds of *Uta* presence against perimeter to area ratio $\left(ln\left(\frac{\pi(\mu)}{1-\pi(\mu)}\right) = \beta_0 + \beta_1(P/A\ ratio)\right)$

```
> polis.glm <- glm(PA ~ RATIO, family = binomial, data = polis)
```

Step 3 (Key 17.3) - Check the (lack of) fit and appropriateness of the model with goodness-of-fit tests

- le Cessie-van Houwelingen normal test statistic

```
> library(Design)

> polis.lrm <- lrm(PA ~ RATIO, data = polis, y = T, x = T)
> resid(polis.lrm, type = "gof")
```

```
Sum of squared errors     Expected value|H0               SD
            2.2784683             2.2633569        0.1462507
                    Z                     P
            0.1033257             0.9177045
```

- Pearson χ^2 p-value

```
> pp <- sum(resid(polis.lrm, type = "pearson")^2)
> 1 - pchisq(pp, polis.glm$df.resid)
[1] 0.5715331
```

- Deviance (G^2) significance

```
> 1 - pchisq(polis.glm$deviance, polis.glm$df.resid)
[1] 0.6514215
```

- Estimated dispersion parameter **(Key 17.4)**

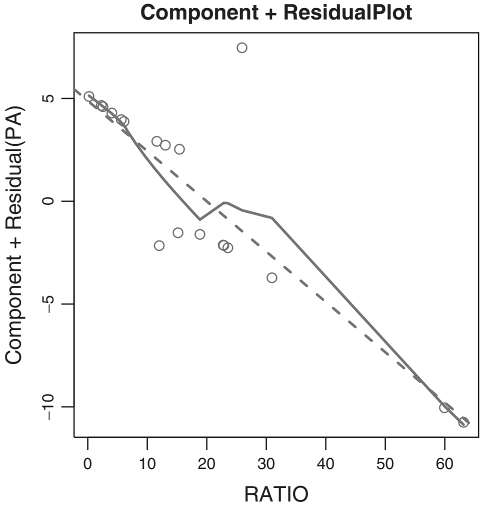
```
> pp/polis.glm$df.resid
[1] 0.901922
```

Conclusions - no evidence for a lack of fit or over-dispersion (value not approaching 2) in the model.

Step 4 (Key 17.3) - Confirm linearity between the log odds ratio of lizard presence and perimeter to area ratio with a component+residual plot

```
> library(car)
> cr.plots(polis.glm, ask = F)
```

Conclusions - no evidence of non-linearity. Thus no evidence to suggest that data did not come from population that follows the logistic regression- not evidence for a lack of fit of the model



Step 5 (Key 17.3) - Examine the influence measures

```
> influence.measures(polis.glm)
Influence measures of
        glm(formula = PA ~ RATIO, family = binomial, data = polis) :
```

```
      dfb.1_  dfb.RATI     dffit cov.r   cook.d      hat inf
1   0.182077 -0.007083  0.447814 1.043 5.50e-02 0.109124
2   0.167005 -0.141263  0.169959 1.235 6.62e-03 0.111730
3  -0.723849  1.079157  1.278634 0.537 8.43e-01 0.151047   *
4  -0.239967  0.028419 -0.546081 0.953 9.01e-02 0.108681
5   0.248270 -0.126175  0.359999 1.117 3.30e-02 0.110025
6   0.028088 -0.196986 -0.437403 1.110 5.00e-02 0.129177
7   0.077131 -0.102575 -0.111591 1.250 2.81e-03 0.108288
8   0.140334 -0.247315 -0.332565 1.242 2.65e-02 0.155414
9  -0.562402  0.338850 -0.723598 0.805 1.89e-01 0.112842
10  0.257651 -0.162838  0.319655 1.157 2.52e-02 0.114067
11  0.176591 -0.147771  0.180516 1.234 7.49e-03 0.113765
12  0.104228 -0.093408  0.104419 1.225 2.46e-03 0.090774
13  0.135395 -0.118138  0.136380 1.233 4.23e-03 0.102909
14  0.000410 -0.000476 -0.000481 1.131 5.14e-08 0.001445
15  0.000218 -0.000251 -0.000254 1.130 1.43e-08 0.000817
16  0.139447 -0.248090 -0.335881 1.239 2.70e-02 0.155114
17  0.143708 -0.240774 -0.311977 1.255 2.31e-02 0.156543
18  0.074831 -0.068694  0.074832 1.211 1.26e-03 0.075520
19  0.108633 -0.097001  0.108890 1.226 2.68e-03 0.092718
```

Conclusions - Although the Dfbeta (Cook's D equivalent) values of islands 3 (Cerraja) and 9 (Mitlan) where elevated relative to the other islands, no observations are considered overly influential.

Step 6 (Key 17.2) - Examine the parameter estimates from the fitted logistic regression model.

```
> summary(polis.glm)
Call:
glm(formula = PA ~ RATIO, family = binomial, data = polis)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6067  -0.6382   0.2368   0.4332   2.0986

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   3.6061     1.6953   2.127   0.0334 *
RATIO        -0.2196     0.1005  -2.184   0.0289 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 26.287  on 18  degrees of freedom
Residual deviance: 14.221  on 17  degrees of freedom
AIC: 18.221

Number of Fisher Scoring iterations: 6
```

Conclusions - reject the null hypothesis. An increase in perimeter to area ratio was associated with a significant decline in the chances of *Uta* lizard presence on Gulf of California islands ($b = -0.202, z = -2.184, P = 0.029$).

Step 7 (Key 17.2) - Compare the fit of full and reduced models (G^2) as an alternative (potentially more reliable given the relatively small sample size) to the individual parameter based approach

```
> anova(polis.glm, test = "Chisq")
Analysis of Deviance Table

Model: binomial, link: logit

Response: PA

Terms added sequentially (first to last)


      Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                     18    26.2869
RATIO  1  12.0662        17    14.2207    0.0005
```

Conclusions - reject the null hypothesis. An increase in perimeter to area ratio was associated with a significant decline in the chances of *Uta* lizard presence on Gulf of California islands ($G^2 = 12.066, df = 1, P < 0.001$).

Step 8 (Key 17.6) - Examine the odds ratio for the occurrence of *Uta* lizards.

```
> library(biology)
> odds.ratio(polis.glm)
      Odds ratio Lower 95
RATIO  0.8028734     0.659303    0.9777077
```

Conclusions - the chances of *Uta* lizards being present on an island decline by 0.803 (20%) for every unit increase in perimeter to area ratio.

Step 9 - Estimate the strength (r^2) of the association

```
> 1 - (polis.glm$dev/polis.glm$null)
[1] 0.4590197
```

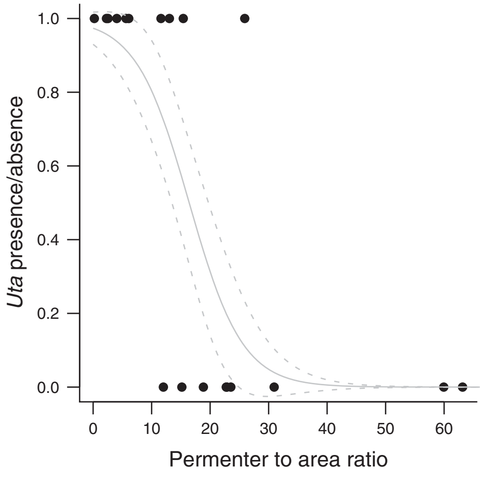
Conclusions - 46% of the uncertainty in *Uta* lizard occurrence is explained by the perimeter to area ratio of the islands.

Step 10 - Calculate the LD50 (perimeter to area ratio at which there is a 50% chance of *Uta* lizard occurrence).

```
> -polis.glm$coef[1]/polis.glm$coef[2]
(Intercept)
   16.42420
```

Step 11 - Summarize the association between *Uta* lizard occurrence and island perimeter to area ratio.

```
> # Calculate predicted values based on fitted model
> xs <- seq(0, 70, l = 1000)
> polis.predict <- predict(polis.glm, type = "response", se = T,
+     newdata = data.frame(RATIO = xs))
> # construct base plot
> plot(PA ~ RATIO, data = polis, xlab = "", ylab = "", axes = F,
+     pch = 16)
> # Plot fitted model and 95% CI bands
> points(polis.predict$fit ~ xs, type = "l", col = "gray")
> lines(polis.predict$fit + polis.predict$se.fit ~ xs,
+     col = "gray", type = "l", lty = 2)
> lines(polis.predict$fit - polis.predict$se.fit ~ xs,
+     col = "gray", type = "l", lty = 2)
> mtext(expression(paste(italic(Uta), "presence/absence")), 2,
+     line = 3)
> axis(2, las = 1)
> mtext("Permenter to area ratio", 1, line = 3)
> axis(1)
> box(bty = "l")
```



Example 17B: Multiple logistic regression

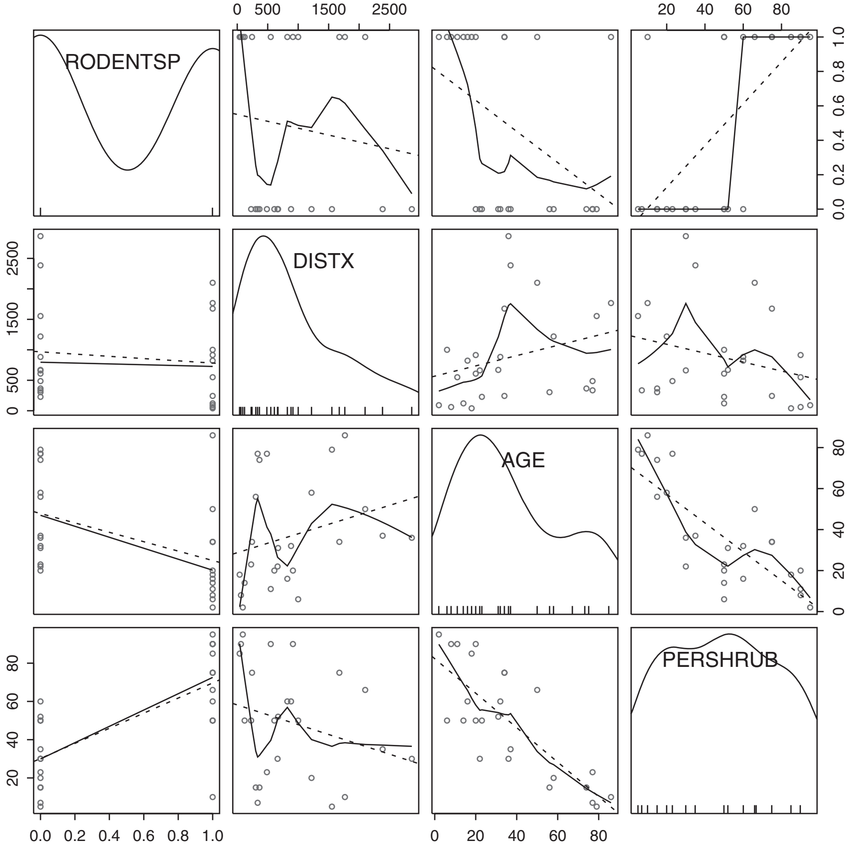
Bolger et al. (1997) investigated the impacts of habitat fragmentation on the occurrence of native rodents. Quinn and Keough (2002) subsequently modelled the presence/absence native rodents against some of the Bolger et al. (1997)'s biogeographic variables (area of the canyon fragment, percent shrub cover and distance to the nearest canyon fragment) (from Box 13.2 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Bolger et al. (1997) data set

```
> bolger <- read.table("bolger.csv", header = T, sep = ",")
```

Step 2 (Key 9.3 & 17.2b) - Investigate the assumption of (multi)collinearity

```
> library(car)
> scatterplot.matrix(~RODENTSP + DISTX + AGE + PERSHRUB,
     data = bolger)
```




```
> bolger.glm <- glm(RODENTSP ~ DISTX + AGE + PERSHRUB, family =
+     binomial, data = bolger)
> vif(bolger.glm)
   DISTX       AGE PERSHRUB
1.117702 1.971138 2.049398
```

Conclusions - Although there is clearly a relationship between fragment age and percent shrub cover, variance inflation values do not indicate a major collinearity issue (values less than 5).

