

Linear Regression
and
Generalized Linear Models

W. John Braun
University of British Columbia

March 8, 2021

... essentially, all models are wrong, but some are useful ...
[G.E.P. Box]

... the answer to all statistical questions: “It depends ...” ...
[anonymous]

© W.J. Braun 2021
This document may not be copied without the permission of the author.

Preface

This book contains an introduction to the theory of linear and generalized linear models, and it serves as a primer on splines, generalized additive models and mixed-effects models. It should be accessible to anyone who has taken courses in elementary statistics as well as differential and integral calculus. Having some familiarity with matrices and vectors as well as prior exposure to simple linear regression will make the experience of reading this book that much easier. A quick review of what is needed from probability, asymptotic statistics and linear algebra can be found in one of the book's appendices.

The book also exposes the reader to computational aspects of regression modelling, primarily through the use of the R language. In order for the reader to appreciate this aspect of the presentation, very little prior experience of R is required, apart from having a reasonably up-to-date working copy and some very minimal understanding of the basic syntax.

The aim of this book is to provide the reader with sufficient theoretical detail so that aspects of regression modelling that are often seen as a ‘black-box’ are illuminated in such a way as to remove most of the mystery. Applications and simulation will be used as props to illustrate the theory. The reader will then be well-placed to read other more applications-oriented account with little difficulty and with a greater depth of understanding. This is my hope, at any rate.

W. John Braun
January, 2021

Contents

Preface	iii
1 R, RStudio and Statistical Modelling	1
1.1 R and RStudio	1
1.1.1 Packages	1
1.1.2 Sources of Additional Information	1
1.2 Types of data	2
1.3 Data collection	2
1.4 Graphic and numeric summaries	3
1.5 Mathematical and statistical modeling	4
1.5.1 Mathematical models	4
1.5.2 The need for statistical models	5
1.6 A model building strategy	7
1.7 Questions that can be addressed by statistical models	7
1.8 Classifying statistical models by data type	8
1.9 Exercises	8
2 Simple Regression	10
2.1 The model	10
2.2 Parameter estimation, fitted values and residuals	11
2.3 Unbiased estimation of σ^2 : MSE	12
2.4 Statistical and geometric consequences of least-squares	13
2.4.1 R calculations	15
2.4.2 Using extractor functions to obtain partial output	15
2.4.3 Moment properties of least-squares estimates	16
2.4.4 Standard error estimators	16
2.4.5 Distributions of $\hat{\beta}_1$ and $\hat{\beta}_0$	17
2.5 Inferences about the regression parameters	17
2.5.1 Inference for β_1	17
2.5.2 $(1 - \alpha)$ Confidence Intervals	18
2.5.3 Confidence interval for mean response	18
2.5.4 Predicted responses	19
2.6 Analysis of variance: breaking down (analyzing) variation	20
2.6.1 Relation between SSR and β_1	21
2.6.2 Expected Sums of Squares	21
2.6.3 Another approach to testing $H_0 : \beta_1 = 0$	22
2.6.4 The ANOVA table	22
2.7 R^2 - coefficient of determination	23
2.7.1 Relation of R^2 to the correlation coefficient	23
2.7.2 Interpreting the correlation coefficient	24

2.7.3	Another interpretation of R^2	24
2.7.4	Cautions	25
2.8	Hazards of regression	25
2.9	Exercises	26
3	Normal Distribution Theory	30
3.1	Normality and the QQ-plot	30
3.2	Random vectors and their expectations	32
3.2.1	The variance-covariance matrix	33
3.2.2	Linear combinations of random variables	33
3.2.3	The variance-covariance matrix of $\mathbf{A}\mathbf{z}$	34
3.2.4	Independence and expectation	35
3.2.5	Characterizing normally distributed random vectors	35
3.2.6	The distribution of $\mathbf{A}\mathbf{x}$	36
3.2.7	The distribution of a linear combination of normal random variables	37
3.3	Understanding independence	37
3.3.1	Graphical views of independence and dependence	37
3.3.2	Independence of the sample mean and variance	41
3.4	Random variables constructed from normals	42
3.4.1	The χ^2 random variable	43
3.4.2	The F random variable	44
3.4.3	The t random variable	45
3.5	Exercises	46
4	Multiple Regression	48
4.1	A motivating example: house price data	48
4.1.1	Developing multiple regression models	48
4.1.2	Matrix form	48
4.2	Least-squares estimation via the QR decomposition	50
4.2.1	Obtaining the QR decomposition of the design matrix	50
4.2.2	Estimation via QR	51
4.2.3	Estimation of the noise variance	52
4.3	Inference	53
4.3.1	Statistical properties of $\hat{\beta}$	53
4.3.2	Estimating the mean response at \mathbf{x}_0	53
4.3.3	t statistics	55
4.3.4	F statistics	58
4.3.5	Significance of regression	59
4.3.6	Obtaining the output all at once with <code>lm()</code>	59
4.4	Variable selection	61
4.4.1	Traditional methods for chasing wild geese	61
4.4.2	Modern approaches to variable selection	61
4.5	Exercises	69
5	Model Assessment and Adjustment	76
5.1	Regression assumptions and diagnostics	76
5.1.1	Consequences of model failure	76
5.1.2	Validation or assessment?	77
5.1.3	Types of residuals	77
5.1.4	PRESS - Predicted Residual Sum of Squares	78
5.2	Residual plots	79
5.2.1	Basic plots of residuals	79
5.2.2	Added variable plots or partial regression plots	81

5.2.3	Checking the normal assumption	83
5.3	Serial correlation among the errors	84
5.3.1	Lag plots and the autocorrelation function	85
5.3.2	Formal testing for autocorrelation	87
5.3.3	Caution in the use of R^2	88
5.4	Transformations on the response variable	88
5.4.1	Variance-stabilizing transformations	88
5.4.2	Box-Cox transformations	90
5.5	Detection and treatment of outliers	94
5.5.1	Handling Outliers	94
5.5.2	Leverage and influence diagnostics	96
5.5.3	Leverage theory	96
5.5.4	Measuring leverage – hat diagonal values	97
5.5.5	Measuring Influence - Cook's D	99
5.5.6	Measuring effects on regression coefficients due to influence – DFBETAS	100
5.5.7	Measuring effects on fitted values due to influence – DFFITS	102
5.5.8	Summary	102
5.6	Exercises	103
6	Bending and Breaking the Regression Assumptions	106
6.1	Use of indicator variables in simple regression	106
6.1.1	Error variance and standard error estimates	109
6.1.2	A test for a difference in the means	110
6.1.3	Implementation with <code>lm()</code>	110
6.1.4	Analysis of variance: comparison of several treatments	111
6.2	Analysis of Covariance	112
6.2.1	Regression with indicator variables	112
6.2.2	Estimation, testing and prediction; implementation in R	113
6.2.3	Two interpretations of the analysis of covariance	115
6.3	Standardized Regression and Multicollinearity	116
6.3.1	Standardized regression	118
6.3.2	Multicollinearity and variance inflation	122
6.3.3	Effects of multicollinearity	122
6.4	Regression with counts	124
6.4.1	Nonlinear least-squares	126
6.4.2	Weighted least-squares	127
6.4.3	Estimating coefficient standard errors	128
6.4.4	Modelling binary responses	130
6.5	Exercises	136
7	Theory for Likelihood	143
7.1	Likelihood properties	143
7.1.1	Invariance	143
7.1.2	The derivative of the log likelihood	144
7.1.3	Using ratios of likelihoods in statistical testing	145
7.1.4	Multiple observations and an asymptotic χ^2 distribution – a simulation example	147
7.1.5	The distribution of the log likelihood ratio in the normal case	149
7.2	The exponential family of distributions	150
7.2.1	Expectation and variance	150
7.2.2	Maximum likelihood estimation within the exponential family	151
7.2.3	Estimation from samples of observations	152
7.2.4	Models for independent and identically distributed observations	152
7.2.5	Existence and uniqueness of the maximum likelihood estimator	153

7.3 Exercises	154
8 Generalized Linear Models - MLE with Covariates	157
8.1 Reducing model complexity through modelling with covariates	157
8.1.1 A normal model via best linear unbiased estimation	157
8.1.2 The normal model via maximum likelihood	159
8.1.3 Example – Poisson regression through the origin	159
8.1.4 Poisson regression through the origin via maximum likelihood	160
8.1.5 Why does the algorithm work without specifying the correct weights at the start?	161
8.2 Iteratively reweighted least-squares for MLE in the exponential family	161
8.2.1 Implementation when the link function is canonical	161
8.2.2 Implementation when the link is not canonical	164
8.3 Statistical inference for generalized linear models	167
8.3.1 The scaled deviance and the chi-squared distribution	168
8.3.2 Analysis of deviance is a new form of ANOVA	169
8.3.3 Assessing the weight of evidence using the bootstrap	170
8.3.4 Inference on the regression coefficients	171
8.4 Residuals	172
8.4.1 Raw residuals	172
8.4.2 Pearson residuals	172
8.4.3 Deviance residuals	172
8.4.4 A bootstrap approach to model-checking	172
8.5 Overdispersion and quasilikelihood	178
8.5.1 Example – cigarette butts data	178
8.5.2 A somewhat more complicated example	178
8.6 Exercises	187
9 Splines and Spline Regression	190
9.1 Basic properties of splines	190
9.1.1 Truncated power functions	190
9.1.2 Spline functions	191
9.1.3 The truncated power basis for splines	191
9.1.4 Exercises	192
9.2 Least-squares splines	193
9.2.1 Example: kiwifruit resistance	193
9.2.2 Using more knots	196
9.2.3 A cubic spline estimate	197
9.2.4 Numerical inaccuracies involving the truncated power basis	197
9.2.5 Exercises	198
9.3 B-splines	199
9.3.1 Example: changing linear spline bases	199
9.3.2 Calculating B-spline values	200
9.3.3 Regression with B-splines	202
9.3.4 Re-visiting the kiwifruit resistance data	202
9.3.5 Exercises	202
9.4 Statistical inference	203
9.4.1 Pointwise confidence intervals	203
9.4.2 Pointwise prediction intervals	203
9.5 Conditioning of the B-spline basis	204
9.5.1 Exercises	205
9.6 Knot selection and regularization	205
9.6.1 Smoothing splines	206
9.6.2 Penalized splines	208

9.7	Multiple predictors	211
9.7.1	Spline regression	212
9.7.2	Additive models	212
9.7.3	A practical overview of generalized additive models	215
9.7.4	GCV and UBRE	215
9.7.5	Plots for gam	216
9.7.6	Exercises	222
9.8	Quantile regression - yet another approach to predictive modelling	222
9.8.1	Required libraries and mapping functions	223
9.8.2	Implementation of the quantile regression function	223
9.9	Appendix - Penalized Spline Code (Eilers and Marx, 1996)	226
10	An Introduction to Mixed-Effects Models	229
10.1	A one-factor random effects model	229
10.1.1	Fitting the model	229
10.1.2	Assessing the model	231
10.1.3	Confidence intervals	232
10.2	Randomized block designs	233
10.3	Modelling growth curves	234
10.3.1	An analysis of covariance model	235
10.3.2	Visualizing the fitted data	237
10.4	Exercises	239
A	Probability Concepts	241
A.1	Basic features of a probability model	241
A.1.1	Properties of density functions	241
A.1.2	Calculation of probabilities using the probability density function	241
A.1.3	Uniform distribution	241
A.1.4	Exponential distribution	242
A.2	Simulation of random variates	242
A.2.1	Simulating and transforming uniform random numbers	242
A.2.2	Built-in simulation functions	244
A.3	Expected value	245
A.3.1	Variance	245
A.3.2	Calculating the mean and variance from a sample	246
A.3.3	Probability calculations	246
A.3.4	Expectation and covariance	247
A.3.5	Correlation	247
A.3.6	Marginal distributions	248
A.3.7	Conditional density functions	249
A.3.8	Independence	250
A.3.9	A Case of Dependence	251
B	Maximum Likelihood Estimation	254
B.1	Which parameter value is the most plausible?	254
B.1.1	The likelihood function	255
B.1.2	Using the fitted model	255
B.1.3	Maximizing the likelihood using calculus	256
B.2	Modelling more than one random variable	256
B.2.1	Likelihood when there are 2 measurements	256
B.2.2	Likelihood with a larger sample	258
B.2.3	The exponential distribution model	259
B.2.4	A Weibull model	259

B.3	Predictive modelling using the likelihood	261
B.3.1	Viewing model parameters as functions of covariates	261
B.3.2	Poisson regression	261
B.3.3	A graphical approach to estimating two parameters	262
B.4	Exercises	267
C	Asymptotic Results for Statistics	269
C.1	Convergence in distribution	269
C.2	Convergence in probability	271
C.2.1	Slutsky's theorem	273
C.3	Mathematical and statistical properties of concave functions	273
C.3.1	Concave functions cannot have more than one maximum	274
C.3.2	Jensen's inequality	274
C.4	Consistency of maximum likelihood estimation	274
C.5	Maximum likelihood estimators are approximately normal in large samples	276
D	Review of Linear Algebra	280
D.1	Vectors	280
D.1.1	Unit Vectors	280
D.1.2	Vector Algebra – Addition	280
D.1.3	Scalar multiplication	281
D.1.4	Inner Product	281
D.1.5	Orthogonal Vectors	282
D.1.6	Vector Norm	282
D.1.7	Projection	282
D.2	Matrices	283
D.2.1	Diagonal Matrices	283
D.2.2	Matrix Transpose	283
D.2.3	Symmetric Matrices	283
D.2.4	Scalar Multiplication	283
D.2.5	Matrix Addition	284
D.2.6	Matrix Multiplication	284
D.2.7	Matrix Identity	285
D.2.8	Matrix Inverse	285
D.2.9	Relations involving Transposes and Inverses	285
D.2.10	Orthogonal Matrices	285
D.2.11	Associativity and Commutativity	285
D.2.12	Idempotent Matrices	286
D.3	Solving Linear Equations	286
D.3.1	Gaussian elimination	286
D.3.2	Trace	286
D.3.3	Determinant	287
D.3.4	Some Determinant Properties	287
D.3.5	Eigenvalues and Eigenvectors	287
D.3.6	Quadratic Forms	287
D.3.7	Positive Definite Matrices	288
D.3.8	Exercises	288
D.4	Linear Spaces	288
D.4.1	Subspaces	288
D.4.2	Span	288
D.4.3	Linear Independence and Dependence	289
D.4.4	Basis	289
D.4.5	Rank of a Matrix	289

D.4.6 Gramm-Schmidt Orthogonalization	289
D.4.7 Exercise: Analysis of an Idempotent Matrix	289
E The QR Decomposition of a Matrix	291
E.1 Householder transformations	291
E.2 Geometric derivation	291
E.3 Algebraic derivation of Householder reflection of an arbitrary vector x	293
E.4 The QR decomposition of an arbitrary matrix	294
E.5 Computational verification of the QR decomposition	294
E.6 The distribution of the sample variance via the QR decomposition	296
F A Note on Nonlinear Least-Squares	299
Index	302

1

R, RStudio and Statistical Modelling

R and RStudio are very useful tools, but becoming acquainted with them requires some effort. A few hours of playing with R code is all that is really required to achieve modest expertise. It is helpful to note that making coding errors from experimentation is harmless when learning a programming language like R. You can learn from your mistakes. Experimentation is the key to learning R.

1.1 R and RStudio

R can be downloaded for free from <http://cloud.r-project.org>. A *binary version* is usually simplest to use and can be installed in Windows and Mac fairly easily. A binary version is available for Windows Vista or above from the web page <http://cloud.r-project.org/bin/windows/base>. The “setup program” is usually a file with a name like R-4.0.3-win.exe. Clicking on this file will start an almost automatic installation of the R system. Clicking “Next” several times is often all that is necessary in order to complete the installation. An R icon will appear on your computer’s desktop upon completion.

RStudio is also recommended. You can download the “Open Source Edition” of “RStudio Desktop” from <http://www.rstudio.com/>, and follow the instructions to install it on your computer.

1.1.1 Packages

There are literally thousands of *packages* such as `graphics`, `ggplot2`, and `MPV`. A package contains functions and data which extend the abilities of R. Every installation of R contains the base packages (e.g. `base`, `stats`, `graphics`) which are automatically loaded when you start R.

To load an additional package, say, called *DAAG*, type

```
library(DAAG)
```

If you get a warning that the package can’t be found, then the package doesn’t exist on your computer, but it can likely be installed. Try

```
install.packages("DAAG")
```

In RStudio, use the Packages menu.

1.1.2 Sources of Additional Information

John Maindonald has written a comprehensive introduction and overview of R which is a very useful reference for scientists. It can be found at

<https://www.researchgate.net/publication/228702931>
_The_R_System-An_Introduction_and_Overview

A handy reference card has been constructed by Jonathan Barron and is available at
<http://www.psych.upenn.edu/~baron/refcard.pdf>

1.2 Types of data

Before taking measurements or observations on some type of phenomenon, they are unknown. A useful way of coping with this lack of knowledge is based on probability. Probability can allow us to quantify our uncertainty about measurements. For example, before throwing a six-sided die, we know that the number of spots that we will observe follows a specific probability distribution, and we refer to that number as a random variable, which we might refer to as Y , and we can say that the probability that $Y = 4$ is $1/6$, and that the probability that $Y = 7$ or $Y = 1.5$ is 0.

The number of spots on the die, Y , is an example of a count, a type of numeric variable. The number of heads, H , in one toss of a coin is another example of a count, but this time with only two possibilities $H = 0$ or $H = 1$. H is an example of a binary random variable or indicator variable. If you think about it, the numbers 0 or 1 are not actually observed, but rather the head or tail. Therefore, the data is, strictly speaking, not of the form of a numeric variable in this case, but rather a categorical variable, with levels Head and Tail. By using the random variable H , we have converted the categorical variable to numeric by a particular type of coding, but note that the coding was arbitrary, since we could have also defined T to be 0 or 1, depending on the number of tails observed.

Other forms of categorical data are possible as well, such as eye-colour, which might include black, brown, blue, and other. In this case, we would do the numeric coding using three binary variables, B_1 , B_2 and B_3 , where B_1 is 1 if the eye-colour is black, and 0, otherwise. $B_2 = 1$ for a brown eye and $B_2 = 0$, otherwise. $B_3 = 1$ for a blue eye and $B_3 = 0$, otherwise. All other eye colours are coded automatically as $B_1 = B_2 = B_3 = 0$.

Another important type of data is continuous data. Continuous variables take on measurements that are not necessarily counting numbers, and are expressed as decimals. Temperature, height, weight and time are often thought of as examples of continuous variables. An important distribution for continuous variables is the normal distribution which gives the familiar symmetric bell-shaped curve. Theoretically, normal random variables can take on any kind of value, positive and negative, so there are situations where this is clearly not appropriate. Time-to-event data, such as the time until someone recovers from a disease, or the time until a lightbulb fails, is a data type which is continuous and where the normal distribution is usually not a good approximation. Sometimes a transformation, such as a log-transformation or a square root transformation yields a new version of the variable which is better approximated by normality.

The heart of statistical modelling lies in determining the relationships between different variables. The goals of statistical analysis are either prediction and explanation. For example, one might want to predict a future value of a random variable, called the response variable, given values of other variables, variously called predictor variables, covariates, or explanatory variables. The latter term is more appropriate when thinking of the modelling problem as one of attempting to explain or understand how the response variable relates or is associated with the other variables.

1.3 Data collection

It is important to recognize the way in which the data have been collected, since this can have a large impact on the way in which the results of any analysis can be used. This short section summarizes three of the standard ways in which you might expect data to have been obtained.

Retrospective study

In this case, data have been collected previously, possibly for another purpose. Not all important predictors may have been measured. Accessing a medical database is a typical situation where this form of data collection would arise. Scraping the web for data is another activity which yields retrospective data.

Observational study

In this case, data are collected on all variables of interest, if possible. However, levels of each factor are not in the control of the scientist. Retrospective data almost always fall into this category, but it is also possible to conduct prospective studies which yield observational data. For example, a traffic engineering study might require observations on the speed of cars passing through a given intersection, together with time of passage, and so on. Because the scientist cannot control any of the variables in such data, it is possible to find associations or dependencies in such data, but it is dangerous to infer a cause-and-effect relationship from such associations.

Designed experiment

Here, the data are collected on all variables of interest. Levels of each factor are under the control of the experimenter. The explanatory variable values are usually fixed by the experimenter. Data from carefully designed experiments can often be used to make inferences regarding cause and effect. Many, though not all, experiments in the physical and engineering sciences fall into this category.

1.4 Graphic and numeric summaries

An important facet of statistics is univariate analysis, whereby the distribution of a given single random variable is studied, often through the use of summary statistics, such as the mean, median, standard deviation and so on, or through the use of graphics, such as the box plot, histogram or dot chart. A bar chart is the most effective way of conveying categorical data; although pie charts are popular, they have been largely discredited as effective data analysis tools, and should be avoided.

We briefly consider two examples here.

River lengths - numeric

The `rivers` data set contains the lengths of 141 important or major North American rivers. This is an example of retrospective data. A quick numeric summary of these data is obtained through

```
summary(rivers)

##      Min.   1st Qu.   Median   Mean   3rd Qu.   Max.
##    135.0    310.0    425.0   591.2   680.0  3710.0
```

A box plot, as shown in the left panel of Figure 1.1, can be constructed using

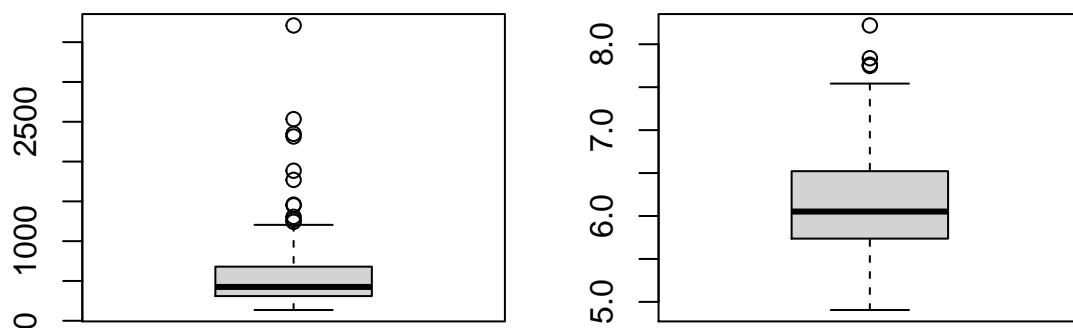


Figure 1.1: Box plot of river lengths, on original and log scales.

```
boxplot(rivers)
```

A histogram could be constructed using `hist` in place of `boxplot`. Both types of plot reveal a distribution which is skewed to the right. A normal distribution is not immediately appropriate due to this fact (which is related to the fact that river length cannot be 0). Taking logs and then computing the box plot gives the graph in the right panel of Figure 1.1. The result is approximately symmetric; the histogram would be hard to distinguish from a normal distribution.

```
boxplot(log(rivers))
```

Eye Colour - Categorical Data

A sample of brown-haired males revealed the following eye colour counts:

	black	brown	blue	green
	53	50	25	15

The table above provides the best form of numeric summary for this kind of data. Converting to percentages is an equivalent alternative – which hides the total number of data points.

The bar chart is constructed using

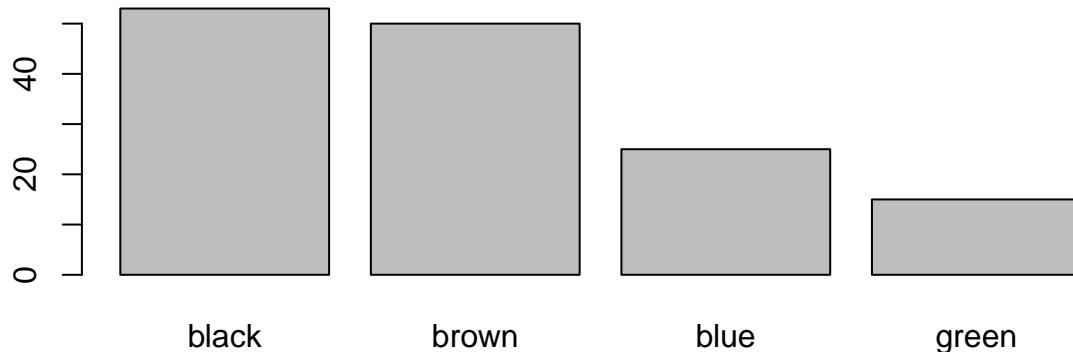


Figure 1.2: Bar chart of brown-haired male eye-colour.

```
barplot(c("black" = 53, "brown" = 50, "blue" = 25, "green" = 15))
```

1.5 Mathematical and statistical modeling

Medical researchers use mouse physiology as a model for human physiology. The hope is the response of a mouse to a specific drug will suggest what response to expect in humans. An engineer constructs a scale model of a proposed bridge. The scale model may be used to help test the effects of high winds on the structure, but may give no information about the rate at which the structure will deteriorate over time. A good model captures enough important features of the object that it represents to be useful for a specific purpose, but it must be expected to fail in other ways.

1.5.1 Mathematical models

Mathematical models, often in the form of equations, have been crucial for describing and quantifying patterns seen in nature and medicine. Mathematical models may be deterministic, or they may incorporate a random component. A deterministic model makes no allowance for statistical or measurement error.

Example

Starting from rest above the earth's surface, a ball falls for a period of time. The formula for the distance traveled is:

$$d = \frac{1}{2}gt^2 \quad (1.1)$$

where t = time in seconds, and $g \approx 9.8\text{ms}^{-2}$ is its acceleration due to Earth's gravity, and d represents the distance traveled in meters.

This is an example of a mechanistic model, since it has a theoretical basis in physics. It is a deterministic model, since it makes predictions of distance, given time traveled, with no allowance for statistical error.

R code to plot the graph of this function (pictured in Figure 1.3) is as follows:

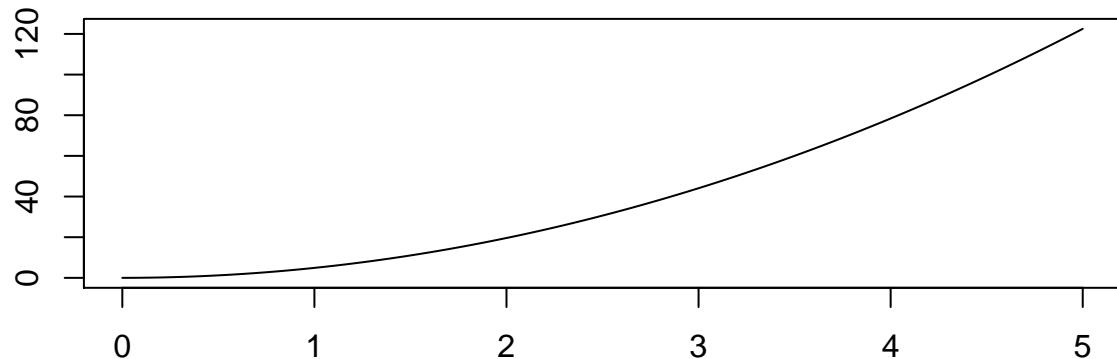


Figure 1.3: Distance travelled by falling ball as a function of time.

```
g <- 9.8 # gravitational constant
curve(g*x^2/2, from = 0, to = 5)
```

Note some of the important aspects of the fall that are not explained by the formula: for example, is the ball going to land in a catcher's glove? Other aspects that are ignored are air resistance, and the fact that the acceleration due to gravity is not exactly constant but dependent upon how far the object is from the earth's surface.

A model is necessarily an incomplete account, but by ignoring what seems irrelevant for a specific purpose, one gets simplicity or tractability – we can easily work with a simple model. Calculating the distance travelled for a given amount of time is simple arithmetic. Taking other factors such as air resistance into account makes the calculations more complicated, and including other factors may even make it impossible to do the required calculations.

1.5.2 The need for statistical models

We saw above that the distance formula is not totally accurate; it neglects the effects of air resistance. Even if it were totally accurate, measurement inaccuracy would introduce small differences between observations and predictions from the formula. A statistical model can be used to model these inaccuracies, given data. When data are used to build a statistical model, one obtains an empirical model. One should always check any relevant theory to ensure that the empirical model is sensible.

Example: a simple-minded analysis

10 measurements were taken on the weight of a lawn roller and the depth of the depression made. The measurements are stored in the `roller` data set:

```
library(DAAG) # contains the roller data set
summary(roller) # provides summary statistics
```

```
##      weight      depression
##  Min.   : 1.900  Min.   : 1.00
##  1st Qu.: 3.675  1st Qu.: 5.00
##  Median : 5.700  Median :15.00
##  Mean    : 6.070  Mean    :14.10
##  3rd Qu.: 7.300  3rd Qu.:22.25
##  Max.   :12.400  Max.   :30.00
```

In an attempt to develop a mathematical model to predict the depression measurements, we might expect depression to be proportional to roller weight:

$$\text{depression} = \beta \times \text{weight} \quad (1.2)$$

Here, the slope β is a constant which could be calculated by dividing depression by weight, if this model explained all variation in depression. Look at the variation exhibited in the values for depression/weight:

```
betahats <- roller$depression/roller$weight # don't ever do this!
betahats

## [1] 1.0526316 0.3225806 1.5151515 1.0416667 3.7735849 3.2786885
## [7] 3.5937500 1.3157895 3.0612245 2.0161290

mean(betahats)

## [1] 2.09712
```

Clearly, there is something in addition to weight that is contributing to the variation seen in the depression measurements.

The scatter plot of depression against weight in Figure 1.4 shows points that lie scattered about a line whose intercept is 0 and whose slope is the average of the estimates of β obtained above. The R code to produce the figure is given below.

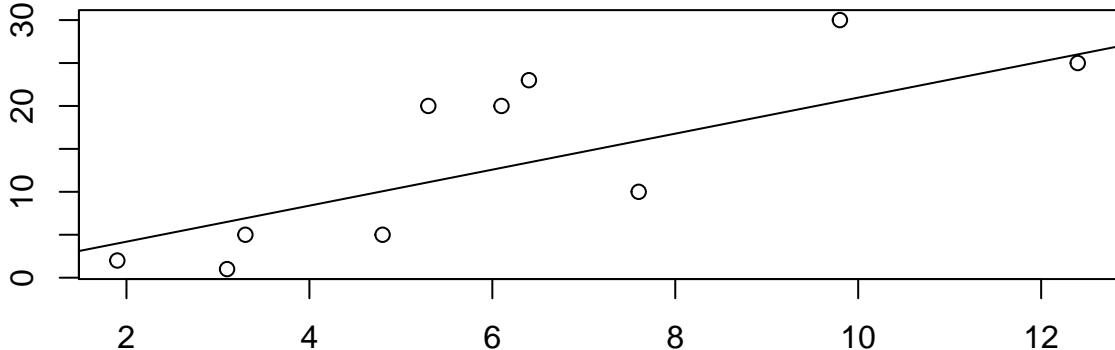


Figure 1.4: Scatter plot of depression measurements as a function of weight for roller data.

```
plot(depression ~ weight, data = roller)
abline(0, 2.097) # overlays line of slope 2.097 and intercept 0
```

We model deviations from the line as random noise or error:

$$\text{depression} = \beta \times \text{weight} + \text{noise} \quad (1.3)$$

The noise or error is different for each different part of the lawn. With no noise or error, all points would lie on a line, and we would be able to

1. predict the amount of depression exactly for a given weight.
2. determine the slope of the line exactly.

The implication of the first point is that there is uncertainty in our predictions, and the noise will become our guide to modeling this uncertainty. The implication of the second point is that there is an additional level of uncertainty in that we cannot know the true value of β . It must be estimated, and the method used above is *not* appropriate; It is better to use least-squares estimation, as described later.

There is, in fact, a third level of uncertainty that is often overlooked: the mathematical model itself (here a linear model) is likely at best only approximate. Determining a better approximation may be difficult if possible at all.

The statistical model above is an example of a model with additive error, since the noise term is added to the mathematical part of the formula. A multiplicative error model is also possible; that case, the noise term would be multiplied by the mathematical part of the formula. Our initial focus will be on additive error models.

1.6 A model building strategy

The following sequence of steps summarizes a typical data analysis:

1. If possible, consult with a subject matter specialist (physicist, biologist, doctor, and so on) to find what factor(s) he or she thinks important.
2. Find proxy (i.e. related) measures for factors that can't be measured directly.
3. Measure these factors and plot the data to examine the relationships visually.
4. Build a mathematical model and estimate the parameters.
5. Check the adequacy of the model. *This step almost invariably requires use of graphical diagnostics.*
6. If assumptions are not satisfied, modify the model and re-check assumptions.
7. Iterate, if necessary.

1.7 Questions that can be addressed by statistical models

Data description

A scientific question often concerns how the measured variables are related. For example, we might want to know the nature of the relation between depression and weight which gave rise to the `roller` data set.

Parameter estimation

Related to the first point, there might be specific information connected with a parameter such as the slope of the regression line that is viewed as important. For example, what is the slope of the line relating depression to weight?

Prediction and estimation

There are two distinct concepts here:

1. We may want to predict the value of a future observation. For example, we may ask what the next depression measurement is, given that we know the amount of weight.
2. What is the expected depression to be made by a roller of a certain weight? This kind of question is concerned with the mean of a population of possible depression measurements that correspond to a given weight level.

The answers to the above two questions are often connected: the predicted value of a future observation is often the same as the estimated value of the mean of future observations, but there can be substantial differences in the uncertainties of the estimate and the prediction.

Control

A typical question here would be: what weight should the roller be in order to produce a depression of a certain amount?

In order to use a regression model for this kind of control purpose, it is vital that there be a cause-and-effect relationship between the predictor and response variables. A simple correlation between two variables does not guarantee that the related regression model can be used for control purposes. For example, crime rate is positively correlated with the size of police force in a sample of cities, but this does not mean that crime rate in a given city can be reduced by decreasing the size of the city's police force.

1.8 Classifying statistical models by data type

The following table may be useful in organizing your thoughts as to the best form of analysis for given types of data. It is important to remember that this table does not exhaustive of the kinds of statistical analyses that could be undertaken. The ones listed are the subject of this book.

response / covariates	continuous	categorical	both
continuous	regression	ANOVA	ANCOVA
categorical	logistic	contingency tables	logistic

Regression refers to both simple and multiple regression (which involves more than 1 covariate). ANOVA refers to the analysis of variance, whereby means of different treatment groups are contrasted, depending on the levels or combination of levels from one or more factors. Factors are essentially another way of referring to categorical variables. Block designs are included in this category, and involve a factor which is not of direct interest to the scientist but which is known or believed to have an affect on the response. By including blocking factors in such analyses, more precision (i.e. less uncertainty) can be gained. ANOVA can also be viewed as a type of regression where the covariates are categorical and are made numeric by binary variable coding which will be described in more detail later on in this book.

ANCOVA is the analysis of covariance, which can be viewed as regression with both continuous and categorical predictors, or as a way of doing ANOVA (i.e. comparing treatment means), accounting for continuous covariates; this is a way of blocking with continuous covariates.

Logistic regression refers to the modelling of the probability distribution of a binary response variable, and the modern view of statistics sees logistic regression as encompassing contingency table analysis. In other words, contingency table analysis can be accomplished by performing logistic regression with categorical covariates.

1.9 Exercises

1. Construct the histogram plot of the data in the `rivers` data set. Describe the shape of the distribution.
2. Construct the histogram of the `rivers` data on the log scale. Describe the shape now.
3. Why do you think a bar chart is more appropriate than a pie chart for visualizing categorical data?¹
4. The default colour scheme for most plots in R is gray-scale and not colour. What are the reasons for avoiding colour when visualizing data?²
5. The `HairEyeColor` data set in R contains sample information on hair and eye colour for males and females. If you type `HairEyeColor`, you will see two contingency tables of hair colour versus eye colour for the two sexes. You can access the blond female eye colour information directly, by typing

```
HairEyeColor[4, , 2]
```

¹Discerning differences in areas and angles is more difficult than discerning differences in heights.

²Reproducing plots on hard copy often uses gray-scale, and more importantly, a surprisingly large proportion of the human population is colour-blind.

Construct a bar chart for the eye colour of blond females by typing

```
barplot(HairEyeColor[, 4, 2])
```

What happens when you omit the '2'?

- Type `help(InsectSprays)` to find information on this data set. Then construct a histogram of the counts of insects in the various experimental plots by using

```
hist(InsectSprays$count)
```

Note the shape of the distribution and re-draw the histogram using the `sqrt` function, that is, by applying a square root transformation to the counts beforehand. How does the distribution shape change?

- Re-do the previous exercise with box plots. Then try

```
boxplot(count ~ spray, data = InsectSprays)
```

and repeat using the square root transformation of the counts. What is the effect of the square root transformation here?³

- In the previous question, what type of data analysis is recommended?⁴
- Type `help(airquality)` to find information on this data set. Then construct a histogram of `airquality$Ozone`. Repeat using a log-transformation. Which is closer to normality?
- If you were to model Ozone level as it relates to Wind, what analysis technique is recommended?⁵ What if you take temperature into account?⁶ What if you add in the Month variable?⁷

³The variability in the different distributions is better approximated by a constant after applying the square root transformation.

⁴ANOVA

⁵Simple regression.

⁶Multiple regression.

⁷There are choices here, but ANCOVA is a simple option.

2

Simple Regression

Relating one continuous variable to another through the use of a linear model with noise is known as simple regression. In this chapter, we will introduce the notation and terminology associated with the techniques, but most of the theoretical explanation will be deferred until we have developed the more general multiple regression framework.

2.1 The model

The measurement of Y (*response*) changes in a linear fashion with a setting of the variable x (*predictor*):

$$Y = \begin{array}{c} \beta_0 + \beta_1 x \\ \text{linear relation} \end{array} + \begin{array}{c} \varepsilon \\ \text{noise} \end{array} \quad (2.1)$$

The *linear relation* is deterministic (non-random), conditional on x . Noise accounts for the variability of the observations about the straight line. If there is no noise, the relation is deterministic. Increased noise yields increased variability or uncertainty. We usually assume that the noise or error has mean 0 and its variance is σ^2 , and the noise is independent of the explanatory variable x .

Simulating regression data to gain intuition

Experiment with this simulation program, where the default slope is 1 and the default intercept is 0:

```
simple.sim <- function(intercept=0, slope=1, x=seq(1, 10), sigma=1) {
  noise <- rnorm(length(x), sd = sigma)
  y <- intercept + slope*x + noise
  title1 <- paste("sigma = ", sigma)
  plot(x, y, pch=16, main=title1)
  abline(intercept, slope, col=4, lwd=2)
}
```

For example, we can examine what happens as the noise standard deviation increases in the four plots displayed in Figure 2.1.

```
par(mfrow=c(2, 2), mar=c(3, 3, 1, 1))
simple.sim(sigma=.01)
simple.sim(sigma=.1)
simple.sim(sigma=1)
simple.sim(sigma=10)
```

Conditional mean and variance

Under the assumptions given above, the following hold:

1. $E[Y|x] = \beta_0 + \beta_1 x = \mu$.

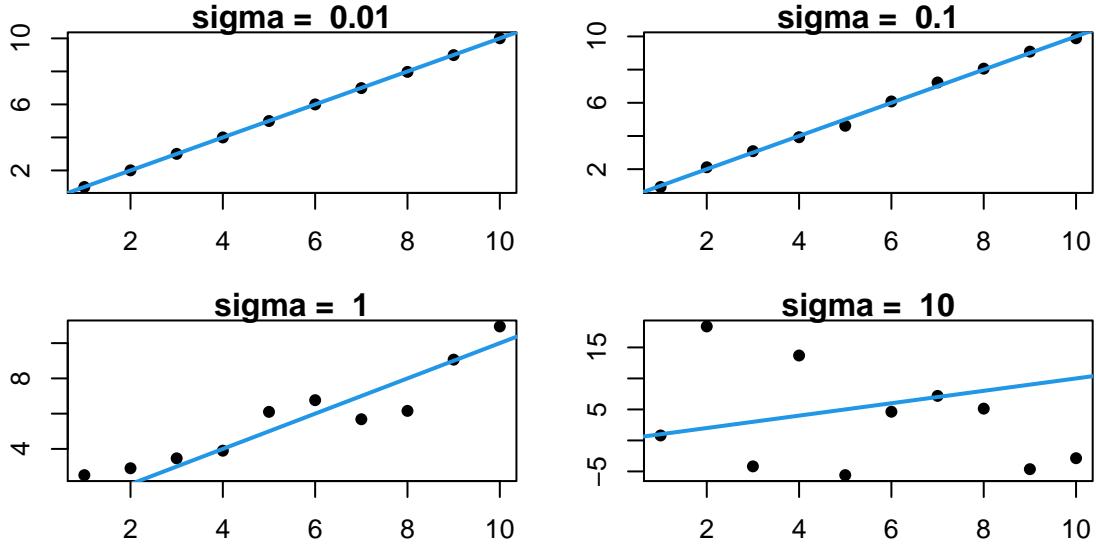


Figure 2.1: Simulated regression data where the slope is 1, the intercept is 0, and the noise standard deviation is .01 (top left), .1, (top right), 1 (lower left), and 10 (lower right).

$$2. \text{Var}(Y|x) = \text{Var}(\beta_0 + \beta_1 x + \varepsilon|x) = \sigma^2.$$

The conditional expectation of \$Y\$ given \$x\$ is often referred to as the regression function. In linear regression, the regression function is linear in \$\beta_0\$ and \$\beta_1\$; we will see later that a linear regression function is not necessarily linear in \$x\$. Note that the mean \$\mu\$ is a linear function of \$x\$, and the variance \$\sigma^2\$ is assumed to be constant for all \$x\$.

Data observations: fixed and random designs

The observed data come in the form of design points: \$x_1, x_2, \dots, x_n\$ and corresponding responses \$Y_1, Y_2, \dots, Y_n\$.

Either

1. the \$x\$'s are fixed values and measured without error (*controlled experiment*); this is a fixed design
OR
2. the analysis is *conditional* on the observed values of \$x\$ (*observational study*); this is a random design.

2.2 Parameter estimation, fitted values and residuals

Least-squares is the standard technique used to obtain estimates of the intercept and slope of the line which passes through the data. Distributional assumptions are *not* required, beyond what was specified in the previous section.

The least-squares problem is to minimize

$$S(\beta_0, \beta_1) = \sum_{i=1}^n (Y_i - \beta_0 - \beta_1 x_i)^2 \quad (2.2)$$

with respect to the parameters or regression coefficients \$\beta_0\$ and \$\beta_1\$: \$\hat{\beta}_0\$ and \$\hat{\beta}_1\$. We do this since we want the fitted line to pass as close as possible to all of the points \$(x_i, Y_i)\$ simultaneously.

The fitted values or predicted values are defined as the fitted regression function evaluated at the design points \$x_i\$:

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i. \quad (2.3)$$

The least-squares objective can also be specified in terms of residuals (that is, observed – fitted response values):

$$\mathbf{e}_i = Y_i - \hat{\mu}_i = Y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i. \quad (2.4)$$

Minimizing $S(\beta_0, \beta_1)$ is equivalent to minimizing $\sum_{i=1}^n (\mathbf{e}_i)^2$.

Using calculus, it is possible to show that the minimizers of

$$S(\beta_0, \beta_1) = \sum_{i=1}^n (Y_i - \beta_0 - \beta_1 x_i)^2 \quad (2.5)$$

are

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{x} \quad (2.6)$$

and

$$\hat{\beta}_1 = \frac{S_{xY}}{S_{xx}} \quad (2.7)$$

where

$$S_{xY} = \sum_{i=1}^n (x_i - \bar{x})(Y_i - \bar{Y}) \quad (2.8)$$

and

$$S_{xx} = \sum_{i=1}^n (x_i - \bar{x})^2 \quad (2.9)$$

Note that S_{xx} is the numerator of the sample variance of the design points, and S_{xY} is the numerator of the covariance between the design points and the responses.

2.3 Unbiased estimation of σ^2 : MSE

The sum of squares for error or SSE is defined as

$$\text{SSE} = \sum_{i=1}^n \mathbf{e}_i^2, \quad (2.10)$$

the sum of the squared residuals. We will see in Section 2.6 that we can obtain unbiased estimates of σ^2 through

$$\frac{\text{SSE}}{n - \# \text{ parameters estimated}}. \quad (2.11)$$

In the case of simple regression, since we have estimated two parameters, β_0 and β_1 , the unbiased estimator for σ^2 is

$$\hat{\sigma}^2 = \frac{\text{SSE}}{n - 2}. \quad (2.12)$$

We often refer to $\hat{\sigma}^2$ as MSE, that is, mean-squared-error.

Some terminology – degrees of freedom

When we have collected n observations, we have n degrees of freedom, which can be thought of n independent pieces of information. 2 degrees of freedom have been required to estimate the parameters. The residuals retain $n - 2$ degrees of freedom which can be used to estimate the noise variance. The mathematical basis for this will be made clear in a later chapter.

2.4 Statistical and geometric consequences of least-squares

The following facts can be derived from the least-squares formulas.

1. $e_i = Y_i - \bar{Y} - \hat{\beta}_1(x_i - \bar{x})$
(follows from intercept formula)
2. $\sum_{i=1}^n e_i = 0$ (follows from 1.)
3. $\sum_{i=1}^n Y_i = \sum_{i=1}^n \hat{\mu}_i$
(follows from 2.)
4. The regression line passes through the centroid (\bar{x}, \bar{Y}) (follows from intercept formula)
5. $\sum x_i e_i = 0$
(set partial derivative of $S(\beta_0, \beta_1)$ wrt β_1 to 0)
6. $\sum \hat{\mu}_i e_i = 0$
(follows from 2. and 5.)

The first result tells us that we could center the data beforehand by subtracting the sample means from both the responses and the predictors and, mathematically, we would obtain the same result as we would with the raw data. Computationally, there are sometimes advantages to this kind of centering; rounding error can be reduced which leads to improved accuracy.

The second result tells us a few things: the residuals are not independent even though the noise is assumed to be; and the inner product of a vector composed of the n residuals and the vector consisting of n 1's is 0, which means that these two vectors are orthogonal.

The third result tells us that the average of the responses is the same as the average of the predicted responses.

The fifth result tells us that the vector composed of the n design points is orthogonal to the vector of the n residuals.

The sixth result tells us that the vector composed of the n fitted values is orthogonal to the vector of the n residuals.

A hand calculator formula for MSE

An additional by-product of the above considerations is an alternative formula for the estimator of the noise variance. If we square both sides of the first result, we have

$$(e_i)^2 = (Y_i - \bar{Y})^2 - 2\hat{\beta}_1(Y_i - \bar{Y})(x_i - \bar{x}) + \hat{\beta}_1^2(x_i - \bar{x})^2. \quad (2.13)$$

Summing over all i , we have

$$\sum_{i=1}^n (e_i)^2 = \sum_{i=1}^n (Y_i - \bar{Y})^2 - 2\hat{\beta}_1 \sum_{i=1}^n (Y_i - \bar{Y})(x_i - \bar{x}) + \hat{\beta}_1^2 \sum_{i=1}^n (x_i - \bar{x})^2 \quad (2.14)$$

or

$$\sum_{i=1}^n (e_i)^2 = S_{YY} - 2\hat{\beta}_1 S_{xY} + \hat{\beta}_1^2 S_{xx}. \quad (2.15)$$

From the least-squares formula for $\hat{\beta}_1$, we can show that

$$\hat{\beta}_1^2 S_{xx} = \hat{\beta}_1 S_{xY}. \quad (2.16)$$

Therefore,

$$SSE = \sum_{i=1}^n (e_i)^2 = S_{YY} - \hat{\beta}_1 S_{xY} \quad (2.17)$$

and, after dividing by $n - 2$, we have

$$MSE = \frac{1}{n-2} (S_{YY} - \hat{\beta}_1 S_{xY}). \quad (2.18)$$

Example – roller data

The least-squares estimates of the slope and intercept parameters for the line relating depression to weight in the *roller* data of Section 1.5.2 can be calculated by hand, as shown below. Normally, such calculations will be carried out on the computer, but it is instructive to work through the arithmetic manually, at least once.

