Modern Analysis of Biological Data. Generalized Linear Models in R

Book · January 2016

CITATIONS
59

READS
2,337

2 authors:

Stano Pekar
Masaryk University
232 PUBLICATIONS **4,094** CITATIONS
SEE PROFILE

Marek Brabec
The Czech Academy of Sciences
170 PUBLICATIONS **1,620** CITATIONS
SEE PROFILE

MODERN ANALYSIS OF BIOLOGICAL DATA

GENERALIZED LINEAR MODELS IN R

STANO PEKÁR
MAREK BRABEC

Masaryk University, Brno 2016

http://www.muni.cz/press/books/pekar_en

Pekár S. & Brabec M. 2016. Modern Analysis of Biological Data:
Generalized Linear Models in R. Masaryk University Press, Brno.

This book was supported by Masaryk University Project No. MUNI/FR/1304/2014.

ISBN 978-80-210-8019-5

CONTENTS

FOREWORD

This book is meant especially for students and scholars of biology, i.e. biologists who work in natural science, at agricultural, veterinary, pharmaceutical and medical faculties or at research institutes of a similar orientation. It has been written for people who have only a basic knowledge of statistics (for example, people who have attended only a Basic statistics/Biostatistics course) but who need to correctly analyse the data resulting from their observations or experiments.

The generally negative attitude of biologists towards mathematics is well known. It is precisely why we have tried to write the book in a relatively simple style – with minimal mathematical requirements. Sometimes, the task turned out to be easy, other times not that easy, and sometimes it became almost impossible. That is why there are still some mathematical equations in almost all chapters of the book (even though they are used in a simplified form in order to be more apprehensible to less experienced readers). Despite this fact, the book includes much less mathematical and statistical theories than it is common for standard statistical literature.

The book is mainly built on examples of real data analyses. They are presented from the very beginning to the end, from a description and determination of objectives and assumptions to study conclusions. They thus simulate (even though in a simplified way) the procedure usually used when preparing a paper for a scientific journal. We believe that practical experience with data analyses is irreplaceable. Because of the anticipated biology-oriented readers, we selected examples from the areas of ecology, ethology, toxicology, physiology, zoology and agricultural production. All these data were analysed during various previous research projects. They have been adjusted in order to suit the pedagogical intentions of this book. For example, the original long and complex Latin names of species have been replaced with a generic short name (e.g., specA).

Finally, we would like to thank all our colleagues without whose help this book would never have been written. First of all, we would like to thank to Vojtěch Jarošík (in memoriam), for introducing GLM to the first author of the book during his studies at the university, thus igniting his interest in statistics generally; Alois Honěk for many consultations, and our colleagues from the Crop Research Institute in Prague-Ruzyně and the students of the Faculty of Science of the Masaryk University in Brno for inspiring comments to the original text of the book. Finally, we would also like to thank the following colleagues of ours who have kindly let us use their (though adjusted) data for presenting examples in this book: T. Bilde, A. Honěk, J. Hubert, D. Chmelař, J. Lipavský, M. Řezáč, P. Saska and V. Stejskal.

We welcome any comments regarding the text and content of the book. Please direct them to the following email addresses: pekar@sci.muni.cz and/or mbrabec@cs.cas.cz.

December 2015

Stano Pekár
Marek Brabec

1 INTRODUCTION

Let us start with a demonstration of two standard situations. The first one took place after a thesis defence when a student complained to another student: "Supposedly I used the wrong statistical test." Other situations can occur, for example, in a hallway of a research institute where a biologist reproaches a colleague for the way he/she presented his/her results: "The data should be analysed more properly." Both situations have one thing in common – a desperate reference to the statistics. Indeed, statistical data analysis forms an integral part of scientific publications in many biological (and other) fields, thus accompanying bio-logists throughout their careers. In some fields, such as taxonomy, statistical analyses may play just a marginal role. For other fields, such as ecology or physiology, it is often almost a cornerstone of many new findings. In these fields, shortcomings in statistical analysis can have catastrophic consequences. It can easily happen that a report on a good experimental study (e.g., in the form of a scientific paper) will not be complete without a statistical analysis, in which case the results of the study can become completely useless and all the previous effort of its authors can thus be wasted.

The only way to prevent such disasters is to strive to understand the statistics or, at least, to find somebody who understands it. Obviously, conducting practical data analyses are much easier today than ever before. This is due to the development of personal computers and subsequent developments in computational algorithms, which have literally meant a revolution for data analyses. Fifty years ago, even a simple statistical analysis, using a calculator and a pen, would take several hours and sometimes even days (while, at the same time, it was not easy to avoid calculation errors etc.). Today, using computers, even a relatively complicated analysis can take less than a minute or even just a few milliseconds. Preparing a plot is often easier and faster than, for example, preparing a coffee. However, technical improvements have led to increased demands for using adequate statistical methods. While simpler statistical methods were preferred in the past, despite the fact they were not the most suitable, today the emphasis is put on using methods that correspond in the relevant aspects to the actual data at hand as closely as possible. This is because the computing requirements do not represent an insurmountable obstacle any more. Unlike the former "universal" simple procedures, application of statistics today utilises such methods and models that realistically consider the important characteristics of the data and of the studied processes. Very often this means that a given method or a statistical model can and should be adjusted to the real situation at hand. In short, models should be adjusted to the data and not vice versa!

Nevertheless, it is obviously not always easy to comply with this requirement. In fact, creative and useful practical analyses need theoretical knowledge about numerous models and methods as a pre-requisite. Moreover, certain experience with practical data analyses, with

the application of various models on real data, with model building as well as estimation procedures, with conducting appropriate tests, etc. is also necessary. It is clear that one cannot become an experienced data analyst only from reading books. To get such experience, you have to put in some work and, most of all, some thinking. Guidance and examples can assist you in this process.

This book attempts to help exactly along these lines by presenting examples of particular analyses, including the specification of the problem of interest, development of a statistical model and formulation of possible conclusions. The book is not, and does not even aspire to be, a manual for selecting the "best" method for a particular data analysis (it is our strong opinion that, for various reasons, such a manual cannot be ever made). Instead, the book tries to demonstrate how to think about particular statistical models, how to use them and also how not to use them – to point out many of the problems and errors that can occur (and do indeed occur in practical analyses and even in published papers). We demonstrate various general approaches on particular (hopefully biologist-engaging) examples and we also present their detailed implementation in the R language, thus allowing everybody to try them for themselves using the data provided in the book as well as their own datasets of a similar nature.

Since regression is a very powerful instrument, used very often in biological studies, this book is almost exclusively dedicated to regression models. It assists in solving important questions of the following type: how does variable y depend on variable x or how can we predict the value of a new observation when we know the value of one or several so-called explanatory variables. However, we will talk about regression in a somewhat wider context compared to what you know from basic statistics courses. Namely, we will use the concept of the so-called Generalized Linear Models (GLM). Their name explicitly emphasises that they are generalized linear regression models – that is, generalisations of models that many people know under the term of general linear model. The GLM generalization is very useful from a practical point of view since it allows correct analyses of various data that cannot be processed using a standard (linear) regression without (very) rough approximations and (crude) simplifications. GLM represents a relatively large class of models with a unified statistical theory and very universal computational implementations. It is a class by the means of which you can analyse a wide range of data, such as weight measurements, concentration, number of individuals, or relative frequency of a phenomenon (i.e. various continuous unrestricted, continuous positive, discrete positive data).

From a formal point of view, we will use only univariate GLM models (or statistical methods based on them), and only those that are suitable exclusively for independent measurements (observations). In order to analyse data with some kind of correlation (for example, in the case of repeated measurements on the same individuals), somewhat different and more complex models and methods need to be used. We address these in another book (Pekár & Brabec 2012).

As we have already stated, the objective of this book is to assist practical users of statistics in the process of formulating and using statistical models – and thus also in selecting suit-

able methods for analyses of various data types. At the same time, we would like to motivate them to seek assistance from a professional statistician if they find that the complexity of the model, study design and/or data complexity exceed their abilities and experience with statistical modelling. Things are not always as easy as they may appear at first. The same data can be analysed using various methods, depending, for example, on the objectives of a particular analyst. In fact, data include a large amount of information but we are often interested only in extracting just some part of it at a given moment. That is why the method selected depends on what we regard as salient features of the data (and what we are willing to leave out), as well as what (and for what purposes) we want to extract from the observations/measurements. Nevertheless, it is often the case that multiple procedures (with different characteristics) can be used for analysing even a single aspect. In this book, we will mostly present only one procedure without denying the existence of other approaches. Otherwise, the book would become significantly longer.

We have applied a similar approach when describing the R language and discussing the use of particular objects. The R language is, in fact, very rich. Like in any other programming language environment, the same result can be often obtained in several ways. There is not enough room here to list them all (if you are interested, you can consult the corresponding manuals, e.g. Zoonekynd 2007). We chose those that we consider to be the easiest and most practical for a beginner.

1.1 How to read the book

The text of this book combines examples of selected statistical methods and descriptions of the R language as a statistical environment. Both are then illustrated on practical data analyses.

How to read this book? It depends on your knowledge and experience with statistical analyses as well as with the R environment. Those of you who have not done any (or almost any) data analysis since they attended their basic statistics/biostatistics course and who have never worked with R before should read the book from the beginning to the end. You who already know the environment R (at least the basics) and have been using regression in your work, can read the book somewhat non-systematically – picking the chapter that deals with data and/or methods you are currently interested in.

The most important part of the first chapter is the section 1.2, which defines variables. You certainly should not skip this part even if you believe that you are clear about distinguishing among variable types. This is because our definitions may differ from the definitions you may have encountered in other books and/or courses.

Chapter 2 describes the installation and use of the R software, which we will be using in this book for various data analyses. If you have never heard about R (or you have heard about it but never actually worked with it), this chapter is here especially for you. Apart from the installation of the software, you will also learn both some general principles for working in the R environment and important commands used repeatedly for various data manipulations later

in this book. Moreover, the chapter explains how to enter data into the R environment. All of the above is absolutely essential for being able to use the software successfully, not only for practicing the examples of this book, but also for practical data analyses of your own.

Chapter 3 shows some of the methods of so-called exploratory data analysis. This analysis means calculation of the basic statistical characteristics of the examined data sets as well as their informal comparison using plots, tables and other instruments. You will find there a general description of several useful R commands, by means of which you can create the majority of plots, with which we will work in later chapters.

The next four chapters (Chapters 4-8) are oriented rather more generally and even include some theory. Do not be misled and do not skip them thinking that they are not important from a practical point of view! In fact, they are some of the most important ones. This is because you can use their content for data analyses even if you use different software or, for example, for general considerations related to the strategy of statistical analyses. Particularly, Chapter 4 addresses the issue of how to work with statistical models of the type that will be used later in the book. Model formulation (traditionally in the form of a mathematical formula) forms a steppingstone, on which statistical analysis is based. In this chapter, we will talk about regression models and we will discuss what GLM actually is.

Chapter 5 presents the first example of a concrete and non-trivial data analysis. The presented example is not as simple as motivational examples usually are. It is more demanding, which allows us to discuss more of the aspects you will encounter when working with other examples (and in practical analyses of your own). The analysis is described in detail and commented abundantly (with references to the basic rules, which analysts should observe). In addition to the previous, some R outputs are described and interpreted and important definitions are stated (for example, the definition of contrasts).

Chapter 6 then goes deeper and deals with various systematic parts of a GLM model. It describes basic model classification based on the character of the explanatory variables. However, most importantly, it demonstrates in general terms how to translate a mathematical model into the R language and suggests how to interpret the model after the analysis is done.

Chapter 7 can be used as a simple "key", based on which readers (beginners) can try to decide which particular method to use (you will see yourself that, once you acquire more knowledge and experience, this decision will require even more thinking). We assume that the reader will be coming back to the book to use it as an aid for his/her own data analyses. If the reader remembers the general knowledge from Chapters 1–6, he/she can go straight to Chapter 7, which will direct him/her to a chapter where he/she will find an analysis of an example similar to his/her own; however, if you cannot decide which method to use, continue reading subsequent chapters in the order they are presented. Either way, we certainly recommend that you initially read Chapters 8–12 completely.

Chapters 8–12 are based on several examples, but they explain also theory, which you will encounter during these (and other) analyses. In general, the analyses of all examples from

these chapters follow a similar plan. The chapters are written in a way that makes them mutually independent (they can be read separately). As a result of this, you encounter similar problems repeatedly. When you pick a chapter, you should always read it as a whole. Do not try to follow it merely as a concrete report of a particular analysis.

1.2 Types of variables

Definitions and a detailed description of the characteristics of various variable types form the topic of a basic statistical course, which we certainly do not intend to repeat here. Let us just remind you of a few important facts that will be the most useful later in the book.

There are several possible viewpoints based on which variables can be classified. For our purposes, we will only need the following variable types:

Response variable: a (random) variable, the variability of which we attempt to explain by means of a statistical (regression) model. In other words, it is a variable that we want to model using a single explanatory variable or multiple explanatory variables. In this book we will exclusively address univariate models. That is, we will always model only one random response variable at a time (in contrast to multivariate models with several response variables considered in one model simultaneously). Depending on the particular GLM model type, our response variable will be either continuous or discrete, but always numeric (i.e. the values are numbers).

Explanatory variable: a variable by the means of which we explain the values of a response variable. Even in the case of univariate models, a response variable can be modelled by a single explanatory variable or by multiple explanatory variables. These can be either numeric or categorical (the values are characters and character chains that correspond to a given code marking groups/categories). Numerical variables can be continuous or discrete. We will mark continuous explanatory variables using lowercase letters, for example x. We will mark their value for the ith observation using index i (e.g. x_i). We will call continuous variables **covariates**. We will mark **categorical** variables using uppercase letters (for example A). a categorical variable can always take at least two different values, which we call **levels** (for example, "male" and "female"). Then the jth level of such a variable is marked with index j, for example as A_j, a categorical explanatory variable (with categories denoted by characters or numbers) will be also called a **factor**, as in the analysis of variance (ANOVA).

Weights represent a special case of variable. They determine relative weights of individual observations within a given data set. By default, functions in R use the same (unitary) weights for all observations. Externally entered weights are useful when a scheme with equal weights is not satisfactory and we need to change it. For example, weights can be based on known relative accuracy of observations (when data are averages of samples of different size, the sample sizes are often taken as weights). The weights must take non-negative values. Zero weights are possible, but somewhat extreme (they exclude the given measurement from the analysis), and we do not recommend using them (selected observations can be excluded from the analysis in a much more elegant manner).

1.3 Conventions

The book uses several font types. We use them for distinguishing the basic text of the book from software commands (and other key words) of the R language. For the names of commands and their arguments, we use the bold **`Courier New`** font. The names of objects that we create during analyses are typed in the normal `Courier New` font. The other text is typed in the Times New Roman font. Names of variables, parameter values and mathematical formulas are written in *italics*, while the names of factor levels are enclosed in quotation marks. The names of packages are underlined.

For transcribing everything that takes place in the command window of the R environment, we use the `Courier New` font of a smaller size. For better orientation, we differentiate between commands entered by the user, which we write in bold (for example, **`a<-1:5`**), and program responses in normal style (for example, `a`). To save space, some rows of output were cut off and substituted with three dots.

Plots made at the beginning of analyses were created using as few commands as possible and that is why they often do not have appropriate captions, legends, etc. Only plots made at the end of the analysis are closer to real presentation-quality and hence include various details (at the cost of longer commands).

The use of the natural logarithm (with base e) is prevalent in the statistics. We will thus label it simply by log. You need to be aware of the fact that this symbol can have a different meaning elsewhere (for example, in MS Excel it means the decadic logarithm – with base 10).

2 STATISTICAL SOFTWARE

There are many commercial as well as non-commercial programs available for data analyses. They widely differ by their quality, extent and price. Many inexperienced users work just with the basic software, for example, MS Excel. Some of the popular specialised programs include Statistica or JMP. And finally, there are large (and expensive) packages, such as SPSS or SAS. The difference among statistical software packages is not only in the number of implemented analysis types, but also in their flexibility, i.e. in the possibilities of user programming and other "customised" modifications and the automation of selected procedures. A simple analysis can, of course, be executed using any of the statistical software, however, more advanced methods are offered only by specialised packages.

2.1 The R environment

The good news is that one of the best statistical software packages for modern data analyses is available completely for free. It is called R (R Core Team 2015) and this book is based on its extensive possibilities. It actually covers a much wider area than is presented in this book.

The R programming environment is similar to the commercial program S-Plus (© Insightful Corporation), which used the S programming language; it was developed in the 1980's in the American AT&T Bell Laboratories (it has been improved several times since then). R uses the dialect of the S language in combination with the Scheme language, which means that, from the user's point of view, the overwhelming majority of the language for R and S-Plus are either the same or similar. R was created by New Zealanders Ihaka & Gentleman (1996). Today, it is managed by a group of people from around the globe, who call themselves the R Core Team. Thanks to that, the development of R is extremely dynamic. It is constantly evolving and self-improving.

On its own, R is not a user-friendly program of the MS Excel or Statistica type. You will not find nice, colourful windows, pull-down menus and clickable buttons, basically an environment where the main control instrument is the mouse. Be prepared that when you open R, you will only see an empty grey window (Fig. 2-1). R is mainly controlled by commands entered on a keyboard.

There are specialised environments available which can provide a standard user higher comfort when using R (for example, the Rstudio downloadable from http://www.rstudio.com/). As we are concentrating on the GLM models and data analysis here, we will not use them in this book.

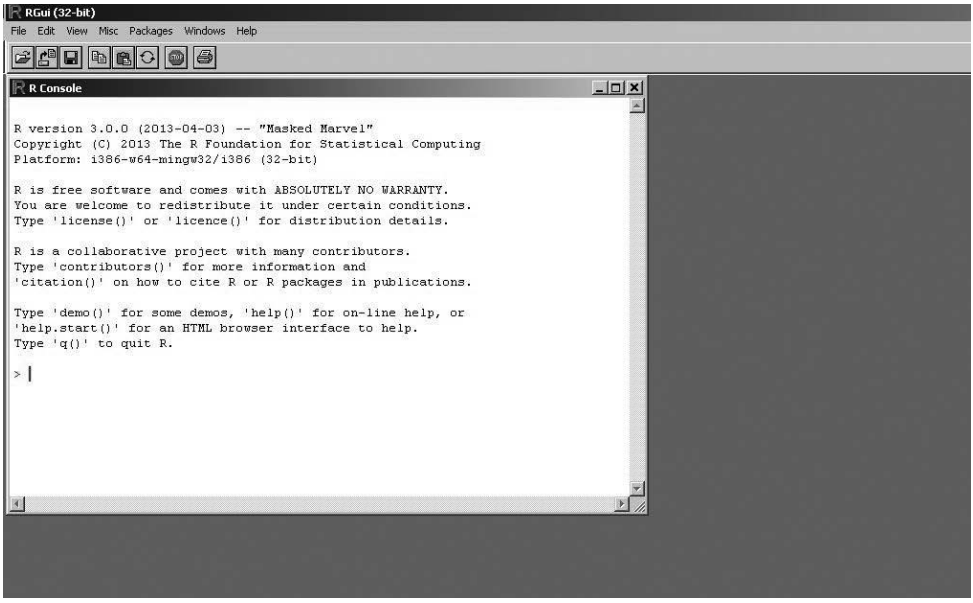


Fig. 2-1 Window of the R environment.

You may ask why we have chosen such "unfriendly" software. We have several reasons for it:

- R is one of the most extensive statistical packages that contain all essential types for modern analyses, which are continuously amended thanks to the continuous work of many people from around the world. Commercial statistical software is usually a closed system (it can be expanded only by purchasing yet another version), while R has been built as an open system. It means that new and additional methods can be added easily and for free at any time.
- Control of this program is not (for a non-statistician) that easy. However, that can be, paradoxically, an advantage because it forces its users to acquire certain knowledge about what they are doing and to think about it (at least) a little bit. Data analyses using commercial software can be somewhat "dangerous" since overly-friendly software allows people who have no idea what they are doing to perform an analysis by a sequence of more or less random clicks. Needless to say that the results of such analyses are often misleading or completely wrong.
- Yet another reason is that "friendly" statistical software will produce a huge volume of information, which they will spout out to you once the analysis is completed. It may not be easy for you to make sense of it. The philosophy of R is completely different – it will display only what the user asks for with the commands. This approach is based on the assumption that users will only use commands that they know something about or, alternatively, that they will look for in help pages or reference literature, thus being stimulated to study the given topic. By default, R outputs are typically quite modest in size. And, in fact, that

is good indeed. As you will see, it is easier to make general sense of the analysis this way. At any rate, specific details can be explicitly requested from the R environment later, if needed.
- One of the strengths of R lies in powerful modern graphics, focused on transparent and efficient data presentation.

Taken together, R is certainly not a hostile program. You will see while reading this book and while analysing your own data that, quite to the contrary, R control is relatively easy and, most of all, very efficient. To make your learning easier, we have described and explained the way to call various important procedures (and their outputs) in detail.

As we have already stated above, R is an environment used for handling objects. Objects can be data, results of their analyses (or results of various intermediate computations and manipulations), and handling means mathematical and statistical computations, manipulations, and constructions of tables and plots, etc. In reality, R is more than just a statistical package: it is a powerful programming language (for object programming), similar to C, C+ or Java. In order to successfully use R, you need to, among other things, learn its basic control commands. Making you familiar with these basic commands is the main objective of this chapter. It is the basic R philosophy that you can learn more details later when you need them. However, first of all, you need to install R on your computer.

2.2 Installation and use of R

The R installation file can be found on the Internet at http://www.r-project.org/. You can get it via the Download dialogue box where, upon clicking on CRAN, a list of servers, from which you can actually download the file, is displayed. Upon selecting a server, a window comes up where you choose the platform on which you will work with R: Linux, Mac OS or MS Windows. All calculations in this book have been done in MS Windows (use of R in alternative platforms is quite similar). Apart from the latest version of the installation file of R, the base folder also includes several info files. What we need now is the installation file with EXE extension. New versions of the software are uploaded in a relatively regular manner every few months. At the time of writing this book, the latest available version was 3.0.0. The name of the installation file thus was **R-3.0.0-win.exe**. This version incorporates 29 basic packages (i.e. libraries that implement various statistical methods and other computations). There are several other packages that include additional methods and other useful commands (at the time of writing this book, there were more than a thousand of them). You can see them by clicking on Packages on the taskbar on the left. A list of packages with a brief description of included functions is then displayed. If you are interested in any of them, you can download them (of course for free and in unlimited number). We will talk about how to do it later.

Once you have successfully installed the software, an icon called **R 3.0.0** will appear on the desktop. Upon launching the program, the main (large) window opens with a command (smaller) window in it – this is the R Console (Fig. 2-1). The upper taskbar of the main window includes several basic commands accessible either via a pull-down menu or using a button.

We will describe the functions of the most important ones. In the File menu, you can find basic options, such as **Display file(s)...** for displaying files in the R-3.0.0 folder. This is a standard, pre-set working folder. It is quite convenient to store your data in this folder because then you do not have to specify the full path when you want to read/save them. To keep things organised, we have saved all data that we will work with in a folder called MABD (abbreviation of the name of the book), located in folder C:\Program Files\R\R-3.0.0. If you want to do the same, you have to first download the data files from http://www.muni.cz/press/books/pekar_en and save them in the folder called MABD. Then you can change the working directory via the **Change dir...** option from inside of the launched R program (using the File option from the main menu).

By selecting **Load Workspace...** (again using the **File** option from the main menu), you can upload your previous work, provided you saved it in the end of your previous session using the **Save Workspace...** option. These options are especially useful when you want to continue your analysis after an interruption, for example, on the next day. You can just upload what you did earlier and you thus do not have to do everything from the scratch again. Nevertheless, be careful! Only the commands and (result) objects are saved (uploaded), not outputs and graphics. This means that if you want to see an output from some of the previous commands, you have to call and launch it again (provided you have not already copied the text output into, for example, a text file using Copy and Paste).

The **Edit** menu includes the copy and inserts functions in the same way we are used to from other programs. Furthermore, the menu offers an option for clearing the console window: **Clear console**, and an option for opening the spreadsheet editor: **Data editor...**, and an option for modifying the program appearance: **GUI preferences...** Here, you can change the size of the windows, font type, background and font colour, etc.

The **Misc** menu includes the useful **Stop current computation** option for suspending a given computation (for example, when the computation "freezes" for some reason, or when you realise that you have entered an input incorrectly, etc.). This can be done even faster using a shortcut key – by pressing the ESC key or by clicking on the STOP button. The **Remove all objects** option is used for removing all currently defined or uploaded objects.

The **Packages** menu offers options for working with additional packages, for example those that contain additional functions. Packages (with the exception of a few basic ones) are not automatically loaded into memory upon launching the program so that they are not occupying it with functions that we do not need at a given moment. When you want to use a function included in a certain package, you need to load the package by selecting the **Load package...** option. However, if the required package is not included in the library of your computer, you have to install it first. We will practice it by installing the sciplot package, which we will use for creating some plots. If you are connected to the Internet, click on the **Install package(s)...** option and select a server from the displayed list; from this, you will download the package. When you do that, your computer connects to the given server and displays a list of all currently available packages. Select the one called sciplot (its installation on your computer is then executed automatically). If you are not connected to the Internet, you have to first download

the zip form of the package from the above address separately and install the package from the zip file manually afterwards. When doing so, you need to be aware of the version of R that you are using since packages can be specific for different versions. Install the zip file into your library using the **Install package(s) from local zip files...** option. Do not forget that in order to activate functions of an already installed package, you need to always load it for each session.

The **Windows** menu includes options for switching between the command and graphical windows. The most useful is probably the **Tile** option, which allows you to see two windows next to each other.

Finally, the **Help** menu offers many useful options. The **Console** option will explain to you how to control the console window. Commands are entered into the console following prompt (>). The prompt is normally red, while program answers are (typically) displayed in blue. Commands can be typed by the user line by line (hitting "Enter" after each complete command), or on the same line, separated by semicolons. No additional spaces are necessary between commands that are separated by a semicolon. Previous commands can be recalled one after another by pressing the up arrow key and back by pressing the down arrow key. Using the arrow keys makes the work much easier. For example, if you make a mistake entering a command, you can recall it by pressing the up arrow key and correct it. Using the left and right arrow keys, you can move within the currently edited line.

You can learn many useful facts about R when using the **FAQ on R** or **Manuals in (PDF)** options. The **R functions (text)...** option is very useful; it displays a detailed description of the function, name of which you enter. Here you can find a description of what the given function can be used for, a list of all its arguments, their legal values, references to literature (related to the method, on which the function is based), and a few examples. This command works only for functions from the package(s) that are currently uploaded. Descriptions of all functions (including those that are included in the installed packages but not uploaded) can be found by using the **Html help** option. If you select this option, pages with some basic information about R are displayed, including the functions included in the pre-installed packages. If you do not know the name of a given function but you know what it is supposed to do, you can try to find it using the **Search help** option. For example, if we want to find a function that computes the Shapiro-Wilk test, type the key word „shapiro". The program then searches all installed packages and displays the functions that include the searched word in their names or descriptions. The name of the package, in which the given function can be found, is displayed in the brackets behind the name of the function. In our case, the function is called **`shapiro.test`** and it is in the stats package.

2.3 Basic operations

The possibilities of R are really wide-ranging – the program includes hundreds and hundreds of different commands. Of course we cannot mention and explain all of them here. We will focus only on the most basic ones and on those you will encounter in this book most often. Some others will be introduced within the context of examples later in the book. A

brief list of the most important commands called the "R Reference Card" can be found at the following address: http://cran.r-project.org/doc/contrib/Short-refcard.pdf. It is basically a four-page sheet. You can also read more about R, in the book by one of the R Core Team experts, Dalgaard (2008).

Let us try some operations, starting with simple manipulations and proceeding to more complicated procedures. Firstly, we use the environment as a scientific calculator. Just type on the command line `2+5`, press ENTER and the software produces a result on a new line (which begins with `[1]`). Basic mathematical operators are: addition (**+**), subtraction (**-**), multiplication (*****), division (**/**), and power (**^**). Logic operators have the following form: less than (**<**), greater than (**>**), equal (**==**), not equal (**!=**), less than or equal (**<=**), greater than or equal (**>=**). They do not produce a number but a logical value: either **TRUE** (abbreviated **T**), or **FALSE** (abbreviated **F**).

Names of mathematical functions are in R very intuitive and thus easily memorable, for example, absolute value (**abs**), logarithm with base of *e* (**log**), logarithm with base of 2 (**log2**), logarithm with base of 10 (**log10**), exponent (**exp**), sine (**sin**), cosine (**cos**), tangent (**tan**), arc sine (**asin**), arc cosine (**acos**), arc tangent (**atan**), sum of several numbers (**sum**), product of several numbers (**prod**). The command for square root is **sqrt**. For other roots it is necessary to type them as powers (the power operator **^** is general and allows even negative and as well as non-integer arguments when they are mathematically correct). These simple functions are called by their name, followed by a number in parentheses, as you can see in the following examples:

```
> 3*2
[1] 6
> 3^4
[1] 81
> sqrt(9)
[1] 3
> 8^(1/3)
[1] 2
> 3==4
[1] FALSE
> log(10)
[1] 2.302585
> log10(10)
[1] 1
> exp(2)
[1] 7.389056
> prod(2,3,4)
[1] 24
```

Answers to these commands were only displayed on the screen. You cannot directly work with them any further. You can copy them and move them to the current command line using Copy and Paste. This can be inconvenient, especially if we want to conduct further operations with the result. In that case, you need to save the result into an object (and subsequently use its name in parentheses when using it as an argument of some other function). The simplest object is a scalar, i.e. a vector with a single element. Creating a scalar

is easy. Choose its name, for example `a`, type an arrow (composed of 'less than' and a dash, **`<-`**) or **`=`** behind it, and enter the value you want to enter in it, for example `8`. All of this is meant as an assignment of the value to the object `a`. You can easily display the content of a vector by simply typing its name on a new line or on the same line behind a semicolon. To create vectors of length larger than 1, you have to use the **`c`** (concatenate) function. This function will bind all entered values (given as arguments, within parentheses and separated by commas) into a vector in the order entered:

```
> a<-8; a
[1] 8
> b<-c(2,1/2,95); b
[1]  2.0  0.5 95.0
```

All names in R are case sensitive – including keywords (thus by typing `b` and `B` you are calling *different* objects). Object names can be almost arbitrary. Nevertheless, it is a good practice to remember some restrictions. Names must not begin with a number or a special symbol (such as comma or dot, etc.). Some names are better not to be used to avoid confusion. This applies to standard R commands and functions which have a predefined function, for example **`break`**, **`c`**, **`C`**, **`D`**, **`diff`**, **`else`**, **`F`**, **`FALSE`**, **`for`**, **`function`**, **`I`**, **`if`**, **`in`**, **`Inf`**, **`mean`**, **`NA`**, **`NaN`**, **`next`**, **`NULL`**, **`pi`**, **`q`**, **`range`**, **`rank`**, **`repeat`**, **`s`**, **`sd`**, **`t`**, **`tree`**, **`TRUE`**, **`T`**, **`var`** and **`while`**. If you use them then the predefined function will be masked (replaced with the new object). This can cause a number of nasty problems that are difficult to discover. You had best avoid using names that collide with the names of R functions (unless your intention is to replace the standard R function by a version of your own).

Values of vector components (and hence of vectors themselves) can be of different types: numeric, e.g. `c(1.5,20,-3.1)`, logical, e.g. `c(TRUE,FALSE,TRUE)`, or character. Characters are always enclosed in quotation marks, e.g. `c("blue","red","green")`. A vector should always include values of the same type. If there are various types, e.g. numbers and characters, the vector will be automatically classified by R as the most general appropriate type (i.e. character). Character vectors cannot be used for mathematical operations. This is useful to remember when searching for errors and reasons why something does not work.

Numeric vectors can be created by entering every number or using a colon, which requests creation of an integer sequence of values from:to.

```
> y<-1:11; y
 [1]  1  2  3  4  5  6  7  8  9 10 11
```

You can also create the same sequence in a different way – using the **`seq`** command. This command is not as simple as the previous command since it can do more – it can create a sequence that does not consist of integers only. To make it work, you need to specify the **arguments** that you need for the given objective. Particularly, you need to specify the initial value (**`from`**), final value (**`to`**) and the step size (**`by`**). It is typical for functions in R to include multiple arguments that have their own name and pre-defined positions (there are also functions with arguments that do not have names. It lets you to specify the arguments either using their names in any order or without names but in the appropriate order. The

latter option is more economical from the writing perspective and that is why we prefer to use it here. Nevertheless, this option has a significant implicit danger. The order of arguments is not quite codified and it can thus theoretically change with the development process of the R language. That is why, if you are using a different version of R than this one, it is absolutely necessary that you inspect the order of individual arguments for all of the used functions. Alternatively, you need to learn how to type commands with the names of arguments.

Now let us go back to the **seq** command and let us create a sequence from 1 to 2 with a step of 0.1:

```
> x<-seq(from=1,to=2,by=0.1); x
 [1] 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
```

Try to play with the arguments of this command – skip their names and change their order – in order to see how dramatically the result can change.

You can select elements from such a vector. The easiest way is to enter the element positions in square brackets. One number can be typed directly as an argument. Multiple numbers have to be connected together, either as a sequence using a colon, or as a vector using **c**. This means that if we want to select the third element, followed by the third through fifth, and finally sixth, eighth, and ninth elements from vector `x`, we will do so in the following manner:

```
> x[3]
[1] 1.2
> x1<-x[3:5]; x1
[1] 1.2 1.3 1.4
> x2<-x[c(6,8,9)]; x2
[1] 1.5 1.7 1.8
```

We can make our selection not only based on a given position but also based on a different (more complex) condition. If you want to find out which elements within a vector are greater than a certain value, use the **which** command, which will return the sequence of numbers of the elements that meet the given condition:

```
> which(x>1.5)
[1]  7  8  9 10 11
```

Vectors can be used in many more operations. We can join them to longer vectors (e.g. join `x1` and `x2` to create `x12`), or alter them using mathematical operators and functions.

```
> x12<-c(x1,x2); x12
[1] 1.2 1.3 1.4 1.5 1.7 1.8
```

Many mathematical functions are "vectorised" in R, i.e. written in a way such that the function is automatically applied to all elements of the given vector (and the result is returned as a vector). We can demonstrate it on, for example, calculating the third power (for the 11 numbers saved in `y`):

```
> y1<-y^3; y1
[1]    1    8   27   64  125  216  343  512  729 1000 1331
```

We often want to find out a length of a certain vector (how many elements it has). For this purpose, you can use the intuitively named **`length`** function. To demonstrate this function, we can determine the length of `y` in the following manner:

```
> length(y)
[1] 11
```

Notice that this function, as well as the previous ones, has processed vector argument `y`. Not only can variables, vectors and scalars be used as an argument, but also a call to another function that will prepare such an argument (inside the originally called function). For example, instead of typing the following two commands

```
> a1<-y^2; prod(a1)
[1] 1.593351e+15
```

you can just type a single, more complex command:

```
> prod(y^2)
[1] 1.593351e+15
```

The possibility of such (even repeated) nesting represents one of the advantages of the R environment – it allows for fast typing of instructions even for quite complex computations. The disadvantage of complicated nesting is that they are not so easily readable (especially for a beginner), so we will use it relatively sparsely.

Sometime we may need to standardise a numeric vector. This means to subtract the mean from the original values (i.e. **centring**) and then divide each value by the standard deviation (i.e. **scaling**). The mean of the standardised vector is equal to zero and the variance is 1. **Standardisation** can be done by calling **`scale`**. The first argument (**`x`**) specifies a vector that we want to standardise. Another two arguments (**`center`** and **`scale`**) specify whether we do want (**`TRUE`**) or do not want (**`FALSE`**) to apply centring and scaling. The default option (i.e. when arguments **`center`** and **`scale`** are not used) includes both centring and scaling.

The mean and variance of the vector `y` is obtained by calling **`mean`** and **`var`**.

```
> mean(y)
[1] 6
> var(y)
[1] 11
> y2<-scale(y); y2
            [,1]
 [1,] -1.5075567
 [2,] -1.2060454
 [3,] -0.9045340
 [4,] -0.6030227
```

```
 [5,] -0.3015113
 [6,]  0.0000000
 [7,]  0.3015113
 [8,]  0.6030227
 [9,]  0.9045340
[10,]  1.2060454
[11,]  1.5075567
attr(,"scaled:center")
[1] 6
attr(,"scaled:scale")
[1] 3.316625
> mean(y2)
[1] 0
> var(y2)
     [,1]
[1,]    1
```

Before standardisation, the mean was *6* and the variance was *11*, after standardisation, the mean is *0* and the variance is *1*. This is because from all values in `y,` a mean value of *6* was subtracted and then each value was divided by the standard deviation, *3.317*.

Two (or more) vectors of the same type (for example, numeric) can be combined in a matrix using two commands. If we want to compose the vectors "vertically" (column-wise), we use **`cbind`**. If we want to compose the vectors "horizontally" (row-wise), we use **`rbind`**. The arguments of these functions are the names of the combined vectors (functions that these vectors create). We need to enter as many arguments as there are vectors we want to combine. You also have to observe the correct dimension – **`cbind`** as well as **`rbind`** can correctly only combine vectors of identical lengths. If you combine vectors of various lengths, then all of them will be automatically (by default) changed to the length of the longest vector, which can be somewhat dangerous and lead to errors that are hard to locate!

```
> a12<-cbind(x1,x2); a12
x1  x2
[1,] 1.2 1.5
[2,] 1.3 1.7
[3,] 1.4 1.8
> a13<-rbind(x1,x2); a13
[,1] [,2] [,3]
x1  1.2  1.3  1.4
x2  1.5  1.7  1.8
```

The results are matrices that also include names of the vectors as their name attributes. Generally, matrices can be assembled in a more efficient way using the **`matrix`** command. Its first argument (**`data`**) is a vector of numbers, its second argument (**`nrow`**) is the number of rows and its third argument is the number of columns (**`ncol`**). Obviously, all of this assumes that the info given to the **`matrix`** function is consistent (i.e. that the length of the inputted vector is really equal to the **`nrow*ncol`**). Thus, for example we have:

```
> b1<-matrix(c(3,5,1,9,7,2),nrow=2); b1
```

```
     [,1] [,2] [,3]
[1,]    3    1    7
[2,]    5    9    2
```

Notice that the data are loaded into the matrix column by column (this can be changed to row by row through the **byrow=T** argument).

When we want to join vectors of different types, e.g. numeric and character types, we need to call **data.frame**. Should we use **cbind**, all vectors would be automatically transformed into the most general of all column types (e.g. the character type) and that is typically not what one would like to do (it could, for instance, loose numeric values). The command **data.frame** is used to create data structures more general than matrix. In the **data.frame**, different columns can have different types (but different rows of the same columns still have to have the same type). We will use the name and structure of the **data.frame** for data arrays throughout the text.

Let us create a data frame by binding two numeric vectors `x` and `y` with a vector `size`, which will be of the character type. We have to create this vector. As the character values in this vector will be repeated, we will call the function **rep**. It has two arguments: a repeated value (**x**) – it can be a vector, and the number of replications (**times**). In the vector `size`, we want a word "small" to be repeated four times, followed by a word "medium" repeated three times, and a word "large" repeated four times. The words must be enclosed in quotation marks.

```
> size<-rep(c("small","medium","large"),c(4,3,4)); size
 [1] "small"  "small"  "small"  "small"  "medium" "medium" "medium"
 [8] "large"  "large"  "large"  "large"
> dat<-data.frame(x,y,size); dat
     x  y   size
1  1.0  1  small
2  1.1  2  small
3  1.2  3  small
4  1.3  4  small
5  1.4  5 medium
6  1.5  6 medium
7  1.6  7 medium
8  1.7  8  large
9  1.8  9  large
10 1.9 10  large
11 2.0 11  large
```

This data frame has three columns and 11 rows, as it is clear from the output (the first column of the output shows the numbers of rows that are generated automatically and are not counted as a column of the data frame). The column called `size` contains character strings, hence it is a vector of the character type. However, the variable `size` is not yet ready to be used in ANOVA and similar models where character vectors enter as factors. We can check whether a given variable (entered as a single vector argument) is a factor or not using the function **is.factor**. To create a factor out of a character variable, we use the function **factor**.

```
> is.factor(size)
[1] FALSE
> size<-factor(size)
```

Levels within the factor are (implicitly) arranged in alphabetical order (in ascending order according to the ASCII codes and lengths), as we can find out using the **levels** command. If we want the first level of the `size` factor to be different (for example, "medium"), we need to use the **relevel** command and to define the reference level using the **ref** argument.

```
> levels(size)
[1] "large"  "medium" "small"
> size1<-relevel(size,ref="medium"); levels(size1)
[1] "medium" "large"  "small"
```

Any text from the command window can be easily copied via the clipboard (by pressing the CTRL and C keys after highlighting the text of interest) to other applications. As a default, R uses a font that is of the same width (Courier New). Upon copying the texts to, for example MS Word, it is therefore convenient to set the font to a similar type. Larger objects, for examples, extensive matrices, can be saved in a file of a specified format using the **write.table** command. Its first argument (**x**) is the name of the object to be written into a file, and it is followed by the path to the location where the file (**file**) is saved, and further, a particular separator (**sep**) character can be specified. For example, **\t** specifies that the columns will be separated from one another by the tab key. Upon launching the command

```
> write.table(dat,file="c:\\data.txt",sep="\t")
```

a data.txt file, containing the created data frame, appears on the hard drive (C:) of the computer (in the root directory).

All objects that we have created or uploaded during a single session are stored in the memory. We can recall their list using the **objects()** command. When we want to get rid of these objects, we can remove them all once the analysis is completed using the **Remove all objects** command. That is what you should do right now. But be careful because you will lose all unsaved objects! Another way of doing the same thing is:

```
> rm(list=ls(all=TRUE))
```

2.4 Data frames

The overwhelming majority of the statistical methods in R require data to be arranged in a column-wise format. This means that values (of the response as well as explanatory variables) are saved in a table-like structure (Table 2-1). The resulting data frame has as many columns as there are variables, and as many rows as there are measurements plus one row for the names of variables. Do not use spaces in the names of the data frame columns (or individual variables). It is advisable to substitute the spaces by dots (for example, plant.species). Moreover, it is (very) important that all columns have the same number of

Table 2-1 An example of a data frame with two categorical variables (*SOIL* and *FIELD*) and two numerical variables (*distance* and *amount*). The categorical variable *SOIL* has two levels: "moist" and "dry". The variable *FIELD* has two levels too, namely "pasture" and "rape". There are two missing values in the variable *distance*. These are denoted by NA.

SOIL	FIELD	Distance	amount
Moist	pasture	12	0.22
Moist	pasture	22	0.11
Moist	pasture	43	0.29
Moist	pasture	23	0.33
Moist	rape	32	0.19
Moist	rape	67	0.39
Moist	rape	54	0.18
Moist	rape	NA	0.29
Dry	pasture	11	1.16
Dry	pasture	33	1.03
Dry	pasture	45	1.11
Dry	pasture	NA	1.33
Dry	rape	55	1.02
Dry	rape	41	1.23
Dry	rape	14	1.05
Dry	rape	27	1.12

rows. There must not even be an empty position here, not even when there is a missing measurement. If a missing value occurs, type NA (which stands for Not Available) to the corresponding cell, instead.

Data can be entered into the R environment in several different ways. If you have just a few numbers, the most efficient way to do it is to enter the values using the concatenation command **`c`**, as we have already done above. If your data set is more extensive and you have already saved it, for example in MS Excel, you can get the data to R in two ways: by export-import, or using the clipboard.

Export-import requires the following steps: place the data on a separate MS Excel sheet, after which you can export them to the TXT format using Excel function **Save as**, Text option (separated by tab keys). Be careful! When doing so, the decimal place separator has to be set to the dot. If you have comma instead (which is common in many non-English local MS Windows settings), you have to change it to the dot in the MS Windows control panel (Local and language preferences option). The reason is that R would interpret commas differently than you would expect – for example, as a separator of columns and not as a decimal point. Subsequently, you can import the data to R by calling **`read.delim`**. The path to the given file is specified by argument (**`file`**), entered in quotation marks. When specifying the path, folders are separated using a double backslash (**`\\`**). The complete path has to be defined only if the file to be uploaded is in a different folder than the working folder. For example,

import of a file called metal.txt from the folder MABD located on a hard drive (marked by the letter c) would look like this:

```
> data<-read.delim("c:\\MABD\\metal.txt"),
```

where `data` is the arbitrary name of the data frame into which we place the uploaded data.

Now we will practice import via the clipboard. In MS Excel, open the pollution.xls file from folder MABD. Highlight the data, including the names of variables, and save it in the clipboard by pressing the CTRL and C keys together. Open the R environment. Name the data frame, into which you will insert the content of the clipboard, for example, `dat`. For this purpose, use the **`read.delim`** command with the **`clipboard`** argument in quotation marks:

```
> dat<-read.delim("clipboard")
```

Data are uploaded by pressing ENTER. Upon successful importation, it is advisable that you make the data frame visible using the **`attach`** command. The main argument (**`what`**) is the name of the data frame. Making the data frame visible will allow you to work directly with the names of variables. You can display them using the **`names`** command with the name of the appropriate data frame as the argument:

```
> attach(dat)
> names(dat)
[1] "soil"     "field"    "distance" "amount"
```

The object `dat` includes four variables: *SOIL*, *FIELD*, *distance*, and *amount*. Two of them, *SOIL* and *FIELD*, were automatically imported as factors, because both are of the character type (this can be overridden by using additional **`stringsAsFactors=F`** argument in the call to the **`read.delim`**). The variables *distance* and *amount* were imported as numeric vectors because both include only numbers. The variable *distance* also includes `NA` values (as you can easily discover by calling **`is.na`** with the name of an object as an argument). This may cause problems later, because some functions, e.g. **`sum`**, do not work properly when such values are present. There are several ways that these missing values can be used. For example, **`sum`** will work only when we specify what to do with `NA` values. The most simple (but not always best) option is to remove them and process the remaining (complete) data. This can be done by supplying argument **`na.rm=T`** in a function like **`sum`**, **`mean`**, etc.

```
> is.na(x=distance)
[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
[12] TRUE FALSE FALSE FALSE FALSE
> sum(distance)
[1] NA
> sum(distance,na.rm=T)
[1] 479
```

Additionally, you may also want to edit the data after the import procedure is completed. You can do that directly in R using the built in editor. Open the data in the editor using the **fix** command with the data frame name, `dat`, as an argument. Save the modified data in a new object, such as `dat1`. In the editor, substitute both `NA` values with, for example, 100 and close the editor. You can make the new data frame visible using the **attach** command, however, do not do it now. If we want to look at some variables of a particular data frame, you can use a name composed of the name of the data frame and the name of the variable, separated from each other by **$**. Verify that no `NA` values have been left in the *distance* variable:

```
> dat1<-fix(dat)
> dat1
    soil   field distance amount
1  moist pasture       12   0.22
2  moist pasture       22   0.11
3  moist pasture       43   0.29
4  moist pasture       23   0.33
5  moist    rape       32   0.19
6  moist    rape       67   0.39
7  moist    rape       54   0.38
8  moist    rape      100   0.29
9    dry pasture       11   1.36
10   dry pasture       33   1.03
11   dry pasture       45   1.61
12   dry pasture      100   1.33
13   dry    rape       55   1.02
14   dry    rape       41   1.23
15   dry    rape       14   1.25
16   dry    rape       27   1.12
> is.na(dat1$distance)
[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[13] FALSE FALSE FALSE FALSE
```

As we can see, there are no longer any present. Finally, remove all of the objects again:

```
> rm(list=ls(all=TRUE))
```

3 EXPLORATORY DATA ANALYSIS (EDA)

Compilation of statistical characteristics in a table or a graph is one of the important instruments used for data checking to obtain a basic idea about the behaviour of the studied variables – to check for gross errors, etc. The use of such techniques, including simple descriptive statistics and their summarised display in tables or plots, forms an initial part of many analyses.

Before we start testing statistical hypotheses or to model a relationship, it is reasonable "to check the situation". Our goal is to:

- Locate obvious mistakes (for example, typing errors)
- Get an idea about the result of the analysis
- Assess suitability of various statistical models
- Inspect whether the data comply with the assumptions of the selected model and methods of the analysis
- Observe new (unexpected) trends or other surprising facts

For these purposes, we will utilise the tabular and graphic capabilities of R. We will learn how to easily and quickly calculate the most important characteristics, such as estimates of expected values and variance.

3.1 Expected value

The expected value (E(y), μ) of a probability distribution with a density $p(y)$ for some "legal" set of values y (e.g. real numbers) is *defined* as

$$\mu = \int_{-\infty}^{\infty} yp(y)dy \quad \text{for continuous and} \quad \mu = \sum_{-\infty}^{\infty} yp(y) \quad \text{for discrete distributions.} \qquad (3\text{-}1)$$

Please note that its existence cannot be assumed – there are various distributions for which no expected value (3-1) exists. Yet another feature is the fact that the expected value generally does not fully determine the distribution. It is only one of many attributes of a given distribution. Various distributions thus can have an identical expected value, while some or all of their other characteristics differ. What is important is the fact that it is a theoretical construction – expected values can be calculated precisely for fully specified distributions, as addressed in probability theory.

The situation in practical statistics is different, however. We do not know the real distribution from which the data are generated. We are just trying to estimate it (if only some of its aspects) based on the collected data. To determine and describe the distribution of the actually observed values is not really a problem. But that is usually not something that we are really interested in. The observed data, their distribution and their computed numerical characteristics (like mean, variance, etc.) are typically interesting only as a source of information for estimating the distribution which generated our random data. That is, the (observable) data are really interesting only for estimating the true underlying (unobservable) distribution. The same thing then applies to various characteristics, including the expected value. The most commonly used *estimate* of the (true but unobservable) expected value is the **arithmetic average** ($\bar{y}$, **`mean`** function). We have to always keep in mind that there is a difference between the theoretical expected value, which we do not know, and its estimate (more or less easily computable from the available data)! This distinction will be present (even though sometimes only implicitly) throughout the entire book (and in the field of statistics generally).

Why is the arithmetic average so popular? Because it is simple to compute and has many great theoretical properties. For example, it has a lot of useful asymptotic (i.e. approximate large sample) properties (related to the laws of large numbers and central limit theorems), it is also the best (unbiased) estimate of the expected value when we work with a normal distribution. When the data are generated from a different than normal distribution, the arithmetic average may not be the best option any more. a typical example would be a situation when normal data are "contaminated" with outliers. Arithmetic average is insufficiently robust in relation to outlier occurrence (even a small number of outliers can lead to a completely meaningless estimate of the expected value). There are many more robust estimates that significantly differ when it comes to their theoretical characteristics. a simple and very robust alternative is the **median** (**`median`**) – especially suitable for symmetric distributions witch have much "heavier tails" than the normal distribution. Note that the median exists even in situations when an expected value does not exist and when the arithmetic average *estimates* a non-existent property (for example, for the Cauchy distribution). The price for the substantial robustness of the median is a reduced accuracy (efficiency) of the estimate of the expected value compared to e.g. the arithmetic average (when no outliers are present). This is caused by the fact that the median uses only a smaller part of the information available in the data. Other estimates attempt to achieve a better estimate by a compromise, while preserving at least a decent robustness. a simple (but not quite ideal) option is, for example, the **trimmed mean**. It is an arithmetic average calculated from trimmed data. In this case, the original data are used for the calculation with the exception of the highest $100.\alpha\%$ and the lowest $100.\alpha\%$ values, where α must comply with the following rule: $0 \leq \alpha \leq 0.5$.

We will practice computation of the mentioned estimates using the example from the Chapter 2 – the file metal.txt. This is a small data set, which includes measurements of amount [g/kg] of heavy metals (*amount*) in soil samples from 16 sites. The sites were on one of two types of habitats (*FIELD*: pasture, rape) with one of two types of soil (*SOIL*: dry, moist). The variable *distance* includes distance [km] from the origin of pollution. Data will be uploaded by means of **`read.delim`** and placed to data frame with the name `dat`. Because the file

is saved in the directory MABD, it is sufficient to use name of the file in quotation marks. For the variable *amount*, which is of our primary interest, we will compute the arithmetic average, median and trimmed mean (without 10 % of the highest and 10 % of the smallest values). In both functions, **mean** and **median**, the first argument (**x**) specifies the variable that we want to use (*amount* in this case). Trimmed mean can be also computed by a call to **mean**, but now with the argument **trim**, which specifies the desired level of trimming (α).

```
> dat<-read.delim("metal.txt")
> attach(dat)
> names(dat)
[1] "soil"     "field"    "distance" "amount"
> mean(amount)
[1] 0.690625
> median(amount)
[1] 0.705
> mean(amount,trim=0.1)
[1] 0.6864286
```

3.2 Variance

There are several commonly used characteristics of variability in random data. For basic orientation in the data, **range**, which is the difference between the maximal and minimal observation, is often used. The **range** function returns (somewhat atypically) the minimal and maximal values from the data (from a given vector). Statistical characteristics of the sample range are often relatively complicated. That is one of the reasons why a different statistical characteristic is often used – namely the **variance** (Var (y),, or σ^2). For a continuous distribution with density $p(y)$, it is defined as

$$\sigma^2 = \int_{-\infty}^{\infty} (y-\mu)^2 p(y) dy \ . \tag{3-2}$$

If it exists (for this is even more stringent than the existence of the expected value), the estimate of this unknown characteristic from the given data can be, once again, obtained in different ways. The most commonly used estimate (which is unbiased under broad, but not all circumstances) from a sample of n (where $n \geq 2$) measurements is

$$s^2 = \frac{\sum_{i=1}^{n} (y_i - \bar{y})^2}{n-1} . \tag{3-3}$$

This estimate can be obtained by a call to **var**, with the name of the data vector as argument. **Standard deviation** (σ), which is (unlike the variance) expressed on the scale of measurements, can be estimated as a square root of variance, or simply by calling the **sd** function. Standard deviation describes variability of *one randomly chosen measurement* of the study variable. Standard deviation of the mean or (more commonly) **standard error of the mean** (SE) estimated from n measurements is quite different. Clearly, if we have used

more than one observation to compute the mean, it must be smaller than the standard deviation of a single observation (indeed, that is often the reason for using mean!). It can be computed using the following formula: $\sqrt{\sigma^2/n}$. Its estimate can be obtained by replacing unknown σ^2 with its estimate, as $\sqrt{s^2/n}$. For data with outliers, it might be better to use a more robust estimate of standard deviation (e.g. median absolute deviation (MAD) via the R function **mad**).

Let's compute some estimates using the same data frame as before. For the variable *amount*, we compute range, variance, standard deviation, and standard error of the mean (`sem`). In all commands, i.e. **range**, **var** and **sd**, the first argument is the name of the variable, i.e. *amount*.

```
> range(amount)
[1] 0.11 1.33
> var(amount)
[1] 0.2162996
> sd(amount)
[1] 0.4650802
> sem<-sd(amount)/sqrt(length(amount)); sem
[1] 0.1162700
```

3.3 Confidence intervals

Confidence intervals (CI) are often used to determine the "quality" of estimates of various characteristics. Their popularity is related to the fact that they very intuitively express the uncertainty of the estimates obtained from given data. Even though confidence intervals can be determined for many different characteristics, one of the most commonly used are confidence intervals for the expected value (for the "true mean"). For data obtained from a normal distribution, the construction of such interval utilises quantiles of appropriate t-distribution. As you may recall from an elementary statistics course, the **95% confidence interval** (CI_{95}) can be calculated in the following manner:

$$\begin{aligned} \text{lower bound} &= \bar{y} - t_{0.975,v} \times SEM, \\ \text{upper bound} &= \bar{y} + t_{0.975,v} \times SEM, \end{aligned} \tag{3-4}$$

where $t_{0.975,v}$ is the 97.5th percentile of a t-distribution with $v = n - 1$ degrees of freedom.

For the variable *amount,* the lower and the upper limit of the 95% confidence interval for the expected value is computed as follows: percentile of the t-distribution can be found by calling **qt** and specifying arguments of the percentile (**p**) and the degrees of freedom (**df**). In our case there are 15 degrees of freedom. Using the previously computed and saved SE we get:

```
> mean(amount)+sem*c(qt(p=0.0255,df=15),qt(p=0.975,df=15))
[1] 0.4428013 0.9384487
```

The resulting confidence interval is symmetric around the estimated expected value, i.e. around the arithmetic average (*0.691*). Because the 97.5th percentile of the t-distribution for non-negligible degrees of freedom (say above 20) is close to 2, a fast way to construct CI95 is to add and subtract two SE to the mean.

For data from other distributions, e.g. Poisson or binominal, the confidence intervals for the expected value are calculated differently (even if the mean is used as an estimate of the unknown expected value). There are several ways to do that. The simplest one is based on normal approximation, i.e. on the use of formulas (3-4) on suitably transformed values. Resulting values of the interval limits are then transformed back using the inverse of the transforming function. For data from the Poisson distribution, square root or logarithmic transformation is often used, while, for data from the binomial distribution, the logit function (7-1) is often utilised. Because of the nonlinearity of the transformation function, the intervals constructed in this manner will not be symmetric around the estimated expected value. This is in striking contrast to what most readers are used to from the normal distribution. In fact, the asymmetry is not a bug, it is a desirable feature! For example, when the distribution has not unlimited support (e.g. it is bounded by zero on the left) the symmetric intervals can easily violate the bound(s), while the asymmetric intervals (for suitably chosen transformation function) will not.

To make the calculation of the confidence intervals easier, we will often use a general function called **`confint`**. This function is able to calculate confidence intervals for the parameters estimated within the scope of various model classes (for example, GLM). The function has several arguments. The first one (**`object`**) determines the name of the object that contains results of a model previously fitted. The **`level`** argument specifies the confidence level (95% level is a default option).

3.4 Summary tables

Tables are used for summarising descriptive characteristics of variables of interest. There are several options available in R. By applying the **`summary`** function, we can obtain the minimum, maximum, 25% and 75% quantiles, median, arithmetic average and the number of missing values simultaneously for all numeric variables present in a given data frame. We are not usually interested in all of these values, however; therefore, if we want to know a particular characteristic of a selected variable, for example, the arithmetic average, for all levels of a selected categorical variable, we use the **`tapply`** command.

This is a very useful command, thus keep it in mind for later use. It has three basic arguments: the first (**`X`**) is the name of the variable for which the characteristic is computed, the second (**`INDEX`**) is the name of the categorical variable (of which levels define groups) for which a characteristic is computed), the third (**`FUN`**) is the function, which computes a requested characteristic. If we need to estimate a characteristic for each combination of levels of two or more variables, their names must be placed after the argument **`list`**, e.g. for the variables *SOIL* and *FIELD* it is: **`INDEX=list(soil,field)`**. We can use any predefined function

(such as **mean**, **var**, etc.) as the **FUN** argument. Alternatively, you can write your own function to be used as **FUN**. a more experienced user will know that the new function can be constructed using the **function(x){**commands**}** scheme. Beside the function **tapply**, one can use **table**, which computes frequency of particular values of a continuous variable or frequency of distinct levels of a categorical variable.

We will try the **table** function on the data frame `dat`, which we created previously, but first of all, the **summary** function:

```
> summary(dat)
    soil        field       distance          amount
 dry  :8   pasture:8   Min.   :11.00   Min.   :0.1100
 moist:8   rape   :8   1st Qu.:22.25   1st Qu.:0.2725
                       Median :32.50   Median :0.7050
                       Mean   :34.21   Mean   :0.6906
                       3rd Qu.:44.50   3rd Qu.:1.1125
                       Max.   :67.00   Max.   :1.3300
                       NA's   : 2.00
```

For factors, *SOIL* and *FIELD*, the output includes only frequencies of each level. For continuous variables, *distance* and *amount*, seven characteristics are presented in the output. Frequency of measurements for each level of factors can be found upon calling **table** with the name of the variable as an argument:

```
> table(field)
field
pasture    rape
      8       8
```

We can obtain a table of the arithmetic averages of the combinations of the levels of both factors and the standard error of the means (SE) only for the levels of the *SOIL* factor in the following manner:

```
> tapply(amount,list(soil,field),mean)
      pasture   rape
dry    1.1575 1.1050
moist  0.2375 0.2625
> tapply(amount,soil,function(x) sd(x)/(sqrt(length(x))))
       dry      moist
0.03776986 0.03223795
```

3.5 Plots

A graphical representation of the measured data is often more efficient and illustrative than just a table. This is especially true for large sets of data. R comes with an extensive graphical library, by the means of which we can create even very sophisticated plots suitable for publication. To get an idea, look at the demo examples. Just type **demo(graphics)** or **demo(image)**.

Table 3-1 Overview of the most important arguments of the function **plot**, description of their use and range of eligible values/codes. The two dots in the first column are to be replaced by values.

Argument	Description and eligible values
type=".."	Style of the plot to be drawn: `n` (empty plot), `p` (scatter plot), `l` (lines plot), `b` (points and lines plot), `h` (vertical lines)
las=	Style of axis labels: `0` (parallel to the axis), `1` (horizontal), `2` (perpendicular to the axis), `3` (vertical)
xlab="." **ylab=".."**	Label of abscissa or ordinate
cex.lab=	Font size to be used for axis labels, relative to the default setting: `1,.`
xlim=c(..., ...) **ylim=c**(..., ...)	Range of the values to be plotted along x or y axes. The first value is the minimum and the second value corresponds to the maximum.
cex.axis=	Font size of axis annotation to be used, relative to the default setting: `1,.`
log=".."	Logarithmic scale of: `x` (only abscissa), `y` (only ordinate), `xy` (both axes)
main=".."	Title of the plot
main.cex=	Font size to be used for title of the plot, relative to the default setting: `1,.`

R can use more than one type of graphics: we will look at simple plots from the base package and more sophisticated plots from the lattice package. We will deal mostly with the basic graphics, which is completely sufficient for creating many types of high quality plots.

The main graphic function is **plot**. It has a vast number of arguments, which can provide various details upon request by the user. Some of them are shown in Table 3-1. The long list of arguments may look quite horrifying, especially for those who are used to creating plots in MS Excel. Fortunately, a simpler plot can be created using a single command, for example, **plot(x,y)**, where x represents the name of the vector of values that will be drawn on the horizontal axis (abscissa) and y represents the vector of values that will be drawn on the vertical axis (ordinate). Of course, both vectors have to be of the same length. However, they can include missing values.

More complicated plots can be created literally piece by piece, because individual plot components in R can be overlaid on the top of each other, very much like transparent films. First, a background is created; after that, individual points, lines, legend, captions, etc. are added. Other important graphical functions are **points** (plots individual data points) and **lines** (draws lines, possibly connecting points). Their most important arguments are summarised in Tables 3-2 and 3-3. These functions can be called for the same plot repeatedly.

Table 3-2 Overview of the main arguments of the function **`points`**, description of their use and range of eligible values/codes.

Argument	Description and eligible values
`pch=`	Types of symbols to be used as points: `0,..,18` (Fig. 3-1A) or a character in quotation marks (e.g., „a“ or „*“)
`cex=`	Point size, relative to the default setting: `1,.`
`col=`	Colour of points: `1` (black), `2` (red), `3` (green), `4` (blue), `5` (cyan), `6` (purple), `7` (yellow), and `8` (grey)
`font=`	Font type for characters used as points: `1` (plain), `2` (bold), `3` (italic), and `4` (bold italic)

Table 3-3 Overview of main arguments of the function **`lines`**, description of their use and range of eligible values.

Argument	Description and eligible values
`x=c`(..., ...) **`y=c`**(..., ...)	Coordinate vectors of points to join, the first vector is for abscissa, the second is for ordinate
`lty=`	Line type: `1,..,6` (Fig. 3-1B)
`conf=`	Colour of line: `1` (black), `2` (red), `3` (green), `4` (blue), `5` (cyan), `6` (purple), `7` (yellow), and `8` (grey)
`lwd=`	Thickness of a line: `1,.`

A legend, describing types and colours of lines and symbols, can be placed into a plot by a call to the **`legend`** function. The first two arguments (**`x`**, **`y`**) of this function are coordinates on abscissa and ordinate which define the position of the upper left corner of the legend rectangle. Another argument (**`legend`**) is the vector of names of levels of a categorical variable. Remember that particular names must be enclosed in quotation marks (since you are, in fact, entering elements of a character vector). Arguments **`pch`** and **`lty`** require numerical codes representing symbols, or lines, used for particular groups (in the same order as in the vector of level names). Instead of specifying coordinates of axes; the legend can be placed on the plot by specifying **`locator()`**. The legend will be placed in the plot at the location where you click the mouse. The bordering rectangle of the legend box can be omitted using the argument **`lty="n"`**.

Fig. 3-1 Numeric values representing types of points (**A**) and lines (**B**).

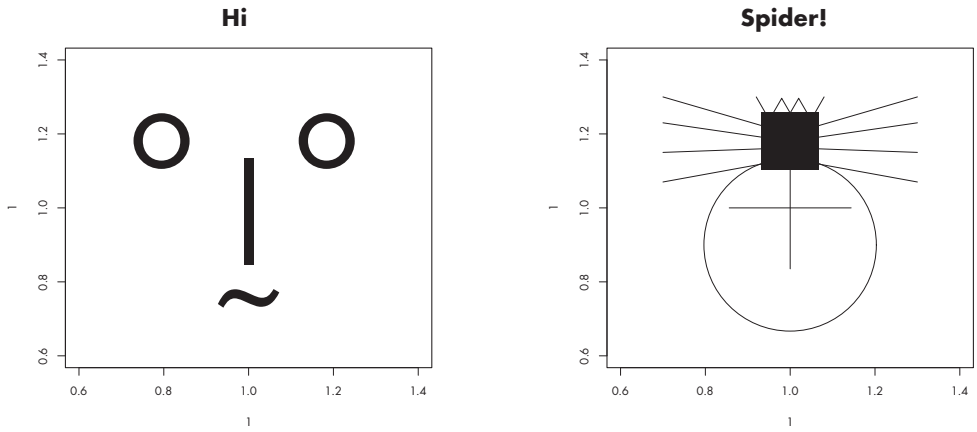


Fig. 3-2 An example of a picture made by means of plot functions.

The list of arguments of the function **`plot`** is far from being complete. There are really many, many more. The list and their description can be obtained by calling **`?par`**. For a detailed description of graphical features of R see Cleveland (1993) or Murrell (2005).

The size of the graphical window can be altered by dragging its corner using the mouse, or redefining its size by calling **`x11`**. Its first arguments (**`width`**, **`height`**) are width and height of the window in inches. In this book, all figures were made with **`x11(7,6)`** for one or **`x11(6,6)`** for two plots side by side. The entire graphical window can be split into several smaller ones, each containing one plot. This is done by calling **`par(mfrow)`**, where we specify a vector of two values. The first defines the number of rows and the second the number of columns in the graphical window. Thus, for example, **`par(mfrow=c(1,2))`**, splits the window in two smaller areas side by side. The graphical window is opened automatically after calling a graphical command. The figure can include basically anything, as you can see in the following "plots" (Fig. 3-2).

Plots can be copied to virtually any application, or they can be saved in three different formats (metafile, bitmap or a postscript file). To copy your plots into MS Windows applications, it is better to use the metafile format (WMF) than bitmap since it can be edited later within the target application, if needed. We will introduce some common plot types in the following paragraphs.

[1] Commands used to make the picture:

```
> plot(1,1,pch="|",cex=10,main="Hi"); points(1,0.7,pch="~",cex=10)
> points(0.8,1.2,pch="o",cex=10); points(1.2,1.2,pch="o",cex=10)
> plot(1,1,type="n",main="Spider!"); points(1,1.18,pch=15,cex=10); points(1,0.9,pch=1,cex=30)
> points(1,1,pch=3,cex=15); points(0.98,1.26,pch=2,cex=3); points(1.02,1.26,pch=2,cex=3)
> lines(c(1.06,1.3),c(1.22,1.3)); lines(c(1.06,1.3),c(1.19,1.23))
> lines(c(1.06,1.3),c(1.16,1.15)); lines(c(1.06,1.3),c(1.12,1.07))
> lines(c(0.94,0.7),c(1.22,1.3)); lines(c(0.94,0.7),c(1.19,1.23))
> lines(c(0.94,0.7),c(1.16,1.15)); lines(c(0.94,0.7),c(1.12,1.07))
> lines(c(1.05,1.08),c(1.24,1.3)); lines(c(0.95,0.92),c(1.24,1.3))
```

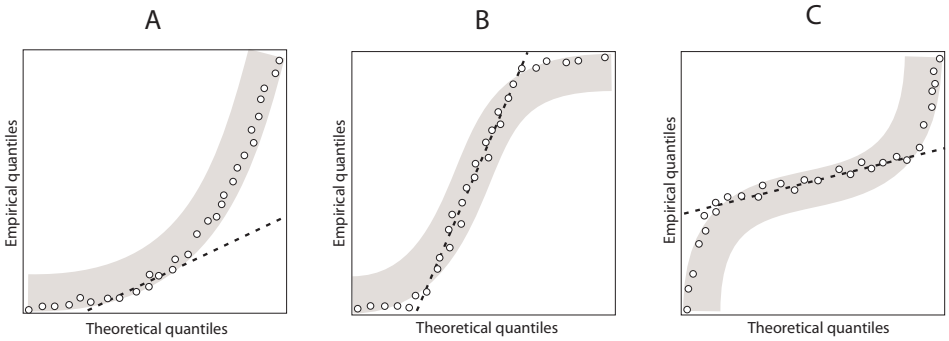


Fig. 3-3 Q-Q normal plots showing examples of few different non-normal distributions. **A**. Asymmetric distribution, skewed to the right. **B**. Symmetric distribution with weak tails. **C**. Symmetric distribution with heavy tails. Grey area highlights the trend.