Step 3 (Key 17.3) - Check the (lack of) fit and appropriateness of the model with goodness-of-fit tests

- le Cessie-van Houwelingen normal test statistic

```
> library(Design)
> bolger.lrm <- lrm(RODENTSP ~ DISTX + AGE + PERSHRUB, data =
+     bolger, y = T, x = T)
> resid(bolger.lrm, type = "gof")
Sum of squared errors     Expected value|H0                SD
            3.1538988             3.0291378         0.1382219
                    Z                     P
            0.9026142             0.3667307
```

- Pearson $\chi^2 p - value$

```
> pp <- sum(resid(bolger.lrm, type = "pearson")^2)
> 1 - pchisq(pp, bolger.glm$df.resid)
[1] 0.4697808
```

- Deviance (G^2)

```
> 1 - pchisq(bolger.glm$deviance, bolger.glm$df.resid)
[1] 0.5622132
```

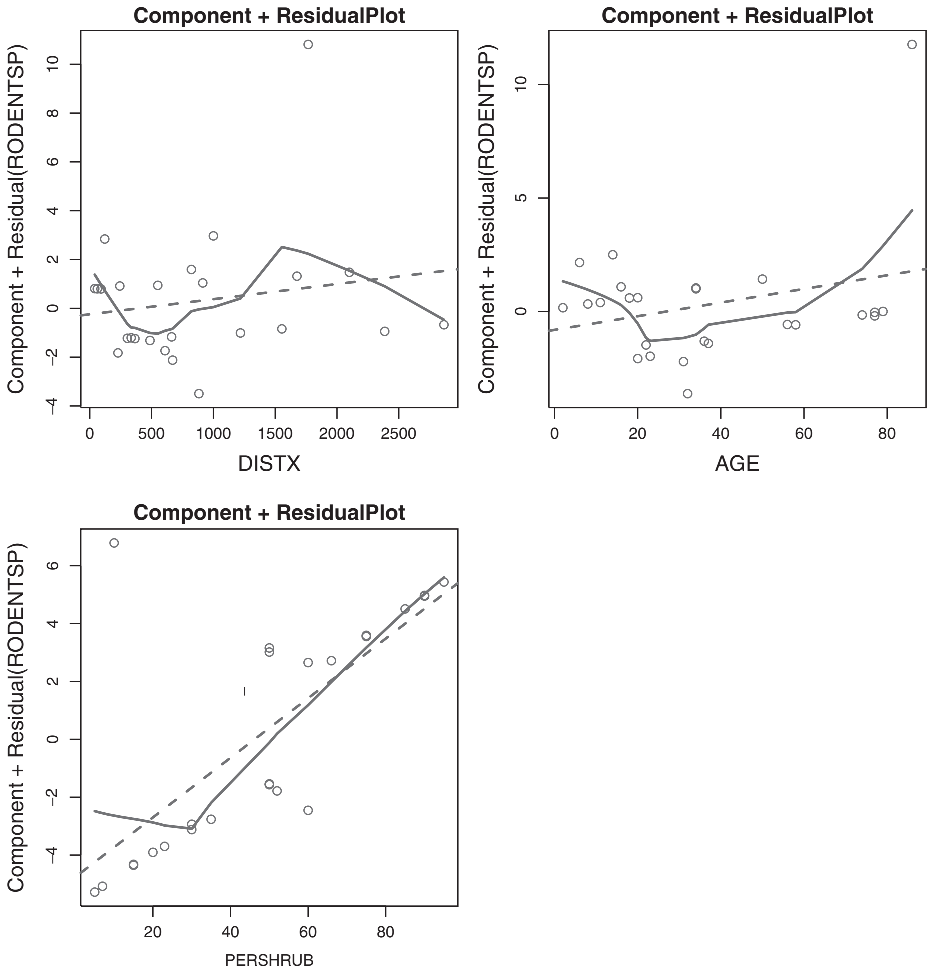
- Estimated dispersion parameter **(Key 17.4)**

```
> pp/bolger.glm$df.resid
[1] 0.991585
```

Conclusions - no evidence to suggest that data did not come from population that follows the logistic regression (no evidence for a lack of fit of the model). Furthermore, the estimated dispersion parameter is essentially one and thus there is no evidence of over-dispersion.

Step 4 (Key 17.3) - Confirm linearity between the log odds ratio of rodent presence and the biogeographic variables with a component+residual plot

```
> library(car)
> cr.plots(bolger.glm, ask = F)
```



Conclusions - no substantial evidence of non-linearity.

Step 5 (Key 17.3) - Examine the influence measures

```
> influence.measures(bolger.glm)

    dfb.1_ dfb.DIST  dfb.AGE dfb.PERS   dffit cov.r   cook.d    hat inf
1  -0.2416  0.17167  0.19092   0.2339  0.2945 1.536 0.010863 0.2486
2  -0.0507  0.01588  0.03176   0.0635  0.0702 1.289 0.000603 0.0693
3  -0.1286  0.08647  0.08633   0.1393  0.1640 1.367 0.003342 0.1397
4  -0.1331 -0.05235  0.14044   0.1744  0.2180 1.366 0.005983 0.1551
5   0.0853  0.02470 -0.14900   0.0436  0.4100 1.101 0.024581 0.1239
```

```
6    0.7766 -0.58021 -0.58883  -0.4920  1.0292 0.822 0.200988 0.2300
7   -0.0112 -0.00293 -0.04378   0.0337 -0.1114 1.317 0.001530 0.0976
8   -0.0474  0.00919 -0.03268   0.0664 -0.1578 1.272 0.003141 0.0906
9   -0.0806 -0.72577  0.22425   0.1696 -0.8547 1.773 0.095918 0.4319   *
10  -0.0302  0.11865 -0.06183  -0.0776 -0.4350 0.952 0.030802 0.0952
11  -0.0291  0.09988 -0.08901   0.0438 -0.1893 1.386 0.004468 0.1563
12  -0.0476  0.00555  0.02388   0.0705  0.0872 1.291 0.000936 0.0756
13  -0.0650 -0.03614  0.05294   0.1024  0.1365 1.324 0.002310 0.1093
14  -0.0592  0.06048 -0.00219   0.0613 -0.1017 1.298 0.001276 0.0841
15  -0.0316 -0.57390  0.12542   0.0908 -0.6971 1.426 0.066536 0.3097
16  -0.2834  0.30568  0.14991   0.1456 -0.4906 1.131 0.035237 0.1595
17  -0.1710  0.05037  0.12748   0.1513 -0.1798 1.322 0.004060 0.1223
18  -0.0429 -0.02244  0.02273   0.0769  0.1100 1.309 0.001494 0.0928
19  -0.3040  0.22753  0.95111  -0.0251  1.5502 0.352 0.813906 0.2120   *
20   0.8191  0.12450 -0.96546  -0.5818  1.1426 0.860 0.240598 0.2675
21  -0.2730  0.12393  0.22277   0.1531 -0.4288 1.090 0.027170 0.1271
22  -0.0278  0.05457 -0.03695   0.0370 -0.1020 1.329 0.001279 0.1022
23  -0.0315 -0.01174  0.01115   0.0580  0.0841 1.297 0.000868 0.0781
24   0.3076 -0.05357 -0.29988  -0.4401 -0.6763 0.731 0.094200 0.1175
25   0.0636  0.20880 -0.33862  -0.0242 -0.4887 1.568 0.030804 0.3042
```

Conclusions - Although the Dfbeta (Cook's D equivalent) value of one of the fragments (19) was substantially higher than the others, it was not considered overly influential[l].

Step 6 (Key 17.2b) - Examine the parameter estimates from the fitted logistic regression model.

```
> summary(bolger.glm)
Call:
glm(formula = RODENTSP ~ DISTX + AGE + PERSHRUB, family = binomial,
    data = bolger)

Deviance Residuals:
    Min        1Q    Median       3Q       Max
-1.5823  -0.5985  -0.2813   0.3699   2.1702

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.9099159  3.1125426  -1.899   0.0576 .
DISTX        0.0003087  0.0007741   0.399   0.6900
AGE          0.0250077  0.0376618   0.664   0.5067
PERSHRUB     0.0958695  0.0406119   2.361   0.0182 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

 (Dispersion parameter for binomial family taken to be 1)
```

[l] Note that as indicated by Quinn and Keough (2002), the model would not converge in the absence of this observation and thus its lack of influence on the fit of the model could not be varified.

```
    Null deviance: 34.617  on 24  degrees of freedom
Residual deviance: 19.358  on 21  degrees of freedom
AIC: 27.358

Number of Fisher Scoring iterations: 5
```

Conclusions - The chances of native rodent occurence increases significantly with increasing shrub cover ($b = 0.096$, $z = 2.361$, $P = 0.0182$), yet were not found to be affected by fragment isolation age or distance.