The 10 data points are listed below

```
roller
##      weight  depression
## 1      1.9       2
## 2      3.1       1
## 3      3.3       5
## 4      4.8       5
## 5      5.3      20
## 6      6.1      20
## 7      6.4      23
## 8      7.6      10
## 9      9.8      30
## 10     12.4     25
```

In the calculations that follow, we take $Y = \text{depression}$, and $x = \text{weight}$.

- o $\sum_{i=1}^{10} x_i = 1.9 + 3.1 + \dots + 12.4 = 60.7$
- o $\bar{x} = \frac{60.7}{10} = 6.07$
- o $\sum_{i=1}^{10} Y_i = 2 + 1 + \dots + 25 = 141$
- o $\bar{Y} = \frac{141}{10} = 14.1$
- o $\sum_{i=1}^{10} x_i^2 = 1.9^2 + 3.1^2 + \dots + 12.4^2 = 461$
- o $\sum_{i=1}^{10} Y_i^2 = 4 + 1 + 25 + \dots + 625 = 3009$
- o $\sum_{i=1}^{10} x_i Y_i = (1.9)(2) + \dots + (12.4)(25) = 1103$
- o $S_{xx} = 461 - \frac{(60.7)^2}{10} = 92.6$
- o $S_{xY} = 1103 - \frac{(60.7)(141)}{10} = 247$
- o $S_{YY} = 3009 - \frac{(141)^2}{10} = 1021$
- o $\hat{\beta}_1 = \frac{S_{xY}}{S_{xx}} = \frac{247}{92.6} = 2.67$
- o $\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{x} = 14.1 - 2.67(6.07) = -2.11$
- o $\hat{\sigma}^2 = \frac{1}{n-2}(S_{YY} - \hat{\beta}_1 S_{xY})$
 $= \frac{1}{8}(1021 - 2.67(247)) = 45.2 = \text{MSE}$

To summarize, the fitted regression line relating depression (Y) to weight (x) is

$$\hat{\mu} = -2.11 + 2.67x$$

The error variance is estimated as $\text{MSE} = 45.2$.

2.4.1 R calculations

The least-squares calculations are efficiently carried out by R using the `lm()` function. The output is quite comprehensive, and some of it is potentially misleading if not interpreted carefully.

Example – roller data

```
roller.lm <- lm(depression ~ weight, data = roller)
summary(roller.lm)

##
## Call:
## lm(formula = depression ~ weight, data = roller)
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -8.180 -5.580 -1.346  5.920  8.020 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -2.0871    4.7543  -0.439  0.67227    
## weight       2.6667    0.7002   3.808  0.00518 **  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.735 on 8 degrees of freedom
## Multiple R-squared:  0.6445, Adjusted R-squared:  0.6001 
## F-statistic: 14.5 on 1 and 8 DF,  p-value: 0.005175
```

The intercept and slope estimates appear in the `Estimate` column of the `Coefficients:` list. The noise standard deviation is listed as the `Residual standard error`.

2.4.2 Using extractor functions to obtain partial output

To obtain the coefficients only, use the `coef()` function:

```
coef(roller.lm)

## (Intercept)      weight
## -2.087148     2.666746
```

The square root of the error variance or root-MSE can be obtained as follows:

```
summary(roller.lm)$sigma

## [1] 6.735482
```

This is also referred to as the residual standard error.

Additional functions yield:

- o fitted values

```
predict(roller.lm)
```

```
##      1       2       3       4       5       6       7
## 2.979669 6.179765 6.713114 10.713233 12.046606 14.180002 14.980026
##      8       9      10
## 18.180121 24.046962 30.980502
```

- o residuals

```
resid(roller.lm)

##      1       2       3       4       5       6
## -0.9796695 -5.1797646 -1.7131138 -5.7132327  7.9533944  5.8199976
##      7       8       9      10
## 8.0199738 -8.1801213  5.9530377 -5.9805017
```

- o diagnostic plots `plot(roller.lm)`

(these include a plot of the residuals against the fitted values and a normal QQ-plot of the residuals)

To obtain a plot of the data with the fitted line overlaid, try

```
plot(roller); abline(roller.lm)
```

2.4.3 Moment properties of least-squares estimates

The following results can be obtained directly from earlier results, assuming the linear regression model is correct and assuming independent noise with mean 0 and constant variance σ^2 . We will prove more general versions of these results in a subsequent chapter using a different technique.

1. $E[\hat{\beta}_1] = \beta_1$
2. $E[\hat{\beta}_0] = \beta_0$
3. $\text{Var}(\hat{\beta}_1) = \frac{\sigma^2}{S_{xx}}$
4. $\text{Var}(\hat{\beta}_0) = \sigma^2 \left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}} \right)$

Results 1. and 2. tell us that the slope and intercept estimates are unbiased, meaning that, on average, the least-squares estimates get the right answers. Results 3. and 4. allow us to gauge the uncertainty in the estimates of the slope and intercept via their standard errors.¹ That is, the standard error of the slope estimate is $\sqrt{\text{Var}(\hat{\beta}_1)}$, for example.

2.4.4 Standard error estimators

From Result 3. of the preceding section, it is reasonable to estimate the variance of the slope estimator as $\widehat{\text{Var}}(\hat{\beta}_1) = \widehat{\text{MSE}}_{S_{xx}}$. Then, the standard error (s.e.) of $\hat{\beta}_1$ is estimated by

$$\sqrt{\frac{\widehat{\text{MSE}}}{S_{xx}}} \tag{2.19}$$

Similarly, we have $\widehat{\text{Var}}(\hat{\beta}_0) = \widehat{\text{MSE}} \left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}} \right)$ so the standard error (s.e.) of $\hat{\beta}_0$ can be estimated by

$$\sqrt{\widehat{\text{MSE}} \left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}} \right)} \tag{2.20}$$

¹Recall that the standard error of a statistical estimator is the square root of the variance of the estimator.

Example – roller data

For the `roller` data, we have $\text{MSE} = 45.2$ and $S_{xx} = 92.6$. Therefore, the standard error of $\hat{\beta}_1$ is $\sqrt{45.2/92.6} = .699$.

Since $\bar{x} = 6.07$ and $n = 10$, the standard error of $\hat{\beta}_0$ is $\sqrt{45.2 \left(\frac{1}{10} + \frac{6.07^2}{92.6^2} \right)} = 4.74$

2.4.5 Distributions of $\hat{\beta}_1$ and $\hat{\beta}_0$

We emphasize that the results that we have obtained so far have not depended on a normal distribution assumption. When conducting statistical inference, we will need the response variable to be at least approximately normally distributed.

When Y_i is normally distributed, it follows the $N(\beta_0 + \beta_1 x_i, \sigma^2)$ distribution. We can then show that $\hat{\beta}_1$ is distributed as $N(\beta_1, \frac{\sigma^2}{S_{xx}})$. Therefore,

$$\frac{\hat{\beta}_1 - \beta_1}{\sqrt{\sigma^2/S_{xx}}} \sim N(0, 1). \quad (2.21)$$

It is also possible to show that $\frac{\text{SSE}}{\sigma^2}$ is χ^2_{n-2} (independent of $\hat{\beta}_1$) so that upon replacing σ^2 by its estimate MSE in the above equation, we have

$$\frac{\hat{\beta}_1 - \beta_1}{\sqrt{\text{MSE}/S_{xx}}} \sim t_{n-2}. \quad (2.22)$$

Similarly, $\hat{\beta}_0$ is distributed as $N(\beta_0, \sigma^2 \left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}} \right))$ so that

$$\frac{\hat{\beta}_0 - \beta_0}{\sqrt{\sigma^2 \left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}} \right)}} \sim N(0, 1). \quad (2.23)$$

Replacing σ^2 by its estimate MSE , we have

$$\frac{\hat{\beta}_0 - \beta_0}{\sqrt{\text{MSE} \left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}} \right)}} \sim t_{n-2}. \quad (2.24)$$

Derivation of these results is deferred until after Chapter 3 where useful results on t - and χ^2 random variables are discussed. For now, the reader is expected to apply the results to gain facility with the techniques.

2.5 Inferences about the regression parameters

Two types of inference are conducted on the regression coefficients: hypothesis tests and confidence intervals. We are often interested in confidence intervals for $\beta_0 + \beta_1 x_0$ as well.

2.5.1 Inference for β_1

The usual hypothesis testing set-up applies in a straightforward manner to the slope coefficient. If β_{10} is a given value, we can set up a two-sided test with the null and alternative hypotheses:

$$H_0 : \beta_1 = \beta_{10} \quad \text{vs.} \quad H_1 : \beta_1 \neq \beta_{10} \quad (2.25)$$

Under H_0 ,

$$t_0 = \frac{\hat{\beta}_1 - \beta_{10}}{\sqrt{\text{MSE}/S_{xx}}} \quad (2.26)$$

has a t -distribution on $n - 2$ degrees of freedom.

$$\text{p-value} = 2P(t > |t_0|) \quad (2.27)$$

Example – roller data

Testing significance of regression for `roller` is as follows. The hypotheses are

$$H_0 : \beta_1 = 0 \quad \text{vs.} \quad H_1 : \beta_1 \neq 0.$$

The test statistic is

$$t_0 = \frac{2.67 - 0}{.699} = 3.82.$$

We calculate the p-value as

$$\text{p-value} = 2P(t > 3.82) = 2(1 - P(t < 3.82)) = .00509$$

The p-value can also be calculated with `pt()` in R using the command: `2 * (1-pt(3.82, 8))`

2.5.2 $(1 - \alpha)$ Confidence Intervals

The $1 - \alpha$ confidence interval for the slope coefficient is given by

$$\hat{\beta}_1 \pm t_{n-2,\alpha/2} s.e. \quad (2.28)$$

or

$$\hat{\beta}_1 \pm t_{n-2,\alpha/2} \sqrt{\text{MSE}/S_{xx}} \quad (2.29)$$

Example – roller data

The 95% confidence interval for β_1 is

$$2.67 \pm t_{8,.025}(.699)$$

or

$$2.67 \pm 2.31(.699) = 2.67 \pm 1.61.$$

In R, we can calculate the t quantile with `qt()` using the command `qt(.975, df = 8)`. It is also possible to obtain confidence intervals more automatically using the `confint()` function. For example, 90% confidence intervals for the slope or the intercept can be obtained using

```
confint(roller.lm, level=0.90)

##           5 %      95 %
## (Intercept) -10.927962 6.753667
## weight       1.364611 3.968881
```

2.5.3 Confidence interval for mean response

It is useful to obtain a confidence interval for the regression function evaluated at a new point $x = x_0$:

$$E[Y|x_0] = \beta_0 + \beta_1 x_0. \quad (2.30)$$

In other words, x_0 is a possible value that the predictor could take. $\widehat{E[Y|x_0]} = \widehat{\beta}_0 + \widehat{\beta}_1 x_0$ is a point estimate of the expected response at that value.

To find a $1 - \alpha$ confidence interval for $E[Y|x_0]$, we need the variance of $\widehat{\mu}_0 = E[\widehat{Y}|x_0]$:

$$\text{Var}(\widehat{\mu}_0) = \text{Var}(\widehat{\beta}_0 + \widehat{\beta}_1 x_0) = \sigma^2 \left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right). \quad (2.31)$$

Replacing σ^2 by its estimate, that is MSE, turns the above variance into an estimator for the variance:

$$\widehat{\text{Var}(\hat{\mu}_0)} = \text{MSE} \left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right). \quad (2.32)$$

The confidence interval is then

$$\hat{\mu}_0 \pm t_{n-2,\alpha/2} \sqrt{\text{MSE} \left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right)} \quad (2.33)$$

Example – roller data

A 95% confidence interval for the expected depression when the weight is 5 tonnes is calculated as follows. At $x_0 = 5$, the estimated regression function takes value

$$\hat{\mu}_0 = -2.11 + 2.67(5) = 11.2 \quad (2.34)$$

Using the following additional information

$$n = 10, \bar{x} = 6.07, S_{xx} = 92.6, \text{MSE} = 45.2$$

we can calculate the interval endpoints:

$$\begin{aligned} 11.2 &\pm t_{8,.025} \sqrt{45.2 \left(\frac{1}{10} + \frac{(5 - 6.07)^2}{92.6} \right)} \\ &= 11.2 \pm 2.31 \sqrt{5.08} = 11.2 \pm 5.21 \\ &= (5.99, 16.41) \end{aligned}$$

The R code for this is:

```
predict(roller.lm, newdata=data.frame(weight = 5),
        interval="confidence")

##          fit      lwr      upr
## 1 11.24658 6.039881 16.45328
```

2.5.4 Predicted responses

If a new observation were to be taken at x_0 , we would predict it to be $Y_0 = \beta_0 + \beta_1 x_0 + \varepsilon$ where ε is independent noise.

The estimated response is $\hat{\mu}_0 + \hat{\beta}_1 x_0$ is a point prediction of the response at that value. To find a $1 - \alpha$ prediction interval for Y_0 , we need the variance of $Y_0 - \hat{\mu}_0$:

$$\text{Var}(Y_0 - \hat{\mu}_0) = \text{Var}(\beta_0 + \beta_1 x_0 + \varepsilon - \hat{\beta}_0 - \hat{\beta}_1 x_0) \quad (2.35)$$

$$= \text{Var}(-\hat{\beta}_0 - \hat{\beta}_1 x_0) + \text{Var}(\varepsilon) \quad (2.36)$$

$$= \sigma^2 \left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right) + \sigma^2 \quad (2.37)$$

$$= \sigma^2 \left(1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right) \quad (2.38)$$

Replacing σ^2 by its estimate, that is MSE, turns the above variance into an estimator for the variance:

$$\widehat{\text{Var}(Y_0 - \hat{\mu}_0)} = \text{MSE} \left(1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right) \quad (2.39)$$

The prediction interval is then

$$\hat{\mu}_0 \pm t_{n-2,\alpha/2} \sqrt{\text{MSE} \left(1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right)} \quad (2.40)$$

Example – roller data

A 95% prediction interval for the depression for a single new observation where the weight is 5 tonnes is calculated as follows. The building blocks are:

$$x_0 = 5, \hat{\mu}_0 = -2.11 + 2.67(5) = 11.2$$

$$n = 10, \bar{x} = 6.07, S_{xx} = 92.6, \text{MSE} = 45.2$$

The interval endpoints are then:

$$\begin{aligned} & 11.2 \pm t_{8,025} \sqrt{45.2\left(1 + \frac{1}{10} + \frac{(5 - 6.07)^2}{92.6}\right)} \\ & = 11.2 \pm 2.31\sqrt{50.3} = 11.2 \pm 16.4 = (-5.2, 27.6). \end{aligned}$$

The R code to compute this prediction interval is

```
predict(roller.lm, newdata=data.frame(weight<-5),
        interval="prediction")

##           fit      lwr      upr
## 1 11.24658 -5.134942 27.62811
```

2.6 Analysis of variance: breaking down (analyzing) variation

Regression models can be usefully interpreted using an Analysis of Variance approach. This approach leads to insights into the hypothesis test for the slope coefficient, and it also provides a way of seeing that the MSE is an unbiased estimator for the error variance σ^2 .

The variation in the data (responses) is summarized by

$$S_{YY} = \sum_{i=1}^n (Y_i - \bar{Y})^2 = \text{SST} \quad (\text{Total sum of squares}) \quad (2.41)$$

There are two sources of variation in the responses:

1. variation due to the straight line relationship with the predictor
2. deviation from the line (noise)

Mathematically, this is translated as

$$Y_i - \bar{Y} = \underset{\text{deviation from data center}}{Y_i - \hat{\mu}_i} + \underset{\text{residual}}{\hat{\mu}_i - \bar{Y}} \quad \underset{\text{difference: line and center}}{} \quad (2.42)$$

$$Y_i - \bar{Y} = e_i + \hat{\mu}_i - \bar{Y} \quad (2.43)$$

so, after squaring and summing over $i = 1, 2, \dots, n$, we have

$$S_{YY} = \sum (Y_i - \bar{Y})^2 = \sum (e_i + \hat{\mu}_i - \bar{Y})^2 = \sum e_i^2 + \sum (\hat{\mu}_i - \bar{Y})^2. \quad (2.44)$$

The cross term disappears because

$$\sum e_i \hat{\mu}_i = 0 \quad \text{and} \quad \bar{Y} \sum e_i = 0. \quad (2.45)$$

$$S_{YY} = \text{SSE} + \sum (\hat{\mu}_i - \bar{Y})^2 = \text{SSE} + \text{SSR}. \quad (2.46)$$

The last term, SSR, is the regression sum of squares.

2.6.1 Relation between SSR and β_1

We saw earlier that

$$\text{SSE} = S_{YY} - \hat{\beta}_1 S_{xY}. \quad (2.47)$$

Therefore,

$$\text{SSR} = \hat{\beta}_1 S_{xY} = \hat{\beta}_1^2 S_{xx} \quad (2.48)$$

Note that, for a given set of x 's SSR depends only on $\hat{\beta}_1$.

$$\text{MSR} = \text{SSR}/d.f. = \text{SSR}/1 \quad (2.49)$$

(1 degree of freedom for slope parameter)

2.6.2 Expected Sums of Squares

Development of $E[\text{MSR}]$

We can obtain an expression for the expected value of MSR in terms of σ^2 and β_1 as follows.

$$E[\text{MSR}] = E[\text{SSR}] = E[S_{xx} \hat{\beta}_1^2] \quad (2.50)$$

$$= S_{xx} \left(\text{Var}(\hat{\beta}_1) + (E[\hat{\beta}_1])^2 \right) \quad (2.51)$$

$$= S_{xx} \left(\frac{\sigma^2}{S_{xx}} + \beta_1^2 \right) \quad (2.52)$$

$$= \sigma^2 + \beta_1^2 S_{xx} \quad (2.53)$$

An important observation here is: if $\beta_1 = 0$, then MSR is an unbiased estimator for σ^2 . This also means that MSR can be used to help distinguish between the null hypothesis where $\beta_1 = 0$ and the alternative hypothesis where $\beta_1 \neq 0$. This is because S_{xx} is always nonnegative and so is β_1^2 ; MSR will tend to be around σ^2 under the null hypothesis, and it will tend to be larger than σ^2 under the alternative hypothesis.

Development of $E[\text{MSE}]$

To find the expected value of MSE, we start with

$$E[S_{YY}] = E[\sum (Y_i - \bar{Y})^2] \quad (2.54)$$

$$= E[\sum Y_i^2 - n\bar{Y}^2] = \sum E[Y_i^2] - nE[\bar{Y}^2]. \quad (2.55)$$

Consider the 2 terms on RHS, separately. The first term is

$$E[Y_i^2] = \text{Var}(Y_i) + (E[Y_i])^2 \quad (2.56)$$

$$= \sigma^2 + (\beta_0 + \beta_1 x_i)^2 \quad (2.57)$$

$$\sum E[Y_i^2] = n\sigma^2 + n\beta_0^2 + 2n\beta_0\beta_1\bar{x} + \sum \beta_1^2 x_i^2. \quad (2.58)$$

The second term is

$$E[\bar{Y}^2] = \text{Var}(\bar{Y}) + (E[\bar{Y}])^2 \quad (2.59)$$

$$= \sigma^2/n + (\beta_0 + \beta_1 \bar{x})^2 \quad (2.60)$$

so that

$$nE[\bar{Y}^2] = \sigma^2 + n\beta_0^2 + 2n\beta_0\beta_1\bar{x} + n\beta_1^2\bar{x}^2. \quad (2.61)$$

Therefore,

$$E[S_{YY}] = E[\text{SST}] = (n-1)\sigma^2 + \beta_1^2 \sum (x_i - \bar{x})^2 \quad (2.62)$$

Meanwhile, from (2.46), we can obtain

$$E[\text{SSE}] = E[S_{YY}] - E[\text{SSR}]. \quad (2.63)$$

With (2.62), this becomes

$$E[\text{SSE}] = (n - 1)\sigma^2 + \beta_1^2 \sum (x_i - \bar{x})^2 - (\sigma^2 + \beta_1^2 S_{xx}). \quad (2.64)$$

Simplifying, we obtain

$$E[\text{SSE}] = (n - 2)\sigma^2. \quad (2.65)$$

Therefore,

$$E[\text{MSE}] = E[\text{SSE}/(n - 2)] = \sigma^2. \quad (2.66)$$

This means that MSE is an unbiased estimator for σ^2 . Importantly, this is true whether $\beta_1 = 0$ (null hypothesis) or $\beta_1 \neq 0$ (alternative hypothesis).

2.6.3 Another approach to testing $H_0 : \beta_1 = 0$

Under the null hypothesis, both MSE and MSR are unbiased estimators for σ^2 . Under the alternative, only MSE is an unbiased estimator for σ^2 .

$$E[\text{MSR}] = \sigma^2 + \beta_1^2 S_{xx} > \sigma^2 \quad (2.67)$$

A reasonable test statistic is

$$F_0 = \frac{\text{MSR}}{\text{MSE}} \quad (2.68)$$

A large value of F_0 implies there is evidence against H_0 , and it turns out (we will see this in Chapter 3) that under H_0 ,

$$F_0 \sim F_{1,n-2}. \quad (2.69)$$

It is a fact that $t_\nu^2 = F_{1,\nu}$ so this is really the same test as the one based on the t -distribution discussed earlier in this chapter. In fact, the following argument shows that the square of the test statistic from before is exactly the test statistic suggested above:

$$t_0^2 = \left[\frac{\hat{\beta}_1}{\sqrt{\text{MSE}/S_{xx}}} \right]^2 = \frac{\hat{\beta}_1^2 S_{xx}}{\text{MSE}} = \frac{\text{MSR}}{\text{MSE}} \quad (2.70)$$

Example – roller data

Look for the F -test results for the `roller` example given in the summary output displayed in Section 2.4.1. You will see that the F statistic has been calculated as 14.5 and the degrees of freedom are 1 and 8 (note that there were $n = 10$ observations, and $10 - 2 = 8$). The p-value is small indicating that, according to the F -test, there is strong evidence that the slope coefficient is nonzero.

2.6.4 The ANOVA table

The analysis of variance calculations are conveniently summarized in the following type of table:

Source	df	SS	MS	F
Reg.	1	$\hat{\beta}_1^2 S_{xx}$	$\hat{\beta}_1^2 S_{xx}$	$\frac{\text{MSR}}{\text{MSE}}$
Error	$n - 2$	$S_{YY} - \hat{\beta}_1^2 S_{xx}$	$\text{SSE}/(n - 2)$	
Total	$n - 1$	S_{YY}		

Example – roller data

We can obtain the ANOVA table using the `anova()` function applied to the `lm()` output previously obtained.

```
anova(roller.lm)

## Analysis of Variance Table
##
## Response: depression
##           Df Sum Sq Mean Sq F value    Pr(>F)
## weight      1 657.97 657.97 14.503 0.005175 **
## Residuals   8 362.93 45.37
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that the F -statistic matches the output from the `summary()` output given in Section 2.4.1 as does the p-value. The value of MSE is given as 45.37, the square of the residual standard error.

Recall, also, that the t-statistic for testing $\beta_1 = 0$ had been $3.81 = \sqrt{14.5}$.

2.7 R^2 - coefficient of determination

The coefficient of determination is defined as the fraction of the response variability explained by the regression. It is the square of the correlation between the responses and the design points, and is denoted by the symbol R^2 :

$$R^2 = \frac{\text{SSR}}{S_{YY}}. \quad (2.71)$$

By definition, it is necessary that $0 \leq R^2 \leq 1$. Values near 1 imply that most of the variability is explained by the regression.

Example – roller data

$$\text{SSR} = 658 \text{ and } S_{YY} = 1021 \quad (2.72)$$

so

$$R^2 = \frac{658}{1021} = .644 \quad (2.73)$$

The R code for this is

```
summary(roller.lm)$r.squared

## [1] 0.6444963
```

2.7.1 Relation of R^2 to the correlation coefficient

Suppose that the design points are random and not fixed, so that the simple regression model becomes

$$Y = \beta_0 + \beta_1 \mathbf{X} + \varepsilon \quad (2.74)$$

where ε and \mathbf{X} are independent $N(0, \sigma^2)$ and $N(\mu_1, \sigma_1^2)$ random variables, respectively. The least-squares estimators for the slope and intercept are essentially the same as before:

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{\mathbf{X}} \quad (2.75)$$

$$\hat{\beta}_1 = \frac{S_{\mathbf{XY}}}{S_{\mathbf{XX}}}. \quad (2.76)$$

The Pearson estimator for the correlation coefficient between \mathbf{X} and Y is given by

$$r = \frac{\sum_{i=1}^n (\mathbf{X}_i - \bar{\mathbf{X}})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (\mathbf{X}_i - \bar{\mathbf{X}})^2 \sum_{i=1}^n (Y_i - \bar{Y})^2}} \quad (2.77)$$

This can be rewritten as

$$r = \frac{S_{XY}}{\sqrt{S_{XX}S_{YY}}} \quad (2.78)$$

and further manipulated, using (2.76) to become

$$r = \hat{\beta}_1 \sqrt{\frac{S_{XX}}{S_{YY}}}. \quad (2.79)$$

Squaring both sides of this, we see that

$$r^2 = \hat{\beta}_1^2 \frac{S_{XX}}{S_{YY}} = R^2 \quad (2.80)$$

That is, when the design is random, the coefficient of determination is the square of the correlation coefficient. For fixed designs, the algebra suggests the same thing mathematically, but the quantity r no longer has the same statistical interpretation.

2.7.2 Interpreting the correlation coefficient

Observe from (2.79) that, because $\sqrt{\frac{S_{XX}}{S_{YY}}}$ is nonnegative, the correlation coefficient will have the same sign as $\hat{\beta}_1$. Thus, a negative relation between Y and \mathbf{X} corresponds to a negative correlation, and a positive relation between Y and \mathbf{X} corresponds to a positive correlation. If $\hat{\beta}_1 = 0$, the variables are uncorrelated.

Example – roller data

The `cor()` function in R can be used to find the correlation coefficient:

```
r <- with(roller, cor(depression, weight))
r
## [1] 0.8028052
```

Depression and weight appear to be positively correlated. When we square r , we obtain the value that we saw before as R^2 :

```
r^2
## [1] 0.6444963
```

2.7.3 Another interpretation of R^2

The expected value of R^2 can be approximated as follows:

$$E[R^2] \doteq \frac{E[\text{SSR}]}{E[S_{YY}]} = \frac{\beta_1^2 S_{xx} + \sigma^2}{(n-1)\sigma^2 + \beta_1^2 S_{xx}} \quad (2.81)$$

$$= \frac{\beta_1^2 \frac{S_{xx}}{n-1} + \frac{\sigma^2}{n-1}}{\sigma^2 + \beta_1^2 \frac{S_{xx}}{n-1}} \doteq \frac{\beta_1^2 \frac{S_{xx}}{(n-1)}}{\sigma^2 + \beta_1^2 \frac{S_{xx}}{(n-1)}} \quad (2.82)$$

for large n . This allows us to see the following additional properties of R^2 . In particular, R^2 increases as

1. S_{xx} increases (design points are more spread out)

2. σ^2 decreases

2.7.4 Cautions

1. R^2 does not measure the magnitude of the regression slope.
2. R^2 does not measure the appropriateness of the linear model.
3. A large value of R^2 does not imply that the regression model will be an accurate predictor.

2.8 Hazards of regression

When developing and applying regression models, it is important to be aware of some of the issues that might affect the usefulness and accuracy of the models. These issues include:

- **Extrapolation:** predicting y values outside the range of observed x values. There is no guarantee that a future response would behave in the same linear manner outside the observed range.
e.g. Consider an experiment with a spring. The spring is stretched to several different lengths x (in cm) and the restoring force F (in Newtons) is measured:

```

x      F
3      5.1
4      6.2
5      7.9
6      9.5
> spring.lm <- lm(F~x - 1, data=spring)
> summary(spring.lm)

```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
x	1.5884	0.0232	68.6	6.8e-06

The fitted model relating F to x is

$$\hat{F} = 1.58x \quad (2.83)$$

Can we predict the restoring force for the spring, if it has been extended to a length of 15 cm?

- **High leverage observations:** x values at the extremes of the range have more influence on the slope of the regression than observations near the middle of the range.
- **Outliers** can distort the regression line. Outliers may be incorrectly recorded OR may be an indication that the linear relation or constant variance assumption is incorrect.
- A regression relationship does **not** mean that there is a cause-and-effect relationship. e.g. The following data give the number of lawyers and number of homicides in a given year for a number of towns:

no. lawyers	no. homicides
1	0
2	0
7	2
10	5
12	6
14	6
15	7
18	8

Note that the number of homicides increases with the number of lawyers. Does this mean that in order to reduce the number of homicides, one should reduce the number of lawyers?

- **Possible nonsense relationships.** e.g. It is possible to show that the area of some lakes in Manitoba is related to elevation. Do you think there is a real reason for this? Or is the apparent relation just a result of chance?

2.9 Exercises

- Consider the `modelcars` data frame in the `DAAG` package. The data frame consists of two columns, `distance.traveled` and `starting.point`. A model car was released from various points up a ramp, released, and the distance traveled (in cm) was measured.

Consider the following R code and output:

```
library(DAAG)
mcar.lm <- lm(distance.traveled ~ starting.point, data = modelcars)
summary(mcar.lm)$coefficients

##             Estimate Std. Error    t value    Pr(>|t|)
## (Intercept) 8.083333  1.0779514 7.498792 2.065661e-05
## starting.point 2.013889  0.1312041 15.349288 2.801914e-08
```

- Identify the slope and intercept of the line relating distance traveled to starting point.² Is there strong evidence of a nonzero slope to this line?³ Is the slope positive or negative?⁴
- Write down the two lines of R code which would produce the graph in Figure 2.2.⁵

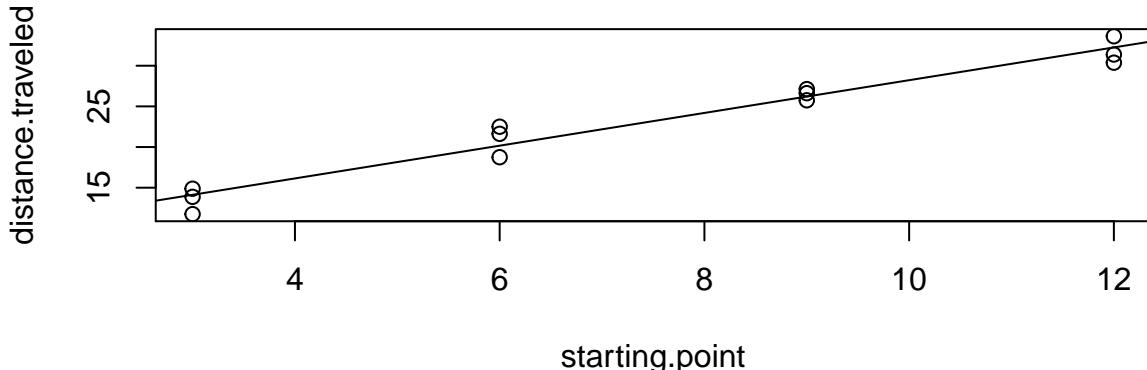


Figure 2.2: Distance travelled by a model car launched from a ramp at various starting points.

- Analyze the `airquality` data to determine the relationship between ozone (y) and wind (x), by finding the slope and intercept of the linear model $y = \beta_0 + \beta_1 x + \varepsilon$. Then overlay the resulting line on a scatterplot of the data.
- Repeat the above analysis using a square root transformation on the Ozone variable. Is this a better way of modelling the data?⁶
- The yield (y , in kg/plot) was measured for various salinity concentrations (x , measured in units of electrical conductivity). 18 measurements were recorded in a file called `tomato.txt` whose contents appear below:

The first column contains the salinity concentration levels, and the second column contains the yield measurements.

- Obtain a scatterplot of the data to determine if a linear relationship is appropriate.

²intercept: 8.083; slope: 2.014

³Yes, the p-value is $2.802e - 08$ which is extremely small.

⁴Positive.

⁵plot(distance.traveled ~ starting.point, data = modelcars); abline(mcar.lm)

⁶Yes, the square root transformation reduces some of the nonlinearity which is apparent when working with raw Ozone data.

x	y
1.60	59.50
1.60	53.30
1.60	56.80
1.60	63.10
1.60	58.70
3.80	55.20
3.80	59.10
3.80	52.80
3.80	54.50
6.00	51.70
6.00	48.80
6.00	53.90
6.00	49.00
10.20	44.60
10.20	48.50
10.20	41.00
10.20	47.30
10.20	46.10

- (b) Calculate the least-squares estimates of the slope and intercept.
- (c) Estimate the error variance.
- (d) Apply the F -test for significance of regression. What is the test statistic value and the p-value? What do you conclude?
- (e) Find a 95% confidence interval for the true slope.
- (f) Find a 95% confidence interval for the mean yield when the salinity is 5.
- (g) Find a 95% prediction interval for a new yield measurement when the salinity is 5.
- (h) Calculate the proportion of variability explained by the regression model.
5. A regression through the origin model is a special case of a simple regression model in which the intercept is set to 0:

$$Y = \beta_1 x + \varepsilon$$

In R, such models can be fit by including `- 1` in the model formula. For the roller example, this would be done using

```
roller.lm <- lm(depression ~ weight - 1, data=roller)
```

Execute this code and use the extractor functions to obtain estimates of the slope and the error variance. Compare the coefficient of determination with the value obtained when the intercept was not included in the model; why do you think the value changed so much?

6. Refer to the previous question. Show that an unbiased estimator for β_1 is given by $\hat{\beta}_1 = \sum_{i=1}^n x_i Y_i / \sum_{i=1}^n x_i^2$, and follow the techniques of Section 2.6 to show that an unbiased estimator for σ^2 in the regression through the origin model is $\hat{\sigma}^2 = \sum_{i=1}^n e_i^2 / (n - 1)$, where e_i is the i th residual. The following steps can be followed to obtain these results.
- (a) Show that $\hat{\beta}_1$ is unbiased by demonstrating that $E[\hat{\beta}_1] = \beta_1$.
- (b) Use the formula for $\hat{\beta}_1$ to show that $\hat{\beta}_1 \sum_{i=1}^n x_i^2 = \sum_{i=1}^n x_i Y_i$ and consequently, $\hat{\beta}_1^2 \sum_{i=1}^n x_i^2 = \hat{\beta}_1 \sum_{i=1}^n x_i Y_i$.
- (c) Show that $\text{Var}(\hat{\beta}_1) = \sigma^2 / \sum_{i=1}^n x_i^2$.

- (d) Show that $\sum_{i=1}^n e_i^2 = \sum_{i=1}^n Y_i^2 - \hat{\beta}_1^2 \sum_{i=1}^n x_i^2$. (You will need to use the result of (b) here.)
 (e) Show that $E[\sum_{i=1}^n Y_i^2] = n\sigma^2 + \beta_1^2 \sum_{i=1}^n x_i^2$.
 (f) Use the results of (a) and (c) to show that $E[\hat{\beta}_1^2] = \sigma^2 / \sum_{i=1}^n x_i^2 + \beta_1^2$.
 (g) Use the results (d), (e) and (f) to show that $E[\sum_{i=1}^n e_i^2] = (n-1)\sigma^2$, and to conclude that $\hat{\sigma}^2$ is an unbiased estimator for σ^2 .

7. Consider the model

$$y_j = \beta_1 x_j + B_j x_j + \eta_j, \quad j = 1, 2, \dots, n \quad (2.84)$$

where B_1, B_2, \dots, B_n are independent normally distributed random variables with mean 0 and variance σ_B^2 , and $\eta_1, \eta_2, \dots, \eta_n$ are independent normal random variables with mean 0 and variance σ^2 . Assume that the η 's are independent of the B 's.

- (a) Rewrite the model (2.84) as a regression through the origin model by setting $\varepsilon_j = \eta_j + B_j x_j$. Of all of the modelling assumptions made in this chapter, which one is most clearly violated? (Hint: what is the relationship between ε_j and x_j ?)
 (b) The least-squares estimator for β_1 is $\hat{\beta}_1 = \sum_{j=1}^n x_j y_j / \sum_{j=1}^n x_j^2$. Show that this estimator is unbiased and has variance $\sigma^2 / \sum_{j=1}^n x_j^2 + \sigma_B^2 \sum_{j=1}^n x_j^4 / (\sum_{j=1}^n x_j^2)^2$.
 (c) Define the j th residual as $e_j = y_j - \hat{\beta}_1 x_j$, and show that $\sum_{j=1}^n e_j^2 = \sum_{j=1}^n y_j^2 - \hat{\beta}_1^2 \sum_{j=1}^n x_j^2$. (Hint: use a similar approach to 2.6 (d) here.)
 (d) Show that $E[y_j^2] = \sigma^2 + \sigma_B^2 x_j^2 + \beta_1^2 x_j^2$.
 (e) Show that $E[\hat{\beta}_1^2] = \sigma^2 / \sum_{j=1}^n x_j^2 + \sigma_B^2 \sum_{j=1}^n x_j^4 / (\sum_{j=1}^n x_j^2)^2 + \beta_1^2$.
 (f) Deduce that

$$E[\text{SSE}] = E \left[\sum_{j=1}^n e_j^2 \right] = (n-1)\sigma^2 + \sigma_B^2 \left(\sum_{j=1}^n x_j^2 - \frac{\sum_{j=1}^n x_j^4}{\sum_{j=1}^n x_j^2} \right).$$

- (g) The variance of $\hat{\beta}_1$ for the true regression through the origin model is $\sigma^2 / \sum_{j=1}^n x_j^2$, so an estimate of this is $\text{MSE} / \sum_{j=1}^n x_j^2$, where $\text{MSE} = \text{SSE}/(n-1)$. Show that the expected value of this estimator is

$$\frac{\sigma^2}{\sum_{j=1}^n x_j^2} + \frac{\sigma_B^2}{n-1} \left(1 - \frac{\sum_{j=1}^n x_j^4}{(\sum_{j=1}^n x_j^2)^2} \right)$$

under the assumptions of the modified regression through the origin model.

- (h) Given what you observed in part (b) and in the previous part, would you recommend using $\text{MSE} / \sum_{j=1}^n x_j^2$ as an unbiased estimator of the variance of $\hat{\beta}_1$? Put another way, would you recommend the use of $\sqrt{\text{MSE} / \sum_{j=1}^n x_j^2}$ when estimating the standard error of $\hat{\beta}_1$?
 (i) The following code runs a simulation of the model considered above where the true value of β_1 is 0.

```
set.seed(3992); x <- 1:10
B <- rnorm(10, sd=2)
y <- B*x + rnorm(10, sd=.1) + 0*x
summary(lm(y ~ x - 1))

## 
## Call:
## lm(formula = y ~ x - 1)
## 
## Residuals:
```

```

##      Min     1Q Median     3Q    Max
## -15.101 -7.701 -3.168  2.815 22.701
##
## Coefficients:
##   Estimate Std. Error t value Pr(>|t|)
## x -1.5347     0.6155 -2.493  0.0342 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.08 on 9 degrees of freedom
## Multiple R-squared:  0.4085, Adjusted R-squared:  0.3428
## F-statistic: 6.217 on 1 and 9 DF,  p-value: 0.03424

sqrt(.01/sum(x^2) + 4*sum(x^4)/(sum(x^2)^2)) # actual SE
## [1] 0.8268388

sqrt(.01/sum(x^2) + 4*(1-sum(x^4)/(sum(x^2)^2))/9) # incorrect SE
## [1] 0.6070509

```

If you only considered the summary output, would you say there is evidence that $\beta_1 \neq 0$? What would the conclusion have been if the correct standard error could have been calculated instead?

- (j) If we knew β_1 , we could estimate the variance of $\hat{\beta}_1$ in an unbiased manner using the formula

$$\frac{\sum_{j=1}^n x_j^2(y_j - \beta_1 x_j)^2}{(\sum_{j=1}^n x_j^2)^2}$$

and the standard error could be obtained by taking the square root. Since we do not know β_1 , we can get an asymptotically unbiased estimator⁷ by replacing β_1 by its least-squares estimator. The following code evaluates this estimate of the standard error for the data simulated above.

```

bhat <- coef(lm(y ~ x - 1))[1]
sqrt(sum(x^2 * (y-bhat*x)^2) / (sum(x^2)^2))

## [1] 0.6755587

```

The estimate is still below the true value, but we would not be quite as inclined to conclude that the true slope estimate is nonzero as when we used the incorrect standard error.

It is possible to reduce the bias somewhat by dividing the variance estimate by $1 - \sum_{j=1}^n x_j^4 / (\sum_{j=1}^n x_j^2)^2$:

```

sqrt(sum(x^2 * (y-bhat*x)^2) / (sum(x^2)^2) / (1 - sum(x^4) / (sum(x^2)^2)))
## [1] 0.7419284

```

Calculate the t -statistic by hand, using each of these new standard error estimates, and compute p-values for the test of $\beta_1 = 0$ versus $\beta_1 \neq 0$. Compare with the result of question (i).

⁷The bias approaches 0 as the sample size grows.

3

Normal Distribution Theory

Although statistical models can be constructed without relying on the normal distribution assumption, statistical inference can be undertaken in a very straightforward way when normality holds. This chapter considers normality concepts, both theoretically and numerically. Much of this chapter should seem like a review, but the perspective is likely different from what would have been seen in a mathematical statistics course. As will be seen throughout this book, simulation is a useful tool to aid in understanding various statistical concepts. The reader is encouraged to experiment with the simulation code on his or her own and to modify it in order to see what works and what might not make sense.

3.1 Normality and the QQ-plot

Theoretically, when we speak of a sample of standard normal random variables, we mean that the marginal probability distribution of each of the random variables is normal with mean 0 and standard deviation 1. The probability density function is the familiar bell-shaped curve shown in Figure 3.1 for the domain $[-3.5, 3.5]$. The R code required to draw the curve is as follows.

```
par(mar = c(2, 4, 1, 1)) # this removes excess space from the plot margins
curve(dnorm(x), from=-3.5, to=3.5, ylab = "f(x)")
```

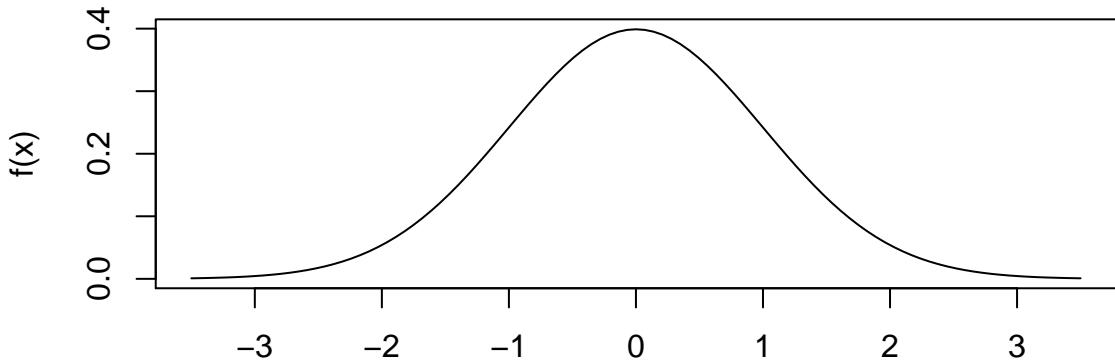


Figure 3.1: The standard normal density curve.

We can simulate data from this distribution, and draw a histogram as in Figure 3.2.

```
n <- 500; Z <- rnorm(n) # n simulated standard normals
par(mar = c(2, 4, 1, 1))
hist(Z, main="", xlab = "") # The use of 'main = " "' removes the graph title
```

A random sample of size $n = 500$ has been simulated in this case. Note the similarities and differences between the histogram and the density curve. In fact, the histogram is a crude *density estimator*.

Although the histogram is easier to interpret, a better way of checking graphically whether a data set is normally distributed is via a normal QQ-plot or *quantile-quantile*-plot. The normal QQ-plot graphs the theoretical normal percentiles against the sample percentiles. Departures from a straight line are indications of non-normality. Code to draw a QQ-plot for the simulated sample obtained earlier is as follows.

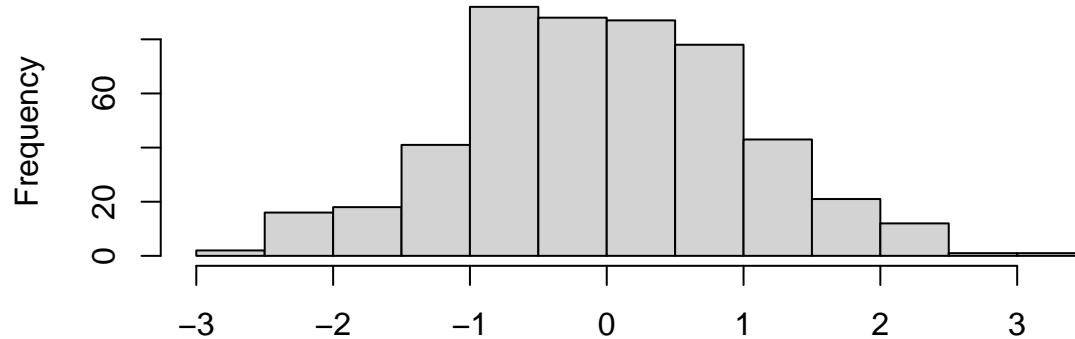


Figure 3.2: A histogram of a standard normal sample.

```
par(mar = c(2, 4, 1, 1))
qqnorm(Z, main = " ")
qqline(Z) # this overlays a reference line
```

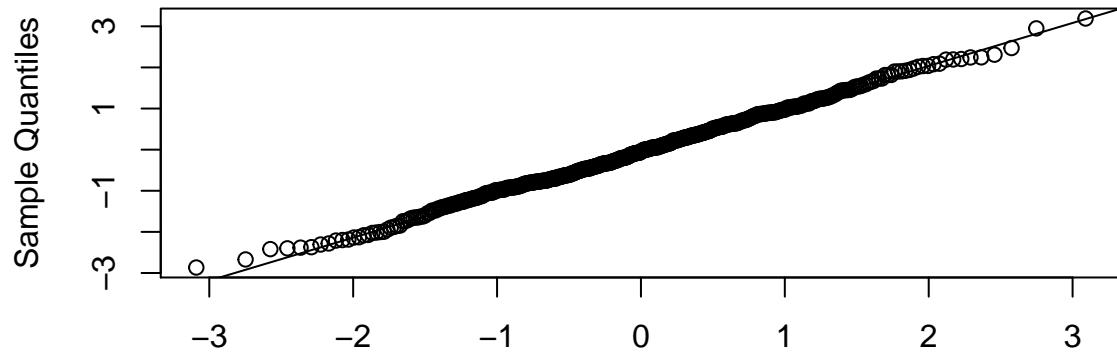


Figure 3.3: A normal QQ-plot for a standard normal sample.

Figure 3.3 displays the normal QQ-plot for our simulated data. Although not a perfect straight line, it would be judged to be adequate for practical purposes.

For an example where the data are not normally distributed, consider a case where data have been simulated from an exponential distribution. This time 250 independent observations have been simulated. Side-by-side plots of the histogram and normal QQ-plot are displayed in Figure 3.4. They were obtained using the `mfrow()` argument to the `par()` function, which controls graphical parameters in R, as in the following code.