3.5.1 DISTRIBUTION PLOTS

Distribution of a variable can be depicted in R in several ways: e.g. as a histogram, stem-and-leaf plot, and Q-Q plot. To get a histogram we call **`hist`**. Although histograms are frequently used, more information about the distribution can be read from a stem-and-leaf plot or a Q-Q plot. a great advantage of the stem-and-leaf plot (**`stem`**) is the fact that it displays particular values of measurements not only classes, as in a histogram. This can help to check the data, identify outliers, etc. Q-Q plot (**`qqplot`**) is used to compare two distributions, such as empirical distribution of measurements with an expected theoretical distribution (or with another empirical distribution). In general, it compares quantiles of one distribution to the quantiles of another distribution. If both distributions are identical then all points should lie on a line with zero intercept and identity slope (a diagonal). a special and very important case of the Q-Q plot is a normal Q-Q plot. This is used to compare distribution of measurements against normal distribution (as a special case of expected theoretical distribution). It compares quantiles of normal distribution (fitted to the data) on the abscissa with empirical quantiles (obtained directly from data without any assumptions) on the ordinate. This plot is created upon typing **`qqnorm`**. Ideally (when the data are exactly normally distributed) all points lie on a line, which can be drawn by calling **`qqline`**. Obviously, since the data are random, we cannot expect an exact match with the theoretical behaviour, but they should be close. Evaluation of whether the data are approximately normal is difficult. Even judgement based on these plots is somewhat subjective. Good judgements certainly require extensive experience. The most frequent departures appear as arrangements of points in the shapes of "J", "S", or inverted "S" (Fig. 3-3).

Let us have a look at the distribution of the explanatory variable *distance* (typically we would study distribution of the response variable *amount*, which is, however, not so illustrative in this case). We make a histogram and, in the same graphical window, split by calling **`par`**; we plot the normal Q-Q plot next to it. All three functions, **`hist`**, **`qqnorm`** and **`qqline`**, use the name of the variable as the first argument.

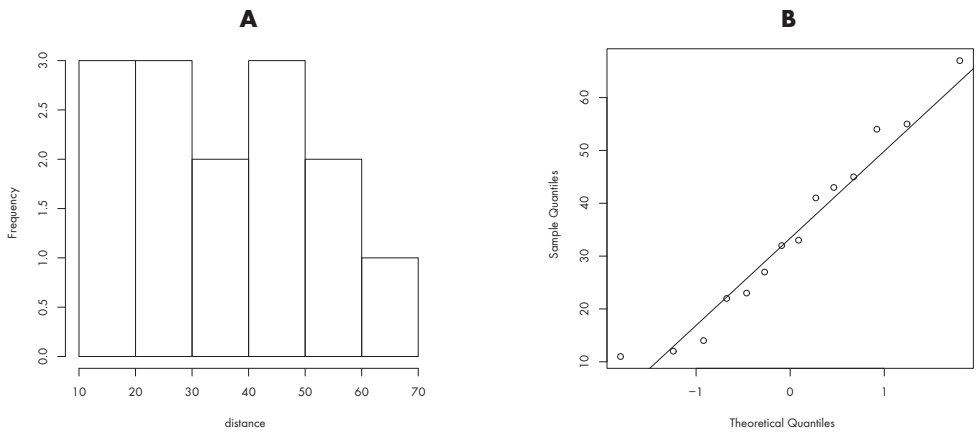


Fig. 3-4 Histogram (**A**) and Q-Q plot (**B**) of the variable *distance*.

```
> par(mfrow=c(1,2))
> hist(distance)
> qqnorm(distance)
> qqline(distance)
```

Do you think the distribution of *distance* is "normal"? Histogram (Fig. 3-4A) suggests that it is not. On the other hand, the Q-Q plot (Fig. 3-4B) looks almost normal. We repeat here that to assess deviances from normality (especially when the sample size is relatively small) is very difficult, if not impossible. We can easily reach both false negative (not detecting departure from normality when the true generating distribution is, in fact, non-normal) and false positive (detecting non-normality when the true generating distribution is normal) decisions. We will demonstrate the phenomenon on a simple simulation. Let us select samples of various sizes from the population of measurements, about which we *know* that it has a normal distribution. Obviously, with the real data, we never know what the truth is, but, with simulations, the truth is under our control. We will use the **`rnorm`** command for this purpose. This function generates (pseudo)random numbers from a specified normal distribution (with user-specified parameters mean and standard deviation). The function has three arguments: sample size (**`n`**), mean value (**`mean`**) and standard deviation (**`sd`**). Let us select 10, 20, 50 and 100 values from the normal distribution with mean value of 0 and standard deviation of 1. We store the resulting random numbers in vectors `y1` through `y4`. Subsequently, we plot them in Q-Q plots with the addition of a diagonal line:

```
> par(mfrow=c(2,2))
> y1<-rnorm(n=10,mean=0,sd=1)
> y2<-rnorm(20,0,1)
> y3<-rnorm(50,0,1)
> y4<-rnorm(100,0,1)
> qqnorm(y1); qqline(y1)
> qqnorm(y2); qqline(y2)
> qqnorm(y3); qqline(y3)
> qqnorm(y4); qqline(y4)
```

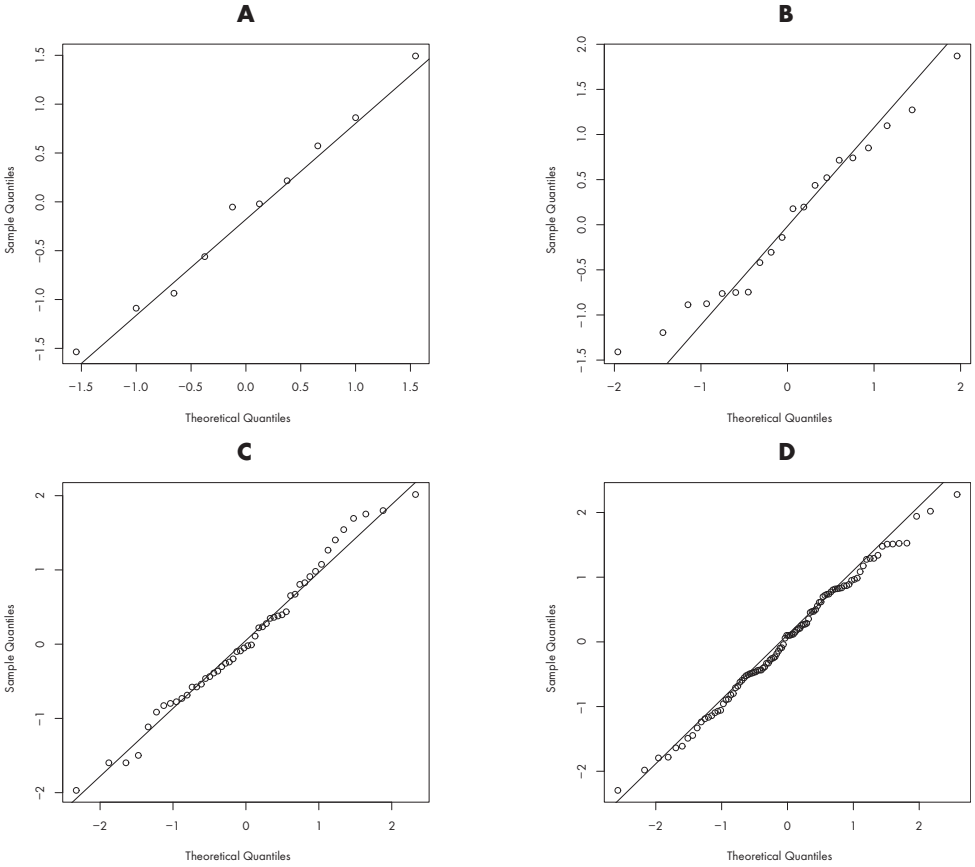


Fig. 3-5 Q-Q normal plots of four random samples from normal distribution. **A**. 10 values. **B**. 20 values. **C**. 50 values. **D**. 100 values.

Each plot (Fig. 3-5) looks completely different. Points are not located exactly on a straight line in any of the plots despite the fact that we always selected them from a normal distribution! Apparent deviances from ideal normality can be quite strongly pronounced, especially for a small sample sizes (Fig. 3-5A–B). With an increasing sample size, the Q-Q plot has the tendency to "look as it is supposed to look", i.e. the points are located approximately on a straight diagonal line (Fig. 3-5C–D). Remember that this is just a tendency, not a guarantee! Moreover, as mentioned before, data generated from a different than normal distribution can look completely *normal* in a Q-Q plot.

It is a good practice to "clean" after using the **`mfrow`** parameter, i.e. to put the graphical window back to its standard mode (where it is in one piece, without any split into sub-windows) in order to prevent a later confusion:

```
> par(mfrow=c(1,1))
```

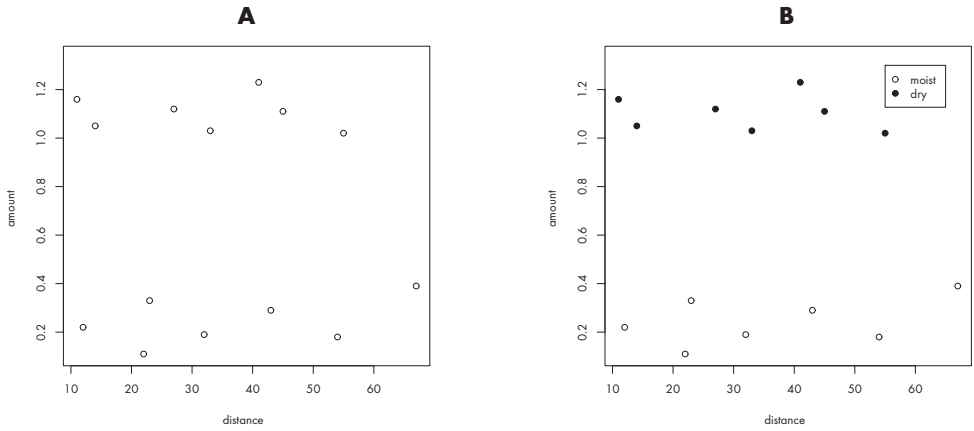


Fig. 3-6 Scatter plot of relationship between *amount* and *distance* for all data (**A**) and with different symbols for levels of *SOIL* (**B**).

3.5.2 SCATTER PLOTS

Scatter plot is used to plot a relationship between two numeric variables. a scatter plot is created by **`plot`**. Plot properties are further specified by its arguments. In this function, the first argument (**`x`**) is a variable, which goes to abscissa (i.e. an explanatory variable), and the second (**`y`**) is a variable that goes to ordinate (i.e. a response variable).

For practical purposes, we will create a plot of a relationship between two numerical variables, *distance* and *amount*. The first plot will be without differentiating the points for the levels of the *SOIL* and the second, more complicated plot, will distinguish between *SOIL* levels. For a better orientation, we will insert a legend into the second plot.

```
> plot(distance,amount)
> plot(distance,amount,type="n")
> points(distance[soil=="moist"],amount[soil=="moist"])
> points(distance[soil=="dry"],amount[soil=="dry"],pch=16)
> legend(55,1.3,c("moist","dry"),pch=c(1,16))
```

No increasing or decreasing trend is obvious from the plot without distinguishing between *SOIL* levels (Fig. 3-6A). The second plot (Fig. 3-6B) confirms the absence of a trend for both levels of the *SOIL* factor, but at the same time, it reveals a more or less constant difference between "moist" and "dry" for various values of *distance*.

3.5.3 BOX PLOTS

Box plots are used for practical and compact displays of multiple descriptive statistics. They can be used for graphical presentation of the characteristics of a single sample or for a fast comparison of the study quantity for multiple levels of the given categorical variable. The **`boxplot`** function called with two arguments (the first one being the name of the response

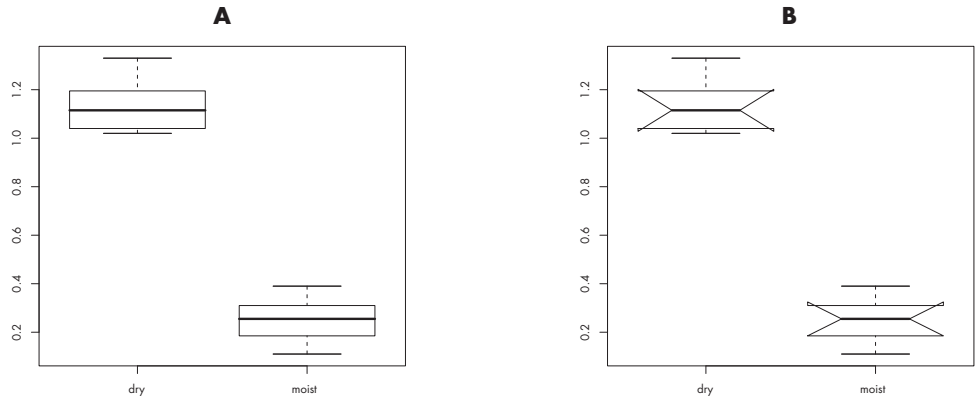


Fig. 3-7 Box plot of *amount* for two levels of *SOIL* without (**A**) and with notches (**B**).

variable and the second one the name of the categorical variable) will draw a "box" for each level of the categorical variable. What does this plot display? The thick line inside of the box is the median. The lower boundary of the box corresponds to the lower quartile and the upper boundary to the upper quartile. Therefore the box contains 50 % of all data (i.e. it depicts the interquartile range). If the median is (approximately) in the middle of the box, the distribution is (approximately) symmetric in its central half. The whiskers are extended from the box boundaries to a length that is 1.5-multiples of the interquartile range (height of the box), as long as there are data as far as this. If there are no data so far away from the box, the whisker is drawn only to the most distant datum. Data which are beyond the 1.5-multiple of the interquartile range limits are plotted as individual points (they can be easily suspected as outliers). This also helps to judge symmetry in the more extreme part of the distribution (within whisker and outlier parts). For this plot type, we can specify notches using the **`notch`** argument, which represents an approximate 95% confidence interval for the median. This means that if the ranges bounded by the notches in two compared boxes do not overlap (it is not sufficient if the median from one box is outside of the notches of the other box), the plot roughly suggests a statistically significant difference of the medians at the 5% level.

We will draw box plots of the *amount* without and with the notches for both levels of the *SOIL*.

```
> boxplot(soil,amount)
> boxplot(soil,amount,notch=T)
```

On the first plot (Fig. 3-7A), we can see several characteristics of the examined distribution. The distribution of *amount* for the level "dry" is slightly asymmetric (skewed to the right), while the distribution for the level "moist" appears to be more or less symmetric. The notches of the 95% confidence interval on the next plot (Fig. 3-7B) do not overlap each other and the difference between "dry" and "moist" is thus large and hence might be easily significant.

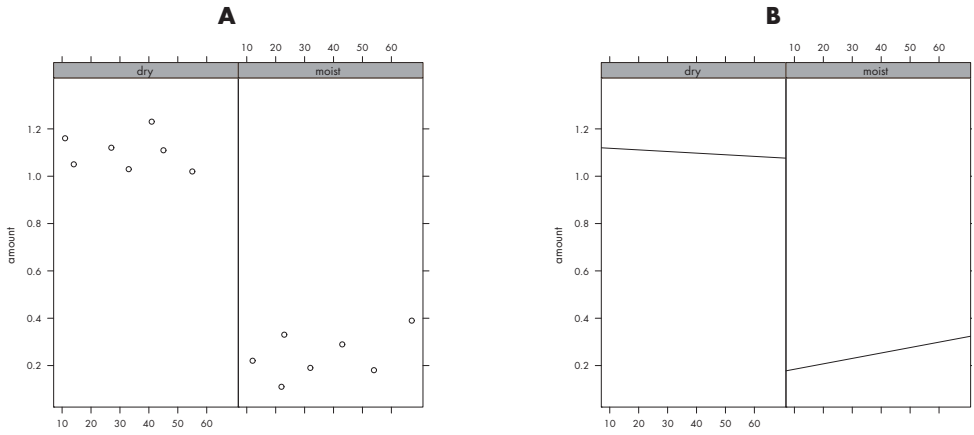


Fig. 3-8 A. Lattice scatter plot of the relationship between *amount* and *distance* for two levels of *SOIL*: "dry" and "moist". **B.** Lattice plot of the relationship with regression lines.

3.5.4 LATTICE PLOTS

These plots are an effective alternative to scatter plots in situations where we need to examine the relationship between a pair of variables, say an explanatory variable (*x*) and a response variable (*y*), but separately for different levels of a categorical variable (*A*). The function **`xyplot`** is part of the lattice package, which is obtained by a call to the **`library`**. The function will produce a plot with as many panels as there are levels of the categorical variable. The first argument of this function is a model formula in the form of `y~x|A`. That is it draws a relationship between *y* and *x* separately for each level of *A*. Another argument is **`type`**, which specifies the style of the plot (scatterplot of points, lines, both, etc.). The whole list of arguments is available from the help page for the **`xyplot`** function – which can be obtained by typing **`?xyplot`**.

We will compare the relationship between *amount* and *distance* for two levels of *SOIL*. First, we make a simple scatter plot and then a plot of regression lines fitting these data. Using the argument **`col`**, we specify the colour black for the points and lines (otherwise these would be cyan by default) and by means of **`type`** we specify plotting of regression lines.

```
> library(lattice)
> xyplot(amount~distance|soil,col=1)
> xyplot(amount~distance|soil,col=1,type="r")
```

In each panel, there are two plots (Fig. 3-8), because the factor *SOIL* has two levels. Names of the levels are displayed in the heading part of each panel. The plot in Fig. 3-8A shows the same situation as Fig. 3-6, but points are in separate panels.

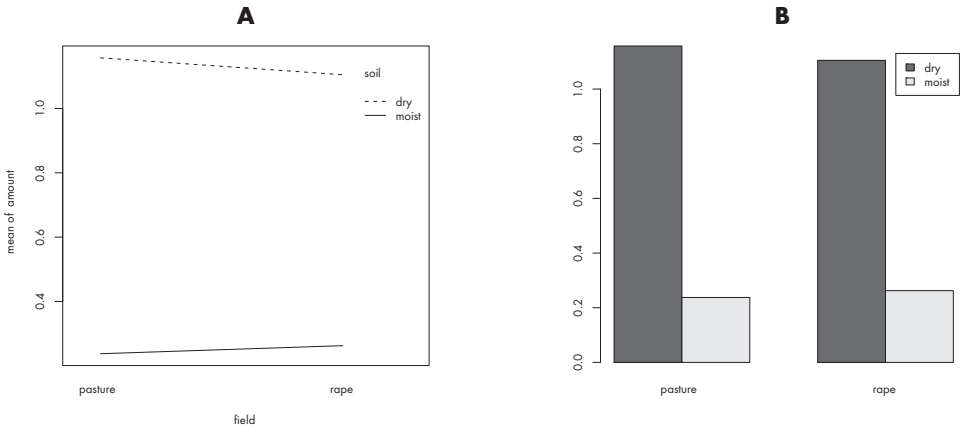


Fig. 3-9 A. Interaction plot between *FIELD* and *SOIL* for *amount*, each having two levels. **B**. Bar plot showing mean of *amount* for combinations of *FIELD* and *SOIL* levels.

3.5.5 INTERACTION PLOTS

When examining the influence of two categorical explanatory variables upon a third (response) variable, the **`interaction.plot`** function can often be useful. It will display the arithmetic averages of the response for all combinations of levels of the two explanatory variables. At the same time, it will connect the levels of the factor on abscissa with a separate line for each level of the explanatory variable displayed in the legend.

The name of this plot type is derived from the fact that this function is used for visual (informal) evaluation for the presence of an **interaction** between two factors. Nevertheless, a statistical test has to be used for a proper formal evaluation of whether the interaction is important or not. We assume that you have heard at least something about the interaction. Let us just briefly remind you that interaction (of two or more explanatory variables) refers to the situation where the influence of explanatory variable(s) upon the response is modified by another explanatory variable(s). If there is no interaction between two factors, say *A* and *B*, we call their influence (upon the response) **additive**. In an interaction plot, this is demonstrated by the connecting lines among the levels of one factor (*A*) being parallel when drawn separately for different levels of the factor (*B*). In other words, the differences between the responses for different levels of factor *B* computed at a given level of factor *A* are identical for all levels of factor *A*. The effect is symmetric for *A* and *B*, so that the absence of the interaction between *A* and *B* also means that the differences between the response for different levels of factor *A* computed at a given level of factor *B* are identical for all levels of factor *B*. On the other hand, we can say that the interaction means that differences in response among *A* levels are different for different *B* levels (or equivalently, that the differences among *B* levels are different for different *A* levels). Obviously, in real (random) data, most often, we will not see exact parallelism of the *A* profiles across *B* levels, but something "close". A drastic example of an interaction is

when the connecting lines cross each other. In that case, the relationship between the response and one factor changes qualitatively for different levels of the second factor. We will encounter interactions in several examples further in this book.

Interaction is often denoted by an asterisk. For example the interaction of two factors *A* and *B* is conventionally denoted as *A*B*, and of three factors as *A*B*C*. R uses a colon instead, i.e. *A*:*B*, and *A*:*B*:*C*. The asterisk stands for something different in R (see Chapter 6.4). Due to compatibility with the R environment we will use colon in the description of models containing interaction(s).

We will explore now our data frame in order to find whether there is an interaction between *FIELD* and *SOIL* in their influence upon the *amount* of heavy metals. The first argument (**`x.factor`**) of the function **`interaction.plot`** is the name of a factor that goes to abscissa. We choose *FIELD* here. For each of its levels, two mean values of *amount* are estimated, corresponding to the number of levels of the second factor (*SOIL*), specified by an argument (**`trace.factor`**). Levels of the second factor are displayed in the legend. Argument (**`response`**) identifies the name of the response variable (*amount*).

```
> interaction.plot(field,soil,amount)
```

The points (connected by lines) are mean values of response for combinations of levels of the two factors, *SOIL* and *FIELD* (Fig. 3-9A). It is apparent from the plot that *amount* of heavy metals is higher on the "dry" soil than on the "moist" soil. This indicates that the effect of *SOIL* is significant. On the contrary, effect of *FIELD* is much weaker (lines connecting the relevant points are rather flat, having slopes of small magnitude). The lines are not parallel here, and this indicates the presence of an interaction (though weak). With "pasture", the response difference between two levels of *SOIL* is approx. 0.9 units, whereas on "rape" it is approx. 0.8 units (to put in another way, difference between "dry" and "moist" soil is different on the "pasture" than on the "rape" field).

3.5.6 BAR PLOTS

Bar plot is used to display counts or relative frequencies. The function **`barplot`** typically works with data arranged in a table or a matrix. The **`barplot`** function has many arguments. The most important argument specifies name of the table (or the matrix) containing the values to plot (argument **`height`**). Our data frame does not include any count, thus we make a bar plot to display means of four combinations of levels of factors *SOIL* and *FIELD*. They can be computed by calling **`tapply`**. The result is then saved to the object `kt`. This object will then be used as one of the arguments for the subsequent call to **`barplot`**. As we have two factors (represented by two types of bars), we need to specify whether we want them to be drawn side by side (setting the argument **`beside=T`**). Further, we can add a legend – using the argument **`legend`**. Names of levels, to be drawn in the legend, can be extracted from the data frame by calling **`rownames`** (Fig. 3-9B).

```
> kt<-tapply(amount,list(soil,field),mean); kt
      pasture   rape
dry    1.1575 1.1050
moist  0.2375 0.2625
> barplot(kt,beside=T,legend=rownames(kt))
```

3.5.7 PAIRED PLOTS

A somewhat special graphical function is the **pairs** function, which we can use when we want to study a mutual relation among multiple (typically continuous) variables. The result is a "matrix" of plots for all pairs of the variables (which appear as panels arranged in a matrix-like structure). Generally, if we have *k* variables, we will have

$$\binom{k}{2} = \frac{k(k-1)}{2} \tag{3-5}$$

plots that we will be interested in since the other half of the plots is similar (only the axes are swapped). We will explain this graphical function in more detail later, using concrete examples (Chapters 8.4 and 10.4).

3.5.8 3D PLOTS

The use of three-dimensional plots is useful for a simultaneous display of a relationship between a single response and two continuous explanatory variables. The result can be, for example, a surface made of two explanatory variables. Its presentation can be useful for examining effects within a multiple regression. For this purpose, the lattice package has two functions: **wireframe** and **cloud**. We will explore one of them in more detail in Chapter 10.4.

3.5.9 PLOTS WITH WHISKERS

It is very popular to show arithmetic averages as empirical estimates of unknown expected values, together with standard errors of the mean (SE). Such plots can be used both as simple data summaries and as tools to explore the structure of the response. Once we fit a model, we might obtain better estimates from the analysis (together with appropriate standard errors). If we need to plot just the empirical estimates, we can use a function from the library sciplot, which we installed in Chapter 2.2. We can make either a bar plot (**bargraph.CI**) or a lines plot (**lineplot.CI**). Bars or points, depict arithmetic means, while the whiskers depict their SEs. Both functions have similar arguments. The first (**x.factor**) is the name of the explanatory categorical variable, with levels to be displayed on the abscissa; the second (**response**) is the name of the response variable. In situations where we need to plot response means for combinations of levels of two factors with the levels of the second one in the legend, we specify the name of the second factor by an argument **group**.

First of all, we will make a bar plot with whiskers for the levels of the *SOIL*, followed by a line plot for combinations of the levels of both *SOIL* and *FIELD*.

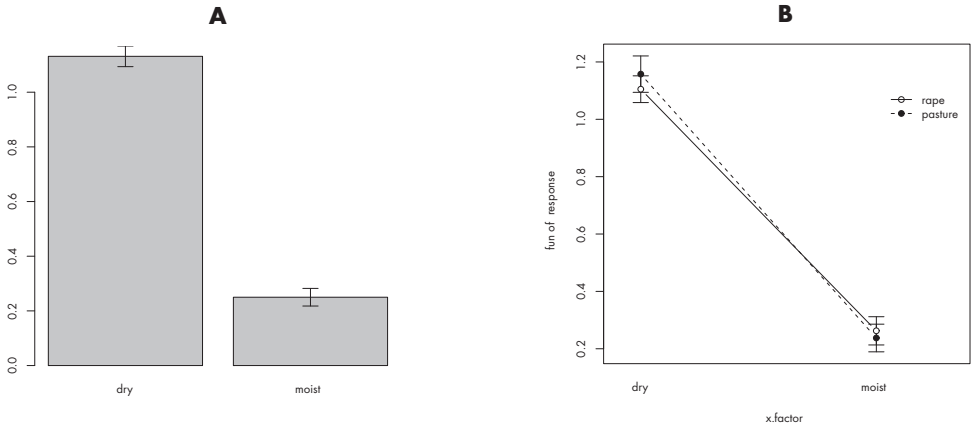


Fig. 3-10 A. Bar plot showing mean (± SE) of *SOIL* levels. **B**. Line plot showing means (± SE) of level combinations of *SOIL* and *FIELD*.

```
> library(sciplot)
> bargraph.CI(soil,amount)
> lineplot.CI(soil,amount,group=field)
```

The label of the ordinate is missing in the bar plot (Fig. 3-10A). It can be added using the **ylab** argument. In the second plot (Fig. 3-10B), the name of the abscissa could be renamed using the **xlab** argument.

3.5.10 CURVES

Lines or curves are added to the scatter plot either using a general commands **lines**, **curve** or by means of a specific command **abline**. The function **lines** connects given points by straight line segments. The resulting piecewise "linear" function can approximate even very complicated nonlinear curves. The main arguments of this function are: name of the vector for abscissa (**x**), name of the vector for ordinate (**y**), and the type of line (**type**). Clearly, in order to get a good approximation of a complicated nonlinear but smooth curve, it is necessary to specify many points (with densely placed x coordinates). The **curve** is more convenient to use when drawing curves. Here we do not need to create an auxiliary vector for values on abscissa and ordinate. Instead, it requires an equation (defining the curve) to be specified. The **abline** function draws only a straight line, with user-specified parameters: intercept (**a**) and slope (**b**). These can be either specified explicitly by the user, or they can be taken from an object obtained previously by a call to a linear regression.

In order to become somewhat familiar with drawing curves, we will refresh the visual image of the shapes of commonly used functions. Particularly, we are interested in rational, square root, logarithmic, power, exponential, logistic and quadratic functions. We will draw two curves into each plot (overlaying them). The procedure for their drawing using **lines** is as follows: first of all, we create auxiliary variable x (with a sufficiently fine step between

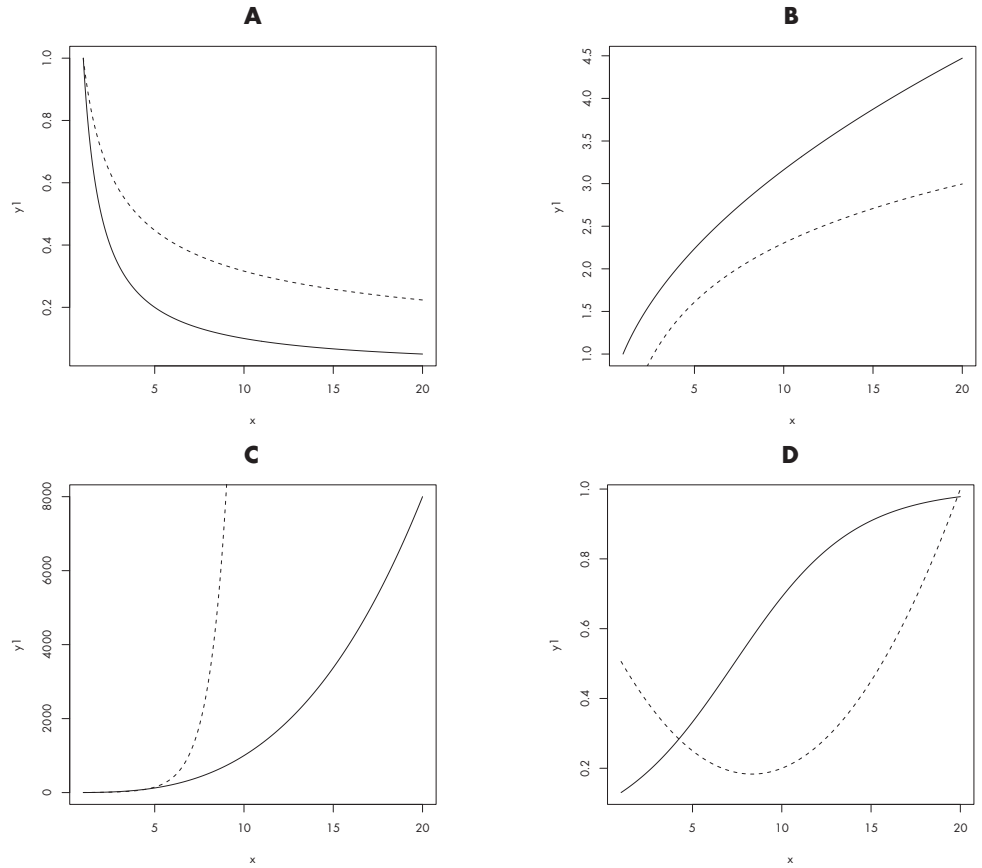


Fig. 3-11 Plots of functions: **A**. $y = \frac{1}{x}$ (—), $y = \frac{1}{\sqrt{x}}$ (----). **B**. $y = \sqrt{x}$ (—), $y = \log(x)$ (----). **C**. $y = x^3$ (—), $y = e^x$ (----). **D**. $y = \frac{1}{1 + 9e^{-0.3x}}$ (—), $y = 0.6 - 0.1x + 0.006x^2$ (----).

two consecutive x entries). We insert values calculated using a mathematical function (for example, $1/x$) in vector y. Subsequently, we create an empty plot (argument **type="n"**) and draw the first function using the **lines** command (implicitly, i.e. using a full line). Next, we draw the second function in the same plot using **curve** with the argument **add=T** and a dashed line (**lty=2**) (Fig. 3-11A).

```
> x<-seq(1,20,by=0.01)
> y1<-1/x
> plot(x,y1,type="n")
> lines(x,y1)
> curve(1/sqrt(x),add=T,lty=2)
```

That was easy, wasn't it? Using the same procedure, we suggest you draw the other six functions, shown in Fig. 3-11B–D (and then any other functions you wish) to practice the approach.

4 STATISTICAL MODELLING

There are many types of statistical model for various structures, various levels of mathematical difficulty and for various objectives (see Davison 2008). Statistical models will be with us from here to the end of this book. We will focus only on (univariate) models of the "regression type", i.e. models that describe relationships between a single response variable and one or more explanatory variable(s). Even though at the first sight, it might seem to be a restrictive view, such a model class is very large and includes many cases of important models used very often in practice. Obviously, this class includes linear regression, but also much more. We will focus on a generalisation of what is called the general linear model. This is a model that includes not only (simple and multiple) linear regression, but also analysis of variance (ANOVA) and combinations, for example, analysis of covariance (ANCOVA). The generalisation which we will be interested in this book is called the generalized linear model (GLM). While GLM includes models with normal (i.e. Gaussian) response, it allows for much more general responses in order to deal with markedly non-normal data (e.g. discrete or heavily asymmetric, or skewed continuous).

4.1 Regression model

A statistical model of the **regression type** (to be short, we will later talk only about a statistical model, or a model) is *something* that almost every reader of this book must have encountered before – either in a basic statistic course or when analysing data. Whoever has analysed data on a relationship between a response and some other variable/variables almost certainly used some regression modelling, either formally or informally (e.g. in a graphical form). A regression model includes **systematic** and **random** parts. For example, think of a study focused on a relationship between fecundity (y) and body size (x) in a garden spider. When the relationship can be approximated with a line, i.e. with a model having two free parameters, intercept and slope (α, β), then the regression model includes a systematic part of the form ($\alpha + \beta x$) and a random part, ε. The random part describes random "errors". These can be measurement errors or other random disturbances – generally this part incorporates a random variability, i.e. deviations of measurements from the systematic part. Taken together, we get a formula

$$y_i = \alpha + \beta x_i + \varepsilon_i, \tag{4-1}$$

by the means of which we postulate that the i^{th} observation (out of n observations, $i = 1,2,...,n$) of fecundity is a linear function of the body size (with intercept α and slope β)

plus random error. However, such a model formulation is neither the only possible, nor automatic. Obviously, the random error could be, for example, multiplicative. The model specification is based (or should be based) on our view of the modelled and measurement processes. To make a specification of the statistical model complete, we have to also specify characteristics of the random part, for example,

$$\varepsilon_i \sim N(0,\sigma^2)\ ,\ \mathrm{cor}\,(\varepsilon_i,\varepsilon_{i'}) = 0 \quad \text{for } i \neq i'. \tag{4-2}$$

Such a specification includes the assumption that the errors have a normal distribution (N) with zero mean (0) and some variance (σ^2). Further, the correlation (cor) between the errors (random parts, in general) of two different observations (say i and i') is zero. Together with the assumed normality, this implies that they are statistically independent (independence is a stronger assumption than lack of correlation in general). The structure of the model 4-1 (with only one random part described in 4-2) says that not only errors, but also any two different observations are independent. This model specification also implicitly includes the assumption that variance is *identical* for all observations (σ^2 is a constant and does not depend on i) – i.e. we are dealing with the so-called **homoscedastic** model. These are typical model assumptions that lead to explicit estimates based on the least squares method. These estimates are easy to use and are now available in many statistical (and other) software packages (even for models with a systematic part much more complicated than that in 4-1). We should note here, that this computation "advantage" should not form the main criterion when specifying a real model. The model should adjust to the data and process of interest, and not the other way around!

The model includes unknown parameters α, β, σ^2. To determine (identify) the model from the data we have to specify their values. If the random errors were identically zero and hence $\sigma^2 = 0$ (for this, we would have to conduct error-free measurements, the spiders would be homogeneous, etc.), then we could compute α and β from the data exactly (provided that the sample size n was larger than one). That is, of course, impossible and that is why we can acquire only an estimate based on the few (and not completely exact) measurements available. To identify or to estimate a model then means to obtain estimates of all of its unknown parameters. In our case, it means to estimate the values of α, β, and σ^2. For simplicity, we will denote parameter estimates by the corresponding Latin letters, in this case a, b, and s^2. If we know the estimates of the given model parameters, we can compute estimates of the other quantities, which can be sometimes even more interesting than the parameters themselves. For example, we can determine an estimate of the average fecundity for a body length of x_0 (where x_0 can easily be a value for which we do not have any measurement) according to

$$\hat{y} = a + bx_0. \tag{4-3}$$

As the "hat" (circumflex) in the formula emphasises, $\hat{y}$ represents an estimate. In this case estimate of the mean value for observations conducted for the size of x_0 (given the simple structure of the model (4-1), it is also an estimate of any observation conducted in x_0).

4.2 General linear model

Looking at the model above, you have certainly recognised a simple linear regression. There, the explanatory variable is continuous. The systematic part of such a model can be easily expanded into a more general form by including more than one explanatory variable:

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \varepsilon, \quad \text{where } \varepsilon \sim N(0,\sigma^2), \text{ independent among measurements.} \tag{4-4}$$

You are likely familiar with this model – for continuous explanatory variables (x_1, x_2, ..., x_k), it is a multiple linear regression. Not all x's need to be continuous variables. Some (or even all) of them can be indicators of levels of one categorical variable or of several categorical variables. In this way, we can write a model of Analysis of Variance (ANOVA) with one or more factors, or a model, which has both factors and continuous explanatory variables, but not only that – the systematic part of the model (4-4) with continuous x's can describe a variety of curved relationships between the response and the explanatory variables. Yes, using the **General Linear Model (LM)**, we can describe even nonlinear relationships (such as, polynomial, logarithmic, and others). The *linear* in the name of LM does not stand necessarily for the linear shape of the relationship between the response and the explanatory variable or variables. In fact, this piece of standard statistical terminology really means that the model is linear with respect to parameters. How do we find this? Easily. If we consider the right side of the equation of the model (4-4) as a function of the parameters, the first derivative with respect to any parameter must not include any parameter (whether it is the same as that by which the derivative was taken, or any other parameter). In other words, the systematic part of the model must be composed of linear combinations of parameters, i.e. sum of multiplications of the type: *parameter*explanatory variable.*

A linear combination of explanatory variables (without the random part, ε) is called the **linear predictor**. It is a function that predicts (i.e. estimates) the expected value of a response variable for a given combination of explanatory variables. In the linear predictor of (4-4), the generic dummy variable x can be replaced by values of any *known* nonlinear function of the original explanatory variable, say u, which does not include unknown parameters, namely $x=\log(u)$, $x=u^3$, or $x=\cos(u)$, etc. Such models will have curved shapes with respect to the explanatory variable u, but will be linear in parameters with respect to x.

Nevertheless, there are many mathematical models for which a mere substitution on the right side of the equation is not sufficient for the transformation to a linear form. Some of them can be transformed to a linear form (with respect to parameters) by **transforming** both sides of the equation. For example, an exponential function can be made linear easily by taking a natural logarithm

$$y = e^{a+bx} \rightarrow \log(y) = a + bx. \tag{4-5}$$

But be careful, things are not as simple as they may seem in (4-5)! This equation describes just the transformation of the systematic part of the model. We have said nothing about what

consequences the transformation will have for the random part. Depending on the concrete model and the transformation, the statistical properties of the transformed error part might (or might not) be much more complicated than those of the original. We can encounter several complications when performing "naive" transformations. Their results have to be thoroughly thought through before using a given model for the analysis. For example, if we "elaborate" the exponential relationship in (4-5) into a statistical model by introducing an additive random part

$$y = e^{a+bx} + \varepsilon\,, \tag{4-6}$$
$$\text{where } \varepsilon \sim N(0,\sigma^2), \text{ independent among measurements,}$$

it is clear, that the **logarithmic transformation** to a linear form is not admissible. This is because if the normal error ε achieves a value lower than $-e^{a+bx}$ with a non-zero probability it puts us in a situation where we are not able to evaluate the logarithm (of a negative argument). This means that for this model the "linearisation" through taking the log of both sides is not possible! Nevertheless, if we stick to the same systematic part and consider a multiplicative error (we should do that not because of our laziness but because it is reasonable for the given data and measurement process, e.g. when we believe that the measurement errors are multiplicative):

$$y = e^{a+bx} e^{\varepsilon}, \tag{4-7}$$
$$\text{where } \varepsilon \sim N(0,\sigma^2), \text{ independent among measurements,}$$

then the logarithmic transformation works well – it transforms the previous model to:

$$\log(y) = a + bx + \varepsilon\,, \tag{4-8}$$
$$\text{where } \varepsilon \sim N(0,\sigma^2), \text{ independent among measurements,}$$

If we replace $\log(y)$ by z we get back to the simple regression model with normally distributed additive errors (and the response variable z).

Note that distributions of the multiplicative errors in the original (4-7) and additive errors in the **transformed model** (4-8) are very different. While e^{ε} has lognormal distribution, ε has normal distribution. Model (4-7) assumes that the original data (y) are log-normally distributed (must be positive, skewed to the right, with non-constant variance that is related to the values of an explanatory variable – i.e. the explanatory variable influences not only expected value, but also variance, and higher moments), whereas transformed values (z) are normally distributed (with a constant variance). The logarithmic transformation here solved three problems at the same time:

- Linearised an originally non-linear relationship
- Transformed originally more complicated errors to normality and additivity
- Removed non-constant variance of measurements (removed dependence of variance on the explanatory variable x, or **heteroscedasticity**)

The result is that, after the transformation, we can use the simple linear model, which is really easy to work with. The parameters can then be estimated by the method of least squares. Such a model has a simple statistical theory. That can assist you in conducting correct and powerful tests, computing confidence intervals, etc. The theory also demonstrates that many commonly used inference procedures derived from the (transformed) model are actually optimal (uniformly best in rather broad classes of competing procedures). Unfortunately, such a clear and simple situation is rather rare in practice. Under only a slightly more complicated model, different problems (error non-normality, error non-additivity, error homoscedasticity, linearity in parameters) would usually require different transformations. That is a problem because, for example, simplification in the sense of linearity can *substantially* complicate the error distribution and other model properties (up to the point that the transformation approach might easily become unfeasible).

Further, it is clear that even if we do not take into account the problems with the random part, not all nonlinear relationships can be linearised. For example, the model

$$y = \alpha(1 - \beta e^{-\gamma x}) \tag{4-9}$$

cannot be transformed into a linear form without knowing the values of some parameters. In these cases, we have to resort to a more general approach based on the **nonlinear regression**. The nonlinear character of a model can then cause complications: parameter estimates may be unstable, they may not be well defined, and their statistical characteristics and interpretations are often much more complicated than in the case of linear models. This is amplified further by the fact that many of these (and other) problems are not easy to detect (especially by the layman) and hence can secretly "spoil" well intended analyses.

A general linear model can be fitted in R by a call to the `lm` function. This function has several arguments. We will learn more about some of them in the following chapters, both in general terms and in using them on practical examples.

4.3 Generalized linear model

The **Generalized Linear Models** (GLM) offer considerably larger freedom in the specification of regression models than the (general) linear model framework. In the GLM the linear predictor does not necessarily model the expected value of measurements (μ, 3-1) directly. In fact, it models the transformation (the so called link transformation) of the expected value. The transformation acts on the unknown μ and not on the observations – so that the GLM is very much different from the transformation models mentioned in the previous chapters. One of the main ideas of the GLM modelling is that the link transformation can be totally separated from distributional assumptions and from the specification of the linear predictor. As we will see later, this is an extremely valuable flexibility, very much appreciated in modern statistical modelling. The statistical model of the GLM class has a linear predictor with k ($k \geq 1$) explanatory variables. This is specified analogously as in the case of LM, but the left side of the systematic part is different – a suitably transformed expected value of measurements y appears there. So that we have

$$f(\mu) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k , \tag{4-10}$$

where $f(\mu)$ is a (monotonous) transformation function f, called **link**. A careful reader must have noticed that we have not yet mentioned the random part of a model, ε. Where did it disappear to? It did not disappear, we have not defined it yet. In the GLM, the two parts are defined separately to get greater flexibility. So, to complete the model (4-10) we have to add

$$y \sim distribution,$$

determined by the type and (unknown) values of the parameters (which we will estimate from the data). In this notation (commonly used in statistics), a tilde (~) will always mean "has a distribution". Do not be confused by the use of tilde in R where it is used to replce =. The linear model from (4-4) can be thus written in the following manner:

$$\mu = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k , \quad \text{where } y \sim N(\mu, \sigma^2) . \tag{4-11}$$

We want to emphasise that this is only a different way of writing model (4-4). The same error stipulations apply to $\varepsilon = y - \mu$ in model 4-11: it comes from a normal distribution (N) with a zero mean value (μ) and a constant variance (σ^2).

All models of the GLM class consist of three parts:

- Linear predictor
- Link
- Random part

We have already learned that the linear predictor can take the same (quite flexible) form as the linear predictor of LM. Therefore, it can include a number of explanatory variables (which can be either continuous or categorical). Exactly as before, the explanatory variables can be either the direct values of various covariates or their transformations (for example, powers, logarithms, trigonometric functions, etc.) of the original variables. The only restriction here (albeit substantial) is that the transformation cannot include any unknown parameters. Known parameters are obviously no problem at all.

Because of the link nonlinearity, but also because of the specificities given by distribution of the random part, the majority of GLM models are inherently nonlinear. Very often, they are nonlinear not only in the parameters, but also in the explanatory variables. This may sound like a paradox but the GLM class is actually a special case of the (much larger) nonlinear regression model class. Why consider it separately from general nonlinear regression? There are many practical (computational, interpretational) and theoretical (rather detailed asymptotics for GLM) reasons for this. In fact, the GLM class is so cleverly defined that it can deal with many practical data situations in a relatively very easy, comfortable and unified manner (which would not be available for all nonlinear regression models).

Table 4-1 Overview of canonical link functions and relationships between expected value (μ) and variance (v) for distributions available in GLM (n for binomial and quasibinomial error structure corresponds to the number of observations).

Distribution	Link name	Link function	Variance function
Gaussian (normal)	Identity	1	μ
Gama	Inverse	$\frac{1}{\mu}$	μ^2
Inverse Gaussian		$\frac{1}{\mu^2}$	μ^3
Poisson	log	$\log(\mu)$	μ
Quasipoissonovo	log	$\log(\mu)$	$\phi\mu$
Binomial	logit	$\log\frac{\mu}{1-\mu}$	$\frac{\mu\,(1-\mu)}{n}$
Quasibinomial	logit	$\log\frac{\mu}{1-\mu}$	$\frac{\phi\mu\,(1-\mu)}{n}$
Quasi	Any (subject to possible compatibility restrictions)	Any (subject to possible compatibility restrictions)	A constant or one of pre-selected functions of μ (subject to possible compatibility restrictions)

There are (infinitely) many link functions usable in GLM. Some of them are used very often and became almost standard. Several of the most popular link functions are listed in the Table 4-1. For a specific distribution within GLM, typically a few of the standard links can be chosen. Note, that one of the available links is identity, i.e. a link that actually does not transform the values of the expected values (so that $f(\mu) = \mu$). For each distribution, there is a unique **canonical link**, which has several nice properties which make computations and statistical properties of the parameter estimates somewhat simpler. Nevertheless, it is by no means necessary to stick to the canonical link if there are subject-matter related (e.g. biological) reasons to pick another link (provided that it will not cause problems with incompatibility of the link domain and eligible range of μ for the given model and data).

The random part in GLM is specified by selecting the distribution. The distributions eligible for GLM must come from the so called exponential class (Morris 2006). This includes distributions like normal (Gaussian), exponential, inverse Gaussian, Poisson, binomial and others. As we can see from Table 4-1, in the R implementation of GLM we can work with distributions that are qualitatively very different. The first three of them are continuous, while the Poisson and binomial are discrete. We will discuss the quasi-types below.

By choosing a certain distribution we also (implicitly) specify the relationship between expected value (μ) and variance (v) (see Table 4-1 and Chapter 7). While for the normal

distribution the expected value is not related to the variance (under the assumption of homoscedasticity), for all other distributions, this is typically not the case. If the variance is a function of the expected value then we, in fact, deal with a **heteroscedastic** model. In other words, should we model an expected value that is related to the explanatory variable(s) x, we model at the same time how the variance depends on x (through the implicit relationship between mean and variance incurred through the choice of a particular distribution). In this way, we deal with (often severe) heteroscedasticity in a natural way. This feature is one of the big GLM advantages. It increases flexibility and real-world applicability of GLMs tremendously.

Nevertheless, in some situations the relationship between the expected value and variance dictated by the distributional choice can be somewhat restricting. For example, many data that otherwise behave very similarly to the Poisson or binomial distributions show significantly greater (or smaller) variance in comparison with the variance expected based on the definition for the Poisson or binomial distributions (Table 4-1). Then it is convenient to introduce a more general specification that allows for specifying variance as a ϕ-multiple (where ϕ is an unknown constant, i.e. one of the parameters estimated from the data). Note that here we have introduced a kind of quasi-model specification. That is, a specification which, unlike in the case of particular distributions, does not specify the statistical model completely (it specifies only the first two moments: expected value and variance). See the quasi-Poisson and quasi-binomial distribution (Chapters 10.3 and 12.3) – they are often very useful for the description of under or overdispersed Poisson or binomial data, i.e. data showing excessive or deficient variance compared to the nominal distribution. Even more general is the option called quasi, where the user can specify the relationship between variance and the expected value by herself/himself (subject to certain compatibility restrictions). Similarly as in the quasibinomial and quasipoisson cases, this quasi-specification does not determine the data distribution completely. Nevertheless, it allows for relatively flexible modelling of its first two moments.

For GLM, the criterion of the quality of the fit is generally not the residual sum of squares (SS) nor residual mean squares (MS) known from LM, but a **deviance**. It is a statistic that is related to the logarithm of the likelihood ratio, i.e. a criterion that is widely used for the construction of (asymptotic) tests of hypotheses. For GLM with a normal distribution, the calculation of the deviance is identical to the residual sum of squares, but for other than normal distributions, it is different. In the model output, you will find two types of deviances: null and residual. **Null deviance** is analogous to the total sum of squares corrected for overall mean (SS_T) in LM. Similarly, the **residual deviance** is analogous to the error sum of squares (SS_E). You can consult other books on GLM, such as Crawley (2002), for the detail of the formulas for calculation of deviances of various distributions. The total deviance can be split into several parts that are related to the model components similarly to the way the total sum of squares are split in standard ANOVA. Results of the so called analysis of deviance are then displayed in a table that is analogous to the ANOVA table. Sometimes it is called an **ANODEV table**. There are differences in the underlying theory, but interpretation for testing effects of different components of the linear predictor (appearing on different lines of the table together with their respective p-values) is similar.

In comparison with LM, fitting a GLM is computationally somewhat more difficult. It is based on optimisation built on the so-called Iteratively Reweighted Least Squares (IRLS). All models of the GLM class have a unified computation approach and a unified implementation (in R, the numerical calculations are based on an extremely stable and efficient state-of-the-art matrix implementation). Apart from that, a relatively extensive statistical theory has been built for GLM over the last 40 years or so. Generally speaking (apart from a special case of normal distribution), this theory is not exact but asymptotic, i.e. "approximate", in the sense that the approximation improves in a well-defined manner with the growing sample size.

The function for fitting GLM models in R is (quite intuitively) called **`glm`**. While it has some arguments that are not present in the **`lm`** function, many of the arguments are intentionally the same. We will discuss some of the most important **`glm`** arguments in Chapters 9–12.

4.4 Searching for the "correct" model

One of the objectives of practical modelling in biology is to find the "correct" model. Quite obviously, it is clear that such a task is not easy for many practical, but also fundamental, reasons.

What is the correct model?

To the surprise of many, the answer will not be absolute. It depends on the objectives. However, generally speaking, it is a model that describes well the study phenomenon/relationship from the subject-matter (for example, biological) perspective by the means of its systematic and random parts. The model *should capture* important characteristics of the studied process, and it *should leave out* those characteristics that are not so important (in the given context and for a given purpose). Different things can be essential for various applications. It means that different models can be found and even successfully applied to the same data when they are analysed for different purposes! From this perspective, it looks as if the idea of a single correct model is not plausible. Very often, it is clear what models are not good for a given situation, but there are typically several models that appear to be quite good for the given data, process and purpose. Very often, no clear winner can be proclaimed by means of automatic, purely statistical procedures. Effective selection between good-quality candidates should also take into account subject-matter expert knowledge, a theory, etc. That is why (competent) modelling is so difficult. Certainly, it cannot be reduced to a mere set of formalised rules, advices, and axioms. Should this be possible, all data analysis could be done completely automatically (after being programmed) – thus letting machines conduct the work. All that a researcher would need to do, would be to throw the data into a computer and wait for an answer which would be automatically obtained, final and indisputable. Something like this is hard to imagine in even the most vivid dreams (despite the fact that some software producers try to leave an impression that they can do it – using popular buzzwords like "big data analysis", "artificial intelligence", etc.). To the contrary, statistical modelling is, intellectually, a highly demanding activity, a big part of which consists of critical thinking, background knowledge, knowledge of the properties of statistical methods used, experiences and last but not least, the ability to learn from your own errors as well as the errors of others.

Let us be more specific. Imagine that you study the fecundity of ten species of spiders that belong to the same family. An obvious result of the study would be the average fecundity for each species, i.e. a model with ten parameters. Should we be really interested in the fecundity of these ten species studied, the correct model would have ten parameters (plus another parameter or parameters that describe variance, which we are not often interested in, but which plays a role in the model, subsequent tests, and other inferences so that we need to estimate them somehow if they are not known a priori). This type of model is often called a **saturated model**. It is quite big (from a certain perspective, the biggest possible). Often, it would be convenient to make it smaller. One reason might be that then we would not have to predict so many parameters and would be thus able to work more efficiently. Another reason might be that we are really interested in extracting knowledge that is generalised beyond the species used in the study. Such a generalisable model can be based, for instance, on the idea of fitting a relationship between size and fecundity. Should a (reasonably linear) relationship exist between the body size of spiders (of a specific taxonomic group) and their fecundity (provided that random errors would behave additively), this relationship could be

$$fecundity = \alpha + \beta\ size + \varepsilon\ . \tag{4-12}$$

As we can see, the systematic part of such a model has only two unknown parameters (α, β). This model is more **parsimonious** than the previous one. The parsimony requirement fully complies with the often mentioned principle of the Occam's razor: "The explanation of any phenomenon should be as simple as possible." Taking this principle into the statistical context, a simpler model is better than a more complex one if it provides an explanation of similar quality. The second model is (much) more ambitious (and "scientifically desirable") than the first, because it implicitly includes a generalisation beyond the ten species observed in the study. It assumes one generally valid relationship based on which the fecundity for various species can be computed. It offers many possibilities to be tested (e.g. on new observations). Hence, it is easily falsifiable (in the Popperian sense, Popper (1959)), or open to criticism and more easily rejected in the case that it is not correct. Certainly, this is a desirable feature. The first model has been rejected on ambitions as it effectively says: "For every dog a different master", leaving very little room for being falsified. These and other reasons are why the second model is often (much) more useful in practice.

Generally speaking, we have to remember that every model represents only an *approximation* of reality. Truly, no statistical model can be completely correct in principle. To this end, a famous statement of the famous statistician G. E. P. Box is often quoted: "All models are wrong, but some of them are useful." Models that can be well used for practical purposes must always simplify the reality somewhat. If they did not, they would be as useful as a map with 1:1 scale. Reading such a map would be as "simple" as walking through the actual area, i.e. it would be practically useless. It is thus obvious that the complexity of models have to be carefully considered. Generally, a more complex model explains things better, however, does it describe so much better that it is worth to endure larger complexity? A more complex model might, for instance, be much harder to interpret, to generalise, and to understand. One should be always weighting an improvement of the fit against increased complexity (with all of its practical and philosophical consequences).

When searching for a model, we should use our brain heavily, i.e. we should consider its interpretation, we should try to judge how it complies with the previous theories and practical experience, etc. It is possible to proceed with "closed" eyes (or thinking turned off); however, we should do that as little as possible, for example, in "emergency situations" only. That is, in situations when we do not have any credible overhead information available about the study phenomenon – and the analysis is supposed to help us only for an initial orientation in the problem of interest.

4.5 Model selection

Once a basic model is specified (presumably based on some preliminary knowledge and experience with the studied phenomenon), it can be modified in various ways, with the objective to either enrich or simplify. We usually adjust the systematic part first (as a component that describes features that are of largest importance). Subsequently, we inspect and modify the random part. The problem is that while the systematic and random parts are well separated from the theoretical model specification viewpoint, the separation is much less clear when we start to modify the models (misspecification of the systematic part can easily *infiltrate* to the random part and *vice versa*).

We can seek the final form of the systematic part either "from the top" by using **backward selection**, i.e. by simplifying the original unnecessarily large model by leaving out its superfluous terms, or "from the bottom" using **forward selection**, i.e. by expanding an originally small model by adding additional terms. Alternatively, we can also use a combination of both approaches (**stepwise selection**), or even more complicated strategies. It is intuitively clear that there are many possible procedures or selection strategies. Obviously, an ultimate answer is obtained by going through the list of all possible models for the given data and select one that optimises a pre-selected criterion (all of the other strategies are its more or less suitable substitutes and shortcuts). However, this is possible only if the set of potential models is not too large.

The "top-down" procedure is often recommended for practical cases (provided the size of data and our capacities allow it), so we will only use this strategy here. It is good practice to verify a resulting model by means of so called "sensitivity analysis". That is, by conducting tests of mildly modified models (e.g. by addition or removal of a single term, by modifying functional form in which a covariate enters the model, or by adding or removing interaction terms, etc.) against the presumably final model. It is good to realise that forward, backward and stepwise selection procedures need not select the same model. More often than not, the selected models differ, sometimes substantially. This is because these strategies are somewhat myopic in character (each step looks at only very local improvements). Neither of these strategies searches through all possible models and hence it overlooks (a lot of) paths that might go to improvement after suffering from deterioration in several steps.

4.6 Model diagnosis

An important aspect of a statistical model's adequacy is how it "fits" the data from which it was estimated, identified or trained. It is clear that this quality can be evaluated in many different ways. One of the simple ways is using suitable summary characteristics which express the quality of the fit, which is often related to the "average size" of suitably defined residuals.

An even better way of checking the model quality is to verify its prediction abilities. In principle, this can be done in a straightforward manner: by comparing predictions and reality using independent data (that have not been used in the estimation of the model parameters). The problem is that independent data are sometimes not available or the total number of the data is small so we cannot afford to keep any data aside. In that case, we can use more sophisticated statistical instruments. One approach is based on **cross-validation**. When using this method, the model is fitted to all measurements with the exception of, for example, one measurement and the result is then used for predicting this excluded measurement. The prediction is subsequently compared with the actual observation. That way, one **cross-validation residual** is obtained (unlike a common residual, which is the result of a comparison of a prediction and the actual observation that has been used for computing the model parameters). The entire procedure is then repeated with another measurement being omitted until all the measurements gradually take the place of the omitted measurement. a set of cross-validation residuals is thus created, based on which a summarised estimate of the model's predictive abilities can be obtained. While the procedure might seem extremely computationally intensive, it can, in many cases, be enhanced considerably by using matrix algebra (Woodbury-Morrison formula) (Press et al. 2007). Similar tools can be used to obtain two-fold cross-validation (residuals from prediction of two omitted observations, for all possible omission pairs), three-fold and, generally, *k*-fold cross-validation. Another possibility is to use bootstrap (Efron & Tibshirani 1993).

Several diagnostic tools are used for critical assessments of the model quality. They are based on exploring residuals and their various aggregated aspects. Such a diagnosis is used for assessing both the systematic and the random components. Nevertheless, we have to keep in mind that we cannot ever *prove* that a model is adequate using any (finite) set of diagnostic tools. It is because there are many, potentially practically infinite, ways in which a model can be incorrect or inadequate. If a model "passes" several tests that we are able and willing to perform (i.e. if we do not detect any serious problems), it does not mean that the model is perfect! Even more importantly, the limited number of (often very generic) checks cannot certainly show that we found what we are looking for (a model that will be good for the purposes of our study). Passing a few checks only means that we have not found anything that the model is inadequate at handling. Obviously, it can be that the model is indeed adequate, or it can be that we have not looked hard enough. We have to realise this when we interpret the results and when we compare the model with other models. It can easily happen that several models can pass the same group of checks in a given situation. The net result is that, based on statistical tools only, we can almost never say that a model is "completely adequate". That decision should take into account also subject-matter expert knowledge. Nevertheless,

it is imperative to do as much as possible – and try to check important features of the model. This is not only a matter of mechanically applying a set of tests and/or graphical procedures (even though they can be useful as a part of model checking and model criticism phases). It is, in fact, a matter of thinking about the model and its purpose and creatively exposing various aspects of the model fit (or prediction abilities). To this end, one often uses plots focused on various features. It is one of R's features that it is easy (and even fun) to prepare and check the plots (both standard and new, devised creatively by you for a problem at hand).

The objective of the model diagnostics is to determine if and to what extent our data comply with the given model assumptions and how to possibly make it better. When the model assumptions are severely violated, it is of course problematic to believe the model predictions, not to mention parameter estimates or conclusions of tests about them. Very often, the checks of the assumptions are conducted on the residuals after the model. As a rule, no model is completely free of bias, there are always some assumptions. Nevertheless, different models are based on different assumptions (some have more prerequisites, some only a few). For example, for the random part, one powerful (and often invoked) assumption is:

$$\varepsilon \sim N(0,\sigma^2), \text{ for } i = 1,2,\ldots, n, \quad \text{and } \operatorname{cor}(\varepsilon_i,\varepsilon_{i'}) = 0 \text{ for } i \neq i'. \tag{4-13}$$

Of course we do not know the real values of the random errors, ε. Therefore, we cannot look at them directly. Nevertheless, once we predict the model, we have their estimates, i.e. **residuals** ($e_i = y_i - \hat{y}$). In a good model, the residuals should behave *similarly* to theoretical errors, ε. Nevertheless, even intuitively, it is clear that if our computations are based on the assumed model and data of a limited size, properties of observable e will be pushed towards ε. Statistical characteristics of e and ε are somewhat different even for correctly specified models due to fundamental statistical reasons (corrections for this problem exist but we will not discuss the details). As an example, consider the residuals from a completely correct model of a simple linear regression with errors satisfying (4-13). There will be some correlation in the residuals (e_i). This is due to the fact that the parameters and residuals are estimated on the same dataset. Similar discrepancies can be found in other aspects of e. To this end, it is clear that in checking the residuals, we cannot insist on exact satisfaction of assumptions postulated for the true random errors, ε. For a good model, we just ensure that the residuals do not behave much differently than we would expect based on the facts we know about, ε. At least, the residuals should not:

- Show any trends against any of the explanatory variables
- Have a (dramatically) heterogeneous variance, i.e. they should not be (markedly) heteroscedastic
- Have a "strange" distribution (e.g. markedly asymmetric, bimodal, etc.)
- Show strong autocorrelation when ordered e.g. in the order of obtaining the measurements/observations.

Examinations of the residuals behaviour is often rather informal. For instance, it is common to inspect various diagnostic plots. Nevertheless, there are also formal tests for checking specific departures from various assumptions. Displaying "elementary" diagnostic plots is

relatively easy in R. You just need to enter the **plot** command with the name of the fitted model as argument. Of course it is also possible to do these (and many – almost infinitely many other) plots manually by extracting e.g. residuals from the fitted model object and plotting them against appropriate variables of interest using the **plot** function (as we mentioned in Chapter 3.5). If we are interested in just the handful of basic diagnostic plots, however; the **plot** function applied to the fitted model object will do that for us quickly and conveniently. There are six pre-set diagnostic plots:

- Residuals versus fitted/predicted values
- Normal Q-Q plot of the residuals
- Standardised residuals versus fitted/predicted values
- Cook's distances
- Residuals versus leverage
- Cook's distance versus leverage

Only four of them (1, 2, 3 and 5) will be displayed automatically after calling **plot** (without specifying the **which** argument). For the purpose of a rough model quality evaluation, we will focus on the first four plots only. The last two plots (5 and 6) are also interesting and useful but only for those who know a little bit more about the basic regression theory. Nevertheless, in no way do we want to suggest that you should focus only on these plots when conducting real analyses. Real verifications will have to be more thorough, more creative. What we are showing you in the text is only a very minimalistic example of how to *start* with model checking. Maybe you will be surprised to know that even in the case of these simple diagnostic tools, things are not as simple and unambiguous (in the sense of being either "good" or "bad"), as they may look at first. Most of all, we want to motivate you to think about the models before you use their results.

The first plot shows residuals against fitted values. The type of residuals displayed depends on the function used. For **lm**, the residual e_i is defined as

$$e_i = y_i - \hat{y}_i , \tag{4-14}$$

where $\hat{y}_i$ is the estimated expected value of the i^{th} measurement – i.e. predicted value (in the case of models with identically independent distributed measurements, the two terms fitted and predicted are equivalent). For **glm**, more than one type of fundamentally different residuals can be defined. **Deviance residuals**, r_D, are defined as:

$$r_{Di} = \text{sign}(y_i - \hat{\mu}_i)\sqrt{d_i} , \tag{4-15}$$

where $\hat{\mu}_i$ is the prediction of the expected value and d_i is the contribution of the i^{th} measurement to the overall deviance. In this plot we mainly evaluate the overall layout of points. We should also check whether there are any substantially outlying residuals (too big, positive or negative). If the model explains systematic variability of the data well, the residuals should not show any significant trend: points will be distributed randomly throughout the entire plot area as "stars in the sky" (Fig. 4-1A). The automatically inserted red **loess** line (it is a non-parametric fit obtained based on a very flexible, local regression) highlights a possible

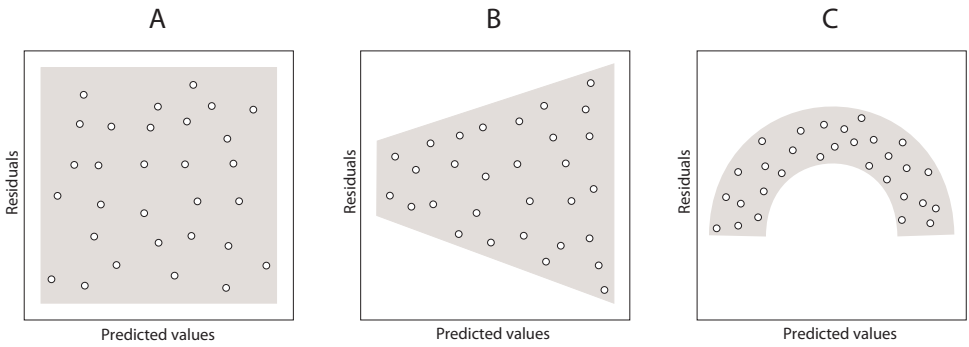


Fig. 4-1 Relationship between residuals and fitted values. **A**. Homoscedastic scatter. **B**. Variance increases with expected values. **C**. Humped pattern. The trend is highlighted with grey colour.

trend to aid the checking process. The loess curve should be more or less straight and close to the abscissa (ideally, it would be exactly on the abscissa, but we cannot expect that in real world). If the residuals have (in some parts of the plot) tendency to be distinctively above or below zero, the model under-estimates (when residuals are positive) or over-estimates (when residuals are negative) the measurements. Variance homogeneity (and/or correctness of the systematic part) is disputed if the points form some kind of recognisable shape. Very often, variance is smaller for smaller observations, leading to a fan shape of the point scatter (Fig. 4-1B). If the residuals form a bent pattern (Fig. 4-1C), they might suggest a misspecification of the systematic part. This is detected quite frequently in plots of residuals against various continuous explanatory variables. If the pattern appears in a plot against a covariate not present in the model, presence of such pattern suggests that the model should be expanded by including that variable. If the pattern appears in the plot against a variable already present in the model, it suggests that the present functional form of the variable might not be adequate. For instance, in this way we might detect a need for additional quadratic trend in a model where the variable enters only in the linear form.

The second, normal Q-Q plot, compares distribution of (standardised) residuals with the expected standardised normal distribution. **Standardised residuals**, r_S, (from **`lm`**) are defined as follows:

$$r_{Si} = \frac{e_i}{s\sqrt{(1-h_i)}}, \tag{4-16}$$

where s is a standard deviation of residuals and h_i are diagonal elements of the projection matrix. **Standardised deviance residuals**, r_{SD}, (from **`glm`**) are calculated as follows:

$$r_{SDi} = \frac{r_{Di}}{\sqrt{\hat{\phi}\,(1-h_i)}}, \tag{4-17}$$

where $\hat{\phi}$ is a dispersion parameter (i.e. parameter that is related to variability) predicted from the model (for example, its value is always 1 for the Poisson model). Using this plot, we visually (and quite informally) check the assumption about the *approximately* normal

distribution of the residuals. The word "approximately" is to be really highlighted here. For other than normal distributions of the response variable, the exact normality of the distribution of the residuals is impossible even for completely specified models. The points in the plot should thus approximately lay along a diagonal line (see Chapter 3.5.1). If the points substantially deviate from the diagonal line, you should try to transform the response variable (and refit the model with the new response). Alternatively, you can also try to specify another type of the distribution or link function, or examine whether the systematic part is appropriate.