Step 7 (Key 17.2b) - Compare the fit of full and reduced models (G^2) as an alternative (potentially more reliable given the relatively small sample size) to the individual parameter based approach.

```
> # saturated model
> bolger.glmS <- glm(RODENTSP ~ DISTX + AGE + PERSHRUB,
+ family = binomial, data = bolger)
> # Reduced model for distance
> bolger.glm.Dist <- glm(RODENTSP ~ AGE + PERSHRUB,
+ family = binomial, data = bolger)
> #OR
> bolger.glm.Dist <- update(bolger.glmS, "~.-DISTX")
> anova(bolger.glmS, bolger.glm.Dist, test = "Chisq")
Analysis of Deviance Table

Model 1: RODENTSP ~ DISTX + AGE + PERSHRUB
Model 2: RODENTSP ~ AGE + PERSHRUB
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        21    19.3576
2        22    19.5135 -1  -0.1559    0.6929

> # Reduced model for age
> bolger.glm.Age <- update(bolger.glmS, "~.-AGE")
> anova(bolger.glmS, bolger.glm.Age, test = "Chisq")
Analysis of Deviance Table

Model 1: RODENTSP ~ DISTX + AGE + PERSHRUB
Model 2: RODENTSP ~ DISTX + PERSHRUB
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        21    19.3576
2        22    19.8022 -1  -0.4446    0.5049

> # Reduced model for shrub cover
> bolger.glm.Shrub <- update(bolger.glmS, "~.-PERSHRUB")
> anova(bolger.glmS, bolger.glm.Shrub, test = "Chisq")
Analysis of Deviance Table
```

```
Model 1: RODENTSP ~ DISTX + AGE + PERSHRUB
Model 2: RODENTSP ~ DISTX + AGE
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        21    19.3576
2        22    28.9152 -1  -9.5577    0.0020
```

Step 8 (Key 17.6) - Examine the odds ratio for the occurrence of native rodents.

```
> library(biology)
> odds.ratio(bolger.glm)
         Odds ratio Lower 95
DISTX      1.000309    0.9987922     1.001828
AGE        1.025323    0.9523639     1.103871
PERSHRUB   1.100615    1.0164046     1.191803
```

Conclusions - the chances of native rodents being present in fragments increases slightly (1%) for every 1% increase in shrub cover.

Step 9 (Key 17.8) - Compare the fit of all additive combinations of predictor variables to select the most parsimonious model and perform model averaging to estimate the relative contribution of each of the predictor variables (based on AIC_c).

```
> library(biology)
> Model.selection.glm(bolger.glm)
Model selection
Response: RODENTSP
            Deviance      AIC     AICc  deltaAIC        wAIC qAIC
1. DI       34.24479 38.24479 38.79024 14.195496 0.000466208 15.5
2. AG       28.92306 32.92306 33.46852  8.873775 0.006670784 15.5
3. PE       20.04929 24.04929 24.59474  0.000000 0.563757818 15.5
4. DI+AG    28.91524 34.91524 36.05810 11.463358 0.001827494 17.0
5. DI+PE    19.80219 25.80219 26.94505  2.350306 0.174072445 17.0
6. AG+PE    19.51350 25.51350 26.65636  2.061614 0.201103152 17.0
7. DI+AG+PE 19.35758 27.35758 29.35758  4.762839 0.052102098 18.5
               qAICc Select
1. DI       16.04545
2. AG       16.04545
3. PE       16.04545      *
4. DI+AG    18.14286
5. DI+PE    18.14286
6. AG+PE    18.14286
7. DI+AG+PE 20.50000      *


Model averaging
Response: RODENTSP
               Sum     Estimate Unconditional_SE     Lower95CI
DISTX    0.2284682 8.008503e-05     6.688235e-05 -5.100437e-05
```

```
AGE      0.2617035 6.202364e-03     5.591199e-03 -4.756386e-03
PERSHRUB 0.9910355 8.106718e-02     6.891972e-03  6.755891e-02
            Upper95CI
DISTX    0.0002111744
AGE      0.0171611152
PERSHRUB 0.0945754434
attr(,"heading")
[1] "Model averaging\n"     "Response: RODENTSP \n"
```

Conclusions - The most parsimonious model relates the presence of native rodents to percentage shrub cover only (on the basis of AIC_c). Model averaging indicated that the percentage of shrub cover was substantially more influential than the other predictors.

Step 10 - construct the predictive model

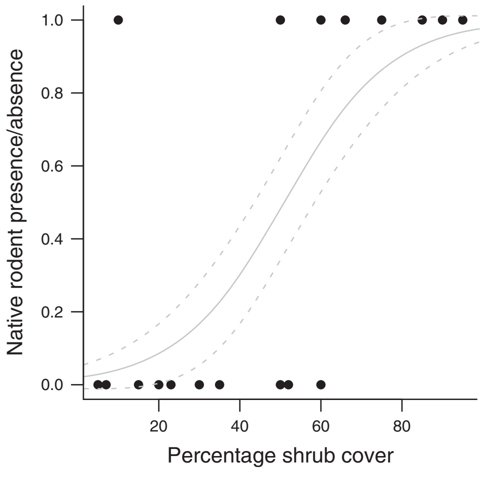
```
> bolger.glm <- glm(RODENTSP ~ PERSHRUB, family = binomial,
      data = bolger)
> summary(bolger.glm)
Call:
glm(formula = RODENTSP ~ PERSHRUB, family = binomial, data = bolger)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4827  -0.6052  -0.2415   0.5421   2.5218

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.90342    1.55775  -2.506  0.01222 *
PERSHRUB     0.07662    0.02878   2.663  0.00775 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 34.617  on 24  degrees of freedom
Residual deviance: 20.049  on 23  degrees of freedom
AIC: 24.049

Number of Fisher Scoring iterations: 5
```

Conclusions - The predictive model is: $g(PArodents) = (0.08 \times Shrubcover) - 3.90$. Expressing this in terms of likelihood of rodents being present, the predictive model becomes:

$$presence = \frac{1}{(1 + \exp^{-(0.08 \times Perc\ Shrub) - 3.9)})}$$

Step 11 - Summarize the association between native rodent occurrence and percentage shrub cover.

```
> xs <- seq(0, 100, l = 1000)
> bolger.predict <- with(bolger, (predict(bolger.glm, type =
+     "response", se = T, newdata = data.frame(DISTX = mean(DISTX),
+         AGE = mean(AGE), PERSHRUB = xs))))
> plot(RODENTSP ~ PERSHRUB, data = bolger, xlab = "", ylab = "",
+     axes = F, pch = 16)
> points(bolger.predict$fit ~ xs, type = "l", col = "gray")
> lines(bolger.predict$fit + bolger.predict$se.fit ~ xs, col =
+     "gray", type = "l", lty = 2)
> lines(bolger.predict$fit - bolger.predict$se.fit ~ xs, col =
+     "gray", type = "l", lty = 2)
> mtext("Native rodent presence/absence", 2, line = 3)
> axis(2, las = 1)
> mtext("Percentage shrub cover", 1, line = 3)
> axis(1)
> box(bty = "l")
```



Example 17C: Multiple logistic regression

Gotelli and Ellison (2002) investigated the biogeographical determinants of ant species richness at a regional scale. Ellison (2004) then used an excerpt of those data to contrast inferential and Bayesian approaches. Specifically, ant species richness was modelled against latitude, elevation and habitat type (bog or forest) using Poisson regression.

Step 1 - Import (section 2.3) the Gotelli and Ellison (2002) data set

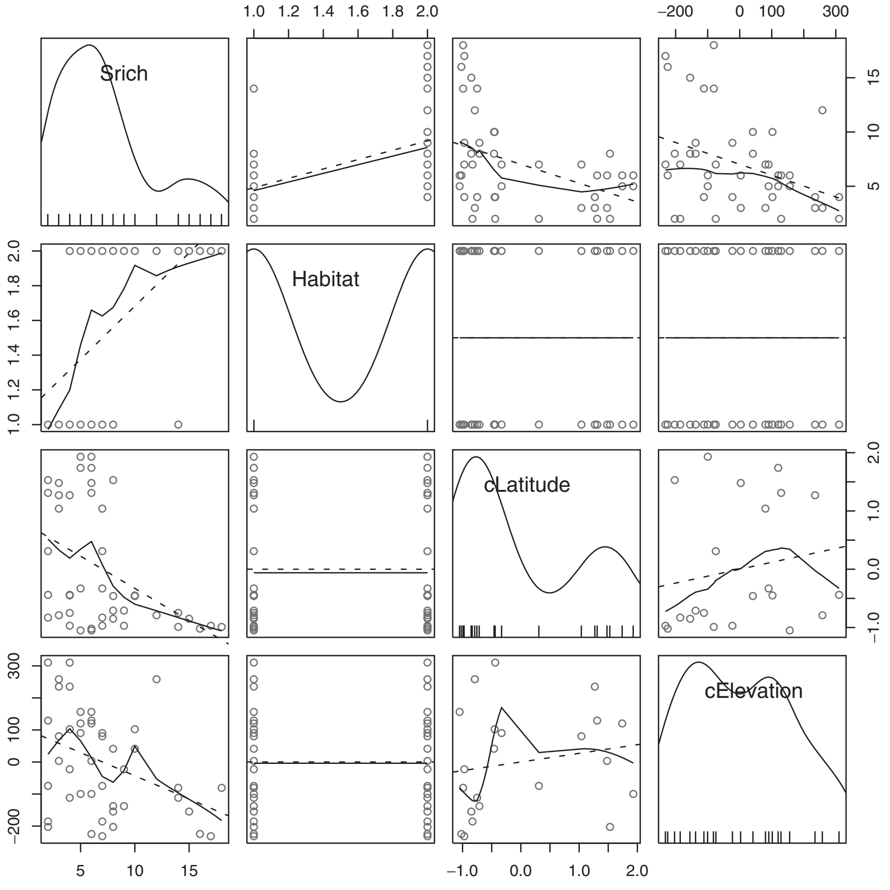
```
> gotelli <- read.table("gotelli.csv", header = T, sep = ",")
```

Step 2 (Key 9.3b & 17.2b) - In anticipation of fitting a multiplicative poisson regression model, the continuous predictor variables should be centered to avoid obvious collinearity issues.

```
> gotelli$cLatitude <- scale(gotelli$Latitude, scale = F)
> gotelli$cElevation <- scale(gotelli$Elevation, scale = F)
```

Step 3 (Key 9.3 & 17.2b) - Investigate the assumption of (multi)collinearity

```
> library(car)
> scatterplot.matrix(~Srich + Habitat * cLatitude * cElevation,
+     data = gotelli)
```




```
> gotelli.glm <- glm(Srich ~ Habitat * cLatitude * cElevation,
+     family = poisson, data = gotelli)
> vif(gotelli.glm)
                      Habitat                     cLatitude
                     1.167807                      3.113812
                   cElevation             Habitat:cLatitude
                     3.563564                      3.220434
           Habitat:cElevation          cLatitude:cElevation
                     3.609016                      3.477485
Habitat:cLatitude:cElevation
                     3.644151
```

Conclusions - no evidence of collinearity for the centered predictor variables.

Step 4 (Key 17.3) - Check the (lack of) fit and appropriateness of the model with goodness-of-fit tests

- Pearson χ^2

```
> pp <- sum(resid(gotelli.glm, type = "pearson")^2)
> 1 - pchisq(pp, gotelli.glm$df.resid)
[1] 0.2722314
```

- Deviance (G^2)

```
> 1 - pchisq(gotelli.glm$deviance, gotelli.glm$df.resid)
[1] 0.3057782
```

Conclusions - no evidence for a lack of fit of the model).