```
E <- rexp(250)
par(mfrow=c(1,2), mar = c(2, 4, 1, 1))
hist(E, main = " ")
qqnorm(E, main = " "); qqline(E)
```

The histogram in 3.4 clearly shows the skewness in the exponential data, while the QQ-plot shows the departure from a straight line.

Consider one more example. 150 independent observations are simulated from a t -distribution on 3 degrees of freedom. Figure 3.5 shows the histogram and normal QQ-plot.

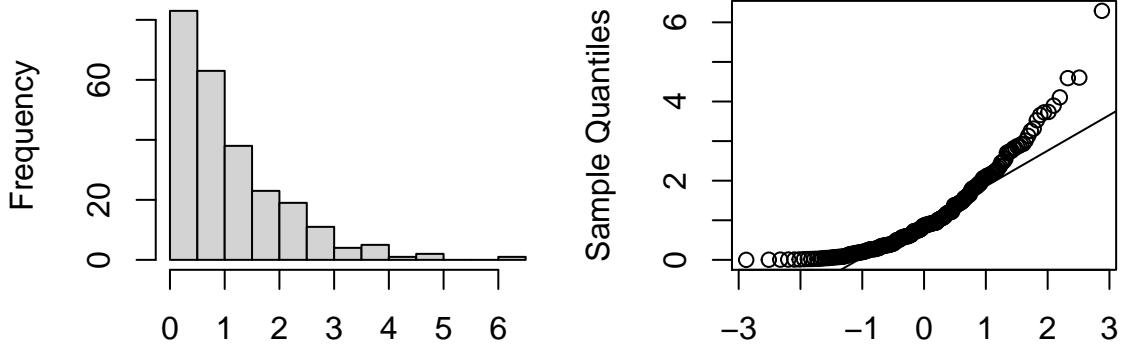


Figure 3.4: A histogram and normal QQ-plot for a sample of independent unit exponential random variables.

```
T <- rt(150, df=3)
par(mfrow=c(1,2), mar = c(2, 4, 1, 1))
hist(T, main = " ")
qqnorm(T, main = " "); qqline(T)
```

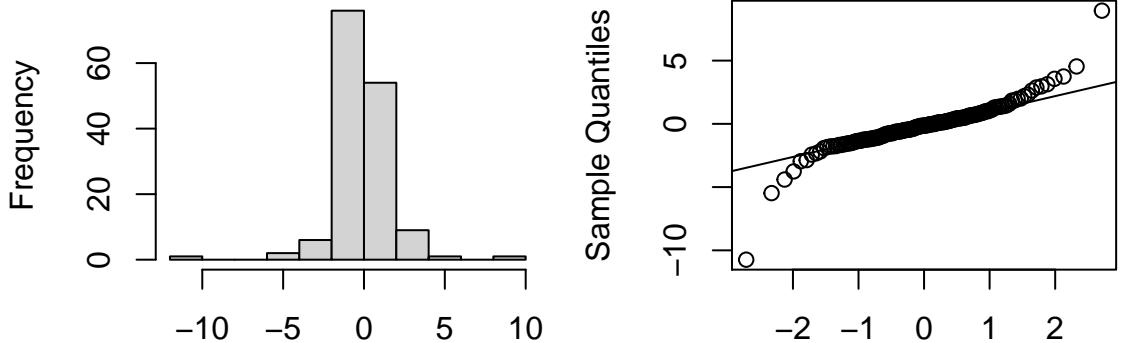


Figure 3.5: A histogram and normal QQ-plot for a sample of independent t random variables on 3 degrees of freedom

This time, the data are not skewed, so the histogram in Figure 3.5 may be misleading. The QQ-plot is not. There is clear evidence of non-normality.

3.2 Random vectors and their expectations

If Z_1, Z_2, \dots, Z_n is a collection of random variables, we can define a random vector Z as

$$\mathbf{z}^\top = [Z_1 \ Z_2 \ \cdots \ Z_n] \quad (3.1)$$

where \top denotes the matrix transpose operator.¹ For a review of the essential concepts needed from linear algebra, please see Appendix D.

The expected value of \mathbf{z} is defined as the vector of expected values of \mathbf{z} :

$$E[\mathbf{z}^\top] = [E[Z_1] \ E[Z_2] \ \cdots \ E[Z_n]]. \quad (3.2)$$

¹Most of the time, we will assume untransposed vectors to be column vectors.

For example, if the elements of \mathbf{z} are n standard normal random variables, then

$$E[\mathbf{z}] = \mathbf{0}, \text{ the } n\text{-vector consisting only of 0's.} \quad (3.3)$$

3.2.1 The variance-covariance matrix

A useful summary of the variability in a random vector is provided by the variance-covariance matrix. It also provides a view of a special type of dependence, i.e. linear dependence among pairs of elements in the random vector.

Note first that if \mathbf{z} is a random n -vector, then its outer product is given by $\mathbf{z}\mathbf{z}^\top$. Recall that the variance of a random variable is defined as the expected value of the quadratic expression $(X - E[X])^2$. The appropriate generalization to the vector case is through the outer product:

$$\text{Var}(\mathbf{z}) = E[(\mathbf{z} - E[\mathbf{z}])(\mathbf{z} - E[\mathbf{z}])^\top]. \quad (3.4)$$

The result is an $n \times n$ variance-covariance matrix whose (i, j) element is

$$E[(Z_i - E[Z_i])(Z_j - E[Z_j])]. \quad (3.5)$$

This is $\text{Var}(Z_i)$ when $i = j$, and this is the covariance between Z_i and Z_j , $\text{Cov}(Z_i, Z_j)$, otherwise. Note that $\text{Cov}(Z_i, Z_j) = \text{Cov}(Z_j, Z_i)$. Thus, $\text{Var}(\mathbf{z})$ is a symmetric $n \times n$ matrix whose diagonal elements are the variances of the elements of \mathbf{z} . The off-diagonal elements consist of the pairwise covariances.

3.2.2 Linear combinations of random variables

If a random vector is multiplied by a scalar constant, its expectation is also multiplied by that constant:

$$E[a\mathbf{z}] = aE[\mathbf{z}]. \quad (3.6)$$

If \mathbf{A} is an $m \times n$ matrix, then

$$E[\mathbf{A}\mathbf{z}] = \mathbf{A}E[\mathbf{z}]. \quad (3.7)$$

The number of columns of \mathbf{A} must match the number of elements in \mathbf{z} .

When the matrix \mathbf{A} consists of only 1 row, $\mathbf{A}\mathbf{z}$ reduces to an inner product of \mathbf{A} and \mathbf{z} . That is, if \mathbf{a} is an n -vector, then

$$\mathbf{a}^\top \mathbf{z} = \sum_{j=1}^n a_j Z_j. \quad (3.8)$$

According to the linearity of expectation expressed by (3.7), we can say that

$$E[\mathbf{a}^\top \mathbf{z}] = \mathbf{a}^\top E[\mathbf{z}] = \sum_{j=1}^n a_j E[Z_j]. \quad (3.9)$$

For example, if $a_j = 1/n$ for all values of j , then

$$\mathbf{a}^\top \mathbf{z} = \bar{Z} \quad (3.10)$$

and

$$E[\bar{Z}] = \sum_{j=1}^n E[Z_j]/n. \quad (3.11)$$

When the mean of Z_j is a constant, say μ , this implies that

$$E[\bar{Z}] = \mu. \quad (3.12)$$

Note that the assumptions we have made are very minimal. In particular, there has been no mention made of independence or normality here.

3.2.3 The variance-covariance matrix of $\mathbf{A}\mathbf{z}$

An important formula to remember is

$$\text{Var}(\mathbf{A}\mathbf{z}) = \mathbf{A}\text{Var}(\mathbf{z})\mathbf{A}^\top. \quad (3.13)$$

This is the appropriate generalization of the rule for the scalar case

$$\text{Var}(aX) = a^2\text{Var}(X). \quad (3.14)$$

The formula (3.13) can be obtained from the definition of the variance-covariance matrix:

$$\text{Var}(\mathbf{A}\mathbf{z}) = E[(\mathbf{A}\mathbf{z} - E[\mathbf{A}\mathbf{z}])(\mathbf{A}\mathbf{z} - E[\mathbf{A}\mathbf{z}])^\top] = E[\mathbf{A}(\mathbf{z} - E[\mathbf{z}])(\mathbf{z} - E[\mathbf{z}])^\top\mathbf{A}^\top] \quad (3.15)$$

$$= \mathbf{A}E[(\mathbf{z} - E[\mathbf{z}])(\mathbf{z} - E[\mathbf{z}])^\top]\mathbf{A}^\top \quad (3.16)$$

This yields (3.13) upon application of the definition of $\text{Var}(Z)$. A key step in this derivation depended on the matrix algebra fact: $(\mathbf{AB})^\top = \mathbf{B}^\top\mathbf{A}^\top$.

The mathematical benefit of a constant variance

An important special case occurs when the elements of \mathbf{z} have a common variance σ^2 and the pairwise covariances are 0. Then the variance-covariance matrix of \mathbf{z} is $\sigma^2\mathbf{I}$, and

$$\text{Var}(\mathbf{A}\mathbf{z}) = \mathbf{A}\sigma^2\mathbf{I}\mathbf{A}^\top = \sigma^2\mathbf{A}\mathbf{A}^\top. \quad (3.17)$$

A single linear combination of random variables

Another important case arises when the matrix \mathbf{A} has only one row consisting of elements a_1, a_2, \dots, a_n . In that case, $\mathbf{A}\mathbf{z} = \sum_{j=1}^n a_j Z_j$, and we see that

$$\text{Var}\left(\sum_{j=1}^n a_j Z_j\right) = \sum_{j=1}^n \sum_{i=1}^n a_i a_j \text{Cov}(Z_i, Z_j) \quad (3.18)$$

where we have written $\text{Cov}(Z_i, Z_i)$ in place of $\text{Var}(Z_i)$. When the pairwise covariances of the elements of Z are 0, we can use the result at (3.17) to obtain

$$\text{Var}\left(\sum_{j=1}^n a_j Z_j\right) = \sigma^2 \mathbf{a}^\top \mathbf{a} = \sigma^2 \sum_{j=1}^n a_j^2. \quad (3.19)$$

When $a_j = 1/n$, this leads to a familiar result:

$$\text{Var}(\bar{Z}) = \sigma^2/n. \quad (3.20)$$

Again, note what has been assumed: pairwise covariances of the elements of \mathbf{z} must be 0. In other words, the elements of \mathbf{z} must be uncorrelated with each other.

Orthogonal transformations

Finally, when \mathbf{A} is an orthogonal matrix, it has the property that $\mathbf{A}\mathbf{A}^\top = \mathbf{I}$, where \mathbf{I} is the identity matrix. In that case, equation (3.17) tells us that the variance-covariance matrix of $\mathbf{A}\mathbf{z}$ is simply the identity matrix multiplied by σ^2 . In other words, the components of $\mathbf{A}\mathbf{z}$ are uncorrelated, and they have the same constant variance as the components of \mathbf{z} .

3.2.4 Independence and expectation

Theoretically, random variables are independent if their joint distribution can be factored. That is, Z_1 and Z_2 are independent, if their joint density is

$$f_{Z_1}(z_1)f_{Z_2}(z_2) \quad (3.21)$$

where the two functions are the respective marginal densities of Z_1 and Z_2 . For a sample of n independent random variables, the joint density would be

$$f_{Z_1}(z_1)f_{Z_2}(z_2) \cdots f_{Z_n}(z_n). \quad (3.22)$$

Equivalently, independence of Z_1, Z_2, \dots, Z_n can be seen through expectation: Z_1, Z_2, \dots, Z_n are independent if and only if

$$E[g(Z_1)g(Z_2) \cdots g(Z_n)] = E[g(Z_1)]E[g(Z_2)] \cdots E[g(Z_n)] \quad (3.23)$$

for all continuous functions $g(x)$.

This concept allows us to determine that uncorrelated normal random variables are independent.

3.2.5 Characterizing normally distributed random vectors

The characteristic function of a random vector \mathbf{z} is defined as

$$\phi(\mathbf{t}) = E[e^{i\mathbf{t}^\top \mathbf{z}}]. \quad (3.24)$$

Here, \mathbf{t} is a vector having the same number of elements as \mathbf{z} , and i denotes the imaginary number $\sqrt{-1}$. The characteristic function (or Fourier transform) is a unique characterization of its distribution. In other words, for a given characteristic function, there is one and only one probability distribution. For a normal random vector with mean vector μ and variance-covariance matrix \mathbf{V} , the characteristic function is given by

$$\phi(\mathbf{t}) = e^{i\mathbf{t}^\top \mu - \mathbf{t}^\top \mathbf{V}\mathbf{t}/2}. \quad (3.25)$$

Standard normal random variables and vectors

A standard normal random variable Z has scalar mean 0 and variance 1, so its characteristic function is

$$\phi(t) = e^{-t^2/2}. \quad (3.26)$$

Now, suppose Z is a vector of n independent standard normal random variables. The characteristic function of Z is then

$$E[e^{i\mathbf{t}^\top \mathbf{z}}] = E[e^{i\sum_{j=1}^n t_j Z_j}] = E\left[\prod_{j=1}^n e^{it_j Z_j}\right].$$

Because expected values of products of independent random variables are equal to the product of the expected values, i.e. (3.23), the final expression above becomes $\prod_{j=1}^n E[e^{it_j Z_j}]$. Applying (3.26) to each of the factors in this last expression, we conclude that

$$E[e^{i\mathbf{t}^\top \mathbf{z}}] = \prod_{j=1}^n e^{-t_j^2/2} = e^{-\sum_{j=1}^n t_j^2/2} = e^{-\mathbf{t}^\top \mathbf{t}/2}. \quad (3.27)$$

This means that a vector of independent standard normal random variables is a normal random vector with mean vector 0 and variance-covariance matrix I , the $n \times n$ identity matrix.

Uncorrelated normal random variables are independent

The characteristic function of a vector of independent random variables, Z , can be factored as follows:

$$E[e^{i\mathbf{t}^\top \mathbf{z}}] = E[e^{\sum_{j=1}^n it_j Z_j}] = E[e^{it_1 Z_1}] \cdots E[e^{it_n Z_n}]. \quad (3.28)$$

Now, suppose \mathbf{x} is a vector of uncorrelated normal variables with mean vector μ and diagonal variance-covariance matrix \mathbf{V} . Its characteristic function is

$$E[e^{it^\top \mathbf{x}}] = e^{it^\top \mu - \mathbf{t}^\top \mathbf{V} \mathbf{t}/2} = e^{i \sum_{j=1}^n t_j \mu_j - \sum_{j=1}^n t_j^2 V_{jj}/2}. \quad (3.29)$$

The last expression can be rewritten as

$$e^{it_1 \mu_1 - t_1^2 V_{11}} \cdots e^{it_n \mu_n - t_n^2 V_{nn}} \quad (3.30)$$

which is clearly the product of characteristic functions of normal random variables having mean μ_1, \dots, μ_n and variances V_{11}, \dots, V_{nn} . These are the marginal distributions of the components of \mathbf{x} . Thus, according to equation (3.28), the components of \mathbf{x} must be independent.

A model for nonstandard normal random variables

A nonstandard normal random variable with scalar mean μ and variance σ^2 has characteristic function

$$\phi(t) = e^{it\mu - t^2\sigma^2/2}. \quad (3.31)$$

Note that this case could be obtained from the model

$$X = \mu + \sigma Z \quad (3.32)$$

where Z is standard normal:

$$E[e^{itX}] = E[e^{it\mu + it\sigma Z}] = e^{it\mu} E[e^{i(t\sigma)Z}] \quad (3.33)$$

$$= e^{it\mu} e^{-(t\sigma)^2/2} = e^{it\mu - t^2\sigma^2/2}. \quad (3.34)$$

Thus, if Z is standard normal, and $X = \mu + \sigma Z$, then X is normally distributed with mean μ and standard deviation σ .

3.2.6 The distribution of \mathbf{Ax}

When \mathbf{x} is a vector of normal random variables with mean μ (a vector) and variance-covariance matrix \mathbf{V} , we can show that \mathbf{Ax} is also a normal random vector, when \mathbf{A} is an $m \times n$ matrix. We do this by obtaining the characteristic function of \mathbf{Ax} and identifying it coming from a particular normal distribution. The characteristic function we seek is

$$E[e^{it^\top (\mathbf{Ax})}] \quad (3.35)$$

where \mathbf{t} is a vector of length m (since \mathbf{Ax} is an m -vector here). This can be rewritten as

$$E[e^{i(t^\top \mathbf{A})\mathbf{x}}] = E[e^{i(\mathbf{A}^\top \mathbf{t})^\top \mathbf{x}}] \quad (3.36)$$

where we now note that $\mathbf{A}^\top \mathbf{t}$ is an n -vector. Since \mathbf{x} is a normal vector with known mean vector and variance-covariance matrix, we can write its characteristic function as

$$E[e^{is^\top \mathbf{x}}] = e^{is^\top \mu - s^\top \mathbf{V} s/2}. \quad (3.37)$$

We are using the n -vector s here to avoid confusion. In fact, we can take $s = \mathbf{A}^\top \mathbf{t}$, and then it follows that the characteristic function of \mathbf{Ax} must be

$$E[e^{i(\mathbf{A}^\top \mathbf{t})^\top \mathbf{x}}] = e^{i(\mathbf{A}^\top \mathbf{t})^\top \mu - (\mathbf{A}^\top \mathbf{t})^\top \mathbf{V} (\mathbf{A}^\top \mathbf{t})/2} = e^{it^\top (\mathbf{A}\mu) - \mathbf{t}^\top (\mathbf{A}\mathbf{V}\mathbf{A}^\top)\mathbf{t}/2}. \quad (3.38)$$

From this, we see that \mathbf{Ax} is a normal random vector with mean vector $\mathbf{A}\mu$ and variance-covariance matrix $\mathbf{A}\mathbf{V}\mathbf{A}^\top$.

Uncorrelated normals with common variance and orthogonality

In the case where the variance-covariance matrix of \mathbf{z} is $\mathbf{V} = \mathbf{I}$, and the mean vector is 0, the preceding result simplifies: the variance-covariance matrix of the normally distributed vector \mathbf{Az} becomes \mathbf{AA}^\top .

If \mathbf{A} is an orthogonal matrix, the result just obtained simplifies further so that we can say that \mathbf{Az} is normally distributed with variance-covariance matrix \mathbf{I} . In other words, the components of \mathbf{Az} are uncorrelated normal random variables. In fact, the characteristic function of \mathbf{Az} must then be $e^{-t^\top t/2}$, but this means, because of (3.27) that \mathbf{Az} must be a vector of independent standard normal random variables.

3.2.7 The distribution of a linear combination of normal random variables

The result of the preceding subsection contains an important special case: *the distribution of a linear combination of independent normal random variables is normal*.

To see this, take \mathbf{A} to have a single row, that is, $m = 1$. Let's call this vector \mathbf{a}^\top . Suppose that \mathbf{z} is an n -vector of normal random variables with mean vector μ and variance-covariance \mathbf{V} . The result of the preceding section tells us that the characteristic function of $\mathbf{a}^\top \mathbf{z}$ must be

$$e^{ita^\top \mu - t^2 a^\top \mathbf{V} a / 2} \quad (3.39)$$

which is the characteristic function of a normal random variable with mean $a^\top \mu = \sum_{j=1}^n a_j \mu_j$ and variance $a^\top \mathbf{V} a$.

When the elements of \mathbf{z} are uncorrelated, with a common variance σ^2 , it can then be seen that $\mathbf{a}^\top \mathbf{z}$ has mean $a^\top \mu$ and variance $\sigma^2 a^\top a$.

3.3 Understanding independence

3.3.1 Graphical views of independence and dependence

We can get an intuitive feel for (3.21) by graphing some simulated random variables, both independent and for some forms of dependence. First, let's consider two independent standard normal random variables, Z_1 and Z_2 . In both cases, we will sample from their respective distributions and look at a scatterplot of the corresponding pairs of data points. Figure 3.6 has been obtained using the following code.

```
par(mar = c(4, 4, 1, 1)); n <- 2000
Z1 <- rnorm(n); Z2 <- rnorm(n)
plot(Z2 ~ Z1)
```

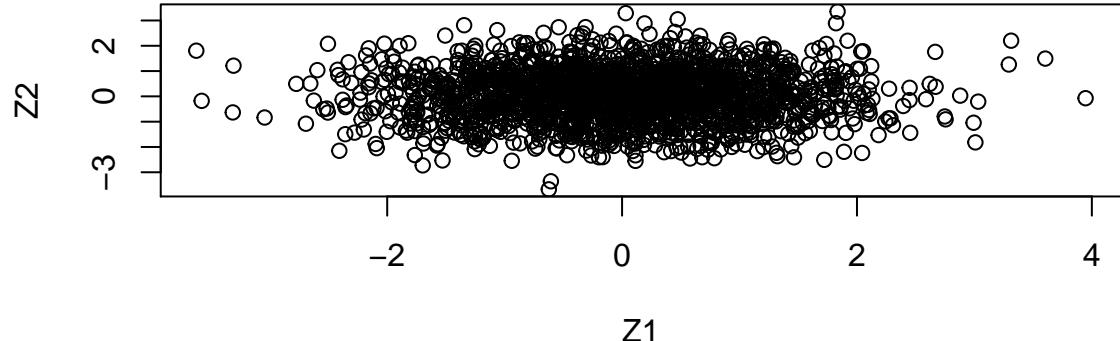


Figure 3.6: A scatterplot of values taken from the distribution of Z_2 against corresponding to values taken from the distribution of Z_1 , when Z_1 and Z_2 are independent random variables.

What you should observe in this figure is that it is not possible to predict the value of Z_2 from knowledge of Z_1 . This is a random collection of points. In order to demonstrate that Z_1 really gives us no new information about

Z_2 , consider the following sequence of plots, for which we condition on information about Z_1 . Specifically, we suppose $Z_1 \in [-2, -1]$, $Z_1 \in [-1, 0]$, $Z_1 \in [0, 1]$ and $Z_1 \in [1, 2]$ successively. In each of the four cases, we plot the histogram of the corresponding Z_2 values. The code to produce Figure 3.7 is as follows:

```
par(mfrow=c(2, 2), mar = c(2, 4, 1, 1))
output <- sapply(split(z2, cut(z1, -2:2)), hist, main = " ")
```

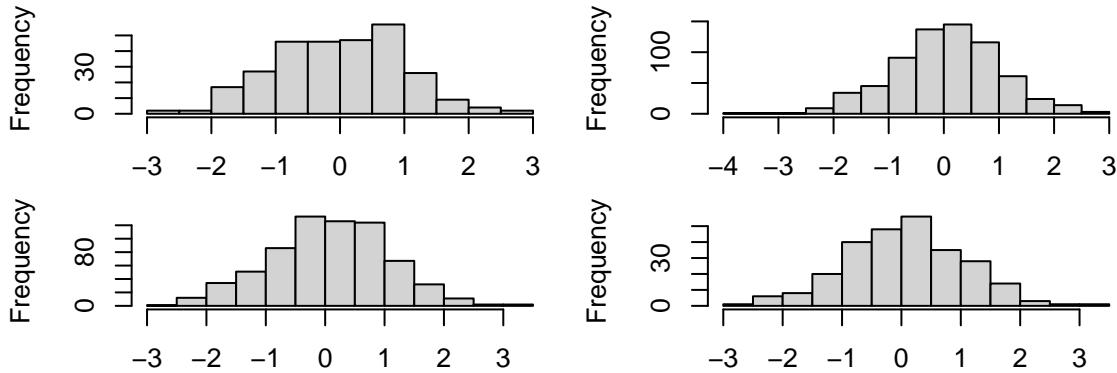


Figure 3.7: Histograms of Z_2 values, conditional on whether Z_1 is in $(-2, -1)$, $(-1, 0)$, $(0, 1)$ or $(1, 2)$. The plot in the top-left corner corresponds to $Z_1 \in (-2, -1)$; the top-right corner corresponds to $Z_1 \in (-1, 0)$; the lower-left corner corresponds to $Z_1 \in (0, 1)$; and the lower-right corner corresponds to $Z_1 \in (1, 2)$.

Note the use of `cut()` to partition the values of Z_1 into subsets with breaks at $-2, -1, 0, 1, 2$. The `split()` function has been used to break up the $z2$ vector into a list of smaller vectors, each corresponding to the different values coming from `cut(z1, -2:2)`. Finally, `sapply()` allows us to apply the same function (here, `hist()`) to each of the four list elements that were obtained from the splitting operation.

The takeaway message from Figure 3.7 is that the distribution of Z_2 appears to be normal with mean 0 and standard deviation 1, no matter what the value of Z_1 is.

A case of dependence

Let's change the story a bit now. Suppose Z_3 and Z_1 are related. In particular, suppose

$$Z_3 = Z_1^2 + Z_4 \quad (3.40)$$

where Z_4 is another normal random variable (which is independent of both Z_1 and Z_3 and which has mean 0 and standard deviation 0.25). Our interest remains in understanding how Z_1 and Z_3 relate to each other. Clearly, there is now a mathematical relation. Now we will apply our simulation-based graphical analysis.

The code to obtain Figure 3.8 is as follows.

```
z4 <- rnorm(1000, sd=0.25)
z3 <- z1^2 + z4
par(mar = c(4, 4, 1, 1))
plot(z3 ~ z1)
```

In case it is not clear from Figure 3.8 that the distribution of Z_3 is changing, depending on the corresponding value of Z_1 , we can make it more clear with Figure 3.9 which is produced in exactly the same way as for Figure 3.7. That is,

```
par(mfrow=c(2, 2), mar = c(2, 4, 1, 1))
output <- sapply(split(z3, cut(z1, -2:2)), hist, main = " ", freq=FALSE)
```

This time, it can be seen that when Z_1 is closer to 0, the distribution of Z_3 is very different than when Z_1 is farther from 0. Z_2 is clearly dependent on Z_1 .

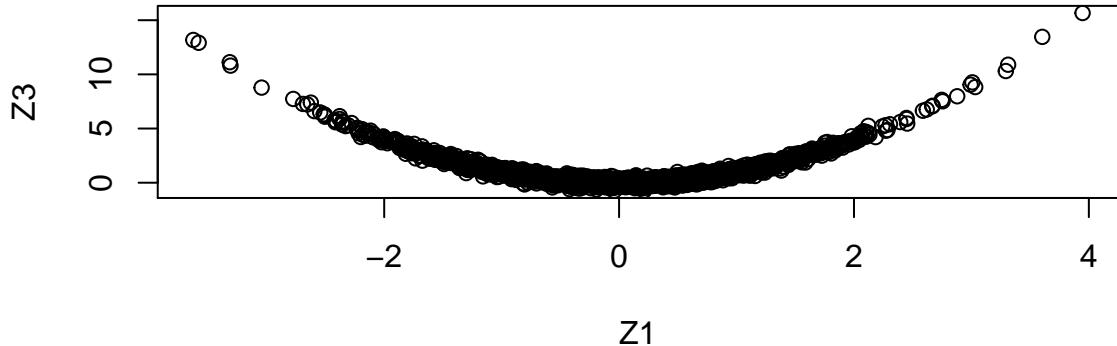


Figure 3.8: A scatterplot of values taken from the distribution of Z_3 against corresponding to values taken from the distribution of Z_1 , when Z_1 and Z_3 are dependent random variables.

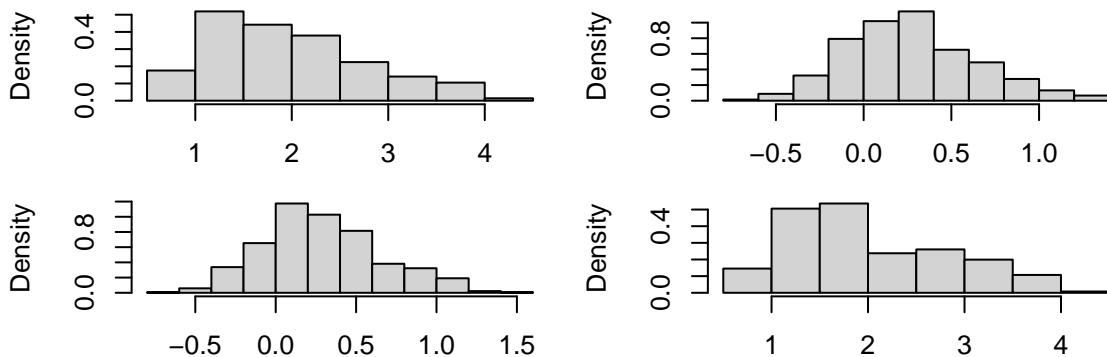


Figure 3.9: Relative frequency histograms of Z_3 values, conditional on whether Z_1 is in $(-2, -1)$, $(-1, 0)$, $(0, 1)$ or $(1, 2)$. The plot in the top-left corner corresponds to $Z_1 \in (-2, -1)$; the top-right corner corresponds to $Z_1 \in (-1, 0)$; the lower-left corner corresponds to $Z_1 \in (0, 1)$; and the lower-right corner corresponds to $Z_1 \in (1, 2)$. This time, Z_3 and Z_1 are dependent.

Orthogonality and independence

Recall that Z_1 and Z_2 are independent standard normal random variables. We now define the following orthogonal transformation:

$$Y_1 = (Z_1 - Z_2)/\sqrt{2} \quad (3.41)$$

and

$$Y_2 = (Z_1 + Z_2)/\sqrt{2}. \quad (3.42)$$

We can use simulation again to see whether Y_1 and Y_2 are dependent, as follows.

```
Y1 <- (Z1-Z2)/sqrt(2)
Y2 <- (Z1+Z2)/sqrt(2)
par(mar = c(4, 4, 1, 1))
plot(Y2 ~ Y1)
```

Figure 3.10 displays the result of the simulation. It is evident that Y_1 and Y_2 are independent, which may seem somewhat surprising. Mathematically, it appears that Y_1 and Y_2 should be dependent, since both variables depend explicitly on Z_1 and Z_2 . However, the nature of the dependence is special. We can write the above two equations in matrix form as

$$\mathbf{y} = \mathbf{P}\mathbf{z} \quad (3.43)$$

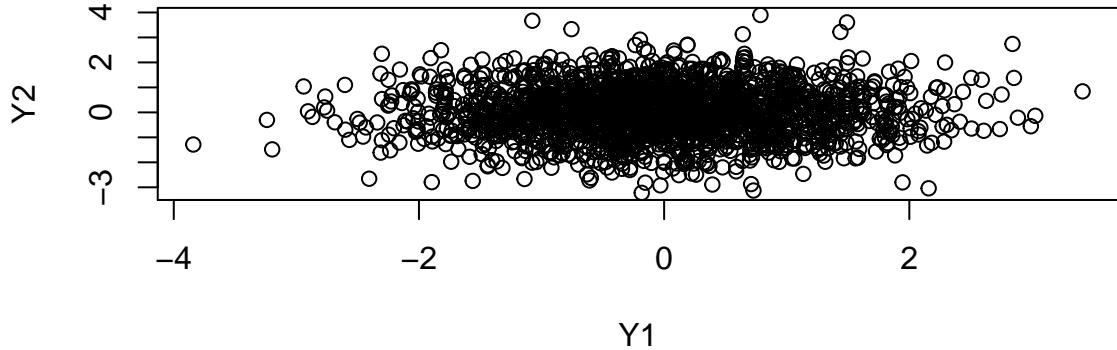


Figure 3.10: Independence of orthogonally transformed independent standard normal random variables.

where \mathbf{P} is a 2×2 matrix. It is easy to see that $\mathbf{P}\mathbf{P}^\top = \mathbf{I}$.

From the theory developed in Section 3.2, we observe that \mathbf{y} must be a normally distributed random vector with mean vector $\mathbf{P}E[\mathbf{z}] = \mathbf{0}$ and variance-covariance matrix $\mathbf{P}\text{Var}(\mathbf{z})\mathbf{P}^\top = \mathbf{P}\mathbf{P}^\top = \mathbf{I}$. Thus, \mathbf{y} is a vector of uncorrelated standard normal random variables, and what we are seeing numerically is that uncorrelated normal random variables are independent.

This is not true for any other kind of random variable. For example, consider the exponential distribution. Here, we simulated 5000 replicates of two independent random exponentials, and carry out the same computations as for the independent normals. Figure 3.11 shows what happens when the orthogonal transformation is applied to the pairs of variables.

```
Z1 <- rexp(5000); Z2 <- rexp(5000)
Y1 <- (Z1-Z2) / sqrt(2); Y2 <- (Z1+Z2) / sqrt(2)
par(mar = c(4, 4, 1, 1))
plot(Y2 ~ Y1)
```

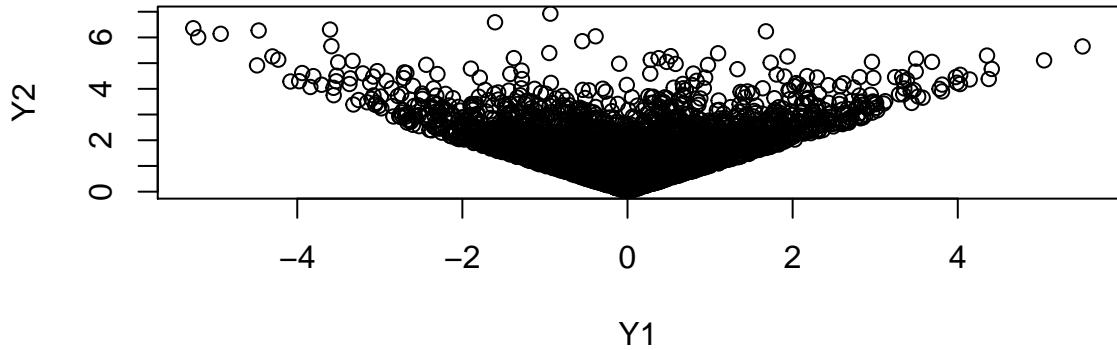


Figure 3.11: Dependence of orthogonally transformed independent exponential random variables.

Note the clear pattern in the graph. Knowledge of Y_1 provides distributional information about Y_2 . For example, if Y_1 is far from 0, it is impossible for Y_2 to be close to 0, while if Y_1 is near 0, Y_2 can be near 0. The two random variables are uncorrelated², but they are *not* independent.

²Why?

3.3.2 Independence of the sample mean and variance

The independence of uncorrelated normal random variables leads to an interesting result concerning the sample mean and sample variance. For normally distributed samples, these two quantities are independent. This will be proved rigorously later, but for now, we will see evidence for this from simulated data.

For the simulation, we wish to obtain distribution information about the sample mean \bar{Z} and the sample variance S_Z^2 , defined as

$$S_Z^2 = \frac{1}{n-1} \sum_{j=1}^n (Z_j - \bar{Z})^2. \quad (3.44)$$

Any single sample of size n will provide us with only one value each of \bar{Z} and S_Z^2 . Therefore, we will need to simulate several samples in order to visualize the distributions of \bar{Z} and S_Z^2 .

Let us consider a samples of $n = 20$ uncorrelated standard normal variables, and let us draw 1000 such samples. Here is a way to do this:

```
m <- 1000; n <- 20
Z <- matrix(rnorm(m*n), nrow=n)
```

Each column of the $n \times m$ matrix Z is one of the samples. We can use the `apply()` function to calculate the sample means and variances:

```
zbar <- apply(Z, 2, mean); S2z <- apply(Z, 2, var)
```

The `MARGIN` argument here is 2 in order to obtain the marginal totals for the second dimension (columns). If we had used 1, we would have obtained totals for the first dimension (rows). Since we are curious about the relation between `zbar` and `S2z`, we will construct a scatterplot.

```
par(mar = c(4, 4.5, 1, 1))
plot(S2z ~ zbar, xlab=expression(bar(Z)), ylab=expression(S[Z]^2))
```

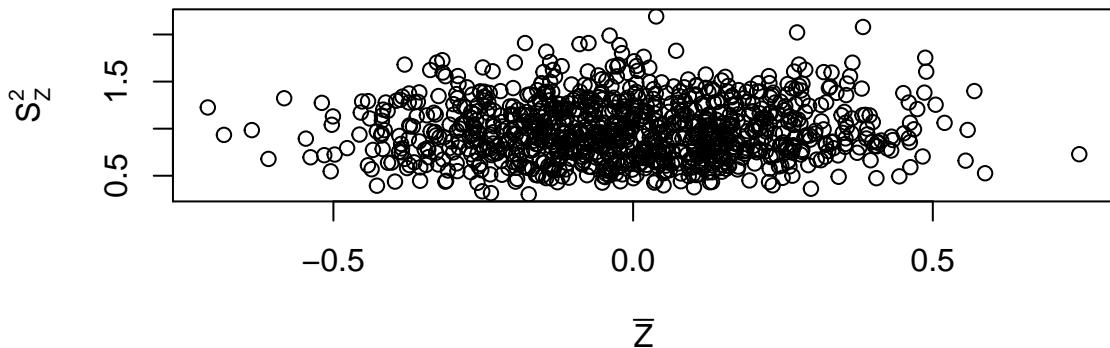


Figure 3.12: Graphical demonstration of independence of \bar{Z} and S_Z^2 in the normal case.

The result is shown in Figure 3.12. The conclusion to be drawn from the pattern on the plot is that \bar{Z} and S_Z^2 are independent. A similar result will be obtained if the entries of Z have a common mean μ and common variance σ^2 . The entries of Z must be uncorrelated, and they must be normal.

Figure 3.13 displays an example to illustrate what happens when the entries of Z are uncorrelated, but not normal.

```
Z <- matrix(rexp(m*n), nrow=n) # exponential data
zbar <- apply(Z, 2, mean); S2z <- apply(Z, 2, var)
```

```
par(mar = c(4, 4.5, 1, 1))
plot(S2z ~ zbar, xlab = expression(bar(z)), ylab = expression(S[z]^2))
```

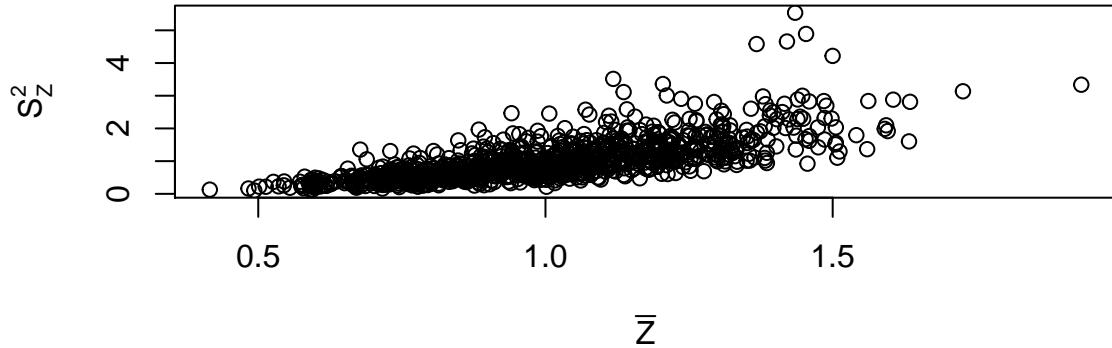


Figure 3.13: Graphical demonstration of dependence between the sample mean and variance in the case of non-normal data.

Now, we see a clear pattern in Figure 3.13; as \bar{Z} increases, we see an increase in S_Z^2 .

In case, the reader thinks that this is an artifact of the skewness of the exponential distribution, Figure 3.14, based on the symmetric t -distribution (see Section 3.4.3), shows that this is not the case. A larger number of simulation runs are required in order to clearly see the pattern.

```
m <- 5000
Z <- matrix(rt(m*n, df=3), nrow=n) # t data on 3 degrees of freedom
zbar <- apply(Z, 2, mean); S2z <- apply(Z, 2, var)
par(mar = c(4, 4.5, 1, 1))
plot(S2z ~ zbar, xlab=expression(bar(z)), ylab=expression(S[z]^2))
```

Again, we see a clear pattern, as \bar{Z} increases in absolute value, we see an increase in S_Z^2 . Because the samples do not consist of uncorrelated normal random variables, the sample average and the sample variance are not independent.

3.4 Random variables constructed from normals

If Y is a normal random variable with mean μ and standard deviation σ , then

$$Z = \frac{Y - \mu}{\sigma} \tag{3.45}$$

is a standard normal random variable.

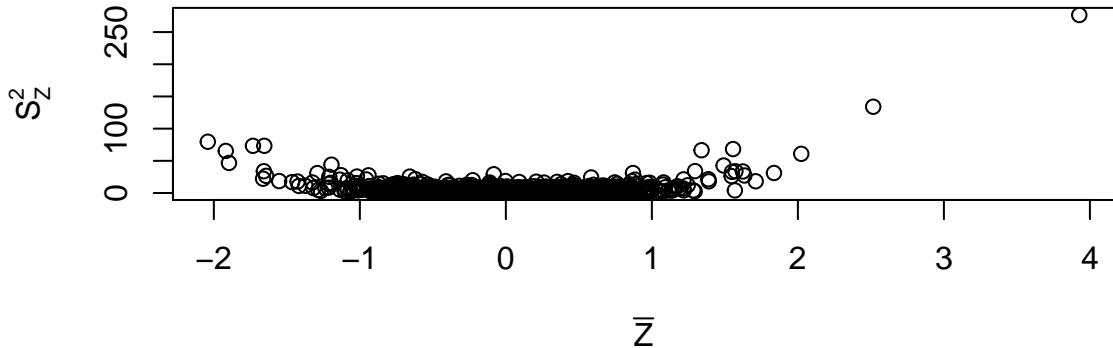


Figure 3.14: Graphical demonstration of dependence between the sample mean and variance in the case of non-normal data. In this case, t distributed data have been simulated.

3.4.1 The χ^2 random variable

Squaring Z leads to a χ^2 random variable on 1 degree of freedom. Note that since

$$E[Z^2] = 1 \quad (3.46)$$

it follows immediately that the expected value of a $\chi_{(1)}^2$ random variable must also be 1. If Z_1, \dots, Z_n is a sequence of n independent standard normal random variables, then

$$X = \sum_{j=1}^n Z_j^2 \quad (3.47)$$

is a $\chi_{(n)}^2$ random variable on n degrees of freedom. A relative frequency histogram of 1000 simulated values of X for the case where $n = 7$ is displayed in Figure 3.15. The graph of the density function of a $\chi_{(7)}^2$ random variable is overlaid.

```
X <- rchisq(1000, df = 7)
par(mar = c(2, 4, 1, 1))
hist(X, freq = FALSE, main = " ")
curve(dchisq(x, df = 7), from = 0, to = 25, add = TRUE)
```

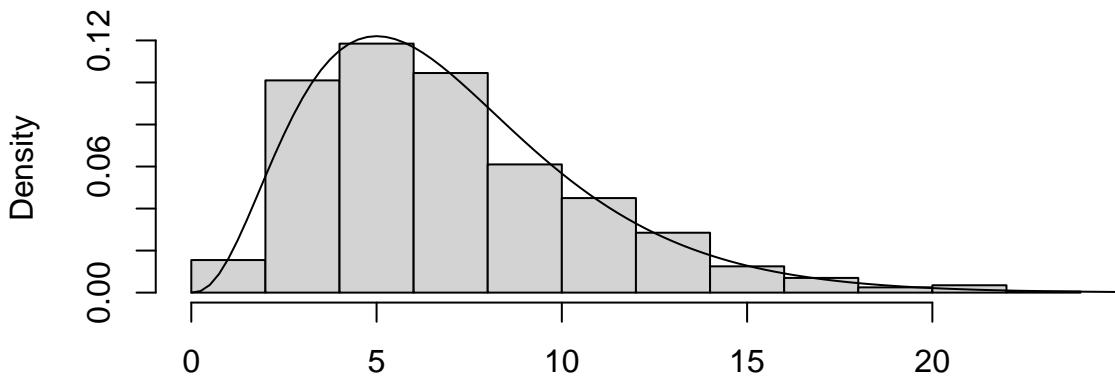


Figure 3.15: Relative frequency histogram of 1000 simulated $\chi_{(7)}^2$ random variables with overlaid density curve.

The expected value of X must be n , by the linearity of the expected value operator. Observe also that if $Y_j = \mu + \sigma Z_j$, i.e. Y_j is $N(\mu, \sigma^2)$, then

$$X = \sum_{j=1}^n \left(\frac{Y_j - \mu}{\sigma} \right)^2. \quad (3.48)$$

This implies that if μ was known, but σ^2 was unknown, we could estimate σ^2 from a sample of Y 's in an unbiased manner by using the formula

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{j=1}^n (Y_j - \mu)^2. \quad (3.49)$$

Unbiasedness follows from noting that

$$E[\hat{\sigma}^2] = \frac{1}{n} \sum_{j=1}^n E[(Y_j - \mu)^2] = \sigma^2. \quad (3.50)$$

Usually μ is not known, so a slight modification to the σ^2 estimator is required in practice. In fact, it will be shown later that the sample variance S_Y^2 can be used as an unbiased estimator for σ^2 , and that $(n - 1)S_Y^2/\sigma^2$ is a χ^2 random variable on $n - 1$ degrees of freedom.

3.4.2 The F random variable

If X_1 and X_2 are independent χ^2 random variables on m and n degrees of freedom, respectively, then

$$F = \frac{X_1/m}{X_2/n} \quad (3.51)$$

is an F random variable on m and n degrees of freedom. m is sometimes referred to as the numerator degrees of freedom, and n is the denominator degrees of freedom.

A relative frequency histogram of 1000 simulated values of F for the case where $m = 3$ and $n = 7$ is displayed in Figure 3.16. The graph of the density function of a $F_{(3,7)}$ random variable is overlaid.

```
F <- rf(1000, df1 = 3, df2 = 7)
par(mar = c(2, 4, 1, 1))
hist(F, freq = FALSE, ylim = c(0, 0.7), main = " ")
curve(df(x, df1 = 3, df2 = 7), from = 0, to = 15, add = TRUE)
```

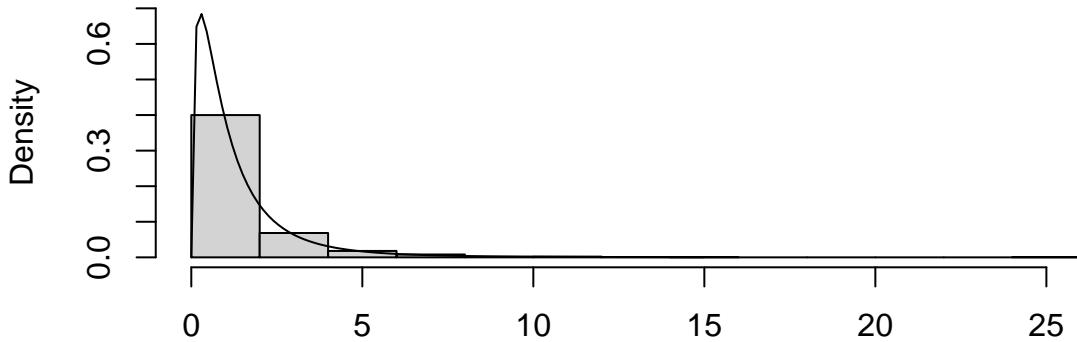


Figure 3.16: Relative frequency histogram of 1000 simulated $F_{(3,7)}$ random variables with overlaid density curve.

3.4.3 The t random variable

Suppose Z is a standard normal random variable and suppose X is a χ^2 random variable on n degrees of freedom, then

$$T = \frac{Z}{\sqrt{X/n}} \quad (3.52)$$

is a t random variable on n degrees of freedom, provided that Z and X are independent of each other.

A relative frequency histogram of 1000 simulated values of t for the case where $n = 7$ is displayed in Figure 3.17. The graph of the density function of a $t_{(7)}$ random variable is overlaid. The symmetry of the t -distribution is evident; it has a somewhat bell-shape like the normal distribution, but its tails decrease towards 0 more slowly.