The third plot shows the relationship between modified standardised residuals $\sqrt{|r_s|}$ (for **`lm`**) or standardised deviance residuals $\sqrt{|r_{SD}|}$ (for **`glm`**) and the fitted values. This plot is convenient for evaluating variance heterogeneity. As we mentioned previously, the residuals should ideally show homogenous variance. The residuals from definitions (4-14, 4-15) do not have exactly the same variance; very often, the variance of residuals closer to the left or right ends of the plot is different from the variance in the middle (for reasons that have nothing to do with the model quality). Standardised residuals (4-16, 4-17) correct for this inherent heteroscedasticity. The overlaid red loess curve helps us judge if the variance changes substantially and systematically with the predicted value. For homogenous variance, this red line should be *approximately* horizontal (but obviously not at zero vertical position).

The fourth plot shows residuals versus leverage. Leverage measures in some (well defined) sense how far the observation is from the centre in the space of explanatory variables. Observations that are far from the centre get a lot of influence in fitting the model parameters (their leverage is high). Ideally, this should not happen and leverage of all observations should not be more than moderate. High leverage constitute a potential problem. The problem is then manifested when the high leverage occurs at the observation which is outlying (highly atypical) from the response variable viewpoint. Then the outlier gets a lot of weight. It can then happen that in its influence upon model parameter estimates, the outlier outweighs many other "normal" observations. The residual versus leverage plot explores informally whether a problem of this type occurred. The phenomenon is studied even more closely in the Cook's distance versus leverage plot. If we detect highly influential observations (high leverage) and especially when they have large Cook's distance, we should check their validity (check for data entry and other errors carefully), think twice as to whether they are from the same process as the bulk of the other data (and hence consider whether they should be left in the dataset or omitted).

Diagnostic plots for checking the behaviour of the residuals against various explanatory variables are not provided automatically by the **`plot`** function call with a fitted model object as an argument. They need to be done separately. This means manually, extracting needed variables and then plotting them. We will graphically study dependency on the explanatory variable here only if this variable is continuous (for categorical explanatory variables, we need to assess behaviour of the residuals for each level separately – for example, their averages, standard deviations or summaries of the entire distribution using **`boxplot`** or **`hist`**). For the model from the **`lm`** function, you will use standardised residuals, r_S, from (4-16), and for the model from the **`glm`** function, you can use the more general **Pearson residuals**, r_P:

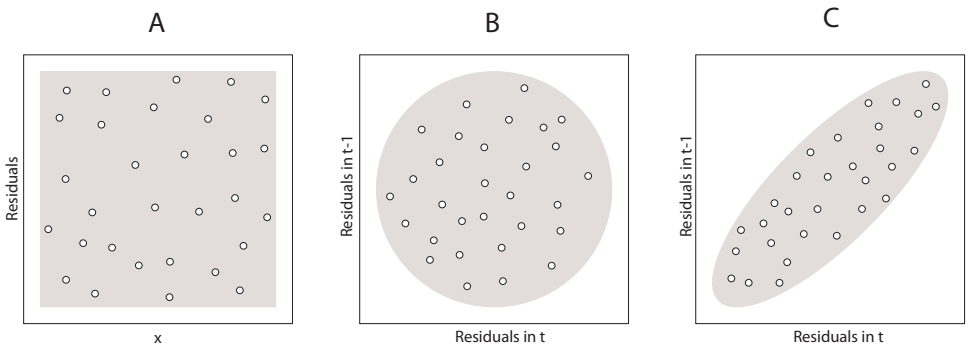


Fig. 4-2 A. Relationship between residuals and values of a continuous explanatory variable (x). **B**. Residual autocorrelation of 1st order is not present. **C**. Autocorrelation of the 1st order between consecutive residuals is positive. The pattern is highlighted with a grey ellipse.

$$r_{Pi} = \frac{y_i - \hat{\mu}_i}{\sqrt{\mathrm{Var}\,(\hat{\mu}_i)}}\,. \tag{4-18}$$

When residuals are independent of the explanatory variable, they will be arranged randomly and show no trend (Fig. 4-2A). Under normality and if the model is valid, the random vectors of fitted values and residuals are orthogonal, which gives independence; for more general models specified correctly, this should hold approximately. When the points create a recognisable shape (for example, a cloud around an ascending or descending line or curve), we might consider changing the functional form in which we enter the explanatory variable into the model (for example, we might need to add a higher order polynomial term), or we might need to add interaction, etc.

We will study the relationship between consecutive residuals only if we have relevant information about the ordering of observations in time or space. Then we might study the relationship between the consecutive residuals by (auto)correlation. We can examine many **autocorrelations** for residuals – based on the time distance (lag) between subsequent residuals (for example, observations at times 0 and 3, also 1 and 4, 2 and 5 have lag 3). The simplest autocorrelation is that of the 1st order, i.e. the relationship between the residuals that immediately follow each other (time t and time t – 1). Obviously, we can replace the word time with space if the observations are ordered along transect in space. Shall autocorrelation not be present in the residuals, the points in the plot of residual in time t compared to the residual in time t - 1 are arranged approximately in a circular pattern (Fig. 4-2B). The presence of autocorrelation changes the situation – the cloud of points will show a nonzero slope (Fig. 4-2C).

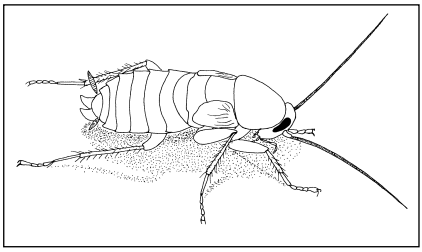
Residuals from any model fitted by **`lm`** or **`glm`** are obtained by calling **`resid`**. In the GLM different types of residuals, can be computed. They are quite fundamentally different, being on different scales. On the scale of link-transformed expected values (these have better statistical properties, but trickier interpretations), or on the scale of measurements (due to

practical reason these are of interest to us). In case of **glm** objects, the type of residuals can be requested using the **type** argument. **Working residuals** (**type="working"**) are on a transformed scale, Pearson residuals (**type="pearson"**) have been explained above, and response residuals (**type="response"**) are on the original scale. Standardised residuals from **lm** (4-16) are obtained calling **rstandard**. Fitted or predicted values from any model are obtained by calling the function **predict**. Yes, these are fits and not predictions on new data if the **newdata** argument is not specified. If it is specified and new values of the explanatory variables (or their combinations) are supplied, then predictions are returned. Fitted or predicted values from **glm** objects can be also on the original (**type="response"**) or link-transformed scale (**type="link"**).

5 THE FIRST TRIAL

In this, relatively long chapter, we will present a simple analysis of the first example, explain the R outputs produced during the analysis and discuss a few basic principles, which should be kept in mind in practical modelling.

5.1 An Example



The nutritional quality of a diet affects growth of organisms. In an experimental study, the effect of five diet types on the body mass of cockroaches was measured. Two diet types were enriched by lipids, two were enriched by proteins, and one was a control (*DIET*: ctrl, lipid1, lipid2, protein1, protein2). Altogether 85 cockroach individuals were used; ten females and seven males for each diet type (*SEX*: female, male). Individual males and females were randomly assigned to the treatments. Body mass [g] of each individual was recorded before (*start*) and after (*weight*) the experiment, which lasted for one month. The aim of the study was to find out how the diet types affected final mass of the cockroaches. Specifically, we were interested to know: (1) Was final mass influenced by the diet type? (2) If so, how did lipid and protein enrichment affect the mass? (3) Was final mass on the diets similar for males and females?

5.2 EDA

The data frame of this example includes four variables. The response variable (*weight*), is a body mass, it is continuous. Then there are two categorical explanatory variables, *DIET* and *SEX*, and one continuous explanatory variable (*start*). We are primarily interested in the effect of *DIET*. Its effect can, however, be combined with an effect of any of the two other explanatory variables (the effect of diet upon weight can depend on the sex and/or it can depend on the initial mass). Therefore, it is necessary to look at the interactions between these variables. The effect of *DIET* on the body mass can be read (informally) from the box plot (**`plot`**), whereas to display possible interactions between *DIET* and *SEX* we will use **`interaction.plot`**.

```
> dat<-read.delim("cockroach.txt"); attach(dat)
> dat
```

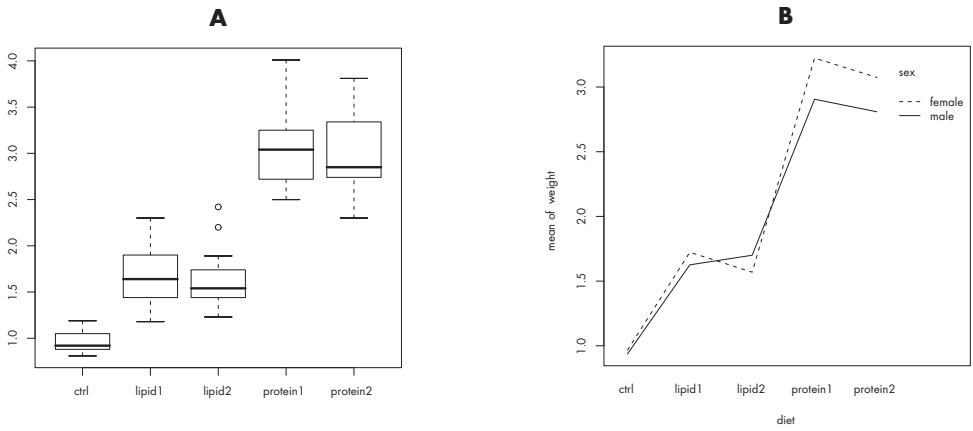


Fig. 5-1 A. Comparison of final mass of cockroaches reared on five diets. **B**. Comparison of final mass means of male and female cockroaches reared on five diets.

```
sex     diet weight start
1    male protein1   3.19 0.373
2    male protein1   3.15 0.321
3    male protein1   2.97 0.445
...
85 female     ctrl   1.19 0.254
> plot(diet,weight)
> interaction.plot(diet,sex,weight)
```

A striking difference in the mass of cockroaches from different diet groups is clearly visible on the first plot (Fig. 5-1A). Cockroaches on the control diet had the lowest mass, cockroaches on lipid-enriched diets had a medium mass and cockroaches on the protein-enriched diets had the largest mass. We can guess from the plot that the impact of the *DIET* variable will be important/significant. The interaction plot (Fig. 5-1B) demonstrates that the profiles are not completely parallel: females behaved differently than males especially on the protein-enriched diets (and partially also on the lipid-enriched diets). This fact suggests that there is an interaction. However, everything has to be verified properly using appropriate tests.

It is possible that bigger individuals increased their mass on different diets in a different way to smaller ones; therefore, in the next step we will inspect the effect of the initial body mass combined with *DIET* and with both factors (*DIET* and *SEX*); using **`xyplot`** from the library lattice. The initial mass (*start*) will be on the abscissa. Plots of relationship between *weight* and *start* for particular levels of *DIET* will be drawn separately into panels. Within each panel we will distinguish between males and females (i.e. levels of *SEX*) by means of different symbols (**`pch=1:2`**).

```
> library(lattice)
> xyplot(weight~start|diet,groups=sex,pch=1:2,col=1)
```

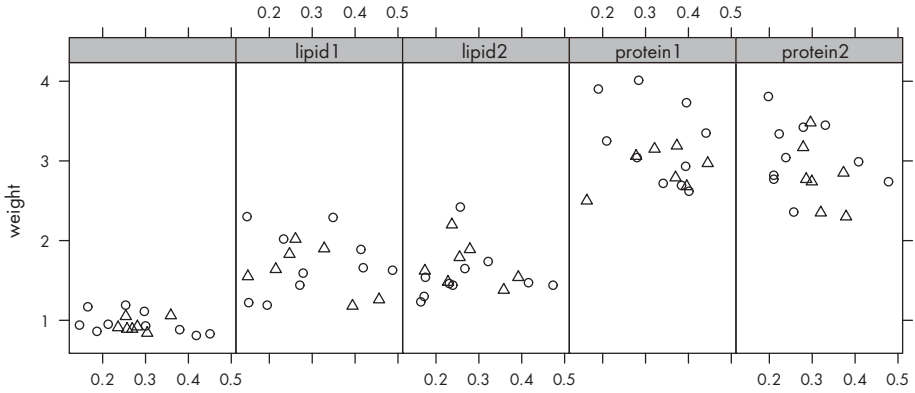


Fig. 5-2 Relationship between the final mass (*weight*) and the initial mass (*start*) in female (○) and male (△) cockroaches reared on five diet types.

Gender differences in the relationship between *weight* and *start* are generally small and do not markedly change among different diet levels (Fig. 5-2). It can be thus expected that neither the *SEX*:*start* nor *DIET*:*SEX*:*start* interactions will be significant.

5.3 Presumed model

It is generally known that the mass of living organisms is influenced by their diet type – especially by its energy and nutritional values. We can establish the basic structure of the systematic part of the model on this fact: average mass is different for different diets. With regard to the random part, we will assume normal, additive, homoscedastic independent errors (because, this assumption has proved to be useful when analysing similar data in the past). For the i^{th} measurement of the j^{th} diet level, we have the following model in the so-called **textbook parametrization**:

$$weight_{ij} = DIET_j + \varepsilon_{ij}\,, \qquad (5\text{-}1)$$

where $\varepsilon_{ij} \sim N(0,\sigma^2)$, independent among measurements.

We will discuss the meaning of model components further in the text.

5.4 Statistical analysis

We want to explain the final mass of the cockroaches in the model (5-1) only by a diet type. However, the previous exploratory analysis suggested that apart from the *DIET*, the mass of the cockroaches could have been also influenced by two other variables. That is why we start the analysis with a larger model that includes a linear relationship to the initial mass (*start*) for all combinations of the categorical variables. In such a model, we will have 10 lines (20

parameters), one for each combination of the *DIET* and *SEX* levels. The model in the textbook parametrization will look like this:

$$weight_{ijk} = \alpha_{jk} + \beta_{jk}\ start_{ijk} + \varepsilon_{ijk}, \quad \text{where } \varepsilon_{ijk} \sim N(0,\sigma^2), \text{ independent among measurements.} \tag{5-2}$$

Here *i* is used to code measurements of initial mass, *j* and *k* are used for levels of two factors: *j* = (ctrl, lipid1, lipid2, protein1, protein2), and *k* = (female, male). α_{jk} is the intercept. β_{jk} is the slope. The model allows the two coefficients to be different for different combinations of levels of *DIET* and *SEX*. Note that, the fact that the model allows for difference in slopes (or intercepts) among *DIET* and *SEX* combinations does not mean that it enforces such differences. If the data does not support the heterogeneity, the estimated coefficients will be the same (or very similar), hence allowing for such differences is not a restriction, rather it provides flexibility.

Results of the fit will be saved into an object `m1` so that we will be able to work with it later. The number one will remind us that these are results of the first model. Because we expect the response variable to have a normal distribution, we will use function **`lm`**. This function always assumes normally distributed additive errors! Thus it is sufficient to specify the systematic part by means of the model formula (more about it in Chapter 6.4). Keep in mind that the model formula does not specify the random part of the model. In **`lm`** we enter the model formula as the first argument (and in this example as the only one). In the model formula, response is specified on the left of the tilde. Here, the response variable is *weight*. On the right side, we specify the linear predictor (utilising a few rules of the general R model formula syntax). The linear predictor will include three main effects (*DIET*, *SEX*, *start*), three two-way interactions (*DIET*:*SEX*, *DIET*:*start*, *SEX*:*start*) and one three-way interaction (*DIET*:*SEX*:*start*). Such syntax is quite long; fortunately, R offers a short way to specify the same model formula using **`*`**: **`weight~diet*sex*start`**. This means to include in the model all main effects and all possible interactions (three two-way and one three-way, in this case). We defined the model in (5-2) using textbook parametrization. Nevertheless, we will fit it in a more complicated **treatment parametrization**, because it is a default option in R. Later we will explain the differences.

We will include the explanatory variables in the order of their importance, i.e. *DIET*, *SEX* and *start*. The order matters for subsequent testing when the design matrix is not orthogonal. In our case here, it is not, because the experimental design is not balanced (due to a different number of females and males used). Both types of the explanatory variables, continuous as well as categorical, are specified in a similar way (without explicitly distinguishing between them) in this particular example. This is because, when we uploaded the data frame, the character variables are automatically converted to factors. To rely on this might be quite dangerous – so we should always check the degrees of freedom in the model output to prevent foolishly fitting linear trends to factors with discrete numerical levels when we do not intend to do so. If the automatic conversion of character vectors to factors were suppressed (by specifying the **`stringsAsFactors=F`** option when reading in the data object), or if

we need to convert a numerically coded discrete variable into a factor, we can do the conversion manually (calling the **factor** function).

We can take a look at the ANOVA table, which includes the results of the tests of the various model terms, using the **anova** command with the name of the object that resulted from the call to the **lm** function as an argument.

```
> m1<-lm(weight~diet*sex*start)
> anova(m1)
Analysis of Variance Table

Response: weight
               Df Sum Sq Mean Sq  F value Pr(>F)
diet            4 58.484  14.621 118.7198 <2e-16 ***
sex             1  0.277   0.277   2.2483 0.1386
start           1  0.243   0.243   1.9770 0.1645
diet:sex        4  0.519   0.130   1.0531 0.3869
diet:start      4  0.137   0.034   0.2773 0.8916
sex:start       1  0.023   0.023   0.1905 0.6640
diet:sex:start  4  0.831   0.208   1.6874 0.1636
Residuals      65  8.005   0.123
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.4.1 ANOVA TABLE OF TYPE I

We obtain an ANOVA table that has eight rows (seven model terms + residuals). In the first column, there are names of the terms – firstly the main effects, followed by two-way interactions and then three-way interaction, and, finally, there is a residual component (Residuals). The second column presents degrees of freedom (Df), then there are sum of squares (Sum Sq), mean sum of squares (Mean Sq, i.e. Sum Sq divided by Df), values of the F-criterion (F-value), calculated as Mean Sq of a given term divided by residual Mean Sq. For example, for the factor *SEX*, F = 0.277/0.123. In the last column, there are p-values for F-tests (Pr(>F)). Next to the p-value column, asterisks or dots might appear. They show which model terms are significant at several conventional levels (see legend at the bottom of the table). Obviously, this marking with stars is not necessary since the significance of each term can be easily found by comparing appropriate p-value with the significance level desired. That is also why it is highly recommended to report p-values and not only statements like "significant at 5% level" or "p<0.05" in publications – then a reader can compare the p-value with whatever level is desired.

This is an **ANOVA table of type I**, known as sequential, because it uses tests based on the **sequential** sum of squares. This means that the terms are included in the model and tested in a sequence specified in the linear predictor. The main effects are tested before two-way interactions, and they are tested before three-way interactions, etc. The testing procedure automatically begins with the simplest terms (main effects) and continues to more complicated terms (interactions of higher and higher order). a user can only define an order for

terms at the same level. The order can have an important effect on test results (but it has no effect upon the quality of the model fit). For example, in a model `y~A*B`, the main effect of *A* is tested as coming first, then the contribution of the main effect of *B* is tested (after *A* is already in the model). Then the contribution of the *A*:*B* is tested (after both main effects *A* and *B* are in the model already). Should we compare the ANOVA table of a model specified with a different order, `y~B*A`, we would obtain identical results in the row for interaction *A*:*B*. But in rows for main effects, *B* and *A*, the results could differ because in the latter model effect of *B* is tested first and the effect of the *A* contribution is tested subsequently only after *B* is in the model. Identical results for *A* and *B* would be produced if the data come from an **orthogonal design**. This will happen for example, if the data come from a balanced factorial design – i.e. when all combinations of levels of factors *A* and *B* have identical number of replications. This also applies to models with more main effects and interactions when data come from larger balanced factorial designs.

We will demonstrate the dependence-on-order phenomenon with our example data. If we switch the order of *DIET* and *SEX*, we will see that the statistics for the two-way interactions, *DIET*:*start* and *SEX*:*start*, will change. The highest order interaction is unchanged. The residual mean square will not change either as the fit and all aspects of its quality remain the same.

```
> anova(lm(weight~sex*diet*start))
Analysis of Variance Table

Response: weight
               Df Sum Sq Mean Sq  F value Pr(>F)
sex             1  0.277   0.277   2.2483 0.1386
diet            4 58.484  14.621 118.7198 <2e-16 ***
start           1  0.243   0.243   1.9770 0.1645
sex:diet        4  0.519   0.130   1.0531 0.3869
sex:start       1  0.015   0.015   0.1247 0.7251
diet:start      4  0.145   0.036   0.2937 0.8810
sex:diet:start  4  0.831   0.208   1.6874 0.1636
Residuals      65  8.005   0.123
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We need to explain here what **orthogonality** is, because it is a very important characteristic. You surely remember from geometry that orthogonal vectors are perpendicular to each other (non-orthogonal vectors are oblique to each other at various angles). Orthogonality in statistics is related to the relationship among individual variables. Within the context of an ANOVA (and linear models generally), it is important that if the explanatory variables are orthogonal, the overall variability of the response variable can be easily split into non-overlapping and easily interpretable parts corresponding to the influence of individual model components. Apart from various theoretically motivated advantages, it is really convenient from the viewpoint of practical data analysis; the interpretation of the results is simple and indisputable under orthogonality. It may be a surprise for some people that with substantial departures from orthogonality, the interpretation of results might be very problematic – for example, the variability, which we nominally assign to *B*, can also be related to *A* to a varying extent. One of the manifestations is, for example, inconsistency of the results obtained above

by using **lm** with formulas `y~A*B` and `y~B*A`. At the extreme, when the linear relationship between explanatory variables *A* and *B* is perfect (departure from orthogonality is largest), the model is not even identifiable. Then the model is not well defined in the sense that we cannot distinguish between net *a* and *B* effects from data only (we would need some external information to do the separation). Other model features might still be identifiable (e.g. the effect of the common contribution of *A* and *B* together). R normally detects perfect collinearity and either issues an error message or produces NAs in appropriate places (so that a user should notice). In the intermediate situation, when there is some linear relation between *A* and *B*, but it is not perfect (they are partially collinear), we might encounter a lot of problems that might not be very easy to detect and interpret (especially for larger, complicated models). Generally, estimated standard errors of parameters will be inflated; the test will have reduced power and there will be serious computation difficulties (slower convergence of the IRLS algorithm in GLMs). The technical (statistical and computational) difficulties result from a problem that can be perceived very well intuitively. If we have a data set, in which, for example, two explanatory variables always go together, it will be difficult to differentiate between the effect of one variable and of the other one – with or without statistics. The closer these two variables are to a linear relationship (i.e. a situation where variables x_1 and x_2 are related to each other according to formula $x_2 = a + bx_1$ for given constants *a* and *b*), the harder the separation will be. The separation becomes impossible when the linear relationship is perfect.

To avoid (or at least, alleviate) this problem, a good study design will help. Ideally, one would collect the data from a study with orthogonal design. If that is not possible (e.g. due to financial, logistical or other constraints), one should try to come up with at least a reasonable design in which the explanatory variables are not collinear. A consultation with a statistician might be necessary to this end. The theory of experimental design is a well-established field in statistics, see, for example, Cochran & Cox (1957), Underwood (1997), Montgomery (2001), Scheiner & Gurevitch (2001), and Quinn & Keough (2002). Nevertheless, many basic principles of good design are relatively easy to follow (after a bit of reading and thinking). In general, it is good to remember that it is better to think well (or ask a statistician) about the effect the design (i.e. about the explanatory variables combination used in the study) may have on the analysis *before* a study is conducted. Consequences of a bad design cannot be overcome later in the analysis phase (For instance, some desired effects might not be estimable even after a lot of field or lab work has been done).

5.4.2 NONLINEAR TRENDS

Verification of the model quality can also lead to model expansion by addition of terms. When conducting a formal comparison of the initially proposed model with the enlarged one, the latter one should not be substantially better. If it is, the original model needs to be reassessed and improved.

One obvious way to enlarge a model with a continuous variable present in the form of a linear trend is to add higher order powers (introduce higher order polynomial trends). For example, we can try to enrich the originally linear model with a quadratic trend and subsequently test the improvement by the enlargement in order to determine if the data show

a strong curved tendency (with respect to the variable whose quadratic trend was added). a quadratic model (in variable x) has three unknown coefficients (α, β, γ):

$$y = \alpha + \beta x + \gamma x^2 + \varepsilon. \tag{5-3}$$

This model can be an effective approximation to a (relatively simple) non-monotonous trend. a quadratic term can be included in the linear predictor in one of two ways:

- By means of calling **`poly`**, specifically **`poly(x,2)`**
- By specifying linear and quadratic terms separately, i.e. **`x+I(x^2)`**

Similarly, higher polynomial terms can be included. The power function can be included in the model formula, but, because the power operator (`^`) has a different interpretation in the syntax of the linear predictor than in the usual mathematical sense, we have to wrap it into the **`I()`** function in order to tell the software that we really wish to take power of `x`. You can find more about this topic in Chapter 6.4.

Both ways of specifying polynomial models are estimating an identical model (same relationship between y and x), but interpretation of coefficients is very much different. These two specifications are, in fact, examples of two parametrizations of the same model. By means of **reparametrization** we can unambiguously *translate* one set of parameter values to the other set of variables (without losing any information). A similar phenomenon occurs in the ANOVA context (many readers will recall that the parametrization is connected with a particular choice of contrasts among factor levels) and many other models. When using **`x+I(x^2)`** estimates of a, b, c produced by **`lm`** correspond to estimates of α, β, γ from (5-3), while when using **`poly`** estimates correspond to other parameters, but the two approaches lead to fitting the same model which fits the data equally well or poorly. It is a common beginner's error to not be aware of the reparametrization and of the fact that models in different parametrization fit the data equally, even if they have different coefficients. It is not clever to think that reparametrised models are different and identical overall performance is a mere coincidence. Even worse is the attempt to directly compare parameter values from different parametrizations without understanding that they mean different things. We have seen many attempts to make new "discoveries" based on totally wrong comparisons of coefficients coming from differently parametrised models. For instance, it is not uncommon to find papers on the same subject using different parametrizations (either inadvertently or just because they use different software packages which favour different parametrization choices)!

Let's go back to our example. We cannot see any curved trend in the plot (Fig. 5-2) with the naked eye. However, we have to verify the linearity using a somewhat more rigorous method. As we have already mentioned above, the simplest (but by far not the only possible!) way is to compare the models with and without a quadratic term (while other terms remain the same). We will compare model (5-2) with the following model:

$$weight_{ijk} = \alpha_{jk} + \beta_{jk} start_{ijk} + \gamma_{jk} start_{ijk}^2 + \varepsilon_{ijk}, \tag{5-4}$$
$$\text{where } \varepsilon_{ijk} \sim N(0, \sigma^2), \text{ independent among measurements.}$$

This includes ten quadratic curves (altogether 30 parameters). Similarly to (5-2), α is an intercept, and β is a coefficient of the linear trend (for *start*). Parameter γ is a coefficient of the quadratic trend. Subscripts j and k show that for a particular combination of j^{th} *DIET* and k^{th} *SEX* we allow different coefficients.

We should always test if it is necessary to add a quadratic term to our model. The comparison of the original model without the quadratic term (`m1`) and the new model (`m2`) is done using **`anova`** function called with the two models as arguments. This function compares both models from the perspective of residual variance (i.e. how much unexplained variability remains in the data after the model) in a formal way. The comparison is based on a correct F-test. Models can be compared in this way only if one of them is a special case of the other one – i.e. if they are nested. Here, the nesting condition is met because `m1` is a special case of `m2`: we get `m1` when we leave out the quadratic term from `m2`..

```
> m2<-lm(weight~diet*sex*poly(start,2))
> anova(m1,m2)
Analysis of Variance Table

Model 1: weight ~ diet * sex * start
Model 2: weight ~ diet * sex * poly(start, 2)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     65 8.0052
2     55 6.9299 10    1.0752 0.8534 0.5808
```

In the output of this ANOVA table, the first two lines are occupied by the formulae of the two models, the first (Model 1) and the second model (Model 2). They are there to remind us of what is being compared. A test of a term (or terms) by which the two models differ follows. Below, there is an ANOVA table with a column for residual degrees of freedom (Res. Df), residual sum of squares (RSS), difference in the degrees of freedom between the two models (Df), difference in the sum of squares between the two models (Sum of Sq), the F-test value that compares the two models (F) and its p-value (Pr(>F)).

Generally, a larger model (with more explanatory variables, a larger polynomial degree, etc.) will fit data at least as well as a smaller model (and probably better). And indeed, in our example, the sum of squared residuals is smaller for the bigger model `m2`. This is obvious and comes from the fact that `m1` is a restricted version of `m2`. Nevertheless, the question remains: is the improvement big enough to be important? I.e. is it more different from what we would expect, under given circumstances (given by the model specification), than that which might occur only by chance? The low value of F and the high p-value (*0.581*) show that the enrichment of model `m1` by the quadratic term is very likely not an important improvement. It is not substantially larger than what can be expected purely by chance. Hence the improvement is not significant at the 5% level. It means that we cannot demonstrate that addition of the quadratic term is really beneficial (however, it does not mean that yet another curved type could not be useful). So we will prefer the smaller model (`m1`), which is built exclusively on a linear relationship. The "size" difference between models `m1` and `m2` amounts to 10 degrees of freedom. They correspond to 10 coefficients of the quadratic terms newly added to model `m2`.

Let's recall one of the main properties of statistical tests: **significant** and **non-significant** results provide very different kind of information. While a significant result is a *strong* result (the chance of a **Type I error** is under control by selecting a low significance level as the probability of an incorrect result), a non-significant result is *weak*. a non-significant result of a test can arise either because the null hypothesis is really true (e.g., when an explanatory variable has no effect on the study aspect), or because the null hypothesis is not correct but we are not able to reject it due to various other reasons (too few data, weak test, etc.). The chance of a **Type II error** is out of our direct control. It can be controlled at the phase of designing an experiment/study (indeed, this is done routinely for clinical trials in order to minimise chance of false negative findings). Without careful design control, it is possible that the chance of a Type II error is surprisingly high (it is more common to say that the power = 1-(Type II error chance) is large). From what has been said so far, it should be clear that it would be *fundamentally wrong* to suppose that a non-significant result can be regarded as a confirmation of the null hypothesis (e.g., that a study factor has no effect, or that a relationship is linear, etc.)! A popular is the saying: the absence of evidence is not evidence of absence! A non-significant result is necessary but, by far, not a sufficient condition to show absence of an effect. Non-significant results just say that we were not able to find an effect. Nothing more and nothing less. Therefore, we should never come to definite conclusions based on non-significant results.

In many situations, however, such as removal of terms in a model simplification, we need to make a decision whether to retain an explanatory variable in the model or to delete it. Non-significant terms are then removed, not included etc. We must be aware that such a use of a formal test is quite informal and therefore provisional. In particular, it generally does not guarantee (but also does not deny) good statistical or practical properties of the selected model. The selected model should be verified on independent data and/or by other means, e.g. in the model diagnostic style.

5.4.3 REMOVAL OF MODEL TERMS

A popular and, at the same time, simple strategy is the backward selection of a model, i.e. simplification of the originally larger model. Model simplification means removing components, i.e. individual variables and their interactions, from the initial model based on their non-significance (see the general warnings in the previous paragraph). We should remove individual components in accordance with the **marginality rule**: non-significant terms that form a part of a significant higher-order term (e.g., a non-significant main effect of a variable that is a part of a significant interaction) normally should not be removed from the model. When simplifying, we therefore remove the highest interaction first (of course only if it is non-significant), followed by lower interactions and eventually by non-significant main effects. Specifically, if a model with three factors (*A*, *B*, *C*) has a non-significant three-way interaction (*A:B:C*), we remove it and, subsequently, examine all three two-way interactions (*A:B*, *A:C*, and *B:C*). If they are all significant and if, for example, main effect *A* is not significant, according to the rule of marginality, we cannot remove *A* since it forms a component of interactions *A:B* and *A:C*.

A similar kind of rule is, usually invoked when simplifying models with a continuous explanatory variable. Quadratic terms are superior to linear terms and linear terms are superior to the intercept. According to this rule, we can remove neither the linear term, nor the intercept if the quadratic term is significant. The same applies to polynomial models of a higher order (higher powers are superior to lower). Nevertheless, as with many other things in statistics, the rule is not absolute and one's careful reasoning should be given much greater priority than fixed rules. We can omit a lower order term when it is justified theoretically (e.g. based on physical facts, biological theory, etc.) or by the results of previous empirical research. For example, when studying the relationship between mass of an organism and its body length, a theory predicts that if an organism is of a zero length, it cannot weight anything either. Therefore, the intercept should be set equal to 0, independently of what the removal rule says (but see also the Chapter 8.2). Many more specific restrictions can arise in a specific context. For instance the background theory can dictate that the given polynomial should have only even powers.

When simplifying, it is necessary to make sure that every step is really justifiable. Often one of the following two criteria are used for this purpose:

- A test that produces a p-value, based on which we either remove the given component from the model or not. For this purpose, the **`anova`** function is used, which always uses the F-test for **`lm`** models. On the other hand, no specific test type is pre-set for **`glm`** models. The F-test is commonly used for data from normal and Gamma distributions, while for other data the χ^2-test is used very often.
- The Akaike information criterion (AIC). It is an important example of a modern approach to model selection which takes into account both how the model fits data and the size of the model. Recall that as we build a model larger and larger, we are improving the fit with the analysed dataset, but that does not necessarily translate into any generalisability of the fit improvements (the phenomenon is often called overfit in the artificial intelligence and big data communities). To this end, AIC combines the value of the logarithm of the likelihood function (LogLik), a criterion of the model quality, and the number of parameters (p), as a penalisation for the model size. This is done according to the following formula:

$$\mathrm{AIC} = -2LogLik + 2p \tag{5-5}$$

AIC is one of the possible formalisations of the idea that a larger model will always fit the data at least as well as a smaller one (but typically better), but to accept it (and to deal with its complexity and worse interpretation), it has to be considerably better. We can find the value of AIC at the end of the summary output (**`summary`**) or by calling **`AIC`**. Selection of models based on AIC follows this simple rule: the lower the AIC value, the better the model. It is important to realise that, due to non-negligible AIC sampling variability, two models with small difference in AIC values (say < 2 units) are essentially equivalent. An often given recommendation says that models differing by more than 10 units are to be safely regarded as being very different (Burnham & Anderson 2002).

In this example, we will use the F-test to check for potential model simplifications. An ANOVA table of the model `m1` (page 65) shows that the effects of *SEX*, *start*, and all two-way and one

three-way interactions are non-significant (on the conventional level of 5%). Applying the marginality rule, first we remove the highest interaction, i.e. the three-way one. Removal can be done conveniently by means of function **update**. This function has two arguments. The first one is the name of the model to be altered (m1), the second one is an action to be taken. Now we need to remove the three-way interaction, thus we type: **~.-diet:sex:start**. The tilde with the dot stands for the original model. The minus sign denotes terms to be removed. These are listed afterwards. If we would like to add some terms, we would use the **+** sign. We save the altered model to the object named m2. We inspect the change in the model by calling **anova**.

```
> m2<-update(m1,~.-diet:sex:start)
> anova(m1,m2)
Analysis of Variance Table

Model 1: weight ~ diet * sex * start
Model 2: weight ~ diet + sex + start + diet:sex + diet:start + sex:start
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     65  8.0052
2     69  8.8364 -4   -0.8313 1.6874 0.1636
```

Removal of the three-way interaction did not cause any significant change (p-value = 0.164). Yes, things work as they should: the p-value is the same as that for the same interaction computed at the beginning of section 5.4.1. The non-significant interaction test means that the data do not seem to go strongly against the hypothesis that there is no three way-interaction. We will prefer model m2 over m1 because it is simpler and, at the same time, explains just a little bit less of overall data variability (so little that the F-test has not even detected it as an important piece). If we apply the **anova** function to model m2, we will see that the values of the test statistics have changed for all components that are present in the smaller model. When simplifying, we have to be careful to inspect the results of the test statistics after each step (one of the reasons is the fact that the data are not orthogonal – they include a different number of replications for males and females). It could easily happen that a component that is non-significant in the current model will become significant after removing another component, so we should not remove too hastily and do it carefully step by step (omitting at most one term at a time). For now, only the effect of *DIET* (its main effect) is significant.

```
> anova(m2)
Analysis of Variance Table

Response: weight
           Df Sum Sq Mean Sq  F value Pr(>F)
diet        4 58.484  14.621 114.1699 <2e-16 ***
sex         1  0.277   0.277   2.1622 0.1460
start       1  0.243   0.243   1.9013 0.1724
diet:sex    4  0.519   0.130   1.0128 0.4069
diet:start  4  0.137   0.034   0.2667 0.8984
sex:start   1  0.023   0.023   0.1832 0.6700
Residuals  69  8.836   0.128
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Let us continue with simplification. We will attempt to remove the two-way interactions and see if it is possible. We need to remove one after another and we have to start with the highest p-value (while keeping to the marginality rule), i.e. *DIET*:*start*. Let us see how the model results change upon removing each component.

```
> m3<-update(m2,~.-diet:start)
> anova(m3)
Analysis of Variance Table

Response: weight
          Df Sum Sq Mean Sq  F value Pr(>F)
diet       4 58.484  14.621 118.8423 <2e-16 ***
sex        1  0.277   0.277   2.2507 0.1379
start      1  0.243   0.243   1.9791 0.1637
diet:sex   4  0.519   0.130   1.0542 0.3854
sex:start  1  0.015   0.015   0.1248 0.7249
Residuals 73  8.981   0.123
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> m4<-update(m3,~.-sex:start)
> anova(m4)
Analysis of Variance Table

Response: weight
          Df Sum Sq Mean Sq  F value Pr(>F)
diet       4 58.484  14.621 120.2646 <2e-16 ***
sex        1  0.277   0.277   2.2776 0.1355
start      1  0.243   0.243   2.0028 0.1612
diet:sex   4  0.519   0.130   1.0669 0.3791
Residuals 74  8.997   0.122
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> m5<-update(m4,~.-diet:sex)
> anova(m5)
Analysis of Variance Table

Response: weight
          Df Sum Sq Mean Sq  F value Pr(>F)
diet       4 58.484  14.621 119.8537 <2e-16 ***
sex        1  0.277   0.277   2.2698 0.1360
start      1  0.243   0.243   1.9959 0.1617
Residuals 78  9.515   0.122
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

F- and p-values have changed, but not really dramatically. In the model `m5`, there are only three main effects left. The non-significant effect of the interaction *DIET*:*start* shows that we failed to find the difference among the slopes belonging to the levels of *DIET*. The non-significant effect of the interaction *SEX*:*start* shows that the trend of initial mass is similar for both sexes. Finally, the non-significant effect of the interaction *DIET*:*SEX* shows that the difference in the final mass between females and males among levels of *DIET* is similar.

Among all the main effects, only the effect of the *DIET* is significant. We will continue with the simplification procedure by removing variables *start* and *SEX*. Their non-significant ef-

fect suggests that there is no (linear) relationship between the final and initial mass and that the difference between the mass of males and females is negligible.

```
> m6<-update(m5,~.-start)
> anova(m6)
Analysis of Variance Table

Response: weight
          Df Sum Sq Mean Sq  F value Pr(>F)
diet       4 58.484  14.621 118.3616 <2e-16 ***
sex        1  0.277   0.277   2.2416 0.1383
Residuals 79  9.759   0.124
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> m7<-update(m6,~.-sex)
> anova(m7)
Analysis of Variance Table

Response: weight
          Df Sum Sq Mean Sq F value    Pr(>F)
diet       4 58.484  14.621  116.55 < 2.2e-16 ***
Residuals 80 10.036   0.125
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The only remaining variable in the model `m7`, the effect of which is highly significant, is *DIET* – exactly as we expected in the beginning (5-1). We have used the expansion of the initial model and its subsequent formal simplifications by means of the tests based on the study data. The final mass of the cockroaches subjected to five different diets are thus not similar. We cannot remove anything else from this model using the procedure described above. Nevertheless, it is not the final model yet, as you will see in a moment. The *DIET* variable has five levels (amounting to four degrees of freedom), and we want to know which of them are different.

5.4.4 COMPARISON OF LEVELS USING CONTRASTS

If we manage to get to a model that includes at least one categorical variable (with more than two levels), we want to know which coefficients (intercept, linear terms, etc.) or their groups are different. The result of the overall test for a factor with more than one degree of freedom (more than two levels) says: "Yes, there is at least one difference among the levels." But we do not know if there is only one significant difference, if each coefficient is different from every other coefficient, or if the truth is somewhere in-between. To arrive to a conclusion, we need to look "inside" the model somewhat more closely. For this purpose, we recommend to use one of the following methods:

- Apriori contrasts
- Posterior simplification

Both these approaches are based on the method of contrasts. The only difference is that for apriori contrasts, we decide in advance (when preparing an experiment or a study),

which levels will be compared. Such a decision can be based on the results of a previous experiment or on a theory to be tested. This method is often used in laboratory studies and almost exclusively in clinical trials, for example. When we use the posterior simplification procedure, we do not know apriori what we want to compare with what. Our decisions are based on "interesting facts" that only come to light during the analysis (just like in the previous sections where we added or removed model terms, based on more or less formalised selection procedures). This is, unfortunately, the most common but also the most criticised method used, especially in observational studies. Why is it criticised? Because we construct and propose to test comparisons based only on the facts that we *read* from the data, being led by the conclusions of previous tests, graphical displays and so on. This way, we actually conduct many more comparisons (which are often very much inter-dependent) in our head and exclude those that do not seem to be very interesting or "promising" in some (often very vague) sense. Their number is large and the dependence structure is very hard to determine and hence to take into account during the proper calibration of the testing procedures. The formally conducted tests represent a very select group (after a lot of "mental screening"). For example, we do not count comparisons that we think are non-significant. We calculate tests only for the comparisons that look interesting to us, based on what we know from the same data. Interpretation of the p-values for the tests actually conducted cannot be correct under these circumstances. The Type I error, i.e. false positive (falsely significant) rate can be substantially inflated, compared to the nominal error rate (for example, 5% level which we deliberately set for the test procedures). It is thus doubtful what the results of such tests are good for. Apriori contrasts (influenced only by our theories or experiences with *past* data and not by particular results of *current* observations) do not suffer from such a problem.

Let's look at the contrasts. You may ask: "What is a linear contrast?" Generally speaking, it is a linear combination of parameters (with known weights). For example, in the following quadratic model

$$y = \alpha + \beta x + \gamma x^2 + \varepsilon \tag{5-6}$$

a linear contrast can be the value of K defined as:

$$K = w_1\alpha + w_2\beta + w_3\gamma \ , \tag{5-7}$$

for some given values of w_1, w_2 and w_3. A particular contrast is specified by choosing coefficients (w). Between the theoretical contrast and its estimate there is a similar relationship as there is between the unknown value of a parameter and its estimate. An estimate of such a (theoretical) contrast is a random variable that is obtained when we replace the unknown parameters by their estimates (a, b, c), so that we get:

$$\hat{K} = w_1 a + w_2 b + w_3 c \ . \tag{5-8}$$

What is it good for? For one thing, when we want to estimate an expected value of the response variable at x_0, we find from (5-7) that this is, in fact, a particular case of a contrast with $w_1 = 1$, $w_2 = x$ and $w_3 = x^2$.

Contrasts are probably more often used in models that have a categorical variable in their systematic part. Let us consider an ANOVA model with a single factor *A*:

$$y_{ij} = A_j + \varepsilon_{ij}. \tag{5-9}$$

A_j represents the effect of the j^{th} level of this factor – it is the so-called **textbook parametrization**, which we have already encountered in Chapter 5.3. As we will see, the model and constants will be a little more difficult to write for other parametrizations, but it is not hard to get used to them. a contrast for (5-9) in the textbook parametrization will generally be

$$K = \sum_{j=1}^{J} w_j A_j . \tag{5-10}$$

One of the ways to achieve that is to assign the coefficients w_j within the frame of a single contrast to individual levels in accordance with the following rule: we will assign opposite signs to the levels we want to compare, the same signs to the levels that we want to combine into the same group, and zero to the levels we want to omit. If we want to compare only the averages of the levels of two groups, then all levels that are assigned the positive sign should be simply given coefficient 1/k, where *k* is the number of levels to which we assign positive signs. An analogous stipulation applies to the levels with negative signs. We thus get, as special cases, the mean values for all levels of factor *A* (in the textbook parametrization for the j^{th} level by selecting $w_j = 1$, and $w_k = 0$ for $k \neq j$). We can put together several contrasts but we have to think a little bit. For example, for a model with *DIET*, we may want to compare 2.1 times the average mass of the lipid-enriched diets with the average of the protein-enriched diets (because, for example, a published study claims that the relative conversions of nutrients into mass from both diet types should correspond to the above ratio). In the textbook parametrization, this corresponds to the contrast: $w_1 = 0$, $w_2 = w_3 = 2.1/2$, $w_4 = w_5 = -1/2$.

In models with categorical variables and other than textbook parametrization, the ANOVA model is (intentionally) over parametrised and written as: $y_{i,j} = \mu + A_j + \varepsilon_{i,j}$. The contrasts coefficient for the intercept (w_0) is then typically set to zero. In that case, it seems that we need *J* coefficients for the contrasts specification. So, for example, with $DIET_j$ denoting the effect of j^{th} diet level, we have:

$$K = \sum_{j=1}^{J} w_j \, DIET_j , \tag{5-11}$$

but for fundamental statistical reasons related to identifiability (estimability) of the various model coefficients and their linear combinations, we have to impose some other restrictions. In the case of a single factor model, it is just one restriction. In the so-called treatment parametrization, we impose $DIET_1 = 0$. Generally, in the treatment parametrization, we set the effect of the reference level to zero (and R takes as the reference level the one with the lowest code ordering). In the treatment parametrization $DIET_j$ for $j > 1$ will then correspond to the difference between expected values of j^{th} level and the reference level.

Another parametrization (called sum parametrization in R) imposes another restriction on the *w*'s, namely:

$$\sum_{j=1}^{J} w_j = 0\,. \tag{5-12}$$

This will lead to a different interpretation of the over parametrised model coefficients and a different way of writing the desired contrasts (for the same contrasts, we will have different *w*'s than in the treatment or textbook parametrizations).

5.4.5 CONTRASTS AND THE MODEL PARAMETRIZATION

Six types of contrasts are available in R. Five of them are pre-defined and one is user-defined. The pre-defined includes **textbook**, **treatment**, **Helmert**, **sum**, and **polynomial** contrasts. As you saw before, the type of contrast defines the model parametrization and the interpretation of the estimated coefficients that we obtain by applying the **`summary`** function. Pre-defined contrast matrices allow exactly as many contrasts as there are degrees of freedom in the model (in order to explain all variability in the model with given explanatory variables).

The most frequently used type is **treatment** contrasts, which is a default option in R. Other (e.g. Helmert or sum contrast) parametrizations can be specified by the user globally (in the **`contrasts`** argument of the **`options`** function). The choice of the treatment parametrization is related to the ease of the interpretation of its parameters. The parameters correspond to the contrasts comparing the reference level of a factor with all other levels (the comparison of a reference level with a reference level is obviously zero and hence omitted). If we have a factor A with levels A_1, A_2, and A_3, then treatment parametrization will have parameters which correspond to two following contrasts: A_2 *versus* A_1 and A_3 *versus* A_1. Altogether there would be two contrasts (because there are two degrees of freedom for the factor A, which has 3 levels). This type of parameters/contrasts is very useful in cases when we want to compare, for example, a control with all other treatment levels, given that the control is specified as a reference level. This is exactly what we need in our example.

As we mentioned above, the final model `m7` has the form that we specified in the beginning (5-1). Rewriting or reparametrising it using treatment-contrasts-related model coefficients, we get

$$weight_{ij} = \alpha + DIET_j + \varepsilon_{ij}, \tag{5-13}$$

where $DIET_{\text{ctrl}} = 0$ and $\varepsilon_{ij} \sim N(0,\sigma^2)$, independent among measurements.

In this parameterisation, estimated parameters have a *completely* different interpretation than the parameters of the textbook parametrised model (5-1) had. But do not worry, it is not going to be too difficult. α is the mean mass for the level of the diet ("ctrl"). It is taken as the reference automatically by R since its level code is the first according to the lexicographic ordering. The effect of j^{th} level (i.e. $DIET_j$) stands for a difference in weight means of a j^{th} level and "ctrl". Recall that the model parameterisation has a practical implication for parameter interpretation. The same data and the same model will produce different estimates of coefficients when using different contrasts! The estimates can be transformed one to another,

however. Reparametrization changes the interpretation of the coefficients, but does not alter the model itself. In particular, the parametrization does not change the fit of the model (it does not change how well or poorly the model fits the data, it does not change model predictions, residuals, residual mean square, etc.).

In model `m7`, the factor *DIET* has a significant effect. This is a kind of pre-requisite for "looking inside it", i.e. for trying to explore which levels differ from each other. Exploration of a model with a non-significant factor effect is a mistake! Besides incurring deeper statistical problems, it contradicts common sense: if the overall test says "there is no difference" then it is not reasonable to look inside for some difference.

Now we will explore differences among factor levels using treatment contrasts. To do so, just type **`summary`** and the model name in parentheses.

```
> summary(m7)

Call:
lm(formula = weight ~ diet)

Residuals:
     Min       1Q   Median       3Q      Max
-0.66471 -0.18294 -0.05294  0.16706  0.91706

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.9547     0.0859  11.114  < 2e-16 ***
dietlipid1     0.7282     0.1215   5.994 5.59e-08 ***
dietlipid2     0.6682     0.1215   5.501 4.41e-07 ***
dietprotein1   2.1382     0.1215  17.601  < 2e-16 ***
dietprotein2   2.0100     0.1215  16.545  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3542 on 80 degrees of freedom
Multiple R-Squared: 0.8535,     Adjusted R-squared: 0.8462
F-statistic: 116.6 on 4 and 80 DF,  p-value: < 2.2e-16
```

The summary table looks complicated. It has more information than the ANOVA table we saw previously. The first two lines include the model formula (Call), followed by summary statistics of the model residuals. If residuals have a normal distribution then Min, Max, 1Q, and 3Q values will be symmetric around the median, which should be very close to zero. Apriori, one expects to see more symmetry in the 1st and the 3rd quartiles (1Q, 3Q) around median, than in the extremes (Min, Max), as the latter two characteristics are inherently much more variable. We are particularly interested in the Coefficients. This output section contains the **table of coefficients**. Here we find estimates of coefficients – interpretable as contrasts of effects (Estimate), standard errors of estimates (Std. Error), t-statistics (t value) and p-values for the test of the null hypothesis that the true value of a certain coefficient (or parameter or contrasts) is zero (Pr(|>|)). Asterisks at the end of rows highlight those tests which turned out to be significant at different nominal levels.

How to interpret the estimates? The factor *DIET* has five levels, so we may expect to get five coefficients in the table, each corresponding to a *weight* mean for one level. Indeed, the means are there but we have to "put them together". The first row, Intercept, corresponds to the expected value of the reference level ("ctrl"). The value *0.9547* is an estimate of the expected value of the *weight* for "ctrl". In the rows below, all coefficients have a name composed of the factor and the level names. These values are not means but differences between the mean of a given level and the "ctrl" *weight* mean. To obtain the *weight* mean for "lipid1", we have to add 0.9547 + 0.7282 = *1.6829*. Similarly, we find means for all the other levels. For "protein1", we get it by adding the fourth row to the first one: 2.1382 + 0.9547 = *3.0857*. This simple rule applies only when the model has just one factor. Remember that this interpretation of the coefficients and the way we put together coefficients is only valid for this type of contrasts! In other contrasts parametrizations we have to use a different method – see below.

You may wonder why it is so complicated. Why is there only a mean for the first level? Well, this is simply because it is from the differences that we can quickly compute which levels differ. If the table included means of all levels (this is actually the case with textbook parametrization) then a t statistic in any given row would test the null hypothesis that a *weight* mean for the corresponding factor level is zero. These kind of tests are almost always totally uninteresting. To compare two means, one needs to know the difference and the standard error of the difference. These can be found easily in the output of the treatment parametrised model, but not in the textbook parametrised model. The standard error of the difference can be found in the column named Std. Error. The t-value is obtained by dividing the coefficient value by its standard error. Thus for the last row, the t value is obtained as follows: 2.01/0.1215 = *16.545*. The p-value shown in the last column is then computed for each t value (using residual standard deviation and t distribution with residual degrees of freedom).

Below the coefficient table, there is a three line summary, which is very similar to that appearing at the end of the output of an ANOVA table. In addition, there are two coefficients of determination. These are important for regression models, therefore we will deal with them later (Chapter 8.1).

In some cases, the treatment contrasts do not produce information we really want. For example, we might want to compare other levels or combinations of levels. Our aim in this study was to find the effect of four types of diets on the mass increase. Specifically, we wanted to:

- Compare control with enriched diets
- Find difference among two lipid and two protein diets
- Compare two lipid diets
- Compare two protein diets

Only the first aim can be met with the treatment contrasts. To meet the other ones, we need to construct **user-defined contrasts**. The factor *DIET* has five levels, so we can construct at maximum 5 – 1 = *4* orthogonal contrasts (more contrasts would not be even allowed by the function). We construct them to reflect the four aims. In the first one we compare "ctrl"

with all other levels. As the sum of contrast coefficients in a vector must be *0* (5-12), we will assign the "ctrl" with *1* and all other levels will be assigned with *–1/4*. The "ctrl" level will not be used in any other contrast vector, thus we will assign it with *0*. In the second contrast, we will compare two lipid levels (combined) with two protein levels (combined). Therefore, the levels "lipid1" and "lipid2" will get *–1/2* each, whereas the levels "protein1" and "protein2" will be assigned with *1/2* each. The sum of the contrast coefficients is again zero. In the third contrast we will compare "protein1" against "protein2", thus the former will get *1/2*, and the latter will get *–1/2*. All other levels will be assigned with zeros. In the fourth contrast we will compare "lipid1" with "lipid2" similarly as in case of the two protein levels. The sum of the two latter contrasts is zero too. Let's construct the transformation matrix needed for the definition of contrasts (to be used by the R function computing the model). At first, we need to find the order of levels in the factor.

```
> levels(diet)
[1] "ctrl"     "lipid1"   "lipid2"   "protein1" "protein2"
```

We will construct the matrix using **cbind**, i.e. we will bind the vectors of particular contrasts together. The vectors are created by means of the **c** command. The entire command is longer than a command line, thus following a comma press ENTER to move to a new line. Here we continue typing commands following the plus sign automatically generated by the software.

```
> contrasts(diet)<-cbind(c(1,-1/4,-1/4,-1/4,-1/4),c(0,-1/2,-1/2,1/2,1/2),
+ c(0,0,0,1/2,-1/2),c(0,-1/2,1/2,0,0))
> contrasts(diet)
          [,1] [,2] [,3] [,4]
ctrl      1.00  0.0  0.0  0.0
lipid1   -0.25 -0.5  0.0 -0.5
lipid2   -0.25 -0.5  0.0  0.5
protein1 -0.25  0.5  0.5  0.0
protein2 -0.25  0.5 -0.5  0.0
```

Now, the matrix is ready. Before actually using it, let us check whether the contrasts are orthogonal: select any two columns, and sum the products of corresponding vector elements. If you calculated everything correctly, the sums will always be zero. It is important to note that the following analysis would be legal even if the contrasts were not orthogonal (if they were linearly independent and if there were less contrasts than the degrees of freedom), but their interpretation would be more complicated (because then different model coefficients, corresponding to different contrasts, would be correlated).

So let's fit a new model nested in the **summary** command.

```
> summary(lm(weight~diet))

Call:
lm(formula = weight ~ diet)

Residuals:
     Min       1Q   Median       3Q      Max
-0.66471 -0.18294 -0.05294  0.16706  0.91706
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.06365    0.03842  53.718   <2e-16 ***
diet1       -1.10894    0.07683 -14.433   <2e-16 ***
diet2        1.37588    0.08590  16.017   <2e-16 ***
diet3        0.12824    0.12148   1.056    0.294
diet4       -0.06000    0.12148  -0.494    0.623
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3542 on 80 degrees of freedom
Multiple R-Squared: 0.8535,     Adjusted R-squared: 0.8462
F-statistic: 116.6 on 4 and 80 DF,  p-value: < 2.2e-16
```

The overall statistics, shown at the bottom line, did not change. This is a good check to confirm that we did proceed correctly. If they change, it would mean that we missed something and the new model is not just a reparametrization of the previous, but that it is somehow different (fitting the data differently). The content of the table of coefficients is, however, very different from the previous model. This is just because we used different contrasts. This model is just a re-parametrised version of the former model (it is just a different view on the same model). In the first row, "Intercept" represents the **grand mean** (*2.06365*) – i.e. the mean computed from all measurements disregarding the levels of *DIET*. In the second through to the fifth row, are results of the four predefined contrasts in the same order as they were included in the matrix. We can see that the last two contrasts are not significant. This means that the difference between two lipid levels or two protein levels is not significant. Only the first two contrasts are significant. So "ctrl" is significantly different from all other levels and the two lipid levels (combined) are significantly different from the two protein levels (combined).

Other types of user-defined contrasts, including non-orthogonal ones, can also be specified via the **`contrasts`** function. Alternatively, it might be more convenient to use the **`glht`** function from the multcomp package.

Helmert and sum contrasts offer another two views on the same model. In agreement with the degrees of freedom for the factor *DIET*, these matrices also include four contrasts. In contrast to the treatment contrasts **Helmert contrasts** (**`contr.helmert`**) are orthogonal: they compare a certain level against an average of levels with a lower subscript, i.e. A_2 against A_1, then A_3 against an average of $A_1 + A_2$, etc. We will call Helmert contrasts similarly to the way we called user-defined contrasts. Be aware that the contrasts will take effect only after fitting a new model *after* using **`contrasts`**:

```
> contrasts(diet)<-'contr.helmert'
> summary(lm(weight~diet))

Call:
lm(formula = weight ~ diet)

Residuals:
     Min       1Q   Median       3Q      Max
-0.66471 -0.18294 -0.05294  0.16706  0.91706
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.06365    0.03842  53.718  < 2e-16 ***
diet1        0.36412    0.06074   5.994 5.59e-08 ***
diet2        0.10137    0.03507   2.891  0.00495 **
diet3        0.41819    0.02480  16.864  < 2e-16 ***
diet4        0.22526    0.01921  11.727  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3542 on 80 degrees of freedom
Multiple R-Squared: 0.8535,     Adjusted R-squared: 0.8462
F-statistic: 116.6 on 4 and 80 DF,  p-value: < 2.2e-16
```

There are again five coefficients, but having different values than before. The first row is again the grand mean. The first contrast (diet1) tests a difference between "ctrl" and "lipid1". The second contrast (diet2) compares average of "ctrl" + "lipid1" against "lipid2". The third contrast (diet3) tests the difference between the average of "ctrl" + "lipid1" + "lipid2" against "protein1". The last contrast (diet4) compares the difference between the average of "ctrl" + "lipid1" + "lipid2" + "protein1" against "protein2". Tests of all contrasts (i.e. tests of null hypotheses that estimates are zero) are significant. The estimates are not pure differences between particular contrasts but differences normalised with the sum of squares of weights used for the contrast.

You may agree that this type of contrasts was not very useful for our example. It could be useful if the levels of the factor were ordinal, arranged e.g. from the lowest to largest, from worst to the best, etc. In such case, it could help us to identify a first instance when a level differs substantially from the average of the previous levels.

Sum contrasts (**`contr.sum`**) compare a certain level with the grand mean, and the comparison of the alphabetically last level is omitted (due to a restriction of the number of orthogonal contrasts): so A_1 is compared with the average of $A_1 + A_2 + A_3$ and A_2 is compared with average of $A_1 + A_2 + A_3$. The comparison of A_3 with the average is omitted in this R parametrization. We will call these contrasts similarly to calling Helmert contrasts.

```
> contrasts(diet)<-'contr.sum'
> summary(lm(weight~diet))

Call:
lm(formula = weight ~ diet)

Residuals:
     Min       1Q   Median       3Q      Max
-0.66471 -0.18294 -0.05294  0.16706  0.91706

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.06365    0.03842  53.718  < 2e-16 ***
diet1       -1.10894    0.07683 -14.433  < 2e-16 ***
diet2       -0.38071    0.07683  -4.955 3.96e-06 ***
diet3       -0.44071    0.07683  -5.736 1.66e-07 ***
diet4        1.02929    0.07683  13.396  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.3542 on 80 degrees of freedom
Multiple R-Squared: 0.8535,     Adjusted R-squared: 0.8462
F-statistic: 116.6 on 4 and 80 DF,  p-value: < 2.2e-16
```

In the table of coefficients, the first row shows the grand mean. The first contrast (diet1) compares "ctrl" against the grand mean. The second compares "lipid1", the third compares "lipid2" and the fourth compares "protein1", each against the grand mean. All contrasts are significant, which means that all levels (except for the last one "protein2") are significantly different from the grand mean.

5.4.6 POSTERIOR SIMPLIFICATION

Let's go back to the table of coefficients of the model `m7` fitted in the treatment parametrization (page 78). We will now simplify the model using a kind of "posterior lumping". By comparing the values of coefficients, we can see that the ones for "protein1" and "protein2" are similar – both are about 2. We expect that the difference between these two levels is negligible. To test our hypothesis we lump together the two levels and we test the lumping by means of the F-test. At first, we create a copy of the factor with the name `diet1`. The new factor *DIET1* has the same levels as *DIET*, as we can see by calling **`levels`**.

```
> diet1<-diet
> levels(diet1)
[1] "ctrl"     "lipid1"   "lipid2"   "protein1" "protein2"
```

We want to combine the last and the next to last, i.e. the fourth and fifth levels. To do so, we will use the **`levels`** command with the argument of square brackets, inside of which we specify the position of the combined levels. We will name the new level "prot". Subsequently, we fit the new model with the modified *DIET1*, after which we will test the combination by comparing the previous (`m7`) and the new (`m8`) models using the **`anova`** function. Prior to fitting the model, do not forget to "return" to treatment parametrization using the **`contrasts`** command.

```
> levels(diet1)[4:5]<-"prot"
> levels(diet1)
[1] "ctrl"   "lipid1" "lipid2" "prot"
> contrasts(diet1)<-'contr.treatment'
> m8<-lm(weight~diet1)
> anova(m7,m8)
Analysis of Variance Table

Model 1: weight ~ diet
Model 2: weight ~ diet1
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     80 10.0357
2     81 10.1755 -1   -0.1398 1.1142 0.2943
```

The comparison did not produce a significant change (p-value = 0.294) in the residual sum of squares (RSS) and we thus accept it. This way, we "saved" one degree of freedom – the new model now has only four parameters. It is also clear from the table of coefficients of model `m7` that the coefficients for the lipid-enriched diets, i.e. "lipid1" and "lipid2", are similar. We

will try to combine them in a similar manner. This time, we create a copy of the *DIET1* factor, since otherwise we could not come back to the original model if there was a significant difference detected in the test to come.

```
> diet2<-diet1
> levels(diet2)[2:3]<-"lipid"
> m9<-lm(weight~diet2)
> anova(m8,m9)
Analysis of Variance Table

Model 1: weight ~ diet1
Model 2: weight ~ diet2
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     81 10.1755
2     82 10.2061 -1   -0.0306 0.2436  0.623
```

It is not significant. We will thus focus on the table of coefficients of model `m9` with the objective to find out if we can simplify the model even further.

```
> summary(m9)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.95471    0.08557  11.158  < 2e-16 ***
diet2lipid   0.69824    0.10480   6.663 2.87e-09 ***
diet2prot    2.07412    0.10480  19.792  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3528 on 82 degrees of freedom
Multiple R-Squared: 0.851,      Adjusted R-squared: 0.8474
F-statistic: 234.3 on 2 and 82 DF,  p-value: < 2.2e-16
```

The smallest difference (*0.698*) is between the first and second coefficient, i.e. between “ctrl” and “lipid”. Nevertheless, the test statistic in the second row shows that this difference is significant. We cannot find the test of the difference between “lipid” and “protein” in the table. We have to determine it, once again, by combining both levels. We thus create a copy of the *DIET2* variable and combine both levels under the name “other”.

```
> diet3<-diet2
> levels(diet3)[2:3]<-"other"
> m10<-lm(weight~diet3)
> anova(m9,m10)
Analysis of Variance Table

Model 1: weight ~ diet2
Model 2: weight ~ diet3
  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1     82  10.206
2     83  42.388 -1   -32.182 258.56 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

This lumping of factor levels together led to a very significant effect – the model became substantially worse! Therefore, we return to the model `m9` which has three coefficients in the treatment parametrization: mean for "ctrl" and two differences, one for "lipid" and one for "prot". Both differences are significant. Please note that the F-test value at the bottom of the output increased markedly (from *116.6* in `m7` to *234.3* in `m9`). This is because we saved two degrees of freedom and the quality of the fit still did not decrease.

To what extent has the simplifying step-by-step strategy been reasonable? Remember what we said above, i.e. that such a strategy is somewhat myopic in nature: it does not compare (by far) all models and, therefore, it cannot guarantee that the final model will not be fundamentally worse than the initial model after we gradually omitted non-significant terms. Let us then compare simplified model `m9` with initial model `m1`.

```
> anova(m9,m1)
Analysis of Variance Table

Model 1: weight ~ diet2
Model 2: weight ~ diet * sex * start
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     82 10.2061
2     65  8.0052 17    2.2009 1.0512 0.4188
```

Indeed, model `m9` is not significantly worse than the initial model. The residual sum of squares has increased only by 2 units as a result of the modification, while the size of the model (as expressed by the degrees of freedom) has decreased by 17.

5.4.7 DIAGNOSIS OF THE FINAL MODEL

We have already mentioned several times that a model we arrived at, based on previous the-oretical knowledge of the studied process, analysis of empiric data or a combination of both, does not have to "fit" the given data set reasonably well. Consequently, the **inference** that is based on it does not have to be correct. That is why we should attempt to verify it as thoroughly as possible. Since there are indeed many ways in which a model can be bad, there are also many techniques with which to examine these possibilities. It is utterly impossible to verify all aspects of a model, but we should examine as many of the important aspects as possible. This is true despite the fact that we make more work for ourselves in this way – detected problems result in the necessity to develop a new model, thus making us to continue the analysis. Nevertheless, we can justify and defend a model only after it has been verified. Otherwise, the results would represent a mere summary (which might even be unnecessarily complicated), and might even provide completely false conclusions (both falsely positive as well as falsely negative).

We will explore the final model `m9` using several diagnostics tools which are quite standard for linear models and which are easily available in R. This does not mean that diagnosis of your model should be limited to these tools only. It rather shows some basic kinds of tools that we can start with. We can obtain basic diagnostic plots using the **`plot`** command, one after another. To obtain all four plots, which we discussed in Chapter 4.6, we will use the

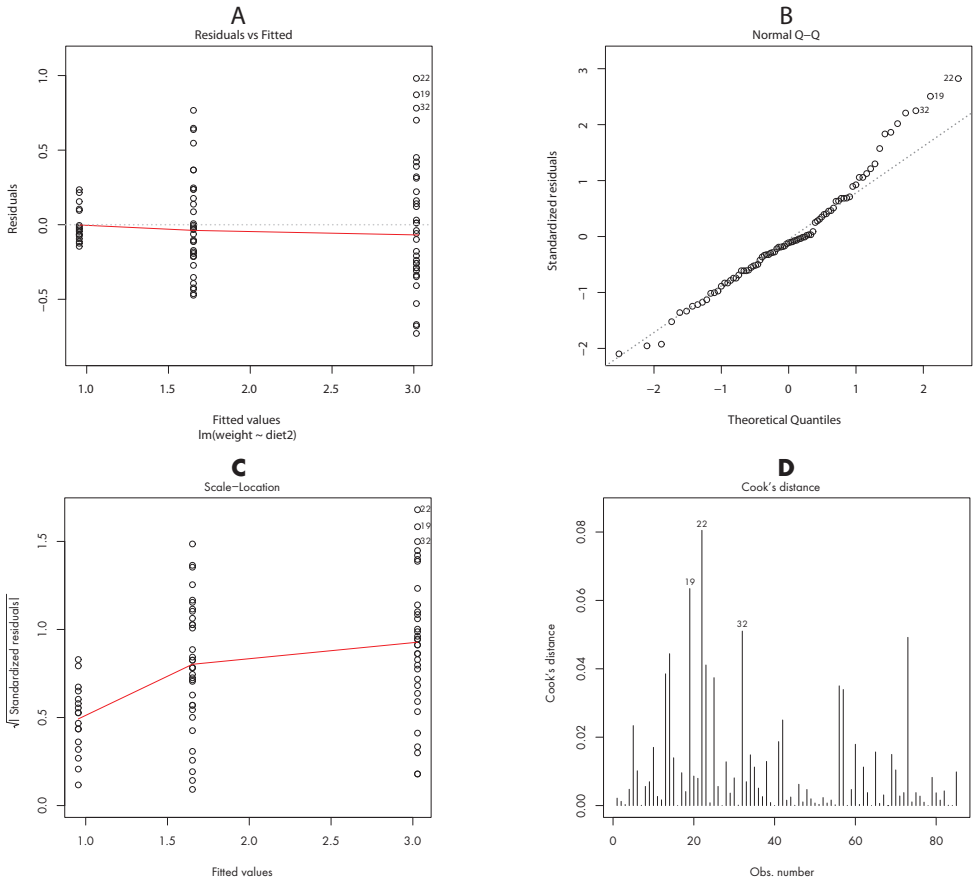


Fig. 5-3 A. Relationship between residuals and fitted values. **B**. Q-Q normal plot of standardised residuals. **C**. Relationship between the square roots of absolute values of standardised residuals and fitted values. **D**. Plot of Cook's distance for every measurement.