Step 5 (Key 17.3) - Examine the influence measures (I have truncated the output to save space).

```
> influence.measures(gotelli.glm)

     dfb.1_ dfb.HbtF  dfb.cLtt  dfb.cElv dfb.HbF.L  dfb.HF.E  dfb.cL.E
1   2.57e-16 -0.18734  2.40e-16 -9.36e-17   0.21122 -0.151987 -7.36e-17
2  -9.16e-18  0.01983 -1.62e-17  2.03e-17  -0.01912 -0.035302 -3.52e-17
3   9.54e-17  0.17021  2.74e-16  6.18e-17  -0.16883 -0.016553  1.08e-16
4   3.98e-17  0.04115  1.93e-17 -1.34e-16  -0.03238 -0.078512 -1.30e-16
5   8.01e-18 -0.07957 -4.75e-17  2.98e-17   0.07837 -0.017767 -1.23e-17
6   1.84e-17  0.05225 -1.49e-17  2.78e-17  -0.03326 -0.041660 -8.27e-18
7   2.63e-16 -0.19210  7.70e-17 -3.16e-16   0.10281  0.223120 -7.27e-16
8  -1.45e-19  0.32864  4.13e-17 -7.36e-17  -0.29425  0.368322 -1.51e-16
9  -2.71e-17  0.06155 -1.61e-17  1.35e-17  -0.03276 -0.027660  1.35e-17
...
```

Conclusions - no observations are overly influential.

Step 6 (Key 17.4) - Estimate the dispersion parameter to evaluate over (or under) dispersion in the fitted model

```
> # via Pearson residuals
> pp/gotelli.glm$df.resid
[1] 1.129723
> # OR via deviance
> gotelli.glm$deviance/gotelli.glm$df.resid
[1] 1.104765
```

Conclusions - the dispersion parameter is not substantially greater than 1, overdispersion is unlikely to be an issue.

Step 7 (Key 17.2b) - Examine the parameter estimates and associated null hypothesis tests.

```
> summary(gotelli.glm)
Call:
glm(formula = Srich ~ Habitat * cLatitude * cElevation, family =
    poisson, data = gotelli)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1448  -0.7473  -0.0856   0.5426   2.6453

Coefficients:
                                    Estimate Std. Error z value Pr(>|z|)
(Intercept)                        1.5237266  0.1044276  14.591  < 2e-16 ***
HabitatForest                      0.6284757  0.1292095   4.864 1.15e-06 ***
cLatitude                         -0.2257304  0.1059277  -2.131   0.0331 *
cElevation                        -0.0006575  0.0006878  -0.956   0.3391
HabitatForest:cLatitude           -0.0089115  0.1314652  -0.068   0.9460
HabitatForest:cElevation          -0.0006053  0.0008531  -0.710   0.4780
cLatitude:cElevation               0.0004718  0.0007208   0.655   0.5127
HabitatForest:cLatitude:cElevation -0.0003348  0.0008941  -0.375   0.7080
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 102.763  on 43  degrees of freedom
Residual deviance:  39.772  on 36  degrees of freedom
AIC: 216.13

Number of Fisher Scoring iterations: 4
```

Conclusions - Species richness of native rodents was found to be significantly greater in forest than bog habitats ($P < 0.001$) and was found to decline significantly with increasing latitude ($b = -0.226, z = -2.131, P = 0.0331$)

Step 8 (Key 17.8) - Select the most parsimonious[m] model on the basis of AIC_c.

```
> library(MuMIn)
> model.avg(get.models(dredge(gotelli.glm)))
Model summary:
        Deviance AICc Delta Weight
1+2+3       40.7  210  0.00  0.510
1+2+3+5     40.3  212  2.14  0.175
1+2+3+4     40.3  212  2.19  0.170
1+2+3+6     40.7  213  2.51  0.145

Variables:
                   1                    2                    3
          cElevation            cLatitude              Habitat
                   4                    5                    6
cElevation:cLatitude   cElevation:Habitat    cLatitude:Habitat

Averaged model parameters:
                        Coefficient Variance       SE Unconditional SE
cElevation                -1.07e-03 4.05e-14 0.000436         0.000449
cLatitude                 -2.33e-01 2.32e-05 0.067900         0.070000
HabitatForest              6.31e-01 2.14e-04 0.121000         0.125000
(Intercept)                1.53e+00 9.72e-05 0.099300         0.102000
cElevation:cLatitude       4.35e-05 1.61e-15 0.000117         0.000119
```

[m] model with greatest fit considering the number of predictor terms (including interactions).

```
cElevation:HabitatForest   -8.65e-05 1.88e-14 0.000223         0.000227
cLatitude:HabitatForest    -3.66e-03 5.60e-06 0.021600         0.022200
                         Lower CI  Upper CI
cElevation               -0.00195 -0.000190
cLatitude                -0.37000 -0.095300
HabitatForest             0.38700  0.876000
(Intercept)               1.33000  1.730000
cElevation:cLatitude     -0.00019  0.000277
cElevation:HabitatForest -0.00053  0.000357
cLatitude:HabitatForest  -0.04710  0.039800

Relative variable importance:
          cElevation            cLatitude              Habitat
                1.00                 1.00                 1.00
  cElevation:Habitat cElevation:cLatitude    cLatitude:Habitat
                0.18                 0.17                 0.15
```

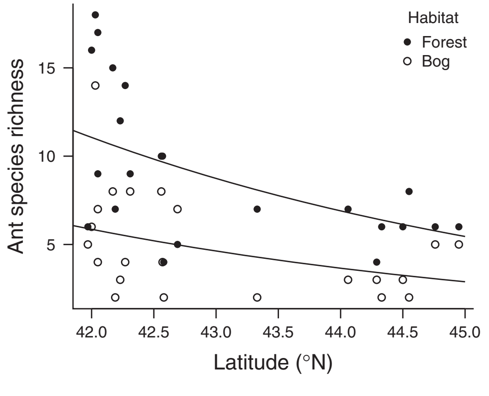
Conclusions - the most parsimonious model includes only the three main factors (elevation, habitat and latitude), which are of roughly equivalent relative importance.

Step 9 - Examine the parameter estimates from the best fitting model. Note , there is no need for these variables to be centered as there are no interactions.

```
> gotelli.glm <- glm(Srich ~ Habitat + Latitude + Elevation,
+     family = poisson, data = gotelli)
> summary(gotelli.glm)
Call:
glm(formula = Srich ~ Habitat + Latitude + Elevation, family =
    poisson, data = gotelli)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.20939  -0.72643  -0.05933   0.51571   2.60147

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)   11.9368121  2.6214970   4.553 5.28e-06 ***
HabitatForest  0.6354389  0.1195664   5.315 1.07e-07 ***
Latitude      -0.2357930  0.0616638  -3.824 0.000131 ***
Elevation     -0.0011411  0.0003749  -3.044 0.002337 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 102.763  on 43  degrees of freedom
Residual deviance:  40.690  on 40  degrees of freedom
AIC: 209.04

Number of Fisher Scoring iterations: 4
```

Step 10 - Produce a summary figure relating the species richness of ants to latitudinal variation for forests and bog habitats.

```
> # Produce base plot
> xs <- seq(40, 45, l = 1000)
> plot(Srich ~ Latitude, data = gotelli, type = "n", axes = F,
+ xlab = "", ylab = "")
> # Plot the points and predicted trends
> points(Srich ~ Latitude, data = gotelli, subset = Habitat ==
+ "Forest", pch = 16)
> pred <- predict(gotelli.glm, type = "response", se = T, newdata
+ = data.frame(Latitude = xs, Habitat = "Forest", Elevation =
+ mean(gotelli$Elevation)))
> lines(pred$fit ~ xs)
> points(Srich ~ Latitude, data = gotelli, subset = Habitat ==
+ "Bog", pch = 21)
> pred <- predict(gotelli.glm, type = "response", se = T, newdata
+ = data.frame(Latitude = xs, Habitat = "Bog", Elevation =
+ mean(gotelli$Elevation)))
> lines(pred$fit ~ xs)
> # Axes titles
> mtext("Ant species richness", 2, line = 3)
> axis(2, las=1)
> mtext(expression(paste("Latitude (", degree*N, ")")), 1,
+ line = 3)
> axis(1)
> legend("topright", legend = c("Forest", "Bog"), pch = c(16, 21),
+ title = "Habitat", bty = "n")
> box(bty = "l")
```



Example 17D: Log-linear modelling

Sinclair and Arcese (1995) investigated the association between predation, sex and health (via marrow type) in Serengeti wildebeest. Quinn and Keough (2002) used these data to illustrate log-linear modelling (Box 14.5 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Sinclair and Arcese (1995) data set

```
> sinclair <- read.table("sinclair.csv", header = T, sep = ",")
```

Step 2 (Key 17.5b & 17.8) - Fit the various combinations of log-linear models heirarchically (starting with the highest order, or saturated, model).

- Fit the full saturated model

```
> sinclair.glm <- glm(COUNT ~ SEX * MARROW * DEATH, family =
+     poisson, data = sinclair)
```

- Perform model selection to identify the most parsimonious model (on the basis of *AIC*)

```
> library(MuMIn)
> dredge(sinclair.glm, rank = "AIC")
Model selection table
   (Intr) DEATH MARROW SEX DEATH:MARROW DEATH:SEX MARROW:SEX
19  3.258     1      1   1            1         1          1
9   2.944     1      1                1
16  2.971     1      1   1            1                    1
18  3.072     1      1   1            1         1          1
12  2.953     1      1   1            1
15  2.976     1      1   1            1         1
5   3.146     1      1
14  3.173     1      1   1                                 1
8   3.155     1      1   1
17  3.195     1      1   1                      1          1
13  3.178     1      1   1                      1
3   3.341            1
11  3.367            1   1                                 1
7   3.350            1   1
2   2.741     1
6   2.750     1          1
10  2.773     1          1                      1
1   2.936
4   2.944                1
   DEATH:MARROW:SEX  k        Dev.   AIC  delta weight
19                1 12 -1.776e-15 24.00  0.000  0.435
9                    6  1.326e+01 25.26  1.260  0.231
16                   9  8.465e+00 26.46  2.465  0.127
18                  10  7.188e+00 27.19  3.188  0.088
12                   7  1.324e+01 27.24  3.243  0.086
15                   8  1.316e+01 29.16  5.156  0.033
5                    4  4.278e+01 50.78 26.780  0.000
14                   7  3.798e+01 51.98 27.980  0.000
8                    5  4.276e+01 52.76 28.760  0.000
17                   8  3.790e+01 53.90 29.900  0.000
13                   6  4.268e+01 54.68 30.680  0.000
```

```
3                    3  4.990e+01 55.90 31.900  0.000
11                   6  4.510e+01 57.10 33.100  0.000
7                    4  4.988e+01 57.88 33.880  0.000
2                    2  6.983e+01 73.83 49.830  0.000
6                    3  6.982e+01 75.82 51.820  0.000
10                   4  6.973e+01 77.73 53.730  0.000
1                    1  7.695e+01 78.95 54.950  0.000
4                    2  7.693e+01 80.93 56.930  0.000
```

Conclusions - On the basis of AIC, model 19 (the full saturated model) is the best fit (lowest AIC). However, model 9 (`~DEATH+MARROW+DEATH:MARROW`) is not a significantly poorer fit (its delta[n] is less than 2) than the model with the smallest AIC. Note, this is a slightly different conclusion to that drawn by Quinn and Keough (2002). The model selection procedure used by Quinn and Keough (2002) used a hierarchical step function to generate the set of possible model fits, whereas the function above assesses the fit of all possible model combinations. Furthermore, the AIC values reported by Quinn and Keough (2002) are AIC delta values.