```
T <- rt(1000, df = 7)
par(mar = c(2, 4, 1, 1))
hist(T, freq = FALSE, ylim = c(0, .5), main = " ")
curve(dt(x, df = 7), from = -5, to = 5, add = TRUE)
```

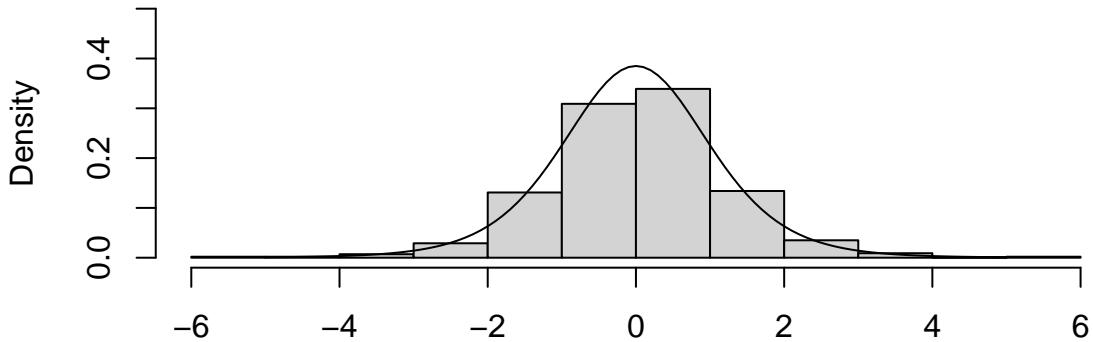


Figure 3.17: Relative frequency histogram of 1000 simulated $t_{(7)}$ random variables with overlaid density curve.

Studentizing yields a t random variable

We have seen that \bar{Y} is normally distributed with mean μ and variance σ^2/n , if the underlying sample consists of n uncorrelated normal random variables with common mean μ and common variance σ^2 . We have also seen, empirically, that for such a sample, \bar{Y} and S_Y^2 are independent. We have also noted that $(n-1)S_Y^2/\sigma^2$ is a $\chi^2_{(n-1)}$ random variable.

We will now show that

$$\frac{\bar{Y} - \mu}{S_Y/\sqrt{n}} \quad (3.53)$$

is a t random variable on $n - 1$ degrees of freedom. To see this, we first set

$$Z = \frac{\bar{Y} - \mu}{\sigma/\sqrt{n}} \quad (3.54)$$

and recognize that this is standard normal. Z is also independent of $(n-1)S_Y^2/\sigma^2$, so

$$Z/\sqrt{S_Y^2/\sigma^2} \quad (3.55)$$

is a t random variable on $n - 1$ degrees of freedom, and the result follows from some elementary algebra.

3.5 Exercises

1. Show, using integration, that the characteristic function of a standard normal random variable Z is $\phi(t) = e^{-t^2/2}$. Deduce that the characteristic function of $X = \mu + \sigma Z$ must be $e^{\mu it - \sigma^2 t^2/2}$.
2. In what follows, Z_1, Z_2, \dots, Z_n is a sequence of independent standard normal random variables. Also, X_1, X_2, \dots, X_n , W_1, W_2, \dots, W_m and V_1, V_2, \dots, V_k are independent sets of normal random variables with common variance σ^2 and respective means μ_X, μ_W and μ_V .

S_X^2, S_W^2 and S_V^2 denote the sample variances of the X 's, W 's and V 's, respectively.

Identify the distributions (specifying values of parameters wherever possible) of the following random variables:

(a)

$$Z_1^2 \quad (3.56)$$

(b)

$$Z_1^2 + Z_2^2 + Z_3^2 \quad (3.57)$$

(c)

$$\frac{1}{2\sigma^2} (X_1 - X_2)^2 \quad (3.58)$$

(d)

$$\sum_{i=1}^n Z_i^2 \quad (3.59)$$

(e)

$$\frac{Z_1^2}{Z_2^2} \quad (3.60)$$

(f)

$$\frac{S_W^2}{S_V^2} \quad (3.61)$$

(g)

$$\frac{\bar{X} - \mu_X}{S_W / \sqrt{n}} \quad (3.62)$$

(h)

$$\frac{(n-1)S_X^2}{\sigma^2} + \frac{n(\bar{X} - \mu_X)^2}{\sigma^2} \quad (3.63)$$

(i)

$$\frac{S_X^2/n}{(\bar{X} - \mu_X)^2} \quad (3.64)$$

(j)

$$\frac{n(\bar{X} - \mu_X)^2}{\sigma^2} + \frac{m(\bar{W} - \mu_W)^2}{\sigma^2} + \frac{k(\bar{V} - \mu_V)^2}{\sigma^2} \quad (3.65)$$

(k)

$$\frac{\sqrt{2n}(\bar{X} - \mu_X)}{\sqrt{m(\bar{W} - \mu_W)^2 + k(\bar{V} - \mu_V)^2}} \quad (3.66)$$

3. Suppose \mathbf{z} is a random vector consisting of 2 independent elements, each having a distribution with mean 3 and variance 7. Calculate the mean vector and variance-covariance matrix of $\mathbf{A}\mathbf{z}$ when

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}. \quad (3.67)$$

Is \mathbf{A} an orthogonal matrix?

4. Suppose \mathbf{z} is a random vector consisting of 3 independent elements, each having a distribution with mean 0 and variance 4. Calculate the mean vector and variance-covariance matrix of $\mathbf{y} = \mathbf{A}\mathbf{z}$ when

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & 0 & 1 \end{bmatrix}. \quad (3.68)$$

Is \mathbf{A} an orthogonal matrix? Are the elements of \mathbf{y} uncorrelated? Under what condition can we say that the elements of \mathbf{y} are independent?

5. Referring to the previous question, suppose that the elements of \mathbf{z} are normally distributed.

- (a) What is the distribution of \mathbf{y} ?
 (b) What is the distribution of $\mathbf{D}\mathbf{y}$ when

$$\mathbf{D} = \begin{bmatrix} 1/\sqrt{12} & 0 & 0 \\ 0 & 1/\sqrt{24} & 0 \\ 0 & 0 & 1/\sqrt{8} \end{bmatrix}. \quad (3.69)$$

- (c) Identify the distribution of $\mathbf{y}^\top \mathbf{D}^2 \mathbf{y}$.

6. Referring to the previous question, suppose that $\mathbf{w} = \mathbf{B}\mathbf{z}$ where

$$\mathbf{B} = \begin{bmatrix} 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \end{bmatrix}. \quad (3.70)$$

- (a) What is the distribution of \mathbf{w} ?
 (b) What is the distribution of $\mathbf{w}^\top \mathbf{w}$?

7. Simulate 1000 independent standard normal random variables using the `rnorm()` function. Use these to construct 1000 independent χ^2 random variables on 1 degree of freedom. Plot these as a relative frequency histogram, and overlay the $\chi^2_{(1)}$ density function.
8. Simulate two independent samples of 1000 χ^2 random variables, using the `rchisq()` function, where the first is on 5 degrees of freedom and the second is on 25 degrees of freedom. Use these samples to construct a single sample of 1000 $F_{(5,25)}$ random variables. Plot these as a relative frequency histogram, and overlay the $F_{(5,25)}$ density function.

4

Multiple Regression

4.1 A motivating example: house price data

The data frame `table.b4` in the *MPV* package contains the following columns:

```
y sale price of the house (in thousands of dollars)
x1 taxes (in thousands of dollars)
x2 number of baths
x3 lot size (in thousands of square feet)
x4 living space (in thousands of square feet)
x5 number of garage stalls
x6 number of rooms
x7 number of bedrooms
x8 age of the home (in years)
x9 number of fireplaces
```

There are 24 observations on these variables in the data frame. A natural question to ask is whether any or all of the given x variables or covariates could be used to predict the sale price of a house. The sale price is an example of a response variable, and the covariates which are used in a model to predict or explain sale price are referred to as predictor variables or explanatory variables.

4.1.1 Developing multiple regression models

Recall the simple linear regression model

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

where the noise or error, $\varepsilon_i \sim N(0, \sigma^2)$ (i.i.d.). The normality assumption could be relaxed. We extend this model to a multiple regression model by adding more variables on the right side, that is, more explanatory variables.

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \varepsilon_i$$

For the house sale price example, if we included all covariates in the model, we would have $k = 9$ terms in the model in addition to the intercept.

The multiple regression framework includes special types of nonlinear models. For example, we could try to model the response as a k th degree polynomial. Such a polynomial regression model is given by

$$Y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_k x_i^k + \varepsilon_i$$

We could also include polynomial terms in several different variables, such as x_1^2 and x_2^2 as well as interaction terms, such as $x_1 x_2$ or $x_1^2 x_2$. Therefore, being able to construct multiple regression models gives a great deal of flexibility to how data can be analyzed.

4.1.2 Matrix form

Handling multiple covariates is more convenient if we use vectors and matrices. The multiple regression model can be written as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{4.1}$$

where \mathbf{y} is the column vector of responses:

$$\mathbf{y} = [Y_1 \ Y_2 \ \cdots \ Y_n]^\top.$$

On the right hand side of (4.1), we have

$$\beta = [\beta_0 \ \beta_1 \ \cdots \ \beta_k]^\top$$

and

$$\epsilon = [\varepsilon_1 \ \varepsilon_2 \ \cdots \ \varepsilon_n]^\top.$$

\mathbf{X} is an $n \times (k + 1)$ matrix, called the model matrix or design matrix:

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix}$$

The following special cases can help to illustrate the notation.

- regression through the origin:

$$Y_i = \beta_1 x_i + \varepsilon_i$$

is

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

with

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \text{ and } \beta = [\beta_1]$$

- simple linear regression:

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

is

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

with

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \text{ and } \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$$

- quadratic regression:

$$Y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \varepsilon_i$$

is

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

with

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \text{ and } \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

- regression with 2 predictor variables

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \varepsilon_i$$

is

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

with

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} \\ 1 & x_{21} & x_{22} \\ \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, \quad \boldsymbol{\epsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \quad \text{and} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

4.2 Least-squares estimation via the QR decomposition

The usual multiple regression model is assumed:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where \mathbf{y} and $\boldsymbol{\epsilon}$ are vectors of length n , $\boldsymbol{\beta}$ is a vector of length $p = k + 1$ and \mathbf{X} is an $n \times p$ matrix ($n > p$).

For the house price example, $n = 24$ and we might start with $p = 10$. In other words, we would consider a linear model of the form

$$Y = \beta_0 + \sum_{j=1}^9 \beta_j x_j + \varepsilon.$$

If the β coefficients were known, then we could predict house price Y , to within the unknown noise value ε , for a new house in the same area (and era) from which the data were sampled, provided the information on taxes, number of baths, and so on. If, for example, the ε term is modelled as a normal random variable with mean 0 and variance σ^2 , we could provide an interval which would contain the true price of the house with a given probability.

The columns of \mathbf{X} are assumed to be linearly independent, and if an intercept is in the model, the first column is a vector of 1's. The other columns are assumed to be nonrandom (i.e. a fixed design), or, as in the case of simple regression, we carry out the analysis, conditional on the values in these columns if we are dealing with a random design. The elements of ε are assumed to be uncorrelated random variables with mean 0 and common variance σ^2 . Thus, the expected value and variance of the vector \mathbf{y} are

$$E[\mathbf{y}|\mathbf{X}] = \mathbf{X}\boldsymbol{\beta}$$

and

$$\text{Var}(\mathbf{y}|\mathbf{X}) = \sigma^2 \mathbf{I}$$

where \mathbf{I} is the $n \times n$ identity matrix.

4.2.1 Obtaining the QR decomposition of the design matrix

Given the model, we can use results from Appendix E to see that it is always possible to find an $n \times n$ orthogonal matrix \mathbf{Q} and an $n \times p$ matrix \mathbf{R} such that

$$\mathbf{X} = \mathbf{QR}$$

where the bottom $n - p$ rows of \mathbf{R} consist of 0's and the top p rows of \mathbf{R} constitute a nonsingular upper triangular matrix \mathbf{U} . That is, $\mathbf{Q}^\top \mathbf{Q} = \mathbf{I}$ and

$$\mathbf{R} = \begin{bmatrix} \mathbf{U} \\ \mathbf{0} \end{bmatrix}.$$

Example – house price data

For the house price data, we can obtain these quantities in the following way:

```
n <- nrow(table.b4) # this counts the number of observations in the data set
X <- table.b4[,-1] # removes the y vector
X <- cbind(x0=rep(1, n), X) # this adds the vector of 1's to the design matrix.
X.QR <- qr(X)
```

```
R <- qr.R(X.QR, complete=TRUE)
Q <- qr.Q(X.QR, complete=TRUE)
```

4.2.2 Estimation via QR

The least-squares objective function to be minimized, with respect to the components of β is

$$(\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta).$$

Since we can write $\mathbf{X} = \mathbf{QR}$, where $\mathbf{Q}^{-1} = \mathbf{Q}^T$ exists, we can re-write this as

$$(\mathbf{y} - \mathbf{QR}\beta)^T(\mathbf{y} - \mathbf{QR}\beta)$$

or

$$(\mathbf{Q}^T\mathbf{y} - \mathbf{R}\beta)^T\mathbf{Q}^T\mathbf{Q}(\mathbf{Q}^T\mathbf{y} - \mathbf{R}\beta).$$

Because \mathbf{Q} is orthogonal, the least-squares objective simplifies to

$$(\mathbf{Q}^T\mathbf{y} - \mathbf{R}\beta)^T(\mathbf{Q}^T\mathbf{y} - \mathbf{R}\beta).$$

Therefore, solving the least-squares problem for the model in its original form is equivalent to solving the least-squares problem for the following model:

$$\mathbf{Q}^T\mathbf{y} = \mathbf{R}\beta + \mathbf{Q}^T\epsilon.$$

Partitioning \mathbf{Q} as $[\mathbf{Q}_1 \ \mathbf{Q}_2]$, where \mathbf{Q}_1 represents the first p columns of \mathbf{Q} , the above equation can be re-written as the pair of matrix equations

$$\mathbf{Q}_1^T\mathbf{y} = \mathbf{U}\beta + \mathbf{Q}_1^T\epsilon \quad (4.2)$$

and

$$\mathbf{Q}_2^T\mathbf{y} = \mathbf{Q}_2^T\epsilon. \quad (4.3)$$

because the lower $n - p$ rows of \mathbf{R} contain only 0's.

To estimate β , it is clear that model (4.2) contains all of the needed information, since (4.3) does not involve β . The least-squares problem for model (4.2) requires minimizing the sum of squares

$$(\mathbf{Q}_1^T\mathbf{y} - \mathbf{U}\beta)^T(\mathbf{Q}_1^T\mathbf{y} - \mathbf{U}\beta)$$

with respect to β . Since this sum cannot be negative, but it can take on the value 0 when

$$\mathbf{U}\beta = \mathbf{Q}_1^T\mathbf{y},$$

we conclude that $\hat{\beta}$ must be the solution of this upper triangular linear system. Thus, mathematically, we have that $\hat{\beta} = \mathbf{U}^{-1}\mathbf{Q}_1^T\mathbf{y}$ is the least-squares estimator for β . The fitted values are then easily calculated as

$$\hat{\mu} = \widehat{E[\mathbf{y}|\mathbf{X}]} = \mathbf{X}\hat{\beta}.$$

Observe that, since $\mathbf{X} = \mathbf{Q}_1\mathbf{U}$, and $\hat{\beta} = \mathbf{U}^{-1}\mathbf{Q}_1^T\mathbf{y}$, the fitted values can also be expressed as $\hat{\mu} = \mathbf{Q}_1\mathbf{Q}_1^T\mathbf{y}$. The matrix $\mathbf{H} = \mathbf{Q}_1\mathbf{Q}_1^T$ is called the influence matrix and is also referred to as the hat matrix.

4.2.3 Estimation of the noise variance

To estimate σ^2 , model equation (4.3) is used. It is easily shown that

$$E[\mathbf{Q}_2^\top \mathbf{y} | \mathbf{X}] = 0$$

and

$$\text{Var}(\mathbf{Q}_2^\top \mathbf{y} | \mathbf{X}) = \mathbf{Q}_2^\top \sigma^2 \mathbf{I} \mathbf{Q}_2 = \sigma^2 \mathbf{I}.$$

Thus, $\mathbf{Q}_2^\top \mathbf{y}$ is a vector of $n - p$ uncorrelated mean 0 and variance σ^2 random variables. Hence $\mathbf{y}^\top \mathbf{Q}_2 \mathbf{Q}_2^\top \mathbf{y}$ must be the sum of squares of such random variables: that is,

$$\mathbf{y}^\top \mathbf{Q}_2 \mathbf{Q}_2^\top \mathbf{y} = \sum_{i=1}^{n-p} Z_i^2 \quad (4.4)$$

where Z_1, Z_2, \dots, Z_{n-p} are uncorrelated with mean 0 and variance σ^2 . Therefore,

$$E[\mathbf{y}^\top \mathbf{Q}_2 \mathbf{Q}_2^\top \mathbf{y} | \mathbf{X}] = \sum_{i=1}^{n-p} \text{Var}(Z_i) = (n - p)\sigma^2$$

and an unbiased estimator for σ^2 is

$$\hat{\sigma}^2 = \frac{\mathbf{y}^\top \mathbf{Q}_2 \mathbf{Q}_2^\top \mathbf{y}}{n - p}. \quad (4.5)$$

In Chapter 2, this was referred to as the MSE for the case where $p = 2$. The form given there is mathematically equivalent to the form here.

Estimating the parameters of the house price model

For the house price data, we can estimate the coefficient vector β as follows:

```
y <- table.b4[, 1]
QTy <- t(Q) %*% y
U <- R[1:10,] # this extracts the top ten rows of R
betahat <- backsolve(U, QTy) # efficient solution of an upper triangular system
betahat # coefficient estimates

## [,1]
## [1,] 14.92764759
## [2,] 1.92472156
## [3,] 7.00053420
## [4,] 0.14917793
## [5,] 2.72280790
## [6,] 2.00668402
## [7,] -0.41012376
## [8,] -1.40323530
## [9,] -0.03714908
## [10,] 1.55944663
```

Estimating σ^2 is as follows:

```
Q2Ty <- QTy[-(1:10)] # removing Q1Ty values from QTy
sigma2hat <- mean(Q2Ty^2) # note that there are n-p elements here
sigma2hat # estimate of error variance

## [1] 8.696297
```

4.3 Inference

In this section, we begin to outline how regression models are analyzed. Estimating model parameters actually plays only a small role in the larger enterprise of regression analysis. The reliability of parameter estimates and their standard errors depends heavily on the validity of the assumptions that we make. The following assumptions will be made throughout the remainder of this chapter:

- $E[\mathbf{y}|\mathbf{X}] = \mathbf{X}\beta$. (i.e. $E[\epsilon] = \mathbf{0}$.)
- ϵ_i is uncorrelated with ϵ_j , for $i \neq j$.
- ϵ is independent of the covariates.
- ϵ has variance-covariance matrix $\sigma^2 \mathbf{I}$. (i.e. constant variance)

At times, we will make the further assumption that \mathbf{y} has a multivariate normal distribution.

4.3.1 Statistical properties of $\hat{\beta}$

Unbiasedness of $\hat{\beta}$ is easily verified, since it can be shown that

$$\hat{\beta} = \beta + \mathbf{U}^{-1} \mathbf{Q}_1^\top \epsilon. \quad (4.6)$$

This last result can also be used to see that

$$\text{Var}(\hat{\beta}|\mathbf{X}) = \mathbf{U}^{-1} \mathbf{Q}_1^\top \text{Var}(\epsilon) \mathbf{Q}_1 \mathbf{U}^{-\top} = \sigma^2 \mathbf{U}^{-1} \mathbf{U}^{-\top}. \quad (4.7)$$

This variance-covariance matrix can be estimated by replacing σ by $\hat{\sigma}$. The diagonal elements of the resulting estimated matrix are the squares of the standard errors of the elements of $\hat{\beta}$. These quantities will be of use later, when we develop confidence intervals for the regression coefficients.

4.3.2 Estimating the mean response at \mathbf{x}_0

The expected value of Y at given values of the explanatory variables, summarized by the row vector \mathbf{x}_0 (which is of length p and whose first component must be ‘1’, if an intercept has been included in the model), is given by

$$E[Y|\mathbf{x}_0] = \mathbf{x}_0 \beta.$$

We estimate this with

$$\widehat{E[Y|\mathbf{x}_0]} = \mathbf{x}_0 \hat{\beta}.$$

Estimation error

We can assess the uncertainty in this estimate by calculating the standard error of this estimate. First, we note that the variance of the estimator is

$$\text{Var}(\mathbf{x}_0 \hat{\beta}|\mathbf{x}_0) = \mathbf{x}_0 \text{Var}(\hat{\beta}|\mathbf{x}_0) \mathbf{x}_0^\top = \sigma^2 \mathbf{x}_0 (\mathbf{U}^{-1} \mathbf{U}^{-\top}) \mathbf{x}_0^\top.$$

Taking the square root of this quantity gives us the standard error of the estimate of the expected response at \mathbf{x}_0^\top :

$$\text{SE} = \sqrt{\mathbf{x}_0 \text{Var}(\hat{\beta}|\mathbf{x}_0) \mathbf{x}_0^\top} = \sigma \sqrt{\mathbf{x}_0 (\mathbf{U}^{-1} \mathbf{U}^{-\top}) \mathbf{x}_0^\top}.$$

Replace σ by its estimate $\hat{\sigma} = \sqrt{\text{MSE}}$ to obtain an estimate of this standard error. Multiplying the estimated standard error by the $(1 - \alpha/2)$ quantile of the t distribution on $n - p$ degrees of freedom gives us the margin of error in a $(1 - \alpha)$ confidence interval for the expected response at \mathbf{x}_0^\top .

Prediction error

To predict a single response value Y for given values of the explanatory variables, summarized by the row vector \mathbf{x}_0 (which is of length p and whose first component must be ‘1’, if an intercept has been included in the model), we use the estimated response for the point prediction, but the prediction error must take into account not only error in the estimate of the mean, it must also account for the prediction error associated with any variables that have not been measured. For the house price example, such variables might include local crime rate, economic conditions, desirability of the neighbourhood, quality of nearby schools, and so on.

Using the techniques developed earlier, we can see that the variance of the prediction error is

$$\text{Var}(\mathbf{x}_0\hat{\beta} + \varepsilon|\mathbf{x}_0) = \mathbf{x}_0(\mathbf{U}^{-1}\mathbf{U}^{-T})\mathbf{x}_0^T\sigma^2 + \sigma^2.$$

We have assumed that the noise term associated with the new response is independent of the data used to fit the model, and we have used the variance of $\hat{\beta}$ as obtained at (4.7). The standard deviation of the prediction error is thus

$$\sigma\sqrt{\mathbf{x}_0(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{x}_0^T + 1}.$$

Again, we can replace σ by $\hat{\sigma}$ to estimate this prediction standard error.

Example – house price data

If we wish to assess the value of a house in the same neighbourhood or set of neighbourhoods as the houses in the original sample, and during the same time period, and if the model we have fit is to be believed, we proceed as follows.

First, we collect the information on all covariates for the new house. For example, suppose the following information has been obtained on a given house (not in the original sample):

- taxes: \$1000
- bathrooms: 1
- lot size: 6000 sq. ft.
- living space: 1500 sq. ft.
- garage stalls: 1
- number of rooms: 6
- number of bedrooms: 2
- age of home: 60 years
- number of fireplaces: 0

In R, we could enter this data as

```
x <- c(1, 1, 6, 1.5, 1, 6, 2, 60, 0)
```

To include the intercept term, type

```
x0 <- c(1, x)
```

The predicted value of the house price is then calculated as

```
x0%*%betahat # remember that this is in 1000's of dollars
## [1,] 23.34271
```

We can estimate the variance of the estimation error and the variance of the prediction error as follows

```

Uinvx <- forwardsolve(t(U), x0)
estVar <- sum(Uinvx^2) * sigma2hat
estVar

## [1] 51.19446

predVar <- (sum(Uinvx^2)+1) * sigma2hat
predVar

## [1] 59.89076

```

The standard error of the mean response and the standard error of the prediction are the square roots of these values:

```

SEM <- sqrt(estVar)
SEM

## [1] 7.155031

SEpred <- sqrt(predVar)
SEpred

## [1] 7.738912

```

We could summarize our results by saying that the expected price of a house with the given characteristics is \$23342 with a standard error of \$715. We would predict that a particular house with those same characteristics might sell for \$23342, and the standard error of our prediction is \$774.

The predictions might be improved (i.e. the prediction standard error might be reduced, for example) by removing extraneous variables from the model. For example, the number of fireplaces might not really affect house price; if not, we might obtain a simpler and more precise model by not including this covariate. The next sections address some of the technology that has been developed to make such model improvements.

4.3.3 *t* statistics

If ϵ is assumed to be normally distributed, then equation (4.4) can be used to deduce that $(n - p)\hat{\sigma}^2 = \text{SSE}$ is a χ^2 random variable on $n - p$ degrees of freedom. Also, $\mathbf{Q}_2^\top \mathbf{Q}_1 = \mathbf{0}$. $\mathbf{Q}_1^\top \mathbf{y}$ and $\mathbf{Q}_2^\top \mathbf{y}$ are vectors of uncorrelated normal random variables, hence they are independent. Therefore, $\hat{\beta}$ (which depends only on $\mathbf{Q}_1^\top \mathbf{y}$) and $\text{MSE} = \text{SSE}/(n - p)$ (which depends only on $\mathbf{Q}_2^\top \mathbf{y}$) must be independent, when the data are normally distributed.

t statistic and confidence interval for estimation of mean response

When \mathbf{y} is normally distributed, we can standardize the mean response estimate at \mathbf{x}_0 according to $\frac{\mathbf{x}_0 \hat{\beta} - \mathbf{x}_0 \beta}{\sigma \sqrt{\mathbf{x}_0 (\mathbf{U}^{-1} \mathbf{U}^{-\top}) \mathbf{x}_0^\top}}$; this quantity is then standard normal, and because SSE/σ^2 is distributed as $\chi^2_{(n-p)}$, $\frac{\mathbf{x}_0 \hat{\beta} - \mathbf{x}_0 \beta}{\sqrt{\text{MSE}} \sqrt{\mathbf{x}_0 (\mathbf{U}^{-1} \mathbf{U}^{-\top}) \mathbf{x}_0^\top}}$ must have a *t* distribution on $n - p$ degrees of freedom, since this ratio consists of a standard normal random variable divided by the square root of a $\chi^2_{(n-p)}$ random variable divided by its degrees of freedom.

A $1 - \alpha$ confidence interval for $E[Y|\mathbf{x}_0] = \mathbf{x}_0 \beta$ is then

$$\mathbf{x}_0 \hat{\beta} \pm t_{n-p, \alpha/2} \sqrt{\text{MSE}} \sqrt{\mathbf{x}_0 (\mathbf{U}^{-1} \mathbf{U}^{-\top}) \mathbf{x}_0^\top}.$$

t statistic and prediction interval for a new response

The argument here is almost identical to the above, with some small but important modifications. The prediction of a new response at \mathbf{x}_0 must necessarily incorporate the noise term, since we saw earlier that a new response should be of the form

$$Y_{\mathbf{x}_0} = \mathbf{x}_0 \beta + \varepsilon.$$

This is standardized according to $\frac{Y_{\mathbf{x}_0} - \mathbf{x}_0\beta}{\sigma\sqrt{1+\mathbf{x}_0(\mathbf{U}^{-1}\mathbf{U}^{-\top})\mathbf{x}_0^\top}}$, and because SSE/ σ^2 is distributed as $\chi^2_{(n-p)}$,

$$\frac{\mathbf{x}_0\hat{\beta} + \varepsilon - \mathbf{x}_0\beta}{\sqrt{\text{MSE}\sqrt{1 + \mathbf{x}_0(\mathbf{U}^{-1}\mathbf{U}^{-\top})\mathbf{x}_0^\top}}}$$

must have a t distribution on $n - p$ degrees of freedom, since this ratio consists of a standard normal random variable divided by the square root of a $\chi^2_{(n-p)}$ random variable divided by its degrees of freedom.

A $1 - \alpha$ prediction interval for a new response Y at \mathbf{x}_0 is then

$$\mathbf{x}_0\hat{\beta} \pm t_{n-p,\alpha/2}\sqrt{\text{MSE}\sqrt{1 + \mathbf{x}_0(\mathbf{U}^{-1}\mathbf{U}^{-\top})\mathbf{x}_0^\top}}.$$

Unlike the confidence interval for the mean response, this is a probability interval; the probability that the true new response lies in this interval is $1 - \alpha$.

t statistic and confidence interval for regression coefficients

From (4.6), it can also be deduced that $\hat{\beta}$ is normally distributed with mean β and variance-covariance matrix $(\mathbf{U}^{-1}\mathbf{U}^{-\top})\sigma^2$. Thus, each component of $\hat{\beta}$ can be standardized as follows:

$$\frac{\hat{\beta}_i - \beta_i}{\sigma\sqrt{(\mathbf{U}^{-1}\mathbf{U}^{-\top})_{i+1,i+1}}}$$

to become a standard normal random variable, for $i = 0, 1, 2, \dots, p - 1$.

Independence of $\mathbf{Q}_1^\top \mathbf{y}$ and $\mathbf{Q}_2^\top \mathbf{y}$ follows from the fact that $\mathbf{Q}_1^\top \mathbf{Q}_2 = 0$. Thus, $\hat{\beta}$ is independent of $\hat{\sigma}^2$. It then follows that

$$\frac{\hat{\beta}_i - \beta_i}{\hat{\sigma}\sqrt{(\mathbf{U}^{-1}\mathbf{U}^{-\top})_{i+1,i+1}}}$$

must have a t -distribution on $n - p$ degrees of freedom.

Let c_{jj} be the $j + 1$ th diagonal element of $(\mathbf{U}^\top \mathbf{U})^{-1}$. Then a $(1 - \alpha)$ confidence interval for β_j is

$$\hat{\beta}_j \pm t_{n-p,\alpha/2}\hat{\sigma}\sqrt{c_{jj}}.$$

Example - house price data

For the house price data, we can obtain these t values (one for each regression coefficient), assuming a null value of 0, in each case, as follows:

```

Uinv <- solve(U) # inverse of U
U.diag <- diag(Uinv%*%t(Uinv)) # diag extracts the diagonal from a matrix
T <- betahat/sqrt(sigma2hat*U.diag)
T

##          [,1]
## [1,]  2.5246106
## [2,]  1.8688408
## [3,]  1.6278905
## [4,]  0.3042043
## [5,]  0.6245612
## [6,]  1.4609912
## [7,] -0.1724264
## [8,] -0.4132581
## [9,] -0.5567916
## [10,]  0.8048774

```

According to the theory, each of these quantities is supposed to be t distributed on $n - p = 24 - 10 = 14$ degrees of freedom, under the null hypotheses that $\beta_j = 0$, for $j = 0, 1, \dots, 9$. We can see what this means graphically by plotting the t distribution curve and constructing a rug plot of the T statistic values as in Figure 4.1.

```
curve(dt(x, df=14), from = -3, 3)
rug(T, lwd = 2, col=2) # lwd makes the ticks a little bit wider than the default
```

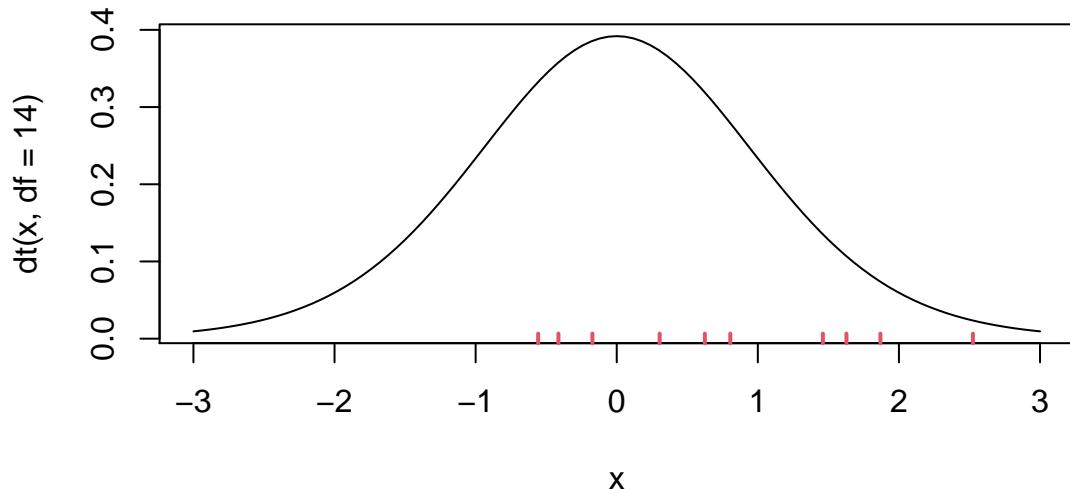


Figure 4.1: The graph of the t -distribution on 14 degrees of freedom, together with a rug plot of the observed T values coming from the house price data, for testing the hypotheses that the β_j values are all 0.

As can be seen on the figure, one of the observed T values is relatively large, relative to the support of the distribution. However, note that it is one of 10 values, and if you compare all 10 of the values, it does not seem very unusual. In any event, the conventional (not necessary the *correct*) approach is to assess the evidence against the null hypothesis that the coefficient is 0 by computing a p -value. This is the probability that a t value on 14 degrees of freedom could be that far from 0. Note that we are concerned with *absolute* distance from 0, so we compute the probability according to the graph given in Figure 4.2.

```
curve(dt(x, df=14), from = -3, 3)
abline(v=c(-1, 1)*T[1], col=c(2, 3)) # draw a vertical line through -T[1] and T[1]
```

The actual calculation of the p -value is thus

```
2*(1-pt(abs(T[1]), df=n-10)) # pt(x) = P(T < x) for a t r.v.
## [1] 0.02428304
```

To obtain the p -values for all 10 tests, use

```
2*(1-pt(abs(T), df=n-10))
```

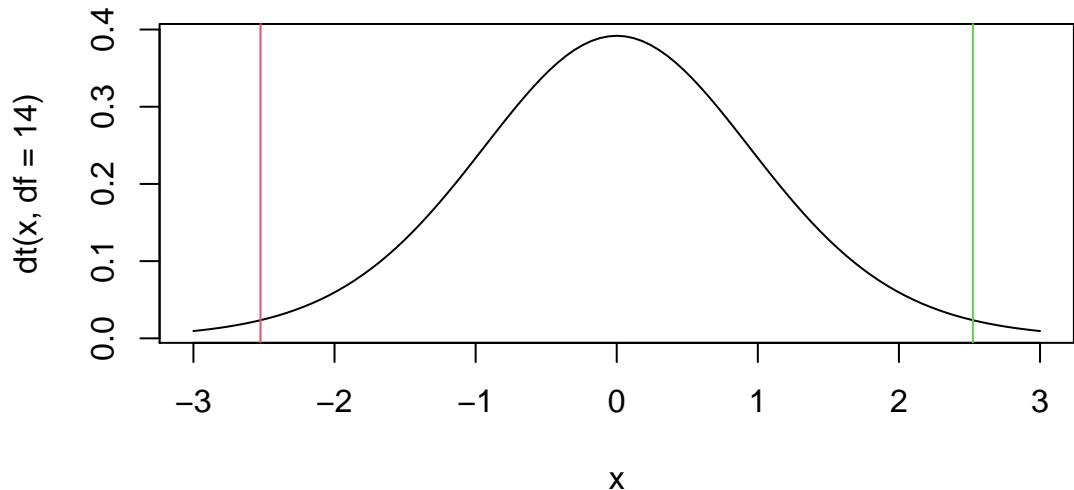


Figure 4.2: The graph of the t -distribution on 14 degrees of freedom, and the illustration of how the p -value is computed for the intercept term. The area under the curve to the left of the red line and to the right of the green line constitutes the p -value, the evidence against the null hypothesis.

```

##          [,1]
## [1,] 0.02428304
## [2,] 0.08271059
## [3,] 0.12583613
## [4,] 0.76544692
## [5,] 0.54230434
## [6,] 0.16609649
## [7,] 0.86557016
## [8,] 0.68567761
## [9,] 0.58646103
## [10,] 0.43434718

```

According to this result, we would judge the intercept to be nonzero, and we might give the coefficient of x_1 some consideration. Unfortunately, this analysis is overly simplistic, or more accurately, wrong! The t tests can be useful, but multiple testing issues render the results questionable. The next section provides a test which is more accurate, but less informative.

4.3.4 F statistics

Given $r \in \{1, \dots, p\}$, consider the (null) hypothesis that

$$\beta_{p-r} = \beta_{p-r+1} = \dots = \beta_{p-1} = 0.$$

From the QR decomposition of \mathbf{X} and model (4.2), we have, under the above hypothesis,

$$E[\mathbf{Q}_1^\top \mathbf{y} | \mathbf{X}] = \begin{bmatrix} U_{1,1}\beta_0 + U_{1,2}\beta_1 + \cdots + U_{1,p}\beta_{p-1} \\ U_{2,2}\beta_1 + \cdots + U_{2,p}\beta_{p-1} \\ \vdots \\ \vdots \\ U_{p,p}\beta_{p-1} \end{bmatrix} = \begin{bmatrix} U_{1,1}\beta_0 + U_{1,2}\beta_1 + \cdots + U_{1,p-r}\beta_{p-r-1} \\ U_{2,2}\beta_1 + \cdots + U_{2,p-r}\beta_{p-r-1} \\ \vdots \\ U_{p-r,p-r}\beta_{p-r-1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Partition the vector $\mathbf{Q}_1^\top \mathbf{y}$ as

$$\mathbf{Q}_1^\top \mathbf{y} = \begin{bmatrix} q_{11} \\ q_{12} \end{bmatrix}$$

where q_{12} denotes the last r components of $\mathbf{Q}_1^\top \mathbf{y}$. From what we have seen, q_{12} is a vector of r independent normal random variables with mean 0 and variance σ^2 , when the null hypothesis is true. Therefore,

$$\frac{q_{12}^\top q_{12}}{\sigma^2}$$

must have a χ^2 distribution on r degrees of freedom, and because $\mathbf{Q}_1^\top \mathbf{Q}_2 = 0$, q_{12} is independent of $\hat{\sigma}^2$, so

$$\frac{q_{12}^\top q_{12}/r}{\hat{\sigma}^2}$$

must have an F distribution on r and $n - p$ degrees of freedom, when the null hypothesis is true.

4.3.5 Significance of regression

The F -test for significance of regression corresponds to the case where $r = p - 1$. Then $E[q_{11} | \mathbf{X}] = U_{1,1}\beta_0$. $E[q_{12} | \mathbf{X}]$ is a $(p - 1)$ -vector of 0's under the null hypothesis and must have at least one nonzero component if the alternative hypothesis is true.

For the house price data, we can test significance of regression as follows.

```
q12Ty <- QTy[2:10] # omitting the 1st element from Q1Ty
F <- sum(q12Ty^2) / (9*sigma2hat) # r = 10-1 = 9
F

## [1] 9.037027
```

The null hypothesis is that all of the coefficients, except possibly the intercept, are zero. Figure 4.3 gives a graphical view of the evidence against the null hypothesis. The p -value is the probability that the observed F value would be as far out as the observed value, if the true distribution was actually an F on 9 and 14 degrees of freedom (the curve that was drawn).

According to the figure, it appears that there is strong evidence that at least one of the regression coefficients is nonzero. Some effort should be expended to determine which coefficient or coefficients are nonzero.

```
curve(df(x, df1=9, df2=n-10), from = 0, to= 10, ylab="F density function")
points(F, 0, pch = 4) # draw an X at the observed value of F
```

4.3.6 Obtaining the output all at once with `lm()`

The `lm()` function will take care of the coefficient estimation, variance estimation, t and F and p -value calculations in one function call. Compare the output from the following code with the pieces of output that have been provided earlier.

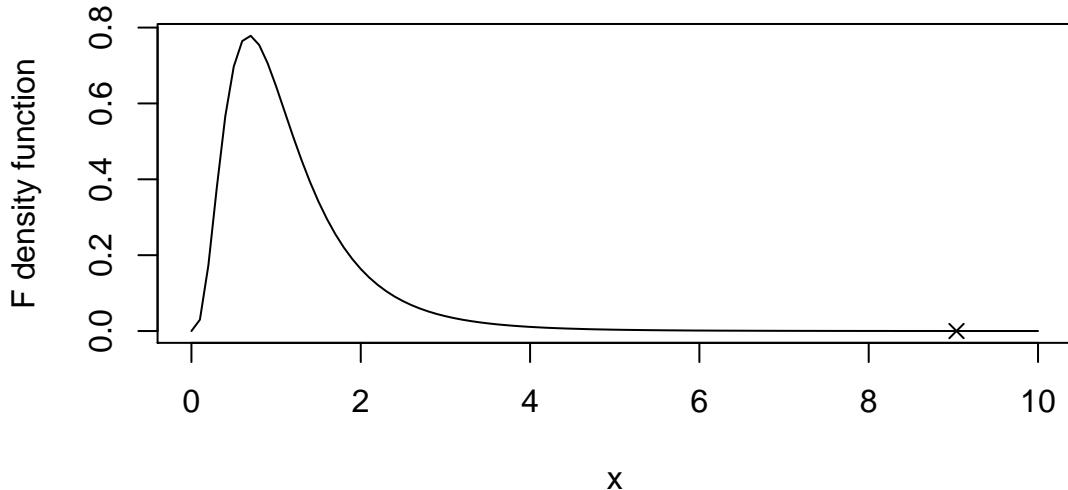


Figure 4.3: The graph of the F -distribution on 9 and 14 degrees of freedom, and the illustration of how the p -value is computed for the significance of regression test. The area under the curve to the right of the point marked with the X constitutes the p -value, the evidence against the null hypothesis

```
house.lm <- lm(y ~ ., data=table.b4)
summary(house.lm)$coefficients

##              Estimate Std. Error    t value  Pr(>|t|)
## (Intercept) 14.92764759  5.9128516 2.5246106 0.02428304
## x1          1.92472156  1.0299013 1.8688408 0.08271059
## x2          7.00053420  4.3003717 1.6278905 0.12583613
## x3          0.14917793  0.4903874 0.3042043 0.76544692
## x4          2.72280790  4.3595535 0.6245612 0.54230434
## x5          2.00668402  1.3735086 1.4609912 0.16609649
## x6         -0.41012376  2.3785444 -0.1724264 0.86557016
## x7         -1.40323530  3.3955419 -0.4132581 0.68567761
## x8         -0.03714908  0.0667199 -0.5567916 0.58646103
## x9          1.55944663  1.9374959  0.8048774 0.43434718

summary(house.lm)$fstatistic

##      value     numdf     dendf
## 9.037027 9.000000 14.000000
```

As noted above, the vertical red line on the graph pictured in Figure 4.3 passes through the calculated F value. According to the figure, the probability of an F statistic on 9 and 14 degrees of freedom exceeding this value is very small, suggesting strong evidence that one or more of the regression coefficients (apart from the intercept) is nonzero. We now are faced with the problem of determining which ones. Clearly, the t -tests are not helpful here; if they were to be believed, we would conclude only that the intercept is nonzero, but the evidence from the F -test contradicts this assertion. Another approach is needed.

4.4 Variable selection

4.4.1 Traditional methods for chasing wild geese

The models obtained from least-squares regression, involving large number of covariates, are often unnecessarily cumbersome. When there are many covariates in a model, it is natural to ask whether a model with fewer covariates would be just as effective. By removing terms from a model, is its ability to make accurate predictions hampered in any way? In fact, it is even possible to improve prediction precision by removing unnecessary model terms.

Another issue concerns the coefficient estimates themselves. There can be a tendency for coefficients to be less precisely estimated in models with large numbers of terms. Thus, model interpretation can be compromised when there is unnecessary complexity.

In the past, many variable selection techniques have been proposed. All have serious shortcomings and none is universally satisfactory. We list them here without much comment before proceeding to model-building approaches that have greater potential for success.

- Backward Selection: This method begins with the fitting of a model which includes all given covariates. The variable with the largest p -value is removed, and the model with all remaining variables is re-fit. Variables are removed sequentially in this manner, until all remaining variables have satisfactorily small p -values.
- Forward Selection: This method builds models, adding terms sequentially, beginning with the one involving the variable giving the smallest residual sum of squares. At each stage, the variable added is the one that reduces the sum of squares most.
- Stepwise Selection: This method usually starts as in Backward Selection, but variables can be added back in to the model at later stages if a large enough reduction in residual sum of squares (or some other criterion) can be achieved.
- All Subsets Regression: This method involves the computation of all possible regression models. The one which is closest to meeting a particular criterion is chosen.

A number of criteria have been devised to help the analyst decide which model is “best”. In other words, when employing any of the above methods, one iterates through a sequence of models and stops at the model which optimizes the given criterion. These criteria include Mallows’ C_p , Akaike Information Criterion (AIC), Bayes Information Criterion (BIC) and Predicted Residual Sum of Squares (PRESS). The first four of these criteria attempt to balance the requirement to minimize the residual sum of squares with the desire to keep the number of parameters in the model (i.e. the model complexity) to a minimum. The PRESS criterion is a form of cross-validation where individual observations are compared with predictions made from regression based on data sets from which those observations were deleted.

We note the *leaps* package, and the `stepAIC()` function in the *MASS* package as examples of software that have been programmed in R to implement some or all of the above techniques. The latter function is an efficient implementation of stepwise regression which identifies the best possible according to AIC. We note it because it is easy to use and can sometimes give good results.

4.4.2 Modern approaches to variable selection

LASSO

A more effective way to select variables is the LASSO (Tibshirani, 1996). This method is based on the premise that any covariate that should not be in the model should have a coefficient of 0. Thus, the method applies a penalty to the original least-squares objective function against nonzero regression coefficients. The method is actually applicable when the number of coefficients to be estimated exceeds the sample size - a situation that ordinary regression cannot handle, since a unique solution to the least-squares problem does not exist. With the LASSO, on the other hand, a unique solution can be found, because the space of possible solutions has been reduced in a certain way. This is an example of what is called regularization.

Specifically, the estimate of the $k + 1$ element vector β is chosen to minimize

$$\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda^2 \sum_{j=0}^k |\beta_j|.$$

When $\lambda = 0$, this reduces to the ordinary least-squares problem. When λ is nonzero, this is a nonlinear optimization problem for which several iterative techniques have been proposed in recent decades. The `lars` package in R gives on implementation.

A few comments are in order. First the LASSO optimization has the property that for large enough λ^2 , at least one of the coefficient estimates becomes identically 0. As λ^2 increases, it can be shown that the coefficients all shrink toward 0, and ultimately, all become identically 0. This means that the LASSO offers an effective way to select variables in a regression model, given an appropriate choice of λ .

An easier optimization problem would have been

$$\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda^2 \sum_{j=0}^k \beta_j^2.$$

In fact, this is well known to be the ridge regression problem which can be solved via a linear system of equations (and the QR decomposition). The solution does not have the variable selection feature however. As λ^2 increases, the coefficients shrink to 0, but they do not become identically 0 in the same way as for the LASSO.

A direct method, based on QR

The principal reason for the failure of the t -tests as bases for assessing strength of evidence against the hypotheses that variable coefficients are 0 is that, in general, the standard errors of the coefficient estimates, $\hat{\beta}_j$, tend to be inflated relative to what they would be if the estimates were uncorrelated with each other. As we have seen in equation (4.7), the variance-covariance matrix of the coefficient vector is $\sigma^2(\mathbf{U}^{-1}\mathbf{U}^{-\top})$. If \mathbf{U} happened to be a diagonal matrix, then the coefficient estimates would be uncorrelated with each other, and the individual t -tests could be carried out, subject to appropriate multiple testing criteria. In general \mathbf{U} is not a diagonal matrix, so the t -tests are not simply compromised by multiple testing issues, but by the intercorrelations between the estimates themselves, a phenomenon referred to as multicollinearity. This problem only gets worse as the number of terms in a regression model increases.

QQ-plots of $\mathbf{Q}_1^\top \mathbf{y}$ against $\mathbf{Q}_2^\top \mathbf{y}$

A simple and effective way around this quagmire is to focus on the elements of the vector $\mathbf{U}\beta$ instead of the elements of β . Under the assumptions that we have imposed (normality, independence, constant variance), the elements of $\mathbf{Q}_1^\top \mathbf{y}$ are independent, with mean $\mathbf{U}\beta$ and variance σ^2 , since

$$\mathbf{Q}_1^\top \mathbf{y} = \mathbf{U}\beta + \mathbf{Q}_1^\top \epsilon$$

The elements of $\mathbf{Q}_2^\top \mathbf{y}$ are also independent, but with mean 0 and variance σ^2 . Therefore, all elements of $\mathbf{Q}_1^\top \mathbf{y}$ corresponding to elements of $\mathbf{U}\beta$ which are 0 will have the same distribution as the elements of $\mathbf{Q}_2^\top \mathbf{y}$. Other elements of $\mathbf{Q}_1^\top \mathbf{y}$ will necessarily follow a different distribution (i.e. having a different mean). Thus, outliers on a QQ-plot of $\mathbf{Q}_1^\top \mathbf{y}$ against $\mathbf{Q}_2^\top \mathbf{y}$ should clearly identify elements of $\mathbf{U}\beta$ which are nonzero. We demonstrate the technique on simulated data first.

The following function similarly calculates the elements of $\mathbf{Q}_1^\top \mathbf{y}$. The name is suggestive of the fact that the resulting vector is an estimate of $\mathbf{U}\beta$.