`which` argument to select the first four plots. We will place the four of them into four panels of the graphical window:

```
> par(mfrow=c(2,2))
> plot(m9,which=1:4)
```

In the first plot (Fig. 5-3A), the residuals are arranged in three columns. This is because each column represents one level of *DIET2* from model `m9`. The red line connecting mean values of the residuals for different levels of the *DIET2* stays very close to abscissa. This does not indicate any fundamental problems related to the model underestimation or overestimation of the data. The spread of residuals is getting bigger as the fitted value increases, indicating that residual variance is not homogenous. It increases with the mean value. This impression is also confirmed by the red line in the third plot – it is increasing (Fig. 5-3C). For those of

you who prefer a test to "just viewing a picture", the variance homogeneity can be tested by the **Bartlett test**. The argument (**`formula`**) of this test will be the model formula from the latest model:

```
> bartlett.test(weight~diet2)

        Bartlett test of homogeneity of variances

data:  weight by diet2
Bartlett's K-squared = 24.2178, df = 2, p-value = 5.51e-06
```

The nominally significant result confirms heterogeneous variance. The Bartlett test does not generally say the same that the plot does. On one hand, it is a formal test with many advantages, on the other hand, it is known to be often too sensitive to various nuisance influences, e.g. deviations from normality. The significant result can then suggest a non-normal distribution of the residuals rather than their heterogeneous variance.

We can see from the second plot (Fig. 5-3B) that distribution of the residuals is skewed to the right. The furthest residuals are on the top right. The potential outliers are labelled by the number under which they appear in the original data frame. We can verify deviation from the normality by a test, for example, the **Shapiro-Wilk normality test**. The argument (**`x`**) of this test is the residuals of model `m9`.

```
> shapiro.test(resid(m9))

        Shapiro-Wilk normality test

data:  resid(m9)
W = 0.9685, p-value = 0.0356
```

The significant result confirms departure from normality. Similarly to the Bartlett test, the normality test is not always better than a "mere" graphic representation. This is not only because a non-significant result does not guarantee normality, but also because we cannot know how the normality is violated (though this is visible, to some extent, in the plot).

Finally, the last plot (Fig. 5-3D) shows the three residuals with the highest Cook's distance values (19, 22 and 32). Nevertheless, the Cook's distance of all of them is very low, less than 0.1, which means that the observations are not extremely influential.

Based on the previous diagnostic explorations, we conclude that model `m9` would not pass the evaluation. The non-constant variance and quite significant departure from normality represent serious violations of assumptions. The non-constant variance especially influences the standard errors of the estimates. This, together with the departures from normality can seriously invalidate the model inferences based on model `m9`. We should, therefore, try to modify the model using a logarithmic transformation, or suitable weighting, or by using a distribution that allows the variance to increase with the mean value, and then restart the analysis from the very beginning. We will show you how to do that in Chapter 9.5.

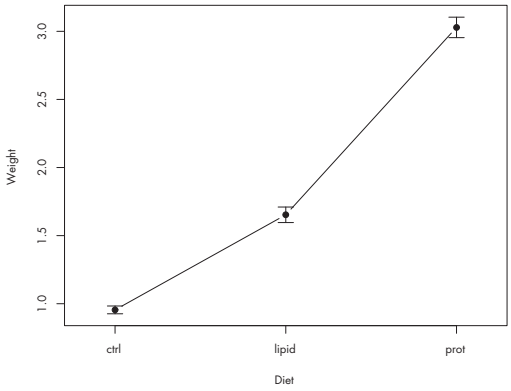


Fig. 5-4 Line plot of mean body mass (Weight) of cockroaches on three diet types (Diet). Whiskers correspond to ± 1 SE.

We will now plot the expected value estimates obtained from empirical means in a line plot with whiskers. The whiskers will show standard errors of the means (SE) as the correct precision assessments of the means. We will do all of this just by calling the **`lineplot.CI`** function from the sciplot package (Fig. 5-4). We will not show any model fits or predictions on the plot yet, since we detected model `m9` as being inappropriate.

```
> par(mfrow=c(1,1))
> library(sciplot)
> lineplot.CI(diet2,weight,ylab="Weight",xlab="Diet")
```

5.5 Conclusion

Since we have not found a suitable model, we should not deduce any conclusion yet! Instead, we will come back to finish this analysis in Chapter 9.5.

6 SYSTEMATIC PART

The type of an analysis depends (among other thigs) on the specification of the systematic part. In linear and generalized linear models, it will come as a specification of the linear predictor. It thus depends on the type of explanatory variable(s) (continuous and/or categorical). If the explanatory variable is clearly specified as continuous or as a factor, the selection of the analysis type is easy. However, the same variable can be categorical under certain circumstances, and continuous otherwise. Let us consider an experiment, by means of which we want to study the relationship of two explanatory variables, temperature and pressure, both of them at two levels (for example 0 and 10 °C, and 1,000 and 1,010 hPa). Should we consider them to be categorical or continuous? The answer is not that clear as it may look at first. Though both variables are inherently continuous, they can also be perceived as discrete or even categorical variables *in our experiment*. Each of them has two levels, low and high. If we specify and interpret them categorically, i.e. as factors, the resulting model will not allow us to directly **interpolate** a value of y for the values of the explanatory variables inside of the study range of levels, for example, for 5 °C and 1,005 hPa. Nevertheless, a knowledgeable reader may have realised that predictions for these values can be obtained by estimating suitably selected linear contrasts (as we discussed in Chapter 5.4.4). Interpolation can be achieved much more easily (almost automatically), if we declare them to be continuous variables. Such a solution allows even for **extrapolation**, that is, for predicting beyond the extent of the explanatory variables used in the actual experiment or observation study. This can represent an extremely *desirable* characteristic. It depends on the situation as to whether it is also *useful*. Generally speaking, we should use interpolation, and all the more extrapolation, only if we really trust our model. Since this trust cannot be verified on the analysed data, it must be based on some external information. For example, on a theoretical justification of the model, previous extensive experiences, or on model checks based on carefully selected additional data, etc. In any case, we can see that it is worth thinking about the model formula a little bit before diving into the actual data analysis, considering various practical consequences.

Depending on the presence of continuous and categorical explanatory variables in the linear predictor, we can define three basic types of analyses:

- With only continuous explanatory variables in the model . . . . . **Regression**
- With only categorical explanatory variables in the model . . . . . **ANOVA** (analysis of variance) or **ANODEV** (analysis of deviance)
- With at least one continuous and one categorical explanatory variable in the model . . . . . **ANCOVA** (analysis of covariance) or **ANCODEV** (analysis of co-deviance).

At present, this division is not as important as it was in the past. We use it here to help the reader to realise the differences in interpretation. For models with normal distribution of the response variable, all these types are special cases of the **general linear model**. Linear regression, ANOVA (Analysis of Variance) and ANCOVA (Analysis of Covariance) were introduced for models with additive normally distributed errors. Analogously, these types can be used in the context of GLM (where the word variance is replaced by deviance). As GLM is a generalized form of the general linear model, these types are often not distinguished at all.

The linear predictor may include a variety of individual terms. For example, it can contain the intercept, linear terms for different explanatory variables (with slope coefficients to be estimated), various powers as parts of a higher order polynomial terms (e.g. quadratic, cubic), main effects, two-way, three-way, and higher order interactions, etc. **Intercept** (α, estimated as a) is a coefficient that has a different interpretation according to the type of analysis (see below) and the model parametrization (see 5.4.5). This applies to other model coefficients as well (linear, quadratic, etc.). **Linear term** is a product of type βx, where x is an explanatory variable (whose values are known). β is a coefficient of a linear trend to be estimated from the data (its estimate will be denoted by b). Generally, there might be several linear terms in the model (e.g., a linear term for x_1 and a linear term for x_2). **Quadratic term** is a product of type γx^2, where γ is a coefficient of the quadratic trend (estimated from data as c). Then there are **cubic** and higher **polynomial terms** (e.g., δx^3). **Main effect** is the complete effect of a certain categorical variable (factor) having one or more degrees of freedom (i.e. two or more levels). An **interaction** is a combined (simultaneous) effect of several explanatory variables, continuous or categorical (see Chapter 3.5.5).

We will discuss meaning of parameters in various models in the following paragraphs somewhat more deeply. We will stress the treatment parametrization viewpoint since it is the parametrization that we use most frequently throughout this book. Keep in mind that using a different parametrization would result in parameters with a completely different meaning! Also note that the interpretation of individual parameters in the GLM depends heavily on the link (Chapter 4.3 and specific cases in Chapter 8–12)!

6.1 Regression

According to the number of explanatory variables, we often distinguish a **simple regression** if there is only one continuous explanatory variable and a **multiple regression** if there are more explanatory variables. Clearly, the model of the simple regression is nothing more than a special case of the multiple regression. Thus, it is sufficient to define the more general model. For k explanatory variables, we have a linear predictor of the following form

$$\alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k , \tag{6-1}$$

where α is a common intercept and β_i are coefficients of the linear terms (or slopes) of x_1, x_2, ..., x_k. The model is more general than it appears at first sight. All *x's* may or may not represent different variables. For example, different powers of x may appear as formally different x_i's (e.g. $x_1 = x, x_2 = x^2$), so that (6-1) may be a polynomial regression. Or, it may include terms

like $x_5 = x_3x_4$, i.e. a multiplication of two variables as an interaction between two continuous explanatory variables. The linear predictor gives an expected value for a given combination of explanatory variables (or a link-transformed expected value for a general GLM).

If we want to obtain the final model through the "simplification" approach (i.e. by removing non-significant terms using a variant of the backward selection procedure), we should start with a model that is sufficiently large. For safety, the initial model should be, much richer than what we guess the final model to be. How much richer, really depends on what the data (or the study design) allow us. For example, we may want to include all the main effects (linear terms), all their interactions (e.g. the products of different variables like x_1x_2, which are formally just a part of interactions), and quadratic terms in the model. Nevertheless, depending on the circumstances (for example, a small data set or an inconvenient structure in relation to explanatory variables), we often have to settle with something more modest. On the other hand, for a well-designed and extensive study, we can consider a model that is even larger, for example, with cubic terms and products of the $x_1x_2x_3$ type. However, it is usually not very smart to work with very high order polynomials (and/or very high order interactions). Very high order model terms are often hard to interpret, and it can lead to instabilities in estimates (due to collinearity, their standard errors can become tremendously large and absolutely intolerable).

Now, let us focus on possible model types using a generic example. If we assume that we have only two explanatory variables (x_1 and x_2), we can try a linear predictor that will include six parameters:

$$\alpha + \beta_1 x_1 + \beta_2 x_2 + \gamma_1 x_1^2 + \gamma_2 x_2^2 + \delta x_1 x_2 . \tag{6-2}$$

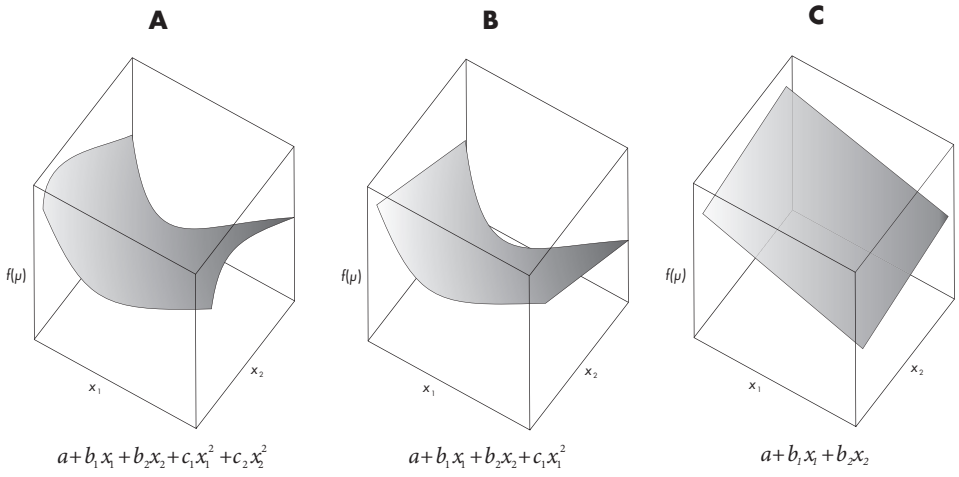


Fig. 6-1 Examples of linear predictor in the model of multiple regression with two explanatory variables x_1 and x_2.

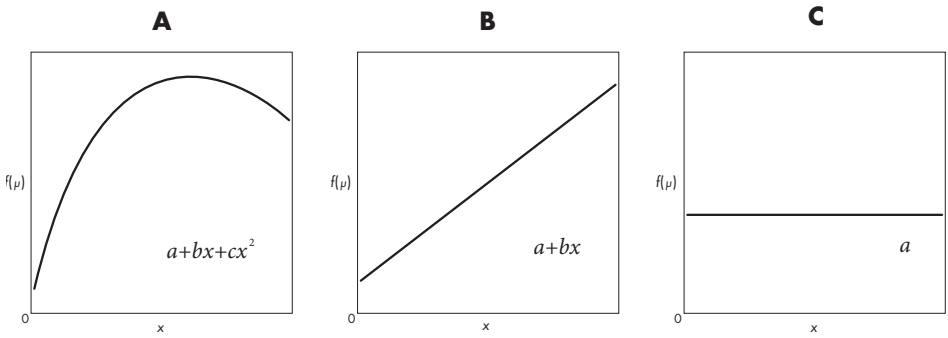


Fig. 6-2 Examples of linear predictor in the model of simple regression.

By simplifying this formula according to what the data (and formalised tests) tell us, we might, for example, arrive at the following:

- A curved surface (Fig. 6-1A) with five estimated parameters a, b_1, b_2, c_1, c_2, where both c_1, and c_2 are significantly different from zero
- A curved surface (Fig. 6-1B) with four estimated parameters a, b_1, b_2, c_1, where c_1 is significantly different from zero
- A straight surface (Fig. 6-1C) with three estimated parameters a, b_1, b_2, where b_1 and b_2 are significantly different from zero

If one of the explanatory variables (say x_2) has a non-significant effect, we can delete it and obtain a (possibly) curved model with a single explanatory variable x, so that the predictor (or systematic model part) has three parameters:

$$\alpha + \beta x + \gamma x^2. \tag{6-3}$$

α is the intercept, β is the slope and γ is the coefficient of the quadratic term (their estimates are a, b, c). c is significantly different from zero (Fig. 6-2A). If it is not significantly different from zero, we can simplify it further to get:

- Straight line (Fig. 6-2B) with two estimated parameters a, b, where b is significantly different from zero
- Straight line parallel to the abscissa with one estimated parameter a, which is significantly different from zero (Fig. 6-2C)

Given the marginality rule, it is not (under a standard situation) permitted to remove a lower non-significant term from the model if any of the higher ones are significant. The intercept is formally *lower* than the linear term, which is, in turn, *lower* than the quadratic term, etc. In special cases, we can remove the intercept during the simplification procedure, while preserving the linear term (for example, if we know that such a model is meaningful from some external theory). For similar reasons, one might occasionally decide to override the marginality rule and remove a linear term while keeping the quadratic trend, etc.

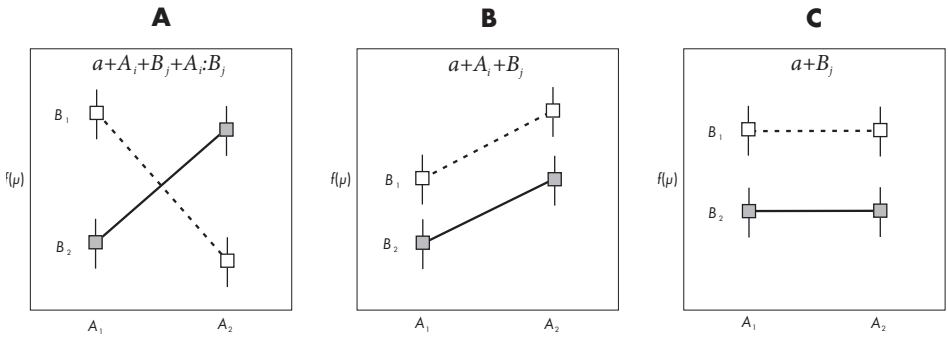


Fig. 6-3 Examples of linear predictors of ANOVA with two factors *A* and *B*. Points are means, whiskers correspond to the SEs.

6.2 ANOVA and ANODEV

Like regression models, ANOVA/ANODEV models can have many different forms. Depending on the number of explanatory variables, we recognise one-way (with a single factor), two-way (with two factors), up to *k*-way ANOVA/ANODEV. Two-way and higher order ANOVA/ANODEV models may or may not include interaction(s).

Let's look at a typical example of an ANOVA model. Given two factors, *A* and *B*, where *I* is the number of levels of the factor *A,* and *J* is the number of levels of the factor *B*, the linear predictor can have the following (most general) form:

$$\alpha + A_i + B_j + A:B_{ij}\,. \tag{6-4}$$

Under the treatment parametrization, this model includes several restrictions, namely, $A_1 = 0$, $B_1 = 0$, $A{:}B_{1j} = 0$, $A{:}B_{i1} = 0$ for $i = 1,\ldots, I$ and $j = 1,\ldots, J$. The interpretation of the model coefficients is a bit complicated, but not much. α is the response mean (or link transformed mean) for the combination of reference levels, namely of the A_1B_1 combination. A_i and B_i are I-1 and J-1 main effects. $A{:}B_{ij}$ are $(I\text{-}1).(J\text{-}1)$ interaction effects. In the summary output of such a model, we will find $I.J$ coefficients altogether.

If each of two factors, *A* and *B*, has two levels (A_1, A_2, B_1, and B_2) then, in case the interaction $A{:}B$ is present, the model has four parameters. Lines connecting two levels of one factor may not be parallel to the two levels of the other factor. The lines can even cross in the plot (Fig. 6-3A). The four coefficients in the summary output have the following interpretation: expected value for (or link-transformed expected value in case of general GLM) the combination A_1B_1 and three differences among means for different combinations: two differences between two different combinations $A_1B_2 - A_1B_1$ and $A_2B_1 - A_1B_1$, and one difference between four different combinations $A_2B_2 - A_1B_2 - A_2B_1 + A_1B_1$. Simplification of the model will lead to several special cases:

- If the interaction is not significant while both main effects are, then the lines connecting factor levels are (approximately) parallel, but not overlapping (Fig. 6-3B). Such a model contains three parameters: (link-transformed) expected value of A_1B_1, and two difference coefficients: $A_2B_1 - A_1B_1$ and $A_1B_2 - A_1B_1$.
- If a single factor is significant, say factor B, the (link-transformed) expected values of levels of this factor are different from each other (Fig. 6-3C). The model has two parameters: expected value of B_1 and difference $B_2 - B_1$.
- If the effect of no factor is significant, we obtain a **null model** with only a single parameter – estimate of (link-transformed) expected value, i.e. of the grand mean.

6.3 ANCOVA and ANCODEV

This model is a "hybrid" of the regression and ANOVA/ANODEV models. Continuous explanatory variables are generally called covariates. Historically, the ANCOVA model with a single factor and one covariate (without interaction) was established for data analyses in situations when the effect of an explanatory categorical variable needs to be tested while taking into account a nuisance effect of an additional explanatory continuous variable (to adjust for the effect of a covariate so that it will not interfere with the testing of interest). We shall describe here a slightly more general (and thus also more complex) model that will also allow for interaction. This model is, by some authors, also called ANCOVA (although this is somewhat misleading). For a single factor A, with J levels (e.g., A_1, A_2) and a continuous covariate x, the linear predictor looks like:

$$\alpha + A_j + \beta x_i + \delta_j x_i . \tag{6-5}$$

In the treatment parametrization, α is an intercept of the reference level of the factor A (i.e. A_1). A_j is an effect of j^{th} level of the factor A. It corresponds to the difference between intercepts of the j^{th} level of factor A and level A_1. β is the coefficient of the linear term (slope) for the reference level. δ_j is a difference in the slope of j^{th} level from β.

Sometimes, it will be reasonable to start with a somewhat larger model that will also include a quadratic term in x and its change with the level of A (interaction with A is then present not only in the linear, but also in the quadratic term), i.e.:

$$\alpha + A_j + \beta x_i + \delta_j x_i + \gamma x_i^2 + \omega_j x_i^2 . \tag{6-6}$$

Interpretation of α, A_j, β and δ_j is similar to that in (6-5). γ is a parameter of the quadratic term for A_1 ω_j (estimated as w) is the difference in quadratic coefficients between the j^{th} and the first level of A. For a two-level factor A, the model will have six estimated parameters: intercepts for the two levels of the A factor ($a_1 = a$, $a_2 = a + A_2$), two coefficients of the linear terms ($b_1 = b$, $b_2 = b + d_2$), and two coefficients of quadratic terms ($c_1 = c$, $c_2 = c + w_2$). The model hence consists of two generally different quadratic curves for two different levels of A (Fig. 6-4A). The curves can be qualitatively similar or different. They can be parallel, they can cross, or there might be one convex and the other concave. In the summary output, there

are six coefficients: values of a_1, b_1 and c_1, and differences $a_2 - a_1$, $b_2 - b_1$ and $c_2 - c_1$. Model simplification would lead to various outcomes. For instance:

- If neither quadratic term, nor its change among *A* levels is significant, but the interaction *A*:*x* is significant, then the model reduces to two lines. The two lines for different *A* levels are not parallel. They may or may not cross (Fig. 6-4B). The model has four parameters. In the output, there are four coefficients: a_1, b_1, differences $a_2 - a_1$, and $b_2 - b_1$.
- If the interactions $A{:}x^2$ and $A{:}x$ are not significant, but the effect of *A* and the quadratic term in the covariate *x* are significant, then we have quadratic curves which are parallel (Fig. 6-4C). The model has four parameters. In the output, there are estimates of four coefficients: a_1, b, c and a difference $a_2 - a_1$.
- If the factor *A* and the covariate *x* are significant, but both quadratic terms are non-significant, then we have two parallel lines (i.e. they have identical slope, but possibly different intercepts, Fig. 6-4D). The model has three parameters. In the output of the model, there are estimates of three coefficients: a_1, b and a difference $a_2 - a_1$.

Further simplification of the model (6-6) will gives us a regression model with a single explanatory variable if the factor *A* is not significant (Fig. 6-2A–C), or an ANOVA/ANODEV

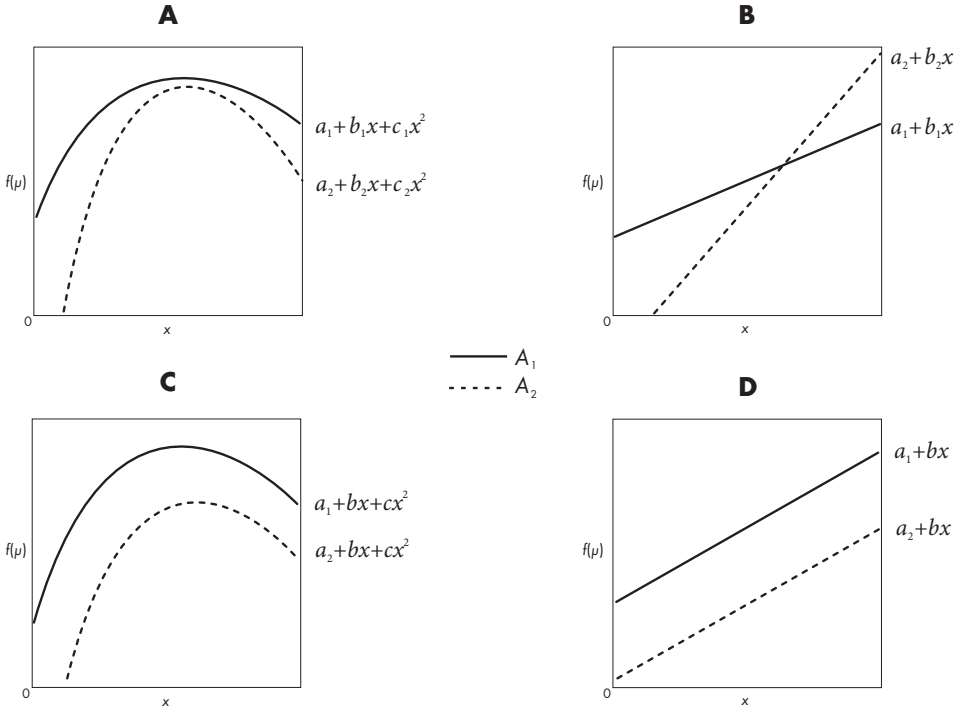


Fig. 6-4 Examples of models containing both a factor (*A*) and a covariate (*x*) showing different complexity and different qualitative features.

model with a single factor if both linear and quadratic coefficients of a covariate are not significant (Fig. 6-3C).

6.4 Syntax of the systematic part

Entering the systematic part in R is governed by several simple syntactic rules. These rules are very intuitive and natural. The systematic part in the **`glm`** as well as **`lm`** functions is entered using the **`formula`** argument. In fact, it is the first argument given to any of these two functions. While in the **`lm`** function, the **model formula** determines the model completely (both link and the random part are determined automatically), the **`glm`** function it is more flexible, so that we have to specify more explicitly what we want. Apart from the linear predictor, we have to specify the distribution type (random part) and the link transformation. We have to be careful though, because if we do not specify them, the Gaussian (or normal) distribution with the identity link will be used by default. This corresponds to what is done by **`lm`** and it might not be what we intend to do. If we specify a random part and no link, **`glm`** uses the so called canonical link for the particular distributional family. Again, that might or might not be what we want to invoke for a particular analysis.

Different distributions imply different assumptions about how variance depends on the mean (different variance functions). So, variance, or scedastic function is selected by specifying the random part. Additionally, one might also specify a **`weights`** argument, but we will do so only in rare occasions.

In all examples used in this book, we will specify only linear combinations of parameters. As we have already explained earlier (Chapter 4.2), explanatory variables, as they enter a model, can be transformations of original explanatory variables. The model formula can be generally expressed as follows:

response variable **~** *linear predictor*.

Tilde (~) reads as "it is modelled by". One response variable is always on the left side of the model formula. On the other side, we specify a linear predictor as a sum of explanatory variables, separated by the plus (+) sign (or other special characters, the meaning of which you will find further in the text). It is a stylised and abbreviated notation. We need to realise that the meaning (and practical interpretation) of this formula *considerably* changes when we move between particular sub-classes of the GLM models. For example, if we select normal distribution (and identity link), then using the linear predictor, we model directly the mean value of the given response variable. That is why the interpretation of coefficients, their SEs, etc. is so simple. For other distributions and links, this can change quite dramatically (see Chapter 9–12). For instance, the logarithm of the intensity (of the mean) might be modelled by the linear predictor in a Poisson GLM; logistic transformation of the mean might be modelled in binomial regression, etc. We have to keep all of this in mind when conducting GLM modelling, making sure we are always clear about the things we do and how we actually specify and interpret the model. At the same time, we have to take into account the

Table 6-1 Overview of basic models, their mathematical formula, syntax and description.

Formula	Syntax in R	Description
$f(\mu_i) = \alpha$	`y ~ 1`	Model containing only the intercept, i.e. the **null model**
$f(\mu_i) = \alpha + \beta x_i$	`y ~ x`	Linear model with a continuous explanatory variable x
$\log(\mu_i) = \beta x_i$	`log(y) ~ x-1`	Linear model with a continuous explanatory variable x, without the intercept, with logarithmically transformed expected values of the response variable (y)
$f(\mu_i) = \alpha + \beta x_i + \gamma x_i^2$	`y ~ x+I(x^2)` or `y ~ poly(x,2)`	A quadratic model of the continuous variable x
$f(\mu_i) = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i}$	`y ~ x1+x2`	Model with two continuous variables x_1 and x_2. It is linear in each of these variables. This is a particular case of multiple regression.
$f(\mu_{ijk}) = \alpha + A_i + B_j + C_k + A:B_{ij} + A:C_{ik} + B:C_{jk} + A:B:C_{ijk}$	`y ~ A+B+C+A:B+A:C+B:C+A:B:C` shortly `y ~ A*B*C`	Model with three factors A, B and C including three main effects, three two-way interactions and one three-way interaction. This is a general form of three-way ANOVA/ANODEV model with interactions.
$f(\mu_{ijk}) = \alpha + A_i + B_j + C_k + A:B_{ij} + A:C_{ik} + B:C_{jk}$	`y ~ (A+B+C)^2`	Model with three factors (A, B, C), including three main effects and three two-way interactions only.
$f(\mu_{ij}) = \alpha + A_j + \beta x_i + \delta_j x_i$	`y ~ x*A`	Model with a continuous variable x and a factor a including two main effects (regression on continuous and effect of a factor) and interaction (change of slope with the levels of the factor A).

specification of the systematic part, distribution specification, link specification and their interaction (i.e. the fact that the meaning of the specification of the systematic part is generally different for different distributions and different link functions).

While on the left side of the model formula, the operators like **+**, **−**, **/** have a mathematical interpretation, on the right side of the model formula (i.e. in the linear predictor), these operators have a different meaning! Specifically, **+** is to add, **−** is to remove, **:** is an interaction, ***** stands for all terms (involving main effects and interaction of variables connected by *****). If

we need to use these operators in the usual mathematical sense within the linear predictor, it is necessary to use an interpreter argument, **`I`**. Number **`1`** has a special meaning in the linear predictor – it stands for the intercept. Thus, for example, **`-1`** does not mean that we want to subtract 1 but that we are fitting a model without the intercept!

The Table 6-1 demonstrates several examples that illustrate the translation between the standard mathematical expression and R notations of the systematic part. We should remind you that we will generally use lowercase letters (x) for continuous explanatory variables and uppercase letters (A, B, C) for categorical explanatory variables. This is just for simplicity. Clearly, you can use whatever (legal) names you like when writing your own models. The left side of the systematic part of the model generally includes a link to the transformed mean value of the response variable, $f(\mu)$.

7 RANDOM PART

By specifying the systematic part, we have not yet described the entire model that we have in mind. At least, not with regard to the function **glm**. In the **glm** case, we need to specify the distribution type and the link function, which determine how the expected value of the response variable will be transformed and modelled by the previously specified linear predictor. In contrast to this, **lm** always assumes a normal distribution of errors and is thus more convenient to use in simple cases, but is also much more restrictive. The **glm** function has default settings for distribution type and link which will be automatically invoked when we do not specify something else. This means that if we specify only the systematic part, **glm** will not warn us and will perform an analysis with Gaussian distribution and identity link. Fortunately, **glm** will report in the output basic information allowing us to check various things that the function did.

Specification of the random part (and also of the default link function) is mainly based on the distribution type from which the response variable comes. The following key should help you to decide.

If the response variable is composed of

- Continuous measurements 7.1
- Counts and frequencies 7.2
- Relative frequencies 7.3

This is far from being really strict. In order to make the right decision, you need to become familiar with basic properties of the distribution types that we will be working with in this book. Obviously, the number of existing distributions is infinite. Nevertheless, here we will select only those that are available in the **lm** and **glm** functions in the R environment. When making our decisions about the distributions within the GLM family, we will be guided either by theoretical considerations (for example, ecological, physical), or by previous experience (our own or that of other authors) with analyses of similar empirical data. However, should there be no theories or experiences regarding a particular case, which is often the case in biology, we need to start searching, experimenting – and critically thinking about the results. If it seems that, for some reason, your data (or the mechanism that generates them) do not comply with any standard assumptions, you need to consult a professional statistician. He or she might help you to develop a customised model for the problem if the departure from available GLM models is really substantial.

7.1 Continuous measurements

Strictly speaking, continuous data are such that each measurement can be obtained with "infinite" accuracy – i.e. with an infinite number of decimal places. It is obvious that something like this is possible only approximately or within a theoretical model. Real measurements are always rounded to a certain number of decimal places, thus they actually become discrete. Why should we then even try to develop continuous data models? Why are not all statistical distributional models discrete?

Continuous models are often much "more convenient" from mathematical and statistical perspectives. Often there is an explicit formula for their density which offers a simple and straightforward interpretation. For such models, it is easy to communicate the model parameters with others (since the meaning of the parameters is intuitive). The estimation procedures for such models tend to be simple from both statistical and numerical points of view. Moreover, inaccuracy caused by approximation of the discrete data is in many cases so small that it can be, in relation to the extent of the measured values, ignored. However, this is not always the case, so we cannot always ignore it automatically. The main aspect here is the *relative* error (caused by rounding) – in relation to the "useful" extent of the measurements. It can easily happen that, we model measurements of the same accuracy continuously, while in other cases, we will do it discretely – depending on the range of the measured values within the given study. For example, accuracy of 0.5 for data in the range of 0 to 1 will reliably create data which will be perceived as discrete. The same accuracy for data effectively varying within the range of 0 to 1000 will most likely tend to be perceived as continuous. The error incurred by such continuous data approximation will be most likely negligible for most practical purposes. The decision might not always be so clearly cut, however. It can be more difficult because discreetness/continuity depends not only on the quantity we measure, but also on the measurement conditions and on the homogeneity of the tested subjects (for example, laboratory animals), etc. We should always think about the discrete vs. continuous choice (and its consequences) thoroughly before actually running models in R.

To describe continuous data, the **Gaussian (normal) distribution** is very often used in practice. To some extent, it is being overused (or even being abused by some). The following characteristics of one-dimensional normal distributions are of practical interest:

- The bell shape of the probability density function, which is symmetric around the mean value, which is also the most common value (the only mode of the probability density).
- The true median and the true mean, i.e. the true expected value (μ), of the distribution are the same. Note that this is typically not the case for the characteristics computed from (a finite) sample of data.
- The mean value (μ) and variance (σ^2) always exist (for any normal distribution). They fully determine a particular normal distribution (among all possible normal distributions). For a given μ, we can have normal distribution with whatever $\sigma^2 > 0$ (and *vice versa*). There are no functional restrictions on their combinations.
- Sample variance (s^2) is independent of the expected value (arithmetic mean of the data, $\bar{y}$).

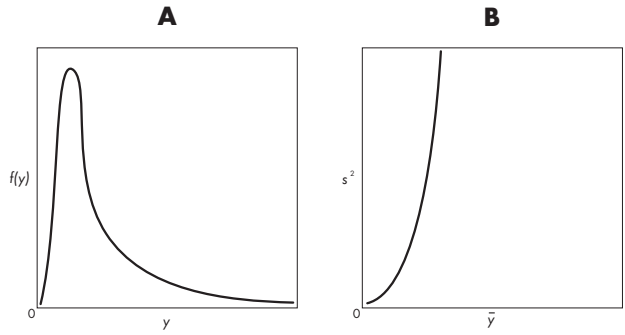


Fig. 7-1 A. Density of a lognormal distribution. **B**. Ideal relationship between estimated standard deviation (s) of the data subgroups and their means $\bar{y}$.

The Gaussian distribution is often a suitable approximation for the behaviour of measurements, whose spread is not limited from either side, which are not distinctly discrete and which do not feature substantial outliers. Nevertheless, occasionally the normal approximation can be still useful when the range of the data distribution is limited but the bounds are relatively far away from the bulk of the data. Obviously, how far depends on both the absolute location of the limits and also on the variability of the data. For instance, left and right bounds of 0 and 100 will tend to be practically negligible for data with a mean of 50 and standard deviation of 1, but they can easily be quite important (and hence invoke distinctly non-normal behaviour) for data with a mean of 50 and standard deviation of 50. We will discuss Gaussian models in a more detail later in Chapter 8.

Measurements of length, weight or time are (often severely) limited from the left – they do not have any negative values. Moreover, zeros are also unacceptable – organisms that have no mass or length cannot exist. If the measurements come from a range that is close to the left bound (such as zero), their distribution is often asymmetric (skewed to the right). There really are many right skewed distributional models available. Out of these, for instance the gamma and inverse Gaussian distributions and of course, the lognormal (after pre-transformation of data) are accessible in the GLM models of R.

The following characteristics of **lognormal distributions** are interesting:

- Asymmetric shape, skewed to the right (Fig. 7-1A).
- The mode is smaller than the median, which is, in turn, smaller than the mean value.
- When comparing multiple groups of data, the variance will grow with increasing mean values. Often, the lognormal distribution is used in a regression context to model data with a constant coefficient of variation (instead of constant variance as in the standard linear regression). For a constant coefficient of variation, the relationship between standard deviation and mean will be approximately linear, with zero intercept (Fig. 7-1B). The variance will then often be reasonably close to a constant after a logarithmic transformation.

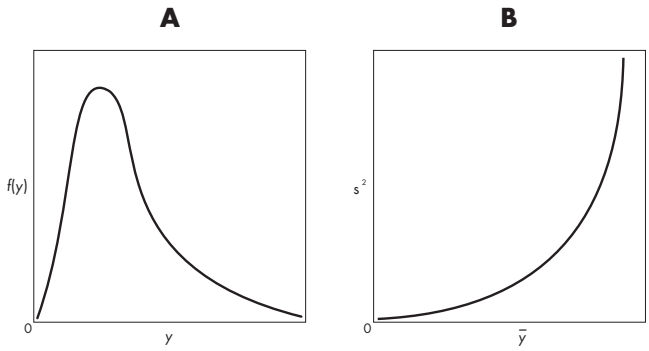


Fig. 7-2 A. Density of gamma distribution. **B**. Ideal relationship between estimated standard deviation (s) of the data subgroups and their means $\bar{y}$.

Several characteristics of the **gamma distribution** are similar to the lognormal distribution (to the extent that the choice of one over the other for a concrete data analysis should be based rather on theoretical and subject-matter arguments rather on simple minded attempts to optimise the appearance of Q-Q plots). Like the lognormal, the gamma is also a distribution of positive quantities. It is skewed to the right (Fig. 7-2A) and often it is used to model data with a constant coefficient of variation (Fig. 7-2B) – for instance, in a regression context, when we assume constant statistical properties of relative rather than absolute error across the range of explanatory variable (variables).

We will address the data from both distributions (lognormal and gamma) in detail in Chapter 9.

Another alternative is the **inverse Gaussian distribution**. Its application is frequent in chemistry and physics, for example, for describing diffusion processes. In biology, it can be used, for modelling the spreading of organisms, etc.

Many additional types of continuous distribution are used for describing time until the occurrence of an event (such as death, birth, attack, morphogenesis, etc.). If we know the times exactly for all measurements, then we can use regression methods based on GLM (if the duration distribution can be approximated by a member of GLM class). On the other hand, if the time is not known for all data (the data are left, right, or interval censored), we have to use the more complicated approach of **survival analysis**, instead.

7.2 Counts and frequencies

The measurements of biological quantities are often non-negative integers. This applies to the **counts** or **frequencies** of individual kinds, specimens, cells, and events that we are interested in. The basic distribution for describing frequencies is the **Poisson distribution**. The following are some of its interesting characteristics:

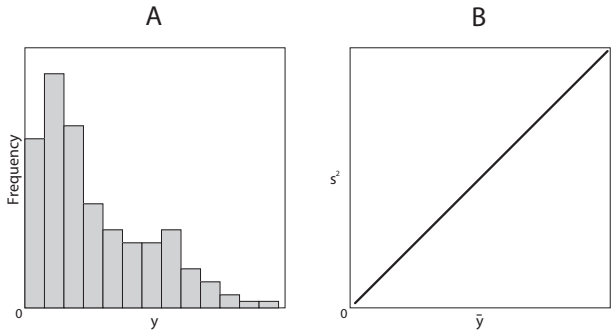


Fig. 7-3 A. Histogram of Poisson distribution. **B**. Ideal relationship between sample variances (s^2) and means ($\bar{y}$).

- It is discrete; it can assume only non-negative integer values (0, 1, 2, …)
- It is typically highly asymmetric if its mean is not very large (Fig. 7-3A)
- Its variance is equal to the mean value
- Plots of variance against mean (e.g. computed from different data subgroups) should show approximately zero intercept and identity slope (Fig. 7-3B)

Examples of analyses based on the Poisson distribution are given in Chapter 10.

If the Poisson distribution is not suitable for a practical analysis because the data are over-dispersed (i.e. they show variance substantially larger than the mean, i.e. **overdispersion**, see Chapter 10.3), a more realistic alternative could be the **negative-binomial distribution**. It can be derived as a compound distribution – i.e. as a hierarchical Poisson model in which the Poisson parameter randomly changes (according to the gamma distribution). It follows

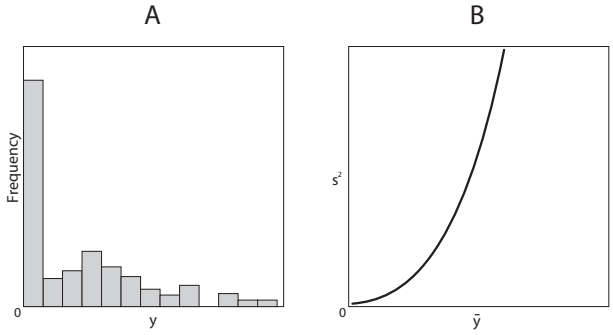


Fig. 7-4 A. Histogram of negative-binomial distribution. **B**. a possible relationship between sample variances (s^2) and means ($\bar{y}$).

from this that the negative-binomial distribution is often used for the description of (random) heterogeneity in Poisson data. Such a distribution is:

- Discrete
- Asymmetric for non-negative integer values (Fig. 7-4A)
- Unlike the Poisson distribution, its variance is greater than the mean value; thus, when we study the relationship between variances and mean values taken from multiple samples of different negative-binomial distributions with identical overdispersion, the relationship is parabolic (Fig. 7-4B)

You can find an example of an analysis based on the negative-binomial distribution in Chapter 11.

7.3 Relative frequencies

If we observe how many times a phenomenon (y) occurred, while knowing the sample size or total number of units inspected/observed (n), we can calculate empirical relative frequencies (p). They express the proportion of "successes" (estimating the probability of the phenomenon studied): $p = y/n$. This type of data is quite fundamentally different from continuous measurements. They are discrete. While an indicator of whether the phenomenon of interest occurred or not in a single unit (in a single trial) it can be viewed as a kind of qualitative characteristic (yes/no); the number of successes out of the total number of trials larger than 1 is quantitative, but in a different way than with previously discussed continuous measurements.

The relative frequencies (p), together with given/fixed sample sizes (n), give (empirical) estimates of unknown theoretical probabilities of success (π). Both relative frequencies and probabilities, which they estimate, are bound from both sides (obviously they have to be within the interval between 0 and 1). This is a complication, which can be suppressed by a suitable transformation. The logit transformation is often used in this context:

$$\log\left(\frac{\pi}{1-\pi}\right), \tag{7-1}$$

Logit is the natural logarithm of the ratio of probability of successes and probability of failures. The plot of the logit function is shown in Fig. 7-5A.

When analysing relative frequencies, we can use either the general binomial distribution (for n trials), or its special case – the **Bernoulli distribution** (for $n = 1$, one trial).

The following are some of the features of the **binomial distribution**:

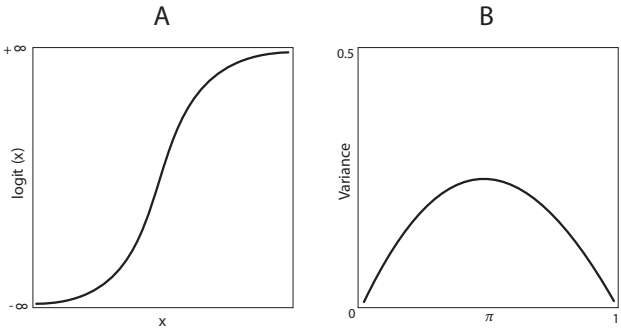


Fig. 7-5 A. Logit transformation. **B**. Relationship between variances and probability (π) for $n = 1$.

- The values of the measurements (y) can be integers 0, 1, ..., n, where n is the number of (independent) Bernoulli trials, for which the occurrence of the given phenomenon has been studied.
- For a given n, the distribution is described by a single parameter π. It is the probability that the given phenomenon has occurred. As any probability, it has to satisfy $0 \leq \pi \leq 1$.
- Variance of samples changes with their mean value. Variability is small if the value of π is close to the extremes (0 or 1). On the other hand, it is maximal in the middle of π range, i.e. when $\pi = 0.5$ (Fig. 7-5B).

You can find examples of analyses based on binomial distributions in Chapter 12.

8 GAUSSIAN DISTRIBUTION

The Gaussian or normal distribution is often used for modelling data whose distribution is reasonably symmetric and without overly fat "tails" (i.e. not having too large a probability of very large or very small values). The normal model assumes that the observations can occur anywhere between -∞ and ∞ (although with very different probability). However, this is, of course, not the case in real life. Measurements are often bounded both from above and from below due to the physical limitations of both the process under the study and limitations of the measurement devices. As is common in statistics, when using normal distribution, we only approximate, more or less successfully, the reality. The normal approximation can be convenient, especially when the upper and lower bounds are relatively distant from our observations and the data behave "reasonably" continuously (i.e. the range of data is significantly bigger than the smallest step of the measuring device – see Chapter 7.1) and when we do not have to cope with the natural occurrence of very large/small outliers.

Many R functions are based on statistical methods derived under normality. Some of them implement techniques that all of you have certainly encountered in elementary statistical courses. For instance, you can perform the **one-tailed** and **two-tailed t-test** using the function **`t.test`**. The statistical theory for normally distributed data is the most developed. The function **`lm`** implements the general linear model. It can cope with a wide range of linear analyses. Among other features, it incorporates regression, ANOVA and ANCOVA with a normally distributed response variable. Normal-based analyses can also be obtained from the more general **`glm`** function. It uses the Gaussian distribution (with identity link) as a default.

8.1 Description of LM and GLM

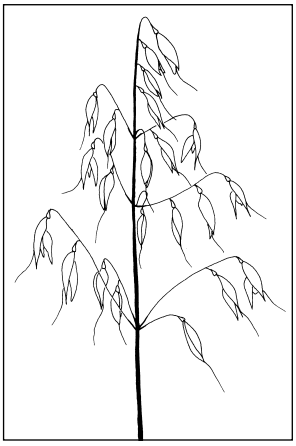
When using **`lm`** or **`glm`** with the default option of **`family=gaussian(link="identity")`**, we assume that measurements y are normally distributed and the expected value μ is a linear function of predictors (as explained in Chapter 4.2). Parameters μ and σ^2 are completely "free" – variance is not related to the expected value.

Note that by choosing Gaussian GLM with a different non-identity link, we choose among different kinds of nonlinear regressions with normal errors. Only a relatively small subclass of nonlinear regressions are accessible this way, however.

Both functions, **`lm`** and **`glm`** (with the default options) model measurements at the ori-ginal scale. Then there is no need to transform estimated parameter values, predictions, etc., un-

like in more complicated GLM model cases. When using **lm** and Gaussian **glm**, the parameter estimates are the same. For standard analysis of normal data, it is natural to use **lm** (it offers some features specialised for the normal models which are not available in the more general **glm**). The **glm** function is useful when working with other than normal GLMs. Nevertheless, it is used also for normal data with other than identity links.

8.2 Regression



The number of grains in ears affects the yield of cereals. On 20 oat plots, a sample of ears was taken and the mean number of grains *per* ear was estimated (*grain*). At harvest, the yield [t/ha] for each plot was estimated (*yield*). We are interested in the relationship between yield and number of grains in order to predict the yield. Specifically: (1) Is the number of seeds related to the yield? (2) If the answer to the first question is positive, can we built a predictive model for yield, based on its relationship to the number of grains?

EDA

The response variable is *yield*, while *grain* is the explanatory variable. Both of them are continuous. Hence, when exploring their relationship, it is natural to draw them in a scatter plot.

```
> dat<-read.delim("oat.txt"); attach(dat); dat
   grain yield
1  18.07 4.913
2  22.98 5.170
3  28.94 5.820
...
20 23.68 5.554
> plot(grain,yield)
```

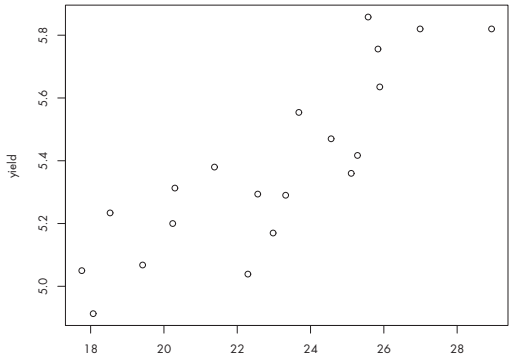


Fig. 8-1 Scatterplot of *yield* and the mean number of grains (*grain*).

The relationship between *yield* and the number of grains in the ear corresponds to what we would suppose, based on "common sense": the yield increases when the number of grains increases (Fig. 8-1). The straight line does not seem to be a bad model choice. Nevertheless, by visual inspection alone, we cannot decisively exclude curvature in the relationship (e.g. presence of the quadratic trend).

MODEL

The relationship between the *yield* and the number of grains in the ear has, to a certain extent, a linear character. That is why we might, apriori (before looking at the data), expect the following model for the measurement on the i^{th} plot:

$$yield_i = \alpha + \beta grain_i + \varepsilon_i, \quad \text{where } \varepsilon_i \sim N(0,\sigma^2), \text{ independent among plots.} \tag{8-1}$$

Unknown values of parameters α, β (and σ^2) will be estimated from the data. Notice that we assume a normal distribution of errors ε, in agreement with a generally accepted assumption about the behaviour of the relatively precise measurement of heavy objects. We are assuming that the information about the *grain* is very precise, hence ignoring the so called "errors in variables" problem (Carroll *et al.* 1995). This is reasonable if the measurement error of the variable *grain* is rather small compared to the measurement error of *yield*.

ANALYSIS

The initial model should not be too small – if it is, it could unnecessarily exclude components that are significant. That is why we will enrich the linear trend model with a quadratic term. This could be seen as a crude check for nonlinearity in the yield to grain relationship. The quadratic is the simplest among the non-linear polynomial terms. Polynomial terms of higher order than the cubic (i.e. of the third order polynomial) are used rather rarely, in practice. It is due to the limited interpretability of the coefficients and likely quite erratic behaviour of high degree polynomials fitted to the random data. Polynomials (especially those of high degree) typically behave nicely within the range well covered by the data, but they might go totally wrong close to the boundaries of the data (and in areas less covered by data in general). There are better alternatives to polynomials for fitting the details of complicated nonlinear curves – most notably the splines (De Boor 2001), but we will not pursue them here. Another alternative can be to include various non-linear trends motivated by a background theory. For simplicity, we will fit a just quadratic model here:

$$yield_i = \alpha + \beta grain_i + \gamma grain_i^2 + \varepsilon_i, \quad \text{where } \varepsilon_i \sim N(0,\sigma^2), \text{ independent among plots.} \tag{8-2}$$

We use the functions **`lm`** and **`poly`** to specify the quadratic term. Specifically, **`poly(grain,2)`** used within the model formula stands for a quadratic trend (polynomial of the 2nd order) of the explanatory variable *grain*. Model output is shown by calling **`summary`**.

```
> m1<-lm(yield~poly(grain,2))
> summary(m1)

Call:
lm(formula = yield ~ poly(grain, 2))
Residuals:
      Min        1Q    Median        3Q       Max
-0.261562 -0.121112 -0.003686  0.142558  0.281990

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      5.38205    0.03417 157.520  < 2e-16 ***
poly(grain, 2)1  1.04875    0.15280   6.864 2.75e-06 ***
poly(grain, 2)2  0.15416    0.15280   1.009    0.327
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1528 on 17 degrees of freedom
Multiple R-Squared: 0.739,      Adjusted R-squared: 0.7083
F-statistic: 24.06 on 2 and 17 DF,  p-value: 1.101e-05
```

We saw a similar summary output in Chapter 5.4.5, but there, it was output of an ANOVA model. It is a regression here. The first part of the output is similar to what we saw previously, but the table of coefficients is different. On the first line there is an intercept. On the second line, there is a linear term (poly(grain, 2)1), and on the third line there is a quadratic term (poly(grain, 2)2). Estimates of the three parameters are shown in the column labelled Estimate. Next to them, there is a column with the respective standard errors, labelled Std. Error. These are standard errors derived and computed under the model assumptions (if the model is not correct, estimates of SE can be completely wrong). Then there are t values for each parameter in the next column. These are obtained by dividing estimates by their standard errors (e.g., 1.04875/0.1528 = *6.864* for the linear term coefficient). These are testing the null hypothesis that the value of a parameter is zero.

The bottom part of the output, i.e. the statistics describing the quality of the whole model, is also similar to that for the ANOVA model we saw earlier. The residual standard error is an estimate of the residual standard deviation. There are 17 residual degrees of freedom. In this example, we had 20 measurements. Subtracting *3* (the number of estimated parameters), we obtain the residual degrees of freedom. Further down, there is the **coefficient of determination** (Multiple R-squared) and **adjusted coefficient of determination** (Adjusted R-squared). Adjusted R^2 takes into account the size (or complexity) of the model and hence is a better descriptor of the model quality than the multiple R^2. It penalises the size of the model and in this way it opposes the widely known problem that by increasing the number of explanatory variables, the standard coefficient of determination cannot be worse. At the very bottom of the summary output, there is an overall F-test. This is a test of the null hypothesis that our model is completely inappropriate, i.e. that it does not explain variation in the response variable better than the model having only an intercept and no other systematic terms. It is clear that a significance of this test is its (very) minimal requirements. This test is typically not used very much, but it appears in other statistical software as well.

Be careful: the **poly** function, which we called during the model fitting, uses somewhat different parametrization (orthogonal parametrization) to the model described in (8-2). That is why the coefficients have different meaning, but the model fit is the same under the two parametrizations (exactly as with the reparametrizations of the ANOVA models which we met earlier). Since we do not directly work with the coefficients and since we will not interpret their values, this fact does not concern us (the quality of the fit and their characteristics as well as the residual standard error or R^2 remain the same). However, should we want to know the coefficients in the parametrization (8-2), we just need to specify a linear predictor, in a slightly different way: **grain+I(grain^2)**, as we already explained in Chapter 5.4.2.

Apart from the quadratic term, all other terms are significantly different from 0 in our model. The quadratic term is thus clearly not necessary and we will leave it out of the following model. By this simplification, we are thus approaching a model that we suspected intuitively at the very beginning (8-1)! However, we have verified that the quadratic trend has not been unjustifiably omitted and that our (originally quite informal) linearity assumption appears to be quite reasonable in the light of the data.

```
> m2<-lm(yield~grain)
> summary(m2)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.63509    0.25694   14.15 3.42e-11 ***
grain        0.07617    0.01110    6.86 2.03e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1529 on 18 degrees of freedom
Multiple R-Squared: 0.7233,     Adjusted R-squared: 0.708
F-statistic: 47.06 on 1 and 18 DF,  p-value: 2.033e-06
```

In model `m2`, both parameters are significantly different from zero. The first one with the p-value $3.42x10^{-11}$ and the other with the p-value $2.03x10^{-6}$. The latter value is, in this case, identical to the p-value of the overall F-test, because t^2 is, in this case, equal to the F-statistics. Notice, that p-values are often printed in R with overwhelming precision (e.g. 10^{-11})! For practical purposes (and publications), it is certainly sufficient to report four decimals and to use p-value < 0.0001 for all p-values that are lower.

Based on the relatively high coefficient of determination, $R^2 = 0.708$ (about 71% of the variability in variable *yield* can be explained by our simple linear model), the quality of the fit appears to be relatively good. But we have to be careful! The decision as to what is and what is not a satisfactorily high coefficient of determination depends on the context. For example, $R^2 \approx 0.3$ can be quite good for predicting behaviour of various share prices but it will be completely unsatisfactory for predicting values, based on which medical cures of patients are determined. While it is clear that the higher R^2 is, the better, general recommendations with regard to its values can hardly be given without knowledge of the impacts of prediction inaccuracy.

Can we make the already small model m2 even smaller? In other words, do we really need both parameters, the intercept and the slope in the model? Let us think about the biological interpretation of the parameters: can we expect any yield if there is no grain in the ears? Probably not. This "theoretical" deliberation suggests that we might try to break the "marginality rule" (introduced previously) and fit a smaller model:

$$yield_i = \beta grain_i + \varepsilon_i, \quad \text{where } \varepsilon_i \sim N(0,\sigma^2), \text{ independent among plots.} \tag{8-3}$$

It is a restricted version of the previous model. The intercept is now forced to be zero and not freely estimated from the data, as it was in the previous model (8-1). The only coefficient of the restricted model is the slope. Together with the residual variance parameter, they completely determine the fitted model. The required model changes of m2 can be achieved by the **update** function. There, we will use the **-1** argument, which stands for "remove the intercept".

```
> m3<-update(m2,~.-1)
> summary(m3)
...
Coefficients:
      Estimate Std. Error t value Pr(>|t|)
grain 0.231855   0.005006   46.32   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.518 on 19 degrees of freedom
Multiple R-Squared: 0.9912,     Adjusted R-squared: 0.9908
F-statistic:  2146 on 1 and 19 DF,  p-value: < 2.2e-16
```

In model m3, the slope (*0.2319*) is significantly different from zero. The coefficient of determination has significantly increased in comparison with the first model (*0.991*). But watch out! This value cannot be compared with the value obtained prior to removing the intercept: R^2 is defined differently in models with and without the intercept (Rawlings 1988). If we want to compare both models, we can use, among others, the Akaike information criterion. It can be calculated by the **AIC** function.

```
> AIC(m2,m3)
   df       AIC
m2  3 -14.47461
m3  2  33.42234
```

In terms of AIC, the model m2 is better than m3. AIC prefers small models due to penalisation of model size (in this case the penalisation is not large enough to prefer a smaller model, which fits the data less well). Penalisation is related to the generalisation of a model, and is therefore a wanted property. However, if we ask which model fits the data better, the answer is clear: the larger model, m2, of course. What is not clear in general, is whether the fit of the larger model m2 is so much better that it is still preferred after the complexity (measured by model size) is taken into account. In this case, the answer is affirmative - m2 is also better from the AIC point of view. How much differently the models m2 and m3 fit the data will become clear when we draw both models (lines) in the same plot using the **abline** func-

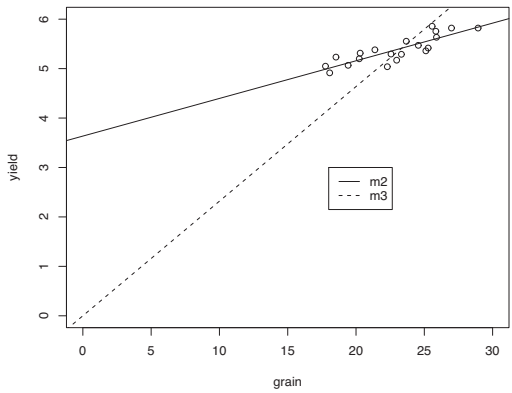


Fig. 8-2 Comparison of the fitted models of relationship between *yield* and *grain*: two-parameter model (`m2`) and one-parameter model (`m3`).

tion. We use a solid line for model `m2` (solid line is a default in **`abline`**) and a dashed line (specified by the argument **`lty`**) for the model `m3` (Fig. 8-2):

```
> plot(grain,yield,xlim=c(0,30),ylim=c(0,6))
> abline(m2)
> abline(m3,lty=2)
> legend(18,3,c("m2","m3"),lty=1:2)
```

Should we take the results of both models (8-1 and 8-3) completely seriously and should we use them for predicting very small values of the explanatory variable *grain* (that are not present in our data), we get results that are, to say the least, really daring (or even incorrect). Common sense suggests that such a brutal extrapolation should not be given too much attention. Why? Simply because it is based on an undue trust in linearity (or generally speaking, in the exact form of the model) far beyond the area covered by the observed data. In many real situations, linearity represents a good approximation within a narrower range of the explanatory variable. However, from a more global perspective, the form of the given relationship can often drastically change (especially close to zero). In order to verify the real shape of a functional relationship, we would have to have data from the entire full range of grain possible values. Alternatively, we might have a good theoretical (for example, physiological) model that would allow us to extrapolate to very small values. The point here (and in many other applications) is that, for many purposes, we do not actually need such extrapolation at all. For example, we may not be interested in the form of the function for poor yields at all. It is very likely that a practically oriented researcher is interested only in "reasonably good yields". In such a case (and with the understanding that the approximation only applies to a limited width of the grains range), we can use a simple linear model very well.

A general moral from the previous discussion is that "good modelling" should strive to find a model that "sits" on the data well but at the same time, is also practical (for example, not too big but readily interpretable) and most of all, it is useful for the purpose of the study. As soon as the purpose changes, the model can (drastically) change as well. Let us consider

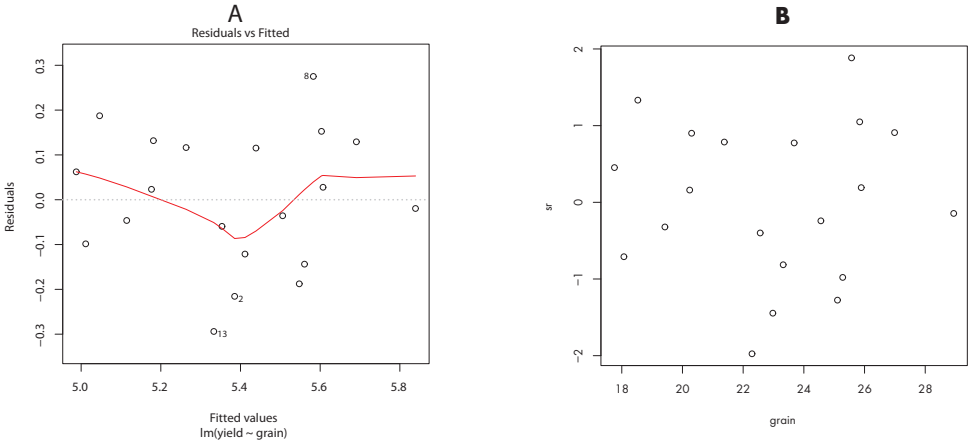


Fig. 8-3 A. Relationship between residuals and fitted values. **B**. Relationship between standardised residuals (`sr`) and the explanatory variable *grain*.

a different scenario where we are interested in how the yield changes for very small numbers of grains in the ear (e.g. from a physiological study viewpoint). Such a study would require completely different data, probably a different and more complex model type (most likely, it would be highly nonlinear close to zero), etc. Keep in mind that the moral of the previous discussion is important not only for a model-builder, but also to a model-user.

It is time to check that the model assumptions do not go against the actual data behaviour. We assumed that the model has a normal error distribution, with zero mean and a constant variance. We will look at just a few simple diagnostic plots (the reader can try others as well). We start with the first plot of residuals against fitted values. This is the first plot type offered by the **`plot`** function called with the model object argument. It is, in fact, one of the most important and most often used plots for model checking. Later on, we will also look at the relationship between standardised residuals (`sr`) and the explanatory variable *grain*.

```
> plot(m2,which=1)
> sr<-rstandard(m2)
> plot(grain,sr)
```

It might seem that the loess curve (Chapter 4.6) suggests that there is some kind of residual information in the residuals, which the linear model does not capture (Fig. 8-3A), but let us look at the scale of the ordinate – the deviances are very small. When we look at their relative size (calibrated by the mean error calculated from the model), i.e. at standardised residuals (Fig. 8-3B), we can see that the residuals are not big, not even *from a statistical* point of view – they are basically located within the ± 2 SE band. Moreover, it is also clear from the first plot that the residuals have more or less homogenous variance and no obvious systematic behaviour. The residuals do not show any systematic behaviour with respect to the explanatory variable – see the random spread of the points in the plot (Fig. 8-3B). Examination of other diagnostic plots (not shown here) does not suggest any remarkable deviations from

the assumptions. Try to look at the same model diagnostics for the model `m3`. They will indicate that model `m3` is not good at all. In fact, the model diagnostic results are much worse than those for the model `m2`. That is why we accept model `m2`. In a real study, we should verify it much more thoroughly, ideally also on new, independent data.

Finally, we will calculate the 95% confidence interval for the slope using (3-4). We will use the estimate of *b* and its SE from model `m2`:

```
> 0.07617+0.0111*c(qt(0.025,19),qt(0.975,19))
[1] 0.05293743 0.09940257
```

CI_{95} can be useful for comparing the predicted value with a given value (e.g., supplied by an external theory). Based on some previous physiological research, let us assume that the yield should increase with the increasing number of grains *per* ear linearly, with a slope of *0.1*. This value does not fall within the 95% confidence interval and we can therefore already state that our data do not comply with such an assumption (at the 5% level). Should we be interested in knowing the exact p-value of the rejected null hypothesis, we can simply find it by dividing the difference between the estimated and theoretical values by the SE value from model `m2`. This random variable has, provided the null hypothesis is valid, a t-distribution with a number of degrees of freedom identical to the residual one (18). We can then calculate the p-value of the (two-sided) test as follows:

```
> 2*(1-pt(abs(0.1-0.07617)/0.0111,18))
[1] 0.04568062
```

The estimated slope is significantly different from the theoretical one, though the p-value (0.046) is very close to the conventional level. For the one-sided test of the null hypothesis, that slope ≥ 0.1, against the alternative, slope ≤ 0.1, we get the p-value:

```
> 1-pt((0.1-0.07617)/0.0111,18)
[1] 0.02284031
```

We cannot choose between the two-sided and (stronger) one-sided test arbitrarily. We should decide which test to use before seeing what the data say. For example, if the theory claims that the growth will have a slope of *0.1* or greater, we can use the one-sided test (with higher power and hence lower p-value as a bonus). However, should the tested theory claim that the slope is exactly *0.1*, we have to use the two-sided test (for such a theory, both up and down deviations from zero represent "blows" of the same seriousness).

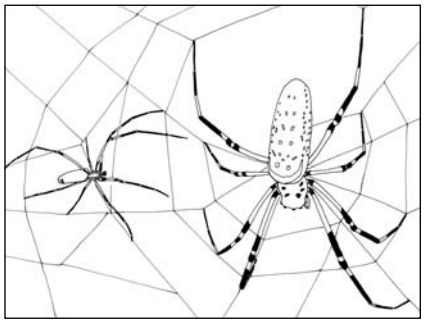
CONCLUSION

The yield of oats significantly increases with the growing number of grains per ear (LM, $F_{1,18} = 47.1$, $p < 0.0001$). The model prediction can be obtained by substituting unknown parameters in the formula (8-1) by their estimates:

$$\widehat{yield} = 3.635 + 0.076\,grain\,.$$

The 95% confidence interval for the slope is (0.053; 0.099).

8.3 Weighted regression



Weighting allows balancing of the influence of selected measurements while fitting the model. This is easy to do in R – you just need to supply an additional variable to the model fitting function (using the **`weights`** argument). This variable includes weights of individual measurements. Weighting is suitable if individual measurements or observations are not of the same quality or if we know that the variance changes from one observation to another. For example, some measurements might come from a more accurate device than the others, or they can be from a more or less reliable observer. In order to balance the effect of such measurements, we assign a greater weight to more reliable measurements. Although conceptually simple, the main obstacle is that it is quite rare to have precise information about the relative precision of different observations (needed for the construction of weights). We should also use different weights if the measurements represent averages from various numbers of observations. It is obvious that results from a greater number of observations will be more exact – they will have smaller dispersion. In this context, the construction of weights is easy, unambiguous and straightforward. As we know from the previous chapters (Chapter 3.2), variance of the mean decreases with sample size according to the following formula:

$$\frac{\sigma^2}{n}. \tag{8-4}$$

It is then clear that individual weights should be proportional to the reciprocal variance. They can be equal either to the sample sizes from which the individual means were computed, or they could be proportional to the sample sizes (with a constant proportionality factor). Obviously, the weights must not be negative.

Should we specify weight values that are not identical for all observations, the method of weighted least squares is used instead of the standard method of least squares in the general linear model (**`lm`**). This is called weighted regression. It minimises the residual sum of the squares in such a way that measurements with a greater weight (lower expected variance) pull the model closer to themselves at the expense of the fit to the measurements with a smaller weight.

EXAMPLE

Sexual size dimorphism in spiders may increase with ambient temperature. Males and females of one spider species were sampled at 13 sites with a different average air temperature [°C] (*temp*). In each individual, the size of the prosoma [mm] was measured. For each site, size ratio (*ratio*) was calculated using the average prosoma size of males and females. The number of individuals (*number*) used to estimate the ratio varied among sites (2 to 62 individuals). We are interested in the relationship between size ratio and temperature. Specifi-

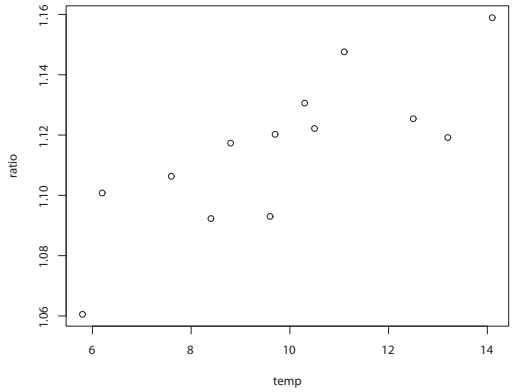


Fig. 8-4 Relationship between size *ratio* and temperature (*temp*).

cally: (1) Is there a relationship between the ratio and the temperature? (2) If so, what is the model of the relationship?

EDA

Once again, we have only one continuous explanatory variable, and we will thus draw a scatter plot of *ratio* versus temperature (*temp*).

```
> dat<-read.delim("zodarion.txt"); attach(dat); dat
   temp    ratio number
1   5.8 1.060554      3
2   6.2 1.100800      7
3   7.6 1.106300     22
...
13 14.1 1.158930      2
> plot(temp,ratio)
```

It is clear from the plot (Fig. 8-4) that the size *ratio* increases with increasing temperature (*temp*). The relationship appears to be linear.