Step 3 (Key 17.5b) - Fit a range of full and reduced models (according to Table 17.2) to examine conditional dependence.

- Full saturated model

```
> sinclair.glm <- glm(COUNT ~ SEX * MARROW * DEATH, family =
+     poisson, data = sinclair)
```

- Complete dependence (`SEX:MARROW:DEATH`= 0)

```
> sinclair.glm1 <- update(sinclair.glm, ~. - SEX:MARROW:DEATH,
+     data = sinclair)
> anova(sinclair.glm, sinclair.glm1, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ SEX * MARROW * DEATH
Model 2: COUNT ~ SEX + MARROW + DEATH + SEX:MARROW + SEX:DEATH
               + MARROW:DEATH
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         0 -6.883e-15
2         2     7.1883 -2  -7.1883    0.0275
```

- Conditional independence (`SEX:DEATH`= 0)

```
> sinclair.glm2 <- update(sinclair.glm1, ~. - SEX:DEATH, data =
      sinclair)
> anova(sinclair.glm1, sinclair.glm2, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ SEX + MARROW + DEATH + SEX:MARROW + SEX:DEATH
               + MARROW:DEATH
Model 2: COUNT ~ SEX + MARROW + DEATH + SEX:MARROW + MARROW:DEATH
```

[n] Delta is the difference between a models' AIC and the smallest AIC.

```
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         2     7.1883
2         3     8.4647 -1  -1.2763    0.2586
```

- Conditional independence (`SEX:MARROW`$= 0$)

```
> sinclair.glm4 <- update(sinclair.glm1, ~. - SEX:MARROW, data =
       sinclair)
> anova(sinclair.glm1, sinclair.glm4, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ SEX + MARROW + DEATH + SEX:MARROW + SEX:DEATH
               + MARROW:DEATH
Model 2: COUNT ~ SEX + MARROW + DEATH + SEX:DEATH + MARROW:DEATH
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         2     7.1883
2         4    13.1560 -2  -5.9677    0.0506
```

- Conditional independence (`DEATH:MARROW`$= 0$)

```
> sinclair.glm3 <- update(sinclair.glm1, ~. - DEATH:MARROW,
       data = sinclair)
> anova(sinclair.glm1, sinclair.glm3, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ SEX + MARROW + DEATH + SEX:MARROW + SEX:DEATH
               + MARROW:DEATH
Model 2: COUNT ~ SEX + MARROW + DEATH + SEX:MARROW + SEX:DEATH
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         2      7.188
2         4     37.898 -2  -30.710 2.145e-07
```

Conclusions - reject the null hypothesis of no three-way interaction. There is an association between cause of death, sex and marrow type (health condition) in Serengeti wildebeest ($G^2 = 7.19, df = 2, P = 0.028$).

Step 4 - Investigate the patterns of association further

- Pearson residuals

```
> xtabs(resid(sinclair.glm1, type = "pearson") ~ SEX + MARROW +
+     DEATH, sinclair)
, , DEATH = NPRED

        MARROW
SEX               OG        SWF         TG
  FEMALE  0.9479718 -0.8907547 -0.4245814
  MALE   -1.0876228  1.2484046  0.3639733

, , DEATH = PRED
```

```
        MARROW
SEX              OG         SWF          TG
  FEMALE -0.7301089  0.5406249  0.7186928
  MALE    0.7090967 -0.6413988 -0.5215390
```

Conclusions - there were more healthy males (SWF marrow type) and fewer undernourished (OG marrow type) that died of non-predation causes than expected, whereas the reverse was the case for females.

- Split the analysis up and investigate the associations between cause of death and marrow type for each sex separately.

```
> # females
> sinclair.glmR <- glm(COUNT ~ DEATH + MARROW, family = poisson,
+ data = sinclair, subset = SEX == "FEMALE")
> sinclair.glmF <- glm(COUNT ~ DEATH * MARROW, family = poisson,
+ data = sinclair, subset = SEX == "FEMALE")
> anova(sinclair.glmR, sinclair.glmF, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ DEATH + MARROW
Model 2: COUNT ~ DEATH * MARROW
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         2    13.9626
2         0 -2.220e-15  2  13.9626    0.0009

> # males
> sinclair.glmR <- glm(COUNT ~ DEATH + MARROW, family = poisson,
+ data = sinclair, subset = SEX == "MALE")
> sinclair.glmF <- glm(COUNT ~ DEATH * MARROW, family = poisson,
+ data = sinclair, subset = SEX == "MALE")
> anova(sinclair.glmR, sinclair.glmF, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ DEATH + MARROW
Model 2: COUNT ~ DEATH * MARROW
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         2     23.935
2         0  3.331e-15  2   23.935 6.346e-06
```

Conclusions - an association exists between cause of death and marrow type for both males and females, although it is perhaps strongest for the latter.

- Odds ratios of being killed by predation vs non-predation for each sex and marrow type combination (**Key 17.6**)

```
> # Males
> library(biology)
> male.tab <- xtabs(COUNT ~ DEATH + MARROW, data=sinclair,
+ subset=SEX == "MALE")
```

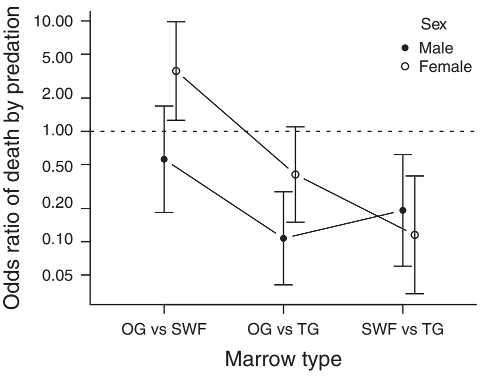
```
> # transpose to express in the context of cause of death
> male.tab <- t(male.tab)
> oddsratios(male.tab)
  Comparison  estimate      lower     upper   midp.exact
1  OG vs SWF 0.5581395 0.18389618 1.6939979 3.199869e-01
2   OG vs TG 0.1073345 0.04067922 0.2832085 2.247925e-06
3  SWF vs TG 0.1923077 0.06004081 0.6159519 5.465174e-03
  fisher.exact   chi.square
1 3.762577e-01 2.998756e-01
2 2.928685e-06 1.865442e-06
3 5.794237e-03 4.123680e-03

> # Females
> female.tab <- xtabs(COUNT ~ DEATH + MARROW, data=sinclair,
+ subset=SEX == "FEMALE")
> female.tab <- t(female.tab)
> oddsratios(female.tab)
  Comparison  estimate      lower     upper   midp.exact
1  OG vs SWF 3.5208333 1.26009037 9.8376018 0.0137649202
2   OG vs TG 0.4062500 0.15034804 1.0977134 0.0788761506
3  SWF vs TG 0.1153846 0.03378972 0.3940136 0.0003808416
  fisher.exact   chi.square
1 0.0206121992 0.0133633241
2 0.0914047377 0.0718385552
3 0.0003765135 0.0002797362
```

Conclusions - the odds of being killed by predation for males with TG marrow type are less than that for either OG or SWF marrow types, whereas female wildebeest with SWF marrow type were less and more likely to be killed by predation than females with OG and TG marrow type respectively.

Step 5 (Key 17.6) - Summarize the predation odds ratios for bone marrow type pairs according to sex.

```
> # make a table for females
> female.tab <- xtabs(COUNT ~ DEATH + MARROW, data = sinclair,
+     subset = SEX == "FEMALE")
> library(biology)
> # calculate the odds ratios for females
> # the table should be transposed such that cause of death
# #     are in columns
> sinclair.or <- oddsratios(t(female.tab))
> plot(estimate ~ as.numeric(Comparison), data = sinclair.or,
+     log = "y", type = "n", axes = F, xlab = "", ylab = "", ylim =
+         range(c(upper, lower)), xlim = c(0.5, 3.5))
> # plot the female data displaced to the right slightly
> with(sinclair.or, points(as.numeric(Comparison) + 0.1, estimate,
+     type = "b"))
```

```
> with(sinclair.or, arrows(as.numeric(Comparison) + 0.1, upper,
+     as.numeric(Comparison) + 0.1, lower, ang = 90, length = 0.1,
+     code = 3))
> # make the male table
> male.tab <- xtabs(COUNT ~ DEATH + MARROW, data = sinclair,
+     subset = SEX == "MALE")
> sinclair.or <- oddsratios(t(male.tab))
> # plot the male odds ratios
> points(estimate ~ Comparison, data = sinclair.or, type = "b",
+     pch = 16)
> with(sinclair.or, arrows(as.numeric(Comparison), upper,
+     as.numeric(Comparison), lower, ang = 90, length = 0.1,
+     code = 3))
> abline(h = 1, lty = 2)
> with(sinclair.or, axis(1, at = as.numeric(Comparison),
      lab = Comparison))
> axis(2, las = 1, cex.axis = 0.75)
> mtext("Marrow type", 1, line = 3)
> mtext("Odds ratio of death by predation", 2, line = 3)
> legend("topright", legend = c("Male", "Female"), pch = c(16,
+     21), bty = "n", title = "Sex")
> box(bty = "l")
```



Example 17E: Log-linear modelling

To investigate the effects of logging (treatment) on the demographics of southern flying squirrels, Taulman et al. (1998) recorded the age and sex of squirrels captured over three years in experimentally logged and unlogged sites. Quinn and Keough (2002) used these data to illustrate log-linear modelling in which squirrel age has considered and interpreted as a response variable (Box 14.6 of Quinn and Keough (2002)).

Step 1 - Import (section 2.3) the Taulman et al. (1998) data set

```
> taulman <- read.table("taulman.csv", header = T, sep = ",")
```

Step 2 - Define year of capture as a categorical, factor vector

```
> taulman$YEAR <- as.factor(taulman$YEAR)
```

Step 3 (Key 17.5b & 17.8) - Fit the various combinations of log-linear models heirarchically (starting with the highest order (saturated) model). As the investigators were primarily interested in demographic (age) patterns, age (juvenile or adult) was considered a response and the investigators were primarily interested in the conditional independence of year by treatment interactions. Consequently, when examining the selection of possible fitted models, it is logical to include this interaction in all the possible models.