```
Ubetahat <- function(lm.object, y) {
  Ub <- t(qr.Q(lm.object$qr)) %*% y
  Ub
}
```

The following function calculates the elements of $\mathbf{Q}_2^T \mathbf{y}$, using the QR decomposition calculated in a call to the `lm()` function.

```
Qresid <- function(lm.object, y) {
  p <- summary(lm.object)$df[1]
  (t(qr.Q(lm.object$qr, complete = TRUE)) %*% y)[-c(1:p)]
}
```

Simulated data in a data frame called `xy` are generated as follows, using independent noise which is normally distributed with standard deviation 2.5. The model intercept is 0.1, and there are 7 randomly generated covariates, 2 of which actually have an effect on the response.

```
set.seed(162834662) # included so the reader can reproduce results exactly
n <- 200
beta <- c(.1, 0, .3, 0, 0, .5, 0, 0)
k <- length(beta)-1
xy <- data.frame(matrix(rnorm(k*n, sd = 2.5), nrow=n))
names(xy) <- paste("x", 1:k, sep="")
y <- as.matrix(xy) %*% beta[-1] + beta[1] + rnorm(n)
xy$y <- as.vector(y); rm(y)
```

The full model is fit to the data using the `lm()` function:

```
xy.lm <- lm(y ~ ., data=xy)
```

The vectors $\mathbf{Q}_1^T \mathbf{y}$ and $\mathbf{Q}_2^T \mathbf{y}$ are then calculated:

```
Ub <- Ubetahat(xy.lm, xy$y)
Ub

##          [,1]
## [1,] -4.53258214
## [2,]  5.36925668
## [3,]  9.43498612
## [4,] -1.23340236
## [5,]  1.54243800
## [6,] 15.33245333
## [7,] -0.02981546
## [8,]  0.50199577

Qres <- Qresid(xy.lm, xy$y)
```

Figure 4.4 displays the resulting QQ-plot. The 45° reference line has been replaced with a plot of the elements of $\mathbf{Q}_2^T \mathbf{y}$ against themselves as a visual check on the appropriateness of the assumptions on the error. Of course, if there had been no regression effects, the plotting characters corresponding to the elements of $\mathbf{U}\widehat{\beta}$ would line up with the grey circles on the reference line. In fact, several do not, indicating strong evidence of a departure from the assumption that the elements of $\mathbf{U}\beta$ are all 0. In particular, note the effect outliers near -5, 5, 10 and 15.

```
par(mar=c(4, 4, .1, .1))
qqplot(Ub, Qres, pch = 16, ylab = "residuals", xlab = "effects")
points(Qres, Qres, col="grey")
```

To see what the plot looks like when only the intercept differs from 0, we conduct another simulation, this time, using 0 coefficients for all 7 covariates.

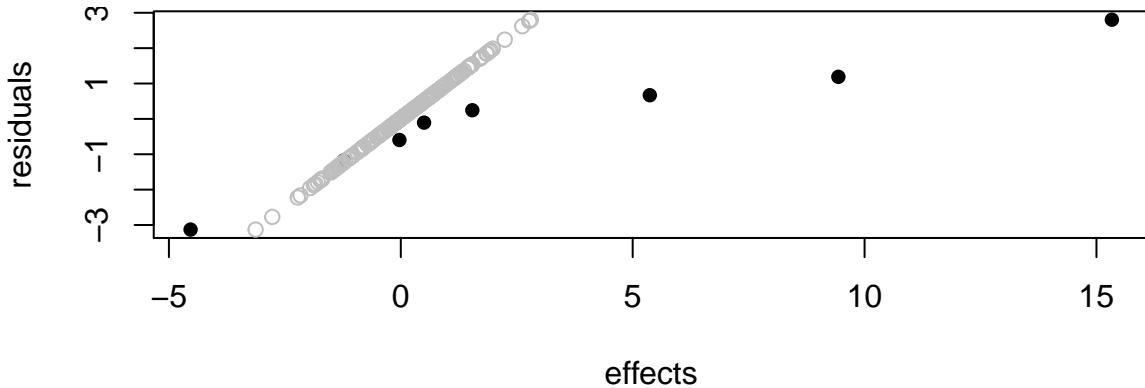


Figure 4.4: QQ-plot of estimates of $\mathbf{U}\beta$ and \mathbf{Q}_2 transformed residuals, $\mathbf{Q}_2^T\mathbf{y}$ for simulated data.

```
beta <- c(.2, rep(0, 7))
xy <- data.frame(matrix(rnorm(k*n, sd = 2.5), nrow=n))
names(xy) <- paste("x", 1:k, sep="")
y <- as.matrix(xy) %*% beta[-1] + beta[1] + rnorm(n)
xy$y <- as.vector(y); rm(y)
xy.lm <- lm(y ~ ., data = xy)
Ub <- Ubetahat(xy.lm, xy$y)
Qres <- Qresid(xy.lm, xy$y)
```

The result is displayed in Figure 4.5. There is only one effect outlier this time; it likely corresponds to the nonzero intercept; the rest of the solid black dots are within the range of the error distribution.

```
par(mar=c(4, 4, .1, .1))
qqplot(Ub, Qres, pch = 16, ylab = "residuals", xlab = "effects")
points(Qres, Qres, col="grey")
```

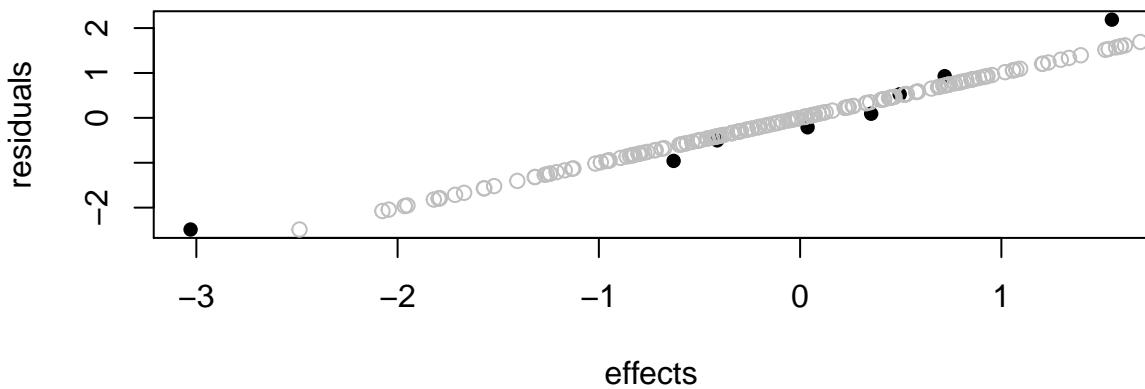


Figure 4.5: QQ-plot of estimates of $\mathbf{U}\beta$ and \mathbf{Q}_2 transformed residuals, $\mathbf{Q}_2^T\mathbf{y}$ for simulated data for a model with no effects.

The QQ-plot for the house price data set is pictured in Figure 4.6. In this case, all but two of the effects lie very close to the reference line of $\mathbf{Q}_2^T\mathbf{y}$ residuals. The extreme one on the left is likely due to the intercept term, but the one on the far right relates to the covariates and is responsible for the significant F value reported earlier.

```
Ub <- Ubetahat(house.lm, table.b4$y)
Qres <- Qresid(house.lm, table.b4$y)
par(mar=c(4, 4, .1, .1))
qqplot(Ub, Qres, pch = 16, ylab = "residuals", xlab = "effects")
points(Qres, Qres, col="grey")
```

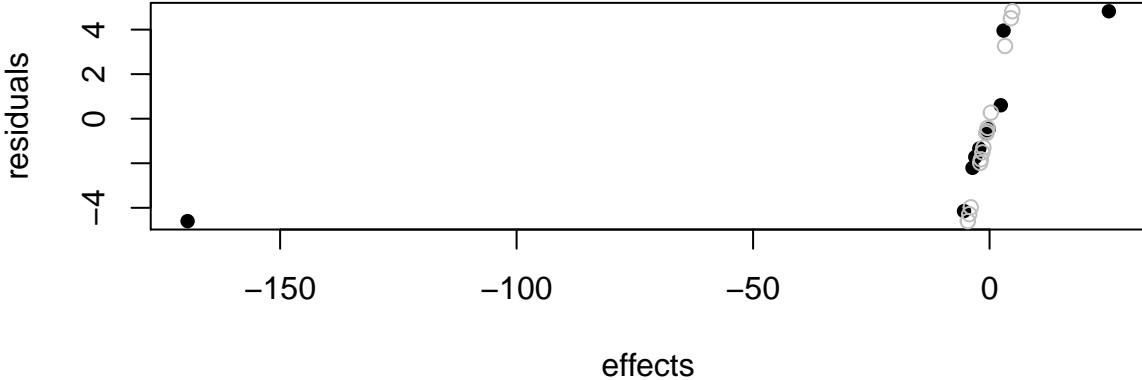


Figure 4.6: QQ-plot of estimates of $\mathbf{U}\beta$ and \mathbf{Q}_2 transformed residuals, $\mathbf{Q}_2^T\mathbf{y}$ for the house price data.

A QR pre-test estimate for β

The QQ-plot provides an informal test for which elements of $\mathbf{U}\beta$ are nonzero, but it suggests a more formal procedure which can be used as a preliminary test prior to computing improved estimates of β .

The essence of the idea is related to how one might construct a control chart. The $\mathbf{Q}_2^T\mathbf{y}$ values are “in control”, and some or all of the $\mathbf{Q}_1^T\mathbf{y}$ values might be as well. A value of $\mathbf{Q}_1^T\mathbf{y}$ might be said to be out of control if it lies beyond the .995 or the .005 quantile of the distribution of $\mathbf{Q}_2^T\mathbf{y}$. Borrowing a symmetrizing idea from Meloche (1991), since our normal assumption suggests the distribution of $\mathbf{Q}_2^T\mathbf{y}$ is symmetric, we simply use the 99th percentile of the absolute values of $\mathbf{Q}_2^T\mathbf{y}$ as the cut off value.

We keep any values of $\widehat{\mathbf{U}\beta}$ whose absolute value exceeds the cut off value. All other elements of $\widehat{\mathbf{U}\beta}$ are set to 0. We then premultiply by \mathbf{U}^{-1} to obtain our improved estimate of β . (Of course, we don’t actually calculate the inverse of \mathbf{U} ; rather, we use backward substitution to solve the upper triangular system of equations.)

Here is what happens when we apply the technique to the house price data:

```
cutoff <- quantile(abs(Qres), .99)
indices <- (abs(Ub) > cutoff)
UbKeep <- Ub * indices
betahat <- as.vector(backsolve(qr.R(house.lm$qr), UbKeep))
names(betahat) <- names(coef(house.lm))
betahat

## (Intercept)           x1           x2           x3           x4
## 10.041765   2.713399   6.164266  0.0000000  0.0000000
##           x5           x6           x7           x8           x9
## 0.0000000  0.0000000  0.0000000  0.0000000  0.0000000
```

Finally, we have a simpler model for house price:

$$\hat{y} = 10.04 + 2.71x_1 + 6.16x_2.$$

It is interesting to compare with the output from `stepAIC()` which is in the MASS package.

```
library(MASS)
```

The following code can be used to reshape the output so that it matches the format obtained from the QR approach:

```
house.AIC <- stepAIC(house.lm, trace = 0)
tmp <- coef(house.AIC)
houseAIC <- numeric(length(beta))
houseAIC[(names(beta) %in% names(tmp))] <- tmp; rm(tmp)
names(houseAIC) <- names(beta)
houseAIC

## (Intercept)          x1          x2          x3          x4
## 13.509056   2.419159   8.480250   0.000000   0.000000
##           x5          x6          x7          x8          x9
## 2.000586   0.000000  -2.182251   0.000000   0.000000
```

It is well known that the AIC can lead to models with too much complexity. This might be an example of such a situation.

Revisiting the simulated data

The following function can be used to convert the ordinary least-squares coefficient estimates to the QR-based pre-test estimates discussed in the foregoing. Note that the function contains

```
beta <- function(lm.object, y, plotit = TRUE) {
  Ub <- Ubeta(lm.object, y)
  Qres <- Qresid(lm.object, y)
  cutoff <- quantile(abs(Qres), .99)
  indices <- (abs(Ub) > cutoff)
  UbKeep <- Ub * indices
  beta <- as.vector(backsolve(qr.R(lm.object$qr), UbKeep))
  names(beta) <- names(coef(lm.object))
  if (plotit == TRUE) {
    qqplot(Ub, Qres, pch = 16, ylab = "residuals", xlab = "effects")
    points(Qres, Qres, col = "grey")
  }
  beta
}
```

The simulated data and `lm()` object can be reproduced easily, since we set the seed for the random number generator:

```
set.seed(162834662) #
n <- 200; beta <- c(.1, 0, .3, 0, 0, .5, 0, 0)
k <- length(beta)-1
xy <- data.frame(matrix(rnorm(k*n, sd = 2.5), nrow=n))
names(xy) <- paste("x", 1:k, sep="")
y <- as.matrix(xy) %*% beta[-1] + beta[1] + rnorm(n)
xy$y <- as.vector(y); rm(y)
xy.lm <- lm(y ~ ., data=xy)
```

Evaluating the QR pre-test estimator for β proceeds as follows:

```
betahat(xy.lm, xy$y, plotit = FALSE)

## (Intercept)           x1           x2           x3           x4
## 0.24994121 -0.08826018  0.27881458 -0.04010039 -0.02086968
##           x5           x6           x7
## 0.49033107  0.00000000  0.00000000
```

Stepwise regression with AIC gives:

```
xy.AIC <- stepAIC(xy.lm, trace = 0)
coef(xy.AIC)

## (Intercept)           x1           x2           x5
## 0.2500939 -0.0900277  0.2814556  0.4909749
```

Turning to the second simulated data set, we have

```
beta <- c(.2, rep(0, 7))
xy <- data.frame(matrix(rnorm(k*n, sd = 2.5), nrow=n))
names(xy) <- paste("x", 1:k, sep="")
y <- as.matrix(xy) %*% beta[-1] + beta[1] + rnorm(n)
xy$y <- as.vector(y); rm(y)
xy.lm <- lm(y ~ ., data = xy)
betahat(xy.lm, xy$y, plotit = FALSE)

## (Intercept)           x1           x2           x3           x4
## 0.2141524  0.0000000  0.0000000  0.0000000  0.0000000
##           x5           x6           x7
## 0.0000000  0.0000000  0.0000000
```

Stepwise regression with AIC gives:

```
xy.AIC <- stepAIC(xy.lm, trace = 0)
coef(xy.AIC)

## (Intercept)           x1
## 0.2276421 -0.0432856
```

Thus, the stepwise AIC approach appears to give a somewhat better answer for the first simulated data set, while in the second case, the QR approach identifies the correct model, where the stepwise AIC approach is incorrect.

Reordering columns of X to improve the QR pre-test estimator

Return to the first simulated data set one more time.

We compute the least-squares estimates of β and re-order the columns of the model matrix so that the least “significant” terms come after the more “significant” terms. Then apply the QR pre-test estimator to the re-ordered data.

```
xy.lm <- lm(y ~ ., data=xy)
coefOrder <- order(summary(xy.lm)$coefficients[-1,4], decreasing = FALSE)
betaSort <- c(beta[1], (beta[-1])[coefOrder])
xySort <- (xy[, -(k+1)])[, coefOrder]
xySort$y <- xy$y
xySort.lm <- lm(y ~ ., data=xySort)
betaQR <- betahat(xySort.lm, xy$y, plotit=FALSE)
betaQR
```

```

## (Intercept)          x5          x2          x1          x4
## 0.2500939   0.4909749   0.2814556 -0.0900277   0.0000000
##           x7          x3          x6
## 0.0000000   0.0000000   0.0000000

```

The result is still not correct, but at least it matches the stepwise AIC result very closely.

Turning to the second simulated data set, we have

```

## (Intercept)          x1          x4          x2          x7
## 0.2141524   0.0000000   0.0000000   0.0000000   0.0000000
##           x3          x6          x5
## 0.0000000   0.0000000   0.0000000

```

The re-ordering did not lead to any deterioration in our previously obtained result.

A simulation comparison of AIC and QR

In order to compare the QR pre-test estimator with the ordinary least-squares estimator and the stepwise AIC estimator, we can use simulation. The model is set up in a flexible manner, using random Poisson numbers, multiplied by random uniform numbers to obtain a set of coefficients, some of which are identically 0. In this case, $k = 15$ covariates have been simulated. Samples of size 200 are used in this example.

The true values of the β coefficients are listed below for this simulation run.

```

set.seed(12346566)
n <- 200
k <- 15
lambda <- 0.5
maximum <- 4
minimum <- -2
beta <- runif(k+1, min = minimum, max = maximum) * rpois(k+1, lambda = lambda)
beta

## [1] -0.98606368  0.00241446  0.00000000  0.29051481  0.00000000
## [6]  0.00000000  0.00000000  0.00000000  0.00000000  3.04563647
## [11]  0.00000000  3.77338520  0.00000000  0.00000000  2.02010442
## [16]  3.77447522

```

In order to compare the three approaches, we simulate 500 samples using the β coefficients listed above, and with standard normal noise. We then calculate the three types of β estimates. The squared differences between the β estimates and the true value of β are then calculated and averaged. All 500 of these mean squared errors are stored, and then finally averaged, using the `apply()` function.

```

MSEs <- NULL
for (i in 1:500) {
  xy <- data.frame(matrix(rnorm(k*n), nrow=n))
  names(xy) <- paste("x", 1:k, sep="")
  y <- as.matrix(xy) %*% beta[-1] + beta[1] + rnorm(n)
  xy$y <- as.vector(y); rm(y)
  xy.lm <- lm(y ~ ., data=xy)
  coefOrder <- order(summary(xy.lm)$coefficients[-1,4], decreasing = FALSE)
  betaSort <- c(beta[1], (beta[-1])[coefOrder])
  xySort <- (xy[,-(k+1)])[, coefOrder]
  xySort$y <- xy$y
  xySort.lm <- lm(y ~ ., data=xySort)
}
```

```

betaQR <- betahat(xySort.lm, xy$y, plotit=FALSE)
betalm <- coef(xy.lm)
AIC.lm <- stepAIC(xy.lm, trace = 0)
tmp <- coef(AIC.lm)
betaAIC <- numeric(length(betalm))
betaAIC[names(betalm) %in% names(tmp)] <- tmp; rm(tmp)
MSE <- c("lm" = mean((beta-betalm)^2),
        "AIC" = mean((beta-betaAIC)^2),
        "QR" = mean((betaSort-betaQR)^2))
MSEs <- rbind(MSEs, MSE)
}
apply(MSEs, 2, mean)

##           lm          AIC          QR
## 0.005291553 0.003876853 0.002628537

```

The results for this simulation demonstrate quite clearly that the ordinary least-squares estimates are not as accurate as the stepwise AIC or the QR pre-test estimates. In fact, we see that the QR pre-tests are, in fact, the most accurate.

4.5 Exercises

1. Find the model matrix \mathbf{X} for each of the following.
 - (a) $Y_i = \mu + \varepsilon_i$
 - (b) $Y_{ij} = \mu_i + \varepsilon_{ij}, i = 1, 2, j = 1, 2, 3.$
 [Hint: $\beta = [\mu_1 \mu_2]^\top$ and $\mathbf{y} = [Y_{11} Y_{12} \cdots Y_{23}]^\top.$]
 (This is an example of a 1-way ANOVA model.)
 - (c) $Y_{ij} = \mu_i + \beta x_{ij} + \varepsilon_{ij}, i = 1, 2, j = 1, 2, 3.$
 (This is an example of an analysis of covariance model.)
 - (d) $Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_{12} x_{i1} x_{i2} + \varepsilon_i$
 - (e) $Y_i = \beta_0 + \beta_1 \cos(x_i) + \beta_2 \sin(x_i) + \varepsilon_i$
 - (f) $Y_i = \beta_1 B_1(x_i) + \beta_2 B_2(x_i) + \beta_3 B_3(x_i) + \beta_4 B_4(x_i)$ where B_1, B_2, B_3 and B_4 are given real-valued functions.
2. Obtain the matrices for the QR decomposition of the matrix $[1 \ 0 \ 1 \ 0]^\top.$ Do this exercise without a computer at first, and then check your result using the `qr` functions in R.
3. Suppose \mathbf{Q} is an $n \times n$ orthogonal matrix and ε is a vector n independent standard normal random variables. Suppose also, that the matrix composed of the first k columns of \mathbf{Q} is denoted by \mathbf{Q}_1 and the matrix consisting of the remaining columns is denoted by $\mathbf{Q}_2.$
 - (a) Identify the distributions of the following random variables:
 - i. $\mathbf{Q}^\top \varepsilon$
 - ii. $Y_1 = \mathbf{Q}_1^\top \varepsilon$
 - iii. $Y_2 = \mathbf{Q}_2^\top \varepsilon$
 - iv. $x_1 = Y_1^\top Y_1$
 - v. $x_2 = Y_2^\top Y_2$
 - (b) As defined in the previous question, are x_1 and x_2 independent random variables? Explain briefly.

- (c) Identify the distribution of

$$\frac{(n-k)x_1}{kx_2}.$$

4. Do the following exercise without using R.

Suppose $(3/2, 1), (3/2, 3), (1/2, 5), (1/2, 6)$ are 4 independently observed data points, and a simple regression model (with intercept) is to be fit to these data.

- (a) Write down the design matrix \mathbf{X} .
 - (b) Determine the QR decomposition for \mathbf{X} .¹
 - (c) Determine the slope and intercept estimates, using the QR decomposition.
 - (d) Determine the residual sum of squares, using the QR decomposition.
 - (e) Estimate the error variance.
 - (f) Calculate the test statistic used to determine whether the slope is 0 or not
 - (g) What is the distribution of the test statistic that you calculated in the previous question, if the null hypothesis is true? Do you have strong evidence against the null hypothesis?
5. Use R for this question, but do not use the `lm()` function.
- Suppose $(0, 1), (1, 3), (2, 5), (3, 6), (4, 5), (5, 7)$ are 6 independently observed data points, and a simple regression model (with intercept) is to be fit to these data.
- (a) Write down the design matrix \mathbf{X} .
 - (b) Determine the QR decomposition for \mathbf{X} .
 - (c) Calculate \mathbf{U}^{-1} .
 - (d) Determine the slope and intercept estimates, using the QR decomposition.
 - (e) Determine the residual sum of squares, using the QR decomposition.
 - (f) Estimate the error variance.
 - (g) Calculate the test statistic used to determine whether the slope is 0 or not.
 - (h) What is the distribution of the test statistic that you calculated in the previous question, if the null hypothesis is true? Do you have strong evidence against the null hypothesis?
6. Compare the results of the previous two questions with what you obtain when you use the `lm()` function.
7. Consider the multiple regression model

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

where \mathbf{X} is $n \times (k+1)$ and its first column consists only of 1's, that is, the n -vector $\mathbf{1}$, and ϵ is normally distributed with mean vector $\mathbf{0}$ and variance-covariance matrix $\sigma^2 \mathbf{I}$. Let \mathbf{H} denote the corresponding influence matrix.

- (a) By going through the initial steps of the QR decomposition procedure, and noting carefully how the first column of \mathbf{Q} , denoted by \mathbf{Q}_{11} , is constructed, show that $\mathbf{Q}_{11} = \mathbf{1}/\sqrt{n}$.
- (b) Deduce that $(\mathbf{H} - \mathbf{1}\mathbf{1}^\top/n) = (\mathbf{Q}_1\mathbf{Q}_1^\top - \mathbf{Q}_{11}\mathbf{Q}_{11}^\top) = \mathbf{Q}_{12}\mathbf{Q}_{12}^\top$, where \mathbf{Q}_1 and \mathbf{Q}_{12} are $n \times (k+1)$ and $n \times k$ matrices defined through the partition of $\mathbf{Q} = [\mathbf{Q}_1 \ \mathbf{Q}_2] = [\mathbf{Q}_{11} \ \mathbf{Q}_{12} \ \mathbf{Q}_2]$.
- (c) Conclude that when $\beta_1 = \beta_2 = \dots = \beta_k = 0$, $\text{SSR}/\sigma^2 = \mathbf{y}^\top(\mathbf{H} - \mathbf{1}\mathbf{1}^\top/n)\mathbf{y}/\sigma^2$ follows a χ^2 distribution on k degrees of freedom, and $(\text{SSR}/k)/\text{SSE}/(n-k-1)$ must have an F distribution on k and $n-k-1$ degrees of freedom.

¹ $v = [-1 \ 1 \ 1 \ 1]^\top$; only one iteration is necessary.

8. Consider the model

$$Y_i = \beta^{\varepsilon_i} x_i$$

for $i = 1, 2, \dots, n$. The ε 's are independent standard normal random variables. β is assumed to be a positive, but unknown constant. The x_i 's are also assumed to be positive.

- (a) Show that the least-squares estimator for β is given by

$$\hat{\beta} = \exp \left[\sqrt{\frac{\sum_{i=1}^n (\log(Y_i) - \log(x_i))^2}{n}} \right].$$

- (b) Identify the distribution of

$$n \left(\frac{\log(\hat{\beta})}{\log(\beta)} \right)^2.$$

- (c) Assuming correctness of the model otherwise, how would you solve this problem if some of the x_i 's were negative?

9. The following matrices contain information about the QR decomposition of a model matrix \mathbf{X} , together with data on a set of responses y :

```
# 15*Q:
10   -5   -5    5    5    5
 1    13   -5    5    2   -1
 7     1   10    5   -1   -7
 -7   -1    5   10    1    7
 -1    2    5   -5   13    1
  5    5    5   -5   -5   10
```

```
# U:
-2  -9   0
 0   4   2
 0   0   1
```

```
# y:
 0   5  10   6  11  16
```

- (a) Use the above information to reconstruct the \mathbf{X} matrix.
(b) Estimate the coefficients in the model

$$Y = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon.$$

- (c) Estimate the error variance for the above model.
(d) Estimate the variance-covariance matrix of the vector $[\hat{\beta}_1 \ \hat{\beta}_2 \ \hat{\beta}_3]^\top$, and estimate the standard error of the coefficient of x_2 .

10. Consider the data set

x	y
1	1.5
2	2.5
3	1.5
4	2.5

5	3.0
6	4.5
7	5.0
8	5.0
9	4.5
10	5.5

and the model $y = \beta x + \varepsilon$. Without using the `lm()` or the `qr()` functions in R, answer the following questions:

- (a) Write down the model matrix \mathbf{X} .
 - (b) Obtain the QR decomposition for \mathbf{X} .
 - (c) Use the QR decomposition to estimate β .
 - (d) Estimate the error variance.
 - (e) Estimate the standard error of $\hat{\beta}$.
 - (f) Calculate a 95% confidence interval for β .
 - (g) Calculate a 95% prediction interval for a future observation y when $x = 9.5$.
11. Refer to the previous question. Use the `qr` functions in R to assist in fitting the model

$$y = \beta_0 + \beta_1 x + \varepsilon$$

to the data of the previous exercise.

12. Show that the influence matrix can also be expressed as $\mathbf{H} = \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$.
13. Suppose $(0, 1), (1, 3), (2, 5), (3, 6), (4, 5), (5, 3), (6, 2), (7, 5), (8, 7), (9, 9)$ are 10 independently observed data points, and the model

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \varepsilon$$

is under consideration. Use the `qr` functions in R to answer the following questions.

- (a) Write down the design matrix \mathbf{X} .
 - (b) Determine the QR decomposition for \mathbf{X} .
 - (c) Determine the residual sum of squares, using the QR decomposition.
 - (d) Estimate the error variance.
 - (e) Is there evidence that any of β_1, β_2 or β_3 is nonzero? Use the F -test to check.
 - (f) Calculate the test statistic used to determine whether either of β_2 and β_3 is nonzero.
 - (g) What is the distribution of the test statistic that you calculated in the previous question, if the null hypothesis is true? Do you have strong evidence against the null hypothesis?
14. Consider the `p4.18` data set in the *MPV* package. This data frame has 13 observations on an experiment to produce a synthetic analogue to jojoba oil. The response y represents yield, while the three predictors, x_1 , x_2 and x_3 , represent measurements of temperature, catalyst and pressure, respectively. Fit the regression model:

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon.$$

- (a) On the basis of the individual coefficient t -tests, what would you conclude about this model?
- (b) What does the overall F -test tell you?
- (c) Consider all variables as potential covariates now. Use `stepAIC()` and the QR pre-test estimator to select models to predict yield.

15. Consider the data in `table.b1` of the *MPV* package which contains 20 observations on 1976 National Football League (NFL) performance. The response variable is the number of games won in a 14 game season, and the predictor variables are rushing yards, passing yards, punting average (yards/punt), field Goal Percentage (FGs made/FGs attempted), turnover differential (turnovers acquired - turnovers lost), penalty yards, percent rushing (rushing plays/total plays), opponents' rushing yards, and opponents' passing yards.
- Fit a regression model relating the response to all covariates using the `lm()` function.
 - On the basis of the individual coefficient *t*-tests, what would you conclude about this model?
 - What does the overall *F*-test tell you?
 - Obtain a plot of the residuals versus the fitted values. Are there any outliers? If so, does this affect your conclusions above?
 - Consider all variables as potential covariates now. Use `stepAIC()` and the QR pre-test estimator to select models to predict number of games won.

16. To see what can happen when there is no relation between the response and the predictors, simulate 40 observations from a standard normal distribution and take them as responses, and simulate nine predictor variables which are correlated amongst themselves but not with the response variable. The following R code can be used:

```
set.seed(453371)
Z <- matrix(rnorm(400), ncol=10)
A <- matrix(rnorm(81), ncol=9)
simdata <- data.frame(Z[,1], Z[,-1] %*% A)
names(simdata) <- c("y", paste("x", 1:9, sep=""))
```

- Fit a regression model relating the response to all covariates using the `lm()` function.
 - On the basis of the individual coefficient *t*-tests, what would you conclude about this model?
 - What does the overall *F*-test tell you?
17. Consider the gas mileage data in `table.b3` of the *MPV* package.
- Fit a multiple regression model to estimate mean gas mileage y for cars with x_7 number of transmission speeds and having weight x_{10} .
 - Assess the model using the residual plot.
 - Use the model to estimate mean gas mileage for cars having weight 5000 pounds and 4 transmission speeds. Use a 95% confidence interval.
 - Use the *F*-test for significance of regression to decide if any of the coefficients for your fitted model are nonzero.
 - Use another *F*-test to decide if variables x_1, x_2, x_4 or x_5 should be added into the model you have already developed.
 - Consider all variables as potential covariates now. Use `stepAIC()` and the QR pre-test estimator to select models to predict gas mileage. (For simplicity, you may assume cases 23 and 25 are missing completely at random, and delete them from the data frame before carrying out the analysis.)

18. (a) Making the usual assumptions, find an expression for the estimator of β in the model

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

under the condition that $\mathbf{C}\beta = 0$ for some full rank $m \times p$ matrix \mathbf{C} . Use Lagrange multipliers.

- (b) Now, suppose δ is a given positive scalar. Find an expression for the estimator of β using the *penalized least-squares criterion*

$$\|y - \mathbf{X}\beta\|^2 + \delta\|\beta\|^2.$$

19. Consider the regression through the origin model

$$y_i = \beta x_i + \varepsilon_i$$

for $i = 1, 2, \dots, n$. The ε 's are independent random variables with density function

$$f(\varepsilon) = \frac{1}{2}e^{-|\varepsilon|}.$$

It can be shown that the maximum likelihood estimator for β is the minimizer of the sum of absolute deviations, i.e. by

$$\sum_{i=1}^n |y_i - \beta x_i|.$$

Now, consider a penalty term on the least absolute deviation problem:

$$\text{Minimize } \sum_{i=1}^n |y_i - \beta x_i| + \lambda |\beta|$$

with respect to β , where λ is a positive constant. We know that in the lasso (i.e. with least-squares instead of least-absolute-deviations), when λ is large enough, the minimizer is $\beta = 0$. Show that the same is true here, by answering the following questions:

- (a) Show that if $\beta = 0$, then the objective function value is $\sum_{i=1}^n |y_i|$.
(b) Use the fact that

$$|y_i - \beta x_i| \geq |y_i| - |\beta||x_i|$$

to show that

$$\sum_{i=1}^n |y_i - \beta x_i| + \lambda |\beta| \geq \sum_{i=1}^n |y_i| + (\lambda - \sum_{i=1}^n |x_i|)|\beta|.$$

- (c) Conclude that if $\lambda > \sum_{i=1}^n |x_i|$, then the minimizer is $\beta = 0$.

20. Referring to the previous exercise, consider the ridge regression version of the least absolute deviations problem:

$$\text{Minimize } \sum_{i=1}^n |y_i - \beta x_i|^2 + \lambda |\beta|^2$$

with respect to β .

- (a) Show that the minimizer is

$$\hat{\beta} = \frac{\sum_{i=1}^n x_i y_i}{\sum_{i=1}^n x_i^2 + \lambda}$$

- (b) Show that the above estimator is biased.

- (c) Show that, under the assumption of i.i.d errors with constant variance, the variance of the estimator is

$$\frac{\sigma^2 \sum_{i=1}^n x_i^2}{(\sum_{i=1}^n x_i^2 + \lambda)^2}$$

21. Referring to the previous two exercises, suppose that there is a strong prior belief that $\beta = 1$. A possible approach is to estimate β by solving the problem:

$$\text{Minimize } \sum_{i=1}^n |y_i - \beta x_i|^2 + \lambda |\beta - 1|^2$$

(a) Show that the minimizer of this expression is

$$\hat{\beta} = \frac{\sum_{i=1}^n x_i y_i + \lambda}{\sum_{i=1}^n x_i^2 + \lambda}$$

(b) Show that this estimator is unbiased when $\beta = 1$.

(c) Show that the variance of this estimator decreases as λ increases.

5

Model Assessment and Adjustment

5.1 Regression assumptions and diagnostics

Recall that for the multiple regression model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

the following assumptions are usually made::

1. The relationship between \mathbf{y} and the predictors (columns of \mathbf{X}) is **linear**.
2. The noise vector $\boldsymbol{\epsilon}$ has **zero mean**.
3. All components of $\boldsymbol{\epsilon}$ have the **same variance** σ^2 .
4. The components of $\boldsymbol{\epsilon}$ are **uncorrelated**. In other words, the observations are uncorrelated with each other.
5. The components of $\boldsymbol{\epsilon}$ are **independent of the predictors**.
6. The components of $\boldsymbol{\epsilon}$ are **normally distributed**.¹

5.1.1 Consequences of model failure

If any of these assumptions fail, model predictions and statements about prediction or estimation uncertainty may be inaccurate. Of course, the magnitude of the inaccuracy will depend on the degree of departure from an assumption. In many situations, small errors in accuracy can be tolerated with the corresponding models being judged as adequate. In a situation where the level of inaccuracy is judged to be intolerable, the assessment would be one of model failure, and either a new model is sought, or the analyst concludes that the data are not amenable to a linear model.

Failure of the linearity assumption leads to both estimation and prediction inaccuracy; serious departures from linearity would render coefficient standard error estimates as virtually useless, since the coefficients in a linear model are virtually meaningless in the presence of nonlinearity. Fortunately, this assumption is one of the simplest to check, and it can often be remedied in relatively straightforward ways, such as adding polynomial terms to the model or by the use of a nonlinear transformation of the response variable.

On the other hand, checking that the noise vector has a mean of zero is not usually possible, unless there is additional information external to the data. The consequence of a nonzero mean is bias in the intercept term. As long as one does not over-interpret the intercept term, this bias would normally be tolerable and not of serious concern.

The constant variance assumption is important. If this assumption is violated, regression coefficients can still be estimated in an unbiased manner, and predictions will be unbiased, but the amount of uncertainty in the estimates and predictions will be larger than necessary. Furthermore, assessments of uncertainty through standard error estimates and prediction error estimates will be incorrect. Obviously, a single variance estimate such as the MSE is meaningless when it does not correspond to a single value of σ^2 . Fortunately, it is often possible to detect a changing variance, and there are techniques for remedying this situation. Weighted least-squares is one option;

¹This is not always required.

transformation of the response variable is another. In many circumstances, a changing variance is an indicator that an important variable is missing from the model, so finding and including such a variable is helpful where possible.

The requirement of uncorrelated errors is crucial to both estimation and prediction accuracy. It is possible to check this assumption in limited ways, but there are many ways in which failure of this assumption could elude detection. The best way to avoid trouble here is to use data from experiments in which the observations are made independently of each other, though this is not nearly always possible. Where departure from this assumption has been detected, it is sometimes possible to use generalized least-squares to cope with the dependence.

When the errors depend on the predictors, coefficient estimates and predictions will be biased. It is difficult to distinguish this problem from nonlinearity without additional information external to the data. Again, experiments can be designed to avoid this situation, but the situation may be unavoidable in an observational study.

Failure of the normality assumption will lead to inaccuracies in prediction error estimates, since those are calculated from the normal distribution explicitly. On the other hand, estimates and predictions will be unbiased when errors follow other distributions, provided there is limited skewness and kurtosis (i.e. heavy tails). Estimation uncertainty is also relatively robust to failure of this assumption. We have seen that the normal QQ-plot is a good way to assess the normal assumption. The Box-Cox transformation of the response variable is one way to correct for serious non-normality.

5.1.2 Validation or assessment?

The distinction between model validation and model assessment is important. The former suggests that the analyst can, on the basis of a limited amount of data, judge the model to be factually correct; this is a tall order, and usually impossible without the support of additional science. The latter term is more realistic, reflecting the notion of determining how inaccurate a model may be and whether the level of inaccuracy is tolerable.

It is the objective of this chapter to describe techniques which can be used to diagnose violations of some or all of the model assumptions. It is important to note that these techniques are mostly based on a study of the patterns in residuals, and that graphics play a useful role, and statistics such as R^2 and p-values are almost completely useless in assessing model adequacy.

5.1.3 Types of residuals

Raw residuals are the most common form of residual used in diagnosing problems. The i th raw residual is defined as

$$e_i = Y_i - \hat{\mu}_i, \quad \text{for } i = 1, 2, \dots, n.$$

There is a scale problem. How big is a large residual? In some problems, a residual of 0.01 is large and in other problems a residual of 10000 would be small. This is an important question, since outlying residuals are often a strong indicator of model failure, so we must know if a residual has a large magnitude or not. Judgement here depends on the amount of variability in the noise. For this reason, some analysts prefer to use the standardized residuals which are defined as

$$d_i = \frac{e_i}{\sqrt{\text{MSE}}}, \quad \text{for } i = 1, 2, \dots, n.$$

The variance of d_i is approximately 1, but the actual value depends on x_i . Thus, standardized residuals (and raw residuals for that matter) should not be expected to reflect a perfectly constant variance. For this reason, the studentized residuals are sometimes preferred. These are defined as

$$r_i = \frac{e_i}{\sqrt{\text{MSE}(1 - h_{ii})}}, \quad \text{for } i = 1, 2, \dots, n.$$

where h_{ii} = i th diagonal element of the influence matrix \mathbf{H} . Note that the variance-covariance matrix for the residual vector is

$$\text{Var}(e) = \text{Var}((\mathbf{I} - \mathbf{H})\mathbf{y}) = (\mathbf{I} - \mathbf{H})\sigma^2.$$

By dividing the raw residual by $\sqrt{(1 - h_{ii})}\sigma$, we obtain a quantity that has mean 0 and standard deviation 1, independent of i .

The PRESS residuals should also be mentioned. They can be calculated from

$$e_{(i)} = y_i - \hat{y}_{(i)} = \frac{e_i}{1 - h_{ii}}.$$

They are an attempt at estimating prediction error. Specifically, $\hat{y}_{(i)}$ is obtained by deleting the i th observation; fitting the model to the remaining data, and predicting at x_i . This is done for each observation. The formula above is mathematically equivalent to carrying out what would otherwise be lengthy calculations.

5.1.4 PRESS - Predicted Residual Sum of Squares

A good way of checking the ability of a model to make predictions is to omit an observation, build the model with the other observations, and to calculate the error in prediction of the omitted observation. The PRESS statistic is based on this concept, where each observation is held back from the data, in turn, and the sum of the squared prediction errors (actually, the PRESS residuals) is calculated:

$$\begin{aligned} \text{PRESS} &= \sum_{i=1}^n e_{(i)}^2 \\ &= \sum_{i=1}^n \left(\frac{e_i}{1 - h_{ii}} \right)^2 \end{aligned}$$

This indicates how well a regression model can predict new data.

Small values of PRESS are desired.

Example – litters

For the `litters` data in the *DAAG* package, we compare four possible models in terms of predictive ability, using the PRESS statistic:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon$$

$$y = \beta_1 x_1 + \beta_2 x_2 + \varepsilon$$

$$y = \beta_0 + \beta_1 x_1 + \varepsilon$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \varepsilon$$

Note that all models being compared have the same response variable but different combinations of predictors. Comparing different forms of response variable with PRESS is not appropriate, since the scales will usually be different, leading to potentially incorrect conclusions.

The R calculations to compare the four models are:

```
library(DAAG) # contains the litters data
library(MPV) # contains the PRESS function
# regression of brain weight against body weight and litter size:
litters.lm <- lm(brainwt ~ bodywt + lsize, data = litters)
PRESS(litters.lm) # same regression as above, but without the intercept term:
## [1] 0.003501323

litters.0 <- lm(brainwt ~ bodywt + lsize -1, data=litters)
PRESS(litters.0) # brain weight vs. body weight only, with intercept:
```

```

## [1] 0.004085556
litters.1 <- lm(brainwt ~ bodywt, data=litters)
PRESS(litters.1) # regression of brain weight against both variables plus
## [1] 0.003845278
# an interaction term:
litters.2 <- lm(brainwt ~ bodywt + lsize + lsize:bodywt, data=litters)
PRESS(litters.2)
## [1] 0.00370311

```

Based on the PRESS criterion, the best predictor is the 1st model.

Model diagnostics involving the PRESS statistic are also referred to as leave-one-out cross-validation or CV. In machine learning terminology, the observation that is omitted at each stage of the calculation is called the testing data, while the observations used in fitting the model are called the training data.

5.2 Residual plots

The goal of regression diagnostics is to detect departures from the assumptions. The various plots of residuals are the most important diagnostics. Plots of residuals versus fitted values or predictors are used

- * for detecting changes in variance
- * for detecting nonlinearity
- * for detecting outliers
- * for detecting dependence on a predictor.

Partial residual plots are used for checking whether variables enter the model linearly.

A time plot of the residuals can be used for detecting dependence in time. Such dependence often takes the form of autocorrelation which occurs when successive errors are related to each other through a linear model. Lag plots of the residuals and the autocorrelation function plot are effective tools here.

As we have seen, the normal QQ-plot of the residuals can be used for assessing normality.

5.2.1 Basic plots of residuals

At a minimum, the data analyst should plot residuals, preferably studentized, against the fitted values or against one or more of the predictor variables. This basic plot can be used to detect outlying observations, nonlinearity and a nonconstant variance.

Example - biochemical oxygen demand

Capability of subsurface flow wetland systems to remove biochemical oxygen demand (BioOxyDemand) and various other chemical constituents resulted in 13 observations on BioOxyDemand mass loading (x) and BioOxy-Demand mass removal (y). Interest centers on how to predict BioOxyDemand mass removal.

```
library(MPV) # contains the BioOxyDemand data
```

Calculation of the four types of residuals is as follows.

```

BioOxyDemand.lm <- lm(y ~ x, data = BioOxyDemand)
BioOxyDemand.res <- resid(BioOxyDemand.lm) # ordinary residuals
BioOxyDemand.stand <- BioOxyDemand.res/5.715 # standardized residuals
BioOxyDemand.stud <- BioOxyDemand.res/(5.715*sqrt(1-hat(
  model.matrix(BioOxyDemand.lm)))) # studentized
# PRESS residuals:
BioOxyDemand.press <- BioOxyDemand.res/(1- hat(model.matrix(BioOxyDemand.lm)))

```

The `hat()` function supplies the hat diagonal elements, h_{ii} , using the design matrix or model matrix, \mathbf{X} , extracted by the `model.matrix()` function from the `lm()` object.

Figure 5.1 displays the four kinds of residuals, and is based on a straightforward use of the `plot()` function. The first thing to observe on the figure is that the scales are different in each of the plots. The scale of the raw residuals is such that we cannot really tell if the outlying observation near -15 is problematic.

The studentized and standardized residuals have scales which are somewhat along the lines of standard normal data, with ranges between -3 and 3. Values outside the range (-2, 2) are outlying, with the seriousness of the problem corresponding to the magnitude of the value. The standardized residuals are somewhat different from the studentized residuals, and in particular, the outlier with a standardized value below -2 has a studentized value very near -3. Thus, the use of studentizing (which is more accurate) leads to an assessment of a more serious problem than the standardizing does in this example.

Note that there is an increase in the variance of the residuals as a function of x . This will be addressed later in the chapter. Another observation to make is the concentration of points at small values of x with a few observations at larger values of x . This pattern sometimes suggests that a log or square root transformation might be useful in order to reduce the influence of the extreme observations.

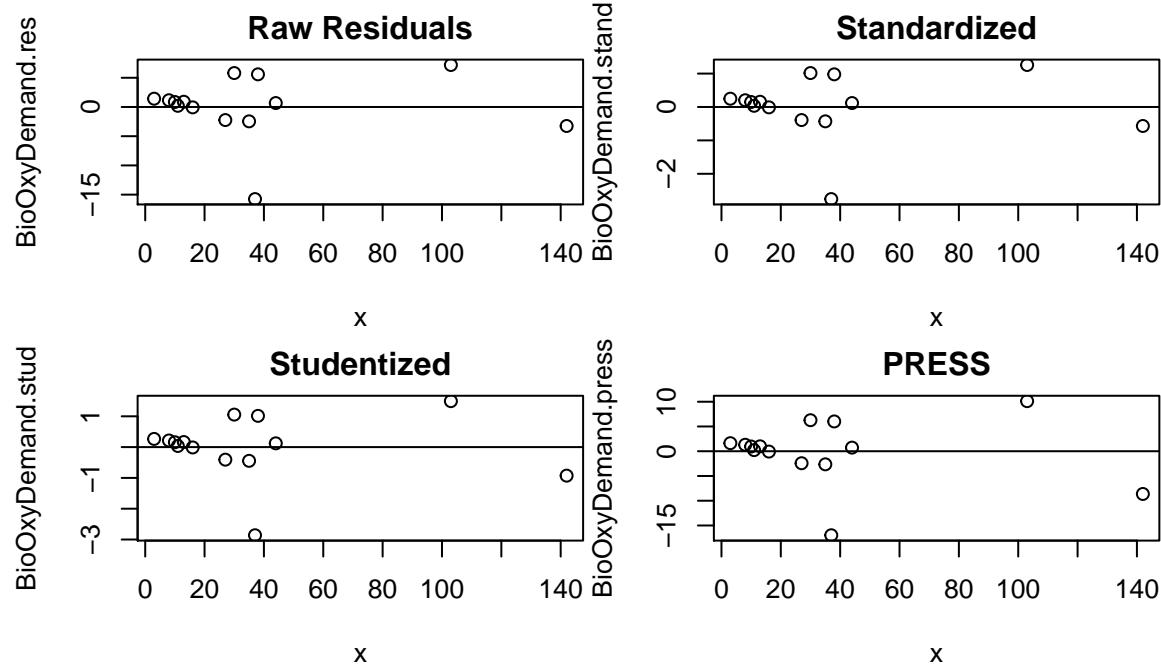


Figure 5.1: Plots of the four kinds of residuals.

We can also plot the residuals versus the fitted values using the object-oriented version of the `plot()` function:

This version of the `plot()` function detects that the object to be acted upon is an `lm()` object, and thus, the specific function `plot.lm()` is employed. The `which=1` argument specifies that the first of the default plots is

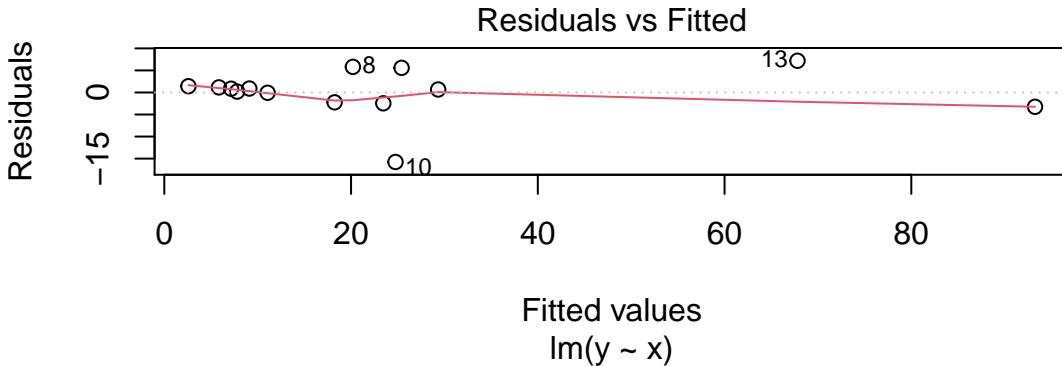


Figure 5.2: The default plot of the residuals versus fitted values.

to be drawn - the residuals versus fitted values. Other values of the `which` parameter will be considered later in this chapter.

The result of this code is displayed in Figure 5.2. Note that the raw residuals are used. A smoothing curve is overlaid, in red, to alert the viewer to any systematic strong curvature; the smoother is robust, in the sense that isolated outliers do not overly distort the curve.

5.2.2 Added variable plots or partial regression plots

Suppose observations are taken on a response variable y and three other variables x_1, x_2 and x_3 . A possible linear model formulation is

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon.$$

One should always check plots of the residuals versus the fitted values, and versus each of the predictors. An additional way to check whether each predictor should enter the regression model linearly is to look at partial regression plots for each variable. If nonlinearity is detected, the plot might indicate what kind of transformation of the variable should be considered in order to give an improved model. For example, if a parabolic shape is observed, a quadratic term might be appropriate.

Constructing a partial regression plot for x_1

- Regress y against x_2 and x_3 (i.e. all variables but x_1).
- Regress x_1 against x_2 and x_3 .
- Obtain residuals for both regressions.
- Plot the y residuals against the x_1 residuals

If x_1 enters the model linearly, you should see a points scattered about a straight line of slope β_1 . Otherwise, the plot may indicate what kind of transformation to apply to x_1 .

Some home-made code for a function called `partial.plot()` which can construct such a partial plot is as follows

```
partial.plot

## function (x, y, i = 1)
## {
##   xnames <- names(x)
##   x <- as.matrix(x)
```

```

##      yname <- names(y)
##      y <- as.matrix(y)
##      xi.lm <- lm(x[, i] ~ x[, -i])
##      xi.res <- residuals(xi.lm)
##      y.lm <- lm(y ~ x[, -i])
##      y.res <- residuals(y.lm)
##      plot(xi.res, y.res, pch = 16, xlab = xnames[i], ylab = yname)
##  }

```

Example – artificial data with known nonlinearity

Some artificial data are in `partial.data`: the first three columns are x_1, x_2 and x_3 ; the last column is y .