MODEL

A previous study (Li 2002) showed that the body size of spiders changes linearly with temperature (within the range of biologically relevant temperatures). To start with, we will (naively) pay no attention to the different accuracies of the observations from different locations. We will thus assume a homoscedastic model (with constant variance) of the relationship between the body size and temperature for the i^{th} measurement:

$$ratio_i = \alpha + \beta temp_i + \varepsilon_i\,, \tag{8-5}$$

where $\varepsilon_i \sim N(0,\sigma^2)$, independent among sites.

ANALYSIS

Even though we expect a linear relationship (based on the explorative analysis as well as the theoretical model), we will, just to be sure, build the initial model with a quadratic term, to have some chance of detecting curvature in the relationship:

$$ratio_i = \alpha + \beta temp_i + \gamma temp_i^2 + \varepsilon_i, \quad \text{where } \varepsilon_i \sim N(0,\sigma^2), \text{ independent among sites.} \tag{8-6}$$

```
> m1<-lm(ratio~poly(temp,2))
> summary(m1)
...
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     1.114961   0.004597 242.538  < 2e-16 ***
poly(temp, 2)1  0.069484   0.016575   4.192  0.00185 **
poly(temp, 2)2 -0.010728   0.016575  -0.647  0.53206
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01657 on 10 degrees of freedom
Multiple R-Squared: 0.6428,     Adjusted R-squared: 0.5713
F-statistic: 8.996 on 2 and 10 DF,  p-value: 0.005818
```

The effect of the temperature is significant. However, the value of the quadratic term is not significantly different from *0*. This suggests that (as supposed originally), the relationship might be reasonably linear and hence the quadratic term is actually not needed in the model. So we will simplify `m1` to get a model with two parameters only.

```
> m2<-lm(ratio~temp)
> summary(m2)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.036897   0.018667  55.547 7.94e-15 ***
temp        0.007941   0.001843   4.307  0.00124 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01613 on 11 degrees of freedom
Multiple R-Squared: 0.6278,     Adjusted R-squared: 0.594
F-statistic: 18.55 on 1 and 11 DF,  p-value: 0.001240
```

In the purely linear model `m2`, the relationship between the size *ratio* and temperature is significantly positive. The intercept and the slope are significantly different from zero.

Let us think critically about the quality and credibility of such a model once more. We know that our samples were not always of the same size. Common sense then suggests that means computed from larger samples should provide more reliable information than means computed from smaller samples. That is why we should weight the larger samples more when fitting the model. They should have a greater impact on the estimation of model parameters. How much greater? The answer is simple, the weights should be based on the formula (8-4), which

we discussed right at the beginning of this chapter. In other words, we should fit a somewhat more sophisticated (but, most of all, more realistic) model using a non-constant variance:

$$ratio_i = \alpha + \beta temp_i + \varepsilon_i, \qquad (8\text{-}7)$$
$$\varepsilon_i \sim N\left(0, \frac{\sigma^2}{number}\right), \text{ independent among sites.}$$

How do we do that? Easily. We just insert argument **`weights`** when estimating the new model `m3`. Through the **`weights`** argument, we then supply a variable containing the individual sample sizes (*number*).

```
> m3<-update(m2,weights=number)
> summary(m3)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.084297   0.015481  70.038 6.24e-16 ***
temp        0.003265   0.001510   2.162   0.0535 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.04727 on 11 degrees of freedom
Multiple R-Squared: 0.2982,     Adjusted R-squared: 0.2345
F-statistic: 4.675 on 1 and 11 DF,  p-value: 0.0535
```

Note the dramatic change of the results. In the weighted model, the slope is not significant at the 5% level any more (despite the fact that the p-value is just above the conventional level, 0.0535)! The weighted analysis does not suggest that the size ratio increases linearly with temperature. That is a completely different result than in the case of the previous model! Only the value of the intercept is significantly different from zero. However, that value is of no importance for the tested hypothesis.

Why has such a significant change occurred? If you thoroughly look at the data, you will find that two extreme (and hence very influential) points, corresponding to the smallest and the biggest ratios, were obtained from only 3 and 2 specimens respectively. In model `m2` (which *did not know* about their unreliability), these points got a lot of (undeserved) influence and pulled the resulting line to themselves. After weighting, their undue influence was suppressed.

By drawing both model lines using the **`abline`** function in the original plot, we will see that the difference in their fits is really striking (Fig. 8-5).

```
> plot(temp,ratio,xlab="Temperature",ylab="Size ratio")
> abline(m2)
> abline(m3,lty=2)
> legend(6,1.15,c("m2","m3"),lty=1:2)
```

Which model we should rely on: `m2` or `m3`? If we do not doubt accuracy of the weighting conducted based on the sample size, we should believe model `m3`, as model `m2` weighted all

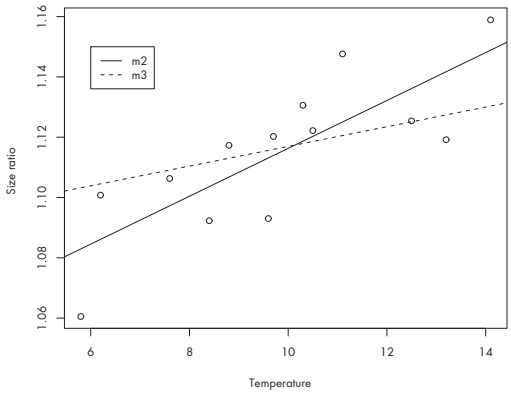


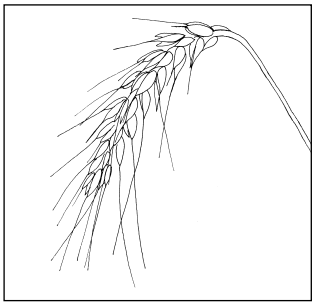
Fig. 8-5 Models of relationship between size *ratio* and temperature without weighting (`m2`) and with weighting (`m3`).

the data uniformly. Equal weighting might seem fair, however, this is really a false impression since the data collection led, in fact, to an unequal amount of information in different samples. The saying "All that glitters is not gold!" is valid here too. It is worth thinking about your data from the beginning to the end (and also about how they have been obtained). There is another thing that becomes apparent here: it is very easy to be misled (and to mislead others) by the statistics. Honest researchers should think and check hard to see if the results can be challenged (for example, if a given significance could be the result of an incorrect model, incorrect perception of the data collection mechanism, etc.), despite the fact that they are often seeking significance.

CONCLUSION

The attempt to confirm a linear increase in the body size ratio between genders with temperature for the spider species has not been successful (LM, $F_{1,11} = 4.7$, $p = 0.054$). To check such a relationship more thoroughly, we would need to collect and measure more specimens, especially from the sites with a low and high temperatures (where we now have only a few measurements).

8.4 Multiple regression



Yield of cereals is determined by a number of variables. On 100 plots, the yield of wheat [t/ha] (*yield*) was estimated at harvest. During the season, the following six variables were measured: (1) mean number of overwintering plants *per* m^2 (*winter*), (2) mean number of ears *per* m^2 (*ears*), (3) pH of soil (*pH*), (4) content of phosphorus [mg/kg of soil] (*P*), (5) content of potassium [mg/kg of soil] (*K*), and (6)

content of magnesium [mg/kg of soil] (*Mg*). We are interested in predicting yield with high accuracy using the available covariates. Specifically, we want to know: (1) Do any of the six variables affect the yield? (2) If so, what is the model for prediction of yield?

EDA

All six explanatory variables are continuous. To demonstrate their relationships, we could create individual scatter plots. We can efficiently do this using the **pairs** command. It will create a "matrix" of plots for all possible pairs of different variables. Here, we have seven variables to consider (six explanatory variables + one response). Using the **panel=panel.smooth** argument, we make sure that a loess curve is inserted into each of these plots. The curve describes a possible trend in a non-parametric way. Loess is a very flexible smoother (Cleveland & Devlin 1988). Here, it is not intended for formal analyses, but just for orientation in the data and for visual inspection of various systematic trends in the plots.

```
> dat<-read.delim("wheat.txt"); attach(dat); dat
    yield winter ears  pH   P   K Mg
1   7.060    235  628 7.0 118 222 62
2   7.094    307  764 6.9 117 114 47
3   7.171    213  636 6.0  60 127 46
...
100 9.738    320  564 6.1  84  88 44
> pairs(yield~winter+ears+pH+P+K+Mg,panel=panel.smooth)
```

The first row includes plots showing the response variable *yield* against various explanatory variables (Fig. 8-6). The relationships of *yield* to *winter*, *ears* and *Mg* are clearly positive. The relationships with *pH* and *P* are somewhat curved. Furthermore, we have to explore mutual relationships between all explanatory variables. They should not be "too strong". We know from the basic theory of linear regression that when there is a complete linear relationship between explanatory variables (i.e. a linear relationship between two and more variables), the model is singular and thus unidentifiable (its parameters cannot be unambiguously estimated). When the explanatory variables are *only* strongly correlated, instead of completely collinear, the model can be still identified; nevertheless, the model results are unstable and badly conditioned from both numerical and statistical perspectives. The ideal would be to have an orthogonal design plan. Statistical characteristics of such a design are in many ways the best possible. This is usually an impossible requirement for observational studies, such as this one. Apart from a relatively strong linear relationship between *pH* and *P* (as a result of the used alkali phosphorous fertiliser), the relationships among the other pairs are not too bad. If we want to proceed correctly with the process of examining collinearity of explanatory variables, we should do much more. As we have already mentioned above, we need to explore not only the relationships among pairs of variables that we can see on the picture, but also more complex relationships involving combinations of more than two variables. For this purpose, we can use the principle component analysis (routine check of the eigenvalues and eigenvectors of the $X'X$ matrix is often useful). We will not conduct it here, however. Interested readers can compute the principal component analysis of the $X'X$ matrix derived from the design matrix X by themselves using the **princomp** function.

MODEL

Even though the influence of the study variables on the *yield* of wheat has been studied many times in the past, we will pretend here that we do not have any information about their influence that would guide the model building. We do this to show you how to proceed when searching for a model in situations such that researches face when studying a new or scarcely explored phenomenon. For example, in ecology, such situations occur more often than not. We will assume that the yield is influenced by the study variables in a linear way. The model for the i^{th} measurement shall have the following form:

$$yield_i = \alpha + \beta_1 winter_i + \beta_2 ears_i + \beta_3 pH_i + \beta_4 P_i + \beta_5 K_i + \beta_6 Mg_i + \varepsilon_i, \qquad (8\text{-}8)$$
$$\varepsilon_i \sim N(0,\sigma^2), \text{ independent among plots.}$$

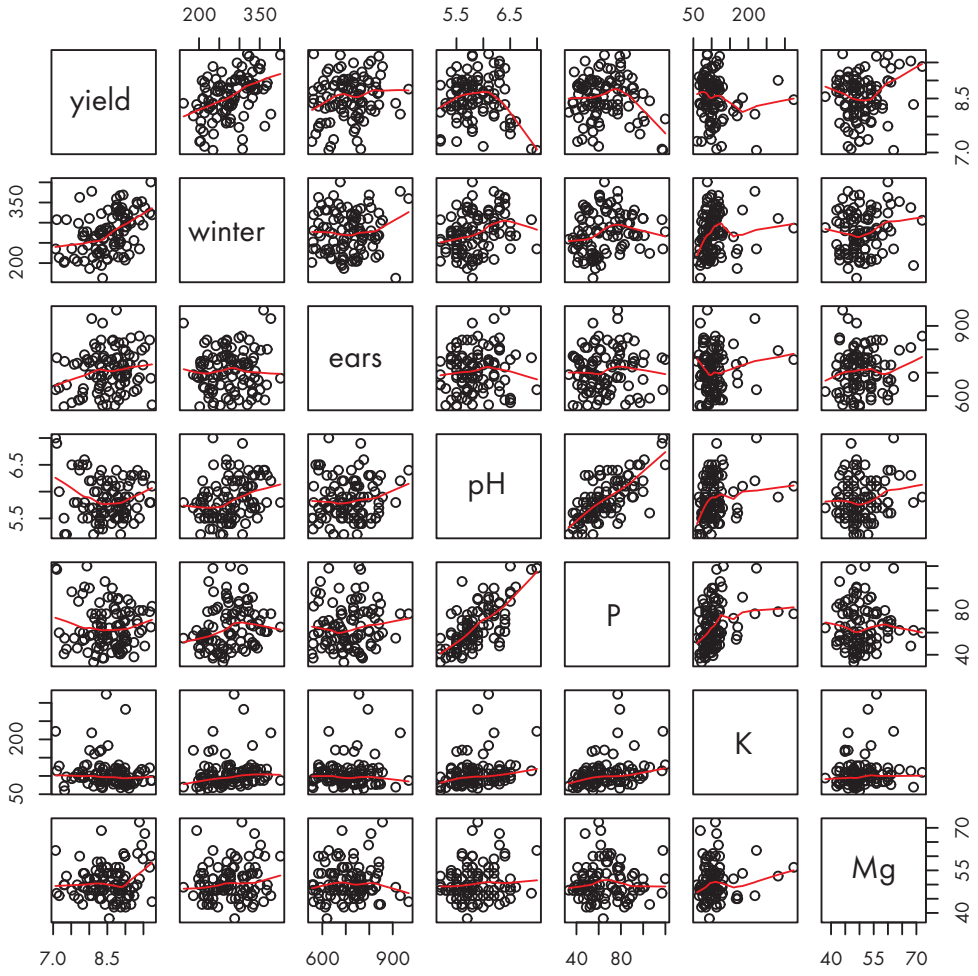


Fig. 8-6 Matrix of scatter plots showing relationships among all available variables: response variable (*yield*) and explanatory variables (*winter*, *ears*, *pH*, *P*, *K*, and *Mg*).

ANALYSIS

The explorative analysis has shown a relatively strong (however, not completely catastrophic) relationship between the explanatory variables *pH* and *P*. We will now naively pretend that we do not know about it (in order to be able to point out problems related to it later) and we will include all the explanatory variables that we have available (including *pH* and *P*) in the model. This is a typical approach of a beginner data analyst. A more experienced person might inspect the design matrix carefully first (as suggested in the EDA paragraph above) and, after detecting collinearity problems, he/she would try to use this information, for example, by leaving some of the variables out of the model, or by orthogonalising the design matrix (using principal components scores or otherwise).

If we want to be really careful and start with a large model (in order not to leave out something important, for example, interaction components), we could consider expanding the initial model by inclusion all two-, three-, four-, five- and six-way interactions. Let us note here that an interaction in this case will mean the product of two, three and more variables (formally, this would be just a part of a saturated interaction in the ANOVA sense). Moreover, we may also want to insert quadratic (as well as higher polynomial) terms for all variables in order to test if the given relationship does not show a significant curvature. This would amount to 70 or more parameters – and the model would still be relatively simple. As a matter of principle, we can think of many other additional non-linear, higher order polynomial or other than polynomial terms. The number of parameters, however, must be smaller than the sample size (n) in order to be identifiable. To get reasonable parameter estimates, it should be actually much smaller. How much smaller the model should be, depends on the structure of the data. According to a rather liberal suggestion it should be less than $n/3$ (Crawley 2002), while according to a more conservative rule, it should be less than $n/10$ (Faraway 2004). Should we use the less conservative one, we should not use more than 30 model parameters (since we have 100 measurements). In reality, we should be actually much more careful since collinearity and other problems increase *very quickly* as we start to add interactions or power terms. That is why, in this case, we will not include more complex components than two-way interactions together with the linear and quadratic terms in the explanatory variables. Such an approach is typical for practical data analysis. It is the "art of the possible": on one hand, we would like to have a *big* model that would include many complicated components not to miss something. On the other hand, we are limited by what the data (their size, configuration of the explanatory variable combinations, etc.) really allows. The result is thus a compromise. Nevertheless, we should take care to avoid a situation where "we cannot see the forest for the trees." In other words, the model should describe important features of the study problem and simplify the details that are not too important (and not the other way around).

When specifying a linear predictor, the order, in which terms are included in the model, is often important. This is especially true for situations where we try to determine whether a particular explanatory variable is important by using sequential testing. If we are concerned only with the quality of the overall fit and prediction, then the order is irrelevant. First of all, we should include the most important variables, followed by less important ones. However, which are important? We do not know it in this case, so we will include them (somewhat haphazardly) in an arbitrary order.

The first model will include all the main effects (seven parameters = six linear terms + one intercept), all quadratic terms (six parameters), and all two-way interactions (15 parameters). Altogether, it is 28 parameters. This time, we will not use the **poly** function for quadratic terms. Instead, we will use an alternative parametrization and write it into the model formula by means of the **I** function. This way, the model formula will be longer. It may not even fit onto a single line. Therefore, at the end of incomplete line press ENTER and continue (following a + sign generated by the program) on another line. To get an output we call **anova**.

```
> m1<-lm(yield~(winter+ears+pH+P+K+Mg)^2+I(winter^2)+I(ears^2)+I(pH^2)+
+ I(P^2)+I(K^2)+I(Mg^2))
> anova(m1)

Analysis of Variance Table

Response: yield
             Df  Sum Sq Mean Sq F value    Pr(>F)
winter        1  6.5802  6.5802 32.3949 2.542e-07 ***
ears          1  0.7288  0.7288  3.5882  0.062208 .
pH            1  2.1735  2.1735 10.7005  0.001646 **
P             1  0.0964  0.0964  0.4748  0.492988
K             1  0.9223  0.9223  4.5407  0.036513 *
Mg            1  0.4403  0.4403  2.1675  0.145310
I(winter^2)   1  0.0232  0.0232  0.1140  0.736647
I(ears^2)     1  0.2189  0.2189  1.0776  0.302707
I(pH^2)       1  5.2872  5.2872 26.0295 2.629e-06 ***
I(P^2)        1  0.0390  0.0390  0.1919  0.662635
I(K^2)        1  0.6269  0.6269  3.0861  0.083213 .
I(Mg^2)       1  0.7946  0.7946  3.9120  0.051770 .
winter:ears   1  0.1109  0.1109  0.5461  0.462318
winter:pH     1  0.0001  0.0001  0.0005  0.981871
winter:P      1  0.3435  0.3435  1.6911  0.197610
winter:K      1  0.2448  0.2448  1.2051  0.275966
winter:Mg     1  0.1437  0.1437  0.7072  0.403144
ears:pH       1  0.1750  0.1750  0.8616  0.356386
ears:P        1  0.3084  0.3084  1.5182  0.221900
ears:K        1  0.0010  0.0010  0.0048  0.945214
ears:Mg       1  0.0699  0.0699  0.3442  0.559238
pH:P          1  0.0076  0.0076  0.0372  0.847532
pH:K          1  0.0695  0.0695  0.3421  0.560434
pH:Mg         1  0.4321  0.4321  2.1272  0.149055
P:K           1  0.6919  0.6919  3.4061  0.069067 .
P:Mg          1  0.0921  0.0921  0.4535  0.502830
K:Mg          1  0.3609  0.3609  1.7766  0.186774
Residuals    72 14.6249  0.2031
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the ANOVA table, there are 27 (not 28) terms. Intercept is not included in the ANOVA table because its value is not tested here. The first six terms in the ANOVA table are linear. They are followed by six quadratic terms, and 15 two-way interactions. Most of the model terms are not significant. In particular, no interaction is significant. That leaves room for simplification. We can do the simplification using one of two strategies. Preferably, we should

gather all knowledge about the data and the processes behind them and use an expert assessment to decide which terms are important. This will take some time, but the results will be of higher quality and probably much more useful than those obtained by the other strategy. In situations when only very little is known about the processes and we have no idea which variables are more or less important, we can adopt a different strategy. We use an automatic model selection procedure (it is easy to use, but the results are, of course, of lower quality, in particular they can be quite dangerously misleading). In R, we can do an automatic model selection, for instance, by employing one of the following two methods:

- Akaike information criterion (AIC)
- F-test (it is necessary to select a level of significance, α, for this method)

The quality of the model obtained by automatic selection is typically lower (sometimes dramatically lower) than for a model simplified by an expert.

Now we will use the other strategy just to demonstrate it, but not to universally endorse it. We emphasise that such a strategy should be used only when expert knowledge is not available! We are using it now because we are pretending that nothing is known about the effect of the study variables on the yield. We use the automatic stepwise selection, function **`step`**, which uses AIC (Rawlings 1988). This function removes terms in sequence in order to minimise the value of AIC (corresponding to the simple rule: the lower AIC, the better model). It will select a model, which better fits the data but which is not too complex (not too large). This is because AIC relies on both model fit and model size penalties. The function **`step`** will remove only such terms that will not cause a marked change in AIC.

The output of the selection is very long (spanning over several pages), so we omit it here and we directly look at the ANOVA table of the final model `m2`.

```
> m2<-step(m1)
...
> anova(m2)
Analysis of Variance Table

Response: yield
          Df  Sum Sq Mean Sq F value    Pr(>F)
winter     1  6.5802  6.5802 34.6755 6.454e-08 ***
ears       1  0.7288  0.7288  3.8408  0.053080 .
pH         1  2.1735  2.1735 11.4538  0.001054 **
K          1  1.0085  1.0085  5.3143  0.023425 *
Mg         1  0.4506  0.4506  2.3744  0.126809
I(pH^2)    1  5.1106  5.1106 26.9315 1.272e-06 ***
I(Mg^2)    1  0.6744  0.6744  3.5541  0.062591 .
K:Mg       1  1.6123  1.6123  8.4961  0.004480 **
Residuals 91 17.2685  0.1898
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Many terms have been removed from the initial model: all interactions but one, all quadratic terms except those for *pH* and *Mg*, and the linear term for *P*.

We should mention the collinearity between *pH* and *P*, which we determined in the beginning. One of the consequences is the fact that it is relatively difficult to differentiate the effects of both variables. Thus, (nearly) collinear variables can mask each other in the sense that if one is selected to be in the model, the other(s) collinear with it, will not be selected by the automatic procedure. It depends on the order in which candidate variables appear in the model formula which of the two (or more) collinear variables will be fitted first. The first listed variable will have much better chance of being entered in the model, thus masking the others and effectively preventing them from entering subsequently. In reality, the effect of both variables can be significant. For clear separation of their effects, we would need better data – for example, from an experimental study that includes combinations of *pH* and *P*, which are missing in the current study. Remember that collinearity can have much more bizarre consequences than those we will see here. It can lead to instability. Even more paradoxically, all collinear variables can be detected as important (significant), while their coefficients can be completely unrealistic and non-interpretable (for example, having opposite signs to those that we would expect based on expert considerations).

Before we continue with simplification, we will look at the estimates of correlations among coefficients. If the design matrix was orthogonal, the correlation among estimates of different coefficients would be zero. Due to the non-orthogonality, we see some nonzero correlation values. Depending on the linear relationships among the explanatory variables, they can be quite substantial. We get the correlation matrix using the **`summary`** function with the **`corr`** argument.

```
> summary(m2,corr=T)
...
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.448e+01  9.131e+00  -3.776 0.000284 ***
winter       5.402e-03  9.132e-04   5.916 5.72e-08 ***
ears         7.469e-04  5.018e-04   1.488 0.140106
pH           1.662e+01  3.017e+00   5.510 3.31e-07 ***
K           -3.624e-02  1.167e-02  -3.106 0.002531 **
Mg          -2.050e-01  8.633e-02  -2.374 0.019676 *
I(pH^2)     -1.429e+00  2.529e-01  -5.650 1.82e-07 ***
I(Mg^2)      1.388e-03  7.799e-04   1.780 0.078428 .
K:Mg         6.419e-04  2.202e-04   2.915 0.004480 **
...

Residual standard error: 0.4356 on 91 degrees of freedom
Multiple R-Squared: 0.515,      Adjusted R-squared: 0.4724
F-statistic: 12.08 on 8 and 91 DF,  p-value: 1.300e-11

Correlation of Coefficients:
        (Intercept) winter ears  pH    K     Mg    I(pH^2) I(Mg^2)
winter   0.07
ears     0.07        0.00
pH      -0.96       -0.10  -0.12
K        0.17       -0.02   0.02 -0.30
Mg      -0.29        0.03   0.06  0.02  0.22
I(pH^2)  0.96        0.09   0.12 -1.00  0.30 -0.01
I(Mg^2)  0.34       -0.04  -0.06 -0.10  0.05 -0.96  0.10
K:Mg    -0.17        0.01  -0.03  0.30 -1.00 -0.23 -0.31   -0.04
```

The table of coefficients is similar to what we saw in the previous chapter. On the first line there is an estimate of the intercept (a). Then there are five estimates of the linear coefficients (b_1, b_2, b_3, b_4, b_5), two estimated quadratic coefficients (c_1, c_2), and an estimate of one interaction effect (d). Recall that lines are denoted by the names of explanatory variables (intercept corresponds to the coefficient of an explanatory variable composed only of 1's). Overall F-test evaluates the quality of the total model against a null model (including only the intercept). Typically, it is not an interesting test because the model was built to explain variability in the data by means of preselected explanatory variables, so it is not really surprising if it is significant. If it was not significant, it would indicate absence of any predictive power for the covariates used. On the other hand, a significant outcome here does not necessarily mean that the model is good!

Further, there are estimates of correlations (Correlation of Coefficients) among coefficient estimates (i.e. among *random* variables a, b_1, b_2, b_3, b_4, b_5, c_1, c_2, and d) constructed from the data in order to estimate the unknown fixed coefficients α, β_1, ...δ. The correlations can be calculated using the basic theory of the least square method and the underlying statistical model. You may have heard that correlation among parameter estimates is related to the correlation among the explanatory variables (and to the problem of collinearity). Ideally, estimates of coefficients should be un-correlated (orthogonal), then their interpretation is simple – each estimate can be interpreted separately. Here, we are confronted with a less ideal case, often encountered in practice. We can see almost perfect "textbook" collinearity between the linear and quadratic terms for *pH*. In the output, their correlation is "perfect" –1.00. (Such a value cannot be exact if the model is not identifiable. It occurred only due to rounding.) There is also a collinearity between the interaction effect estimate and the linear terms. Taken all together, it is clear that this is not a good model!!! It is particularly dangerous when we try to report and interpret the coefficient estimates. Estimates of (correlated) coefficients are unstable, as with even a very small error in the data, their estimates can change dramatically. Truly, this is less important as we are interested in the overall prediction only (without looking at the detailed structure of the model and its coefficients), but, the (severe) correlation among the coefficient estimates can easily mislead the model selection procedure. What should we do now?

One option to get rid of strong correlations with the intercept (but also with quadratic coefficients) is a standardisation of explanatory variables (to get zero expected value and identity variance), for example, by calling **`scale`** (beware: this will also affect the interpretation of estimated coefficients). Then we will fit a model with a similar systematic part as in `m1`, but with standardised variables, to which we give the following names: *w1*, *e1*, *pH1*, *P1*, *K1*, and *Mg1*. Such a step is relatively innocuous because we are interested only in the prediction of a model.

```
> w1<-scale(winter); e1<-scale(ears); pH1<-scale(pH)
> P1<-scale(P); K1<-scale(K); Mg1<-scale(Mg)
> m3<-lm(yield~(w1+e1+pH1+P1+K1+Mg1)^2+I(w1^2)+I(e1^2)+I(pH1^2)+
+ I(P1^2)+I(K1^2)+I(Mg1^2))
```

```
> anova(m3)
Analysis of Variance Table

Response: yield
          Df  Sum Sq Mean Sq F value    Pr(>F)
w1         1  6.5802  6.5802 32.3949 2.542e-07 ***
e1         1  0.7288  0.7288  3.5882  0.062208 .
pH1        1  2.1735  2.1735 10.7005  0.001646 **
P1         1  0.0964  0.0964  0.4748  0.492988
K1         1  0.9223  0.9223  4.5407  0.036513 *
Mg1        1  0.4403  0.4403  2.1675  0.145310
I(w1^2)    1  0.0232  0.0232  0.1140  0.736647
I(e1^2)    1  0.2189  0.2189  1.0776  0.302707
I(pH1^2)   1  5.2872  5.2872 26.0295 2.629e-06 ***
I(P1^2)    1  0.0390  0.0390  0.1919  0.662635
I(K1^2)    1  0.6269  0.6269  3.0861  0.083213 .
I(Mg1^2)   1  0.7946  0.7946  3.9120  0.051770 .
w1:e1      1  0.1109  0.1109  0.5461  0.462318
w1:pH1     1  0.0001  0.0001  0.0005  0.981871
w1:P1      1  0.3435  0.3435  1.6911  0.197610
w1:K1      1  0.2448  0.2448  1.2051  0.275966
w1:Mg1     1  0.1437  0.1437  0.7072  0.403144
e1:pH1     1  0.1750  0.1750  0.8616  0.356386
e1:P1      1  0.3084  0.3084  1.5182  0.221900
e1:K1      1  0.0010  0.0010  0.0048  0.945214
e1:Mg1     1  0.0699  0.0699  0.3442  0.559238
pH1:P1     1  0.0076  0.0076  0.0372  0.847532
pH1:K1     1  0.0695  0.0695  0.3421  0.560434
pH1:Mg1    1  0.4321  0.4321  2.1272  0.149055
P1:K1      1  0.6919  0.6919  3.4061  0.069067 .
P1:Mg1     1  0.0921  0.0921  0.4535  0.502830
K1:Mg1     1  0.3609  0.3609  1.7766  0.186774
Residuals 72 14.6249  0.2031
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In model `m3`, there are four terms significant at 5 % level. Non-significant terms will be removed sequentially one after another, based on the test statistics. First, we will remove interactions, then quadratic terms and finally linear terms starting with those having the largest p-value. Removal will be done this time using the function **`update`**. Every removal will be checked by displaying an ANOVA table of the new model, because p-values of terms remaining in the model will change during the process of simplification. At first we remove `w1:pH1`, then `e1:K1`, etc. To save space, we omit particular steps and we show only the final model (`m26`).

```
> m4<-update(m3,~.-w1:pH1); anova(m4)
> m5<-update(m4,~.-e1:K1); anova(m5)
...
> anova(m26)
Analysis of Variance Table

Response: yield
          Df  Sum Sq Mean Sq F value    Pr(>F)
w1         1  6.5802  6.5802 30.4123 3.001e-07 ***
pH1        1  1.9487  1.9487  9.0066  0.003436 **
```

```
K1          1  0.8711  0.8711  4.0262  0.047642 *
I(pH1^2)    1  5.6527  5.6527 26.1256 1.653e-06 ***
Residuals 95 20.5547  0.2164
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Only four significant terms have been left in model `m26`. We will thus pragmatically consider this model to be the final model (in reality, we should certainly test such an *optimised* model on new data). Let us repeat once again that by standardising the explanatory variables, we have markedly changed the form and interpretation of the model. The predicted model is now:

$$a + b_1 \frac{winter - \bar{y}_{winter}}{s_{winter}} + b_2 \frac{pH - \bar{y}_{pH}}{s_{pH}} + c_1 \left(\frac{pH - \bar{y}_{pH}}{s_{pH}} \right)^2 + b_3 \frac{K - \bar{y}_K}{s_K}, \qquad (8\text{-}9)$$

in which the values of averages ($\bar{y}$) and standard deviations (s) of individual explanatory variables can be found out by applying the **`mean`** and **`sd`** functions on a particular explanatory variable, i.e.:

```
> mean(winter);sd(winter)
[1] 275.64
[1] 50.94392
> mean(pH);sd(pH)
[1] 5.852
[1] 0.3812473
> mean(K);sd(K)
[1] 106.66
[1] 40.39657
```

We can determine estimates of parameters a, b_1, b_2, b_3 and c_1 from the table of coefficients. None of the correlations between individual parameter estimates is alarming in this model.

```
> summary(m26,corr=T)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.71416    0.05917 147.278  < 2e-16 ***
w1           0.28494    0.04941   5.766 1.00e-07 ***
pH1         -0.01134    0.05490  -0.206   0.8368
K1          -0.09666    0.04885  -1.979   0.0508 .
I(pH1^2)    -0.18880    0.03694  -5.111 1.65e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4652 on 95 degrees of freedom
Multiple R-Squared: 0.4227,     Adjusted R-squared: 0.3984
F-statistic: 17.39 on 4 and 95 DF,  p-value: 9.75e-11

Correlation of Coefficients:
          (Intercept) w1    pH1   K1
w1        -0.06
pH1        0.25       -0.28
K1         0.00       -0.10 -0.22
I(pH1^2) -0.62         0.10 -0.40 -0.01
```

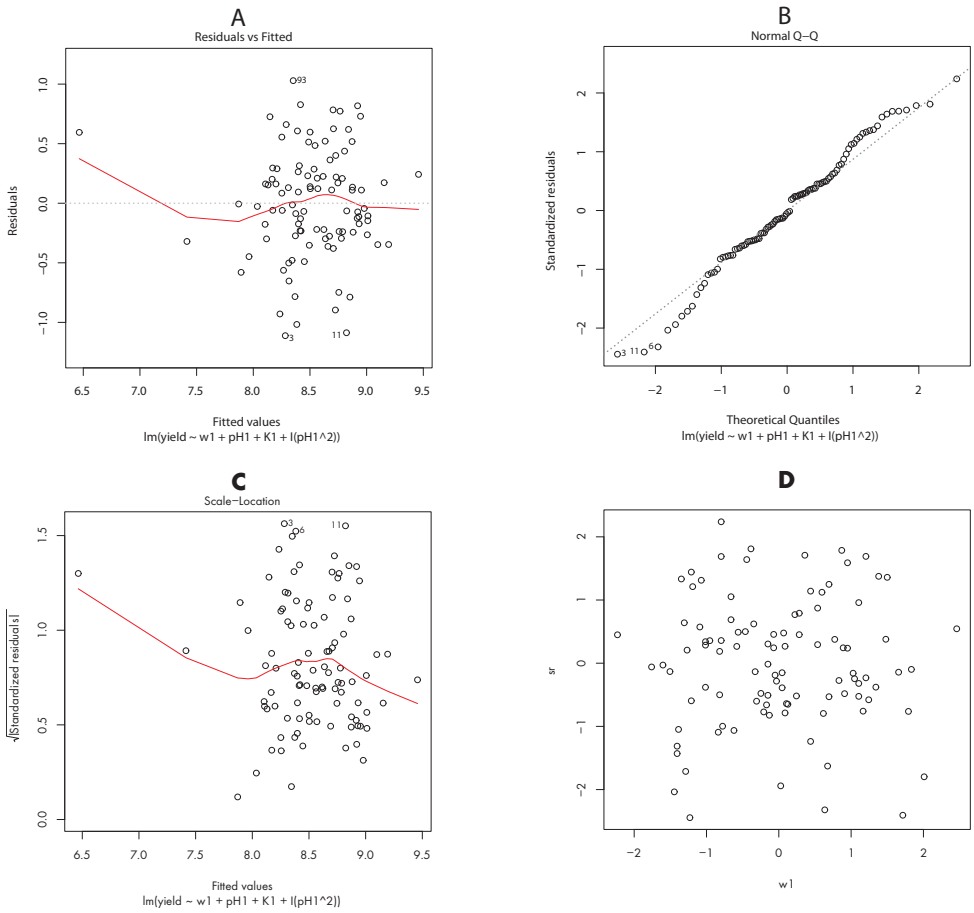


Fig. 8-7 A. Relationship between residuals and fitted values. **B**. Q-Q normal plot of standardised residuals. **C**. Relationship between the square root of standardised residuals and the fitted values. **D**. Relationship between standardised residuals and values of the explanatory variable `w1`.

The final model, `m26` is certainly not the only suitable model. Try to find a better one yourself!

Finally, we will quickly check some of the assumptions upon which the model is based via diagnostic plots produced by the **`plot`** function.

```
> par(mfrow=c(2,2))
> plot(m26,which=1:3)
```

The most important is Fig. 8-7A. We know from the regression theory that the vector of residuals must be orthogonal to the vector of predicted values. We should not be able to see any systematic features in the picture (trends, variance heterogeneity, etc.). Indeed, we can-

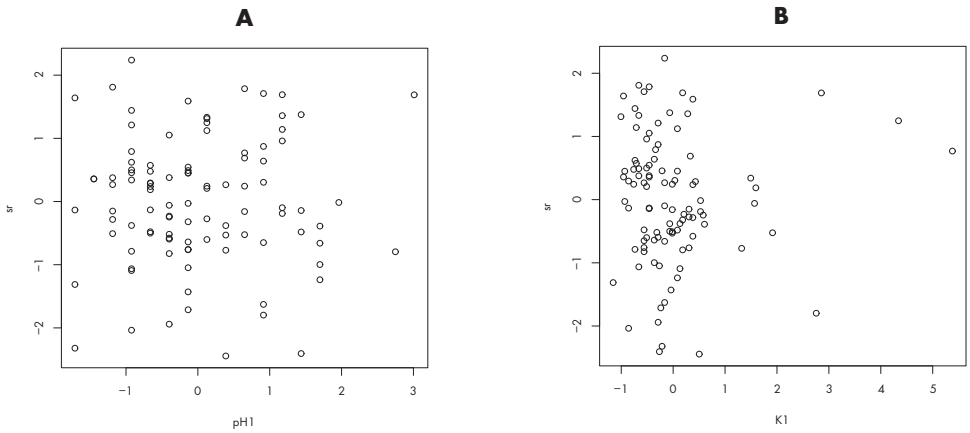


Fig. 8-8 Relationship between standardised residuals (`sr`) and explanatory variables: `pH1` (**A**), and `K1` (**B**).

not see anything like that in this figure – and that is good. The distribution of neither raw, nor standardised residuals shows any signs of variance heterogeneity (Fig. 8-7A, C). The Q-Q plot of the residuals (Fig. 8-7B) is not completely ideal but, on the other hand, it does not suggest any insurmountable problems with normality. Now we will explore how the residuals behave when plotted against the explanatory variables. Ideally, the influence of a given explanatory variables should be completely exhausted by the model, leaving no relationship between the residuals and explanatory variables. We will use standardised residuals (`sr`), and plot them against explanatory variables *w1*, *pH1*, and *K1*.

```
> sr<-rstandard(m26)
> plot(w1,sr)
> par(mfrow=c(2,1))
> plot(pH1,sr); plot(K1,sr)
```

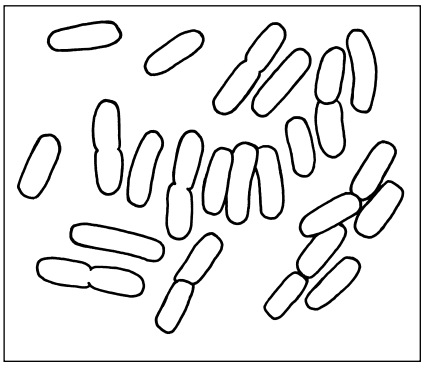
It does not look like the standardised residuals show any obvious trend in any of the cases (Fig. 8-7D, 8-8A–B). So far, we have not found any significant defect of model `m26`. Nevertheless, we should emphasise that our searching was very simple and short (illustrative). Should we be serious about the model, we would have to continue with the testing. However, even if we do more tests, we cannot ever expect that they would entirely confirm the model. The conducted checks can only disprove a model or "not disprove it for now". You should spent some time by thinking about how big the difference is between *confirming* and *not disproving*. This should help you to interpret and use modelling results (both yours and of others) with care and due respect.

CONCLUSION

A total of three study variables, overwintering, pH, and the content of potassium, had a significant effect on the wheat yield (LM, $F_{4,95} = 17.4$, $P < 0.0001$). A yield prediction can be obtained from the following formula:

$$8.714 + 0.285\frac{winter - 275.6}{50.94} - 0.011\frac{pH - 5.852}{0.381} - 0.189\left(\frac{pH - 5.852}{0.381}\right)^2 - 0.097\frac{K - 106.7}{40.39}$$

8.5 Two-way ANOVA



The occurrence of carcinoma might be related to the presence of certain pathogenic bacteria in the body of patients. In a clinical study, the amount of a toxin [units/µl] (*toxin*) produced by four bacteria species (*SPECIES*: bacterA, bacterB, bacterC, bacterD) was measured in the gastro-intestinal tract of patients. These patients were diagnosed with one of two carcinogenic or two non-carcinogenic diseases (*DIAGNOSIS*: carc. rectum, carc.intestine, appendicitis, skin.absces). For each disease, there were 20 patients. In each patient, only a single bacterial toxin was measured so there were five replications *per* bacteria species. Presence of toxins might be a potential indicator of certain carcinoma. So, we are interested in testing: (1) Is the amount of toxin similar for four bacteria species and four diseases? (2) If not, what is the difference? (3) Which species can be used as indicators?

EDA

To inspect the effect of both explanatory variables, *SPECIES* and *DIAGNOSIS*, on the amount of toxin we use the **`interaction.plot`**.

```
> dat<-read.delim("bacteria.txt"); attach(dat); dat
   species      diagnosis toxin
1  bacterA carc.intestine 1.735
2  bacterA carc.intestine 2.076
3  bacterA carc.intestine 1.648
...
80 bacterD    apendicitis 1.122
> interaction.plot(species,diagnosis,toxin)
```

Patients with carcinogenic diagnoses, "carc.rectum" or "carc.intestine", show greater values of toxins produced by any of the four bacteria species in comparison with patients with the non-carcinogenic diagnoses, "appendicitis" or "skin.absces" (Fig. 8-9). With regard to the differences among the four bacteria species, it appears that the amount of toxins was similar for species "bacteria" and "bacterD", and for species "bacterB" and "bacterC". Since the connecting lines are not parallel, we can expect that the *SPECIES*:*DIAGNOSIS* interaction will be significant.

MODEL

As the preliminary studies showed, the amount of toxins produced differs according to diagnosis. Moreover, the exploratory analysis revealed that the toxin production also differs

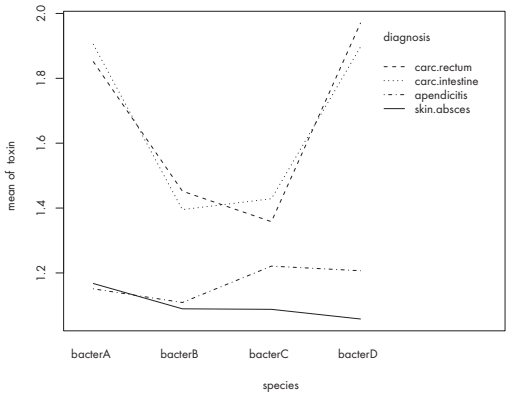


Fig. 8-9 Comparison of mean amount of *toxin* in each of four bacteria species and four diseases (*DIAGNOSIS*).

with bacteria species: different bacteria produced different amount of toxins for different diagnoses. In other words we expect the following model with interaction for the i^{th} measurement:

$$toxin_{ijk} = \alpha + SPECIES_j + DIAGNOSIS_k + SPECIES : DIAGNOSIS_{jk} + \varepsilon_{ijk}, \quad \text{where } \varepsilon_{ijk} \sim N(0,\sigma^2), \text{ independent among patients.} \tag{8-10}$$

This is a model of a two-way ANOVA with interaction as you may remember it from a biostatistical course. We will use treatment parametrization which means that the following restrictions are imposed on the model coefficients: $SPECIES_{bacterA} = 0$, $DIAGNOSIS_{apendicitis} = 0$, $SPECIES{:}DIAGNOSIS_{bacterA,k} = 0$ for any k, $SPECIES{:}DIAGNOSIS_{j\text{-}apendicitis} = 0$ for any j). So, α (intercept) is the mean amount of toxin for a combination of alphabetically first levels in each factor ("bacterA" with "carc.rectum"), $SPECIES_j$ is an effect of j^{th} level of the factor *SPECIES* (it corresponds to the difference between mean amount of toxin for a given species of bacteria and "bacterA"), $DIAGNOSIS_k$ is an effect of k^{th} level of the factor *DIAGNOSIS* (it is a difference between the mean amount of toxin between the given level of *DIAGNOSIS* and "carc.rectum"), $SPECIES{:}DIAGNOSIS_{jk}$ is an interaction effect for combinations of j^{th} species level and k^{th} diagnosis level.

ANALYSIS

The initial model will include two main effects and their interaction according to (8-10). We fit the model for the expected normal data distribution and we thus use the **`lm`** function:

```
> m1<-lm(toxin~species*diagnosis)
> anova(m1)
Analysis of Variance Table
```

```
Response: toxin
                 Df Sum Sq Mean Sq  F value    Pr(>F)
species           3 1.3364  0.4455  28.0325 1.077e-11 ***
diagnosis         3 5.4775  1.8258 114.8965 < 2.2e-16 ***
species:diagnosis 9 1.2704  0.1412   8.8827 1.528e-08 ***
Residuals        64 1.0170  0.0159
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The interaction is highly significant. This means that the initial model cannot be further simplified by removing any term. We will look at the table of coefficients with pre-set treatment contrasts in order to be able to determine the differences between individual level combinations.

```
> summary(m1)
...
Coefficients:
                                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                                1.15100    0.05638  20.417  < 2e-16
speciesbacterB                            -0.04220    0.07973  -0.529 0.598427
speciesbacterC                             0.07000    0.07973   0.878 0.383233
speciesbacterD                             0.05580    0.07973   0.700 0.486536
diagnosiscarc.intestine                    0.75400    0.07973   9.457 9.07e-14
diagnosiscarc.rectum                       0.70040    0.07973   8.785 1.34e-12
diagnosisskin.absces                       0.01640    0.07973   0.206 0.837678
speciesbacterB:diagnosiscarc.intestine    -0.46760    0.11275  -4.147 0.000101
speciesbacterC:diagnosiscarc.intestine    -0.54620    0.11275  -4.844 8.42e-06
speciesbacterD:diagnosiscarc.intestine    -0.06420    0.11275  -0.569 0.571083
speciesbacterB:diagnosiscarc.rectum       -0.35700    0.11275  -3.166 0.002366
speciesbacterC:diagnosiscarc.rectum       -0.56340    0.11275  -4.997 4.78e-06
speciesbacterD:diagnosiscarc.rectum        0.06300    0.11275   0.559 0.578282
speciesbacterB:diagnosisskin.absces       -0.03580    0.11275  -0.318 0.751889
speciesbacterC:diagnosisskin.absces       -0.14960    0.11275  -1.327 0.189287
speciesbacterD:diagnosisskin.absces       -0.16520    0.11275  -1.465 0.147771
...
```

The table is relatively complex since it contains 16 coefficients. This is because we had four levels for each factor (4 x 4 = *16*). The order of the coefficients in the table corresponds to their order in the model formula. If we switch the order (to **`diagnosis*species`**), the coefficient values would remain the same, but be placed in a different order.

Now we will explain the meaning of the coefficients. You are right to expect that this will not be entirely easy. To make it easier, we will calculate the mean values for all combinations of the levels estimated from model `m1`. This is because, if we have unbalanced data, these estimates can be better than the empirical means acquired from raw data. For this purpose, we use the **`tapply`** command with the **`predict`** argument.

```
> tapply(predict(m1),list(species,diagnosis),mean)
        apendicitis carc.intestine carc.rectum skin.absces
bacterA      1.1510         1.9050      1.8514      1.1674
bacterB      1.1088         1.3952      1.4522      1.0894
bacterC      1.2210         1.4288      1.3580      1.0878
bacterD      1.2068         1.8966      1.9702      1.0580
```

All estimates of expected values are also in the table of coefficients. We just need to put them together from the coefficients. The first four lines in the summary output are means for "appendicitis" (because it is the reference level for *DIAGNOSIS*) and its combination with all levels of the factor *SPECIES*. Specifically, Intercept represents the mean for the combination "appendicitis-bacterA" (*1.151*). In the second line of the table, with the name `speciesbacterB`, there is a difference for mean of "appendicitis-bacterB" from mean of "appendicitis-bacterA". Adding them together, we get a mean for "appendicitis-bacterB" (1.151 – 0.0422 = *1.1088*). Similarly, in the third and the fourth lines there are differences of means for "appendicitis-bacterC" and "appendicitis-bacterD" respectively, from the mean of the reference combination "appendicitis-bacterA". In lines five to seven there are differences of means for "bacterA" in combination with three levels of the factor *DIAGNOSIS* from mean of the reference combination. To get a mean for "bacterA-carc.intestine" we have to add: 1.151 + 0.754 = *1.905*. All lines below are named after the combination of factor names and their levels. Means of these combinations are computed in a more complicated way. We have to add the intercept to all coefficients from lines two to seven, which include name of the level we are interested in. For example, mean for "bacterB-carc.intestine" is obtained by adding the first (*1.151*), the second (*–0.0422*), the fifth (*0.754*), and the eighth (*–0.4676*) coefficients (= *1.3952*). The mean for "bacterD-carc.rectum" is obtained by adding the first (*1.151*), the fourth (*0.0558*), the sixth (*0.7004*), and the thirteenth (*0.063*) coefficients (= *1.9702*). And so on.

Let's have a look at the t-tests of parameters. What do they test? The first tests that the mean of the reference combination "appendicitis-bacterA" is different from 0. The result of this test is highly significant (p-value < 0.0001), but probably not interesting at all. More interesting are other tests which are related to comparisons of different levels. Further down the table, three tests compare means for other bacteria species combined with "appendicitis" against "bacterA". None of these is significant (p-values > 0.38). The fifth to seventh lines show tests of differences for other diagnosis levels combined with "bacterA" against "appendicitis". For example, "carc.rectum-bacterA" is significantly different from "appendicitis-bacterA" (p-value < 0.0001). Then there are tests of differences for combinations of all other bacteria species ("bacterB", "bacterC", "bacterD") and diagnoses ("carc.intestine", "carc.rectum", "skin.absces") against one of the 5th to 7th lines. There are really many tests with different interpretations.

There are non-significant test results in more than half of the rows suggesting that this model could be simplified by merging some of the levels. a difference between the carcinogenic and non-carcinogenic diagnoses was obvious already from the first plot (Fig. 8-9) and that is why we firstly combine similar diagnoses. We could gradually rewrite the levels of the *DIAGNOSIS* factor using the **`levels`** command. But we will do it more efficiently this time. If you look at the data frame by calling **`dat`**, you will see that the first 40 rows represent carcinogenic diagnoses and the second 40 rows non-carcinogenic diagnoses. By doing so, we create a new vector called *DIAGNOSIS1*, which will have two levels: "carc" for two carcinogenic and "non" for two non-carcinogenic diagnoses. Next, we have to make a factor from the vector (for example, of the same name) using the **`factor`** command.

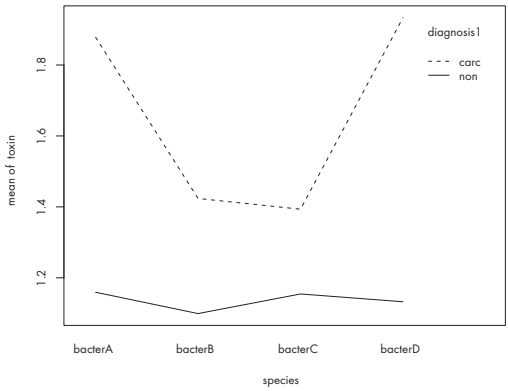


Fig. 8-10 Comparison of mean amount of *toxin* in four bacteria species and two groups of diseases (*DIAGNOSIS1*).

```
> diagnosis1<-c(rep("carc",40),rep("non",40))
> diagnosis1<-factor(diagnosis1)
```

Now we will fit a new model, in which we substitute *DIAGNOSIS* with the *DIAGNOSIS1*. We compare both models and draw the simplified data set in the interaction plot.

```
> m2<-lm(toxin~species*diagnosis1)
> anova(m1,m2)
Analysis of Variance Table

Model 1: toxin ~ species * diagnosis
Model 2: toxin ~ species * diagnosis1
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     64  1.0170
2     72  1.1597 -8   -0.1427 1.1225 0.3605
> interaction.plot(species,diagnosis1,toxin)
```

The simplification of *DIAGNOSIS* levels, has not caused a significant change in the residual sum of squares (p-value = 0.361) and that is why we accept the simplification. It shows that the main difference of the effect of the *DIAGNOSIS* in the amount of toxins is not based on a particular diagnosis but on the fact whether it is of the carcinogenic type or not. Moreover, we can see from Fig. 8-10 that the values for "bacterB" and "bacterC" are similar for both diagnosis types. The same applies to the values for "bacterA" and "bacterD". That is why we will try to combine them one after another. We will test each simplification using the **anova** command. Prior to combining, we create a copy of the *SPECIES*.

```
> species1<-species
> levels(species1)
[1] "bacterA" "bacterB" "bacterC" "bacterD"
> levels(species1)[2:3]<-"bacterBC"
> m3<-lm(toxin~species1*diagnosis1)
> anova(m2,m3)
```

```
Analysis of Variance Table

Model 1: toxin ~ species * diagnosis1
Model 2: toxin ~ species1 * diagnosis1
  Res.Df      RSS Df Sum of Sq      F Pr(>F)
1     72  1.15974
2     74  1.17962 -2  -0.01988 0.6171 0.5423
> levels(species1)
[1] "bacterA"  "bacterBC" "bacterD"
> levels(species1)[c(1,3)]<-"bacterAD"
> m4<-lm(toxin~species1*diagnosis1)
> anova(m3,m4)
Analysis of Variance Table

Model 1: toxin ~ species1 * diagnosis1
Model 2: toxin ~ species1 * diagnosis1
  Res.Df      RSS Df Sum of Sq      F Pr(>F)
1     74  1.17962
2     76  1.19845 -2  -0.01883 0.5905 0.5566
```

Neither of the executed combinations has caused a significant change. It means that the given pairs of bacteria species ("bacterA" with "bacterD", and "bacterB" with "bacterC") behave similarly. This can be a very useful information. For example, it could suggest that instead of complicated measurements of toxins for four bacteria species, we could monitor only two *indicator* species when conducting screening. Let us look at the ANOVA table and at the table of coefficient of the last model.

```
> anova(m4)
Analysis of Variance Table

Response: toxin
                   Df Sum Sq Mean Sq F value    Pr(>F)
species1            1 1.3328  1.3328  84.522 5.690e-14 ***
diagnosis1          1 5.4267  5.4267 344.139 < 2.2e-16 ***
species1:diagnosis1  1 1.1434  1.1434  72.508 1.134e-12 ***
Residuals           76 1.1984  0.0158
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> summary(m4)
...
Coefficients:
                              Estimate Std. Error t value Pr(>|t|)
(Intercept)                    1.90580    0.02808  67.872  < 2e-16 ***
species1bacterBC              -0.49725    0.03971 -12.522  < 2e-16 ***
diagnosis1non                 -0.76000    0.03971 -19.139  < 2e-16 ***
species1bacterBC:diagnosis1non  0.47820    0.05616   8.515 1.13e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1256 on 76 degrees of freedom
Multiple R-Squared: 0.8683,     Adjusted R-squared: 0.8631
F-statistic: 167.1 on 3 and 76 DF,  p-value: < 2.2e-16
```

Compared to the first model, we saved 12 degrees of freedom. The interaction is still significant. The number of levels in each factor have been reduced to two, thus in the table

of coefficients we will find four parameters only. Intercept is a mean for "bacterAD-carc". In the second line there is a difference for "bacterBC-carc" from the intercept. In the third line there is a difference for "bacterAD-non" from the intercept. In the fourth line there is a difference of "bacterAD-carc" + "bacterBC-carc' from "bacterAD-non' + "bacterBC-non". The mean for "bacterBC-non" is obtained by adding all coefficients (1.9058 – 0.49725 – 0.76 + 0.4782 = *1.127*).

All t-tests are highly significant, but the first one is not of interest as it is a comparison of the expected value for the combination of reference levels against zero (indeed, we expect that the toxin amount should be higher than zero). The second test shows that mean for "bacterBC-carc" is significantly lower than the mean for "bacterAD-carc", similarly the third one shows that mean for "bacterAD-non" is significantly lower than the mean for "bacterAD-carc". The last one confirms the significant effect of interaction – the effect of both factors is not additive (if it was, then the value of the coefficient would be close to 0).

It is clear from the output that there is nothing more to simplify. Model `m4` is thus our final model:

$$toxin_{ijk} = \alpha + SPECIES1_j + DIAGNOSIS1_k + SPECIES1{:}DIAGNOSIS1_{jk} + \varepsilon_{ijk}, \quad \text{(8-11)}$$
$$\text{where } \varepsilon_{ijk} \sim N(0,\sigma^2), \text{ independent among patients.}$$

It is formally identical to the first model (8-10), except for the fact that we have reduced the factor levels.

Next, we will look at the first two pre-set diagnostic plots:

```
> par(mfrow=c(1,2))
> plot(m4,which=1:2)
```

Note that the residuals have homogenous variance (Fig. 8-11A). Their distribution does not look abnormal either (Fig. 8-11B). This means that we do not see any serious disruptions of the assumptions for now. The reader can try to verify other assumptions that we have not explored so far (but were outlined in Chapter 4.6).

If we want to estimate 95% confidence intervals for the expected values, we can get them by applying the function **`confint`**. Model `m4` is in the treatment parametrization, however, to estimate the intervals for expected values we need to have the model in the textbook parametrization. So we have to re-parametrise the model. A simple way to do this is to create a new variable which is a combination of the two explanatory variables. It will have four levels. Using the function **`paste`** we will combine the two variables into one, named `both`. This will have four levels: "bacterAD carc", "bacterAD non", "bacterBC carc", and "bacterBC non". This factor will be used in a new model (`m5`) without an intercept, which is one way of obtaining the textbook parametrization.

```
> both<-paste(species1,diagnosis1)
> both<-factor(both)
```

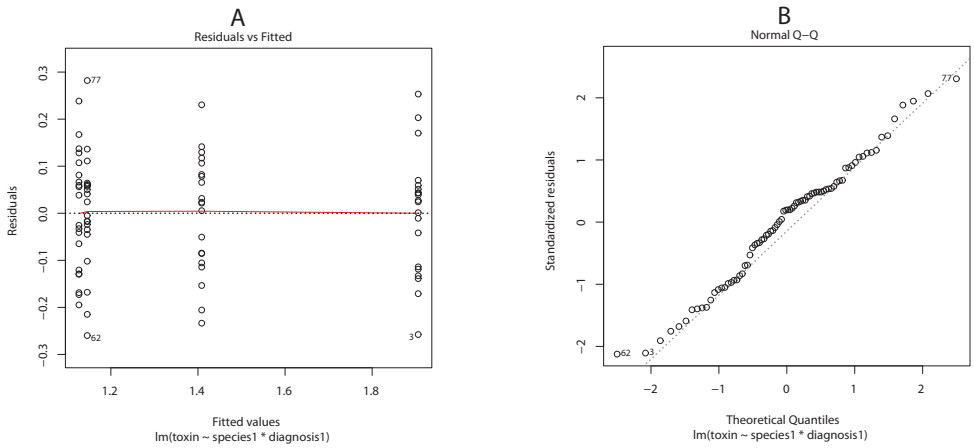


Fig. 8-11 A. Relationship between residuals and predicted values. **B**. Q-Q normal plot of standardised residuals.

```
> m5<-lm(toxin~both-1)
> summary(m5)
...
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
bothbacterAD carc  1.90580    0.02808   67.87   <2e-16 ***
bothbacterAD non   1.14580    0.02808   40.81   <2e-16 ***
bothbacterBC carc  1.40855    0.02808   50.16   <2e-16 ***
bothbacterBC non   1.12675    0.02808   40.13   <2e-16 ***
...
```

The table of coefficients now includes estimates of the expected values and their SE for all levels. Remember that the t-tests for this parametrization tests the differences of individual expected values from zero. We can get the confidence intervals by applying the **`confint`** function.

```
> confint(m5)
                        2.5 %    97.5 %
bothbacterAD carc 1.849875 1.961725
bothbacterAD non  1.089875 1.201725
bothbacterBC carc 1.352625 1.464475
bothbacterBC non  1.070825 1.182675
```

Of all CI_{95}, only intervals for two levels are overlapping, namely the ones for non-carcinogenic diagnoses. Estimated expected values can be plotted together with their confidence intervals. We will use the function **`interaction.plot`**, but we draw only points (without lines) using the argument **`type=p`**. Expected values are in this case identical to the empiric means, thus we plot the latter ones as it takes fewer commands. Confidence intervals will be drawn as vertical lines separately for each expected value using **`lines`** (Fig. 8-12):

```
> interaction.plot(species1,diagnosis1,toxin,type="p",
+ pch=1:2,ylim=c(1,2),ylab="Toxin amount",xlab="Species",legend=F)
```

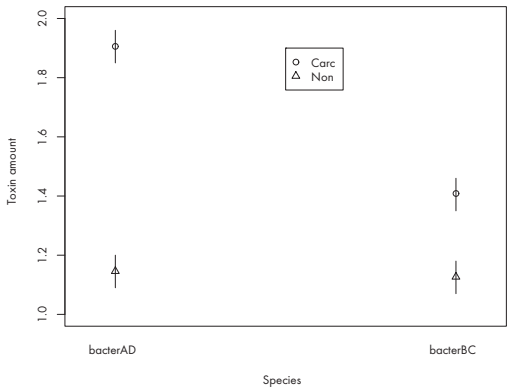


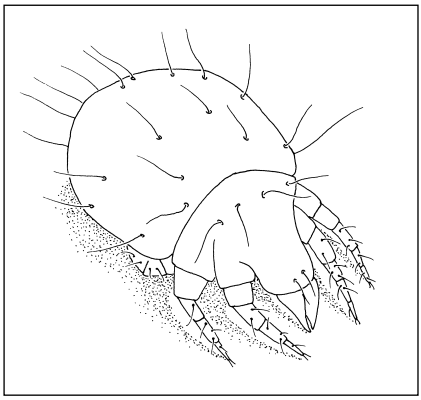
Fig. 8-12 Comparison of mean amount of toxin in two groups of bacteria species and two groups of diseases (*DIAGNOSIS1*). Whiskers are CI_{95}.

```
> legend(1.5,1.9,c("Carc","Non"),pch=1:2)
> lines(c(1,1),c(1.85,1.96))
> lines(c(1,1),c(1.09,1.2))
> lines(c(2,2),c(1.35,1.46))
> lines(c(2,2),c(1.07,1.18))
```

CONCLUSION

The production of toxins differed significantly not only between the bacteria species but also between the diagnoses (LM, $F_{1,76} = 72.5$, $P < 0.0001$). For non-carcinogenic diagnoses, the toxin production was low and similar for all bacteria species. For both carcinogenic diagnoses, the toxin production was higher than for non-carcinogenic diagnoses. There was no significant difference between the two non-carcinogenic diagnoses or between the two carcinogenic diagnoses (contrasts, $P = 0.36$). But a significant difference was found between bacteria species "bacterA" and "bacterD" and between "bacterB" and "bacterC". Nevertheless, a demonstrable difference between the diagnoses does not necessarily mean that there is a causal relationship between the amount of toxin and the disease (it could be, for example, a marker that indicates a complex relationship facilitated via several other variables, to which the occurrence of toxins as well as the diagnosis is related). Even if the relationship between the bacteria and disease were real, the statistical analysis alone cannot discover the direction (whether bacteria cause a disease, or whether a disease facilitates the occurrence of bacteria).

8.6 One-way ANCOVA



The rate of population increase of ectotherms, such as mites, is a function of ambient temperature. In the laboratory, population increase (*rate*) of two mite pest species (*GENUS*: genA, genB) was studied for 11 different temperatures [°C] (*temp*). The jars with mites were placed in temperature-controlled chambers for a few weeks. Each was set to a constant temperature between 10 and 35 °C. The rate of population increase was calculated for each species from initial and final population size measurements, using the exponential population growth assumption. For each temperature, a single measurement for each species was available. a model of the relationship is essential for the control of mite pests, thus we are interested to know: (1) Did temperature affect the rate of increase? (2) Was the rate similar for both species? (3) What could be a suitable temperature-dependent model of the population growth rate for each species?

EDA

The data file includes two explanatory variables. One of them is categorical (*GENUS*), which we will use as a factor. The other one is continuous (*temp*) and we will use it as a covariate. We will display the relationship between *rate* and *temp* using a scatter plot. For better orientation, we will mark each species with a different symbol using the **`points`** function.

```
> dat<-read.delim("mite.txt"); attach(dat); dat
   temp       rate genus
1  10.0 0.00000000  genA
2  12.5 0.06586318  genA
3  15.0 0.09830000  genA
...
22 35.0 0.09352827  genB
> plot(temp,rate,type="n")
> points(temp[genus=="genA"],rate[genus=="genA"])
> points(temp[genus=="genB"],rate[genus=="genB"],pch=16)
```

The change in rate of the population increase is not linear with temperature for either of the species (Fig. 8-13). Note that it is not even monotonous. At first, the rate increases with increasing temperature until it reaches its maximum (somewhere between 25 and 30 °C), beyond which it decreases slowly. The points for both species often overlap, suggesting that the difference between the species might be small and not overwhelmingly important.

MODEL

Even though several models have been proposed for describing the relationship between the rate of development of arthropods and temperature (Kontodimas *et al.* 2004), their form does not fit our situation. One of the problems is that while the developmental rate is always

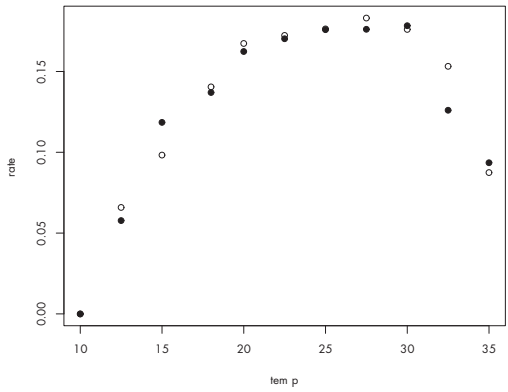


Fig. 8-13 Relationship between *rate* of population increase and temperature (*temp*) for two species of mites: "genA" (○), "genB" (●).

positive, the rate of the population change can also be negative. Based on some background biological reasoning, it appears that a quadratic model might be suitable. a model of the same functional form should be appropriate for both mite species even though the values of individual parameters may differ. The following will be the model for i^{th} observation (i.e. i^{th} temperature) of the j^{th} level of the *GENUS,* invoking the treatment parametrization:

$$rate_{ij} = \alpha + GENUS_j + \beta temp_i + \gamma temp_i^2 + \delta_j temp_i + \omega_j temp_i^2 + \varepsilon_{ij}, \qquad (8\text{-}12)$$
$$\text{where } \varepsilon_i \sim N(0,\sigma^2), \text{ independent among jars.}$$

α, β, γ represent intercept, and coefficients of linear and of quadratic term respectively, all for "genA" (as the reference level of *GENUS*). $GENUS_j$, δ_j, ω_j are the differences between "genB" and "genA" for intercept, and coefficient of linear, and of quadratic terms respectively.

ANALYSIS

From plot (Fig. 8-13), it is clear that the curve is not symmetric around the maximum; this is in contrast to the assumption of a theoretical quadratic model. Therefore, we will include also a cubic term in the first model and for safety reasons also the interaction of covariate (*temp*) with the *GENUS* factor, even though Fig. 8-13 does not support existence of such interaction very strongly:

$$rate_{ij} = \alpha + GENUS_j + \beta temp_i + \gamma temp_i^2 + \tau temp_i^3 + \delta_j temp_i + \omega_j temp_i^2 + \eta_j temp_i^3 + \varepsilon_{ij},$$
$$\text{where } \varepsilon_i \sim N(0,\sigma^2), \text{ independent among jars.} \qquad (8\text{-}13)$$

In contrast to the model (8-12), there are two more parameters in the current model, namely the coefficient of cubic trend for "genA" (τ) and η_j as a difference of this coefficient for "genB" (given the treatment parametrization). The cubic term is specified by calling the **`poly(temp,3)`** function. Beware that **`poly`** uses a different parametrization for linear, quadratic, and cubic terms than the parametrization given in (8-13), as we explained in Chapter 5.4.2.

```
> m1<-lm(rate~poly(temp,3)*genus)
> anova(m1)
Analysis of Variance Table

Response: rate
                    Df   Sum Sq  Mean Sq  F value    Pr(>F)
poly(temp, 3)        3 0.065953 0.021984 210.0675 7.125e-12 ***
genus                1 0.000028 0.000028   0.2644    0.6152
poly(temp, 3):genus  3 0.000108 0.000036   0.3454    0.7930
Residuals           14 0.001465 0.000105
...
```

The ANOVA table demonstrates that the interactions of the linear, quadratic, and cubic terms of temperature with the *GENUS* factor are not significant as seen from the result of the simultaneous test of all three interactions, based on three degrees of freedom. The relationship between the rate of the population change and temperature will thus probably be analogous for both species. Therefore, we remove the interaction from the model and rewrite the model formula without it. We will verify the change incurred by the modification by comparing both models using the **anova** function.