- Full saturated model

```
> taulman.glm <- glm(COUNT ~ TREAT * YEAR * AGE, family = poisson,
+      data = taulman)
```

- Note, as age is considered to be a response variable, all fitted models should include the treatment by year interaction term.

```
> dredge(taulman.glm, rank = "AIC", fixed = ~TREAT:YEAR)
Model selection table
  (Intr) AGE TREAT YEAR AGE:TREAT TREAT:YEAR AGE:YEAR
5  3.843   1     1    1         1          1        1
4  3.813   1     1    1                    1        1
6  3.829   1     1    1         1          1        1
2  3.654   1     1    1                    1
3  3.676   1     1    1         1          1
1  3.332         1    1                    1
  AGE:TREAT:YEAR  k      Dev.    AIC   delta weight
5                10 1.882e+00  21.88  0.0000  0.448
4                 9 4.126e+00  22.13  0.2443  0.397
6               1 12 4.122e-10  24.00  2.1180  0.155
2                 7 4.651e+01  60.51 38.6300  0.000
3                 8 4.627e+01  62.27 40.3900  0.000
1                 6 1.021e+02 114.10 92.2600  0.000
```

Step 4 (Key 17.5b) - Examine patterns of conditional independence.

- Complete dependence (`TREAT:YEAR:AGE`= 0)

```
> taulman.glm1 <- update(taulman.glm, ~. - TREAT:YEAR:AGE,
      data = taulman)
> anova(taulman.glm, taulman.glm1, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ TREAT * YEAR * AGE
Model 2: COUNT ~ TREAT + YEAR + AGE + TREAT:YEAR + TREAT:AGE
     + YEAR:AGE
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         0  4.122e-10
2         2    1.88187 -2 -1.88187   0.39026
> AIC(taulman.glm1) - AIC(taulman.glm)
[1] -2.118125
```

- Conditional independence
  - H_o: AGE:YEAR=0

```
> taulman.glm2 <- update(taulman.glm, ~. - YEAR:AGE -
+     TREAT:YEAR:AGE, data = taulman)
> anova(taulman.glm, taulman.glm2, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ TREAT * YEAR * AGE
Model 2: COUNT ~ TREAT + YEAR + AGE + TREAT:YEAR + TREAT:AGE
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         0  4.122e-10
2         4      46.27 -4   -46.27 2.163e-09
> AIC(taulman.glm2) - AIC(taulman.glm)
[1] 38.27045
```

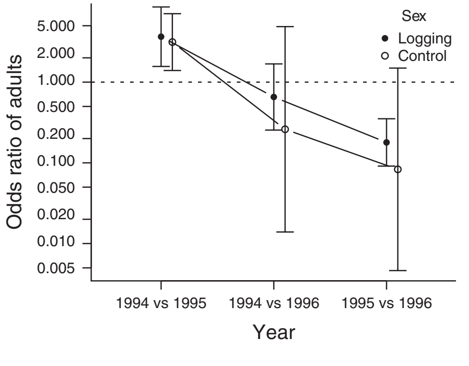
  - H_o: TREAT:AGE=0

```
> taulman.glm3 <- update(taulman.glm, ~. - TREAT:AGE -
+     TREAT:YEAR:AGE, data = taulman)
> anova(taulman.glm, taulman.glm3, test = "Chisq")
Analysis of Deviance Table

Model 1: COUNT ~ TREAT * YEAR * AGE
Model 2: COUNT ~ TREAT + YEAR + AGE + TREAT:YEAR + YEAR:AGE
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         0  4.122e-10
2         3     4.1262 -3  -4.1262    0.2482
> AIC(taulman.glm3) - AIC(taulman.glm)
[1] -1.873847
> dredge(taulman.glm, rank = "AIC", fixed = ~TREAT:YEAR)
Model selection table
  (Intr) AGE TREAT YEAR AGE:TREAT TREAT:YEAR AGE:YEAR
5  3.843   1     1    1         1          1        1
4  3.813   1     1    1                    1        1
6  3.829   1     1    1         1          1        1
2  3.654   1     1    1                    1
3  3.676   1     1    1         1          1
1  3.332         1    1                    1
  AGE:TREAT:YEAR  k      Dev.    AIC   delta weight
5                10 1.882e+00  21.88  0.0000  0.448
4                 9 4.126e+00  22.13  0.2443  0.397
6               1 12 4.122e-10  24.00  2.1180  0.155
2                 7 4.651e+01  60.51 38.6300  0.000
3                 8 4.627e+01  62.27 40.3900  0.000
1                 6 1.021e+02 114.10 92.2600  0.000
```

Conclusions - Whilst, squirrel age was not found to be dependent on the logging treatment in any year ($G^2 = 4.13, df = 3, P = 0.248$), squirrel age was found to be dependent on year within both logging and control treatment sites ($G^2 = 46.27, df = 4, P < 0.001$). The relative abundance of adult squirrels declined between 1994 and 1995 in both logging and control sites, however, this demography was restored by 1996 (see figure below).

Step 5 - Summarize the adult odds ratios for year pairs according to the logging treatment.

```
> control.tab <- xtabs(COUNT ~ AGE + YEAR, data = taulman, subset =
+     TREAT == "CONTROL")
> library(biology)
> taulman.or <- oddsratios(t(control.tab), corr = T)
> plot(estimate ~ as.numeric(Comparison), data = taulman.or,
+     log = "y", type = "n", axes = F, xlab = "", ylab = "", ylim =
+         range(c(upper, lower)), xlim = c(0.5, 3.5))
> with(taulman.or, points(as.numeric(Comparison) + 0.1, estimate,
+     type = "b", pch = 21))
> with(taulman.or, arrows(as.numeric(Comparison) + 0.1, upper,
+     as.numeric(Comparison) + 0.1, lower, ang = 90, length = 0.1,
+     code = 3))
> harvest.tab <- xtabs(COUNT ~ AGE + YEAR, data = taulman, subset =
+     TREAT == "HARVEST")
> taulman.or <- oddsratios(t(harvest.tab), corr = T)
> points(estimate ~ Comparison, data = taulman.or, type = "b",
+     pch = 16)
> with(taulman.or, arrows(as.numeric(Comparison), upper,
+     as.numeric(Comparison), lower, ang = 90, length = 0.1,
          code = 3))
> abline(h = 1, lty = 2)
> axis(1, at = as.numeric(taulman.or$Comparison),
       lab = taulman.or$Comparison)
> axis(2, las = 1, cex.axis = 0.75)
> mtext("Year", 1, line = 3)
> mtext("Odds ratio of adults", 2, line = 3)
> legend("topright", legend = c("Logging", "Control"), pch = c(16,
+     21), bty = "n", title = "Sex")
> box(bty = "l")
```



Example 17F: Generalized additive models

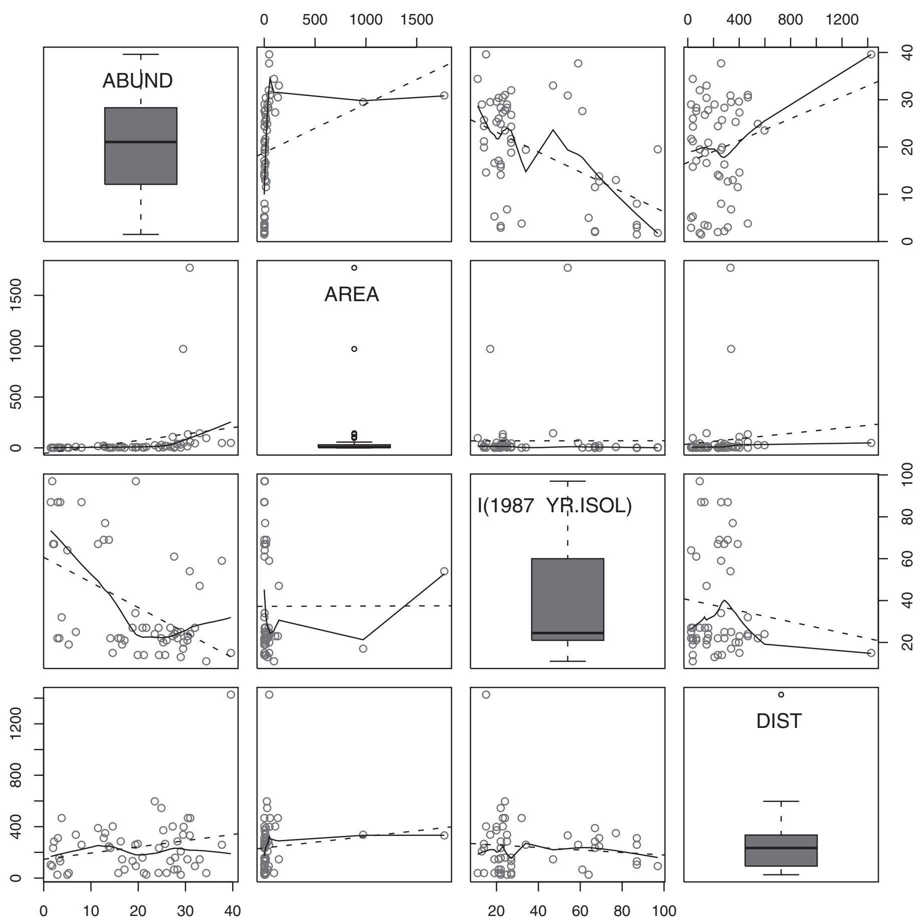
Quinn and Keough (2002) used a subset of the Loyn (1987) bird abundances across fragmented landscapes data to illustrate generalized additive models. Whilst this example is suboptimal in that the fitting of a generalized additive model cannot be entirely justified over a simpler multiple linear regression, there are no more suitable examples throughout the common biostatistics literature (Box 13.3 of Quinn and Keough (2002))[o]

Step 1 - Import (section 2.3) the Loyn (1987) data set

```
> loyn <- read.table("loyn.csv", header = T, sep = ",")
```

Step 2 (Key 9.3) - Investigate the assumptions of normality, predictor linearity (multi)collinearity using a scatterplot matrix.

```
> scatterplot.matrix(~ABUND + AREA + I(1987 - YR.ISOL) + DIST,
+     data = loyn, diag = "boxplot")
```



[o] Note that in Example 9A, years of isolation was effectively treated as the date (year) that patches became isolated, whereas in this example it will be treated as the number of years that fragments have been isolated up to 1987.

Conclusions - there is no evidence of non-normality in the response variable (bird abundance) and therefore a Gaussian probability distribution (and identity link function) is appropriate. Consistent with Quinn and Keough (2002), $\log_{10}$ transformations of fragment area and distance to the nearest patch were applied and improve normality of those variables. As previously indicated, linear conformity would normally mean that a generalized additive model approach is not necessary (or even appropriate) for these data. Nevertheless, concordant with Quinn and Keough (2002), the additive model incorporating Loess smoothing functions for each variable will be fit to the data.