```
source("partial.data.R") # some artificial data
```

The true model underlying the data is

$$y = .2e^{x_1} + x_2 + x_3 + \varepsilon$$

where the standard deviation of the noise is 0.1.

We construct the three partial plots, one for x_1 , one for x_2 and a third for x_3 . The first would be expected to show nonlinearity and the other two should be linear. Figures 5.3-5.5 show the plots, and they are in agreement with this expectation. The first plot is suggestive of either a quadratic term or an exponential term, which, of course, is the correct transformation.

The code for the first plot follows; the codes for the other two plots are similar, where '2' and '3' replace '1' respectively.

```
par(mar=c(4, 4, .5, .1))
partial.plot(partial.data[,-4],partial.data[,4],1) # partial for x1;
```

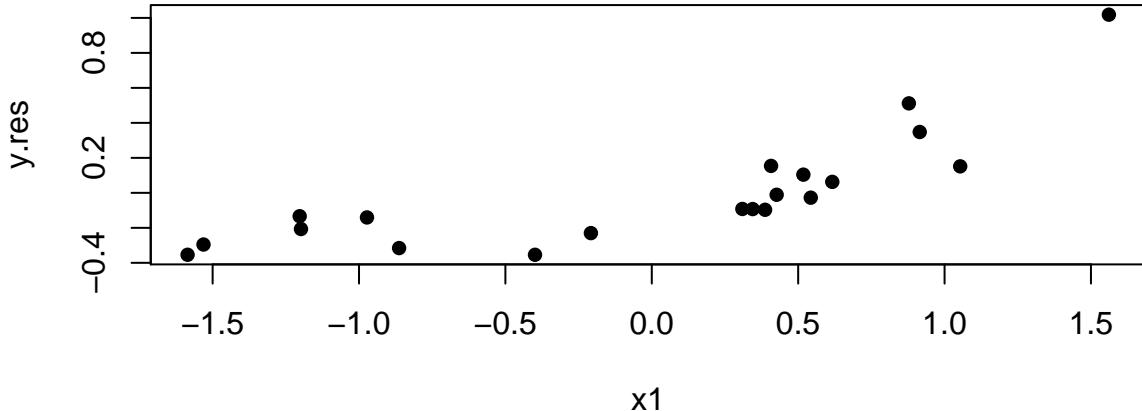


Figure 5.3: Partial residual plot for the first variable in the artificial data model.

Example – litters

Figure 5.6 shows the partial residual plots for the litter size variable and the body weight variable in the model relating brain weight to these two variables. The figure was produced using the following code:

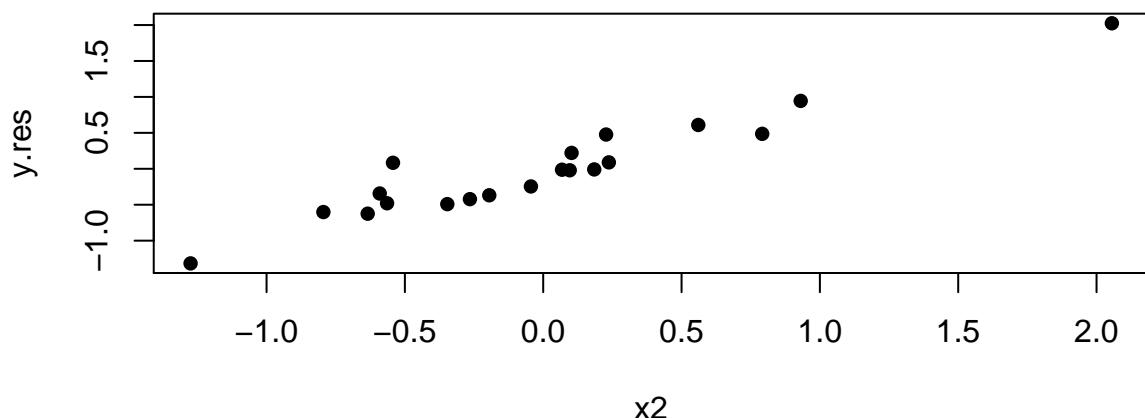


Figure 5.4: Partial residual plot for the second variable in the artificial data model.

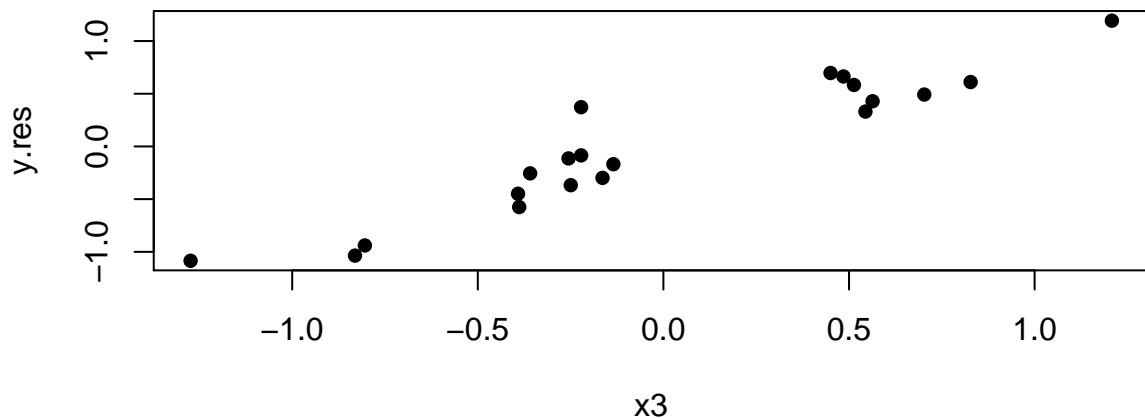


Figure 5.5: Partial residual plot for the third variable in the artificial data model.

```
par(mfrow=c(1,2), mar=c(4, 4, .5, .1))
partial.plot(litters[,-3],litters[,3],1) # partial for litter size
partial.plot(litters[,-3],litters[,3],2) # partial for body weight
```

We observe that there is mild nonlinearity which might be handled with a square root transformation of the litter size. This might be investigated more thoroughly upon collecting more data.

5.2.3 Checking the normal assumption

It is important to keep in mind that real data are not likely to be normally distributed. At best, they will be approximately so, due to central limit theorem effects. Large departures from normality might be of concern, particularly if prediction errors are being calculated on the basis of the normal distribution. Severely skewed error distributions can also lead to inaccuracies in coefficient estimates and their standard errors.

The normal QQ-plot is a straightforward way of checking for departures from the normal distribution. Applying the `plot()` function to an `lm` object with `which = 2` yields a normal QQ-plot of the residuals.

Example – litters data

The code to obtain the normal QQ-plot of the residuals for the model relating brain weight to litter size and body weight in the `litters` data is as follows.

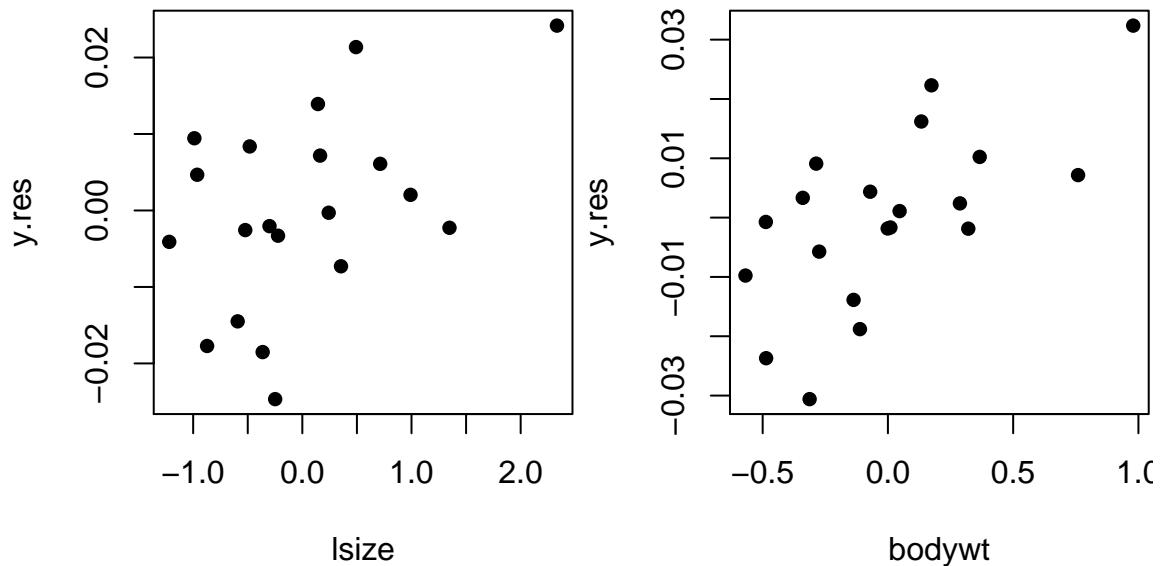


Figure 5.6: Partial residual plots for litter size and body weight in the litters model.

```
par(mar=c(4, 4, .5, .1))
plot(litters.lm, which = 2)
```

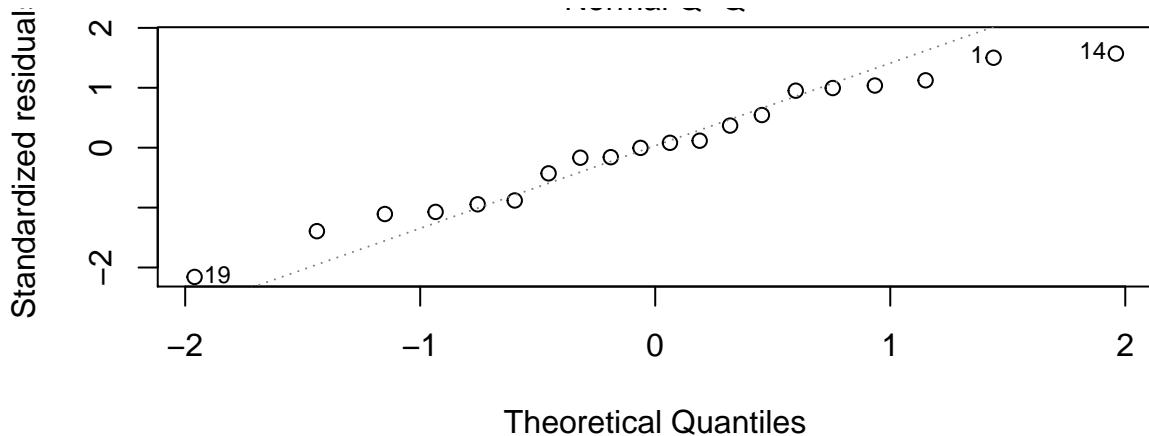


Figure 5.7: Normal QQ-plot for the residuals for the litters model.

Figure 5.7 displays the result, together with a dotted reference line. There would be no cause for concern in this case.

5.3 Serial correlation among the errors

As mentioned at the outset of the chapter, it is not possible to completely assess dependence among the errors, but it is possible to check for autocorrelation or serial correlation in the residuals. Autocorrelation is a measure of the linear dependence between successive measurements. This type of correlation usually only makes sense if there is an ordering, often by time but not always, in the observations.

5.3.1 Lag plots and the autocorrelation function

If we can predict the next value of a residual from the current residual, we say that the residuals are dependent. A way of checking to see if we can predict the current residual from the previous residual is to construct a scatter plot of each residual against the previous residual.

In other words, suppose you have residuals 3.2, -7.8, 9.1, 4.9. A lag plot of these residuals would require plotting the sequence -7.8, 9.1, 4.9 against the sequence 3.2, -7.8, 9.1. If you would see a trend in this scatterplot, you have evidence of dependence, since you would be able to use the trend to make predictions about future residuals based on current residuals. This would be a lag 1 plot.

You can try to predict at other lags. For example, predicting at lag 2 means trying to predict the current residual from a residual taken 2 time units earlier. For the above values, this means you would plot the sequence 9.1, 4.9 against 3.2, -7.8. Obviously, the number of lags you could consider would be limited by the amount of data you have.

Example – Winnipeg temperature data

The data frame `Wpgtemp` (in the *MPV* package) contains Winnipeg's daily maximum temperatures, in degrees Celsius, for the period from January 1, 1960 through December 31, 1980. Figure 5.8 displays the data. A very clear periodic pattern is evident.

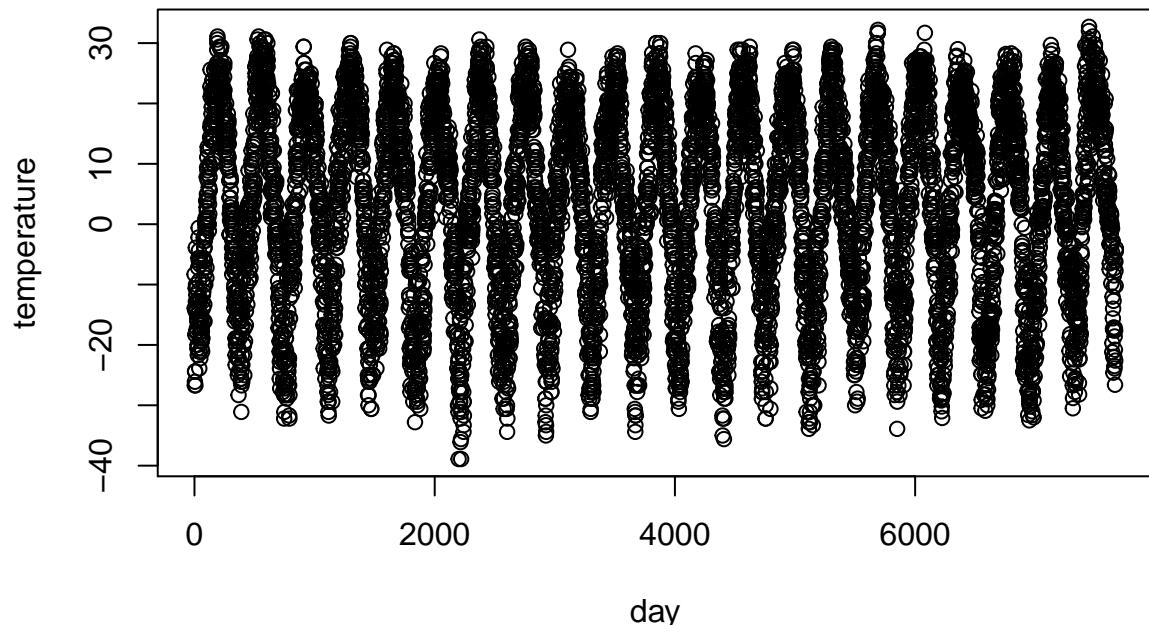


Figure 5.8: Daily maximum temperatures at Winnipeg International Airport for 1960 through 1980.

A very simple model for these data would take into account the periodicity:

$$t = \sin(2\pi d/365.25) + \cos(2\pi d/365.25) + \varepsilon$$

where t represents temperature on day d . We fit the model using

```
temp.lm <- lm(temperature ~ sin(2*pi*day/365.25) + cos(2*pi*day/365.25),
               data=Wpgtemp)
resid.temp <- resid(temp.lm)
```

Figure 5.9 shows lag plots for the Winnipeg temperature data at lags 1, 2, 3 and 4. These correspond to lags of 1 day, 2 days, 3 days, and 4 days. There is a fairly clear positive trend in the first lag plot, meaning that you could

predict a measurement 1 day into the future, using the current measurement. Since the trend is roughly linear, we refer to the association as autocorrelation. More precisely, we are observing a positive autocorrelation at lag 1.

There is a vaguer trend in the second lag plot, and less of a trend in the other plots. We would say that the autocorrelation is decreasing with lag, but there is clearly dependence in the residuals.

```
lag.plot(resid.temp, lag = 4)
```

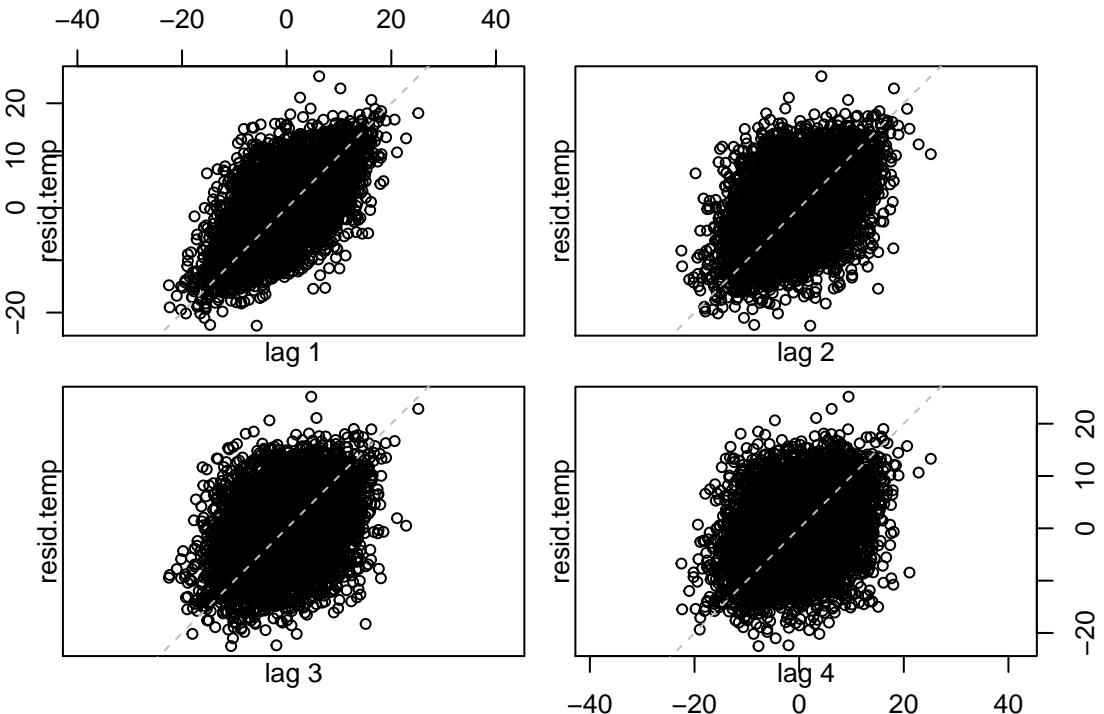


Figure 5.9: Lag plots of the residuals at lags of 1, 2, 3 and 4 days.

The autocorrelation function or ACF lists the autocorrelations for the first several lags.

```
acf(resid.temp, plot=FALSE)

##
## Autocorrelations of series 'resid.temp', by lag
##
##   0    1    2    3    4    5    6    7    8    9    10
## 1.000 0.657 0.442 0.347 0.290 0.236 0.194 0.185 0.174 0.167 0.164
##   11   12   13   14   15   16   17   18   19   20   21
## 0.145 0.126 0.115 0.102 0.095 0.101 0.113 0.100 0.101 0.112 0.110
##   22   23   24   25   26   27   28   29   30   31   32
## 0.112 0.102 0.090 0.088 0.082 0.085 0.086 0.069 0.071 0.062 0.056
##   33   34   35   36   37   38
## 0.044 0.033 0.032 0.030 0.029 0.035
```

From this listing, we see that the correlation between residuals 1 lag apart is 0.657. This matches the positive linear association which is evident in the top left panel of Figure 5.9. The second lag autocorrelation is less: 0.442,

corresponding to the vaguely positive association which is displayed in the second lagplot. For all other lags, the autocorrelation is even less, indicating that it would be difficult to make predictions of the residuals more than 2 time lags in advance.

The autocorrelation function values at the various lags are often displayed in the form of a graph as shown in Figure 5.10.

```
acf(resid.temp)
```

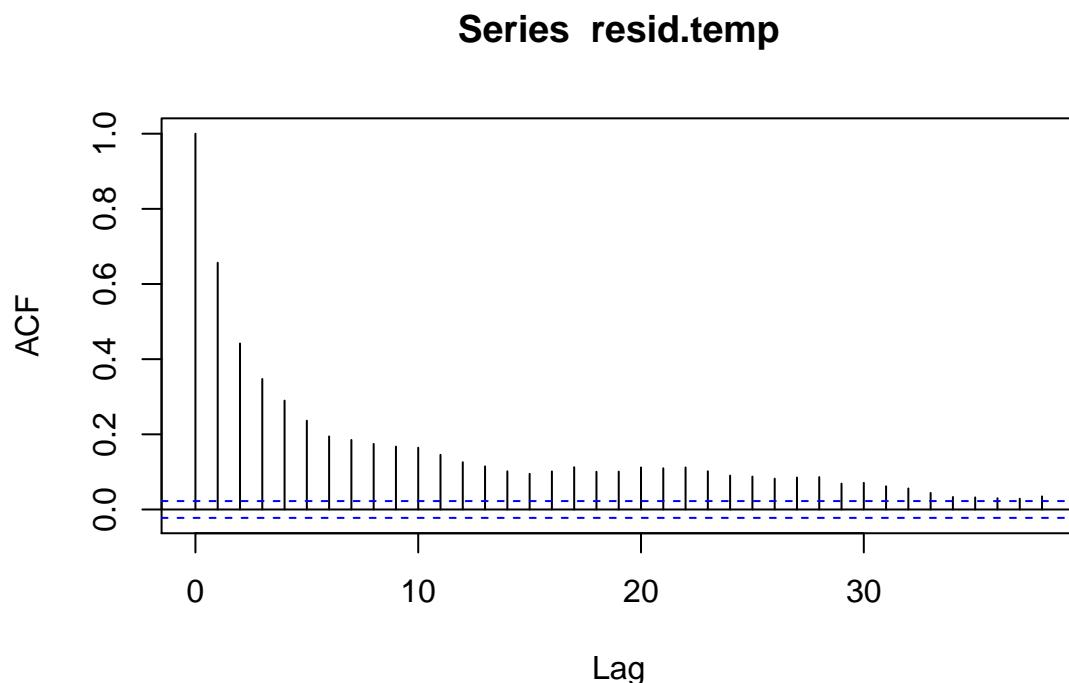


Figure 5.10: Autocorrelations for the Winnipeg temperature model residuals.

The dashed blue lines are test bands which can be used to decide if the autocorrelations differ from 0. Here the only autocorrelation for which there is strong evidence that it is nonzero is the first – just below 0.6, i.e. 0.576 as noted above.

5.3.2 Formal testing for autocorrelation

The Durbin-Watson test is a popular test for autocorrelation, but it examines lag 1 autocorrelation only. It is better to look at the ACF or to use a portmanteau test such as the Box-Ljung test. Here, the test statistic is based on a weighted sum of the autocorrelations up to a specified lag; it is approximately distributed as chi-square. The `Box.test()` function can be used to do the calculation.

Example – Winnipeg temperature data

Based on 10 lags, the Box-Ljung statistic and test can be calculated for the residuals for the Winnipeg temperature model as

```
Box.test(resid(temp.lm), lag=10, type="Ljung-Box")

## 
## Box-Ljung test
## 
## data: resid(temp.lm)
## X-squared = 8014.5, df = 10, p-value < 2.2e-16
```

The p-value is virtually 0, indicating that we have very strong evidence of autocorrelation in the residuals.

Without substantial modification to the model, we would not be able to trust it to make accurate predictions, and coefficient estimates and their standard errors will not be reliable.

5.3.3 Caution in the use of R^2

We have just noted the failure of the model for the Winnipeg temperature data. It should not be used. However, note that value of the coefficient of determination, R^2 , for the model:

```
summary(temp.lm)$r.squared

## [1] 0.842214
```

Unfortunately, the fact that this value is high is often mistaken as an indication of a good model. It is not; we showed above that the regression assumptions are clearly violated. The fact that the R^2 value is large is not of help.

5.4 Transformations on the response variable

When the error variance is not constant and/or there is non-normality, it is often possible to make adjustments by transforming the response variable. We consider variance-stabilizing transformations and Box-Cox transformations to correct for non-normality. It is not unusual for one transformation to resolve both issues simultaneously.

5.4.1 Variance-stabilizing transformations

Consider, again, the model assumptions:

$$E[y|x] = \beta_0 + \beta_1 x$$

and

$$\text{Var}(y|x) = \sigma^2.$$

Note that we are explicitly including the dependence on the predictor variable here.

Set $\mu_y = E[y|x]$. Consider what happens when if $\text{Var}(y|x) = \sigma^2 f(\mu_y)$ for some function $f(x)$ which is not constant. When this happens, we might try to find a function $g(y)$ so that

$$\text{Var}(g(y)|x) = \text{constant}$$

We can proceed by evaluating the first terms of a Taylor expansion of $g(y)$ about μ_y :

$$g(y) = g(\mu_y) + (y - \mu_y)g'(\mu_y) + \frac{(y - \mu_y)^2}{2}g''(\mu_y) + \dots$$

Then

$$\begin{aligned}\text{Var}(g(y)|x) &\doteq \text{Var}(y|x) (g'(\mu_y))^2 \\ &= \sigma^2 f(\mu_y) (g'(\mu_y))^2\end{aligned}$$

$V(g(y)|x)$ will be constant if

$$g'(\mu_y) = \frac{1}{\sqrt{f(\mu_y)}}.$$

In other words, we seek a function $g(z)$ satisfying

$$g'(z) = \frac{1}{\sqrt{f(z)}}.$$

We can find $g(z)$ by integrating:

$$g(z) = \int g'(z) dz = \int \frac{1}{\sqrt{f(z)}} dz.$$

Examples

1. $f(x) = x$ (e.g. Poisson data)

$$\frac{1}{\sqrt{f(x)}} = x^{-1/2}$$

Since

$$\int x^{-1/2} dx = 2x^{1/2},$$

and the constant 2 is irrelevant, we use the transformation

$$g(y) = \sqrt{y}$$

2. $f(x) = x(1-x)$ (e.g. binomial data)

This time,

$$\frac{1}{\sqrt{f(x)}} = \frac{1}{\sqrt{x(1-x)}}.$$

Recall from calculus that

$$\frac{d}{dx} \sin^{-1}(\sqrt{x}) = \frac{1}{2\sqrt{x(1-x)}}.$$

Therefore, the required transformation is

$$g(y) = \arcsin(\sqrt{y}).$$

3. $f(x) = x^2$ (e.g. Exponential data)

Integrating

$$\frac{1}{\sqrt{f(x)}} = \frac{1}{x}$$

leads to $g(y) = \log(y)$.

Example – BioOxyDemand Data

The residual plots for the BioOxyDemand model are suggestive that the error variance is not constant. It appears to be increasing and may be increasing roughly in proportion to the square of the mean. From the above result, a log transformation of the response variable is recommended here. Taking a log of the predictor variable x is also advisable, in order to make the spread of the x values more even: To fit the model

$$\log(y) = \beta_0 + \beta_1 \log(x) + \varepsilon$$

use

```
BioOxyDemand.lmlog <- lm(log(y) ~ log(x), data = BioOxyDemand)
```

Figure 5.11 shows the studentized residuals for this model as obtained using the following code.

```
BioOxyDemand.res <- resid(BioOxyDemand.lmlog) # ordinary residuals
rmse <- summary(BioOxyDemand.lmlog)$sigma # estimate the residual standard error
BioOxyDemand.stud <- BioOxyDemand.res/(rmse*sqrt(1-hat(
  model.matrix(BioOxyDemand.lmlog)))) # studentized
par(mar=c(4, 4, 1,.1))
plot(BioOxyDemand.stud ~ log(x), data=BioOxyDemand, ylab="studentized residuals")
abline(h=0)
```

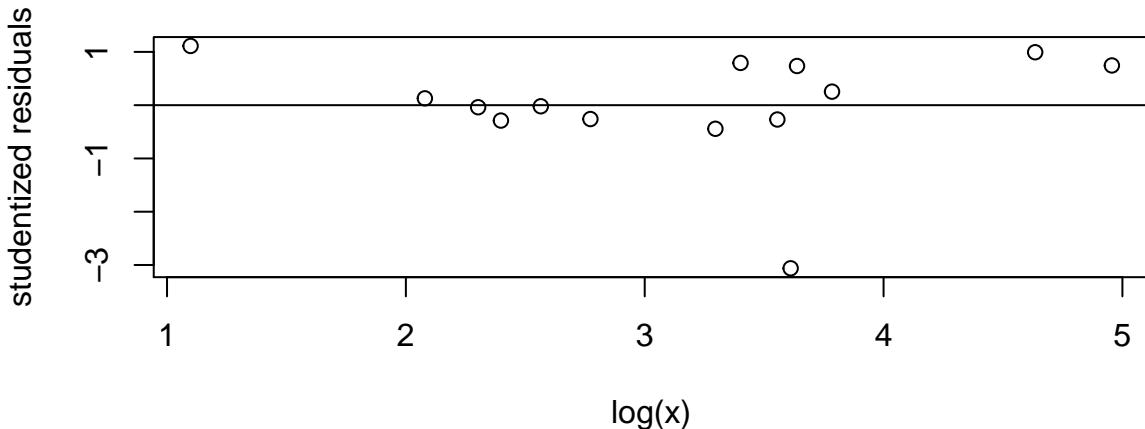


Figure 5.11: Studentized residuals for the BioOxyDemand model on the log scale.

It is evident from the figure that a linear relationship is not appropriate, due to the apparent curvature. There is an extreme outlier.

We conclude that the model is not satisfactory.

5.4.2 Box-Cox transformations

When there is evidence that the errors are non-normal, it is sometimes possible to transform the response variable in such a way that the model errors appear to be more closely approximated by normality. These transformations are either in the form of a power transformation y^λ or a natural log.

The Box-Cox method proceeds by selecting the power λ in the transformation

$$g(y) = y^\lambda$$

by maximum likelihood. This is equivalent to minimizing the SSE with respect to λ (and other parameters).

Unfortunately, the residual sums of squares are not comparable for different values of λ , so caution is warranted. We need to ensure that comparisons are made according to the same standard:

$$y^{(\lambda)} = \begin{cases} \frac{y^\lambda - 1}{\lambda \dot{y}^{\lambda-1}}, & \lambda \neq 0 \\ \dot{y} \log y, & \lambda = 0 \end{cases}$$

where

$$\dot{y} = \text{geometric mean of the } y's$$

The following strategy is easily implemented:

1. Perform the transformation $y_1^{(\lambda)}, \dots, y_n^{(\lambda)}$ for several values of λ .
2. Compute SSE for each value of λ .
3. Select the value of λ which gives the minimum value.
4. Fit

$$\mathbf{y}^\lambda = \mathbf{X}\boldsymbol{\beta} + \epsilon$$

5. Approximate confidence intervals for λ can also be obtained.
6. In R, use the *MASS* function `boxcox()`:

```
boxcox(y ~ x, data= dataset)
```

Example - bacteria data

The average number of surviving bacteria (y) in a canned food product versus time (t , in minutes) of exposure to 300° F heat are given in the *MPV* data set p5.3.

We start by fitting a simple linear model to the data:

```
bact.lm <- lm(bact ~ min, data=p5.3)
```

Figure 5.12 shows the plot of the residuals versus the fitted values and the normal QQ-plot of the residuals. It should be evident that there is nonlinearity that has not been accounted for, and there is non-normality in the residuals.

```
par(mfrow=c(1, 2))
plot(bact.lm, which=1)    # residuals
plot(bact.lm, which=2)    # QQ-plot
```

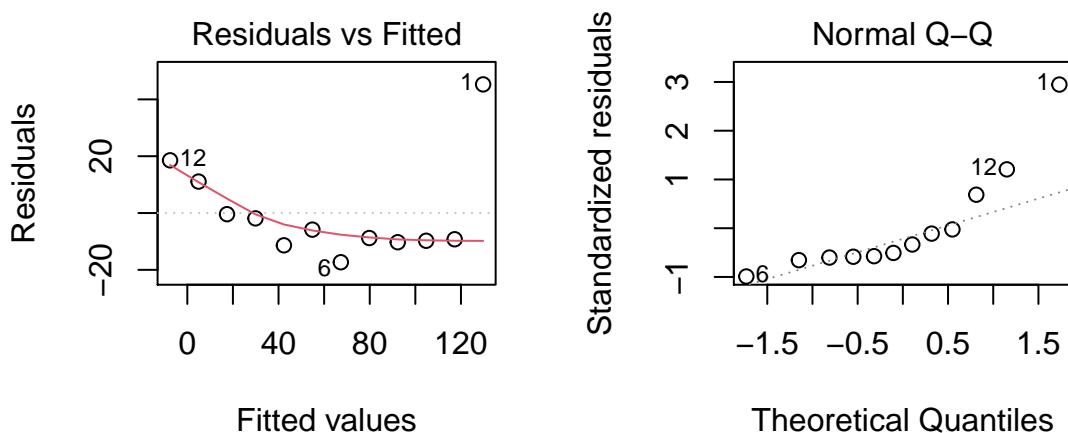


Figure 5.12: Residual diagnostic plots for the bacteria survival data.

Determining the appropriate Box-Cox transformation proceeds as follows, resulting in the plot of the log likelihood displayed in Figure 5.13.

```
library(MASS) # contains the boxcox function
par(mar=c(4, 4, .1, .1))
boxcox(bact.lm) # Box-Cox plot
```

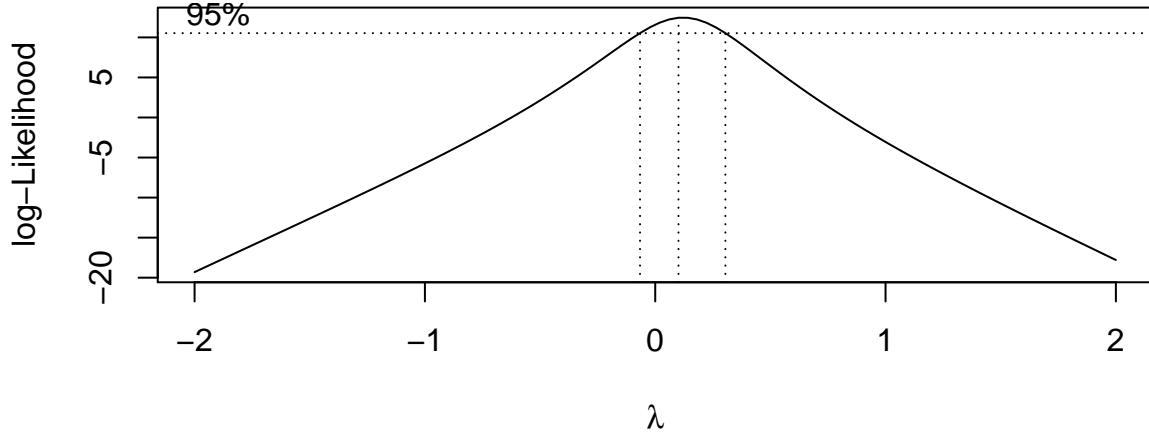


Figure 5.13: Log likelihood function for the Box-Cox parameter in the simple regression model for the bacteria survival data.

Since the 95% interval contains the value 0, we use the log transformation, for simplicity, rather than a value of λ which is close to 0.

Figure 5.14 shows the residual plots after transforming and indicates that the transformation has achieved its goal of making the data appear to be more normal, though we still see evidence of nonlinearity in the plot of the residuals versus fitted values.

```
par(mfrow=c(1, 2), mar=c(4, 4, .1, .1))
bactlog.lm <- lm(log(bact) ~ min, data=p5.3)
plot(bactlog.lm, which=1)
plot(bactlog.lm, which=2)
```

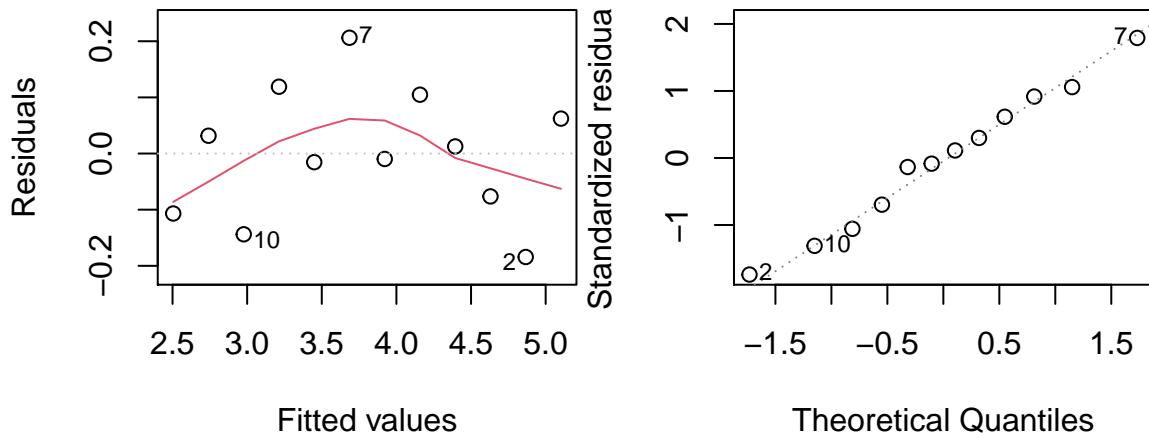


Figure 5.14: Residual diagnostic plots for the Box-Cox transformed bacteria survival data.

Example – trees data

31 observations on Girth (g), Height (h) and Volume (V) are in the `trees` data frame. Geometry suggests that

$$V \doteq \frac{g^2 h}{4\pi}.$$

Taking logs, we might try a model of the form

$$\log V = \beta_0 + \beta_1 \log h + \beta_2 \log g + \varepsilon$$

```
trees.lm <- lm(log(Volume) ~ log(Girth) + log(Height), data = trees)
par(mar=c(4, 4, .1, .1))
boxcox(trees.lm) # (lambda = 1 is OK)
```

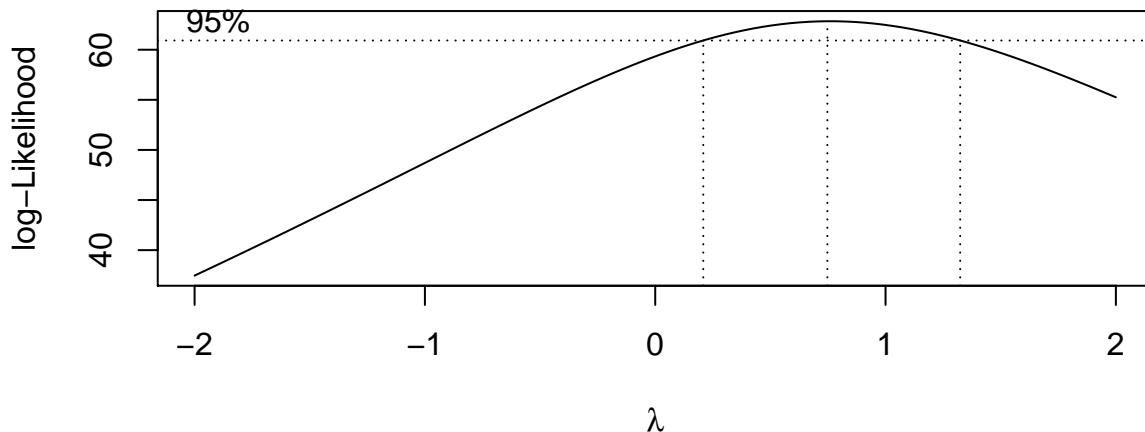


Figure 5.15: Log likelihood function for the Box-Cox parameter in the model for the tree volume data

```
summary(trees.lm)

##
## Call:
## lm(formula = log(Volume) ~ log(Girth) + log(Height), data = trees)
##
## Residuals:
##      Min        1Q    Median        3Q       Max
## -0.168561 -0.048488  0.002431  0.063637  0.129223
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.63162   0.79979 -8.292 5.06e-09 ***
## log(Girth)   1.98265   0.07501 26.432 < 2e-16 ***
## log(Height)  1.11712   0.20444  5.464 7.81e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08139 on 28 degrees of freedom
## Multiple R-squared:  0.9777, Adjusted R-squared:  0.9761
## F-statistic: 613.2 on 2 and 28 DF,  p-value: < 2.2e-16
```

The coefficient of $\log(\text{Height})$ is not distinguishable from 1, and the coefficient of $\log(\text{Girth})$ is not distinguishable from 2. These are geometrically reasonable results.

5.5 Detection and treatment of outliers

An outlier is an extreme observation. If a residual plots more than about 3 standard deviation units away from 0, then the observation should be regarded as an outlier. We have seen that residual plots can reveal outliers.

When detected, outlying observations should be examined closely. Compare the corresponding observation with the rest of the data. If the outlier has been improperly recorded, it should be corrected or discarded.

5.5.1 Handling Outliers

If the outlier has not been improperly recorded, then it could mean that the model is not adequate. In such cases,

- Compare the fitted model with and without the observation.
- If the outlier is making a big difference, one strategy is to report the fitted model obtained without the outlier, but to report the outlier as well. In addition, any plots of the data should properly identify the outlying observation.
- Are there other variables that could be included in the model? A cluster of outliers sometimes indicates that an important variable has been omitted, for example, gender.

If we choose to omit this observation, we should make sure to note its omission clearly.

Example – BioOxyDemand data

In the logarithmic version of the BioOxyDemand model, an outlier was clearly evident. Recall that the `lm` object for this model is `BioOxyDemand.lmlog`. Using the `plot()` function can help us quickly identify the observation number of the outlier:

```
par(mar=c(4, 4, .1, .1))
plot(BioOxyDemand.lmlog, which = 1)
```

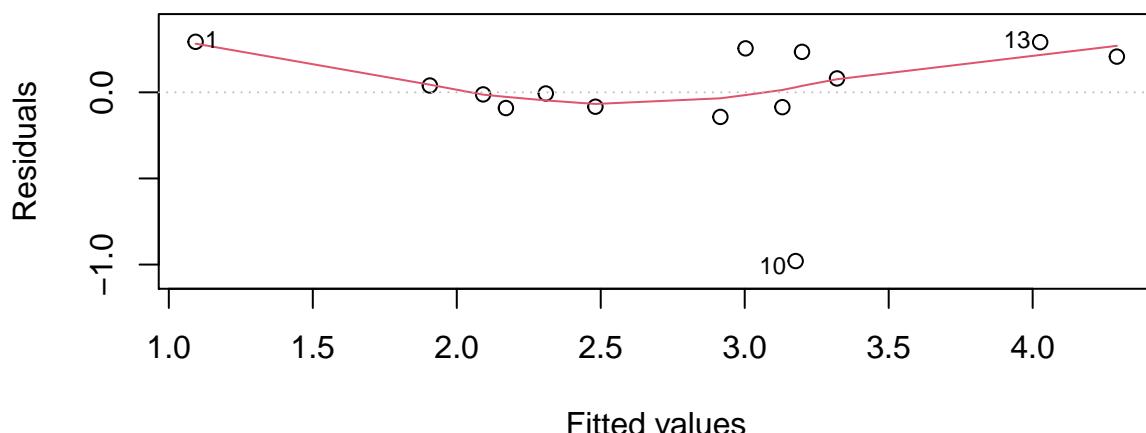


Figure 5.16: Plot of residuals versus fitted values for logarithmic BioOxyDemand model.

On the basis of Figure 5.16, we see that the 10th observation is causing the outlying residual. The 10th observation is

```
BioOxyDemand[10, ]
```

```
##      x  y
## 10 37 9
```

Observations with x values within 10 units of 37 are

```
BioOxyDemand[BioOxyDemand$x > 27 & BioOxyDemand$x < 47, ]  
##      x  y  
## 8 30 26  
## 9 35 21  
## 10 37 9  
## 11 38 31  
## 12 44 30
```

Therefore, we see that the y value of 9 is quite different from the other observations. One might suspect that the value should have been recorded as 29, but we do not know this. All we can say is that we are suspicious about the value, and we are not certain about its veracity. One recourse, if we cannot track down the source of the data to make an inquiry, is to fit the model without the observation and to note its omission. The following code presents one way of doing this:

```
# fit without 10th observation:  
BioOxyDemand.lmlog10 <- lm(log(y) ~ log(x), data = BioOxyDemand[-10,])  
par(mar=c(4, 4, .1, .1))  
plot(log(y) ~ log(x), data = BioOxyDemand) # scatterplot of BioOxyDemand data  
abline(BioOxyDemand.lmlog10) # overlaid fitted curve, omitting observation 10  
points(log(BioOxyDemand[10,]), col=2, pch=16) # highlighting omitted observation  
text(log(37), log(7), "omitted observation")
```

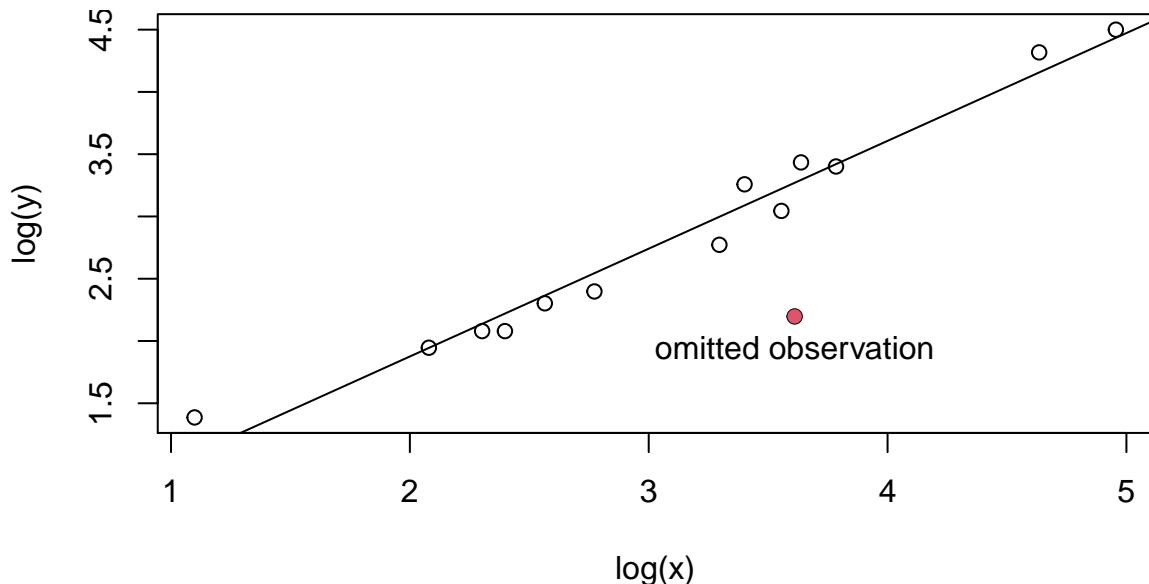


Figure 5.17: Scatterplot of BioOxyDemand data with fitted line overlaid, based on omission of one observation.

Figure 5.17 displays the scatterplot of the data with the fitted line overlaid, together with the observation that had been omitted. By proceeding in this way, a possibly more accurate model has been obtained, while preserving all of the information about the data. It is conceivable that, upon presenting the results in this way, that someone in the intended audience may be able to offer an explanation for the faulty observation and/or a correction.

5.5.2 Leverage and influence diagnostics

A high leverage point is an observation which lies near an extreme of the space of explanatory variables.

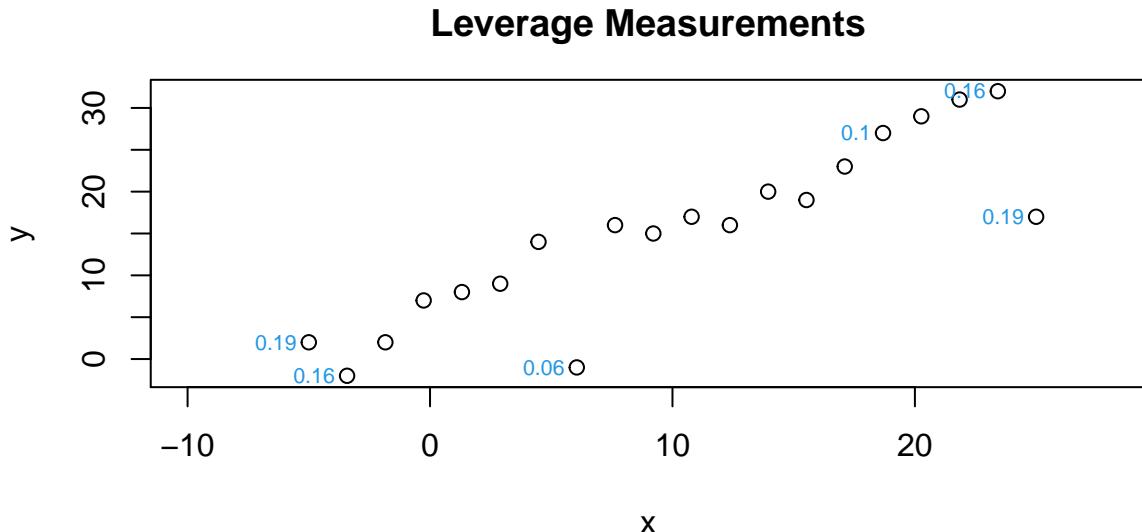


Figure 5.18: A scatterplot of typical simple regression data with response variable y and predictor variable x . The numbers in blue are the hat diagonal values that correspond to the nearby observations.

For example, Figure 5.18 displays a typical situation which occurs in simple linear regression. The response values are plotted against a single predictor x . The numbers in blue give the hat diagonal values; these numbers tend to be larger at the extremes of the range of the x values. Note that there is an outlier, but it is within the range of the x values, and its hat diagonal value is not very large.

An Indexinfluential observation is one which is both a high leverage point and an outlier in the response space. In other words, such an observation is an outlier in both realms.

Figure 5.19 displays an example of this. Three lines are fit to the data. One uses the full data set, one omits an influential observation and one omits a low leverage outlying observation. The numbers in blue correspond to the Cook's distance values which we will see are a measure of influence. The message from this figure is that when the low leverage observation is omitted, very little happens to the regression line. When the influential observation is omitted, the slope of the line changes substantially. This raises the question as to which line is most accurate; in other words, we have increased uncertainty about the model when dealing with influential observations.

5.5.3 Leverage theory

Consider

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

$$\hat{\beta}_1 = \frac{\sum(x_i - \bar{x})y_i}{S_{xx}} = \sum c_i y_i$$

i.e. $\hat{\beta}_1$ is a weighted average of the y 's.

High leverage points are such that $(x_i - \bar{x})^2$ is large. The largest weights are for small or large x 's:

$$\frac{x_i - \bar{x}}{S_{xx}}$$

is large when $x_i >> \bar{x}$ and when $x_i << \bar{x}$. Therefore, the observations that give greatest weight to the determination of the slope are the smallest and largest x 's.

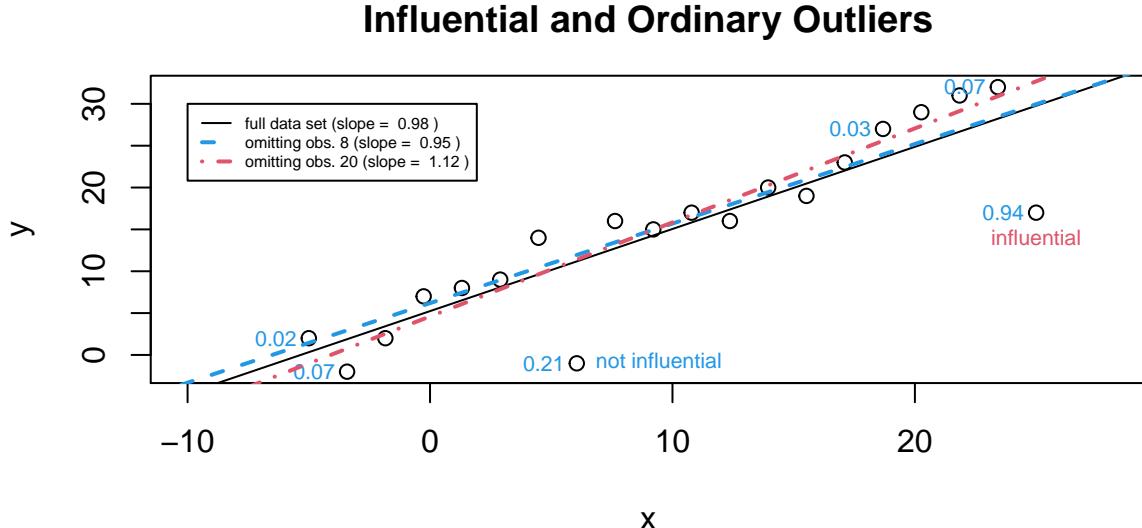


Figure 5.19: A scatterplot of typical simple regression data, together with the resulting fitted regression lines using the full data set, and with omission of an influential observation, and with omission of a low leverage outlier. The numbers in blue are the Cook's Distance values that correspond to the nearby observations. The two outliers are identified as having influence and not having influence.

Example

Suppose we have $n = 5$ observations. Then

$$x_1 = 0, x_2 = .25, x_3 = .5, x_4 = .75, x_5 = 1$$

$$\bar{x} = .5, S_{xx} = .625$$

$$\hat{\beta}_1 = -.8y_1 - .4y_2 + 0y_3 + .4y_4 + .8y_5$$

Note that the middle observation makes no contribution at all, while the two extremes contribute a lot. The correctness of the slope estimate depends heavily on the quality of the 1st and 5th measurements.

5.5.4 Measuring leverage – hat diagonal values

In multiple regression, it is more difficult to identify extreme observations in the multi-dimensional space of the x variables. The norm or distance we use to decide if points have high leverage is based on the positive definite matrix $(\mathbf{X}^\top \mathbf{X})^{-1}$.

The leverage of an observation at \mathbf{x}_i is defined by its $(\mathbf{X}^\top \mathbf{X})^{-1}$ distance:

$$\mathbf{x}_i^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{x}_i$$

but this is the ii element (or i th diagonal element) of the hat matrix:

$$\mathbf{H} = \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top.$$

Therefore, the leverage of the i th observation is $= h_{ii}$.

The average leverage value for a model and data can be computed using the trace of the hat matrix. Since the sum of the hat diagonal values is the trace of the hat matrix, and

$$\text{trace}(\mathbf{H}) = \text{trace}(\mathbf{X}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1}) = \text{trace}(\mathbf{I}) = p$$

where p is the number of columns in the \mathbf{X} matrix, we can compute the average hat diagonal value as

$$\frac{1}{n} \text{tr}(\mathbf{H}) = \frac{p}{n}.$$

A useful benchmark to determine if an observation is high leverage, is to compare it with twice the average, i.e. $\frac{2p}{n}$. Any hat value larger than this is considered high leverage.

Example – litters brain weight model

To obtain the leverage values in R for the model relating brain weight to body weight and litter size for the `litters` data set, type

```
litters.lm <- lm(brainwt ~ bodywt + lsize, data = litters)
round(lm.influence(litters.lm)$hat, 2)

##    1     2     3     4     5     6     7     8     9     10    11    12    13    14
## 0.20 0.17 0.13 0.16 0.09 0.27 0.07 0.07 0.16 0.09 0.13 0.08 0.14 0.07
##   15    16    17    18    19    20
## 0.13 0.09 0.43 0.13 0.20 0.20
```

We can visualize some of these values by plotting the predictors against each other and noting some of the corresponding hat diagonal values. This is done in Figure 5.20 which was constructed using the following code.

```
par(mar=c(4, 4, .1, .1))
plot(bodywt ~ lsize, data=litters, xlim=c(3, 12), ylim=c(5, 10.5))
samp <- c(1, 3, 6, 9, 11, 14, 17, 20) # display hat values for these obs.
attach(litters)
text(lsize[samp], bodywt[samp], round(lm.influence(litters.lm)$hat[samp], 2),
      cex=.7, pos=1, col=4)
```

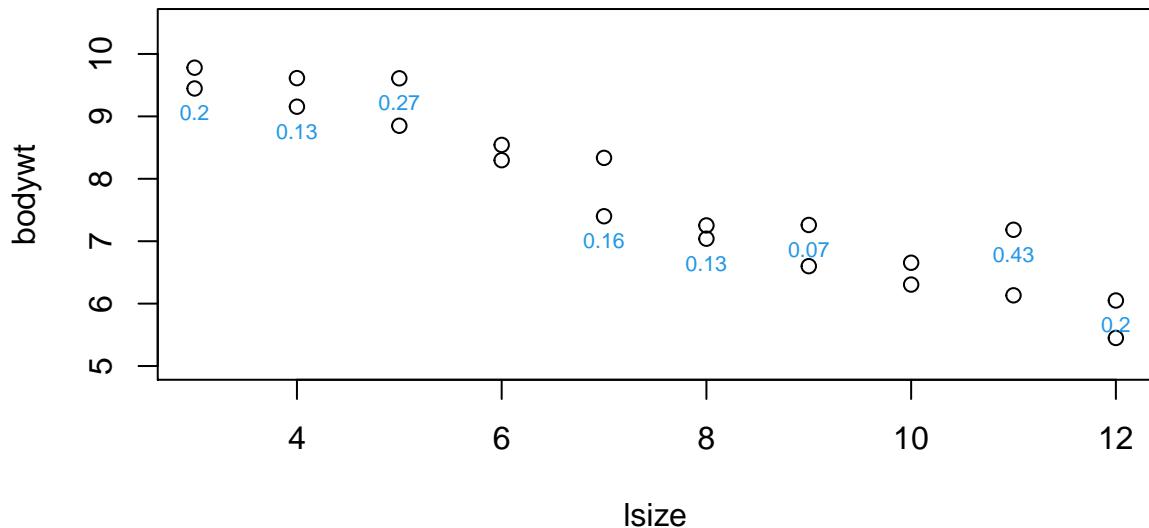


Figure 5.20: Leverage values (i.e hat diagonal values) of some of the litters observations in the brain weight model.

```
detach(litters)
```

The cut-off value for high leverage values in this model is $h_{ii} > \frac{2p}{n} = .3$. Therefore, observation 17 ($h_{17,17} = .4326$) is high leverage under this model.

5.5.5 Measuring Influence - Cook's D

A general method to see if an observation is influential is to **delete it and see how the estimates change**.

$$\hat{\beta} = \text{LS estimates} - \text{full data set}$$

$$\hat{\beta}_{(i)} = \text{LS estimates} - i\text{th observation deleted}$$

Look at the difference: $\hat{\beta} - \hat{\beta}_{(i)}$. If the difference is big, then we conclude that the i th observation is influential. If the difference is small, the i th observation is not influential.

How can we tell if this difference is big or small? One answer is Cook's distance which gives us an idea of how large this difference is, according to a standard scale:

$$D_i = \frac{1}{p\text{MSE}} (\hat{\beta} - \hat{\beta}_{(i)})^\top (\mathbf{X}^\top \mathbf{X}) (\hat{\beta} - \hat{\beta}_{(i)})$$

After some algebra, we can obtain the simpler formula

$$D_i = \frac{r_i^2}{p} \frac{h_{ii}}{1 - h_{ii}}$$

where r_i = i th studentized residual;

Thus, the influence of an observation is related to its **leverage** and to whether it is an **outlier**. Note that:

- * high leverage outliers are influential
- * low leverage outliers are less influential
- * high leverage points that are not outliers are less influential

A rough benchmark is that high influence corresponds to any case where $D_i > 1$. The Cook's distance values for the data values in the artificial data set can be calculated as follows:

```
round(cooks.distance(y.lm), 2)
##    1     2     3     4     5     6     7     8     9    10    11    12    13    14
## 0.02 0.07 0.01 0.01 0.01 0.00 0.03 0.21 0.01 0.00 0.00 0.00 0.00 0.00
##    15    16    17    18    19    20
## 0.00 0.03 0.04 0.07 0.07 0.94
```

It is more useful to display these values graphically as in Figure 5.21, and the object-oriented **plot()** function can do this, using the **which=4** option:

```
par(mar=c(4, 4, .1, .1))
plot(y.lm, which=4)
```

Example – litters data

We can assess the influence of the observations in the model relating brain weight to body weight and litter size based on the litters data set, and display the results in Figure 5.22. This is done using the following code.

```
par(mar=c(4, 4, .1, .1))
plot(litters.lm, which=4)
```

Note that the 19th observation is the most influential observation. If we plot the hat diagonal values (i.e. the leverage values) against observation number and the absolute values of the residuals, as was done in Figure 5.23, we see that the 19th observation is not a very high leverage point; its influence is due mainly to it being a fairly large residual. On the other hand, the 17th observation, which is the 2nd most influential point, has influence due to leverage; it is not an outlying observation. The 17th observation might be worth examining in more detail, because its high leverage might be pulling the fitted surface towards itself.

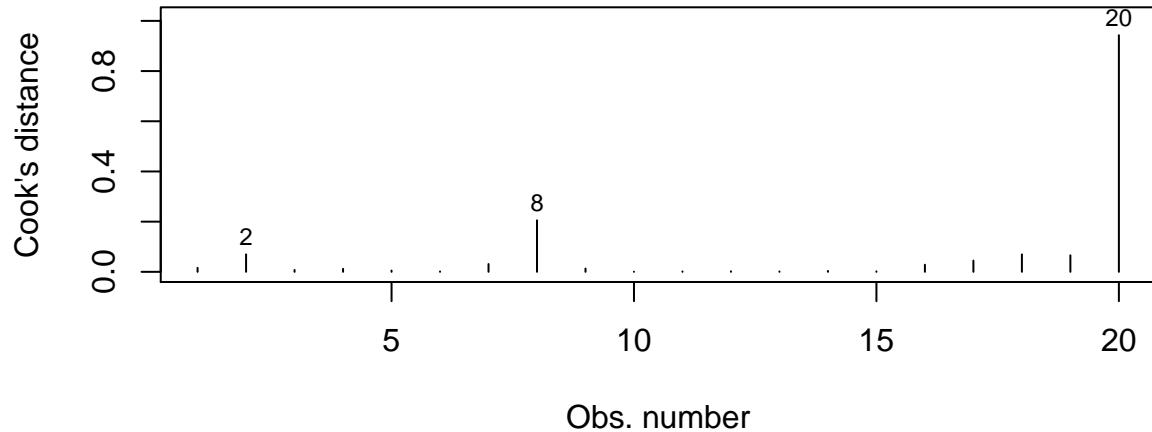


Figure 5.21: Cook's distance plot for the artificial data set and simple regression model.

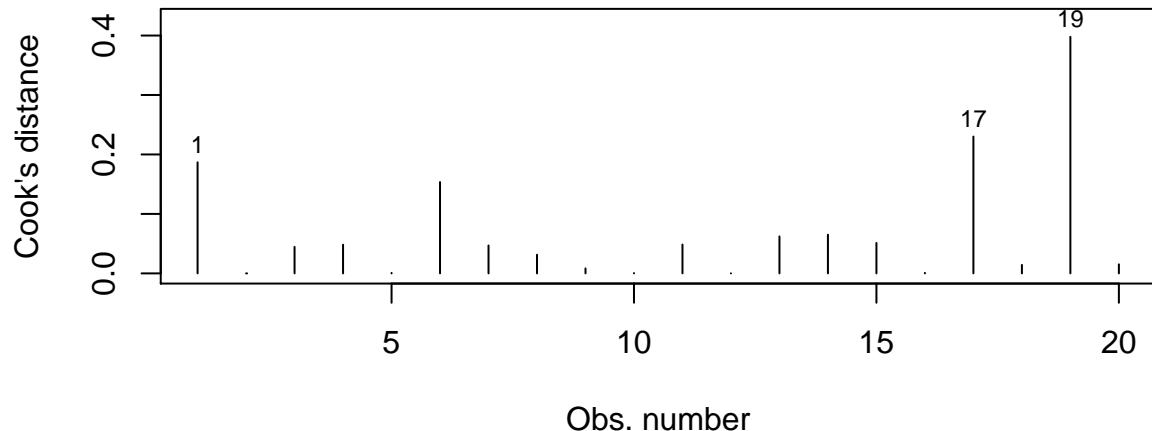


Figure 5.22: Cook's distance plot for the litters data set and model.

5.5.6 Measuring effects on regression coefficients due to influence – DFBETAS

What effect does the i th observation have on the estimate $\hat{\beta}_j$?

To answer this, we compare $\hat{\beta}_j$ with $\hat{\beta}_{j(i)}$ which is the j th coefficient estimate based on data with the i th observation omitted. We would be interested in the difference between the two estimates:

$$\hat{\beta}_j - \hat{\beta}_{j(i)}$$

Again, this difference is not standardized so it is difficult to interpret large and small values. To standardize, we divide by an approximate 'standard deviation': $\sqrt{S_{(i)}^2 C_{jj}}$ where

$$S_{(i)}^2 = \text{MSE}_{(i)} \quad \text{and} \quad C_{jj} = (\mathbf{X}^\top \mathbf{X})_{jj}^{-1}$$

This leads to the difference in β s statistic:

$$DFBETAS_{j,(i)} = \frac{\hat{\beta}_j - \hat{\beta}_{j(i)}}{\sqrt{S_{(i)}^2 C_{jj}}}$$

If this is **positive**, the effect of the i th observation is to **increase** the estimate of $\hat{\beta}_j$. If it is **negative**, the effect of the i th observation is to **decrease** the estimate of $\hat{\beta}_j$.

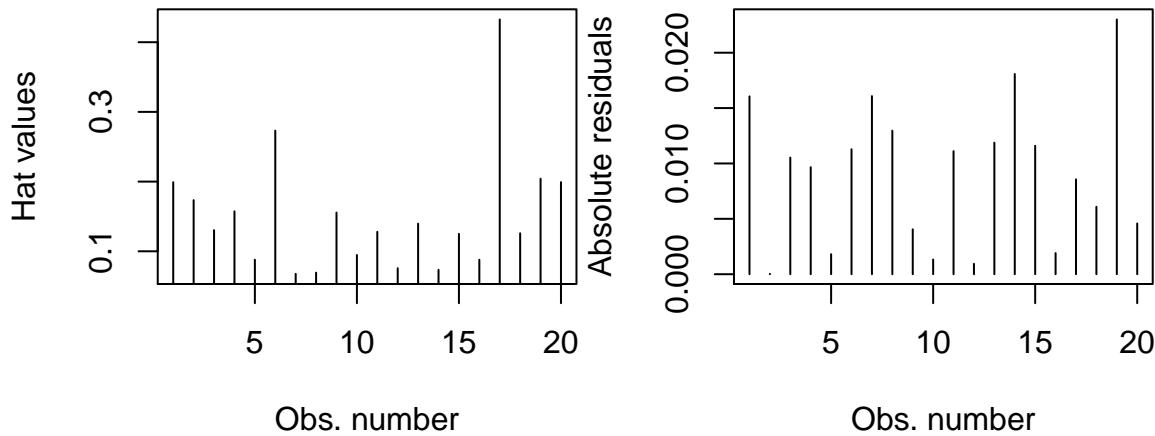


Figure 5.23: Leverage values and residual magnitudes for the litters data set and model.

A rough guideline for the use of this statistic is as follows. The i th observation is [influential](#) on the j th coefficient estimate if

$$|DFBETAS_{j,(i)}| > \begin{cases} 1, & \text{small } n \\ 2/\sqrt{n}, & \text{large } n \end{cases}$$

We use `dfbetas()` to evaluate DFBETAS. For the `litters` model with covariates `bodywt` and `lsize` (i.e. 3 regression coefficients), `dfbetas(litters.lm)` returns a 3 column matrix. The first column corresponds to the intercept, and the other two columns correspond to the covariates. We obtain these and plot them in Figure 5.24.

```
par(mfrow=c(1, 2))
plot(dfbetas(litters.lm)[, 2], pch=16)
plot(dfbetas(litters.lm)[, 3], pch=16)
```

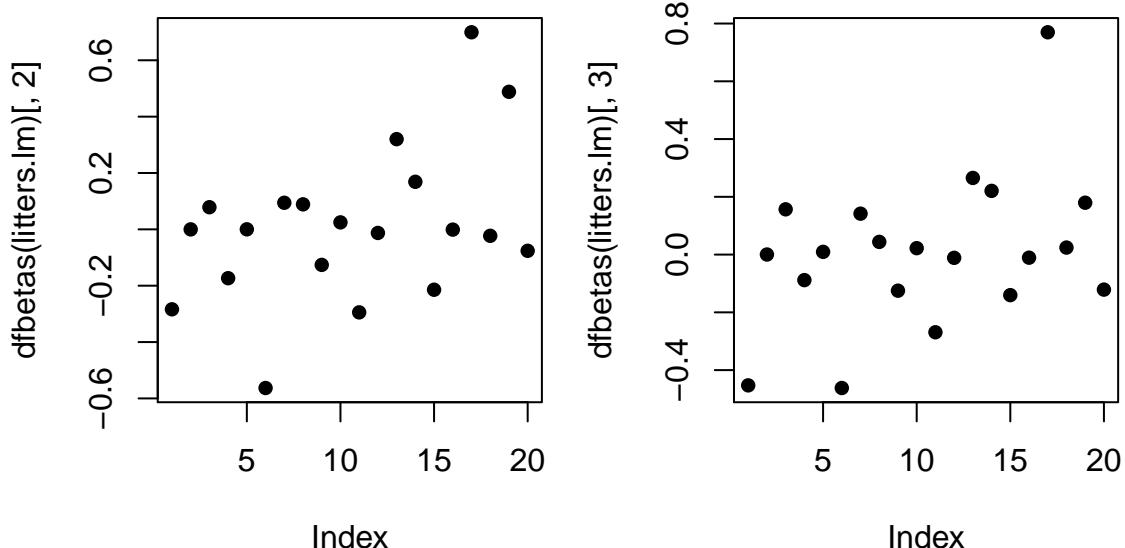


Figure 5.24: DFBETAS values for coefficients of body weight and litter size in the litters brain weight model.

The values in Figure 5.24 are standardized so none of them are strikingly large.

5.5.7 Measuring effects on fitted values due to influence – DFFITS

What effect does the i th observation have on the fitted value \hat{y}_i ? This time, compare \hat{y}_i with $\hat{y}_{i,(i)}$ $\hat{y}_i - \hat{y}_{i,(i)}$. Standardize this by dividing by its standard deviation $\sqrt{S_{(i)}^2 h_{ii}}$, obtaining the difference in fitted values statistic:

$$\begin{aligned} DFFITS_{(i)} &= \frac{\hat{y}_i - \hat{y}_{i,(i)}}{\sqrt{S_{(i)}^2 h_{ii}}} \\ &= \frac{e_i}{S_{(i)} \sqrt{1 - h_{ii}}} \sqrt{\frac{h_{ii}}{1 - h_{ii}}} \end{aligned}$$

If this is **positive** (**negative**), the effect of the i th observation is to **increase**(**decrease**) the estimate of y_i . Roughly speaking, the i th observation is **influential** on the i th fitted value if

$$|DFFITS_{(i)}| > \begin{cases} 1, & \text{small } n \\ 2\sqrt{p/n}, & \text{large } n \end{cases}$$

We use `dffits()` to evaluate DFFITS. For the `litters` model, `dffits(litters.lm)` returns a single vector of length 20, which is the sample size. We obtain these values and plot them in Figure 5.25.

```
plot(dffits(litters.lm), pch=16)
```

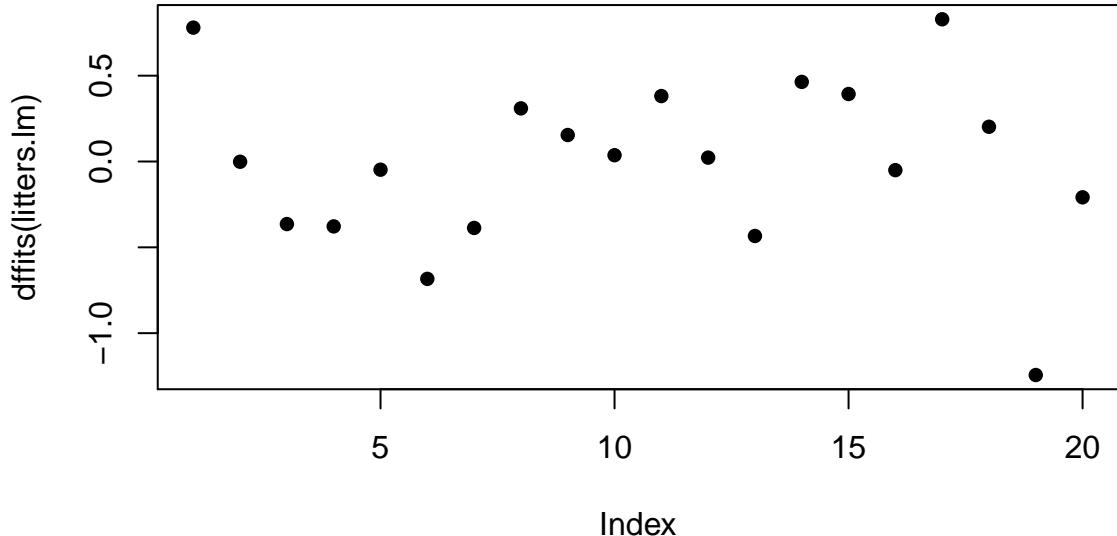


Figure 5.25: DFFITS values for the litters brain weight model.

The 19th value in Figure 5.25 is somewhat large, indicating that the 19th observation is having undue influence on the fitted values.

5.5.8 Summary

Look for

- **Outliers** (in the residual plots)

if such points are influential, the fitted model is not adequate

- **High Leverage Observations** (on the hat diagonal)
such points are potentially influential
- **Influential Observations (Cook's D, DFFITS, DFBETAS)**
such points may be distorting the fitted model

Keep in mind that:

1. Outliers might not be influential.
2. High leverage observations might not be influential.
3. Influential observations might not be outliers.

5.6 Exercises

1. Compute the four types of residuals for the litters model where brain weight has been regressed against body weight and litter size. Plot each type of residual against body weight and identify any mild or extreme outliers. Is there evidence that the error variance is increasing or decreasing with body weight?
2. Consider the data set

x	y
1	7
2	19
3	8

and the model $\hat{y} = .5x + 10.33$.

- (a) Compute the raw residuals. Can you tell if there are any outliers?
- (b) Compute the standardized residuals, using the fact that the residual standard error is 9.39. Can you tell if there are outliers now.
- (c) Compute the PRESS residuals and studentized residuals.
3. (a) What is the marginal distribution of the i th raw residual? (State all assumptions you are making.)
 (b) Is the i th raw residual independent of the MSE, assuming that the responses are independent normal random variables with a constant variance? If not, what might you change in order that the variance estimate is independent of the i th raw residual?
 (c) Why might we be interested in the independence of these two quantities? In particular, what can be said about the distribution of the standardized and studentized residuals?
4. Consider the data in `table.b3` of the *MPV* package.
 - (a) Regress y against x_5 and x_9 , and construct plots of the standardized, studentized, and PRESS residuals versus the fitted values. Comment on the plots.
 - (b) Construct plots of the residuals from part (a) against each of the other variables which were not included in the model. Comment on the plots.
 - (c) Construct partial regression plots for x_5 and x_9 . Comment on the plots.
5. Obtain normal QQ-plots for the residuals from the models of the `BioOxyDemand` data before and after taking logarithms. Is the noise in either model reasonably approximated by normality?

6. Suppose measurements are taken on a predictor variable x and a response variable y where

$$\text{Var}(y) = e^{-2\mu_y} \text{ where } \mu_y = E[y].$$

Derive the variance-stabilizing transformation.

7. Suppose measurements are taken on a predictor variable x and a response variable y where

$$\text{Var}(y) = \sqrt{\mu_y} \text{ where } \mu_y = E[y].$$

Derive the variance-stabilizing transformation.

8. Consider simple regression. What kind of transformation should be used if it is determined that the response variance is proportional to $e^{-\mu^2}/\mu^2$, where μ is the response mean?

9. Consider the model of Exercise 7i from Chapter 2.

(a) Show that for data from such a model, $\text{Var}(y) = \sigma^2 + \sigma_B^2 \mu_y^2 / \beta_1^2$, assuming $\beta_1 \neq 0$.²

(b) Show that the variance-stabilizing transformation in this case is

$$g(y) = \frac{\beta_1}{\sigma_B} \log \left| \frac{\sigma_B y}{\beta_1 \sigma} + \sqrt{1 + \frac{\sigma_B^2 y^2}{\beta_1^2 \sigma^2}} \right|.$$

(c) Identify the practical difficulties involved in using this transformation, and suggest some strategies that might be used to overcome these difficulties.

10. Using the Box-Cox normalizing procedure, it has been determined that $\lambda = 0$ is appropriate when regressing y on x , using the following data

x	0	1	4	9	16
y	4	1	9	25	16

Display a table of the transformed data.

11. Using the Box-Cox normalizing procedure, it has been determined that $\lambda = -1$ is an appropriate power to use when regressing y on x , using the following data

x	0	1	4	9	16
y	4	1	9	25	16

Display a table of the transformed data.

12. Fit the model

$$\log(y) = \beta_0 + \beta_1 t + \beta_2 t^2 + \varepsilon$$

to the bacteria survival data where y represents the counts in `bact` and t represents the time in `min`. Check the plot of the residuals versus fitted values and the normal QQ-plot of the residuals. Is this model better or worse than the simple regression model?

13. Consider the data set

```
##           x      y
## 1     1.0  3.00
## 2     2.0  3.85
## 3     3.0  5.40
## 4     4.0  7.87
## 5     5.0  5.71
```

²This means that this transformation does not apply to the data simulated in the Chapter 2 exercise, since $\beta_1 = 0$ there.

```

## 6   6.0  8.52
## 7   7.0  7.15
## 8   8.0  8.02
## 9   9.0  9.73
## 10 10.0  8.98
## 11 11.0  9.54
## 12 12.0 10.26
## 13 13.0  8.64
## 14 13.6  9.43
## 15 14.9 10.21

```

- (a) Fit a linear model relating y to x . Compute a 95% confidence interval for the slope.
- (b) Check the Cook's distance plot. Which observations appear to be influential?
- (c) Recompute the linear regression after removing observations 13, 14, and 15. What has happened to the regression slope? (Give a 95% confidence interval.)
- (d) Comment on the above results. In particular, note how effective or ineffective Cook's distance is in identifying *groups* of influential observations, and by constructing an appropriate plot, identify a problem with the original fitted model.
14. Thirteen observations were used to fit the regression model
- $$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon, \quad \text{Var}(\varepsilon) = \sigma^2$$
- The fourth observation was
- $$x_1 = -1, x_2 = -1, y = 11$$
- $(X^\top X)$ was a diagonal matrix with entries 13, 12 and 12. The parameter estimates were 1, 5, 4, respectively, and the MSE was 150.0.
- (a) Calculate the fitted value \hat{y}_4 .
- (b) Calculate the ordinary residual e_4 .
- (c) Calculate the leverage for the fourth observation. Is it a high leverage observation?
- (d) Calculate the PRESS residual $e_{(4)}$.
- (e) Calculate the 4th R-student residual.
- (f) Calculate $DFFITS_4$. Interpret.
15. Examine the outlier in the logarithmic model for the `BioOxyDemand` data set by plotting Cook's distance, leverage and DFFITS values for all observations in the data frame. Does the outlier exert high leverage? Is the outlier influential? Is it having an undue influence on its fitted value?
16. Consider the `gasdata` data frame in the `MPV` package, relating daily natural gas consumption to outside temperature and the presence or absence of insulation.
- (a) Fit a model relating the square root of average daily gas usage to temperature. Create a plot of the residuals versus the fitted values. What does this plot tell you?
- (b) Now, add a term which is quadratic in temperature. Again, plot the residuals and note any problems.
- (c) Now, add `I1`, `I3` and `I1:temp` terms to your model. Again, construct a residual plot and identify any problems.
- (d) Construct a normal QQ-plot for the above model. What do you conclude?
- (e) Conduct an ANOVA test to decide if terms involving `I2`, `I2:temp` and `I3:temp` should be added. Interpret the result in terms of the effect of insulation on gas usage.

6

Bending and Breaking the Regression Assumptions

6.1 Use of indicator variables in simple regression

The multiple regression framework is usually thought to be appropriate for modelling the relationships among continuous or numeric variables. Categorical variables can also be used in multiple regression as covariates or predictor variables. The trick is to convert them to a numeric variables, called factors. Mathematically, this requires the use of one or more indicator variables.

An indicator variable is usually defined to take on the value 1, if a particular level of a categorical variable is observed, and the value 0, otherwise. That is, for a factor with levels A and not A, we write

$$x_i = \begin{cases} 1, & \text{if the observation on the } i\text{th categorical variable is in level A} \\ 0, & \text{otherwise} \end{cases}$$

With this coding, we can model the corresponding response, Y_i as

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i.$$

This is just the simple linear regression model, and its analysis runs essentially as before. The least-squares estimates of β_0 and β_1 can be calculated using the simple regression formulas. Among other things, this provides an alternative way of analyzing a two-sample problem, the usual way of comparing the means of two samples. Also, note that other codings are possible, such as the use of ‘-1’ in place ‘0’. As long as the same coding protocol is used consistently throughout the analysis, the conclusions from the analysis should be the same. The model can be fit to the data using least-squares. R, and other high quality statistical software, will use the QR decomposition as shown previously.

The design matrix \mathbf{X} will have n rows and 2 columns in this case. The first column consists of 1’s, and the second column will only contain 1’s and 0’s, depending on the coded values of the categorical predictor variable. The model is then written as

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

as before. The least-squares estimator for β can be obtained by minimizing

$$(\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$$

with respect to β . This expression can be re-written as

$$\mathbf{y}^T\mathbf{y} - 2\beta^T\mathbf{X}^T\mathbf{y} + \beta^T\mathbf{X}^T\mathbf{X}\beta$$

and differentiated with respect to β (vector differentiation) to obtain the gradient

$$-2\mathbf{X}^T\mathbf{y} + 2(\mathbf{X}^T\mathbf{X})\beta.$$

Setting this to 0, we find the minimizer as

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

Again, we emphasize that this calculation would not be carried out by any sensible piece of software, but the result is true mathematically.

Example – height versus sex

Heights of eight people have been recorded, along with their sex. We might be interested in whether sex plays a role in explaining variation in height. The data are recorded in `ht.df`.

```
ht.df

##   height sex
## 1    160   F
## 2    150   F
## 3    175   M
## 4    165   M
## 5    170   M
## 6    172   M
## 7    171   M
## 8    155   F
```

How is height (y , in cm) related to sex? Sex is not a quantitative variable. It is a categorical factor. We use the following coded variable to represent a two-level factor as a numeric variable:

$$x_i = \begin{cases} 0, & \text{if M} \\ 1, & \text{if F} \end{cases}$$

A manual encoding can be done in R as follows. We will see a more automatic method in the next section.

```
ht.df$x <- numeric(8) # creates a column for the indicator values
ht.df$x[ht.df$sex == "F"] <- 1 # the other values are 0
```

After coding the factor, we have the following data set:

```
ht.df

##   height sex x
## 1    160   F 1
## 2    150   F 1
## 3    175   M 0
## 4    165   M 0
## 5    170   M 0
## 6    172   M 0
## 7    171   M 0
## 8    155   F 1
```

We can check if there is a relation between height and sex using model:

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

The design matrix for this version of the model is

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}.$$

We can code up this design matrix using

```
X <- cbind(rep(1, 8), ht.df$x)
```

To find the least-squares estimate of the vector β , we first find the QR decomposition of \mathbf{X} . The first two columns of \mathbf{Q} form the matrix \mathbf{Q}_1 . They are calculated from

```
X.qr <- qr(X)
Q1 <- qr.Q(X.qr)
Q1

##          [,1]      [,2]
## [1,] -0.3535534 -0.4564355
## [2,] -0.3535534 -0.4564355
## [3,] -0.3535534  0.2738613
## [4,] -0.3535534  0.2738613
## [5,] -0.3535534  0.2738613
## [6,] -0.3535534  0.2738613
## [7,] -0.3535534  0.2738613
## [8,] -0.3535534 -0.4564355
```

Then $\mathbf{Q}_1^T \mathbf{y}$ is found using

```
Q1y <- t(Q1) %*% ht.df$height
Q1y

##          [,1]
## [1,] -465.98337
## [2,]  21.36118
```

The \mathbf{U} matrix coming from the QR decomposition is calculated as

```
U <- qr.R(X.qr)
U

##          [,1]      [,2]
## [1,] -2.828427 -1.060660
## [2,]  0.000000 -1.369306
```

Therefore, the estimate of the regression coefficient vector $\hat{\beta}$ is calculated as

```
solve(U, Q1y)

##          [,1]
## [1,] 170.6
## [2,] -15.6
```

Thus,

$$\hat{\beta} = \begin{bmatrix} 170.6 \\ -15.6 \end{bmatrix}$$

and the fitted model is

$$\hat{\mu} = 170.6 - 15.6x$$

The fitted values for the model are easily calculated (there are only 2 possible distinct values):

obs	sex	x	y.hat
1	F	1	170.6 - 15.6 = 155
2	F	1	155
3	M	0	170.6 - 0 = 170.6
4	M	0	170.6
5	M	0	170 .6
6	M	0	170.6
7	M	0	170.6
8	F	1	155

The fitted values are the average heights for each sex. This fact connects this model formulation with the two-sample problem of comparing means as noted above.

6.1.1 Error variance and standard error estimates

Because we have formulated the problem using a simple linear regression model, we can use the formulas from simple regression to handle this situation. For example, we can calculate the error variance estimate using the formula

$$\hat{\sigma}^2 = \text{MSE} = \frac{\text{SSE}}{n - p} = \frac{\mathbf{y}^\top \mathbf{y} - \hat{\beta}^\top \mathbf{X}^\top \mathbf{y}}{n - p}$$

or we can use $\text{SSE} = \mathbf{y}^\top \mathbf{Q}_2 \mathbf{Q}_2^\top \mathbf{y}$. Since this is a simple regression problem, $p = 2$. The \mathbf{Q}_2 matrix is obtained from the QR decomposition as

```
Q2 <- qr.Q(X.qr, complete = TRUE) [, -c(1, 2)]
dim(Q2) # this confirms the dimensions of Q2
## [1] 8 6
```

The SSE is calculated from

```
Q2y <- t(Q2) %*% ht.df$height
SSE <- sum(Q2y^2)
SSE
## [1] 103.2
```

Then we can obtain the MSE by dividing the SSE by $n - p$:

```
MSE <- SSE / (nrow(ht.df) - 2)
MSE
## [1] 17.2
```

Standard error estimates for $\hat{\beta}_1$ and $\hat{\beta}_0$ can also be calculated using the simple regression formulas or from the formulas developed using the QR approach. That is, we estimate the standard error of the slope estimator with

$$\sqrt{\frac{\text{MSE}}{S_{xx}}}$$

Similarly, the standard error (s.e.) of $\hat{\beta}_0$ can be estimated by

$$\sqrt{\text{MSE} \left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}} \right)}$$

or we can obtain both from $\sqrt{\text{MSE} C_{ii}}$, $i = 0, 1$, where \mathbf{C} denotes the diagonal of $\mathbf{U}^{-1}(\mathbf{U}^{-1})^\top$. The latter approach proceeds as follows

```

Uinv <- solve(U) # invert U
UinvUinvT <- Uinv%*%t(Uinv)
C <- diag(UinvUinvT)
sqrt(MSE*C) # intercept and slope SES
## [1] 1.854724 3.028751

```

These results can be compared with the output from `lm()` which is obtained in the next section.

6.1.2 A test for a difference in the means

The t-test that we used to check whether the regression slope is nonzero can be used to test whether there is a difference in mean height:

$$H_0 : \beta_1 = 0 \quad H_1 : \beta_1 \neq 0$$

$$t_0 = \frac{\hat{\beta}_1}{s.e.}$$

$$\text{p-value} = 2P(t_2 > |t_0|)$$

6.1.3 Implementation with `lm()`

As we have seen, the analysis proceeds as in simple linear regression, so we should be able to use the `lm()` function in R to do the computations. The only new feature is the use of the `factor()` function to set up the indicator variable.

Example – height versus sex

```

ht.df$Sex <- factor(ht.df$sex, levels=c("M", "F"))
y.lm <- lm(height ~ Sex, data = ht.df)
summary(y.lm)

##
## Call:
## lm(formula = height ~ Sex, data = ht.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.60   -1.70    0.20    2.15    5.00
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 170.600     1.855  91.981 1.11e-10 ***
## SexF        -15.600     3.029  -5.151  0.00211 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.147 on 6 degrees of freedom
## Multiple R-squared:  0.8155, Adjusted R-squared:  0.7848
## F-statistic: 26.53 on 1 and 6 DF,  p-value: 0.002114

```

Note the use of the `levels` argument to the `factor()` function. The default would have made F the first factor and M the second factor. The resulting model would be equivalent to what we obtained above, but with a different parametrization. In other words, the coded factor for sex would be place a value of 1 on M and 0 on F so that the interpretation of x in the final model would be different, and the regression coefficients would be different. The fitted values would be the same, of course.

The estimated standard deviation of the noise is 4.147, and the standard error of the slope is 3.029. . We see that there is weak evidence against H_0 . This should come as no real surprise, since the sample sizes are very small in this example. Larger samples would likely provide us with stronger evidence against the null hypothesis for data of the kind considered here.

This is a simple example of the regression approach to the Analysis of Variance.

6.1.4 Analysis of variance: comparison of several treatments

The analysis of variance is usually employed to test for mean differences among several samples. Each sample is referred to as a treatment group. In the above example, we compared the mean heights for two samples (males and females). In the case where there are k different groups, one fits the regression model

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_{k-1} x_{k-1} + \varepsilon$$

where the x_j 's are indicator variables.

Example

Suppose there are $k = 3$ treatment groups: A, B and C. Set

$$x_1 = \begin{cases} 1, & \text{if in A} \\ 0, & \text{otherwise} \end{cases}$$

and

$$x_2 = \begin{cases} 1, & \text{if in B} \\ 0, & \text{otherwise} \end{cases}$$

Note that $x_1 = x_2 = 0$ for an observation from group C.

Suppose 2 observations are taken from each group, and the first 2 observations are from group A, and the last 2 observations are from C.. Then the relevant model is

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon,$$

and the design matrix \mathbf{X} takes the form

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

Example – car colour

Customers were asked to rate an automobile model on a number of factors, and the average was recorded in the variable `rating`. The colour was also recorded. The data are below:

```
##   colour rating
## 1   red    7.6
## 2 yellow   7.9
## 3 blue    7.8
## 4   red    8.8
## 5 yellow   7.7
## 6 blue    7.5
```

Are rating and colour related? We can test this by converting `colour` to a factor, which is equivalent to setting up two indicator variables, one for red and one for yellow. In fact, this is done automatically by the `lm()` any time a character vector is used in a model formula. Therefore, we can simply type the following in R:

```
rating.lm <- lm(rating ~ colour, data = colour.test)
summary(rating.lm)

##
## Call:
## lm(formula = rating ~ colour, data = colour.test)
##
## Residuals:
##     1      2      3      4      5      6 
## -0.60  0.10  0.15  0.60 -0.10 -0.15 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 7.6500    0.3617  21.150 0.000231 ***
## colourred   0.5500    0.5115   1.075 0.361062    
## colouryellow 0.1500    0.5115   0.293 0.788456    
## ---      
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5115 on 3 degrees of freedom
## Multiple R-squared:  0.2917, Adjusted R-squared:  -0.1805 
## F-statistic: 0.6178 on 2 and 3 DF,  p-value: 0.5961
```

The output is the same as the output we would have obtained if we had set `colourred` to be the indicator of red and `colouryellow` to be the indicator of yellow.

Because the p-value is large here, we conclude that we have insufficient evidence to say that rating is related to colour.

6.2 Analysis of Covariance

The analysis of covariance (ANCOVA) allows us to model continuous responses as linear functions of continuous and categorical covariates. In this way, it can be viewed as a relatively straightforward extension of multiple regression. It can also be viewed as an extension of ANOVA whereby there is a blocking factor which is continuously measured. For a categorical covariate with two levels, there would be two lines in the regression: parallel if there is no interaction effect; two different slopes if there is an interaction effect.

6.2.1 Regression with indicator variables

Analysis of covariance is essentially a version of regression analysis in which indicator variables or dummy variables are included in the regression model. As in the case of analysis of variance via regression, it all comes down to specifying the design matrix \mathbf{X} in the model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \epsilon$$

and carrying out the least-squares calculations.

Determining the \mathbf{X} matrix is a straightforward matter, once the model has been identified. Two or more lines are actually being fit in these analyses. If there is an interaction effect between the categorical predictor and the continuous variable, the lines will have different slopes. Where there are no interactions, the lines will be parallel, but with different intercepts.

Example – weight versus height and sex

Suppose, in addition to the height and sex observations, we also have weight measurements.