```
> m2<-lm(rate~poly(temp,3)+genus)
> anova(m1,m2)
Analysis of Variance Table

Model 1: rate ~ poly(temp, 3) * genus
Model 2: rate ~ poly(temp, 3) + genus
  Res.Df         RSS Df   Sum of Sq      F Pr(>F)
1     14  0.00146516
2     17  0.00157360 -3 -0.00010844 0.3454  0.793
> anova(m2)
Analysis of Variance Table

Response: rate
             Df   Sum Sq  Mean Sq  F value    Pr(>F)
poly(temp, 3)  3 0.065953 0.021984 237.5038 4.509e-14 ***
genus          1 0.000028 0.000028   0.2989    0.5917
Residuals     17 0.001574 0.000093
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Neither the interaction, nor the main effect of *GENUS* are significant. This indicates that we do not have enough evidence in this data to conclude that there is difference between species, even in the intercepts. Practically, this means that the lower temperature threshold (i.e. a temperature at which the rate is zero) of the rate of population increase is similar for both species. The factor *GENUS* will, therefore, be removed from the model. As a result, there will be only a single variable in the model, namely *temp*. At the same time, we use a different syntax of model formula to specify polynom using **I()**. This choice is convenient when we want to obtain coefficients that are compatible with the parametrization used in formula (8-13). While it is certainly possible to recompute the output from the orthogonal **poly** parametrization to the parametrization in (8-13), it is much more complicated than just calling the model directly in the (8-13) parametrization. Estimates (and their SE) can then be read from the table of coefficients obtained by calling **summary**.

```
> m3<-lm(rate~temp+I(temp^2)+I(temp^3))
> summary(m3)

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.675e-01  5.138e-02  -5.205 5.97e-05 ***
temp         3.191e-02  7.855e-03   4.063  0.00073 ***
I(temp^2)   -3.986e-04  3.704e-04  -1.076  0.29608
I(temp^3)   -6.178e-06  5.464e-06  -1.131  0.27309
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.009432 on 18 degrees of freedom
Multiple R-Squared: 0.9763,     Adjusted R-squared: 0.9723
F-statistic: 247.1 on 3 and 18 DF,  p-value: 8.234e-15
```

The table of coefficients shows parameters common for both species. The intercept, as well as the slope of the linear trend are significantly different from zero, while the coefficients of the quadratic and cubic terms are not (p-value > 0.273). We could get rid of them, could we not? Well, do not rush and do not reduce the model too hastily. Most of the time, it is better not to remove multiple non-significant terms together but just step by step! For now (because of the marginality rule mentioned in Chapter 5.4.3), we only see that we do not need the cubic term and we will thus remove it – once again by rewriting the model formula.

```
> m4<-lm(rate~temp+I(temp^2))
> summary(m4)
...
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.222e-01  1.734e-02  -18.59 1.20e-13 ***
temp         4.060e-02  1.662e-03   24.42 8.20e-16 ***
I(temp^2)   -8.154e-04  3.649e-05  -22.35 4.20e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.009501 on 19 degrees of freedom
Multiple R-Squared: 0.9746,     Adjusted R-squared: 0.9719
F-statistic: 364.7 on 2 and 19 DF,  p-value: 6.978e-16
```

The coefficient of the quadratic term in model `m4` is now significantly different from zero. As you can see, it is really fortunate that we did not remove it – that would have been a serious error! We will consider model `m4` to be our final model. It has the following formula:

$$rate_i = \alpha + \beta temp_i + \gamma temp_i^2 + \varepsilon_i, \quad \text{where } \varepsilon_i \sim N(0,\sigma^2), \text{ independent among jars.} \tag{8-14}$$

Estimates (a, b, c) of unknown parameters α, β, γ, can be read from the table of coefficients of the model `m4`.

By studying the pre-set diagnostic plots, available from the **`plot`** function (called with the fitted model object as argument), you will discover that variance of the residuals is more or less homogenous even though there is a slight indication of curvature. If you check other as-

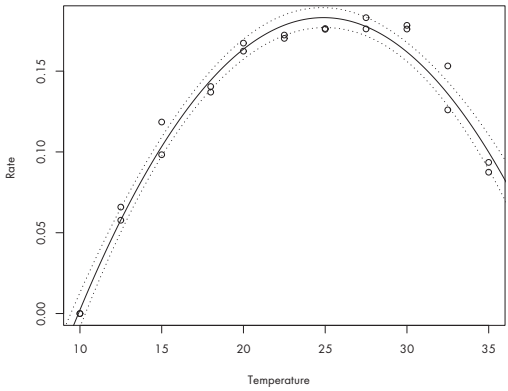


Fig. 8-14 Quadratic model (—) of the relationship between the rate of population increase and temperature for two mite species. Dotted lines show 95% (point-wise) confidence intervals for mean values of *rate*.

sumptions, particularly the relationship between the residuals and the explanatory variable *temp*, using the following commands:

```
> sr<-rstandard(m4); plot(temp,sr)
```

you will find out that no trend is apparent here. The model based on the normal distribution of errors appears to be basically adequate.

We plot the final model `m4` with (pointwise constructed) confidence intervals for expected values of the response variable, *rate*. First, we draw a simple scatter plot. Then we create an auxiliary vector `x` with a range corresponding to the range of *temp*. Using **`lines`** with the arguments **`predict`** and **`list`** (in which we join `x` with *temp*), we draw the estimated model. 95% confidence intervals will be calculated according to (3-4) from the predicted values (`fit`) and their standard errors (`se.fit`) obtained from the function **`predict`** and saved to the object `ci`. The limits of CI (upper `ciU` and lower `ciL`) will be superimposed on the plot using **`lines`**, setting the dotted line type instead of the default solid line (Fig. 8-14):

```
> plot(temp,rate,xlab="Temperature",ylab="Rate")
> x<-seq(from=0,to=40,by=0.1)
> lines(x,predict(m4,list(temp=x)))
> ci<-predict(m4,list(temp=x),se.fit=T)
> names(ci)
[1] "fit"            "se.fit"         "df"             "residual.scale"
> ciU<-ci$fit+qt(.975,19)*ci$se.fit
> ciL<-ci$fit+qt(.025,19)*ci$se.fit
> lines(x,ciL,lty=3)
> lines(x,ciU,lty=3)
```

Be careful when interpreting the collection of confidence intervals as "confidence bands" for the unknown curve! Individual intervals, which seemingly form a band suitable for this

purpose were, in fact, constructed (and thus also "calibrated") point by point. This means that (under the model assumptions), they make sure that at any single temperature value, their coverage is as desired (95%). Nevertheless, that does not mean that the bands produced in this way (even though created for many temperatures within the given interval) have the specified coverage for the entire curve within the given interval. a belt with the prescribed coverage for the curve can be also constructed, but it is somewhat more difficult. It will typically be (much) wider at any temperature point than the pointwise confidence interval.

CONCLUSION

The rate of the mite population change was similar for both species (LM, $F_{1,17} = 0.3$, $P = 0.59$). The rate significantly depends on temperature and the relationship is non-linear (with respect to temperature) (LM, $F_{3,18} = 247.1$, $P < 0.0001$). The rate estimates can be obtained by entering particular temperatures in the following formula:

$$-0.322 + 0.0406\,temp - 0.0009\,temp^2.$$

9 GAMMA AND LOGNORMAL DISTRIBUTIONS

We encounter data following the gamma and lognormal distributions relatively often in practice. These can arise, for instance, when we measure small values with a high accuracy (such as, concentrations of substances that occur in small or even trace amounts, or the weight of organisms) or time. The values of all such data can only be positive. These models do not even allow zero measurement values (zero probability is assigned to the value of zero).

As we have stated in Chapter 7.1, both of these distribution types have quite similar characteristics and we will thus treat them together despite the fact they actually belong to different groups. While the gamma distribution belongs to the GLM distribution class, the lognormal distribution does not belong there directly. However, it can be transformed to GLM (particularly to the normal distribution) using a simple (logarithmic) transformation.

9.1 Description of the gamma model

In **`glm`**, gamma distribution is specified with the argument **`family=Gamma`**. This distribution has two parameters. In contrast to the generally used standard parametrization for this distribution (that uses parameters shape and scale) we will use a parametrization which is more natural in the context of GLM. The first parameter of the measurement y coming from the gamma distribution is the expected value $E(y) = \mu$. This value must always be positive (this also applies to all measurements of y). The second is the **dispersion parameter**, ϕ. This parameter corresponds to the ratio $Var(y)/\mu^2$, so that $Var(y) = \phi\mu^2$. The value of the dispersion parameter can be found in the summary output of GLM objects.

`Gamma` within **`glm`** has an argument which specifies link (see Chapter 4.3). The default option (invoked automatically when we do not specify any link explicitly) is **`Gamma(link="inverse")`**, i.e. $1/\mu$. Alternative link options are **`identity`** and **`log`**. Different links have different uses. Inverse link is used when the observations are bounded from below. Logarithmic link is often used when the data have an exponential trend (the structure of the linear predictor then very conveniently translates into the multiplicative structure on the scale of the original measurements). The **`identity`** link does not do any transformation on the mean at all. This can be useful in some situations and dangerous at others. The danger comes from a problem also common to other situations where the link domain is not the same as the range of allowable mean values. Here, depending on the particular design setup, there is generally no guarantee that for, particular parameter values, the link transformed mean and hence the mean implied by the linear predictor is always positive. As we learned before, gamma distribution cannot have non-positive mean and hence

a conflict may occur. This type of problem can lead to lack of convergence and errors in the iterative algorithm which fits the model. Please note that not all links are "safe" to the same extent. While log link is safe with the gamma distribution, the identity and inverse links might not be always safe.

When interpreting results, keep in mind that estimates in the table of coefficients are on the link-transformed scale. To transform them back to the scale of measurements, it is necessary to use an inverse function to the link function. The inverse function to **`inverse`** link is, again, inverse. The inverse function to **`log`** link is the exponent function. Use of **`identity`** link does not require any transformation (and hence back-transformation is not required either). Calculation of expected values from original measurements by inverse transformation to the link is for links other than **`identity`** biased due to the Jensen inequality (Montgomery & Runger 1994).

9.2 Description of the lognormal model

The lognormal distribution can be easily implemented in the **`lm`** or **`glm`** functions after the response variable is logarithmically transformed. The transformation stabilises variance – it transforms the originally heteroscedastic data (variance of which depends on the mean value) to homoscedastic data (with a constant variance). It also changes the asymmetry of the original lognormal distribution (symmetrifies it).

Just like the normal distribution, this distribution has two parameters: let us call them μ_{tr}, σ_{tr}^2. The expected value and variance of the original lognormal quantity are their functions:

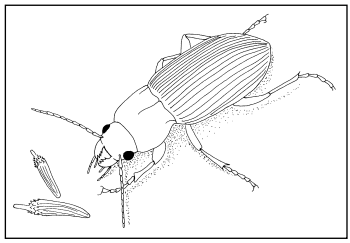
$$\mathrm{E}(y) = \exp\left(\mu_{tr} + \frac{\sigma_{tr}^2}{2}\right), \tag{9-1}$$

$$\mathrm{Var}(y) = \exp(\sigma_{tr}^2 - 1)\exp(2\mu_{tr} + \sigma_{tr}^2). \tag{9-2}$$

Note that while the expected value is independent on variance on the log-transformed scale, on the original scale the variance is functionally related to the mean. If we use the log transformed response, the estimated model coefficients relate to μ_{tr}. They are directly interpretable on the logarithmic scale. To transform them to the original measurement scale, the exponential function is used: $\exp(\mu_{tr})$. Due to the Jensen's inequality, the exponentially back-transformed estimates will be biased even if the estimates on the log-scale were unbiased. For instance, if we do $\exp(\mu_{tr})$, we get a (slightly biased) estimate of the median, not of the expected value (as is clear from 9-1).

The main reasons why analyses based on lognormal distributions are commonly used are both interpretability of the model coefficients and the fact that upon log-transformation of the response, we can continue working with the standard linear model (**`lm`**), without any additional software or theory.

9.3 Regression



In predators, the size of prey is related to their body size. In the laboratory, acceptance of food was studied in 36 species of granivorous beetles, each represented by a single individual. Each beetle was offered seeds of a various size (measured as mass [g]). Consumed seed mass (*seed*) was recorded. For each beetle, body size [mm] (*body*) was recorded as well. We want to find answers to the following questions: (1) Is size of seeds related to carabid body size? (2) What is the shape of the relationship? (3) Do the data support the existence of an upper asymptote in the relationship (e.g. due to the physical limitation induced by the size of mandibles)?

EDA

Only one continuous explanatory variable is included in the data frame. That is why we will draw the relationship between the size of seeds (*seed*) and the size of the beetles (*body*) into a scatter plot.

```
> dat<-read.delim(„granivore.txt"); attach(dat); dat
   body    seed
1   0.9 0.03240
2   0.9 0.04956
3   0.9 0.06564
...
36 31.3 0.65586
> plot(body,seed)
```

The mass of the consumed seeds really increases with the beetles' body size (Fig. 9-1): larger species consume larger seeds. Nevertheless, it is obvious that the relationship is not linear. The increase of seed by a unit of mass is not the same within the full range of the *body* sizes.

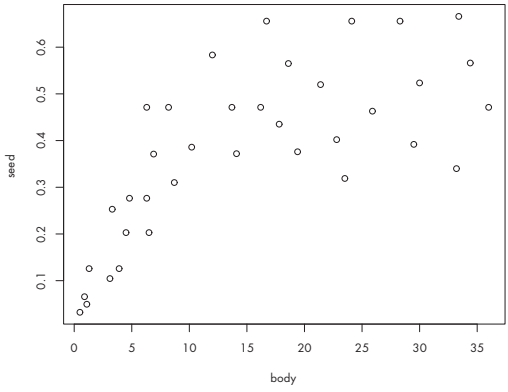


Fig. 9-1 Relationship between the size of consumed seeds (*seed*) and the body size (*body*) of beetles.

As a matter of fact, this increase slows down with increasing body size. Formally speaking, the derivative is positive, but is getting smaller (the relationship is concave). Beetles larger than 12 mm consumed seeds of a quite similar mass. It appears that, in agreement with a basic biological intuition, the data suggest that the size of consumed seeds cannot increase indefinitely, i.e. they support the idea that there should be an asymptote. Also note how dispersion of the points increases with the growing values of the *body* variable.

MODEL

To a certain extent, this example is similar to the studies of the so-called functional response of predators, which relates the quantity of the consumed prey to the offered prey quantity. The shape of the relationship is similar to a functional response of Type II, which is commonly modelled by the Holling's disc equation (Holling 1965). The equation is based on the reciprocal function (Crawley 1993). The model for the i^{th} measurement is then:

$$\frac{1}{\mu_i} = \alpha + \beta \frac{1}{body_i}, \tag{9-3}$$

where $seed_i \sim Gama(\mu_i, \phi)$, independent among beetles.

ANALYSIS

In order to fit Holling's disc equation, we will use **`glm`** with **`Gamma`** distribution. As we mentioned above, a gamma distribution allows for variance increasing with the expected value. Unlike in many other situations, we have a very good reason to use the inverse link function here. It is because of the particular functional form of the model (9-3), where the reciprocal ($1/\mu$) appears on the left side of the model equation. Inverse link is the canonical link for gamma and hence default option for the **`Gamma`** family. Thus it is invoked even if we do not specify it explicitly. After the tilde in the model formula, we have to take the reciprocal of *body* in order to tell the **`glm`** function that we want to do the same thing as in (9-3). Beware that various mathematical operators can have a different meaning on the right side of the model formula compared to what one might naively expect (see Chapter 6.4). For example, forward slash does not stand for a division, but for "nested" effects. So to calculate a reciprocal of body size (`1/body`) we have to use the interpreter function **`I`**. Alternatively, we could make an inverse of *body* prior to analysis and store it in a variable whose name could be entered directly.

Upon fitting the model, we will obtain *a* and *b* estimates of unknown parameters α, β (and an estimate of an unknown dispersion parameter, ϕ). We will explain the interpretation of these parameters later. The ANODEV table, which is an analogue of the ANOVA table (but which is based on more complicated calculations assuming gamma distribution), will be displayed using the **`anova`** function. This is a generic function with a very general scope, thus when using **`glm`** objects, we have to specify the type of the test statistics using the **`test`** argument. We will choose an asymptotic test, which is an *approximate* analogue of the F-test (**`test="F"`**). It is more conservative than the χ^2 test (**`test="Chi"`**). F has a tendency to produce less significant results than the χ^2 test, as it corrects, though in somewhat *ad hoc* fashion, for an uncertainty in the dispersion parameter estimate.

```
> m1<-glm(seed~I(1/body),family=Gamma)
> anova(m1,test="F")
Analysis of Deviance Table

Model: Gamma, link: inverse

Response: seed

Terms added sequentially (first to last)


          Df Deviance Resid. Df Resid. Dev      F    Pr(>F)
NULL                         35    15.3681
I(1/body)  1   8.3662        34     7.0019 42.624 1.787e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ANODEV table is superficially quite similar to the ANOVA table, but it differs in a few details. In the top part of the output, there is information about the distribution, link, and the response variable. One should always check this transcription of what the model did in order to catch errors and discrepancies between what was intended and what was actually submitted to the **glm** function. The main part of the table has only two lines here, but it can have many more lines depending on the number of terms in the model. In the first line, residual deviance (see Chapter 4.3) and degrees of freedom for the null model (NULL) are listed. The NULL is the model which does not include any explanatory variables, just the intercept. In the other line, deviance, residual deviance of the effect and F-test related info for the effect of `1/body` are listed.

The effect of the reciprocal of *body* is significant. Firstly, we should find out if such a model is not too small. For example, we can check if we could improve it by some simple expansion, such as adding a quadratic term for the inverse of the *body*:

$$\frac{1}{\mu_i} = \alpha + \beta \frac{1}{body_i} + \gamma \frac{1}{body_i^2}, \tag{9-4}$$

$$seed_i \sim Gama(\mu_i, \phi), \text{ independent among beetles.}$$

We could (and should) try several other improvements. However, we will not do it now (try them for yourself). Instead, we will explore whether the model will significantly improve when a quadratic term is added. We will test the newly added quadratic term using the **anova** function, now called with two fitted model objects as arguments.

```
> m2<-glm(seed~I(1/body)+I(1/body^2),Gamma)
> anova(m1,m2,test="F")
Analysis of Deviance Table

Model 1: seed ~ I(1/body)
Model 2: seed ~ I(1/body) + I(1/body^2)
  Resid. Df Resid. Dev Df Deviance      F Pr(>F)
1        34     7.0019
2        33     7.0016  1   0.0003 0.0013 0.9713
```

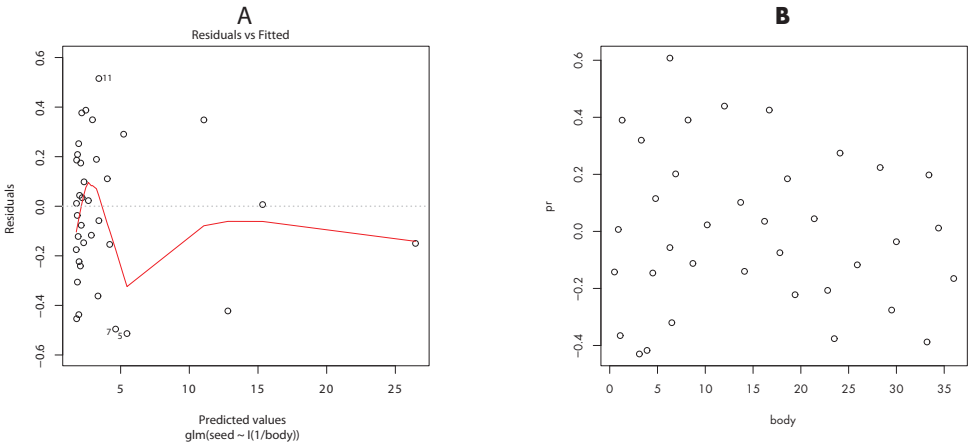


Fig. 9-2 A. Relationship between deviance residuals and fitted values of `m1`. **B**. Relationship between Pearson residuals and the explanatory variable *body*.

Addition of the reciprocal quadratic term did not cause a significant change to the deviance and we will stick to the model `m1`.

Next, we will look at the behaviour of the residuals, making sure there are no essential problems with model `m1`. We have to always keep in mind that problems with residuals can be caused by errors in the specification of the systematic as well as the random parts. The residuals should not depend on predicted values or the explanatory variable included in the model. Let us explore the plots, by the means of which we can (informally) verify whether these assumptions we met:

```
> plot(m1,which=1)
```

Variability of deviance residuals (4-15) appears relatively homogenous (Fig. 9-2A). Now, against *body*, we plot Pearson residuals (4-18) that explicitly correct for inherent heteroscedasticity caused by parameter estimation and the fact that the data has gamma distribution:

```
> pr<-resid(m1,type="pearson"); plot(body,pr)
```

The plot (Fig. 9-2B) does not indicate any serious problems. Estimates of parameters of the model can be displayed by calling **`summary`**.

```
> summary(m1)

Call:
glm(formula = seed ~ I(1/body), family = Gamma)

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-0.7530027  -0.4237538   0.0008676   0.2527096   0.7024871
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.7418     0.3162   5.508 3.76e-06 ***
I(1/body)    11.8626     2.4463   4.849 2.69e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Gamma family taken to be 0.1962785)

    Null deviance: 15.3681  on 35  degrees of freedom
Residual deviance:  7.0019  on 34  degrees of freedom
AIC: -49.676

Number of Fisher Scoring iterations: 5
```

The summary output is very similar to that for **`lm`** objects. It differs mainly in the last few lines. There are no coefficients of determination and the overall F-test. Instead, there is information about the null and residual deviance, degrees of freedom, and the number of iterations used by the iterative optimisation algorithm, which were needed to arrive at the final estimates. In the table of coefficients, there are estimates of two parameters. What is their biological interpretation?

The reciprocal of the first parameter (*a*), i.e. 1/1.7418 = *0.574*, is an asymptote which is approached by the curve at large values of *body*. To find an answer to the question: "Why is it so?", just perform a few elementary algebraic operations with the equation (9-3). Previously, we guessed that the asymptote will be around *0.6*. Truly, our guess was not bad! The reciprocal of the second parameter (*b*), i.e. 1/11.8626 = *0.084*, is the derivative of the curve at $x = 0$. It is approximately the slope of the curve for *small* values of *body*.

How good is the model at explaining the data? The coefficient of determination is not shown in the summary. a *very* simple analogous version can be obtained as a ratio of explained to total deviance: (15.3681 – 7.0019) / 15.3681 = *0.54*. The value 54 % (i.e. slightly more than half of total deviance for *seed*) is not all that bad (though not perfect either). Beware, we stress once again that this is not the same thing as R^2 used for general linear model (Chapter 8.2), but a simple adaptation of the idea of the coefficient of determination. There are several other better, but more complicated, adaptations like that (Mittlböck & Heinzl 2002).

Now we will plot the model. We will add upper and lower limits of 95% (pointwise constructed) confidence intervals to the curve of expected values estimated by the model. Recall the substantial difference in pointwise and simultaneous confidence interval interpretation emphasised at the end of Chapter 8.6.

We will use the scatter plot without actually plotting the points (**`type="n"`**). Create an auxiliary vector (`x`), ranging from 0 to 40. Predicted values can be found by applying **`predict`**. To transform estimates which are at the scale of the linear predictor to the scale of observed values, we have to use argument **`type="response"`**. Confidence limits will be calculated in three steps. At first, we will extract predicted values (`fit` column) and their standard errors (`se.fit` column) on the scale of the linear predictor (**`type="link"`**) into

object `ci`. Then we will calculate the lower (`ciL`) and then the upper limit (`ciU`) from object `ci` using (3-4) – still on the scale of the linear predictor. Finally, to the scale of the observed values, we will transform the values by applying an inverse function of the used link, which is in this case an inverse (Fig. 9-3):

```
> plot(body,seed,type="n",xlab="Body size",ylab="Seed weight")
> x<-seq(from=0,to=40,by=1)
> lines(x,predict(m1,list(body=x),type=""response"))
> ci<-predict(m1,list(body=x),type="link",se.fit=T)
> names(ci)
[1] "fit"            "se.fit"          "residual.scale"
> ciU<-ci$fit-qt(0.975,34)*ci$se.fit
> ciL<-ci$fit+qt(0.975,34)*ci$se.fit
> lines(x,1/ciL,lty=3)
> lines(x,1/ciU,lty=3)
```

The reciprocal function is of course not the only one possible to use when we study a particular relationship. Let us try to express this relationship using a polynomial of a second order (a simple quadratic model with normal distribution):

$$seed_i = \alpha + \beta body_i + \gamma body_i^2 + \varepsilon_i \,, \qquad (9\text{-}5)$$

where $\varepsilon_i \sim N(0,\sigma^2)$, independent among beetles.

We will use the **`lm`** function that is based on a (homoscedastic) normal distribution of errors (the reader may already know that something like this goes against the features of the modelled data).

```
> m3<-lm(seed~poly(body,2))
> summary(m3)
...
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.30842    0.02318  13.305 8.17e-15 ***
```

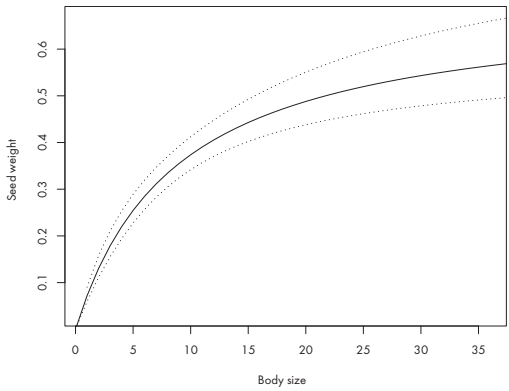


Fig. 9-3 Model (—) of the relationship between seed size and body size of beetles with 95% (pointwise) confidence intervals (·····).

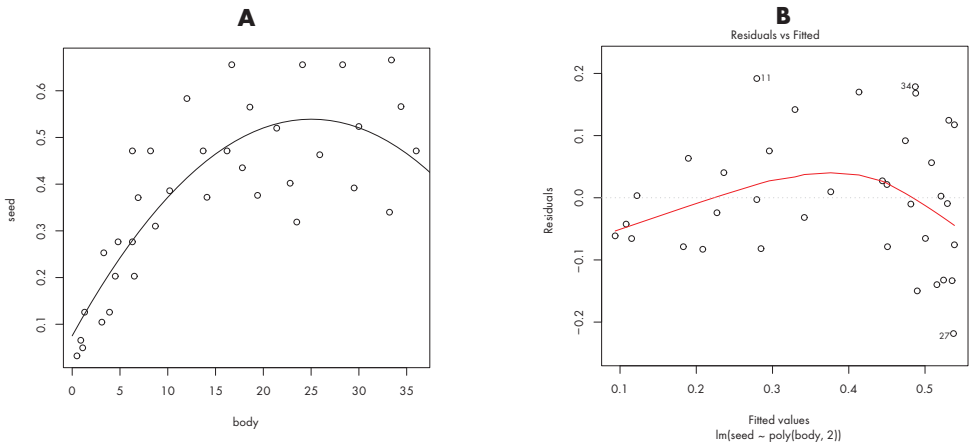


Fig. 9-4 A. Quadratic model of the relationship between *seed* and *body*. **B**. Relationship between residuals and predicted values of model `m3`.

```
poly(body, 2)1  0.55682    0.13908   4.004 0.000333 ***
poly(body, 2)2 -0.41591    0.13908  -2.990 0.005235 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1391 on 33 degrees of freedom
Multiple R-Squared: 0.4307,     Adjusted R-squared: 0.3962
F-statistic: 12.49 on 2 and 33 DF,  p-value: 9.173e-05
```

The relationship is significant and the presence of the quadratic term thus appears to be necessary. Let us look at how the model fit compares to the data, overlaying them in one plot.

```
> plot(body,seed)
> lines(x,predict(m3,list(body=x)))
```

It appears that the quadratic model fits the relationship relatively well (Fig. 9-4A). This is true at least for the low values of seed. However, later the fit starts to bend down. In other words, it predicts that bigger beetles will consume smaller seeds than medium-sized beetles. From a biological perspective, this is very unlikely. How do the residuals from such a model behave? We will only look at the first pre-set plot.

```
> plot(m3,which=1)
```

The relationship between the residuals and the fitted values is departing from the assumptions (Fig. 9-4B). In some areas, the model systematically slightly underestimates and, in other areas, overestimates the data. Variance is also changing systematically: it increases with the predicted values. We should consider model `m3` to be inadequate even if we do not know that it is biologically unreasonable and that it goes against some basic characteristics of the data apparent from EDA.

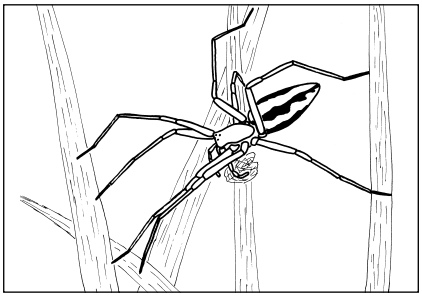
Overall, we can say that the quadratic model does not appear to be suitable from the perspective of either the systematic or the random part. The Holling's disc equation has two major advantages in comparison with the quadratic model. First, it is more parsimonious since it has one less parameter and its variance (realistically) increases with the mean value. Secondly, it shows qualitatively reasonable behaviour that complies with biological intuition (the asymptote).

CONCLUSION

The size of the consumed seeds increased significantly with the size of the beetles' bodies (GLM-g, $_{F1,34}$ = 42.6, P < 0.0001). The relationship is not linear. We can obtain a prediction of the sizes of consumed seeds based on the size of the beetles' bodies from an estimated model:

$$\frac{body}{1.742\,body + 11.86}.$$

9.4 Two-way ANODEV



In the gift-giving spider, a male brings a gift (a wrapped prey) to a female before mating in order to avoid being cannibalised. After approaching a female, the male presents the gift and female explores it. Several variables can potentially influence the time between presentation and acceptance of the gift. The effect of two variables was studied in a laboratory: satiation of female (*FEED*: satiated, starved) and their mating experience (*MATING*: mated, virgin). Time [s] of the gift presentation (*time*) was recorded. The experiment was fully factorial, for each combination, ten males and ten females were randomly selected from a large wild population and used for the experiment. We want to answer the following questions: (1) Is presentation time affected by any of the two variables? (2) If it is, how different the presentation time is on average between different factor levels?

EDA

As there are two categorical explanatory variables, *FEED* and *MATING*, we plot the data using the interaction plot. We also make a histogram of the response variable *time*.

```
> dat<-read.delim("pisaura.txt"); attach(dat); dat
   mating     feed time
1   mated  starved  125
2   mated  starved  540
3   mated  starved    5
...
40 virgin satiated   20
> interaction.plot(mating,feed,time)
> hist(time)
```

The mean values of *MATING* levels are relatively similar (Fig. 9-5A). On the other hand, there is a substantial difference between the levels of *FEED*. Since the connecting lines are almost parallel, we do not expect that the interaction between both factors would be significant.

MODEL

Based on the explorative analysis, we expect that the presentation time is influenced by how full the female is and maybe also by previous mating. The effect of these factors should be additive – they should not interact with each other. Therefore, we expect an additive model with a normal error distribution. For the i^{th} measurement of the combination of the j^{th} *MATING* level and the k^{th} level of *FEED* we have:

$$time_{ijk} = \alpha + MATING_j + FEED_k + \varepsilon_{ijk}, \qquad (9\text{-}6)$$

where $\varepsilon_{ijk} \sim N(0,\sigma^2)$, independent among measurements.

In the treatment parametrization α stands for the expected value for the combination "satiated-mated", $MATING_j$ is the difference between particular *MATING* levels. $FEED_k$ is the difference between particular *FEED* levels. The differences are necessarily 0 for $i = 1$ and $j = 1$ (as they should when differencing the same mean of the same level). It is clear enough that (9-6) is a tentative model because, for instance, in contrast to the assumptions of the normal distribution of data, measurements of time have to be positive. From the histogram (Fig. 9-5B) it is clear that they are markedly skewed. But it is a simple model, and therefore attractive. Now we begin to show its (dis-)advantages.

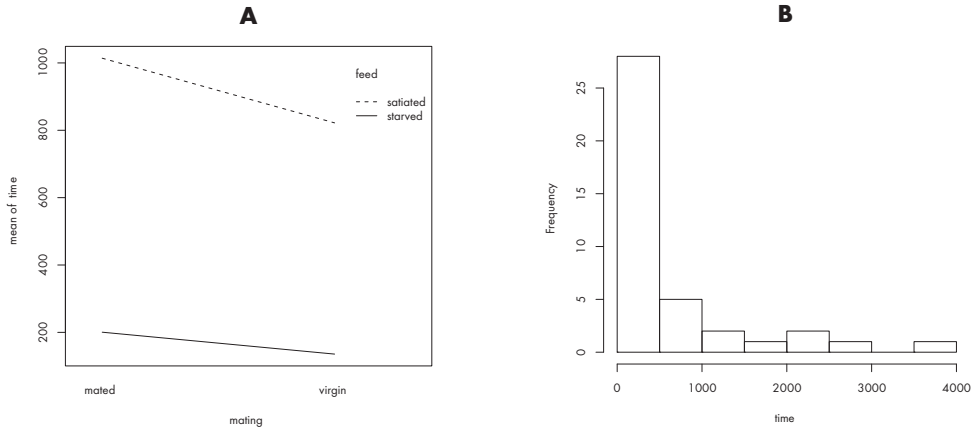


Fig. 9-5 A. Comparison of mean presentation time for "satiated" and "starved" females, which were "mated" or were "virgin". **B.** Histogram of *time*.

ANALYSIS

Just to be sure, we will add an interaction to the initial model (in order to obtain a formal test of the additivity assumption). The model will thus be:

$$time_{ijk} = \alpha + MATING_j + FEED_k + MATING:FEED_{jk} + \varepsilon_{ijk}, \quad \text{where } \varepsilon_{ijk} \sim N(0,\sigma^2), \text{ independent among measurements.} \tag{9-7}$$

For modelling, we will use the **lm** function, which assumes a normal error distribution.

```
> m1<-lm(time~mating*feed)
> anova(m1)
Analysis of Variance Table

Response: time
            Df   Sum Sq  Mean Sq F value   Pr(>F)
mating       1   165122   165122  0.2558 0.616098
feed         1  5625000  5625000  8.7142 0.005528 **
mating:feed  1    40322    40322  0.0625 0.804058
Residuals   36 23237845   645496
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Neither interaction nor the main effect of *MATING* are significant. The expected values of levels of *MATING* are similar for different levels of *FEED*. Therefore, the model `m1` can be simplified by removal of the non-significant terms using the function **update**. We start with the interaction:

```
> m2<-update(m1,~.-mating:feed)
> anova(m1,m2)
Analysis of Variance Table

Model 1: time ~ mating * feed
Model 2: time ~ mating + feed
  Res.Df      RSS Df Sum of Sq      F Pr(>F)
1     36 23237845
2     37 23278168 -1    -40323 0.0625  0.804
```

MATING did not seem significant either, so we will test it – by comparing the two-way additive model with both factors and the one-way model with *FEED* only:

```
> m3<-update(m2,~.-mating)
> anova(m2,m3)
Analysis of Variance Table

Model 1: time ~ mating + feed
Model 2: time ~ feed
  Res.Df      RSS Df Sum of Sq      F Pr(>F)
1     37 23278168
2     38 23443290 -1   -165122 0.2625 0.6115
> anova(m3)
Analysis of Variance Table

Response: time
```

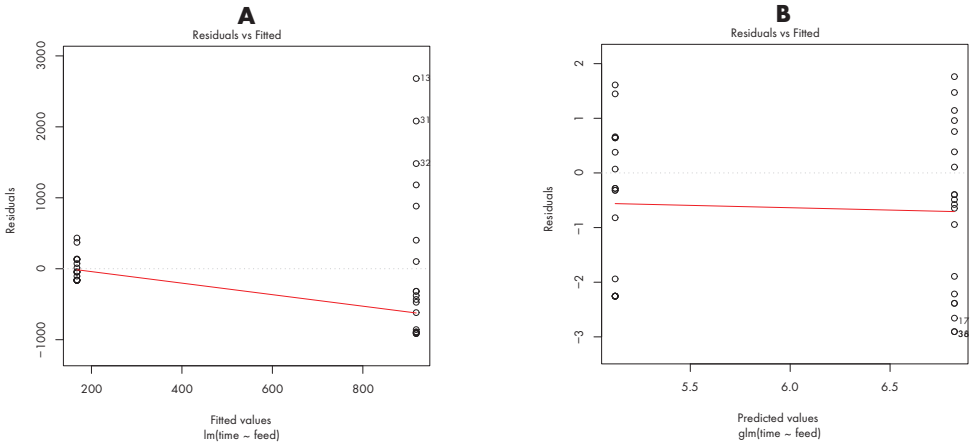


Fig. 9-6 A. Relationship between residuals and predicted values of model `m3`. **B**. Relationship between deviance residuals and fitted values of model `m6`.

```
          Df   Sum Sq  Mean Sq F value   Pr(>F)
feed       1  5625000  5625000  9.1177 0.004507 **
Residuals 38 23443290   616929
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The factor *MATING* remains non-significant. Model `m3` includes only the factor *FEED*, thus it has the following form:

$$time_{ik} = \alpha + FEED_k + \varepsilon_{ik}, \qquad (9\text{-}8)$$

where $\varepsilon_{ik} \sim N(0,\sigma^2)$, independent among measurements.

This means that the presentation time differed only due to whether or not the female was satiated. Now we will explore the behaviour of the residuals. According to (9-8), we can expect that the errors have a normal distribution and are independent. Is this really the case?

```
> plot(m3,which=1)
```

We can see that the residuals do not have a constant variance. Variance of those on the left is several times smaller than those on the right (Fig. 9-6A). There is no reason to doubt independence of the residuals (since individual measurements were made on unrelated subjects in a randomised order), but model `m3` is not appropriate because it assumes unrealistic (homoscedastic) behaviour of errors. a distribution with non-constant variance will have to be used instead.

While the presentation time is of a continuous character, it can have only positive values, and thus has an asymmetric distribution. Let us try the gamma distribution with the logarithmic link. We will specify it using the **`Gamma`** argument in the **`glm`** function. We will use a model that includes the same systematic part, i.e.

$$\log(\mu_{jk}) = \alpha + MATING_j + FEED_k + MATING:FEED_{jk}, \quad \text{where } time_{jk} \sim Gama(\mu_{jk}, \phi), \text{ independent among measurements,} \tag{9-9}$$

which we will gradually simplify by removal of non-significant terms.

```
> m4<-glm(time~mating*feed,Gamma(link=log))
> anova(m4,test="F")
...
            Df Deviance Resid. Df Resid. Dev       F    Pr(>F)
NULL                           39    122.018
mating       1    0.564        38    121.454  0.3888 0.5368618
feed         1   26.258        37     95.196 18.1021 0.0001425 ***
mating:feed  1    0.083        36     95.113  0.0570 0.8126218
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> m5<-update(m4,~.-mating:feed)
> anova(m5,test="F")
...
       Df Deviance Resid. Df Resid. Dev       F    Pr(>F)
NULL                       39    122.018
mating  1    0.564        38    121.454  0.4016 0.5301713
feed    1   26.258        37     95.196 18.6973 0.0001111 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> m6<-update(m5,~.-mating)
> anova(m6,test="F")
...
     Df Deviance Resid. Df Resid. Dev    F    Pr(>F)
NULL                     39    122.018
feed  1   25.922        38     96.096 20.3 6.138e-05 ***
```

The final model includes only *FEED* again:

$$\log(\mu_k) = \alpha + FEED_k, \quad \text{where } time_k \sim Gama(\mu_k, \phi), \text{ independent among measurements.} \tag{9-10}$$

How does it look with the model diagnostics now?

```
> plot(m6,which=1)
```

It is much better – variance of the deviance residuals (4-15) looks quite homogenous (Fig. 9-6B), even though it is not perfect. The model based on a gamma distribution appears to be better than the model based on a normal error distribution. It is interesting, however, that both statistical models arrive at results which are qualitatively the same. This means that, from the biological perspective, we have not introduced any ground-breaking change in the interpretation of the results by switching to a better statistical model, but this will not be always the case!

Let us therefore look at the parameter values. We have only one factor with two levels left in model `m6`. The table of coefficients will thus include just two items.

```
> summary(m6)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.8222     0.2527  26.999  < 2e-16 ***
feedstarved  -1.6982     0.3573  -4.752 2.87e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
...
```

The values of the coefficients appear strange, don't you think? This is because we used a logarithmic link function and treatment contrasts. Thus the first coefficient stands for log-transformed expected value for "satiated". The second coefficient is an estimate of difference between expected values of two levels on the logarithmic scale (i.e. an estimate of the logarithm of the ratio of expected values for "starved" and "satiated"). By taking the exponent of the absolute value of the difference we get the ratio of "satiated" to "starved": exp(1.6982) = *5.464*, i.e. time for "satiated" is more than five-times longer than for "starved". To calculate the mean for "starved", we will first add the two coefficients and then take the exponent. The calculation can be verified by comparing them with the empirical mean values. The arithmetic averages can be easily obtained by calling the **`tapply`** function:

```
> exp(6.8222)
[1] 918.0024
> exp(6.8222-1.6982)
[1] 168.0061
> tapply(time,feed,mean)
satiated  starved
     918      168
```

That is correct. The slight discrepancy is caused by the difference of the model and empirical estimates.

Now, we will use a lognormal distribution on the same data. This means that we, once again, go back to the normal linear model. However, we will logarithmically transform the *time* variable before entering it into the **`lm`** function. When we do so, we get the following model:

$$\log(time_{ijk}) = \alpha + MATING_j + FEED_k + MATING:FEED_{jk} + \varepsilon_{ijk}, \qquad (9\text{-}11)$$
$$\text{where } \varepsilon_{ijk} \sim N(0,\sigma^2), \text{ independent among measurements.}$$

The response variable does not include zeros, therefore, we will not experience any problems during transformation. Otherwise we could, for example, add a small constant to each value. We perform the transformation within the model formula. The model will be simplified step by step, as before, but we will now display only the ANOVA table of the model with an interaction and the table of coefficients for a model with only the factor *FEED* (this is a model that would be obtained by going through the selection procedure anyway).

```
> m7<-lm(log(time)~mating*feed)
> anova(m7)
Analysis of Variance Table
Response: log(time)
```

```
            Df  Sum Sq Mean Sq F value  Pr(>F)
mating       1  11.432  11.432  2.7578 0.10547
feed         1  19.262  19.262  4.6468 0.03787 *
mating:feed  1   0.019   0.019  0.0045 0.94681
Residuals   36 149.226   4.145
...
> m8<-lm(log(time)~feed)
> summary(m8)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.4658     0.4598  11.887 2.27e-14 ***
feedstarved  -1.3879     0.6503  -2.134   0.0393 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.056 on 38 degrees of freedom
Multiple R-Squared: 0.107,      Adjusted R-squared: 0.08355
F-statistic: 4.555 on 1 and 38 DF,  p-value: 0.03933
```

Similarly to the previous analyses of these data, the effect of *FEED* is significant. The table of coefficients of `m8` includes a different p-value for *FEED* than the ANODEV table of `m6` (0.039 *versus* 0.00006). Qualitatively speaking (when insisting on the conventional 5% significance), the conclusions are the same. In the table of coefficients, the value of the first parameter is an estimate of the expected value of the log-transformed time for the level "satiated". On the second line, is the difference between expected log-transformed times for "starved" and "satiated".

For both expected values estimated by the gamma model, we will calculate 95% confidence intervals. These are obtained from a model with the textbook parametrization (by removing the intercept, i.e. adding `-1` to the linear predictor) and calling **`confint`**. As the estimates of expected values and their SE in model `m6` are on the logarithmic scale, we have to transform the intervals back, using the function **`exp`**.

```
> m9<-update(m6,~.-1)
> exp(confint(m9))
Waiting for profiling to be done...
                2.5 %    97.5 %
feedsatiated 581.0279 1574.9483
feedstarved  106.3319  288.2258
```

Finally, we will draw a plot of estimated expected values and their 95% confidence intervals from the model `m9`. Expected values will be taken from the model `m6`. We will use the outline of the plot from the function **`boxplot`**. To print the levels of the factor on the abscissa we need to specify them by the argument **`names`**. Confidence intervals will be drawn as vertical lines using **`lines`**, with arguments specifying their position with respect to the abscissa and their length, which is size of the intervals (Fig. 9-7).

```
> par(mfrow=c(1,1))
> boxplot(918,168,names=c(„Satiated","Starved"),
+ ylab="Presentation time",ylim=c(0,1600))
```

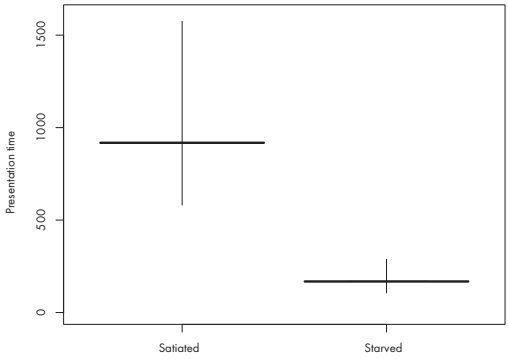


Fig. 9-7 Comparison of the mean presentation time for "satiated" and "starved" females. Vertical lines are 95% confidence intervals.

```
> lines(c(1,1),c(581.03,1574.9))
> lines(c(2,2),c(106.3,288.23))
```

CONCLUSION

The data do not support significant interaction between the two factors (GLM-g, $F_{1,36} < 0.1$, $P = 0.81$). They do not support previous mating effect as being important for the gift presentation either (GLM-g, $F_{1,38} = 0.4$, $P = 0.54$). On the other hand, female satiation shows a significant effect (GLM-g, $F_{1,37} = 18.1$, $P = 0.0001$): the presentation time of males to satiated females was 5.5 times longer than to starved females.

9.5 Two-way ANCOVA

Let us go back to the example presented in Chapter 5. As you might remember, the model based on a normal distribution was not exactly the best one. Diagnostics of this model showed violation of homoscedasticity. We indicated there that the solution could be to use a lognormal distribution assumption. We will therefore propose a model that will have the following form in the treatment parametrization:

$$\begin{aligned}\log(weight_{ijk}) &= \alpha + DIET_j + SEX_k + \beta start_i + DIET:SEX_{jk} \\ &\quad + \delta_{j1}\, start_i + \delta_{1k}\, start_i + \delta_{jk}\, start_i + \varepsilon_{ijk}\end{aligned} \tag{9-12}$$

where $\varepsilon_{ijk} \sim N(0,\sigma^2)$, independent among measurements.

The main difference from (5-2) is the logarithmic transformation of the response variable; otherwise, the linear predictor is identical to (5-2). Here, we use treatment parametrization where i is an index for an individual; j and k are indices for levels of factors *DIET* and *SEX*. α and β are the intercept and slope (coefficient of the linear trend in the explanatory variable *start*) for the combination of reference levels "ctrl-female". Differences from the intercept for

j^{th} diet and k^{th} sex (i.e. male) are represented by $DIET_j$ as the main effect, SEX_k as the main effect. Finally, $DIET{:}SEX_{jk}$ stands for interaction effects of the two factors (four effects altogether). Parameters δ_{j1}, δ_{1k} and δ_{jk} are slope differences (from the baseline slope) for the j^{th} *DIET* and the k^{th} *SEX*.

ANALYSIS

The formulation of the first model will be structurally similar to the model in Chapter 5.4. The model formula will be identical, except for the fact that the response variable *weight* will be subjected to a logarithmic transformation first. We will not include any polynomial term in *start* (but you can try that on your own).

```
> dat<-read.delim("cockroach.txt"); attach(dat); names(dat)
[1] "sex"    "diet"   "weight" "start"
> m1<-lm(log(weight)~diet*sex*start)
> anova(m1)
Analysis of Variance Table

Response: log(weight)
               Df  Sum Sq Mean Sq  F value Pr(>F)
diet            4 16.1349  4.0337 150.3981 <2e-16 ***
sex             1  0.0261  0.0261   0.9732 0.3275
start           1  0.0455  0.0455   1.6956 0.1975
diet:sex        4  0.0866  0.0217   0.8073 0.5250
diet:start      4  0.0244  0.0061   0.2272 0.9222
sex:start       1  0.0315  0.0315   1.1743 0.2825
diet:sex:start  4  0.1829  0.0457   1.7048 0.1596
Residuals      65  1.7433  0.0268
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ANOVA table is very similar to that of the model from Chapter 5.4. As before, only the influence of the *DIET* is significant. Should we simplify the model, step by step, by removing terms in accordance with the marginality rule, we will get to the following model `m7`:

```
> anova(m7)
Analysis of Variance Table

Response: log(weight)
          Df  Sum Sq Mean Sq F value    Pr(>F)
diet       4 16.1349  4.0337  150.77 < 2.2e-16 ***
Residuals 80  2.1403  0.0268
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

What has changed in comparison with the previous analysis? At a first glance, nothing important. The final model `m7` includes the same explanatory variable, i.e. *DIET*. However, the model has the following form:

$$\log(weight_{ij}) = \alpha + DIET_j + \varepsilon_{ij}\,, \qquad (9\text{-}13)$$

where $\varepsilon_{ij} \sim N(0,\sigma^2)$, independent among measurements.

The model form is quite different. Consequently, model parameters have quite different meanings. Also, their standard errors are calculated differently. Remember that the coefficients were estimated on the logarithmic scale and not on the scale of measurements – so that they mean very different things.

```
> summary(m7)
...
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.05319    0.03967  -1.341    0.184
dietlipid1    0.55181    0.05610   9.836 2.02e-15 ***
dietlipid2    0.52190    0.05610   9.303 2.23e-14 ***
dietprotein1  1.17298    0.05610  20.908  < 2e-16 ***
dietprotein2  1.12984    0.05610  20.139  < 2e-16 ***
...
```

We can thus begin attempts for combining individual levels. As before, we can combine both protein and both lipid diets. We will skip detailed outputs of the pooling procedure and will just show the final model `m9`, instead.

```
> summary(m9)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.05319    0.03940   -1.35    0.181
diet2lipid   0.53686    0.04825   11.13   <2e-16 ***
diet2prot    1.15141    0.04825   23.86   <2e-16 ***
...
> anova(m9)
Analysis of Variance Table

Response: log(weight)
          Df  Sum Sq Mean Sq F value    Pr(>F)
diet2      2 16.1115  8.0557   305.3 < 2.2e-16 ***
Residuals 82  2.1637  0.0264
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The row denoted as (Intercept) shows the estimate of the log-transformed median for "ctrl". We will find its value on the scale of measurements by applying an inverse transformation to the logarithm: exp(*–0.05319*) = *0.948*. This is not a reasonable estimate of the expected value, as for this purpose, the estimate would be biased downwards severely. To find an estimate of the expected value of *weight* on the scale of measurements for "ctrl", we have to use the formula (9-1): exp(–0.05319+0.0264/2) = *0.961*. Here variance (*0.0264*) is a residual mean square (from the ANOVA table). The estimated expected value for "ctrl" is not identical to the empirical mean calculated from non-transformed measurements, which is *0.955*.

Let us look at the first two diagnostic plots:

```
> par(mfrow=c(1,2))
> plot(m9,which=1:2)
```

Variance of the residuals does not substantially increase with the mean any more, even though it is not completely constant (Fig. 9-8A). Distribution of the residuals is also closer to normality (Fig. 9-8B). This model is therefore somewhat better than the previous model from Chapter 5.

We can present it in at least three different ways, given here in order from the simplest to the most complicated:

- Mean values on the logarithmic scale
- Medians on the original scale
- Mean values on the original scale

We will show here the second option, for which we use back-transformation of log-transformed values. Coefficients for each level of factor *DIET2* will be obtained after re-parametrization of the model using the textbook contrasts. Estimates of medians are obtained from reparametrised model by calling **`coef`** and applying the exponential function. 95% confidence intervals are obtained by calling **`confint`** and applying the exponential function.

```
> m10<-lm(log(weight)~diet2-1)
> exp(coef(m10))
diet2ctrl diet2lipid  diet2prot
0.9482024  1.6220133  2.9988437
> exp(confint(m10))
                2.5 %    97.5 %
diet2ctrl  0.8767259 1.025506
diet2lipid 1.5345691 1.714440
diet2prot  2.8371733 3.169726
```

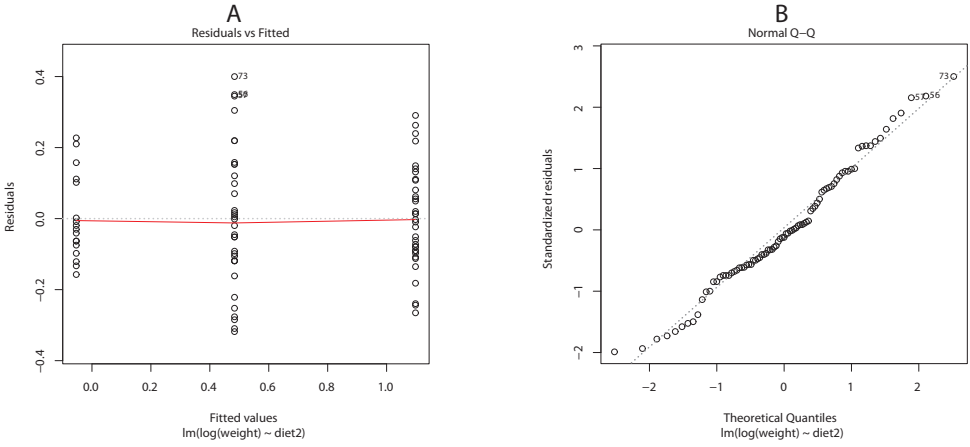


Fig. 9-8 A. Relationship between residuals and fitted values. **B**. Q-Q normal plot of standardised residuals.

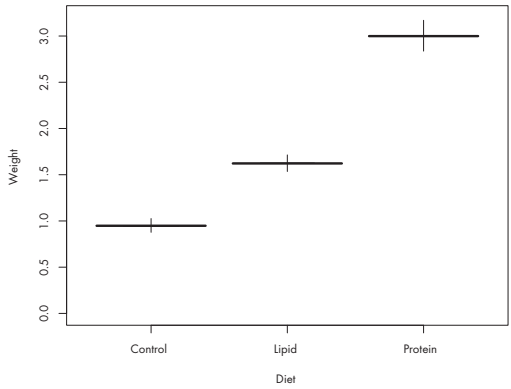


Fig. 9-9. Comparison of median weight of cockroaches on three diets. Vertical lines are 95% confidence intervals.

Finally, we can draw individual values in the plot. We will start with **`boxplot`**, then we will draw the medians using bold parallel line segments. Do not forget to specify the names of levels that will be displayed on the abscissa. Subsequently, using thin vertical lines, we will add confidence intervals using the **`lines`** function (Fig. 9-9).

```
> boxplot(0.948,1.622,2.999,names=c("Control","Lipid","Protein"),
+ ylim=c(0,3.2),ylab="Weight",xlab="Diet")
> lines(c(1,1),c(0.877,1.026))
> lines(c(2,2),c(1.535,1.714))
> lines(c(3,3),c(2.837,3.17))
```

CONCLUSION

The food type significantly influenced the final weight of the cockroaches (LM, $F_{2,82} = 305.3$, $P < 0.001$). We were not able to demonstrate an effect of initial weight (LM, $F_{1,65} = 1.7$, $P = 0.2$) or gender (LM, $F_{1,65} = 1.0$, $P = 0.33$). The protein-enriched as well as lipid-enriched diets led to a significantly higher weight of the cockroaches than the control diet (contrasts, $P < 0.0001$). The protein-enriched diet increased the weight even more than the lipid-enriched diet. We did not find any difference between the protein-enriched diet types or between the lipid-enriched diet types.

10 POISSON DISTRIBUTION

Data in the form of counts (frequencies) are common in ecology, parasitology or faunistics when studying the abundance of microorganisms, plants, fungi or animals, but they are also used in ethology when we study the occurrence of an event, such as an attack of one organism on another, hatching of eggs, consumption of provided food, etc. In all these cases, the possible count (frequency) of the organisms, other objects, events, etc. is given by non-negative integers and is often relatively low, close to zero.

Counts are often effectively arranged in contingency tables and analysed using the χ^2-test or similar methods, which you have certainly learned in your basic statistical course, so we will not deal with them here. These tests are also available in R: **χ^2-test** (**`chisq.test`**), **Fisher exact test** (**`fisher.test`**), **Mantel-Haenszel test** (**`mantelhaen.test`**), **log-linear analysis** (**`loglin`**) – plus many others. To analyse counts (frequencies) here, we will use the Poisson GLM. As in the case of other GLMs we can use it for any form of the linear predictor, i.e. with continuous, categorical or a combination of both types of explanatory variables as defined in Chapter 6.

10.1 Description of the Poisson model

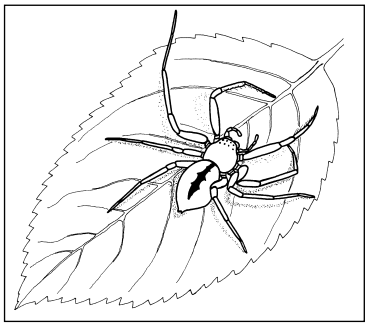
The Poisson regression procedure is invoked by calling the **`glm`** with the **`family=poisson`** argument. By this, we specify the random part of the model. In other words, we postulate that the response variable y has a Poisson distribution. This distribution is frequently parametrised by its expected value, $E(y) = \mu$. An important property of this distribution is that variance is equal to the expected value: $Var(y) = \mu$ (which amounts to quite a severe inherent heteroscedasticity).

The canonical **`link`** function is the **`log`** function. This is the default option for the Poisson GLM. Alternative links include square root (**`sqrt`**) and identity. As in all other GLMs, in the Poisson GLM, the linear predictor is related to the link-transformed expected value. This is quite different from modelling directly the expected values (with the exception of the **`link=identity`** choice). Therefore, all parameter estimates in the summary output are related to the measurements on the transformed scale. For example, an intercept (when using logarithmic link) will be related to the logarithm of the intensity (logarithm of the expected value of an expected count). Estimates of counts themselves are obtained by back-transforming the entire linear predictor using the inverse link function. For the logarithmic link, it is (of course) the exponential (**`exp`**). For **`sqrt`** link, it is the square (**`^2`**). Notice that the logarithmic link avoids any incompatibility problems. Even a negative value

of the linear predictor is back-transformed into a non-negative (the only allowable) mean parameter. This is not the case, however, with the identity (and other) links. If, for whatever reason, the linear predictor produces a negative value, we face a serious problem, because the expected value cannot be negative.

We will first introduce the Poisson analysis using an example with a single categorical variable. Such a model is effectively a more sophisticated (and in the case of low counts far more appropriate) analogue of the one-way ANOVA.

10.2 One-way ANODEV



Species richness changes with the age of the habitat. The species richness first increases and then may decreases with the habitat age, thus being highest for medium values of habitat age. In 15 apple orchards, species richness of arachnids occurring on trees was studied. The resulting data are in the form of number of species (*divers*). The age of individual orchards could not be determined precisely, therefore, it was categorised. The orchards were classified into three age classes (*ORCHARD*: young, older, oldest). The young category included orchards with trees 0-9 years old; older included orchards with trees 10-19 years old, and oldest included orchards with trees 20-30 years old. Each class was represented by five orchards in the study. We want to answer the following questions: (1) Is the species richness related to the age of orchards? (2) If the answer to the first question is affirmative, what kind of trend does the species richness show across the age categories?

EDA

As the dataset is not excessively large, we will practice typing the data in R. At first, we will create a vector with the name `divers` for recorded values of diversity calling the concatenation function **`c`**. The first five age levels will be "young", followed by five "older" and five "oldest" levels. This character vector will be first converted to a factor (by the **`factor`** function) and stored in the variable `orchard` subsequently. Note that, by default, the factor will be assigned the treatment parametrization. Therefore, all analyses done with it will be implicitly based on the treatment parametrization (unless another parametrization is invoked explicitly by specifying another type of contrast). This behaviour can be changed globally in the **`options`** function, if desired. Before we plot the empirical expected values (arithmetic averages) into a box plot, we need to reorder factor levels from "young" to "oldest" by calling **`relevel`**.

```
> divers<-c(9,6,8,13,10,21,14,26,17,29,15,17,12,10,11)
> orchard<-factor(c(rep("young",5),rep("older",5),rep("oldest",5)))
> orchard<-relevel(orchard,ref="young")
> plot(orchard,divers)
```

In "young" orchards, the average species richness was the lowest (Fig. 10-1). In the "oldest" orchards, it was greater, and in "older" orchards, it was the greatest.

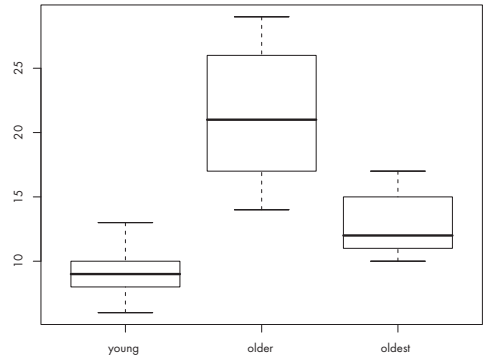


Fig. 10-1 Box plot of species richness of arachnids in three orchards classes ("young", "older", and "oldest").

MODEL

A previous study on spiders (Hurd & Fagan 1992) demonstrated that diversity changes with the age of a given habitat. Since we know the age of the study orchards only approximately, as levels of the *ORCHARD* factor, we assume the following model with categorised orchard age taken as a factor:

$$\log(\mu_j) = \alpha + ORCHARD_j \,, \qquad (10\text{-}1)$$

$$\text{where } divers_j \sim Poi(\mu_j), \text{ independent among orchards.}$$

As we know from earlier chapters, in the (default) treatment parametrization, α represents the mean value for "young" and $ORCHARD_j$ represents the log-mean differences for a j^{th} level of *ORCHARD* with "young" as the reference level.

ANALYSIS

We can certainly perform the analysis in the default treatment parametrization, as many times before. As we saw from the interpretation of the *ORCHARD* factor effects just above, that might not be the most convenient choice. If we are interested in gradual change to the species richness, from the youngest to the oldest orchards, it will pay off to use Helmert contrasts (see Chapter 5.4.5) instead. The first Helmert contrast will compare log-mean diversity of "young" with "older". The second contrast will compare of the "oldest" category with the average of the log-mean diversities for the previous categories ("young" and "older"). At first, we need to find out whether the effect of the factor *ORCHARD* is significant. The test result can be read from the ANODEV table with the χ^2-statistics. Note that for the overall test of a factor, it does not matter which kind of parametrization we use. The test outcome will always be the same (you can check this by setting different parametrizations and comparing the test results). Thus we use the treatment contrasts (the default).

```
> m1<-glm(divers~orchard,family=poisson)
> anova(m1,test="Chi")
```

```
Analysis of Deviance Table

Model: poisson, link: log

Response: divers

Terms added sequentially (first to last)


         Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                        14     38.964
orchard  2   26.246         12     12.718 1.999e-06
> summary(m1)

Call:
glm(formula = divers ~ orchard, family = poisson)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.70841  -0.71876  -0.08674   0.75146   1.55772

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)     2.2192     0.1474  15.051  < 2e-16 ***
orchardolder    0.8442     0.1763   4.788 1.68e-06 ***
orchardoldest   0.3457     0.1927   1.794   0.0727 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 38.964  on 14  degrees of freedom
Residual deviance: 12.718  on 12  degrees of freedom
AIC: 85.296

Number of Fisher Scoring iterations: 4
```

The effect of *ORCHARD* is highly significant. As we are now working in the treatment parametrization, you should know that in the table of coefficients, Intercept stands for the estimate of the mean value of the orchard species richness for the reference level, "young", on a logarithmic scale (because we have selected the default, i.e. logarithmic link). If we exponentiate the estimate, we will get its value on the original scale as the expected value of the species count. The other coefficients are of course the differences from the first one – all on the logarithmic scale. By exponentiating, we will determine the ratio between species richness at a given category and species richness at the reference level.

Note that instead of the t-values, we now see z-values in the output. This is because the test results, including SEs, are only approximate – asymptotic (there is no theory that describes the exact behaviour of the test statistics for finite data sets). In these cases, when testing the null hypotheses of zero parameter values, z-ratios (parameter estimate divided by its SE) are computed and compared to the asymptotic normal distribution, instead of the t-distribution (which we encountered in **`lm`**).

Now, we will fit the model using the Helmert parametrization (Chapter 5.4.5).

```
> contrasts(orchard)<-'contr.helmert'
> m2<-glm(divers~orchard,family=poisson)
> summary(m2)
...
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.61585    0.07186  36.404  < 2e-16 ***
orchard1     0.42209    0.08815   4.788 1.68e-06 ***
orchard2    -0.02545    0.05072  -0.502    0.616
...
```

The table of coefficients shows three parameter estimates. The first one (*2.616*), is the grand mean on the logarithmic scale. The first contrast (orchard1) shows that the richness of "older" orchards is significantly higher than that of the "young" orchards. The second contrast (orchard2) reveals that the richness of the "oldest" orchards is lower than the average of expected values for "young" and "older", but the difference is not significant (p-value = 0.616). This suggests that a dramatic change in the species richness takes place between 10th and 20th year of orchard age and that the change in arachnid richness between 0^{th} and 30^{th} year has a humped shape. If we were to need a more precise model of the species richness change, we would need to know the exact age of each orchard (or at least to have a more detailed categorisation of orchard age).

Although the levels in *ORCHARD* are ordered, the function **`glm`** does not know about it and treated the factor as unordered. This is a pity, because we could make use of such a property in the analysis. So we will make a copy of *ORCHARD* and recognise the ordered levels by calling **`ordered`**. Then we will refit the model. By default, orthogonal **polynomial contrasts** are used for models with ordered factors.

```
> orchard1<-ordered(orchard)
> m3<-glm(divers~orchard1,family=poisson)
> summary(m3)
...
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.61585    0.07186  36.404  < 2e-16 ***
orchard1.L   0.24448    0.13624   1.794   0.0727 .
orchard1.Q  -0.54813    0.11144  -4.919 8.71e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
...
```

The coefficients in the output look different than before. There are still three parameters, but these correspond to orthogonal polynomial trend components. The highest polynomial order is always one less the number of levels of the factor. a polynomial function is fitted here under the assumption that distances between levels are identical (recall that the levels cover range of years and are separated by similar distance). The output lists estimates in the natural order: the intercept is the first, then the linear term coefficient and finally the coefficient of the quadratic term. In contrast to the Helmert contrasts, only the last coefficient is significant. This analysis shows that there is a humped trend with a significant decline for the oldest age.

One more thing we should do is to inspect the model's adequacy. We will skip the graphics now. If you plot the diagnostics, you will see that variance of deviance residuals is quite homogenous and their distribution does not show any serious deviations from normality. There are only a few data points, so it is really quite hard to detect departures from assumptions.

Finally, we will estimate the 95% confidence intervals of the mean values for all levels of the *ORCHARD*. For this purpose, we will use an approximation (3-4). In order to do so, we need to know the mean value and its standard error for each level. We get them from the textbook parametrization. That can be easily obtained if we re-fit the model with the *ORCHARD* factor and no intercept. To that end, we use the **`-1`** notation within the linear predictor.

```
> m4<-glm(divers~orchard-1,poisson)
> summary(m4)
...
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
orchardyoung   2.21920    0.14744   15.05   <2e-16 ***
orchardolder   3.06339    0.09667   31.69   <2e-16 ***
orchardoldest  2.56495    0.12403   20.68   <2e-16 ***
...
```

The table now shows logarithms of mean values for all three *ORCHARD* levels, together with their standard errors. To compute confidence intervals, we use the **`confint`** function. We will, calculate them on the log-transformed scale and exponentiate their limits.

```
> exp(confint(m3))
Waiting for profiling to be done...
                  2.5 %   97.5 %
orchardyoung   6.790864 12.12010
orchardolder  17.597063 25.71441
orchardoldest 10.090235 16.42096
```

We will draw the estimated expected values and their 95% confidence intervals on a bar plot.

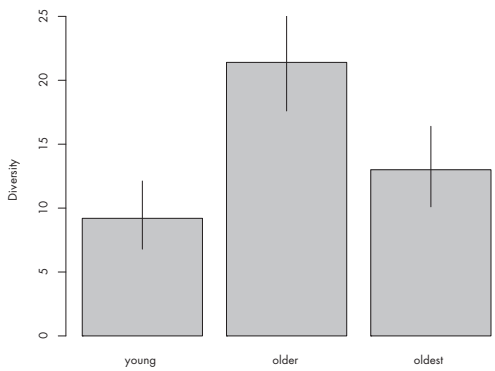


Fig. 10-2 Bar plot of mean species richness of arachnids for three age orchards levels ("young", "older", and "oldest"). 95% confidence intervals for the means are plotted as vertical lines.

We will obtain the predicted values from the first model when calling the **predict** function with the **type="response"** argument. This way, we get estimates on the original response scale (i.e. as mean counts). We will draw the 95% confidence intervals separately for each level as vertical line segments using the **lines** function (Fig. 10-2).

```
> barplot(tapply(predict(m1,type="response"),orchard,mean),
+ ylab="Diversity",ylim=c(0,25))
> lines(c(0.7,0.7),c(6.79,12.12))
> lines(c(1.9,1.9),c(17.6,25.7))
> lines(c(3.1,3.1),c(10.1,16.4))
```

CONCLUSION

The data show that arachnid species richness changes significantly with the orchard age (GLM-p, $\chi^2_2 = 26.3$, $P < 0.0001$). The change is not monotonous. It seems that richness first increases, reaching the maximum somewhere between 10 and 20 years, and then it declines.

10.3 Overdispersion and underdispersion

Even though we assume that observations come from the Poisson distribution when we use the Poisson GLM, it is often not entirely so. The Poisson distribution is a very parsimonious distributional model. It has only one parameter (mean), unlike many commonly used continuous distributional models (normal, lognormal, gamma) which have two parameters. That is very convenient and simple, but, naturally, it does not suit all situations.