Step 3 (Key 9.3) - Confirm the assumptions of (multi)collinearity for the form of model using variance inflation. Note, (multi)collinearity is investigated as if for a regular linear or generalized linear model.

```
> library(car)
> vif(glm(ABUND ~ log10(AREA) + I(1987 - YR.ISOL) + log10(DIST),
+     family = gaussian, data = loyn))
      log10(AREA) I(1987 - YR.ISOL)       log10(DIST)
         1.207990          1.098115          1.114780
```

Step 4 (Key 17.7) - Fit a generalized additive model (GAM) with a Gaussian probability distribution and nonparametric smoothers. Note, to perform an analysis equivalent to the one presented by Quinn and Keough (2002) (who fitted the GAM using S-Plus), we will use the `gam` *package* version of the `gam` *function*. Alternatively, a more sophisticated GAM can be fitted using the `gam` *function* from the mgcv *package*[p].

```
> library(gam)
> loyn.gam <- gam(ABUND ~ lo(log10(AREA)) + lo(I(1987 - YR.ISOL)) +
+     lo(log10(DIST)), family = gaussian, data = loyn)
```

Step 5 (Key 17.3) - Check the (lack of) fit via Deviance (G^2)

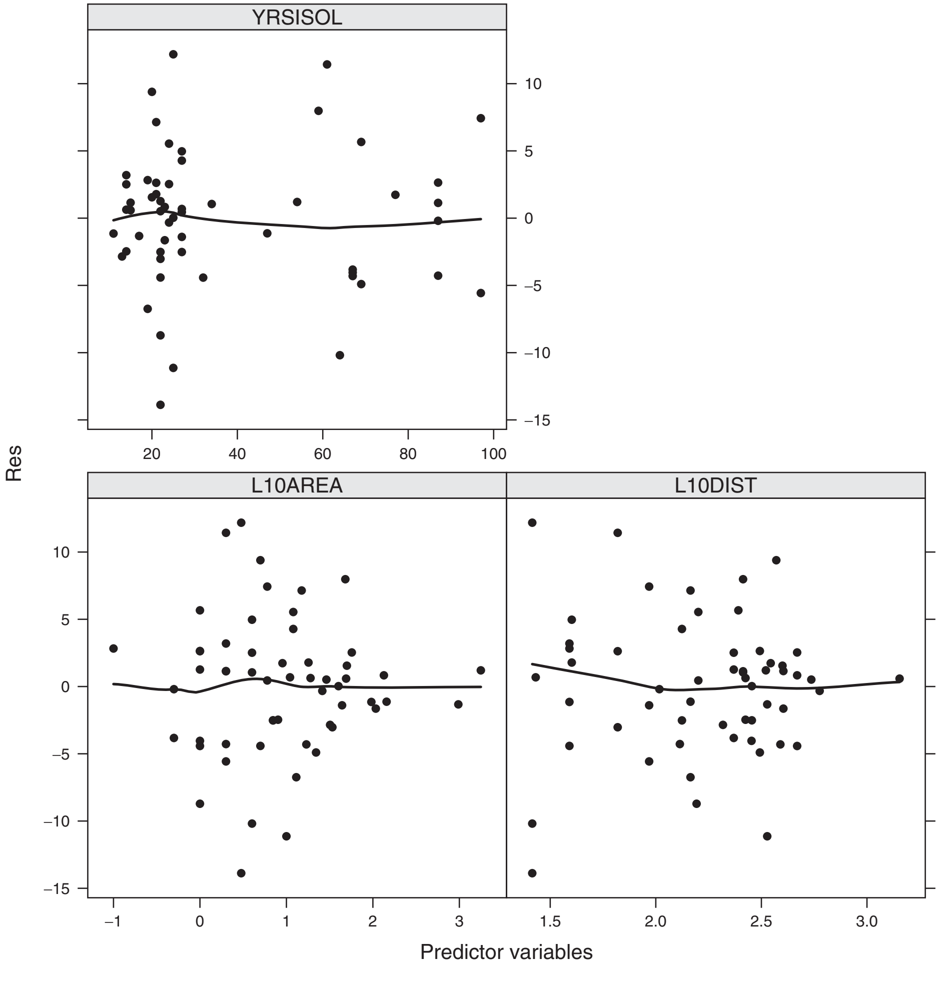
```
> paste("Deviance:", format(loyn.gam$deviance))
[1] "Deviance: 1454.26"
> 1 - pchisq(loyn.gam$deviance, loyn.gam$df.resid)
[1] 0
```

Examine the residuals associated with each of the predictor variables.

```
> # extract the Pearson's residuals from the fitted gam
> loyn.Res <- residuals(loyn.gam, "pearson")
> #generate a data frame with all transformations
> loyn.mod <- with(loyn, data.frame(ABUND, L10AREA = log10(AREA),
+ YRSISOL = I(1987 - YR.ISOL), L10DIST = log10(DIST)))
> # rearrange this data frame such that each of the predictors
> # become levels of a factor vector
```

[p]
```
> library(mgcv)
> loyn.gam <- gam(ABUND ~ s(log10(AREA)) + s(I(1987 - YR.ISOL)) + s(log10(DIST)),
    family = gaussian, data = loyn).
```

```
> loyn.L <- reshape(loyn.mod, direction = "long", varying =
+ list(c(2, 3, 4)), timevar = "Predictor", v.names = "Var", times =
+ names(loyn.mod[, c(2, 3, 4)]))
> # add the residuals to this data frame
> loyn.L$Res <- rep(loyn.Res, 3)
> # construct a lattice graphic
> library(lattice)
> print(xyplot(Res ~ Var | Predictor, data = loyn.L, scales = list(
+ alternating = TRUE, x = list(relation = "free")), xlab=
+ "Predictor variables", panel = function(x,y) {
+  panel.points(x, y, col = 1, pch = 16)
+  panel.loess(x, y, lwd = 2, col = 1)
+  }
+  ))
```



Conclusions - no evidence to suggest that the model did not fit. Note, it is not necessary to investigate overdispersion as this is not an issue for models fitted with a gaussian distribution. the data.

Step 6 (Key 17.7) - Examine the parameter estimates from the fitted GAM.

```
> summary(loyn.gam)
Call: gam(formula = ABUND ~ lo(log10(AREA)) + lo(I(1987 - YR.ISOL))
   + lo(log10(DIST)), family = gaussian, data = loyn)
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-13.8785  -2.8945   0.5522   2.5558  12.1815

(Dispersion Parameter for gaussian family taken to be 35.882)

    Null Deviance: 6337.929 on 55 degrees of freedom
Residual Deviance: 1454.26 on 40.529 degrees of freedom
AIC: 374.2495

Number of Local Scoring Iterations: 2

DF for Terms and F-values for Nonparametric Effects

                       Df Npar Df Npar F   Pr(F)
(Intercept)           1.0
lo(log10(AREA))       1.0     4.2 1.8169 0.14213
lo(I(1987 - YR.ISOL)) 1.0     3.3 0.6189 0.62008
lo(log10(DIST))       1.0     4.1 2.5767 0.05115 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Conclusions - as expected from the linearity displayed in the scatterplot matrix, none of the nonparametric terms fit the data significantly greater than their parametric equivalents (although log_{10} distance is close).

Step 7 - Quinn and Keough (2002) compared the fit of the full model to a series of reduced models (each omitting a single predictor variable) as a way of investigating the importance of each of the factors.

- Patch area

```
> loyn.gam1 <- update(loyn.gam, ~. - lo(log10(AREA)), family =
+     gaussian, data = loyn)
> anova(loyn.gam, loyn.gam1, test = "F")
Analysis of Deviance Table

Model 1: ABUND ~ lo(log10(AREA)) + lo(I(1987 - YR.ISOL))
      + lo(log10(DIST))
Model 2: ABUND ~ lo(I(1987 - YR.ISOL)) + lo(log10(DIST))
```

```
  Resid. Df Resid. Dev      Df Deviance      F    Pr(>F)
1   40.5290     1454.3
2   45.6833     3542.6 -5.1543  -2088.3 11.291 6.021e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Years of isolation

```
> loyn.gam2 <- update(loyn.gam, ~. - lo(I(1987 - YR.ISOL)),
+     family = gaussian, data = loyn)
> anova(loyn.gam, loyn.gam2, test = "F")
Analysis of Deviance Table

Model 1: ABUND ~ lo(log10(AREA)) + lo(I(1987 - YR.ISOL))
      + lo(log10(DIST))
Model 2: ABUND ~ lo(log10(AREA)) + lo(log10(DIST))
  Resid. Df Resid. Dev      Df Deviance      F  Pr(>F)
1   40.5290    1454.26
2   44.7957    1872.38 -4.2666  -418.12 2.7311 0.03912 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Distance to the nearest patch

```
> loyn.gam3 <- update(loyn.gam, ~. - lo(log10(DIST)), family =
+     gaussian, data = loyn)
> anova(loyn.gam, loyn.gam3, test = "F")
Analysis of Deviance Table

Model 1: ABUND ~ lo(log10(AREA)) + lo(I(1987 - YR.ISOL))
      + lo(log10(DIST))
Model 2: ABUND ~ lo(log10(AREA)) + lo(I(1987 - YR.ISOL))
  Resid. Df Resid. Dev      Df Deviance      F Pr(>F)
1   40.5290    1454.26
2   45.5791    1795.78 -5.0501  -341.52 1.8847 0.1177
```

Conclusions - bird abundance in fragmented landscapes is significantly effected by the size and duration of isolation of the habitat patches, but not the distance between patches.

Step 8 (Key 17.8) - Select the most parsimonious model relating bird abundances to the landscape variables.

```
> library(MuMIn)
> dredge(loyn.gam)
Model selection table
  (Intr)  l(I(-Y l(10(A l(10(D k Dev.   AIC  AICc  delta weight
8  20.62 -0.1223  9.107 -2.271 5 1454 374.2 375.4  0.000  0.614
5  16.41 -0.1224  8.221        4 1796 376.0 376.7  1.297  0.321
7  16.23          10.720 -3.026 4 1872 379.9 380.7  5.203  0.046
```

```
3  10.40          9.778        3 2326 381.9 382.4  6.923  0.019
2  27.38 -0.2112               3 4087 411.7 412.2 36.710  0.000
6  20.57 -0.2178         3.188 4 3543 413.8 414.6 39.140  0.000
1  19.51                       2 6338 427.7 428.0 52.520  0.000
4  12.23                 3.287 3 5779 432.7 433.1 57.680  0.000
```

Conclusions - the model with all three predictor variables has the lowest AIC (and AIC_C). However, the delta for the model with patch area and years of isolation is less than two units, indicating that this latter model is not significantly less parsimonious than the former model.

Bibliography

Allison, T., and D. V. Cicchetti. (1976). Sleep in mammals: ecological and constitutional correlates. *Science* **194**, 732–734.

Andrew, N. L., and A. J. Underwood. (1993). Density-dependent foraging in the sea urchin *Centrostephanus rodgersii* on shallow subtidal reefs in New South Wales, Australia. *Marine Ecology Progress Series* **99**, 89–98.

Bolger, D. T., A. C. A., R. M. Sauvajot, P. Potenza, C. McCalvin, D. Tran, S. Mazzoni, and M. E. Soule. (1997). Response of Rodents to Habitat Fragmentation in Coastal Southern California. *Ecological Applications* **7**, 552–563.

Christensen, D. L., B. R. Herwig, D. E. Schindler, and S. R. Carpenter. (1996). Impacts of lakeshore residential development on couarse woody debris in north temperature lakes. *Ecological Applications* **64**, 1143–1149.

Constable, A. J. (1993). The role of sutures in shrinking of the test in *Heliocidaris erythrogramma* (Echinoidea: Echiniometridae). *Marine Biology* **117**, 423–430.

Crawley, M. J. (2002). *Statistical computing: an introduction to data analysis using S-PLUS*. John Wiley & Sons, New York.

Crawley, M. J. (2007). *The R Book*. John Wiley & Sons, New York.

Dalgaard, P. (2002). *Introductory Statistics with R*. Springer-Verlag, New York.

Doncaster, C. P., and A. J. H. Davey. (2007). *Analysis of Variance and Covariance. How to Choose and Construct Models for the Life Sciences*. Cambridge University Press, Cambridge.

Driscoll, D. A., and J. D. Roberts. (1997). Impact of fuel-reduction burning on the frog *Geocrinia lutea* in southwest Western Australia. *Australian Journal of Zoology* **22**, 334–339.

Elgar, M. A., R. A. Allan, and T. A. Evans. (1996). Foraging strategies in orb-spinning spiders: ambient light and silk decorations in *Argiope aetherea Walckenaer* (Araeneae: Araneoidea). *Australian Journal of Ecology* **21**, 464–467.

Ellison, A. M. (2004). Bayesian inference in ecology. *Ecology Letters* **7**, 509–520.

Everitt, B. S. (1994). *A handbook of statistical analyses using S-Plus*. Chapman & Hall, Boca Raton, FL.

Faraway, J. J. (2006). *Extending Linear Models with R: generalized linear mixed effects and nonparametric regression models*. Chapman & Hall/CRC, Boca Raton, FL.

Fowler, J., L. Cohen, and P. Jarvis. (1998). *Practical statistics for field biology*. John Wiley & Sons, New York.

Fox, J. (2002). *An R and S-PLUS Companion to Applied Regression*. Sage Books, Thousand Oaks, CA.

Furness, R. W., and D. M. Bryant. (1996). Effect of wind on field metabolic rates of breeding Northern Fulmars. *Ecology* **77**, 1181–1188.

Gotelli, N. J., and A. M. Ellison. (2002). Biogeography at a regional scale: determinants of ant species density in bogs and forests of New England. *Ecology* **83**, 1604–1609.

Green, P. T. (1997). Red crabs in rain forest on Christmas Island, Indian Ocean: activity patterns, density and biomass. *Journal of Tropical Ecology* **13**, 17–38.

Hall, S. J., S. A. Gray, and Z. L. Hammett. (2000). Biodiversity-productivity relations: an experimental evaluation of mechanisms. *Oecologia* **122**, 545–555.

Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized Additive Models*. Chapman & Hall, Boca Raton, FL.

Hollander, M., and D. A. Wolfe. (1999). *Nonparametric statistical methods, 2nd edition*. 2 edition. John Wiley & Sons, New York.

Ihaka, R., and R. Gentleman. (1996). R: A Language for Data Analysis and Graphics. *Journal of Computational and Graphical Statistics* **5**, 299–314.

Keough, M. J., and P. T. Raimondi. (1995). Responses of settling invertebrate larvae to bioorganic films: effects of different types of films. *Journal of Experimental Marine Biology and Ecology* **185**, 235–253.

Kirk, R. E. (1968). *Experimental design: procedures for the behavioral sciences*. Brooks/Cole, Monterey, CA.

Legendre, P., (2001). Model II regression - User's guide. Départment de sciences biologiques, Université de Montréal.

Loyn, R. H., (1987). *Nature Conservation: the Role of Remnants of Native Vegetation*, Chapter effects of patch area and habitat on bird abundances, species numbers and tree health in fragmented victorian forests. Surrey Beatty & Sons, Chipping Norton, NSW.

Mac Nally, R. M. (1996). Hierarchical partitioning as an interpretative tool in multivariate inference. *Australian Journal of Ecology* **21**, 224–228.

Maindonald, J. H., and J. Braun. (2003). *Data Analysis and Graphics Using R – An Example-based Approach*. Cambridge University Press, London.

Manly, B. F. J. (1991). *Randomization and Monte Carlo methods in biology*. Chapman & Hall, London.

McGoldrick, J. M., and R. C. Mac Nally. (1998). Impact of flowering on bird community dynamics in some central Victorian eucalypt forests. *Ecological Research* **13**.

McKechnie, S. W., P. R. Ehrlich, and R. R. White. (1975). Population genetics of *Euphdryas* butterflies. I. Genetic variation and the neutrality hypothesis. *Genetics* **81**, 571–594.

Medley, C. N., and W. H. Clements. (1998). Responses of diaton communities to heavy metals in streams: the influence of longitudinal variation. *Ecological Applications* **8**, 663–644.

Milliken, G. A., and D. E. Johnson. (1984). *Analysis of messy data. Volume I: Designed Experiments*. Van Nostrand Reinhold, New York.

Minchinton, T. E., and P. M. Ross. (1999). Oysters as habitat for limpets in a temerate mangrove forest. *Australian Journal of Ecology* **24**, 157–170.

Mullens, A., (1993). The effects of inspired oxygen on the pattern of ventilation in the Can Toad (*Bufo marinus*) and the Salt Water Crocodile (*Crocodylus porosus*). Honours thesis, University of Melbourne, Australia.

Murrell, P. (2005). *R Graphics (Computer Science and Data Analysis)*. Chapman & Hall/CRC.

Nelson, V. E., (1964). The effects of starvation and humidity on water content in *Tribolium confusum* Duval (Coleoptera). Ph.D. thesis, University of Colorado.

Partridge, L., and M. Farquhar. (1981). Sexual activity and the lifespan of male fruitflies. *Nature* **294**, 580–581.

Paruelo, J. M., and W. K. Lauenroth. (1996). Relative abundance of plant functional types in grasslands and shrublands of North America. *Ecological Applications*, pp. 1212–1224.

Peake, A. J., and G. P. Quinn. (1993). Temporal variation in species-area curves for invertebrates in clumps of an intertidal mussel. *Ecography* **16**, 269–277.

Pinheiro, J. C., and D. M. Bates. (2000). *Mixed effects models in S and S-PLUS*. Springer-Verlag, New York.

Polis, G. A., S. D. Hurd, C. D. Jackson, and F. Sanchez-Piñero. (1998). Multifactor population limitation: variable spatial and temporal control of spiders on Gulf of California islands. *Ecology* **79**, 490–502.

Powell, G. L., and A. P. Russell. (1984). The diet of the eastern short-horned lizard (*Phrynosoma douglassi brevirostre*) in Alberta and its relationship to sexual size dimorphism. *Canadian Journal of Zoology* **62**, 428–440.

Powell, G. L., and A. P. Russell. (1985). Growth and sexual size dimorphism in Alberta populations of the eastern short-horned lizard *Phrynosoma douglassi brevirostre*. *Canadian Journal of Zoology* **63**, 139–154.

Quinn, G. P. (1988). Ecology of the intertidal pulmonate limpet *Siphonaria diemenensis* Quoy et Gaimard. II Reproductive patterns and energetics. *Journal of Experimental Marine Biology and Ecology* **117**, 137–156.

Quinn, G. P., and K. J. Keough. (2002). *Experimental design and data analysis for biologists*. Cambridge University Press, London.

R Development Core Team, (2005). R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org.

Reich, P. B., D. S. Ellsworth, M. B. Walters, J. M. Vose, C. Gresham, J. C. Volin, and W. D. Bowman. (1999). Generality of leaf trait relationships: a test across six biomes. *Ecology* **80**, 1955–1969.

Roberts, J. (1993). Regeneration and growth of coolibah, *Eucalyptus coolibah* subsp. *arida*, a riparian tree, in the Cooper Creek region of South Australia. *Australian Journal of Ecology* **18**, 345–350.

Sánchez-Piñero, F., and G. A. Polis. (2000). Bottom-up dynamics of allochthonous input: direct and indirect effects of seabirds on islands. *Ecology* **81**, 3117–3132.

Sinclair, A. R. E., and P. Arcese. (1995). Population consequences of predation-sensitive foraging: the Serengeti wildebeest. *Ecology* **76**, 882–891.

Smith, E. J. (1967). Cloud seeding experiments in Australia. *Procedings of the 5th Berkley Symposium* **5**, 161–176.

Smith, F. (1939). A genetic analysis of red-seed coat color in *Phaseolus vulgaris*. *Hilgardia* **12**, 553–621.

Sokal, R., and F. J. Rohlf. (1997). *Biometry, 3rd edition*. W. H. Freeman, San Francisco.

Taulman, J. F., K. G. Smith, and R. E. Thill. (1998). Demographic and behavioral responses of southern flying squirrels to experimental logging in Arkansas. *Ecological Applications* **8**, 1144–1155.

Venables, W. N., and B. D. Ripley. (2002). *Modern Applied Statistics with S-PLUS, 4th edn*. Springer-Verlag, New York.

Walter, D. E., and D. J. O'Dowd. (1992). Leaves with domatia have more mites. *Ecology* **73**, 1514–1518.

Ward, S., and G. P. Quinn. (1988). Preliminary investigations of the ecology of the predatory gastropod *Lepsiella vinosa* (Lemarck) (Gastropoda Muricidae). *Journal of Molluscan Studies* **73**, 109–117.

Wilcox, R. R. (2005). *Introduction to Robust Estimation and Hypothesis Testing*. Elsevier Academic Press, New York.

Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, Boca Raton, FL.

Young, R. F., and H. E. Winn. (2003). Activity patterns, diet, and shelter site use for two species of moray eels, *Gymnothorax moringa* and *Gymnothorax vicinus*, in Belize. *Copeia* **2003**, 44–55.

Zar, G. H. (1999). *Biostatistical methods*. Prentice-Hall, New Jersey.

Zuur, A. F., E. N. Ieno, N. J. Walker, A. A. Saveliev, and G. M. Smith. (2009). *Mixed Effects Models and Extensions in Ecology with R*. Springer, New York.

R Index

Statistics Index