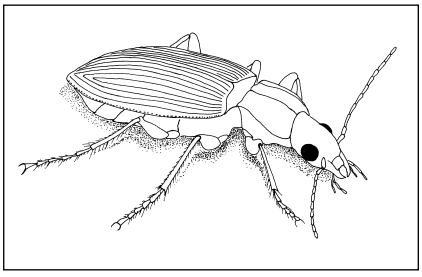
There are many data sets which behave similarly to the Poisson model, but their variance is substantially larger than the mean (unlike the Poisson model, which requires that the mean value is equal to variance). Such a phenomenon is called **overdispersion**. On the other hand, when variance is smaller than the mean value, we deal with **underdispersion**. In these cases, the value of the dispersion parameter (ϕ), computed as the ratio of $\mathrm{Var}(y)/\mathrm{E}(y)$, is not equal to 1, it is greater for overdispersion (or smaller for underdispersion). We may notice the problem if the residual variance is *much* larger (or *much* smaller) than the residual degrees of freedom. In an ideal case, both values should be similar, but, be careful: this is just an approximate comparison – residual deviance has relatively high variability (in many situations it behaves approximately as a multiple of the χ^2 random variable)!

There are several possible explanations for overdispersion. Some of them are related to the details of a studied process generating the data. Then the over/underdispersion can be of direct interest. In biological studies, overdispersion can be related to aggregation (as an opposite to purely random spatial dispersal), for example. Generally speaking, the overdispersion can be caused by a small-scale phenomenon which the standard Poisson model does not acknowledge. For instance, the mean (μ) parameter might not be constant and may change randomly instead. Overdispersion may also have a much more prosaic explanation, however; it can occur as an artefact, as a result of mistakes – for example, if we omit an important explanatory variable in our model or as a result of gross errors committed during data entry, etc.

Acknowledging overdispersion and underdispersion is rather easy in **glm**. Instead of **family=poisson**, we use the argument **family=quasipoisson**. This does not fully specify a legal distribution type, but it affects second moment specification (variance). It will now no longer be fixed (being equal to the mean), but it is estimated from the data using a dispersion parameter, ϕ. This new corrected variance is important for testing as it accounts for higher dispersion in the data. From a pragmatic point of view, it prevents a surplus of false positive results which would otherwise occur if the standard Poisson were used under overdispersion. The value of ϕ is calculated from the Pearson residuals and the residual degrees of freedom. Use of the **quasipoisson** does not affect coefficient estimates. On the other hand, it affects, often quite dramatically, estimates of their standard errors. Therefore, test statistics and quite possibly also test conclusions about parameters can be affected. Specifically, the value of a standard error is multiplied by the estimate of $\sqrt{\phi}$. This means that in case of overdispersion (when $\phi < 1$) the value of SE will increase, while in case of underdispersion (when $\phi < 1$), it will decrease. p-values will increase (or decrease) consequently. Z-statistics will be automatically replaced in the summary output by t-statistics. And χ^2-statistics for the ANODEV table will have to be replaced manually by F-statistics. If the overdispersion is large, an effect which was significant under the wrong (standard Poisson) model might easily become non-significant after the correction for overdispersion. And that is the main point: if we do not correct for large overdispersion, p-values misleadingly favour alternative hypotheses. In other words, neglecting overdispersion leads to an excessively large false positive finding rate (it can be much larger than the nominal significance level).

We will explain overdispersion as well as underdispersion in the following two chapters. There, we will show how they occur and how to deal with them.

10.4 Multiple regression



The abundance of carabid beetles in cereal monocultures depends on many abiotic and biotic variables. Pitfall traps were used to obtain the abundance of carabid beetles (*abun*) in wheat plots. One trap per plot was exposed for one month. Air temperature [°C] and insolation [W/m^2] were also measured on each plot during the trap exposure period. From the recorded data, average temperature (*temp*) and average insolation (*sun*) was computed for the exposure period. Now we are interested in addressing the following two questions: (1) Was the abundance of beetles affected by any of the two variables? (2) If it was affected significantly, what might be a suitable model of the relationship?

EDA

We have two continuous explanatory variables. We need to focus on the relationship between *abun* and *temp* and *sun* as well as on the mutual relationship between explanatory variables. For this purpose, we will use the **pairs** function with the **panel** argument. It will draw scatter plots for all pairs of variables supplied. It will also add a loess curve to each

plot as a flexible, non-parametric estimate of the possible trend. The main argument of the **`pairs`** function is the model formula.

```
> dat<-read.delim("carabid.txt"); attach(dat); dat
          temp  sun abun
1  9.928571429 1160   23
2  14.52857143 1297   12
3  14.35714286  930   24
...
21 27.71428571 3327  170
> pairs(abun~temp+sun,panel=panel.smooth)
```

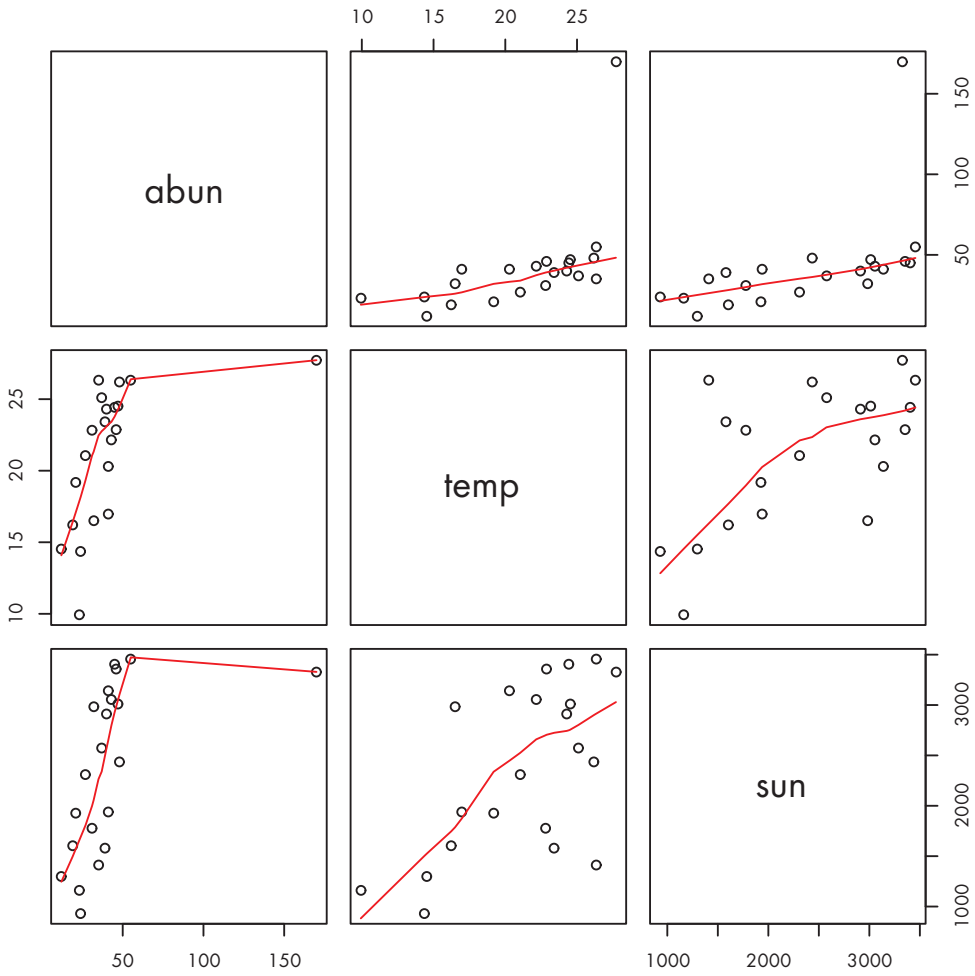


Fig. 10-3 Scatter plots for all pairs of available variables (*abun*, *temp*, and *sun*) with superimposed loess fits.

From the matrix of the scatter plots (Fig. 10-3), we are mainly interested in the first row. The plot in the middle of this row shows the relationship between abundance (*abun*) and temperature (*temp*). We can see a slightly increasing trend there. The plot on the right shows the relationship between abundance of carabid beetles and insolation (*sun*). The trend seems to be also increasing there. a clear outlier can be spotted at the upper right corner of both plots. We have to be careful about it. When assessing plots that represent counts and trying to find suspiciously large values, we need to be generally quite careful, because of the inherently large variance, which in the Poisson distribution is as big as the mean value (and it is even worse in the case of overdispersion). The *optical* assessment is not nearly as easy as it is in the case of a homoscedastic normal distribution (Chapter 8). Note that due to the inherent Poisson heteroscedasticity, data with different means also have different weights in the model fitting. Particularly, the larger the observation, the smaller the weight – and this is not easy to see when exploring the plot. Fortunately, the Poisson regression procedure takes this complication into account, correctly and automatically.

In the third plot in the second row, we can see that the explanatory variables are not mutually independent. There is a hint of a linear relation among them. More sun usually also means a larger temperature. That is why it will be somewhat harder for us to separate the "pure" effect of *sun* and "pure" effect of *temp* for the purposes of estimating, testing and interpreting the data.

MODEL

It is a known fact that temperature has a positive influence on the activity of carabid beetles. Sunlight will probably have a similar influence. We expect that the model will combine both variables additively on the logarithmic scale (hence multiplicatively on the original scale). If we consider only the linear effects of the temperature and insolation, we get the following model for the i^{th} observation:

$$\log(\mu_i) = \alpha + \beta_1 temp_i + \beta_2 sun_i \,, \qquad \text{where } abun_i \sim Poi(\mu_i), \text{ independent among plots.} \tag{10-2}$$

ANALYSIS

When searching for a suitable model, we will start with a model including both main effects and interaction. In fact, we take only the simplest part of the saturated interaction in the ANOVA sense, namely the product of both linear terms:

$$\log(\mu_i) = \alpha + \beta_1 temp_i + \beta_2 sun_i + \delta temp_i sun_i, \qquad \text{where } abun_i \sim Poi(\mu_i), \text{ independent among plots.} \tag{10-3}$$

```
> m1<-glm(abun~temp*sun,family=poisson)
> summary(m1)
...
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.195e+00  4.745e-01   8.840  < 2e-16 ***
temp        -5.386e-02  2.258e-02  -2.385   0.0171 *
```

```
sun         -1.151e-03  2.364e-04  -4.869 1.12e-06 ***
temp:sun     6.257e-05  1.006e-05   6.221 4.95e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 317.229  on 20  degrees of freedom
Residual deviance:  98.657  on 17  degrees of freedom
AIC: 220.55

Number of Fisher Scoring iterations: 4
```

The table of coefficients is quite similar to the table we saw in the multiple regression example (Chapter 8.4 where we used LM), but here, due to the implicitly invoked log link (it is the canonical and hence default link for Poisson GLM), all coefficients are on the logarithmic scale. In the table, the estimate for the intercept (a) appears first, followed by the slope estimate for *temp* (b_1), the slope estimate for *sun* (b_2), and the estimate of the interaction coefficient (d). Partial tests listed in the table compare their differences from zero. We can see that all coefficients are significantly different from zero. At the bottom of the table, residual deviance and residual degrees of freedom are listed. The value of the first (*98.66*) is almost 6 times larger than the value of the residual degrees of freedom (*17*). This indicates overdispersion! The variance of abundance of beetles seems to be much larger than its expected value. This may mean that the abundances do not come from the Poisson distribution but from another, more complicated distribution (e.g. due to natural aggregation of beetles, or due to the nature of the sampling procedure), or, it might mean that we have not recorded some important explanatory variable! A pragmatic approach to correct for overdispersion is to use **`quasipoisson`**. This will correct standard errors of model coefficient estimates. These are then used in the test statistics. For testing, we use an analogue of F-statistics in the ANODEV table.

```
> m2<-update(m1,family=quasipoisson)
> anova(m2,test="F")
...
         Df Deviance Resid. Df Resid. Dev       F    Pr(>F)
NULL                        20     317.23
temp      1   153.10        19     164.12 24.5836 0.0001196 ***
sun       1    27.90        18     136.23  4.4796 0.0493541 *
temp:sun  1    37.57        17      98.66  6.0324 0.0251002 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

All terms in the model are significant even after the correction for overdispersion. Model `m2` cannot be simplified and that is why we will now assess its adequacy using diagnostic plots from the **`plot`** function (called with the fitted model object as an argument). Remember that residuals should not demonstrate any decisive trend in relation to the fitted values or to any of the explanatory variables. When exploring the behaviour of residuals against the explanatory variables, we use the Pearson residuals. They adjust for the inherent heteroscedasticity of non-standardised residuals, so that they are easier to read. Variance of the Pearson residuals should be *approximately* constant (or, at least, it should not significantly change

with explanatory variables). We will also explore whether there are any outliers by means of Cook's distances.

```
> par(mfrow=c(2,2))
> plot(m2,which=c(1,4))
> pr<-resid(m2,type="pearson")
> plot(sun,pr); plot(temp,pr)
```

Deviance residuals (4-15) plotted against fitted values show a marked trend (Fig. 10-4A). Neither plot Fig. 10-4C nor 10-4D look correct. From a detailed look at the first plot (Fig. 10-4A), we identify an outlier in the upper right corner of the plot. This value is an outlier in the plot of Cook's distances too (Fig. 10-4B) – where reaches a value of 3. Recall that mea-

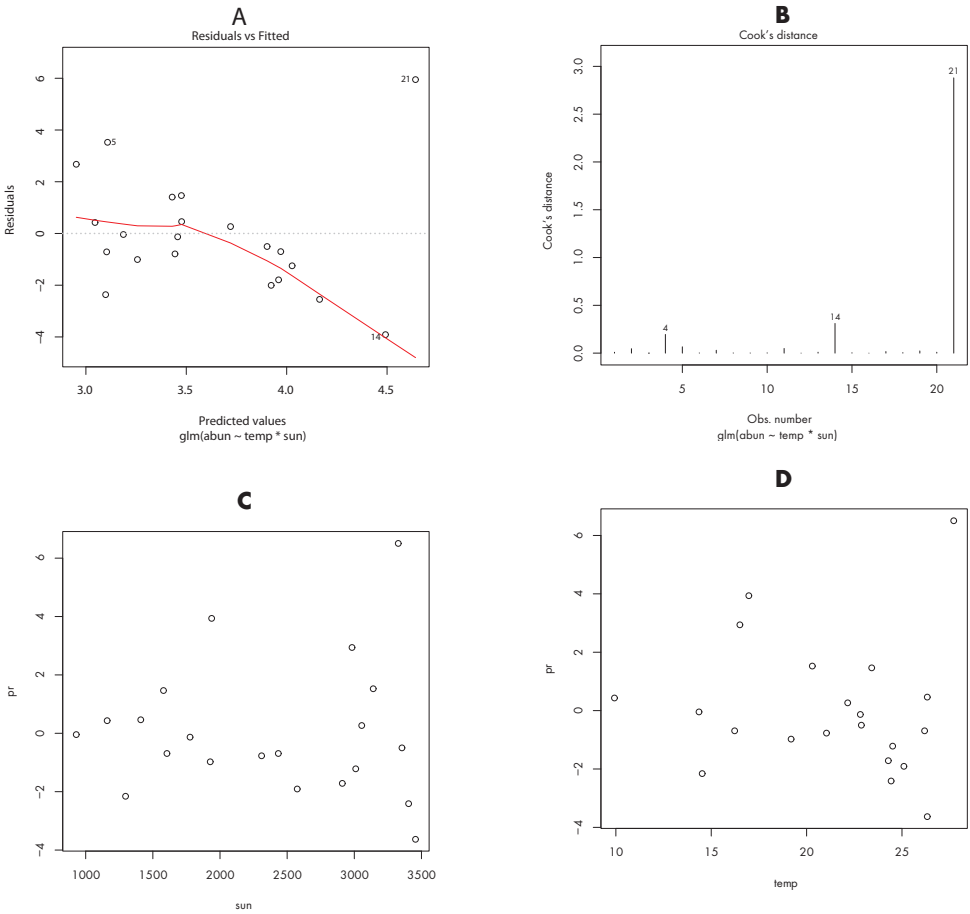


Fig. 10-4 A. Relationship between deviance residuals and fitted values. **B**. Cook's distances of all measurements. **C**. Relationship between Pearson residuals and the explanatory variable *sun*. **D**. Relationship between Pearson residuals and the explanatory variable *temp*.

surements with Cook's distances larger than 1 are suspicious. It is a measurement number 21. What is its value?

```
> abun[21]
[1] 170
```

170? This must be a mistake, because so many beetles could not fit into a single trap. We check the protocols. It turns out that this error is already in the protocols and thus it did not arise during typing into the file. As we do not know what the real value was (maybe it was *17*), we cannot do anything else other than omitting it from the analysis. So we do the modelling once again after omitting measurement 21! A data point(s) can be removed from the data frame before calling the **glm** function. Alternatively, and sometimes more conveniently, the removal can be achieved within the **glm** call, using the **subset** argument. We use the minus sign as a shortcut for "do not include". Generally, we can omit more values using the same argument (or a vector of omitted measurements, e.g. **-c(21,14)**). After fitting a new model we will sequentially remove terms which are not siginificant at the 5 % level.

```
> m3<-glm(abun~temp*sun,poisson,subset=-21)
> anova(m3,test="Chi")
...
         Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                        19     75.292
temp      1   40.291        18     35.001 2.188e-10
sun       1   12.165        17     22.836 4.870e-04
temp:sun  1    0.117        16     22.719     0.732
> m4<-update(m3,~.-temp:sun)
> anova(m4,test="Chi")
...
     Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                    19     75.292
temp  1   40.291        18     35.001 2.188e-10
sun   1   12.165        17     22.836 4.870e-04
> summary(m4)
...
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 2.283e+00  2.088e-01  10.933  < 2e-16 ***
temp        3.781e-02  1.070e-02   3.534 0.000409 ***
sun         1.954e-04  5.655e-05   3.455 0.000550 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 75.292  on 19  degrees of freedom
Residual deviance: 22.836  on 17  degrees of freedom
AIC: 135.76

Number of Fisher Scoring iterations: 4
```

The interaction is not significant in model `m3`, unlike in the model fitted on the complete data (including the outlier). Both main effects, *sun* and *temp*, have remained significant

after the outlier removal, however. We now arrive at the model that we originally anticipated (10-2). The three rows in the table of coefficients are parameter estimates: a, b_1, and b_2.

By omitting the outlier, we have substantially reduced the null deviance from the original value of *317* to about one fourth of its original value (*75*). How much variability does this model explain? That is not as easy to answer as it might seem. We can try to look at a very simple analogue of the coefficient of determination, namely at the ratio of the explained to the residual deviances:

```
> (75.292-22.836)/75.292
[1] 0.6967008
```

Model `m4` explains almost 70% of the deviance.

What about overdispersion? It has essentially disappeared! The residual deviance (*22.8*) is just slightly higher than the residual degrees of freedom (*17*). This means that the excessive variability in model `m1`, was in fact caused by the presence of a single outlier! This kind of thing can happen and that is why we have to be careful when working with outliers. The value of the dispersion parameter of model `m4`, which you can determine after omitting observation 21 by using **`glm`** with the **`quasipoisson`** argument, is *1.3*. Such a low overdispersion does not need to be adjusted for. We will, therefore, stick with the standard Poisson model.

To save space, we will not show the diagnostics of model `m4` here but you should look at the diagnostic plots by yourself. We can just state that the behaviour of the residuals has been adjusted and that they do not suggest significant deviances from the assumptions any more. We, therefore, accept model `m4`.

We will draw estimates of the mean values of the abundance of carabids from model `m4` into a three-dimensional plot, because we have two continuous explanatory variables. The graphical display of such a model is a curved surface. We will draw it using just a few commands, which we have not seen so far. First of all, we determine the range of the values of both explanatory variables. Next, we use these values for defining the range of the plot's axes. To do so, we call the **`expand.grid`** function and save the result in a new object called `xyz`. This is a data frame. Later on, we add a new column called `density` to this object. It will contain fitted values of carabid beetle density from model `m4`. In order to get the mean values on the original (not logarithmic) scale, we will use the argument **`type="response"`**. Finally, using the **`wireframe`** function from the lattice package, we will draw the curved surface to the plot. The main argument of the **`wireframe`** function is the model formula. By applying the **`gray`** argument, we will draw the surface using grey shades (Fig. 10-5).

```
> par(mfrow=c(1,1))
> range(sun)
[1]  930 3456
> range(temp)
[1]  9.928571 27.714286
> xyz<-expand.grid(sun=seq(900,3500,50),temp=seq(9,30,0.5))
```

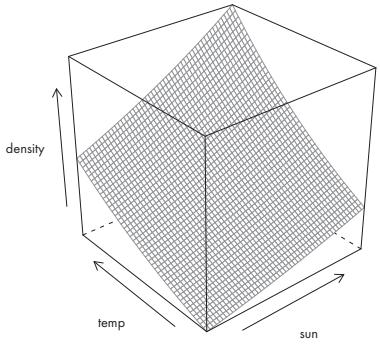


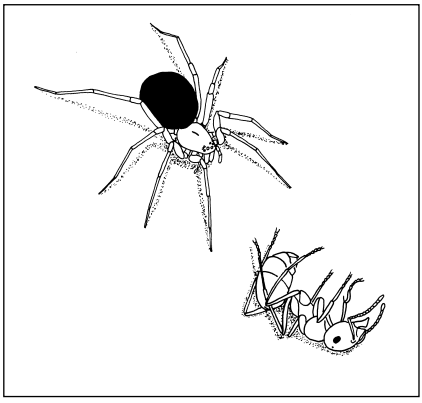
Fig. 10-5 3D plot of the final model showing relationship between carabid density response and two explanatory variables: temperature (*temp*) and insolation (*sun*).

```
> xyz$density<-as.vector(predict(m4,xyz,type="response"))
> library(lattice)
> wireframe(density~sun+temp,xyz,col=gray(0.5))
```

CONCLUSION

The abundance of carabid beetles increased significantly with temperature (GLM-p, χ^2_1 = 40.3, P < 0.0001) as well as with insolation (GLM-p, χ^2_1 = 12.2, P = 0.0005). Their effect seems to be multiplicative (additive on the log scale). The greatest density of carabid beetles was in open areas, i.e. locations with a high temperature and strong sun activity. Using our model, we can get the carabid density estimate by substituting particular values for *sun* and *temp* into a simple formula: $\exp(2.283 + 0.038\,sun + 0.0002\,temp)$.

10.5 One-way ANCODEV



Some spiders specialise on a certain prey species. Ant-eating spiders evolved a specific capture behaviour. Their prey-capture success can be measured by the number of attacks needed to catch prey. In the laboratory, the number of attacks (*number*) of one ant-eating spider species on ants of two subfamilies (*ANT*: famA, famB) was observed. For each ant subfamily, 20 ant species, each represented by a single individual, were used. Body size of each ant (*size*) was recorded. We were interested in finding whether the ant-eating spider species has a higher capture efficiency for any subfamily. Specifically, we wanted to know: (1) Is the number of attacks related to ant size? (2) Is the number of

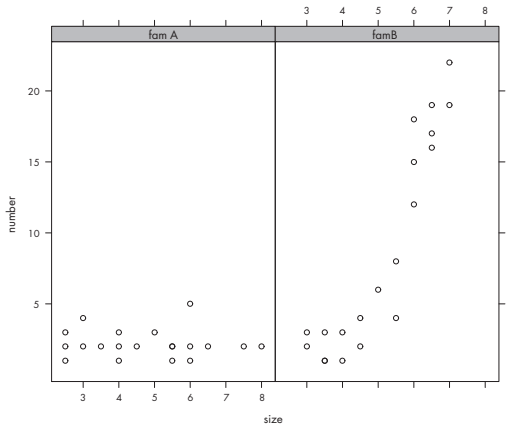


Fig. 10-6 Relationship between the *number* of attacks and body *size* of ants belonging to two subfamilies: "famA" and "famB".

attacks similar for ants of both subfamilies? (3) What could be a possible model of the relationships detected?

EDA

The data contain a response variable (*number*) and two explanatory variables. One explanatory variable is categorical (*ANT*) and the other is continuous (*size*). While the *ANT* variable records the ant species from two subfamilies, *size* records the body size [mm]. For the initial inspection of the data, we will use scatter plot from the lattice library. We will use the **`xyplot`** function in order to draw two panels with the relationship between *number* and *size* shown separately for the two ant subfamilies.

```
> dat<-read.delim("anteater.txt"); attach(dat); dat
   size number  ant
1   6.0      1 famA
2   5.0      3 famA
3   7.0      2 famA
...
40  7.0     19 famB
> library(lattice)
> xyplot(number~size|ant,col=1)
```

It is immediately clear that the relationship between the number of attacks and body size of ants is not the same for both subfamilies (Fig. 10-6). Overall, the number of attacks for "famB" is several times greater than for "famA". While the number of attacks for "famA" is more or less constant for all body sizes of ants, it rapidly increases with size for "famB". The effect of the *size* covariate is thus important. Since the trends are quite different for the two subfamilies, we expect a significant effect from the interaction between the categorical and continuous explanatory variables.

MODEL

We assume that the number of attacks depends not only on the subfamily but also on the body size of ants (*size*). This is because experiments with other predators have demonstrated that catching bigger prey requires a larger number of attacks (Hjelm & Persson 2001). Therefore, we expect that the number of attacks will change exponentially with changing body size of the ants (leading to a linear model on the log scale). We will allow for both intercept and slope to be different for the two subfamilies. The model will then be:

$$\log(\mu_{ij}) = \alpha + ANT_j + \beta size_i + \delta_j size_i \,, \qquad (10\text{-}4)$$

$$\text{where } number_{ij} \sim Poi(\mu_{ij}), \text{ independent among measurements.}$$

In the treatment parametrization, α and β represent intercept and slope for "famA" respectively. ANT_j is the difference between intercept of "famB" and "famA". δ_j is the difference between slope for "famB" and "famA". The reader may recognise that the systematic part can be perceived as an analogue of the analysis of covariance model (it includes interaction between the factor and a covariate).

ANALYSIS

We will start with model (10-4), which will include both main effects and the interaction in the model formula.

```
> m1<-glm(number~size*ant,family=poisson)
> anova(m1,test="Chi")
...
          Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                         39    215.561
size       1   93.395        38    122.167 4.284e-22
ant        1   75.555        37     46.612 3.554e-18
size:ant   1   25.804        36     20.808 3.779e-07
```

It is clear from the last row of the ANODEV table that the interaction is highly significant. This means that the model cannot be simplified. Nevertheless, before we accept it, we need to find out whether it is not suffering from a strong overdispersion.

```
> summary(m1)
...
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.89794    0.64904   1.383 0.166512
size         -0.02154    0.12456  -0.173 0.862735
antfamB      -2.66924    0.80637  -3.310 0.000932 ***
size:antfamB  0.70407    0.14579   4.829 1.37e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 215.561  on 39  degrees of freedom
Residual deviance:  20.808  on 36  degrees of freedom
AIC: 153.15
...
```

Surprisingly, the residual deviance (*20.81*) is smaller than the residual degrees of freedom (*36*). This suggests that variance of the number of attacks is not larger than the mean. It might be even smaller than the mean. The value of the dispersion parameter (which can be found by re-fitting the model with **quasipoisson**) is *0.61*. This is not an overwhelmingly decisive argument for underdispersion. If we do not want to consider (very unlikely) underdispersion, we should leave the model as it is. Note that correcting for underdispersion would increase the significance of effects which would be anticonservative – and this could be dangerous.

In the summary output, there are four coefficients – four parameters as defined in (10-4) (recall that, by default, we used treatment contrasts). The first line is the intercept estimate for "famA" and the second is the slope for "famA". The intercept for "famB" is obtained by adding the first and the third coefficients and the slope for this subfamily is obtained by adding the second and the fourth coefficients. Values of the first two coefficients (*0.898* and *–0.022*) are not significantly different from zero. This means that the estimated model for "famA" is virtually parallel with the abscissa, intersecting the ordinate at exp(0.898) = *2.455*. Both parameters for "famB" are significantly different from those for "famA". Treatment parametrization is very convenient here as it tests the differences between the two levels, but it is not so useful for obtaining estimates of expected values for combinations of explanatory variables. This is not only because we have to sum the coefficients to obtain intercept and slope for "famB" subfamily, but also because we do not get their standard errors easily. They can still be obtained from the model fitted in the treatment parametrization, but it would require several manipulations with the (asymptotic) covariance matrix of the estimates – after extracting it by calling **summary(m1)$cov.scaled**. It is much more convenient to switch to the textbook parametrization and obtain the expected values and their standard errors from there. We can fit the same model in the treatment parametrization if we specify the model formula as: **number~-1+ant+size%in%ant**.

Is the model **m1** adequate? Let us look how the residuals behave, i.e. let us explore variance of the deviance residuals and the relationship between the Pearson residuals and the continuous explanatory variable *size*.

```
> plot(m1,which=1)
> pr<-resid(m1,type="pearson"); plot(size,pr)
```

The relationship between deviance residuals (4-15) and fitted values indicates the presence of an unexplained trend – the model overestimates the number of attacks (i.e. predicts a higher number than observed) for ants of median size. Otherwise, the variance is quite homogenous (Fig. 10-7A). Pearson residuals (4-18) do not appear to be related to *size* (Fig. 10-7B).

The curved pattern of residuals might be corrected when we include a quadratic term for the covariate *size* in the model. Let's do so – we will include the quadratic term via **poly** and then we will compare the two models using χ^2-test.

```
> m2<-glm(number~poly(size,2)*ant,poisson)
> anova(m1,m2,test="Chi")
Analysis of Deviance Table
```

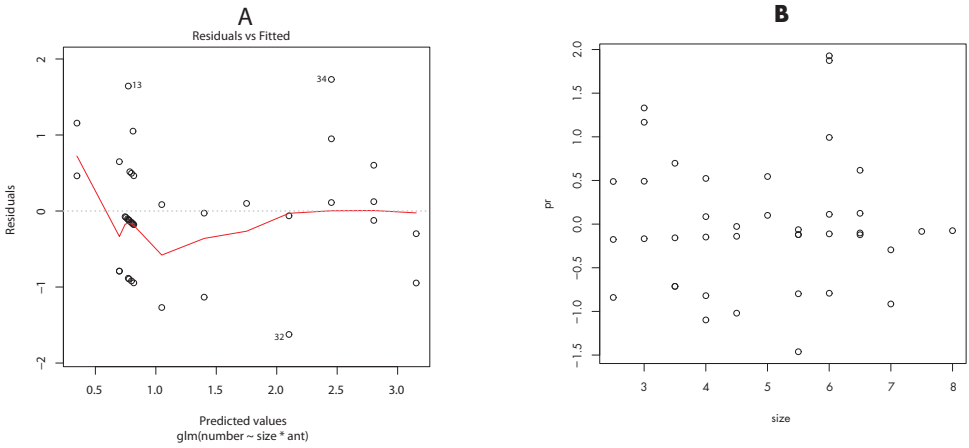


Fig. 10-7 A. Relationship between deviance residuals and fitted values. **B**. Relationship between Pearson residuals and the continuous explanatory variable *size*.

```
Model 1: number ~ size * ant
Model 2: number ~ poly(size, 2) * ant
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        36    20.8084
2        34    20.7673  2   0.0411    0.9797
```

The quadratic term has caused only a negligible change in deviance and it has not improved the model significantly. If you look at the relationship between the deviance residuals and the predicted values of model `m2`, you will discover that there is still overestimation there. Would it help if we replaced the link function with a square root function? Try it for yourself and you will see (do not forget to check how such an estimate would correspond to the measured data).

We will plot curves for both subfamilies from `m1` to one figure. We have to use several functions and their arguments, so we will describe them only briefly. First, we draw an empty plot with axis labels. Then, using different symbols, we will plot data separately for each factor level. Using **`lines`**, we will superimpose the model fit, again separately for different subfamilies, one after another. We have to specify the argument **`type="response"`** in order to get the fit at the original scale (i.e. counts) from the **`predict`** function. Otherwise we would get estimates on the link-transformed, i.e. logarithmic, scale. As the model `m1` contains parameters for two curves, it is necessary to use the argument **`list`** to specify, which parameters belong to which level. Finally we add a legend to the plot (Fig. 10-8):

```
> par(mfrow=c(1,1))
> plot(size,number,type="n",xlab="Ant size",ylab="No. of attacks")
> points(size[ant=="famA"],number[ant=="famA"])
> points(size[ant=="famB"],number[ant=="famB"],pch=16)
> x<-seq(0,10,0.01)
> lines(x,predict(m1,list(size=x,ant=factor(rep("famA",length(x)),
```

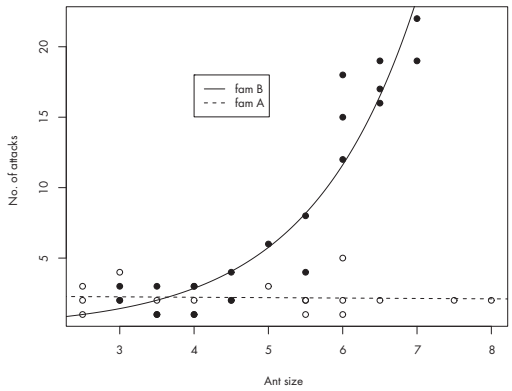


Fig. 10-8 Models of the relationship between the number of attacks and the body size of ants belonging to two subfamilies ("famB" and "famA") compared to the data.

```
+ levels=levels(ant))),type="response"),lty=2)
> lines(x,predict(m1,list(size=x,ant=factor(rep("famB",length(x)),
+ levels=levels(ant))),type="response"))
> legend(4,18,c("famB","famA"),lty=1:2)
```

Using this example, we should recall how similar data were analysed in the past – by means of a normal linear model after a response transformation. We will do this in order to actually demonstrate the advantages of the Poisson analysis. Traditionally, the square root transformation is recommended for analyses of count data. For the Poisson distribution, this transformation can be theoretically deduced as a variance stabilising transformation. However, as we will find out shortly, it does not address other problems. It is the simplest approximative analysis.

We will focus on the relationship for "famB", because the model for "famA" is too simple (it has zero slope). Due to the curved relationship we specify a quadratic model (clearly, it is a drastic simplification, but it has been seen as a doable approximation by practitioners in the past):

$$\sqrt{number_i} = \alpha + \beta size_i + \gamma size_i^2 + \varepsilon_i \,, \qquad (10\text{-}5)$$

where $\varepsilon \sim N(0,\sigma^2)$, independent among measurements.

We select the "famB" level data only using the **`subset`** argument in the **`lm`** call.

```
> m3<-lm(sqrt(number)~size+I(size^2),subset=(ant=="famB"))
> anova(m3)
Analysis of Variance Table

Response: sqrt(number)
          Df  Sum Sq Mean Sq  F value    Pr(>F)
size       1 28.3476 28.3476 161.0631 4.253e-10 ***
I(size^2)  1  1.1930  1.1930   6.7783   0.01855 *
```

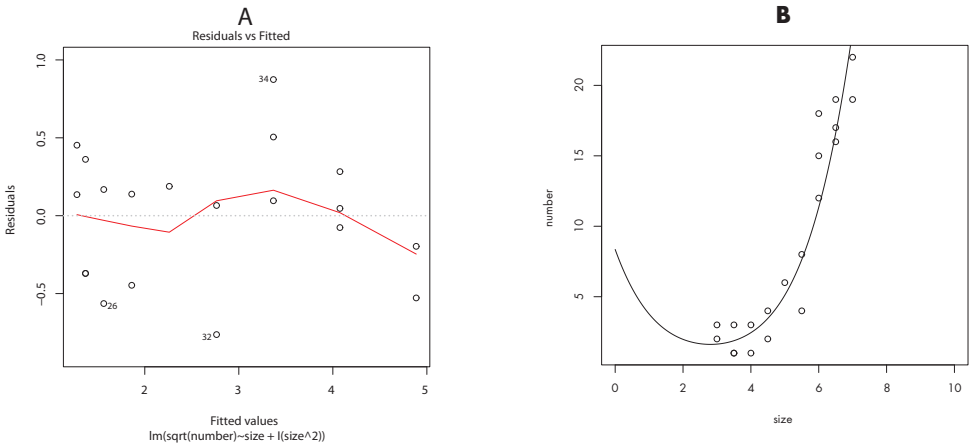


Fig. 10-9 A. Relationship between residuals and fitted values of a model with square root transformed counts. **B**. Model of the relationship between number of attacks and body size with square root transformed response variable for the subfamily "famB".

```
Residuals 17  2.9921  0.1760
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> summary(m3)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.89010    1.71419   1.686   0.1101
size        -1.15372    0.72495  -1.591   0.1299
I(size^2)    0.20557    0.07228   2.844   0.0112 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4373 on 17 degrees of freedom
Multiple R-squared:  0.9025,    Adjusted R-squared:  0.8911
F-statistic: 78.72 on 2 and 17 DF,  p-value: 2.541e-09
```

All three terms of the model `m3` are significant – as expected. Curvature is clearly present in the data (as we know from the exponential model fitted previously in `m1`), so the quadratic term is necessary. And how do the diagnostic plots look for this model? We should remind you that by fitting the `m3` model we pretended that we expected that the errors in this model come from a normal distribution. We will only graphically explore the relationship between the residuals and fitted values. You can perform other checks.

```
> plot(m2,which=1)
```

The residuals show an unexplained pattern. a hint of a similar phenomenon was noticed in the model `m1` as well (Fig. 10-9A). How does the transformed model fit the *original* (untransformed) data? We will draw the model to the plot with extended abscissa, to see its behaviour outside the measured range. To plot the model, we call the **`curve`** function with

the **add=T** argument. The first argument of this function is the linear predictor from model m3, but the dummy variable x replaces the actual explanatory variable name (*size*). We need to raise the outcome of the linear predictor to the power of 2 (to get the estimates on the scale of counts, not on the scale of square roots where the m3 model fitting took place).

```
> plot(size,number,type="n",xlim=c(0,10))
> points(size[ant=="famB"],number[ant=="famB"])
> curve((2.89-1.15*x+0.21*x^2)^2,add=T)
```

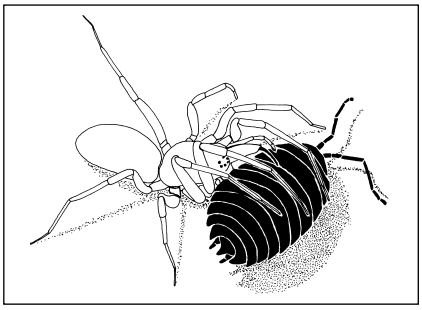
At the interval well covered by the measurements, the fit of transformed model m3 might seem slightly better than that of the model m1, especially in the central part where the Poisson model slightly overestimates (Fig. 10-9B). Nevertheless, outside of this interval (particularly close to zero) the behaviour of the model m3 is really strange. In fact, for small ants, the m3 model suggests an increasing number of attacks for small ants, but that goes against both reality and theoretical considerations. The Poisson model m1 implies that the number of attacks increases monotonously with ant size.

In comparison with the Poisson model (m1), the transformed normal model (m3) is less parsimonious (because it has one extra parameter) and, most of all, produces a grossly incorrect approximation to the number of attacks even slightly beyond the range of sizes covered in the study.

CONCLUSION

Number of attacks was significantly different between the studied ant subfamilies (GLM-p, $\chi^2_1 = 20.8$, $P < 0.0001$). While for "famA" the number of attacks appears to be independent of ant body size and is equal to about 2.5 ($number = e^{0.89}$), for "famB" the number of attacks increased with the ant body size according to the formula: $number = e^{-1.77+0.68\,size}$.

10.6 Three-way ANODEV (Contingency table)



We will demonstrate here how the power of the GLM approach can be useful for the analysis of more complex Poisson data than that we have seen so far. We might, for instance, be interested in a detailed analysis of how three factors simultaneously influence observed counts. This is a case of a multinomial model.

EXAMPLE

Some predators use conditional strategies to catch prey. The use of a strategy often depends on prey characteristics. Woodlice-eating spiders can use three capture strategies (*STRATEGY*: stratA, stratB, stratC) to catch woodlice. In the field, it was observed which of three strategies spiders used to capture woodlice. Captured prey was then classified ac-

Table 10-1 Frequencies of three capture strategies used when hunting four types of prey.

PREY:	slow		fast	
SIZE:	small	large	small	large
stratA	19	10	21	12
stratB	4	10	0	8
stratC	0	1	1	2

cording to its size (*SIZE*: large, small) and speed of movement (*PREY*: fast, slow). Altogether there were 88 observations. The observed frequencies (*freq*) are shown in Table 10-1. We are interested in the following questions: (1) Is the use of strategy influenced by prey size and its movement? (2) If it is influenced, which prey type tend to be captured by each particular strategy?

EDA

Although there are three categorical variables in the data frame, we are interested only in one specific question: how the capture strategy (*STRATEGY*) was affected by traits of prey (*PREY* and *SIZE*). *freq* is the response variable (i.e. frequency of observations for particular combinations of levels of all three explanatory variables). Of all the interactions, we are interested only in the interaction among *PREY*, *SIZE*, and *STRATEGY*. Thus, for a quick orientation to the data, we will use interaction plots, where the first factor is going to be *STRATEGY* and the second one is going to be one of the two other explanatory variables.

```
> dat<-read.delim("predator.txt"); attach(dat); dat
   prey  size strategy freq
1  slow small   stratA   19
2  slow small   stratB    4
3  slow small   stratC    0
...
12 fast large   stratC    2
> interaction.plot(strategy,prey,freq)
> interaction.plot(strategy,size,freq)
```

From the first plot (Fig. 10-10A), it is apparent that to capture both "slow" and "fast" prey, the spiders most frequently used "stratA", less frequently "stratB" and they and almost never used "stratC". As the lines cross only very slightly, interaction might not be significant. The second plot (Fig. 10-10B) shows that when capturing "small" prey, "stratA" was used almost exclusively. This does not apply to "large" prey, which were captured most frequently with both "stratA" and "stratB". Interaction *STRATEGY*:*SIZE* will thus be most likely significant.

MODEL

We will test the null hypothesis, i.e. that the choice of strategy has not been influenced by the two study factors (or their interactions). That is why we will be only interested in the interactions between the *STRATEGY* and the explanatory variables, *PREY* and *SIZE* (i.e. two-way interactions). We are not interested in the main effects of *PREY*, *SIZE* and *STRATEGY* (for example, in the case of *PREY*, we are not interested to know if there was more slow or fast

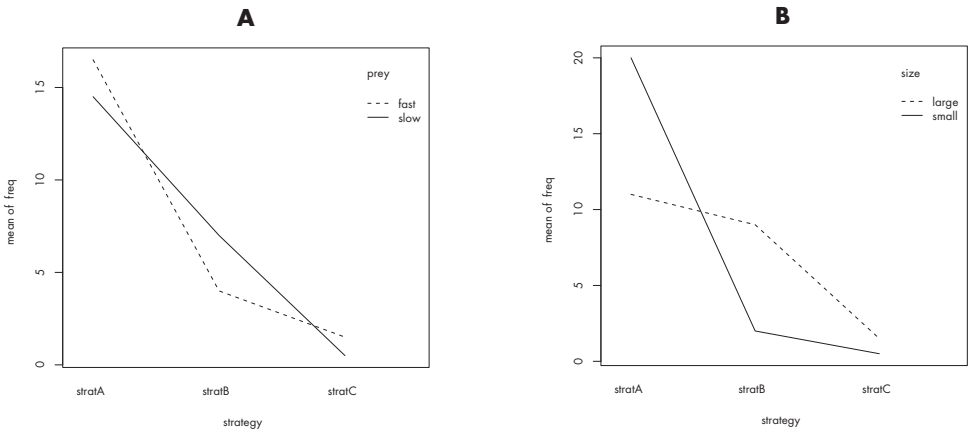


Fig. 10-10 A. Comparison of frequencies (*freq*) of capture strategies (*STRATEGY*) when hunting "fast" and "slow" prey. **B**. Comparison of frequencies (*freq*) of capture strategies (*STRATEGY*) when hunting "large" and "small" prey.

prey). Similarly, we are not interested in testing interaction *PREY*:*SIZE*, i.e. in determining which combinations of the prey were more frequent. Even though we are not interested in the main effects or in the two-way interaction between *PREY* and *SIZE*, we have to include them in the model as without them, the estimate and tests of the effects would not be correct. The following is the expected model for the i^{th} strategy, j^{th} size and k^{th} prey type:

$$\log(\mu_{ijk}) = \alpha + STRATEGY_i + SIZE_j + PREY_k + STRATEGY : PREY_{ik} + STRATEGY : SIZE_{ij}, \quad (10\text{-}6)$$

where $freq_{ijk} \sim Poi(\mu_{ijk})$, independent among measurements.

ANALYSIS

You have likely realised that this is an example of a log-linear model (this method was developed, and frequently and long used, before the GLM). To analyse the data we will not use the function **`loglin`**, but **`glm`** with a Poisson error structure. Both factors and the stimulus variable *STRATEGY* will appear in the linear predictor (i.e. on the right side of the model formula). Let's see what happens if we fit a model, which is much larger than the one we expected. It will include all main effects, three two-way and one three-way interactions:

$$\log(\mu_{ijk}) = \alpha + STRATEGY_i + SIZE_j + PREY_k + STRATEGY : PREY_{ik} + STRATEGY : SIZE_{ij} + SIZE : PREY_{jk} + STRATEGY : SIZE : PREY_{ijk}, \quad (10\text{-}7)$$

where $freq_{ijk} \sim Poi(\mu_{ijk})$, independent among measurements.

```
> m1<-glm(freq~strategy*size*prey,family=poisson)
> summary(m1)
```

```
Call:
glm(formula = freq ~ strategy * size * prey, family = poisson)

Deviance Residuals:
 [1]  0  0  0  0  0  0  0  0  0  0  0  0

Coefficients:
                                    Estimate Std. Error   z value Pr(>|z|)
(Intercept)                        2.485e+00  2.887e-01     8.608   <2e-16
strategystratB                    -4.055e-01  4.564e-01    -0.888   0.3744
strategystratC                    -1.792e+00  7.638e-01    -2.346   0.0190
sizesmall                          5.596e-01  3.619e-01     1.546   0.1220
preyslow                          -1.823e-01  4.282e-01    -0.426   0.6702
strategystratB:sizesmall          -2.594e+01  6.965e+04 -0.000372   0.9997
strategystratC:sizesmall          -1.253e+00  1.277e+00    -0.981   0.3266
strategystratB:preyslow            4.055e-01  6.390e-01     0.635   0.5257
strategystratC:preyslow           -5.108e-01  1.297e+00    -0.394   0.6938
sizesmall:preyslow                 8.224e-02  5.325e-01     0.154   0.8773
strategystratB:sizesmall:preyslow  2.438e+01  6.965e+04  0.000350   0.9997
strategystratC:sizesmall:preyslow -2.269e+01  6.965e+04 -0.000326   0.9997
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 8.7966e+01  on 11  degrees of freedom
Residual deviance: 3.0330e-10  on  0  degrees of freedom
AIC: 60.15

Number of Fisher Scoring iterations: 21
```

This is a **saturated model**. It has no residual degrees of freedom. This is because the model estimates as many parameters (12) as there are degrees of freedom available from the data (as many as there are cells in the contingency table). Residual deviance is therefore null (though, due to small numerical errors, in the summary output we can read a very small number, 3.033×10^{-10} instead of zero). Consequently, the deviance residuals are uniformly equal to zero. The saturated model produced exact estimates of observed frequencies, but we cannot evaluate the quality of the model (at least not by the standard graphical diagnostic tools). Yet, such model might be useful as a starting point for fitting smaller and simpler models (given these data it is impossible to fit a different non-saturated Poisson model).

Let's examine the effect of the individual model terms using ANODEV table.

```
> anova(m1,test="Chi")
...
                Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                                11     87.966
strategy         2   64.205         9     23.761 1.143e-14
size             1    0.045         8     23.715     0.831
prey             1    0.000         7     23.715     1.000
strategy:size    2   15.939         5      7.776 3.458e-04
strategy:prey    2    2.962         3      4.814     0.227
```

```
size:prey           1    0.507         2      4.307     0.476
strategy:size:prey  2    4.307         0  3.033e-10     0.116
```

Of all the terms, only two are significant, the main effect of the *STRATEGY* and interaction *STRATEGY*:*SIZE*. As we have already stated before, we are interested only in the three- and two-way interactions, i.e. those that include *STRATEGY*. The other terms, i.e. all three main effects and the *PREY*:*SIZE* interactions, have to be kept in the model even though we will not attempt to interpret them, because they are not in the direction of our interest. Firstly, we will remove the non-significant three-way interaction.

```
> m2<-update(m1,~.-strategy:size:prey)
> anova(m2,test="Chi")
...
             Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                           11     87.966
strategy      2   64.205          9     23.761 1.143e-14
size          1    0.045          8     23.715     0.831
prey          1    0.000          7     23.715     1.000
strategy:size 2   15.939          5      7.776 3.458e-04
strategy:prey 2    2.962          3      4.814     0.227
size:prey     1    0.507          2      4.307     0.476
```

In the adjusted model, the *STRATEGY*:*PREY* interaction is not significant. This says that movement of the prey had no effect on the use of the strategy. That is actually the last component that we can remove from the model since the *STRATEGY*:*SIZE* interaction is significant.

```
> m3<-update(m2,~.-strategy:prey)
> anova(m3,test="Chi")
...
             Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                           11     87.966
strategy      2   64.205          9     23.761 1.143e-14
size          1    0.045          8     23.715     0.831
prey          1    0.000          7     23.715     1.000
strategy:size 2   15.939          5      7.776 3.458e-04
size:prey     1    0.045          4      7.731     0.831
```

Effect of the *STRATEGY*:*SIZE* interaction is still significant – the use of strategies differed based on the size of prey. There is nothing more in model `m3` that we could further simplify. We have thus arrived at a model that we will consider to be final and that can be formally written as follows:

$$\log(\mu_{ijk}) = \alpha + STRATEGY_i + SIZE_j + PREY_k + SIZE:PREY_{jk} + STRATEGY:SIZE_{ij}, \quad (10\text{-}8)$$

where $freq_{ijk} \sim Poi(\mu_{ijk})$, independent among measurements.

Let's have a closer look at the table of coefficients of this model.

```
> summary(m3)
```

```
Call:
glm(formula = freq ~ strategy + size + prey + strategy:size +
    size:prey, family = poisson)

Deviance Residuals:
      1       2       3       4       5       6       7
-0.3233  1.2076 -1.0111 -0.2297  0.3990 -0.4079  0.3227
      8       9      10      11      12
-1.9777  0.6395  0.2194 -0.4077  0.3585

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)                2.42088    0.26010   9.307  < 2e-16 ***
strategystratB            -0.20067    0.31782  -0.631 0.527782
strategystratC            -1.99243    0.61546  -3.237 0.001207 **
sizesmall                  0.55237    0.34042   1.623 0.104669
preyslow                  -0.04652    0.30508  -0.152 0.878805
strategystratB:sizesmall -2.10191    0.61318  -3.428 0.000608 ***
strategystratC:sizesmall -1.69645    1.18481  -1.432 0.152193
sizesmall:preyslow         0.09097    0.42662   0.213 0.831142
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 87.9661  on 11  degrees of freedom
Residual deviance:  7.7306  on  4  degrees of freedom
AIC: 59.88

Number of Fisher Scoring iterations: 5
```

Of course, the model is not saturated any more – the table includes only eight coefficients. That is also why the residuals are not zero any more. Interpretation of the coefficients may appear cumbersome at first, but it is actually not difficult. We just need to keep in mind that we have used treatment contrasts and that the coefficients are on a logarithmic scale. Frequencies can be obtained from the table relatively simply. For example, we get the mean value for "stratB-large" by adding the first two coefficients and exponentiating the result: exp(2.421–0.201) = *9.207*. Only two partial tests, apart from the test for Intercept, are significant in the table of coefficients of model `m3`. The first one shows a significant difference between strategy "stratC" and strategy "stratA" for large prey, and the second one shows a significant difference between strategy "stratB" and strategy "stratA" for a small prey. For a simple interpretation of the final model, we will enter the predicted values for the combination of the *STRATEGY* and *SIZE* variables in the `attacks` object.

```
> attacks<-tapply(predict(m3,type="response"),list(size,strategy),mean)
> attacks
      stratA stratB stratC
large     11      9    1.5
small     20      2    0.5
```

Now we can nicely see how different strategies are used for different prey sizes. This table corresponds to the plot on Fig. 10-10B.

We expect that the data come from the Poisson distribution and that they are independent. We could check the first assumption informally using the pre-set diagnostic plots. We will consider the assumption that the various data are independent to be correct based on the way in which the study was conducted (properly randomised).

We will calculate the 95% confidence intervals for the mean count values. That is why we will refit the model in the textbook parametrization. Prior to that, we combine both factors into a single factor with six levels. The new factor levels will correspond to the six combinations of levels *STRATEGY* and *SIZE*. Next, we modify the model formula by removing the intercept. In this way, we obtain estimates of the mean values together with their standard errors. Both of them are needed for confidence interval construction. We will calculate the intervals on the logarithmic scale using the **`confint`** function and transform the result to the original scale of counts through exponentiation.

```
> both<-paste(strategy,size)
> m4<-glm(freq~factor(both)-1,poisson)
> exp(confint(m4))

Waiting for profiling to be done...
                                  2.5 %    97.5 %
factor(both)stratA large  7.01961803 16.257527
factor(both)stratA small 14.42415842 26.853438
factor(both)stratB large  5.45596390 13.820389
factor(both)stratB small  0.62054466  4.642473
factor(both)stratC large  0.37275278  3.885165
factor(both)stratC small  0.02850154  2.201752
```

Now we can draw the estimated mean values, which we have saved in the `attacks` object, together with the 95% confidence intervals in the plot (Fig. 10-11) using a few functions with which you should be quite familiar already:

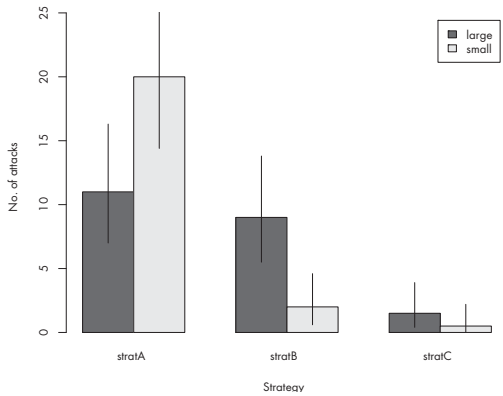


Fig. 10-11 Comparison of frequencies of three capture strategies ("stratA", "stratB", "stratC") when hunting "large" and "small" prey. Vertical lines are 95% confidence intervals.

```
> barplot(attacks,beside=T,ylab="No. of attacks",xlab="Strategy",
+ legend.text=c("large","small"),ylim=c(0,25))
> lines(c(1.5,1.5),c(7,16.3))
> lines(c(2.5,2.5),c(14.4,26.9))
> lines(c(4.5,4.5),c(5.5,13.8))
> lines(c(5.5,5.5),c(0.6,4.6))
> lines(c(7.5,7.5),c(0.4,3.9))
> lines(c(8.5,8.5),c(0.03,2.2))
```

CONCLUSION

The data do not support an effect of the prey movement on the type of the hunting strategy used (GLM-p, $\chi^2_1 < 0.1$, P = 0.83). On the other hand, they support a significant effect of the size of prey (GLM-p, $\chi^2_2 = 15.9$, P = 0.0004). While the spiders hunted small prey, they used almost exclusively the "stratA" strategy. When hunting large prey, they used both "stratA" and "stratB" strategies.

11 NEGATIVE-BINOMIAL DISTRIBUTION

A negative-binomial model is a parametric alternative to the Poisson model with overdispersion. It can be obtained as a compound distribution: by randomly varying the mean parameter (according to a gamma distribution) in the Poisson model. A negative-binomial model has one extra parameter (related to the description of overdispersion) in comparison with the Poisson model, which increases its flexibility. Even though we can also use other models for modelling overdispersion, the advantage of a negative-binomial model is its simplicity. It is implemented in R (and in other environments) and it is frequently used in practice.

11.1 Description of the negative-binomial model

The negative-binomial model is not implemented in GLM. Therefore, it cannot be fitted using **`glm`**. It is, however, closely related to the GLM and thus can be fitted after a slight generalisation of the GLM approach. One function that has implemented the negative-binomial model is **`glm.nb`**. It allows us to specify the model formula in a way that is analogous to what we are used to in **`glm`**. The **`glm.nb`** function is available from the MASS library (Venables & Ripley 2002). When invoking this function, we (implicitly) assume that observations y come from a negative-binomial distribution. Its variance is functionally related to the expected value $\mathrm{E}(y) = \mu$ as follows:

$$\mathrm{Var}(y) = \mu + \frac{\mu^2}{\theta}\,, \tag{11-1}$$

where θ is the parameter controlling how much the model is different from the Poisson distribution. The value of θ has to be in the $(0, \infty)$ interval. We can see that a "pure" Poisson distribution (for which variance is equal to mean) is a limiting case. Negative-binomial distributions for large θ (in comparison to μ^2) behave essentially as a Poisson distribution. For small θ, the negative-binomial shows large overdispersion and behaves very differently to the Poisson distribution. The value of θ can be approximately estimated from the expected value and variance (by the method of moments) using the formula (11-1):

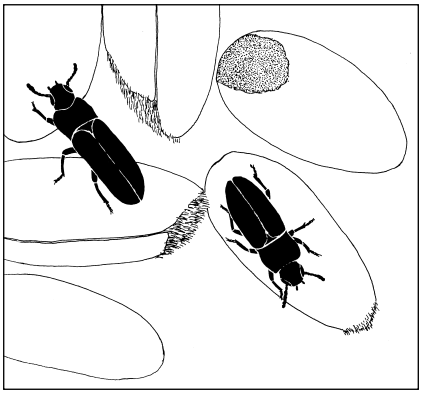
$$\hat{\theta} = \frac{\bar{y}^2}{s^2 - \bar{y}} \tag{11-2}$$

where $\bar{y}$ is a sample mean and s^2 is the sample variance. θ can be estimated more precisely by the method of maximum likelihood. In **`glm.nb`**, the simple-to-compute moment estimate

is used as a preliminary estimate, and also as an initial value for iterative maximisation of the likelihood function.

Similarly to the Poisson GLM, the linear predictor models the link-transformed expected value (μ). The **`log`** link is used very often in practice (due to a close relationship to the Poisson GLM). This is a default option in **`glm.nb`**. Alternatively, we can use the **`sqrt`** or **`identity`** link functions.

11.2 One-way ANODEV



Grain beetles are serious pests in grain stores. They occur in the grain and also in crevices of corridors of the store. It is essential to know exactly where they occur before control methods can be applied. Density of grain beetles was surveyed in a grain store by means of sticky traps. Traps were installed in two places (*PLACE*: floor, grain): 25 traps on the floor in corridors and 25 traps in the grain. After a few days, the number of beetles was recorded in the traps – in the form of abundance per trap area (*density*). We need to know: (1) Is density of beetles similar in both places? (2) If it is not similar, how much different it is?

EDA

We can compare beetle density between both places using a box plot.

```
> dat<-read.delim("beetle.txt"); attach(dat); names(dat)
[1] "place"   "density"
> plot(place,density)
```

The box for the "grain" level cannot be clearly seen in the plot (Fig. 11-1). This is because many values are close to zero. We can see the distribution of measurements better from the frequency table:

```
> table(density)
density
  0   1   2   3   5   6   7   8   9  10  11  14  19  21  24  33  76  88  90
  8   4   4   2   3   1   1   1   3   3   2   1   2   3   1   1   1   1   1
 94 132 174 201 271 505 800
  1   1   1   1   1   1   1
```

Densities between 0 and 800 were observed and the most common value was zero (it occurred eight times). These are densities counted for both levels. We can easily compute the frequency table for densities at "grain" places only by calling **`table(density[place=="grain"])`**. Inspection of the tables suggests that distribution of the beetles in the store do not have a Poisson character. Furthermore, the box plot also reveals that variance was much

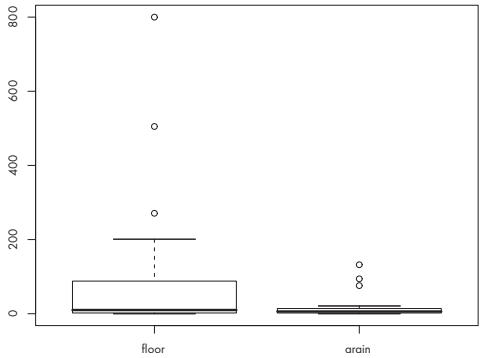


Fig. 11-1 Box plot of beetle density in two places: "floor" and "grain".

greater in the traps placed on the floor than in the grain. We are not able to determine the median values from the plot. Thus we will calculate the arithmetic means:

```
> tapply(density,place,mean)
floor grain
91.48 17.96
```

On average, many more beetles were caught in the traps on the floor of corridors than in the grain. The difference appears rather large; however, we do not know how important it really is without the appropriate analysis.

MODEL

We assume that density of the beetles will depend on the location of the traps. The indication of a spatial aggregation observed in the store suggests that their distribution will be negative-binomial. We will use a logarithmic link for the convenience of interpretation. The model for the j^{th} level of the *PLACE* will be:

$$\log(\mu_j) = \alpha + PLACE_j, \tag{11-3}$$

where $density_j \sim NB(\mu_j, \theta)$, independent among traps.

$NB(\mu,\theta)$ stands for the negative-binomial distribution with parameters μ and θ. Their interpretation has been explained above. Please note an important detail in the model; while the expected value changes according to the place (thus μ is indexed with j), the coefficient θ is *identical* (in another words, overdispersion is identical) for both places. This is a kind of restriction (generally θ may not be identical), which corresponds to the standard use of **`glm.nb`**. We will use the (default) treatment parametrization.

ANALYSIS

Because the response variable includes counts of individuals, we first try to fit a Poisson **`glm`** with overdispersion as we were used to previously. The model will have identical systematic part as (11-3):

$$\log(\mu_j) = \alpha + PLACE_j, \qquad (11\text{-}4)$$

where $density_j \sim Poi(\mu_j)$, independent among traps.

Since we expect overdispersion we use **quasipoisson**.

```
> m1<-glm(density~place,family=quasipoisson)
> anova(m1,test="F")
...

      Df Deviance Resid. Df Resid. Dev      F  Pr(>F)
NULL                     49     8026.5
place  1   1350.1        48     6676.4 6.0434 0.01762 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> summary(m1)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.5161     0.3125   14.45   <2e-16 ***
placegrain   -1.6280     0.7715   -2.11   0.0401 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasipoisson family taken to be 223.3983)

    Null deviance: 8026.5  on 49  degrees of freedom
Residual deviance: 6676.4  on 48  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 6
```

The result of the test is formally significant at the 5% level. This means that beetle density is not the same at the two places. However, we have to be careful – the p-value from the ANODEV table is very close the "magic" level of 0.05. It is quite possible that this could change (to one side or the other) upon a small change to the model (for example, as a result of a more correct consideration of overdispersion). We should test it thoroughly.

The value of the dispersion parameter in model `m1` demonstrates that overdispersion is very high (about 223 times). The Poisson distribution does not appear to be suitable for describing these data. We can see this also from the diagnostic plot.

```
> plot(m1,which=1)
```

Residual deviance (4-15) does not behave the way it should. Its variance is completely heterogeneous. This points to the model's systematic overestimation of reality (Fig. 11-2A). When we look at the ratio between empirical variance and empirical mean values for the two places by calling the **tapply** function,

```
> tapply(density,place,var)/tapply(density,place,mean)
    floor     grain
386.58096  60.20546
```

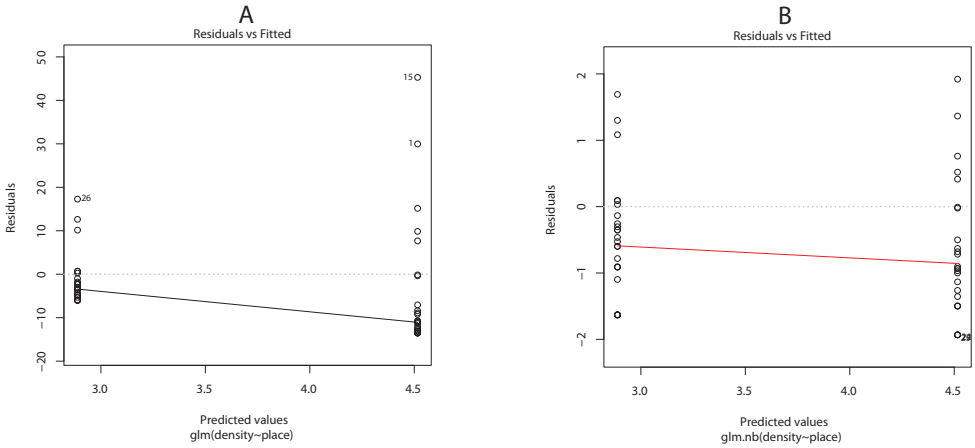


Fig. 11-2 Relationship between deviance residuals and fitted values of the model `m1` (**A**) and the model `m2` (**B**).

we find that it is very high! For any Poisson distribution, it is (theoretically – for the real mean values and dispersions, not for their estimates) equal to 1! A greater variance than mean is typical for aggregated data. That is why the negative-binomial model should be more suitable. We can get a rough moment estimate of parameter θ for each level of the *PLACE* using (11-2):

```
> tapply(density,place,function(x) mean(x)^2/(var(x)-mean(x)))
    floor     grain
0.2372524 0.3033504
```

Values of both estimates are rather close to 0. They are also small compared to the estimate of μ^2. Therefore, the ratio μ^2/θ is very large. Overdispersion is present in the data, supporting the biological intuition that the beetles aggregate. Before using the **`glm.nb`** function, we have to upload the MASS library. We will supply the model formula as an argument to the **`glm.nb`**. The ANODEV table is displayed by calling **`anova`**. The type of the required test statistic does not need to be specified here (in contrast to the **`glm`** function).

```
> library(MASS)
> m2<-glm.nb(density~place)
> anova(m2)
Analysis of Deviance Table

Model: Negative Binomial(0.3318), link: log

Response: density

Terms added sequentially (first to last)
```

```
      Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                     49     70.174
place  1    9.877        48     60.297     0.002
Warning message:
In anova.negbin(m2) : tests made without re-estimating 'theta'
```

The ANODEV table is similar to that for **glm** objects, but it shows also the estimate of θ, which is not present in **glm**. The estimate is $\hat{\theta} = 0.3318$. It was estimated from the data using the method of maximum likelihood. The effect of the factor *PLACE* is now significant at the 5 % level. What are the estimates of the coefficients?

```
> summary(m2)

Call:
glm.nb(formula = density ~ place, init.theta = 0.331844006124825,
    link = log)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.93179  -1.22978  -0.69830  -0.05072   1.92169

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   4.5161     0.3478  12.984  < 2e-16 ***
placegrain   -1.6280     0.4937  -3.297 0.000976 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(0.3318) family taken to be 1)

    Null deviance: 70.174  on 49  degrees of freedom
Residual deviance: 60.297  on 48  degrees of freedom
AIC: 430.95

Number of Fisher Scoring iterations: 1

              Theta:  0.3318
          Std. Err.:  0.0610

 2 x log-likelihood:  -424.9480
```

There are two parameter estimates listed in the table of coefficients: an estimate of the intercept (corresponding to the logarithmically transformed mean value of the counts for "floor") and an estimate of the difference of the logarithms of the mean count values between "grain" and "floor". To decipher the meaning of the coefficients, we just need to realise that we have used treatment contrasts. In the bottom part of the summary output, you can find the value of the theta (θ) estimate, its asymptotic standard error, and the logarithm of the likelihood function (assessed at the maximum likelihood estimate). Note that the p-value has reduced significantly in comparison with the quasi-Poisson model `m1` (the significance is not close to the decisive 5% level any more). This case is an example of a phenomenon that can be expressed by the following saying: "nothing is more practical than a good theory." A model, for which we correctly consider the natural characteristics of the study process, is often much more efficient for the

extraction of a study phenomenon (estimate as well as testing) than a model that is seemingly simple and general. It is always worthwhile to think about your model.

Let us explore the first of the standard diagnostic plots:

```
> plot(m2,which=1)
```

Variance of the deviance residuals is now more or less homogenous (Fig. 11-2B). The residual behaviour is definitely better, compared to what we saw in the previous model. Nevertheless, it is not entirely ideal either. But let us go back to the model assumptions (11-3). One of the possible complications can be the fact that the standard use of **`glm.nb`** implicitly estimates a single common value of θ (for "floor" as well as for "grain"), but the rough moment estimates that we computed above suggest that θ might be different for the levels. We need to check this. We cannot do it directly by fitting a single model. We need to stratify the negative-binomial model by the levels of the *PLACE* factor. In other words, we will fit separate negative-binomial models for each level of the *PLACE* factor. Next, we will compare the stratified models with `m2` using the loglikelihood values. We will split the *density* variable, calling the **`split`** command, by the levels of *PLACE* into two vectors in an object `a`. The first argument of this function (**`x`**) is the name of the response variable and the second one (**`f`**) is the factor name. It is advisable to make the new object visible to R using the **`attach`** command.

```
> a<-split(density,place)
> attach(a); a
$floor
 [1] 505 271   1   9   5  10   1  19   3 201  24   1 174  90 800   2
[17]  88  33  11  10   8   0  21   0   0

$grain
 [1] 132  76   2   6  94  10  14   0  21   0   3   2   5   0   9   0
[17]   7   1  21  19  11   2   0   9   5
```

Now we can fit the negative-binomial model separately for each level by using the names of levels as response variables and by including only the intercept (**`1`**) in the linear predictor.

```
> m3<-glm.nb(floor~1)
> summary(m3)
...
    Null deviance: 31.307  on 24  degrees of freedom
Residual deviance: 31.307  on 24  degrees of freedom
AIC: 245.47

Number of Fisher Scoring iterations: 1

              Theta:  0.2915
          Std. Err.:  0.0719

 2 x log-likelihood:  -241.4670
> m4<-glm.nb(grain~1)
> summary(m4)
...
    Null deviance: 29.197  on 24  degrees of freedom
Residual deviance: 29.197  on 24  degrees of freedom
```

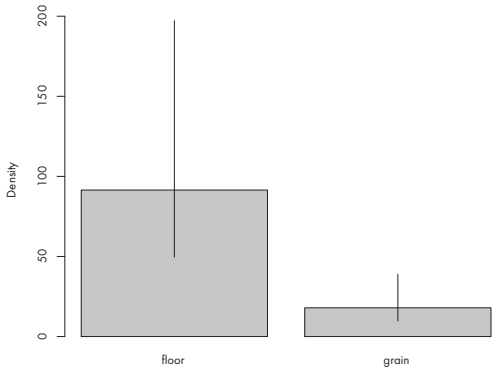


Fig. 11-3 Comparison of expected values of beetle density on the "floor" and in the "grain". 95% confidence intervals are shown as vertical lines.

```
AIC: 186.78

Number of Fisher Scoring iterations: 1

              Theta:  0.399
          Std. Err.:  0.111

 2 x log-likelihood:  -182.780
```

We compare the two stratified models (`m3` and `m4`) with `m2` using the values of 2 x log-likelihood, which can be read from the output. For model `m2` it is *–424.948*, whereas a sum of values for models `m3` and `m4` is *–424.247*. The restricted model `m2` is a little bit worse (obviously, the restricted model can never be better). In this case it is only slightly worse (difference is 0.701) than the stratified model, suggesting that the fit of restricted and unrestricted models is essentially the same. If we wish to test the significance of the difference we can use χ^2 distribution function (**`pchisq`**) with one degree of freedom (as the difference between models):

```
> 1-pchisq(0.701, df=1)
[1] 0.4024479
```

No the difference is not significant. The restricted model `m2` is one parameter smaller and, therefore, we will prefer it.

Finally, we will estimate the confidence intervals, with which we will, subsequently and together with the mean values estimated from the model, draw in a bar plot. The plot drawing procedure is very similar to what we saw before (see, for example, Chapter 10.2). We will not, therefore, comment on it any more here. Instead, we will just show the commands that can be used. The final plot is displayed in Fig. 11-3.

```
> m5<-glm.nb(density~place-1)
> exp(confint(m5))
```

```
Waiting for profiling to be done...
              2.5 %    97.5 %
placefloor 49.57777 197.24605
placegrain  9.67290  38.87768
> barplot(tapply(predict(m2,type="response"),place,mean),ylab="Density",
+ ylim=c(0,200))
> lines(c(0.7,0.7),c(49.6,197.2))
> lines(c(1.9,1.9),c(9.7,38.9))
```

CONCLUSION

Beetle density differed significantly between places within the grain store (GLM-nb, $\chi^2_1 = 9.9$, P = 0.002). There was about five times more beetles on the floor of corridors than in the grain (average = 92 *versus* 18). Beetle occurrence at both places was aggregated (common $\theta = 0.33$).

12 BINOMIAL DISTRIBUTION

In this chapter, we will show how to analyse data from binomial distributions, including the Bernoulli distribution, which is a special case.

In biology, binomial data occur in many different situations, e.g. when assessing responses of organisms to different doses of a toxic substance. Analyses of similar types are often called **dose-response** studies. As the name suggests, they look for the functional form of the relationship between dose and the response of an organism. Even though binomial data probably arise most naturally in toxicological studies, they always occur when we evaluate a phenomenon only qualitatively, i.e. whether it has occurred on a given individual or not. Apart from the often monitored mortality, it can also be birth, germination or attack (for example, crops attacked by pests), consumption (of a prey by predators), commencement of a reaction (chemical or in the form of a certain behaviour).

Similarly as per frequencies, one should think of the notion of replication for binomial data somewhat more carefully. This is because the occurrence of all events may be added together within the frame of a single experiment and expressed as a single relative frequency. For example, if we study the influence of a certain toxic substance on the survival of aphids and we put all 200 of them in one dish, by counting all the aphids that die (e.g. 100), we will get one value of empirical frequency (e.g., $p = 0.5$). Should we repeat the experiment on another day (with a different 300 specimens of aphids) and should 200 of them die this time, we get yet another value of a relative frequency ($p = 0.667$). If both experiments were conducted under the same conditions, we could combine the results of both experiments. We get the average relative frequency by adding the number of all dead (300) and dividing by all used (500) specimen. So we will calculate a single relative frequency ($p = 0.6$). It would be an ERROR to express the average mortality as the average of two relative frequencies, i.e. $p = (0.5 + 0.667) / 2 = 0.5835$.

In R, one can analyse binomial data using several methods. For a simple comparison there is the **Exact binomial test** (`binom.test`) or the **Proportion test** (`prop.test`), which you must have heard about in your elementary statistics course. If we want to compare probability distributions for a variable which has more than two levels, we have to use different methods. Either the **Contingency table** (`xtabs`) or, particularly if we have both categorical and continuous variables, a model of the **Logistic regression** (from the GLM class). In the past, methods based on the pre-transformation of the original data and linear analysis were often used. The angular transformation ($\arcsin\sqrt{p}$). was then typically recommended.

12.1 Description of the binomial model

When using **`glm`** with the argument **`family=binomial`**, we specify (besides other things) that y comes from the binomial distribution with the expected value, $\mathrm{E}(y) = n\pi$, and the variance, $\mathrm{Var}(y) = n\pi(1-\pi)$. This clearly shows that this method inherently corrects for non-constant variance (it includes automatic weighting with respect to various values of n and π as these values affect the variance). Use of this method guarantees that predicted values will always be from the legal range [0,1] and, at the same time, that we work with the binomial model correctly, using the method of maximum likelihood.

When choosing the binomial distribution in the function **`glm`**, the canonical link function is **`logit`** (7-1). Notice that (as in other GLM) we will work with transformed theoretical probabilities (π), and not with transformed relative frequencies (p) (as would be the case of models with, e.g., angularly transformed empirical logits: $\log(p/(1-p))$). The linear predictor is related to the link-transformed expected values of probabilities, π (i.e. to logits of π not of p). This has many theoretical and practical advantages. In particular, it is a correct analysis, which exactly corresponds to the assumed binomial model. Its use is more effective than *ad hoc* models with a transformed response variable.

The logit link has good statistical and simple mathematical properties. In special cases, one may prefer to use alternative links, such as **`probit`**, which is an inverse of the standard normal distribution function – i.e. the standard normal quantile function (see probit analysis used in toxicological and medical research). Then there is **`cloglog`** (complementary log-log link), which is defined as $\log(-\log(1-\pi))$ and which is useful in cases when the relationship between the linear predictor and probability of response is asymmetric (Fig. 12-1). In the examples below we will mostly use the **`logit`** link. The alternative links should not be used haphazardly without a cause. Their choice should be based on fundamental properties of the studied process (e.g. an asymmetric dose-response curve might suggest the complementary log-log link).

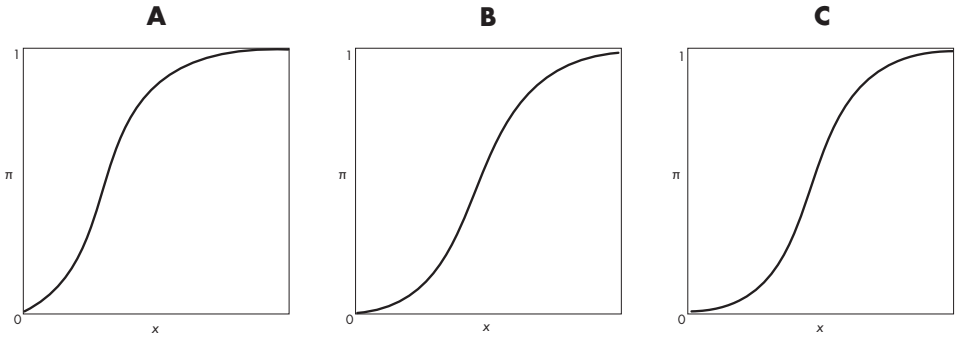


Fig. 12-1 Comparison of three link functions commonly used in binomial GLM models. **A**. cloglog. **B**. logit. **C**. probit.

Since the model uses the logit link transformation of the mean values (which are, in fact the probabilities π of the event of interest), it is clear that the resulting model parameters will be displayed in the table of coefficients on the logit scale $(-\infty, \infty)$. We can transform the value of the linear predictor (for example, Q) back to the original scale, i.e. relative frequency [0,1], using the inverse function to the logit, the standard logistic distribution function, which is sometimes called anti-logit:

$$\frac{1}{1+\exp(-Q)}. \tag{12-1}$$

If we take the exponent of Q, we get the **odds ratio**: $\pi/(1-\pi)$.

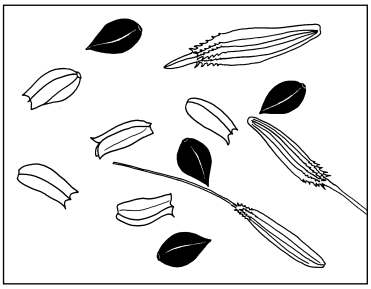
The binomial distribution has, apart from parameter π, also the n parameter. Nevertheless, the n parameter is (in the GLM analyses) not estimated. It is considered to be fixed and known. Information about n has to be supplied to **`glm`** as a kind of supplementary information. Instead of directly entering the information about n, we enter the number of "failures". The response variable is thus formally a matrix with two columns. The first column contains the number of "successes", let us call them y (the number of cases where the phenomenon of interest did happen). The second column contains values $n - y$ (i.e. the number of cases where the phenomenon did not occur). Note that this way of entering the response variable is very advantageous and general. For instance, it allows for correct processing of data from experiments, for which n was not constant across different trials.

Remember that the Bernoulli distribution is a special case of a binomial distribution (with $n = 1$). We will work with it to get an identical but differently specified logistic model. That is why we will be using the **`glm`** function with the argument **`binomial`** (with identical link functions: **`logit`**, **`probit`**, and **`cloglog`**). The only difference is the way we enter the response variable, which is formed only by a single vector y since $n = 1$ for all data rows. The response variable must be coded in a binary manner – either numerically, 0 and 1, or using symbols, for example, present and absent.

Both general binomial and Bernoulli coding of the response variable can be used for data with any combination of explanatory variables. The binomial one (using the matrix method of entering the response variable) will be convenient when, the individual data are already aggregated into the table of success counts (y) and, either sample sizes (n) or failure counts (n-y) for each combination of the explanatory variables are specified separately. If the data include a continuous explanatory variable, values of which are different for each individual observation (and hence they cannot be effectively aggregated), the Bernoulli type specification will be more convenient.

We will first explain the use of GLM with the binomial distribution on an example with categorical explanatory variables, which is an analogue of ANOVA.

12.2 Two-way ANODEV



Some weed seeds may germinate following water priming (e.g., by rain) more than others. The effect of water priming (*TREATMENT*: control, water) on the germination (*germ*) of weed seeds belonging to four genera (*GENUS*: genA, genB, genC, genD) was studied in a laboratory. The experiment lasted five days. On each day, 400 seeds of each genus were sown. 200 seeds were sown on the control soil and 200 seeds on the wet soil. Altogether 2000 seeds per genus were sown. Germination of seeds (yes or no) was recorded four days later. Based on the assumption of similar conditions during five days of sowing, data from the five days were pooled. We are interested in answering the following questions: (1) Does water priming promote germination? (2) If it does, was the priming effect similar for all four genera? (3) Which genus germinated with the highest and which with the lowest frequency?

EDA

In the data frame, there are two categorical explanatory variables, *TREATMENT* and *GENUS*. We will use the function **`interaction.plot`** to plot the data in order to see whether the two factors interact. From the number germinated (*germ*) and the number of used seeds (n) we calculate the empirical relative frequency (p) for each combination of the j^{th} level of *TREATMENT* and the k^{th} level of *GENUS* using a formula:

$$p_{jk} = \frac{germ_{jk}}{n_{jk}}, \tag{12-2}$$

We will use the relative frequencies as a response variable in the following plot to assess the *TREATMENT*:*GENUS* interaction informally.

```
> dat<-read.delim("weed.txt"); attach(dat); dat
  germ    n genus treatment
1  696 1000  genA     water
2  635 1000  genA   control
3  318 1000  genB     water
...
8  378 1000  genD   control
> p<-germ/n
> interaction.plot(genus,treatment,p)
```

Water priming has increased germination for all genera (Fig. 12-2). On average, it increased by about 5%. That is not much, but we shall see if the influence of water priming is significant or not. Even though the difference between the control and water priming treatment is not the same for all genera, we do not expect a significant effect from the interaction since the connecting lines of the means are quite parallel. On the other hand, there will probably be a significant difference among genera, or at least between "genA" with the highest, and "genB" with the lowest germination. Germination of "genC" and "genD" show little difference between water priming and control.

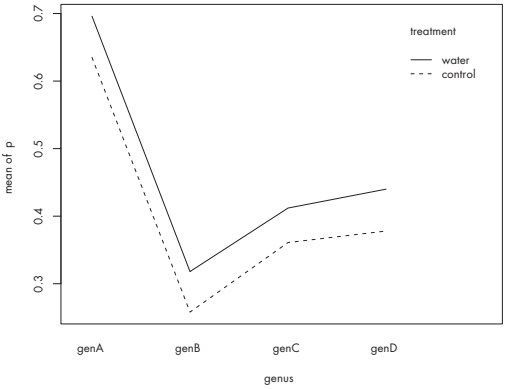


Fig. 12-2 Comparison of germination rate (p) of seeds belonging to four weed genera ("genA", "genB", "genC" and "genD") under two treatments ("control" and "water").

MODEL

Germination will depend on water priming and should be different for the genera of weeds, a fact which was also confirmed by an explorative analysis. We expect an additive effect of the *TREATMENT* and *GENUS* factors (Be careful to keep in mind that the interaction on the logit scale and the probability scale displayed in the picture above are different). We will explain the number of germinated seeds *germ* for the j^{th} level of *TREATMENT* and the k^{th} level of the *GENUS* using a treatment parametrization as follows:

$$\log\left(\frac{\pi_{jk}}{1-\pi_{jk}}\right) = \alpha + TREATMENT_j + GENUS_k, \tag{12-3}$$

where $germ_{jk} \sim Bin(\pi_{jk}, n_{jk})$, independent among pots.

ANALYSIS

Just to be sure, we will start with a model that will also include interaction of the factors, despite the fact that we do not expect it be significant. The model will look like this:

$$\log\left(\frac{\pi_{jk}}{1-\pi_{jk}}\right) = \alpha + TREATMENT_j + GENUS_k + TREATMENT : GENUS_{jk}, \tag{12-4}$$

where $germ_{jk} \sim Bin(\pi_{jk}, n_{jk})$, independent among pots.

We will use the (default) treatment parametrization for the model. We will now prepare the matrix of the values of the response variable by binding the *germ* counts and the *n–germ* counts. Next, we fit a two-way ANOVA-like model with interaction. We look at the results using the ANODEV table and χ^2-statistics.

```
> y<-cbind(germ,n-germ)
> m1<-glm(y~genus*treatment,family=binomial)
> anova(m1,test="Chi")
Analysis of Deviance Table

Model: binomial, link: logit

Response: y

Terms added sequentially (first to last)

               Df Deviance Resid. Df Resid. Dev  P(>|Chi|)
NULL                                7     669.34
genus           3   638.74         4      30.60 4.026e-138
treatment       1    30.23         3       0.37  3.840e-08
genus:treatment 3     0.37         0  1.212e-13       0.95
> m2<-update(m1,~.-genus:pesticide)
> summary(m2)
...
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.56138    0.05256  10.681   <2e-16 ***
genusgenB     -1.59933    0.06860 -23.313   <2e-16 ***
genusgenC     -1.15462    0.06614 -17.457   <2e-16 ***
genusgenD     -1.06030    0.06583 -16.106   <2e-16 ***
treatmentwater  0.25859    0.04710   5.491    4e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 669.34086  on 7  degrees of freedom
Residual deviance:   0.37316  on 3  degrees of freedom
AIC: 68.413

Number of Fisher Scoring iterations: 3
```

We can see from the table of coefficients that model `m2` has five parameters. The first one is a logistically transformed germination value for the "genA" in the control group. Below it are the differences for the other genera from the control treatment. We get the germination probabilities (relative frequency) by applying the anti-logit formula (12-1; do not forget to add the negative sign). Now we can see that the germination probability estimates for the first two genera under the control conditions, "genA" and "genB", are:

```
> 1/(1+exp(-0.56138))
[1] 0.6367718
> 1/(1+exp(-0.56138+1.59933))
[1] 0.2615457
```

Partial z-tests of all coefficients are significant, thus "genA" had higher germination than "genB", "genC" and "genD", both in the control and water-primed treatment. The last coefficient represents the additive effect for the "water" level. It shows how much we have to increase all previous values of coefficients to get probabilities of germination (on the logit scale) for the particular genera in the water-primed treatment.

Only two significant terms are now left in the model `m2`. We could finish the analysis here; however, apart from the effect of both factors, we are also interested in genus-specific germination, i.e. between which study genera there is a significant difference. Let us see if we can make the model even simpler. When we look at the values of the estimates for "genC" and "genD", we can see they are not much different (*–1.155 versus –1.06*). We will lump them together and see if it makes our previous model significantly worse. We create a copy of the *GENUS* factor and, using the **`levels`** command, we lump both genera together under a joint name of "genCD". We will test the simplification using the **`anova`** command and χ^2-statistics.

```
> genus1<-genus
> levels(genus1)
[1] "genA" "genB" "genC" "genD"
> levels(genus1)[3:4]<-"genCD"
> m3<-glm(y~genus1+treatment,binomial)
> anova(m2,m3,test="Chi")
Analysis of Deviance Table

Model 1: y ~ genus + treatment
Model 2: y ~ genus1 + treatment
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         3    0.37316
2         4    2.49523 -1 -2.12207   0.14519
```

The lumping did not cause a significant change in the residual deviance. So we will keep "genC" and "genD" lumped together. Once again, we will check the effects estimated after the pooling in the table of coefficients.

```
> summary(m3)
...
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.56141    0.05256   10.68  < 2e-16 ***
genus1genB   -1.59933    0.06860  -23.31  < 2e-16 ***
genus1genCD  -1.10723    0.05749  -19.26  < 2e-16 ***
treatmentwater  0.25852    0.04709    5.49 4.02e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 669.3409  on 7  degrees of freedom
Residual deviance:   2.4952  on 4  degrees of freedom
AIC: 68.535

Number of Fisher Scoring iterations: 3
```

In the model `m3`, there are four coefficients. The differences for "genB" and "genCD" from "genA" are both significant. The most similar are coefficients for "genB" and "genCD" (*–1.599 versus –1.107*). We try to lump them together. We make a copy of the variable *GENUS1* in which we replace "genB" and "genCD" by "genBCD". We compare the model obtained after the pooling with the model `m3` using χ^2-statistics.

```
> genus2<-genus1
> levels(genus2)
[1] "genA"  "genB"  "genCD"
> levels(genus2)[2:3]<-"genBCD"
> m4<-glm(y~genus2+treatment,binomial)
> anova(m3,m4,test="Chi")
Analysis of Deviance Table

Model 1: y ~ genus1 + treatment
Model 2: y ~ genus2 + treatment
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         4      2.495
2         5     73.684 -1  -71.189 3.246e-17
```

This time, the lumping led to a significant deviance change (significant worsening of the model). It is thus not appropriate to lump together all three genera. So, we will choose model `m3`:

$$\log\left(\frac{\pi_{jk}}{1-\pi_{jk}}\right) = \alpha + TREATMENT_j + GENUS1_k, \qquad (12\text{-}5)$$

where $germ_{jk} \sim Bin(\pi_{jk}, n_{jk})$, independent among pots.

From the original model (12-3), the model (12-5) differs only in the *GENUS1* factor levels. After pooling the original categories, it has smaller number of levels (k = genA, genB, genCD).

The first of the pre-set diagnostic plots, which we get by entering **`plot(m3, which=1)`**, will show us the relationship between the residual deviance (4-15) and the fitted values. It does not look like the model would systematically under-/overestimate the data. You can also inspect the other diagnostic plots; however, we will not present them here.

What will the result of the analysis be then? Instead of recalculating one coefficient after another, we will calculate the predicted values for all combinations of the levels from model `m3`, using a combination of the **`tapply`** and **`predict`** functions. The **`predict`** function will be called with the argument **`type="response"`** because we need to calculate probability estimates, not estimates of their logits. We will save these values to an object called `ge` in order to be able to work with them later.

```
> ge<-tapply(predict(m3,type="response"),list(treatment,genus1),mean)
> ge
              genA      genB    genCD
control 0.6367787 0.2615513 0.366835
water   0.6942213 0.3144487 0.428665
```

Germination in the control group was about 5% lower for all three genera than after water priming, as was already obvious from the former plot (Fig. 12-2). Despite that, the effect of the *TREATMENT* variable was highly significant. From a biological point of view, such a small increase of germination is practically negligible. The highly significant result is related to the great power of the large sample sizes used (1,000 seeds for each combination of levels). Generally speaking, the accuracy of estimates of study parameters improves with a growing number of replications. This also demonstrates that rejection of the null hypo-

thesis does not have to be important from a practical point of view. Statistical and factual importance might or might not go together. We should not connect them automatically! These are generally two different things.

Let's compute the 95% confidence intervals (CI_{95}) for all combinations of levels to visualise the significant differences among levels. First, we need to create a factor (which we will name `both`), with levels that will correspond to particular combinations of two original factors (*TREATMENT* and *GENUS1*). This new factor will have six levels (i.e. "control-genA", "control-genB", "control-genCD", "water-genA", "water-genB", and "water-genCD"). We use the function **`paste`** to combine the two factors together. Then we fit a model (`m4`) with the textbook parametrization, i.e. without the intercept (`-1`). Using the function **`confint`**, we obtain confidence intervals on the logit scale. They have to be transformed to the scale of relative frequencies using the antilogit function (12-1):

```
> both<-paste(pesticide,genus1)
> m4<-glm(y~factor(both)-1,binomial)
> 1/(1+exp(-confint(m4)))
Waiting for profiling to be done...
                              2.5 %    97.5 %
factor(both)control genA  0.6048442 0.6644666
factor(both)control genB  0.2315221 0.2857134
factor(both)control genCD 0.3485230 0.3908104
factor(both)water genA    0.6670153 0.7239840
factor(both)water genB    0.2896266 0.3473026
factor(both)water genCD   0.4044330 0.4477560
```

Here are the lower and upper limits of the 95% confidence intervals. We can see that none of the intervals in the control group or in the water priming group overlap inside the genera. However, the gap between the intervals is very small (on the second or third decimal place).

Finally, we will draw the estimated germination probability values and their intervals in a bar plot. We will call the **`barplot`** function. We will use the table with germination estimates (`ge`) for all levels from model `m3` as an argument. With the **`beside`** argument, we will place individual bars next to each other, and, by using the **`legend.text`** argument, we will add the legend. We will draw individual confidence intervals separately, first for the control group followed by the water priming group as vertical line segments (Fig. 12-3):

```
> barplot(ge,beside=T,ylab="Germination",legend.text=c("control","water"),
+ ylim=c(0,0.8))
> lines(c(1.5,1.5),c(0.605,0.665))
> lines(c(4.5,4.5),c(0.232,0.286))
> lines(c(7.5,7.5),c(0.349,0.391))
> lines(c(2.5,2.5),c(0.667,0.724))
> lines(c(5.5,5.5),c(0.29,0.347))
> lines(c(8.5,8.5),c(0.404,0.448))
```

CONCLUSION

While water priming caused significantly higher seed germination (GLM-b, $\chi^2_1 = 30.2$, $P < 0.0001$) in comparison with the control for all genera of the weeds, the difference was

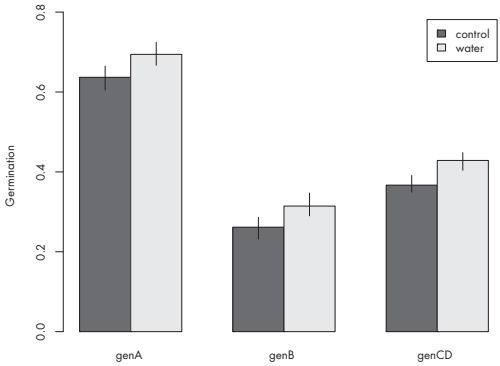


Fig. 12-3 Germination probability of seeds belonging to three weed genera ("genA", "genB", and "genCD") sown on control and water-primed ("water") soil. 95% confidence intervals are shown as vertical lines.

only 5 %. Germination was genus-specific (GLM-b, $\chi^2_3 = 638.8$, $P < 0.0001$). Weed seeds of "genA" had greater germination than the other genera (contrasts, $P < 0.0001$). Seed germination of genera "genC" and "genD" was similar (contrast, $P = 0.15$) and significantly greater than that of "genB" (contrast, $P < 0.0001$).

12.3 Overdispersion and underdispersion

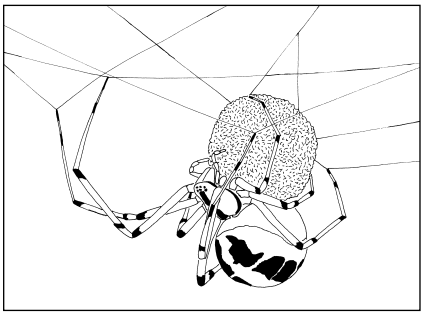
Similarly to the Poisson distribution, we might encounter overdispersion (or underdispersion) for otherwise binomial-like data. This means that the value of the dispersion parameter ($\phi = \mathrm{Var}(y)/\mathrm{E}(y)$) is not equal to one, as it should be under a binomial setting. Instead, it is markedly greater (or smaller in the case of underdispersion). This is because dispersion of the frequencies is greater (or smaller) than it should be according to the variance implied by the binomial distribution (see Chapter 12.1). You can expect this type of problem when you encounter a *markedly* greater (or smaller) residual deviance (unexplained variability indicator) than the residual degrees of freedom.

Overdispersion might arise when the model is incorrectly defined, for example, it lacks some important explanatory variable(s), or if the values of relative frequencies are not constant – they vary randomly within the same level (i.e. individuals in the sample are heterogeneous). A pragmatic solution to this is similar to what we saw in the context of the Poisson model. We can use **`quasibinomial`**, which does not specify a proper distribution; only a quasi-likelihood function can be used to calculate reasonable and pragmatic estimating equations based on the first two moments. The variance of the quasi-binomial model is calculated as $n\phi(1 - \pi)\phi$. The value of ϕ is frequently estimated from the ratio of Pearson residuals and the residual degrees of freedom (by the method of moments) – instead of $n\phi(1 - \pi)$, which would be the case for an exact binomial distribution (where $\phi = 1$). The **`quasibinomial`** choice does not affect estimates of probabilities, but it affects estimates of their standard errors, and consequently also the results of tests (p-values). In case of overdispersion (when

$\phi > 1$ the value of (overdispersion corrected) standard error will increase, while in case of underdispersion ($\phi < 1$) it will decrease. This will increase (or decrease) p-values. When the overdispersion is large, it is not surprising to find that a significant effect of an explanatory variable can become non-significant after the correction. Likewise, in case of underdispersion, the non-significant effect can become significant (which is quite dangerous). When overdispersion is severe, results based on naively applying the binomial model have a tendency to increase the number of false positive results (the false positive rate can be much larger than the nominal 5% significance level). After correction, by means of the **`quasibinomial`** argument the z-statistics will be automatically replaced by t-statistics. In the ANODEV table one needs to (manually) request the F-statistic instead of the χ^2-statistic.

Since overdispersion occurs often, for example, in environmental data, the next three examples will introduce models with overdispersion and underdispersion.

12.4 Regression



The laying of egg sacs (clutches of eggs) by spiders might be influenced by many variables, such as body size. In one experimental study, the laying of egg sacs was recorded in a spider species with a variable body size [mm] (*body*). As the body size was measured to a precision of 0.5 mm, all 160 female individuals used were classified into discrete size classes, each containing 15 to 30 individuals (*n*). The number of females that produced egg sac (*eggs*) within a couple of days was recorded for each size class. We will test the null hypothesis that egg sac laying is not related to female body size. If it is related, then we want to know: (1) What is the shape of the relationship? (2) We want to predict probability of egg sac laying for spider sizes ranging between 3 and 12 mm.

EDA

There is only one explanatory variable (*body*). We will consider it to be continuous despite the fact that the measured body sizes are discrete (the measurement step is small compared to the range of the *body* distribution). We will plot the relationship between egg sac production and body size using a scatter plot. Since sample sizes (*n*) differ among size groups, it is necessary to draw the relative frequency of egg sacs production (*p*).

```
> dat<-read.delim("spider.txt"); attach(dat); dat
   n body eggs
1 20  3.0    5
2 15  3.5   10
3 25  4.0   20
...
7 25  7.0   23
> p<-eggs/n
> plot(body,p)
```

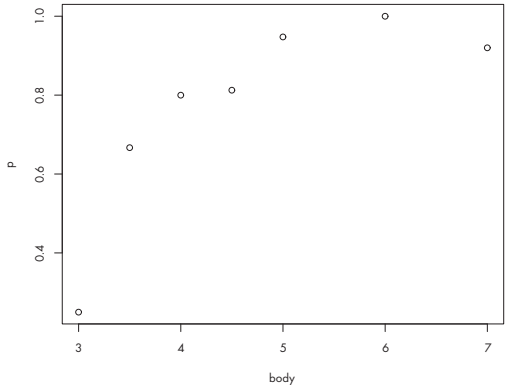


Fig. 12-4 Relationship between relative frequency (*p*) of spiders with an egg sac and their body size (*body*).

It is clear from the plot (Fig. 12-4) that the probability of egg sacs production increases with body size. This relationship is obviously not linear.

MODEL

Based on previous experience, we expect a linear trend on the logit scale. In order to be able to explain the relationship, we want to fit the logistic model, in which we get the following formula for the number of spiders with an egg sac (*eggs*) for the i^{th} body size:

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \alpha + \beta body_i, \tag{12-6}$$

where $eggs_i \sim Bin(\pi_i, n_i)$, independent among spiders.

ANALYSIS

First we (naively) fit a linear model (**`lm`**) based on normal error distribution. In order to stay within bounds given for the probabilities [0,1], we have to transform the relative frequencies. An angular transformation is often recommended. The shape of the relationship of the transformed response variable and the body size is also curved (Fig. 12-5A) and that is why it is advisable to use (at least) a quadratic model of the form:

$$\arcsin\sqrt{p_i} = \alpha + \beta body_i + \gamma body_i^2 + \varepsilon_i, \tag{12-7}$$

where $\varepsilon_i \sim N(0, \sigma^2)$, independent among spiders.

Since the relative frequency, p_i, was calculated from samples of different sizes, we will supply the information about varying precision in the form of weights to the model fitting procedure. In fact, we will use the **`weights`** argument within the **`lm`** call, which is what we have explained before, in Chapter 8.3.

```
> tr<-asin(sqrt(p))
> m1<-lm(tr~body+I(body^2),weights=n)
> summary(m1)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.34592    0.59329  -3.954  0.01676 *
body         1.30161    0.24776   5.254  0.00628 **
I(body^2)   -0.11121    0.02433  -4.571  0.01025 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4521 on 4 degrees of freedom
Multiple R-Squared: 0.9377,     Adjusted R-squared: 0.9066
F-statistic: 30.11 on 2 and 4 DF,  p-value: 0.003879
```

The presence of the quadratic term appears to be necessary. Let us try to find out how much worse the model will be without the quadratic term:

$$\arcsin\sqrt{p_i} = \alpha + \beta body_i + \varepsilon_i, \qquad (12\text{-}8)$$

where $\varepsilon_i \sim N(0,\sigma^2)$, independent among spiders.

```
> m2<-update(m1,~.-I(body^2))
> summary(m2)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.28836    0.31429   0.918   0.4010
body         0.17649    0.06279   2.811   0.0375 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.009 on 5 degrees of freedom
Multiple R-Squared: 0.6124,     Adjusted R-squared: 0.5349
F-statistic: 7.901 on 1 and 5 DF,  p-value: 0.03751
```

Overall, the model fits data substantially less well. While the value of the coefficient of determination (R^2) was for the quadratic model high (0.907), for the linear model it is much lower (0.535). This is well apparent from the plots. To draw them this time, we will use **`curve`** with the argument **`add=T`** for the quadratic model.. We take the estimated coefficients form `m1`. For the linear model we use **`abline`** (Fig. 12-5A):

```
> plot(body,tr)
> curve(-2.35+1.3*x-0.11*x^2,add=T)
> abline(m2,lty=2)
> legend(2.5,1.5,c("m1","m2"),lty=1:2)
```

The quadratic model (Fig. 12-5A) explains the relationship on the transformed scale relatively well. How would the fit of the models look on the original scale of relative frequencies? We will draw both models on a single plot again. To see the fit and predictions for both models, we will extend the y axis for the entire scale (i.e. 0 to 1) and the x axis for all spider

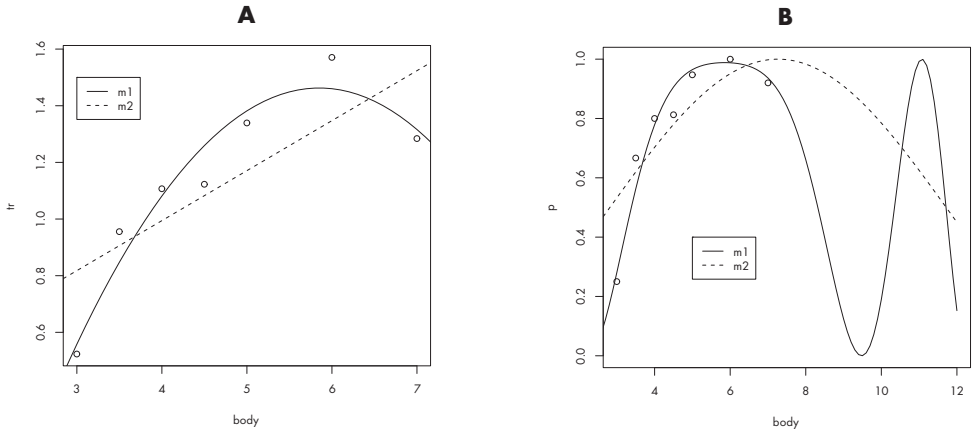


Fig. 12-5 A. Relationship between angularly transformed relative frequency of egg sacs (`tr`) and spider body size (*body*) with the quadratic (`m1`) and linear model (`m2`). **B.** Back-transformed quadratic (`m1`) and linear (`m2`) model of the relationship between relative frequency of females laying egg sacs and their body size.

sizes for which we would like to use the model (provided we are willing to extrapolate), for example, from 3 to 12 mm. We draw the curves using the **`curve`** function and the argument **`add`**. Nevertheless, we have to transform the linear predictor equation back to the relative frequency by using the inverse functions, i.e. the power of two and sine:

```
> plot(body,p,xlim=c(3,12),ylim=c(0,1))
> curve(sin(-2.35+1.3*x-0.11*x^2)^2,add=T)
> curve(sin(0.29+0.18*x)^2,add=T,lty=2)
> legend(5,0.4,c("m1","m2"),lty=1:2)
```

These curves are really interesting (Fig. 12-5B). The quadratic model (`m1`) is wavy, while the linear model (`m2`) is humped. What does it mean for the interpretation of the relationship? The quadratic model, `m1`, predicts that the probability of egg sac production will wildly fluctuate between 0 and 100%. That must seem absurd even to the most enthusiastic fans of complicated models. Even though the model describes the relationship within the study interval (3–7 mm) quite well, it is *absolutely* inappropriate for predictions outside this interval. Based on biological knowledge and experience, it is natural to expect a monotonous trend to the relationship: with probability close to zero for minimal body sizes and maximal probability for sizes over 6 mm. Linear model `m2` behaves a little better, but it also fails for values over 8 mm (i.e. in the interval not covered by data).

The models based on transformed data were not so great. Thus we now will proceed to logistic models, which are used almost exclusively in practical analyses today. Similarly to the previous linear predictor, we start with a quadratic formulation. For the i^{th} body size, we get:

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \alpha + \beta body_i + \gamma body_i^2, \tag{12-9}$$

where $eggs_i \sim Bin(\pi_i, n_i)$, independent among spiders.

In the first step we create a matrix of the response variable (`y`), which will have two columns: number of females with an egg sac (*eggs*) in the first one, and the number of females without the egg sac (*n–eggs*) in the second column. Right after fitting the model we look at the estimated parameters using **`summary`**.

```
> y<-cbind(eggs,n-eggs)
> m3<-glm(y~body+I(body^2),family=binomial)
> summary(m3)
...
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -13.7857     3.8482  -3.582 0.000340 ***
body          5.7218     1.6771   3.412 0.000645 ***
I(body^2)    -0.4825     0.1695  -2.846 0.004427 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 44.2136  on 6  degrees of freedom
Residual deviance:  3.3357  on 4  degrees of freedom
AIC: 26.135

Number of Fisher Scoring iterations: 4
```

Both parameters, the quadratic and the linear, are significantly different from zero; we thus have no reason to further simplify the quadratic model, but what about its shape? Does it really describe the trend of the relationship between egg sac production and body size better than the logistic model with only a linear term? We will again compare both of these models in a joint plot. Do not forget to fit linear model `m4` by removing the quadratic term.

```
> m4<-update(m3,~.-I(body^2))
> summary(m4)
...
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -3.9270     1.1038  -3.558 0.000374 ***
body          1.2079     0.2756   4.383 1.17e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 44.214  on 6  degrees of freedom
Residual deviance: 11.072  on 5  degrees of freedom
AIC: 31.872

Number of Fisher Scoring iterations: 5
```

This time we compare the two logistic models. We will use the estimated coefficients to specify the linear predictor formulas in the **curve** function. To transform the values to the scale of the relative frequencies we have to apply the anti-logit formula (12-1) on the linear predictor. Do not forget to reverse the signs in front of each parameter.

```
> plot(body,p,xlim=c(3,12),ylim=c(0,1))
> curve(1/(1+exp(13.79-5.72*x+0.48*x^2)),add=T)
> curve(1/(1+exp(3.93-1.21*x)),add=T,lty=2)
> legend(5,0.7,c("m3","m4"),lty=1:2)
```

The quadratic logistic model (Fig. 12-6) is better than the previous quadratic model `m2`, which had a transformed response variable, because it lacks the oscillatory behaviour, but, its prediction beyond 7 mm is still poor. As expected, the linear logistic model `m4`, which has a common logistic shape – meaning in this case that egg-laying increases with body size and for specimens larger than 6 mm it is nearly 100% – behaves more naturally. The quadratic model seems to be "too flexible". We know that the quadratic model is an extension of the linear model, and therefore it cannot be worse, certainly not for the data it was trained on, but it can certainly be worse when used for predictions.

Let us explore whether there is an overdispersion. Residual deviance is about twice as big as the residual degrees of freedom (as we can see from the output of the **summary** below). That is why we will fit a quasi-binomial model that corrects for overdispersion.

```
> m5<-update(m4,family=quasibinomial)
> summary(m5)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -3.9270     2.0150  -1.949   0.1088
body          1.2079     0.5031   2.401   0.0616 .
---
```

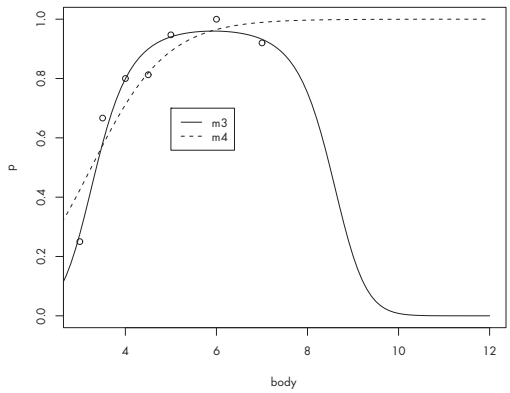


Fig. 12-6 Quadratic logistic (`m3`) and linear logistic (`m4`) models of the relationship between probability of egg sac laying (`p`) and body size (*body*).

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasibinomial family taken to be 3.332466)

    Null deviance: 44.214  on 6  degrees of freedom
Residual deviance: 11.072  on 5  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 5
```

The dispersion parameter is not equal to 1 anymore (as it was for the binomial error structure), but it is now *3.332*. Why could this be? Data for larger sample sizes show considerable variation, which causes overdispersion. This means that beside body size, at least one more explanatory variable (if it was measured) should be used (it could be quality of consumed prey, for example), which has an important effect on the egg-laying. Please note, that estimates of α and β have not changed in comparison with those for `m4`, but the standard errors increased and z-values were replaced by t-values corrected with ϕ. While in model `m4`, estimates of both parameters were significantly different from 0, in model `m5` none are significantly different at the 5 % level. As for the ANODEV table, used in case of overdispersion, we specify F-statistics – this change reflects the analogy to the normal linear model and the fact that overdispersion is corrected by using estimate of the unknown ϕ parameter.

```
> anova(m5,test="F")
...
     Df Deviance Resid. Df Resid. Dev     F  Pr(>F)
NULL                     6     44.214
body  1   33.141         5     11.072 9.945 0.02528 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Even after applying the overdispersion correction, the effect of *body* remains significant. This means that nothing serious (i.e. qualitatively different) has happened with the interpretation after the model change.

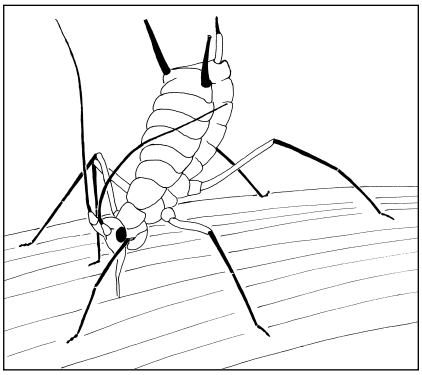
We expected the measured relative frequencies in the model to have binomial distributions and be independent. If you do the diagnostics calling **`plot`**, you will find out that it is hardly possible to assess the homogeneity of the deviance residuals or the shape of their distribution from those few points. We know that the experiment has been prepared in a way that made the measurements independent. That is why we believe it really is the case and hope that the binomial distribution assumptions are met. That does not entitle us to think we definitely have "the correct" model, but we do not currently have a better model available. The model should be verified using new (independent) data. Only that can verify the predictive abilities and generalisability.

CONCLUSION

The egg sac production probability for the spider studied significantly increased with increasing body size (GLM-b, $\chi^2_1 = 33.1$, $P = 0.025$). We can estimate the egg sac laying probability estimate using the following formula:

$$\frac{100}{1+\exp(3.927-1.208\,body)}.$$

12.5 One-way ANCODEV



Synthetic insecticides often have a species-specific efficiency. Recommended doses or concentrations then have to be adjusted. The effect of a particular insecticide preparation upon the mortality (*dead*) of two aphid species (*SPECIES*: A, B) was studied in a laboratory. The insecticide was applied at six concentrations [ppm] (*conc*). Each concentration was tested on 30 individuals (*n*) of both aphid species. We are interested in answering the following three questions: (1) Was mortality affected by the concentration? (2) Was the insecticide efficiency similar for both aphid species? (3) What is the LC_{50} (for each species separately)?

EDA

There are two explanatory variables in the data frame – one is categorical (*SPECIES*), the other is continuous (*conc*). So we will plot data for both aphid species onto one plot. Levels of the factor *SPECIES* were named on purpose using a single letter ("A" and "B") in order to use letters in the plot instead of symbols. The function **`text`** with an argument **`labels=as.character`** can do for us, but first we have to calculate relative frequency of mortality (*p*) using numbers of dead (*dead*) and the sample sizes (*n*).

```
> dat<-read.delim("aphid.txt"); attach(dat); dat
   species conc dead  n
1        A 0.14    7 30
2        A 0.34   18 30
3        A 0.48   20 30
...
12       B 5.50   27 30
> p<-dead/n
> plot(conc,p,type="n")
> text(conc,p,labels=as.character(species))
```

Mortality for both aphid species increases with the concentration, but differently in each species. In species "A", it increases from 20 to 100 % and in species "B" from 0 to 90 % (Fig. 12-7). While species "A" has the highest mortality already at a concentration around 0.5 ppm, species "B" has similar mortality at about a 7 times higher concentration. Notice that tested concentrations are not placed equidistantly, but they increase approximately by a multiple of 2.

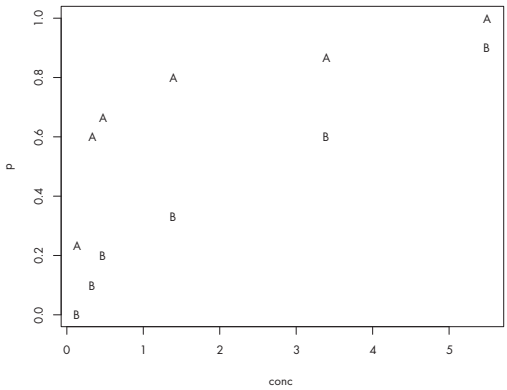


Fig. 12-7 Relationship between empirical estimates of mortality (*p*) for two aphid species ("A", "B") and the concentration of pesticide (*conc*).

MODEL

We can expect that mortality increases with increasing concentration of the pesticide. We will logarithmically transform the continuous explanatory variable, which is commonly done when the concentrations are powers of the lowest concentration – provided no zero concentration has been used. We will expect a logistic model for both species (linear on the logit scale). The values of the intercept and the slope can generally differ among species of aphids. We expect the following model for the i^{th} concentration and the j^{th} *SPECIES* (we will use the default treatment parametrization):

$$\log\left(\frac{\pi_{ij}}{1-\pi_{ij}}\right) = \alpha + SPECIES_j + \beta \log(conc_i) + \delta_j \log(conc_i)\,, \qquad (12\text{-}10)$$

where $eggs_i \sim Bin(\pi_i, n_i)$, independent among dishes.

ANALYSIS

Since we are interested in the difference between the species after the correction for concentration, the linear predictor will have a covariate first, followed by the studied factor. Prior to that, we still have to create the response variable `y` from the number of killed and the total number of tested specimens.

```
> y<-cbind(dead,n-dead)
> m1<-glm(y~log(conc)*species,binomial)
> anova(m1,test="Chi")
...
                  Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                                    11    185.807
log(conc)          1  110.170        10     75.638 8.996e-26
species            1   62.087         9     13.551 3.286e-15
log(conc):species  1    1.343         8     12.207     0.246
```

Interaction is not significant. On the other hand, both main effects are significant. We will simplify the model by removing the interaction using the **update** function.

```
> m2<-update(m1,~.-log(conc):species)
> anova(m2,test="Chi")
...
          Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                         11    185.807
log(conc)  1  110.170        10     75.638 8.996e-26
species    1   62.087         9     13.551 3.286e-15
```

In this model, both terms are significant and, therefore, cannot be removed. How many parameters did this model estimate? Model `m1` had four parameters, two intercepts and two slopes. Removal of the interaction showed that fewer parameters are sufficient. A significant effect of *SPECIES* says that both intercepts are needed, but the non-significant effect of the interaction log(*conc*):*SPECIES* indicates that just one universal slope (common for both species) is sufficient. Model `m2` then has three parameters.

```
> summary(m2)
...
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.3825     0.2201   6.280 3.39e-10 ***
log(conc)     1.2328     0.1348   9.146  < 2e-16 ***
speciesB     -2.2117     0.3180  -6.955 3.52e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 185.807  on 11  degrees of freedom
Residual deviance:  13.551  on  9  degrees of freedom
AIC: 54.031

Number of Fisher Scoring iterations: 4
```

Intercept is an estimate for aphid species "A" (on the logit scale). In the second line there is an estimate of the common slope (*b*). In the third line, there is a difference between intercept for species "B" and species "A". Species "B" has a significantly lower intercept value (p-value < 0.0001). If we apply the exponential function to the absolute difference, exp(2.2117) = *9.131*, we will get an odds ratio for the mortality of the two aphid species. In this case, the odds for species "A" are nine times higher than those of species "B".

Furthermore, we can read from the output that in the model `m2`, there is a weak hint of possible overdispersion. Should we specify **quasibinomial**, we would learn that the dispersion parameter is *1.4*. Correction for overdispersion would not lead to dramatic changes in p-values.

Let's look at the deviance residuals related to the fitted values and Pearson residuals related to the covariate log(*conc*).

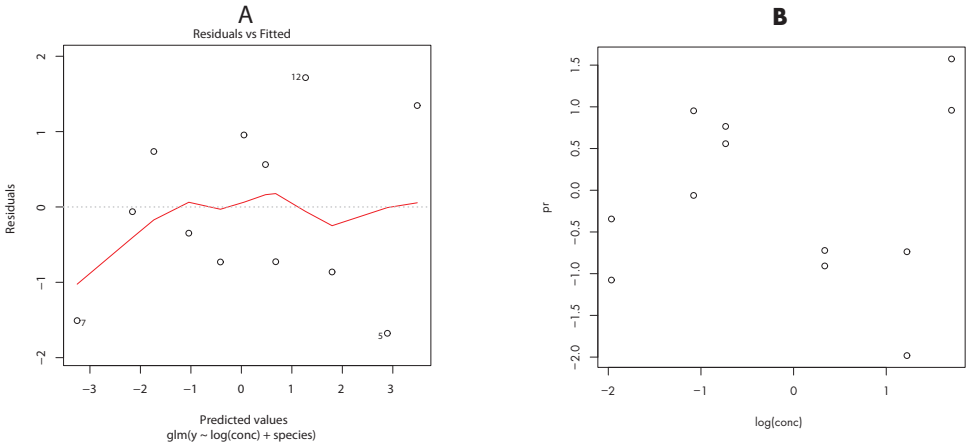


Fig. 12-8 A. Relationship between deviance residuals and fitted values. **B**. Relationship between Pearson residuals and values of log(*conc*).

```
> plot(m2,which=1)
> pr<-resid(m2,type="pearson")
> plot(log(conc),pr)
```

The diagnostic plots (Fig. 12-8A, B) do not suggest any essential problems. The estimates look reasonable, as we will demonstrate in a moment. Hence, we will report the following model (in the treatment parametrization) as a result of our analysis:

$$\log\left(\frac{\pi_{ij}}{1-\pi_{ij}}\right) = \alpha + SPECIES_j + \beta \log(conc_i), \tag{12-11}$$

where $eggs_i \sim Bin(\pi_i, n_i)$, independent among dishes.

Before we plot estimated values of mortalities, we have to do the following steps: (1) Draw an empty plot with axis labels and the abscissa on the log-scale; (2) Create an auxiliary vector x; (3) Use **lines** with several arguments to draw predicted values – in order to get real probabilities (and not logits) we have to use the **type="response"** argument in the **predict** function; (4) Add a legend. The graph (Fig. 12-9) is then obtained as follows:

```
> plot(conc,p,type="n",log="x",xlab="Log(Concentration)",
+ ylab="Mortality")
> x<-seq(0,6,0.1)
> lines(x,predict(m2,list(conc=x,species=factor(rep("A",length(x)),
+ levels=levels(species))),type="response"))
> lines(x,predict(m2,list(conc=x,species=factor(rep("B",length(x)),
+ levels=levels(species))),type="response"),lty=2)
> legend(2,0.3,c("A","B"),lty=1:2)
```

We thus achieved our first goal. Apart from determining the model parameters, we also wanted to determine the 50% lethal concentrations (LC_{50}) for both species, i.e. estimates of

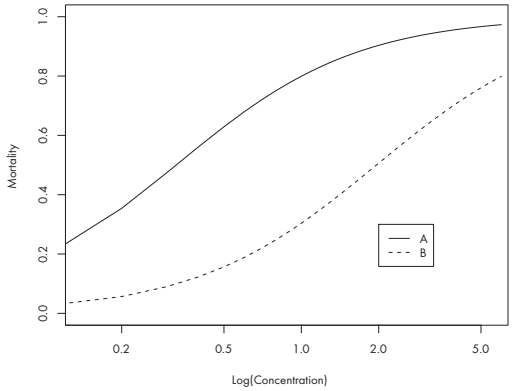


Fig. 12-9 Logistic models of the relationship between mortality and the logarithm of concentration for two aphid species ("A", "B").

the concentrations, at which 50 % of aphids die. These values can be simply computed from the parameter estimates using the following formula:

$$\mathrm{LC}_{50} = \exp\left(-\frac{a}{b}\right), \tag{12-12}$$

There is a function which also estimates standard errors of LC values. It is in the library MASS and it is called **`dose.p`**. It has three arguments: the first is the name of the fitted model object, the second (**`cf`**) specifies which coefficients from the model are to be used, and the third (**`p`**) is the level for LC ranging between 0 and 1 (for LC_{50}, we use $p = 0.5$). Because in the summary output of the model `m2`, the intercept for species "B" is not shown, instead there is a difference from the intercept for species "A", we have to re-parametrise the model using the textbook parametrization in order to get both values of intercept and slope conveniently. This means to fit a model of the form:

$$\log\left(\frac{\pi_{ij}}{1-\pi_{ij}}\right) = SPECIES_j + \beta \log(conc_i). \tag{12-13}$$

We will get this model by removing the explicit intercept parameter. We will tell **`glm`** to do so by using **`-1`** in the model formula.

```
> m3<-glm(y~species+log(conc)-1,binomial)
> summary(m3)
...
Coefficients:
          Estimate Std. Error z value Pr(>|z|)
speciesA    1.3825     0.2201   6.280 3.39e-10 ***
speciesB   -0.8293     0.2020  -4.106 4.02e-05 ***
log(conc)   1.2328     0.1348   9.146  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 186.708  on 12  degrees of freedom
Residual deviance:  13.551  on  9  degrees of freedom
AIC: 54.031

Number of Fisher Scoring iterations: 4
```

The first coefficient is the intercept for species "A". Similarly, the second coefficient is the intercept for species "B". The third coefficient is a common slope (all on a logistic scale). This means that to calculate LC_{50} for species "A" we will use the first and the third coefficients, and for species "B" we will use the second and the third coefficients. Note and remember that the argument **`cf`** is sensitive to the order of coefficients: the first value has to refer to the intercept and the second to the slope. So, LC_{50} and the standard error (SE) for both species are:

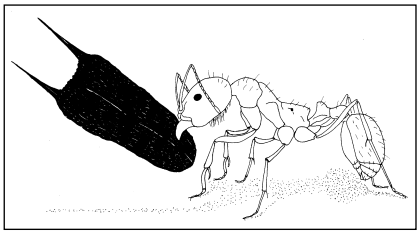
```
> library(MASS)
> dose.p(m3,cf=c(1,3),p=0.5)
             Dose        SE
p = 0.5: -1.121418 0.1627097
> dose.p(m3,cf=c(2,3),p=0.5)
             Dose        SE
p = 0.5: 0.6726813 0.159251
```

–1.121 for species "A" and *0.673* for species "B". Both values are on the logarithmic scale, because in the model we used a logarithmically transformed explanatory variable *conc*. To get values on the original scale, i.e. concentration in ppm, we have to apply the exponential function: exp(–1.121) = *0.326* and exp(0.673) = *1.96*.

CONCLUSION

Efficiency of the insecticide preparation increased significantly with the concentration (GLM-b, $\chi^2_1 = 110.2$, $P < 0.0001$), but differently for the two aphid species (GLM-b, $\chi^2_1 = 62.1$, $P < 0.0001$). Species "B" had overall lower mortality than species "A". The estimated model for species "A" that gives percentage efficiency is: $\frac{100}{1+\exp(-1.383-1.233\log(conc))}$. Similarly for species "B", it is: $\frac{100}{1+\exp(0.829-1.233\log(conc))}$. LC_{50} for species "A" is 0.33 ppm and for species "B" it is 1.96 ppm.

12.6 Binary one-way ANCODEV



Granivorous (i.e. seed eating) ants collect various seeds and bring them into their underground nests. Sympatrically occurring ant species may show trophic niche partitioning with respect to the size of collected seeds. Seed preference of two ant species (*SPECIES*: specA, specB) was studied in a laboratory. Each of 25 ant individuals of both species was offered seeds of variable size (*seed*) expressed as its weight [mg]. The response of ants was classified in a binary way, as "yes" or "no", if ant took or refused to take a seed re-

spectively. We wish to know the answers to the following questions: (1) Was acceptance related to the seed size? (2) Did both species have similar preferences for seed sizes? (3) If not, what is the threshold size (the threshold is defined as a size that is accepted with higher than 90% probability) of seeds for both species?

EDA

As mentioned above, in contrast to previous examples in this chapter, the response variable (*take*) is binary, i.e. coded using 0 or 1. "1" stands for taken and "0" stands for rejected. Thus for each value of *seed* there is only one observation ($n = 1$). There are two explanatory variables, one is categorical, the factor (*SPECIES*), and the other is continuous, the covariate (*seed*). To show the data we will use a scatter plot **`xyplot`** function from the library lattice.

```
> dat<-read.delim("ant.txt"); attach(dat); dat
     seed take species
1  0.0540    1   specA
2  0.0757    1   specA
3  0.0826    1   specA
...
50 2.6917    1   specB
> library(lattice)
> xyplot(take~seed|species,col=1)
```

We obtain two strange, but frequently used, plots arranged as panels (Fig. 12-10). The data are arranged only at two values of the response variable (0s or 1s), so it is not easy to see any pattern. After a closer look, we can see that points for "specA" and "specB" are not overlapping, which means that the two ant species show a different preference for seeds. Ants of "specA" accepted rather small seeds, while ants of "specB" accepted larger seeds. This indicates that interaction between *seed* and *SPECIES* might be significant.

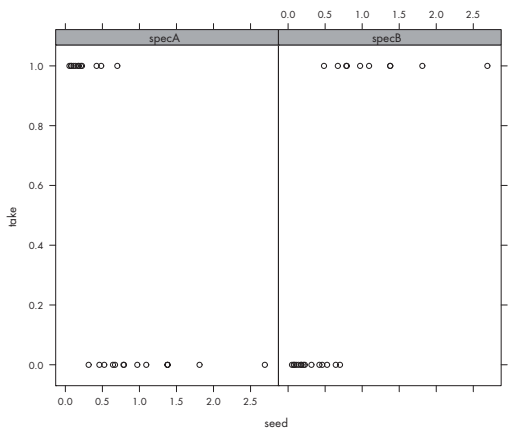


Fig. 12-10 Comparison of acceptance (*take*) of various sized seeds for two ant species: "specA" and "specB".

MODEL

Based on previous experience with similar data we expect that the probability of accepting seeds will change with the seed mass in a logistic fashion. So we specify a model with species specific intercepts and slopes. We expect the model to take the following form and use it in the treatment parametrization:

$$\log\left(\frac{\pi_{ij}}{1-\pi_{ij}}\right) = \alpha + SPECIES_j + \beta seed_{ij} + \delta_j seed_{ij}, \qquad (12\text{-}14)$$

where $take_{ij} \sim Bin(\pi_{ij}, n_{ij})$, independent among ants.

α and β are the intercept and slope, respectively, for "specA", while $SPECIES_j$ and δ_j represent differences from intercept and slope, respectively, for "specB" from those of "specA".

ANALYSIS

The responses are given as Bernoulli variables so that $n = 1$ for each observation and we do not need to specify a vector representing number of observations for each row. In contrast to all previous examples in this chapter, the response variable will be supplied just as a single vector. The logistic model will include both main effects and their interaction.

```
> m1<-glm(take~seed*species,family=binomial)
> summary(m1)
...
Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)          4.012      1.646   2.437  0.01480 *
seed                -8.346      3.315  -2.517  0.01182 *
speciesspecB       -10.957      3.697  -2.964  0.00304 **
seed:speciesspecB   19.147      6.141   3.118  0.00182 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 68.593  on 49  degrees of freedom
Residual deviance: 24.726  on 46  degrees of freedom
AIC: 32.726

Number of Fisher Scoring iterations: 8
```

Should we fit a model with **quasibinomial** family, we would learn that the dispersion parameter is slightly smaller than 1. This indicates that there is underdispersion in the model, which is rather uncommon in models with binomial error structure but not uncommon in models with binary error structure. Yet all p-values of z-statistics in the summary table are already significant at $\alpha = 0.05$, so implementation of a correction for underdispersion would further decrease the p-values. This would not change the interpretation of the model.

```
> anova(m1,test="Chi")
```

```
Analysis of Deviance Table

Model: binomial, link: logit

Response: take

Terms added sequentially (first to last)

             Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                         49     68.593
seed          1    0.054        48     68.539     0.817
species       1    0.325        47     68.214     0.568
seed:species  1   43.488        46     24.726 4.267e-11
```

Of all terms in the model, only interaction is significant. As the interaction stands above the two main effects, it cannot be removed according to the marginality rule. The model `m1` includes four parameters. These are the coefficients of the two intercepts (α_1, α_2) and the two slopes (β_1, β_2). The first two values in the summary output represent intercept and slope for "specA", followed by a difference (on the logistic scale) from "specA" for intercept and slope, respectively, of "specB".

We may try to improve the model by using logarithmic transformation of the covariate *seed*. Let's specify the model `m2` again in the treatment parametrization:

$$\log\left(\frac{\pi_{ij}}{1-\pi_{ij}}\right) = \alpha + SPECIES_j + \beta \log(seed_i) + \delta_j \log(seed_i), \qquad (12\text{-}15)$$

where $take_{ij} \sim Bin(\pi_{ij}, n_{ij})$, independent among ants.

Now we compare the two models. These two models are not nested, i.e. the latter is not a simplified form of the former, as the only difference between (12-14) and (12-5) is the logarithm of *seed*. In such cases, we cannot use a test statistic as we did many times before simply because the two models have identical degrees of freedom. We have to use another measure, such as AIC, which we get using a command of the same name.

```
> m2<-glm(take~log(seed)*species,binomial)
> AIC(m1,m2)
   df      AIC
m1  4 32.72631
m2  4 32.23823
```

The AIC values of the two models are very similar (differing in less than 1 unit) – transformation of the covariate *seed* did not improve the model substantially. Diagnostic plots of the two models (not shown) reveal that the distribution of the residuals of `m2` is similar to that of `m1`, thus we will prefer the somewhat simpler model `m1`.

Eventually, we will draw the model for the two species onto one plot (Fig. 12-11) using the procedure described in detail in the Chapter 10.5.

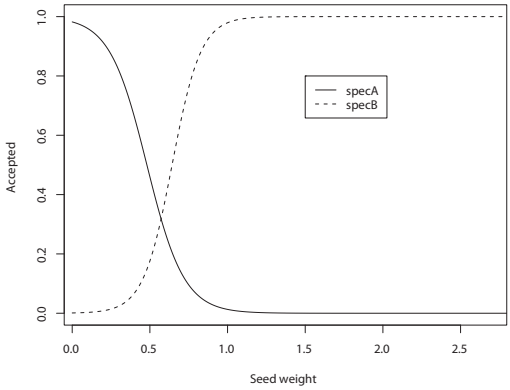


Fig. 12-11 Models of the relationship between acceptance probability and seed weight for two ant species ("specA", "specB").

```
> par(mfrow=c(1,1))
> plot(seed,take,type="n",xlab="Seed weight",ylab="Accepted")
> x<-seq(0,3,0.01)
> lines(x,predict(m1,list(seed=x,species=factor(rep("specA",length(x)),
+ levels=levels(species))),type="response"))
> lines(x,predict(m1,list(seed=x,species=factor(rep("specB",length(x)),
+ levels=levels(species))),type="response"),lty=2)
> legend(1.5,0.8,c("specA","specB"),lty=1:2)
```

It remains to identify how big are the seeds the two ant species accepted. Specifically, we need to estimate threshold limits of seed mass that would be accepted by each species with 90 % chance. We use the LC approach, but specify a different level. So we will modify the logit regression equation for $p = 0.9$ and extract x. For 90% acceptance the equation will be

$$x = \frac{\log\left(0.9/0.1\right) - a}{b}. \tag{12-16}$$

Replacing estimates for intercept (a) and slope (b) for each species into (12-16) we obtain the requested threshold limits. At first for "specA" and then for "specB":

```
> (log(0.9/0.1)-4.012)/-8.346
[1] 0.2174425
> (log(0.9/0.1)-4.012+10.957)/(-8.346+19.147)
[1] 0.8464239
```

So the 90% upper limit mass of seed for "specA" is estimated to be *0.217* mg, whereas the 90% lower limit mass of transported seed for "specB" is estimated to be *0.846* mg.

CONCLUSION

The two ant species accepted seeds of significantly different mass (GLM-b, $\chi^2_1 = 43.5$, $P < 0.0001$). "specA" selected tiny seeds up to 0.22 mg, whereas specB" selected bigger seeds larger than 0.85 mg. The estimated model of seed acceptance probability for "specA" is $\frac{1}{1+\exp(-4.012+8.346\,seed)}$ and for "specB" it is $\frac{1}{1+\exp(6.945-10.8\,seed)}$.

REFERENCES

Burnham K. P. & Anderson D. R. 2002. *Model Selection and Multimodel Inference: a Practical Information-Theoretic Approach.* 2nd ed. Springer, New York.

Carroll R. J., Ruppert D. & Stefanski L. A. 1995. *Measurement Error in Nonlinear Models*. Chapman and Hall/CRC, New York.

Cleveland W. S. 1993. *Visualizing Data*. Hobart Press, Summit.

Cleveland W. S. & Devlin S. J. 1988. Locally-weighted regression: An approach to regression analysis by local fitting. *Journal of American Statistical Association* 83: 596–610.

Cochran W. G. & Cox G. M. 1957. *Experimental Designs.* Wiley & Sons, New York.

Crawley M. J. 1993. *GLIM for Ecologists*. Blackwell Science, Oxford.

Crawley M. J. 2002. *Statistical Computing. An Introduction to Data Analysis Using S-Plus.* Wiley & Sons, Chichester.

Dalgaard P. 2008. *Introductory Statistics with R*. Springer, New York.

Davison A. C. 2008. *Statistical Models*. Cambridge University Press, Cambridge.

De Boor C. 2001. *A Practical Guide to Splines.* Revised Ed. Springer, New York.

Efron B. & Tibshirani R. 1993. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, Boca Raton.

Faraway J. J. 2004. *Linear Models with R*. Chapman & Hall/CRC, Boca Raton.

Hjelm J. & Persson L. 2001. Size-dependent attack rate and handling capacity: inter-cohort competition in a zooplanktivorous fish. *Oikos* 95: 520–532.

Holling C. S. 1965. The functional response of predators to prey density and its role in mimicry and population regulation. *Memoirs of the Entomological Society of Canada* 45: 1–60.

Hurd L. E. & Fagan W. F. 1992. Cursorial spiders and succession: age or habitat structure? *Oecologia* 92: 215–221.

Ihaka R. & Gentleman R. 1996. R: a language for data analysis and graphics. *Journal of Computational and Graphical Statistics* 5: 299–314.

Kontodimas D. C., Eliopoulos P. A., Stathas G. J. & Economou L. P. 2004. Comparative temperature-dependent development of *Nephus includens* (Kirsch) and *Nephus bisignatus* (Boheman) (Coleoptera: Coccinellidae) preying on *Planococcus citri* (Risso) (Homoptera: Pseudococcidae): evaluation of a linear and various nonlinear models using specific criteria. *Environmental Entomology* 33: 1–11.

Li D. 2002. The combined effects of temperature and diet on development and survival of a crab spider, *Misumenops tricuspidatus* (Fabricius) (Araneae: Thomisidae). *Journal of Thermal Biology* 27: 83–93.

Mittlböck M. & Heinzl H. 2002. Measures of explained variation in gamma regression models. *Communications in Statistics - Simulation and Computation* 31: 61–73.

Montgomery D. C. 2001. *Design and Analysis of Experiments.* Wiley & Sons, New York.

Montgomery D. C. & Runger G. C. 1994. *Applied Statistics and Probability for Engineers*. John Wiley & Sons, New York.

Morris C. N. 2006. Natural exponential families. *Encyclopedia of Statistical Sciences*, Vol 8. Wiley & Sons, New York.

Murrell P. 2005. *R Graphics.* Chapman & Hall/CRC, Boca Raton.

Pekár S. & Brabec M. 2012. *Modern Analysis of Biological Data. 2. Linear Models with Correlation in R*. Masaryk University Press, Brno. [In Czech]

Popper K. 1959. *The Logic of Scientific Discovery*. Hutchinson, London.

Press W. H., Teukolsky S. A., Vetterling W. T. & Flannery B. P. 2007. *Numerical Recipes: The Art of Scientific Computing.* 3rd ed. Cambridge University Press, New York.

Quinn G. P. & Keough M. J. 2002. *Experimental Design and Data Analysis for Biologists*. Cambridge University Press, Cambridge.

R Core Team 2015. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. Avalable at https://www.R-project.org/.

Rawlings J. O. 1988. *Applied Regression Analysis: A Research Tool*. Wadsworth & Brooks/Cole, Pacific Grove.

Scheiner S. M. & Gurevitch J. (eds) 2001. *Design and Analysis of Ecological Experiments.* 2nd ed. Oxford University Press, Oxford.

Underwood A. J. 1997. *Experiments in Ecology. Their Logical Design and Interpretation Using Analysis of Variance*. Cambridge University Press, Cambridge.

Venables W. N. & Ripley B. D. 2002. *Modern Applied Statistics with S*. Springer-Verlag, New York.

Zoonedkynd V. 2007. Programming in R. Available at http://zoonek2.free.fr/UNIX/48_R/02.html

Subject index

R functions and their arguments

Modern Analysis of Biological Data:
Generalized Linear Models in R

Illustrated by Stano Pekár
Design by Ivo Pecl, Stano Pekár, and Grafique
English proof-reading by Michael Palamountain

Published by Masaryk University
Brno 2016
First Edition
Printed by Reprocentrum, a.s., Bezručova 29, 678 01 Blansko

ISBN 978-80-210-8019-5

http://www.muni.cz/press/books/pekar_en