```
## [1] 64 60 65 63 68 69 70 58
```

A linear model relating weight to height and sex is

$$w = \beta_0 + \beta_1 h + \beta_2 s + \varepsilon$$

where s denotes the indicator of female versus male.

This model will give two parallel lines, one for each sex. The design matrix is

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 160 \\ 1 & 1 & 150 \\ 1 & 0 & 175 \\ 1 & 0 & 165 \\ 1 & 0 & 170 \\ 1 & 0 & 172 \\ 1 & 0 & 171 \\ 1 & 1 & 155 \end{bmatrix}.$$

A model which could possibly give two different slopes for the male and female lines is

$$w = \beta_0 + \beta_1 h + \beta_2 s + \beta_3 hs + \varepsilon.$$

The design matrix is

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 160 & 160 \\ 1 & 1 & 150 & 150 \\ 1 & 0 & 175 & 0 \\ 1 & 0 & 165 & 0 \\ 1 & 0 & 170 & 0 \\ 1 & 0 & 172 & 0 \\ 1 & 0 & 171 & 0 \\ 1 & 1 & 155 & 155 \end{bmatrix}.$$

6.2.2 Estimation, testing and prediction; implementation in R

Again, the technology available for computing estimates by least-squares and constructing confidence intervals and prediction intervals is essentially the same as before. The `lm()` function in R can be used to carry out all computations.

If y is a continuous response, x_1 is a continuous predictor and x_2 is a categorical predictor or factor, we can use the command

```
lm(y ~ x1 + x2)
```

If we want to include the interaction between x_1 and x_2 , we use

```
lm(y ~ x1 + x2 + x1:x2)
```

or

```
lm(y ~ x1*x2)
```

The output from `summary()` can be interpreted in the usual manner.

Example – weight versus sex and height

We can model weight versus height and sex, using two parallel lines as follows:

```
weight.lm <- lm(weight ~ sex + height)
summary(weight.lm)

## 
## Call:
## lm(formula = weight ~ sex + height)
## 
## Residuals:
##      1      2      3      4      5      6      7      8 
## 1.541  1.126 -3.578 -1.992  1.215  1.498  2.857 -2.667 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 5.0950    42.8620   0.119   0.910    
## sexM        0.7403    4.7734   0.155   0.883    
## height      0.3585    0.2763   1.297   0.251    
## 
## Residual standard error: 2.807 on 5 degrees of freedom
## Multiple R-squared:  0.6919, Adjusted R-squared:  0.5686 
## F-statistic: 5.614 on 2 and 5 DF,  p-value: 0.0527
```

The line for females can be read directly from the output as

$$\hat{w} = 5.095 + 0.3585h$$

while the line for males has an adjusted intercept which is obtained by adding the coefficient of the sex indicator:

$$\hat{w} = 5.8353 + 0.3585h$$

The reader will be asked to include an interaction term in the model in the exercises.

Example – tooth growth data

The ToothGrowth data frame in R concerns the length of odontoblasts, cells connected with the growth of teeth, in a sample of 60 guinea pigs. One of three dose levels of vitamin C were supplied to the guinea pigs in one of two forms: supp = VC refers to ascorbic acid and supp = OJ refers to orange juice. Figure 6.1 uses a conditioning plot to display the data.

```
library(lattice)
xyplot(len ~ sqrt(dose) | supp, data = ToothGrowth, ylab="length")
```

The figure shows that there are possibly two different lines relating length to vitamin C dose; it is possible that there is a treatment effect, that is, an effect due to the different supplements used.. We use lm() to check this, first by allowing for two intercepts and two slopes:

```
TG.lm <- lm(len ~ sqrt(dose) * supp, data = ToothGrowth)
summary(TG.lm)

## 
## Call:
## lm(formula = len ~ sqrt(dose) * supp, data = ToothGrowth)
## 
```

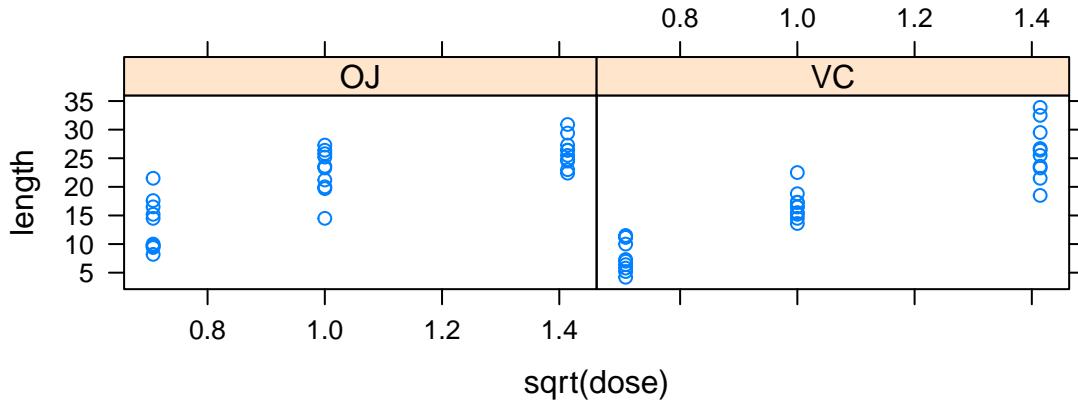


Figure 6.1: Tooth growth data: length of tooth versus square root of vitamin C dose, for each of the two treatment methods.

```
## Residuals:
##      Min     1Q   Median     3Q    Max
## -7.9867 -2.6768 -0.1716  2.7380  7.4133
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                2.478     2.628   0.943  0.34976
## sqrt(dose)                 17.479    2.433   7.185 1.71e-09 ***
## suppVC                   -12.024    3.716  -3.236  0.00204 **
## sqrt(dose):suppVC          8.000    3.441   2.325  0.02370 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.865 on 56 degrees of freedom
## Multiple R-squared:  0.7576, Adjusted R-squared:  0.7446
## F-statistic: 58.35 on 3 and 56 DF,  p-value: < 2.2e-16
```

Looking at the interaction between the square root of dose and supp, we see a fairly small p -value which is highly suggestive of different slopes. There is no reason to consider the model without different slopes (which would have been obtained by replacing the * with +). The value 8 indicates that for VC, the slope is 8 units higher than for OJ: $17.48 + 8 = 25.48$. The intercept for VC is 12.02 units lower: $2.478 - 12.024 = -9.546$.

We can graphically summarize the data with a simple scatterplot with the lines overlaid:

```
par(mar=c(4, 4, .5, .1))
plot(len ~ sqrt(dose), pch = as.numeric(supp), data = ToothGrowth, ylab="length")
abline(2.478, 17.479) # OJ line
abline(2.478-12.024, 17.479 + 8, lty=2) # VC line, dashed
```

6.2.3 Two interpretations of the analysis of covariance

The analysis of covariance has a similar-sounding name to the analysis of variance, and this is not an accident. One interpretation of the analysis of covariance is that the response could vary between treatment levels, but where

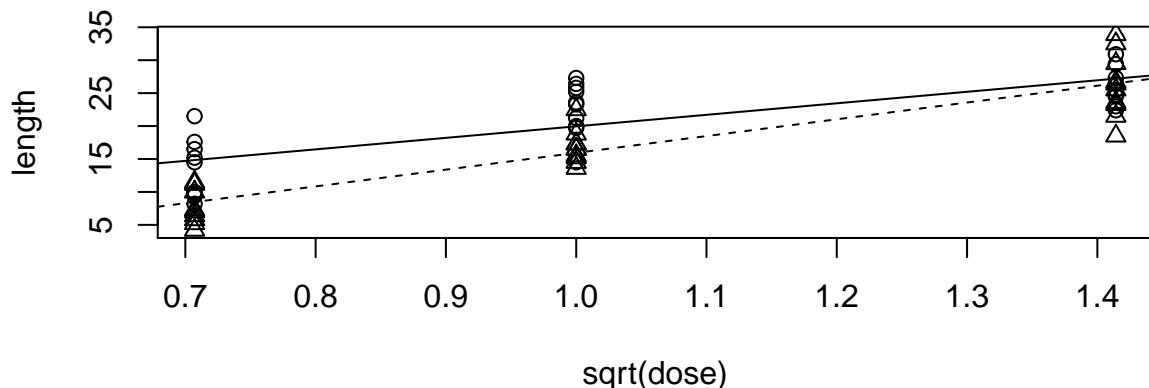


Figure 6.2: Tooth growth data: length of tooth versus square root of vitamin C dose, for each of the two treatment methods. The solid line corresponds to the orange juice treatment, and the dashed line corresponds to the ascorbic acid treatment

there is a possible continuous factor which needs to be accounted for. For example, we might be interested in comparing weights of males and females, but taking account of individual heights, i.e. blocking with respect to the factor height, gives more precise comparisons between the sexes.

On the other hand, we might be primarily interested in the relation between two continuous variables. Can we predict weight from height? This is an ordinary regression problem, but the answer can be made more precise by taking into account sex. In other words, including sex as a dummy variable in the regression model reduces the residual standard error and leads to more precise estimates of the regression coefficients of interest.

6.3 Standardized Regression and Multicollinearity

The way in which the data are collected can have an impact on the accuracy and stability of parameter estimates in regression models. When the columns of the design matrix are almost linearly dependent, a common occurrence in observational data, standard error estimates can be inflated. In other words, the uncertainty in the regression coefficient estimates will be larger than it would have been for a similar sample size and error variance, for predictor variables that are closer to orthogonality.

In this section, we show how this problem of variance inflation can arise, how to check for it, using variance inflation factors and make a few suggestions as to how to mitigate the problem.

In order to understand how standard error estimates can be inflated, we need a way to standardize estimates so that the quantities are being compared are on the same scale.

Choice of scale affects coefficient values in regression

The response variable Y and the explanatory variables x_1, x_2, \dots, x_k are based on certain scales of measurement. Thus, the regression coefficients depend on these measurement scales.

Example – grams versus milligrams

The `litters` data frame in the `DAAG` package contains measurements of the weights of the bodies and brains for baby mice born in different sized litters.

```
library(DAAG)
summary(litters)

##      lsize          bodywt         brainwt
##  Min.   : 3.0   Min.   :5.450   Min.   :0.3680
##  1st Qu.: 5.0   1st Qu.:6.641   1st Qu.:0.4065
##  Median : 7.5   Median :7.330   Median :0.4155
```

```
##   Mean    : 7.5    Mean    :7.748    Mean    :0.4168
##   3rd Qu.:10.0   3rd Qu.:8.926   3rd Qu.:0.4333
##   Max.    :12.0   Max.    :9.780   Max.    :0.4440
```

The `brainwt` and `bodywt` columns are measured in grams. We will construct two additional columns: `brainwtmg` and `bodywtmg` which will contain the weights measured in milligrams:

```
litters$brainwtmg <- 1000*litters$brainwt
litters$bodywtmg <- 1000*litters$bodywt
summary(litters)

##      lsize          bodywt        brainwt       brainwtmg
##  Min.   : 3.0   Min.   :5.450   Min.   :0.3680   Min.   :368.0
##  1st Qu.: 5.0   1st Qu.:6.641   1st Qu.:0.4065   1st Qu.:406.5
##  Median : 7.5   Median :7.330   Median :0.4155   Median :415.5
##  Mean   : 7.5   Mean   :7.748   Mean   :0.4168   Mean   :416.8
##  3rd Qu.:10.0   3rd Qu.:8.926   3rd Qu.:0.4333   3rd Qu.:433.2
##  Max.   :12.0   Max.   :9.780   Max.   :0.4440   Max.   :444.0
##      bodywtmg
##  Min.   :5450
##  1st Qu.:6641
##  Median :7330
##  Mean   :7748
##  3rd Qu.:8926
##  Max.   :9780
```

Interest centers on the relation between brain weight and body weight, where litter size is accounted for. If we do the regression analysis in g units, we have

```
littersg.lm<-lm(brainwt~bodywt+lsize,data=litters)
summary(littersg.lm)

##
## Call:
## lm(formula = brainwt ~ bodywt + lsize, data = litters)
##
## Residuals:
##      Min      1Q      Median      3Q      Max
## -0.0230005 -0.0098821  0.0004512  0.0092036  0.0180760
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.178247  0.075323  2.366  0.03010 *
## bodywt      0.024306  0.006779  3.586  0.00228 **
## lsize       0.006690  0.003132  2.136  0.04751 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01195 on 17 degrees of freedom
## Multiple R-squared:  0.6505, Adjusted R-squared:  0.6094
## F-statistic: 15.82 on 2 and 17 DF,  p-value: 0.0001315
```

$$\hat{y} = .178 + .0243x_1 + .00669x_2$$

In milligram units, we have

```

littersmg.lm<-lm(brainwtmg~bodywtmg+lsiz,
                     data=litters)
summary(littersmg.lm)

##
## Call:
## lm(formula = brainwtmg ~ bodywtmg + lsiz, data = litters)
##
## Residuals:
##       Min     1Q   Median     3Q    Max
## -23.0005 -9.8821  0.4512  9.2036 18.0760
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.782e+02 7.532e+01  2.366 0.03010 *
## bodywtmg    2.431e-02 6.779e-03  3.586 0.00228 **
## lsiz        6.690e+00 3.132e+00  2.136 0.04751 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.95 on 17 degrees of freedom
## Multiple R-squared:  0.6505, Adjusted R-squared:  0.6094
## F-statistic: 15.82 on 2 and 17 DF,  p-value: 0.0001315

```

$$\hat{y} = 178 + .0243x_1 + 6.69x_2$$

Therefore, the relative sizes of the coefficients of x_1 and x_2 depend upon what scale is used.

This leads to potential difficulties in interpreting the relationship between brain weight and body weight, since the coefficient of body weight looks very small in the first instance and relatively large in the second instance.

6.3.1 Standardized regression

Standardized regression offers a way to determine the relation between the variables without depending on scale. Physicists would refer to this as the use of dimensionless quantities.

Analogous to the standardizing that one does to compute a Z -score or t -score, we standardize both the response variable values and the predictor variable values:

$$w_{ij} = \frac{x_{ij} - \bar{x}_j}{\sqrt{S_{x_j x_j}}}$$

where

$$S_{x_j x_j} = \sum_{i=1}^n (x_{ij} - \bar{x}_j)^2 \text{ and}$$

$$y_i^* = \frac{y_i - \bar{y}}{\sqrt{S_{YY}}}$$

$$\text{where } S_{YY} = SS_T = \sum_{i=1}^n (y_i - \bar{y})^2$$

This is called unit length scaling.¹

¹Unit normal scaling is similar, but use $S_*/(n - 1)$ in place of S_* .

Example - 2 predictor variables

To see what is going on, we consider a linear model with two predictor variables x_1 and x_2 :

$$y_j = \beta_0 + \beta_1 x_{j1} + \beta_2 x_{j2} + \varepsilon_j$$

for $j = 1, 2, \dots, n$. Taking the average over the n observations gives

$$\bar{y} = \beta_0 + \beta_1 \bar{x}_1 + \beta_2 \bar{x}_2 + \bar{\varepsilon}.$$

Subtracting the average from the j th equation, we have

$$y_j - \bar{y} = \beta_1(x_{j1} - \bar{x}_1) + \beta_2(x_{j2} - \bar{x}_2) + \varepsilon_j - \bar{\varepsilon}$$

Standardizing by dividing by $\sqrt{S_{YY}}$ gives

$$\frac{y_j - \bar{y}}{\sqrt{S_{YY}}} = \frac{\beta_1}{\sqrt{S_{YY}}}(x_{j1} - \bar{x}_1) + \frac{\beta_2}{\sqrt{S_{YY}}}(x_{j2} - \bar{x}_2) + \frac{\varepsilon_j - \bar{\varepsilon}}{\sqrt{S_{YY}}}$$

Setting

$$b_j = \beta_j \left(\frac{S_{x_j x_j}}{S_{YY}} \right)^{1/2} \quad (6.1)$$

and using the definitions of the standardized variables given earlier, together with

$$\varepsilon_j^* = \frac{\varepsilon_j - \bar{\varepsilon}}{\sqrt{S_{YY}}},$$

we have

$$y_j^* = b_1 w_{j1} + b_2 w_{j2} + \varepsilon_j^*$$

Fitting the standardized regression model

The next step is to fit the model

$$y_i^* = b_1 w_{i1} + \dots + b_k w_{ik} + \varepsilon^*$$

which, written in matrix form is

$$\mathbf{y}^* = \mathbf{W}\mathbf{b} + \boldsymbol{\epsilon}^*.$$

We could use the QR decomposition method directly, but in order to see the variance inflation problem associated with multicollinearity, we first set up the normal equations – the traditional approach to solving least-squares regression problems. A simple way to get to the normal equations is to pre-multiply the model equation by \mathbf{W}^\top :

$$\mathbf{W}^\top \mathbf{y}^* = \mathbf{W}^\top \mathbf{W}\mathbf{b} + \mathbf{W}^\top \boldsymbol{\epsilon}^*.$$

Noting that the left-hand-side of this is an estimate of its expected values, and that the expected value of the right-hand-side is approximately $\mathbf{W}^\top \mathbf{W}\mathbf{b}$, due to the error mean being 0, we have the estimating equation

$$\mathbf{W}^\top \mathbf{y}^* = \mathbf{W}^\top \mathbf{W}\mathbf{b}.$$

This is a $p \times p$ system of linear equations in \mathbf{b} which can be solved using the QR method, for example. Mathematically, we have

$$\hat{\mathbf{b}} = (\mathbf{W}^\top \mathbf{W})^{-1} \mathbf{W}^\top \mathbf{y}^*.$$

Observe that

$$(\mathbf{W}^\top \mathbf{W})_{jk} = r_{jk}$$

where r_{jk} is the correlation between \mathbf{x}_j and \mathbf{x}_k , and

$$(\mathbf{W}^\top \mathbf{y}^*)_j = r_{jy}$$

where r_{jy} is the correlation between \mathbf{x}_j and \mathbf{y} .

Relations between the scaled and unscaled models

Working from equations like (6.1), we can translate back and forth between the standardized and the original coefficients estimates using

$$\hat{\beta}_j = \bar{b}_j \left(\frac{S_{YY}}{S_{x_j x_j}} \right)^{1/2}$$

and

$$\hat{\beta}_0 = \bar{y} - \sum_{j=1}^k \hat{\beta}_j \bar{x}_j.$$

Running the calculations in R

The following function can be used to standardize any variable, by first calculating its mean, and the numerator of its variance. This function can be applied to the response variable as well as to any predictor variable.

```
standardize <- function (x) {
  xbar <- mean(x)
  ss <- sum((x-xbar)^2)
  (x - xbar) / sqrt(ss)
}
```

Example – litters data

We can use the `sapply()` function to apply the `standardize()` function directly to all columns of the litters data at once:

```
litters.std <- sapply(litters, standardize)
```

The first few rows of the result are as follows:

```
head(litters.std)

##          lsize    bodywt    brainwt   brainwtmg   bodywtmg
## [1,] -0.3503245 0.2862613 0.326875158 0.326875158 0.2862613
## [2,] -0.3503245 0.3423679 0.230911809 0.230911809 0.3423679
## [3,] -0.2724746 0.2370628 0.002998855 0.002998855 0.2370628
## [4,] -0.2724746 0.3142304 0.146943878 0.146943878 0.3142304
## [5,] -0.1946247 0.1856739 0.098962204 0.098962204 0.1856739
## [6,] -0.1946247 0.3137249 0.206920971 0.206920971 0.3137249
```

Note that having appended the additional columns on the other scale, these columns have been standardized as well. As expected, the standardized values for both versions of body weight are identical, and the same can be said for both versions of litter size.

We can now carry out the standardized regression calculation using the `lm()` function:

```
litters.std <- data.frame(litters.std)
litters.std.lm <- lm(brainwt ~ bodywt + lsize - 1, data = litters.std)
```

It was necessary to convert the `litters.std` matrix to a data frame before `lm()` would do the calculation. We can now look at the summary output, as usual:

```
summary(litters.std.lm)
```

```

## 
## Call:
## lm(formula = brainwt ~ bodywt + lsize - 1, data = litters.std)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.275901 -0.118540  0.005413  0.110401  0.216829
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## bodywt     1.730     0.469    3.690  0.00168 **  
## lsize      1.031     0.469    2.198  0.04127 *   
## ---      
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.1393 on 18 degrees of freedom
## Multiple R-squared:  0.6505, Adjusted R-squared:  0.6117 
## F-statistic: 16.75 on 2 and 18 DF,  p-value: 7.774e-05

```

Thus, the regression equation relating brain weight to litter size and body weight in standardized units is

$$\hat{y}^* = 1.73w_1 + 1.031w_2$$

*Correlations and the **W** matrix*

We suggested before that the entries of the $\mathbf{W}^\top \mathbf{W}$ matrix are equal to the correlations among the x variables. We can check this as follows:

```

W<-model.matrix(litters.std.lm)
t(W) %*% W

##           bodywt      lsize
## bodywt  1.0000000 -0.9548494
## lsize   -0.9548494  1.0000000

```

```

cor(litters$bodywt,litters$lsize)

## [1] -0.9548494

```

We also suggested that the entries of $\mathbf{W}^\top \mathbf{y}^*$ are equal to the correlations between the x variables and y :

```

t(W) %*% litters.std$brainwt

##          [,1]
## bodywt  0.7461485
## lsize   -0.6214719

```

```

cor(litters$bodywt,litters$brainwt)

## [1] 0.7461485

```

```
cor(litters$lsizel, litters$brainwt)
## [1] -0.6214719
```

These calculations are agreement with what the theory predicts.

6.3.2 Multicollinearity and variance inflation

If the explanatory variable vectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ are all orthogonal, then

$$\mathbf{W}^\top \mathbf{W} = \mathbf{I}$$

and

$$(\mathbf{W}^\top \mathbf{W})^{-1} = \mathbf{I}.$$

The ε^* 's are approximately independent with variance σ^{*2} . Therefore, we have the approximate result

$$\text{Var}(\mathbf{b}) \doteq \sigma^{*2} \mathbf{I}$$

If there is linear dependence amongst the x variables, then some or all of the off-diagonal elements of $\mathbf{W}^\top \mathbf{W}$ will be nonzero, causing $(\mathbf{W}^\top \mathbf{W})^{-1}$ to differ from \mathbf{I} . Some of the elements of the diagonal of $(\mathbf{W}^\top \mathbf{W})^{-1}$ will increase, and none will decrease. Thus, the variances of some or all of the b_j estimates will increase:

$$\text{Var}(b_j) = (\mathbf{W}^\top \mathbf{W})_{jj}^{-1} \sigma^{*2}$$

We say that the variance inflation factor for the j th coefficient is

$$\text{VIF}_j = (\mathbf{W}^\top \mathbf{W})_{jj}^{-1}.$$

If VIF_j is larger than about 10 for some j , this indicates serious multicollinearity: there is too much linear dependence among some or all of the x variables.

Multicollinearity is thus the result of near-linear dependencies among the predictor variables.

Causes of multicollinearity

Poor study design can lead to multicollinearity. Often regression models are fit to data collected in an observational study; since the modeller has no control over the design points, the predictor variables can be very dependent upon each other. In many situations, there are **constraints** on the population which cause multicollinearity.

The **choice of model** can lead to multicollinearity.

Example.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

The columns of the X matrix are far from orthogonal. It is sometimes possible to re-formulate the model as

$$y = \beta_0 + \beta_1 P_1(x) + \beta_2 P_2(x) + \varepsilon$$

where $P_1(x)$ and $P_2(x)$ are orthogonal polynomials. This avoids any multicollinearity. Even if orthogonality is not possible, some formulations can reduce multicollinearity.

Over-parametrization can lead to multicollinearity. Having more predictor variables than observations leads to a very serious multicollinearity problem.

6.3.3 Effects of multicollinearity

The value of the estimate of the regression parameter (with large VIF) may vary substantially, depending on what other x -variables are added to the regression equation. The standard errors of the estimates of the coefficients will be large. We could decide that they are not significant when they really are. Or estimates will be imprecise since the confidence intervals are wide.

Example

```

litters.lm <- lm(brainwt ~ bodywt + lsize,
  data = litters)
vif(litters.lm)

## bodywt  lsize
##  11.33  11.33

```

This is in agreement with the scatter plot of `bodywt` versus `lsize`. (Look at Figure 6.3 and compare how the relation between `bodywt` and `lsize` with the relations between `brainwt` and the 2 regressors.)

```
pairs(litters)
```

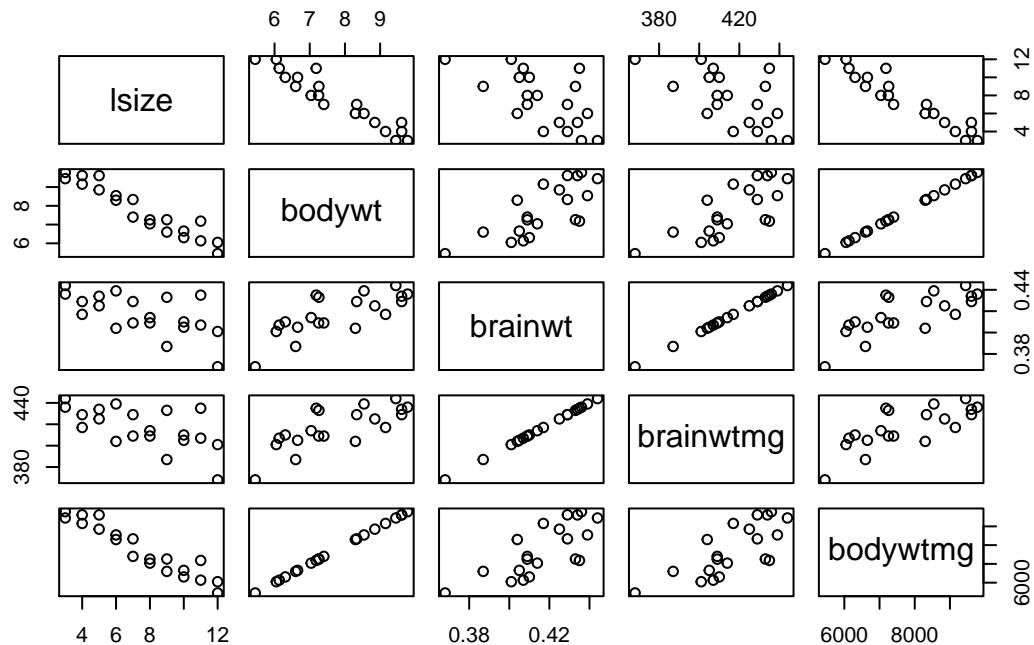


Figure 6.3: Pairwise scatter plots for litters data.

In the litters data, the predictors body weight and litter size happen to be highly correlated with each other. The data collector had no control over these variables. It would be difficult to find data where body weight and litter size were not so dependent. If the modeller can control the design, the predictor variables can be chosen orthogonal to each other (or nearly so).

Example – properly designed experimental data

The data in `table.b7` of the *MPV* package were collected in a designed experiment.

```

library(MPV)
summary(table.b7)

```

```

##      x1          x2          x3          x4
##  Min. :415.0   Min. :25   Min. : 5   Min. :40
##  1st Qu.:415.0 1st Qu.:25  1st Qu.: 5  1st Qu.:40
##  Median :482.5 Median :60  Median :10  Median :50
##  Mean   :482.5  Mean  :60  Mean   :10  Mean   :50
##  3rd Qu.:550.0 3rd Qu.:95 3rd Qu.:15 3rd Qu.:60
##  Max.  :550.0  Max. :95  Max. :15  Max. :60
##      x5          y
##  Min. :1.280   Min. :21.00
##  1st Qu.:1.280 1st Qu.:30.75
##  Median :2.665  Median :58.50
##  Mean   :2.665  Mean   :54.25
##  3rd Qu.:4.050 3rd Qu.:71.75
##  Max.  :4.050  Max.  :99.00

```

```

peanuts.lm <- lm(y ~ x1 + x2 + x3 + x4 + x5, data=table.b7)
vif(peanuts.lm)

## x1 x2 x3 x4 x5
## 1 1 1 1 1

```

There is no multicollinearity; the predictors were chosen to be mutually orthogonal. The modeller could control this experiment.

Possible remedies for multicollinearity

Several methods have been suggested for dealing with multicollinearity including:

- collecting additional data
- re-parametrizing the model
- variable selection
- principal components regression (a topic in multivariate analysis)
- ridge regression

6.4 Regression with counts

On a university campus there are a number of areas designated for smoking. Outside of those areas, smoking is not permitted. One of the smoking areas is towards the north end of the campus near some parking lots and a large walkway towards one of the residences.

Along the walkway, cigarette butts are visible in the nearby grass. Numbers of cigarette butts were counted at various distances from the smoking area in $200 \times 80\text{cm}^2$ quadrats located just west of the walkway. The data are contained in a data frame called `cigbutts` that can be found in the *MPV* package.

```

library(MPV)
summary(cigbutts)

##      distance      count
##  Min.   : 200   Min.   : 0.000
##  1st Qu.: 900   1st Qu.: 1.000
##  Median :1600   Median : 2.000
##  Mean   :1600   Mean   : 5.733
##  3rd Qu.:2300   3rd Qu.: 9.500
##  Max.  :3000   Max.  :28.000

```

A reasonable way to model these data (interest would center on how the distribution of cigarette butts changes as a function of distance from the smoking area) is to use Poisson regression, and this will be done – later. First, we will see what happens when we use the least-squares technology that we have discussed already. Thus, plot the data to visualize the relationship, as in Figure 6.4.

```
par(mfrow=c(1,2), mar=c(4, 4, .1, .1)) # create a 1x2 layout of plots
plot(count ~ distance, data = cigbutts)
plot(log(count+1) ~ distance, data = cigbutts)
```

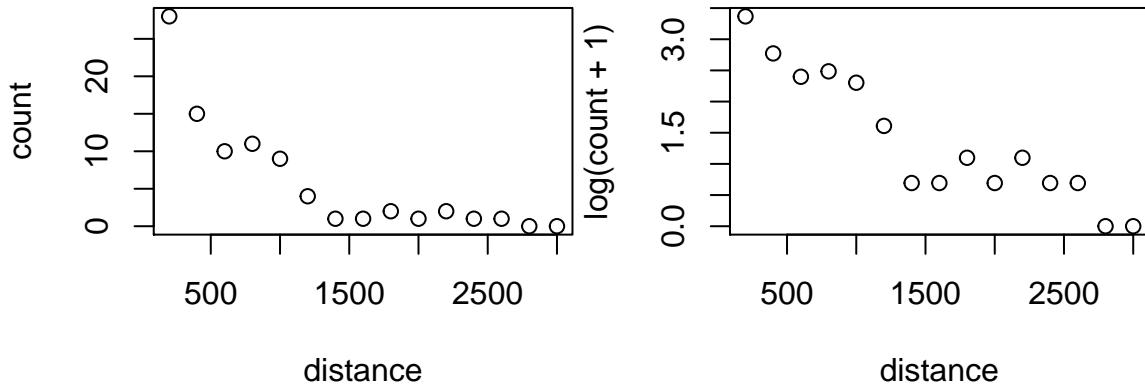


Figure 6.4: Numbers of cigarette butts observed at various distances from a designated smoking area.

The left panel of the figure shows a clear rapid decrease at short distances followed by an almost constant pattern (at or near 0) at larger distances. The second panel shows the counts on the log scale (note that 1 was added to the counts, so that the log is always defined). On the log scale, the relationship is almost linear.

Fitting the linear model using ordinary least-squares:

```
cig.lm <- lm(log(count+1) ~ distance, data = cigbutts)
cig.lm

##
## Call:
## lm(formula = log(count + 1) ~ distance, data = cigbutts)
##
## Coefficients:
## (Intercept)      distance
##       3.114218     -0.001088
```

we obtain the model which results in the solid curve in Figure 6.5. Note that the model we have really fit is the log-linear model:

$$z = \beta_0 + \beta_1 x + \varepsilon \quad (6.2)$$

where $z = \log(y+1)$, where y is an observed count. In order to draw the curve in Figure 6.5, we back-transformed using the function $g(z) = e^z - 1$. It is important to keep in mind that the curve does not represent an estimate of the expected value of $E[z]$ but rather an estimate of $e^{\beta_0 + \beta_1 x} - 1$. As we now show, these quantities are not the same.

On the original scale, the model corresponding to (6.2) is the multiplicative error model:

$$y = e^{\beta_0 + \beta_1 x} \epsilon - 1 \quad (6.3)$$

where $\epsilon = e^\varepsilon$. Note that the expected value of this is

$$E[y] = e^{\beta_0 + \beta_1 x} E[e^\varepsilon] - 1.$$

If ε had been a normal random variable with mean 0 and variance σ^2 , we would have had

$$E[e^\varepsilon] = e^{\frac{1}{2}\sigma^2}$$

which is larger than 1. Thus, in this case, the plotted curve would be underestimating the expected value, and the degree of underestimation is larger when the expected counts are larger. Because of the count nature of the data, the distribution of ε is not normal; the actual form of the distribution of ε is not convenient to work with, but we can estimate $E[e^\varepsilon]$ using the residuals from the fitted log-linear model.

```
mean(exp(resid(cig.lm)))
```

In this case, we find that the value is estimated as 1.064. A bootstrap could be used to estimate the standard error of this quantity.

6.4.1 Nonlinear least-squares

If we really believe the model for the counts is

$$y = e^{\beta_0 + \beta_1 x} + \varepsilon \quad (6.4)$$

we cannot use linear least-squares to estimate β_0 and β_1 . However, least-squares can still be used, in principle, to solve minimization problems of the form

$$\sum_{j=1}^n (y_j - g(x_j; \beta))^2$$

where $g(x)$ is a function of x , known up to a parameter vector β . For the case we are interested in, $g(x) = e^{\beta_0 + \beta_1 x}$. Such nonlinear least-squares problems can be solved using the `nls()` function.

Basic usage is as follows:

```
cig.nls <- nls(count ~ exp(b0 + b1*distance), data = cigbutts,
  start = c(b0 = 3.1, b1 = -.001) )
coef(cig.nls)

##          b0          b1
##  3.63510 -0.00186
```

Note, in particular, that the functional form of $g(x)$ needs to be explicitly specified, including the way in which the parameter vector β enters the model. The standard method for solving least-squares problems is the iterative Gauss-Newton algorithm which is described succinctly by Wood (2006).² As such, it requires a starting value, and in this case, we can use the estimated values of β_0 and β_1 coming from the fitted loglinear model. Since the convergence of the iterative method depends on how far the initial guesses are from the truth, it is wise to avoid using arbitrary initial values, and to use preliminary estimates, possibly coming from the fitting of simpler models.

The following code has been used to obtain Figure 6.5, which compares the log-linear and nonlinear approaches to fitting the exponential model to the cigarette count data.

```
par(mar=c(4, 4, .1, .1))
plot(count ~ distance, data = cigbutts)
a0 <- coef(cig.lm)[1]
a1 <- coef(cig.lm)[2]
curve(exp(a0+a1*x)-1, from = 0, to = 3000, add = TRUE)
b0 <- coef(cig.nls)[1]
b1 <- coef(cig.nls)[2]
curve(exp(b0+b1*x), from = 0, to = 3000, add = TRUE, lty=2)
```

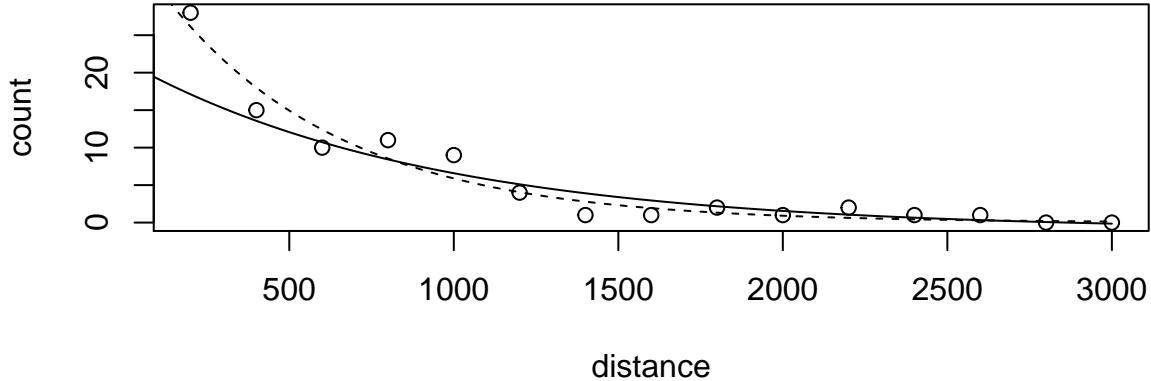


Figure 6.5: Comparison of linear and nonlinear model fits to the cigarette butts data set. The solid curve is from the least-squares fit and the dashed curve is from the nonlinear least-squares fit.

The dashed curve in Figure 6.5 represents the nonlinear least-squares fit of the exponential model. Note that there is a substantial difference in the two curves: we had explained earlier that there is a systematic underestimation in the first curve, due to the use of the back-transformation; using nonlinear least-squares avoids this device, and provides us with a curve that tracks the data more effectively.

There are other issues with the data, however, that are not treated satisfactorily by the nonlinear least-squares approach. In particular, when dealing with count data, an error variance that does not depend on the mean would be unexpected. For example, a standard model for count data is the Poisson distribution where the mean and variance are equal.

In the model (6.4), we said little about the error term ϵ , and we only exhibited the estimates of β_0 and β_1 . As we have seen, in order for these estimates to be unbiased, the error term should have mean 0. However, if something like a Poisson distribution is at all appropriate, it is impossible to believe that ϵ has a constant variance. Thus, we have not considered inference, and we should not, without adjusting the model further.

6.4.2 Weighted least-squares

If we truly believe the Poisson assumption, we are better off considering a generalized linear model which makes explicit use of the Poisson distribution. Otherwise, if we want to remain in the least-squares framework, one way of coping with a nonconstant variance is to employ weighted least-squares.

Recall the form of the linear model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

but this time, suppose that the variance-covariance matrix of the vector $\boldsymbol{\epsilon}$ is a diagonal matrix \mathbf{D} , instead of $\sigma^2\mathbf{I}$. If we multiply through this equation on the left by the positive definite square root of the inverse of \mathbf{D} , we obtain

$$\mathbf{D}^{-1/2}\mathbf{y} = \mathbf{D}^{-1/2}\mathbf{X}\boldsymbol{\beta} + \mathbf{D}^{-1/2}\boldsymbol{\epsilon}$$

and setting $\mathbf{y}' = \mathbf{D}^{-1/2}\mathbf{y}$, $\mathbf{X}' = \mathbf{D}^{-1/2}\mathbf{X}$ and $\boldsymbol{\epsilon}' = \mathbf{D}^{-1/2}\boldsymbol{\epsilon}$, we obtain

$$\mathbf{y}' = \mathbf{X}'\boldsymbol{\beta} + \boldsymbol{\epsilon}'.$$

It is easy to see that the variance-covariance matrix of $\boldsymbol{\epsilon}'$ is \mathbf{I} , so we have recovered a linear model with constant variance through this procedure. Applying ordinary least-squares to this transformed model is equivalent to minimizing

$$(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top \mathbf{D}^{-1} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

which is weighted least-squares with a weight of \mathbf{D}_{ii}^{-1} corresponding to the i th observation. Notice that when the variance of an observation is large, the weight is correspondingly lessened.

²See Appendix F as well.

In general, choosing weights is not straightforward. However, there are various choices that can be tried. In the case of count data, where a Poisson model is a possibility, a reasonable guess is for the error variance to be proportional to the mean of the counts. In the cigarette butt model, we saw that the mean count is roughly proportional to $e^{-0.00186x}$ where x is distance from the gazebo. Thus, we could try weights which are proportional to the inverse of this as in

```
 cig.wnls <- nls(count ~ exp(b0 + b1*distance), data = cigbutts,
   start = c(b0 = 3.1, b1 = -.001) , weights = 1/exp(-.00186*distance))
 coef(cig.wnls)

##          b0          b1
##  3.54406 -0.00168
```

The resulting estimate is plotted as a solid curve in Figure 6.6, contrasted with the unweighted least-squares estimate which is plotted as a dashed curve. This choice of weights has allowed the curve to be pulled away from the points at small distances (high counts).

```
 par(mar=c(4, 4, 1, .1))
 plot(count ~ distance, data = cigbutts, ylim = c(0, 30))
 a0 <- coef(cig.wnls)[1]
 a1 <- coef(cig.wnls)[2]
 curve(exp(a0+a1*x)-1, from = 0, to = 3000, add = TRUE)
 b0 <- coef(cig.nls)[1]
 b1 <- coef(cig.nls)[2]
 curve(exp(b0+b1*x), from = 0, to = 3000, add = TRUE, lty=2)
```

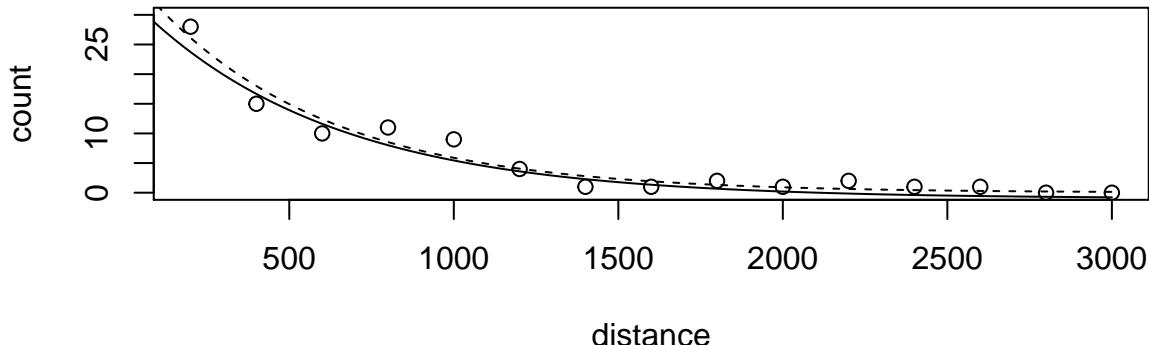


Figure 6.6: Comparison of weighted nonlinear and nonlinear model fits to the cigarette butts data set. The solid curve is from the weighted least-squares fit and the dashed curve is from the ordinary least-squares fit.

6.4.3 Estimating coefficient standard errors

We can use a parametric bootstrap to calculate coefficient standard error estimates, assuming a Poisson model as follows:

```
cigbuttsSim <- cigbutts
n <- nrow(cigbutts)
R <- 9999
coef.sim <- matrix(0, nrow=R, ncol=2)
```

```

for (j in 1:R) {
  cigbuttsSim$count <- rpois(n, predict(cig.nls))
  cigSim.wnls <- nls(count ~ exp(b0 + b1*distance), data = cigbuttsSim,
    start = c(b0 = 3.1, b1 = -.001), weights = 1/exp(-.0017*distance))
  coef.sim[j, ] <- coef(cigSim.wnls)
}
SE <- apply(coef.sim, 2, sd)
SE

## [1] 0.182132 0.000237

```

Thus, the standard error estimates obtained from the parametric (Poisson) bootstrap are 0.182 and 2.369×10^{-4} . On the other hand, the standard error estimates obtained from the nonlinear least-squares output are 0.152 and 1.618×10^{-4} as seen from

```

summary(cig.wnls)$coefficients

##      Estimate Std. Error t value Pr(>|t|)
## b0    3.54406   0.151756   23.4 5.33e-12
## b1   -0.00168   0.000162  -10.4 1.14e-07

```

The bootstrap estimate of the standard error of β_1 is considerably larger than the one estimated from the weighted nonlinear least-squares fit. This suggests that the Poisson model might not be very accurate.

We can employ a nonparametric bootstrap by the case-resampling procedure using the `boot()` function in the `boot` package.³ In the example below, entire cases or rows are sampled with replacement from the data frame, resulting in a new data frame whereby the bootstrap nonlinear least-squares estimates are obtained. Again, this resampling procedure is carried out a large number of times (here, 10000 times).

```

nlsfun <- function(data, indices) {
  simdata <- data[indices,]
  fit.wnls <- nls(count ~ exp(b0 + b1*distance), data = simdata,
    start = c(b0 = 3.6, b1 = -.0018), weights = 1/exp(-.0017*distance),
    control = nls.control(tol = 1e-3))
  coef(fit.wnls)
}
cig.boot <- boot(cigbutts, statistic = nlsfun, R = 10000)
cig.boot

##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = cigbutts, statistic = nlsfun, R = 10000)
##
##
## Bootstrap Statistics :
##      original     bias    std. error
## t1*    3.5534 -1.68e-02    0.178106
## t2*   -0.0017  1.68e-06    0.000188

```

³For more details about resampling and bootstrapping, see the text by Davison and Hinkley (1997).

As can be seen, the nonparametric bootstrap estimates of the coefficient standard errors are 0.178 and 1.884×10^{-4} . These values are in better agreement with the values coming from the nonlinear least-squares calculations. Note the use of the `control()` function within the `nls()` function here: the default tolerance (`tol = 1e-5`) was too stringent for rapid enough convergence of the nonlinear least-squares iteration for all resamples. By taking a larger value of `tol`, we achieve convergence for all resamples, with a slight cost in terms of accuracy. We emphasize that this cost is indeed slight, since the coefficient estimates themselves are subject to relatively large errors.

6.4.4 Modelling binary responses

The data in `p13.1` in the *MPV* package describes successes and failures of surface-to-air missiles as they relate to target speed. The data are plotted in Figure 6.7, with successes on the vertical axis being represented by a ‘1’ and failures being represented by a ‘0’.

Such binary data are obviously not normally distributed and are more appropriately modelled by a binomial distribution or relative. Because the variance of such a distribution depends on the success probability p , the constant variance assumption is violated, so the efficacy of ordinary least-squares becomes very questionable here. In this section, we indicate what could and should not be done with least-squares for such data.

```
par(mar=c(4, 4, .1, .1))
plot(p13.1, xlab = "target speed", ylab = "success/failure")
```

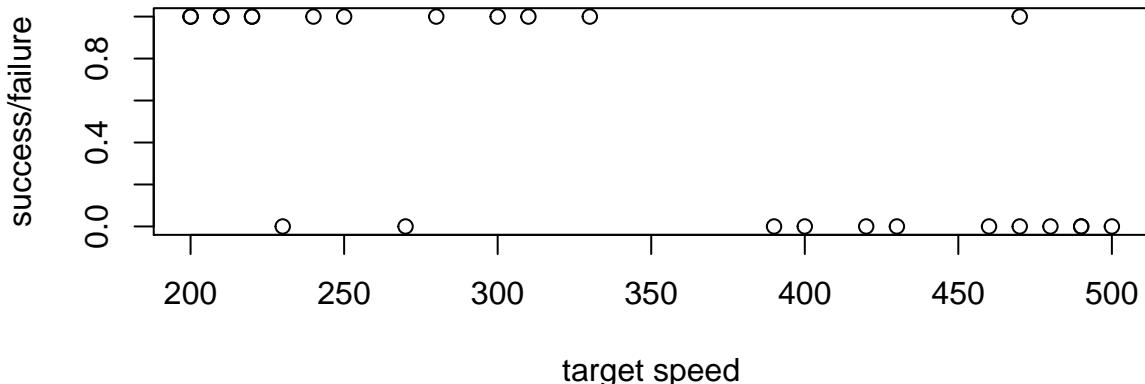


Figure 6.7: Surface-to-air missile successes (1) and failures (0) as they relate to target speed (in knots).

The first observation to make is that fitting a straight line to such data makes no sense, since the plotted points do not at all scatter about such a line. Furthermore, if such a line were to be fit to the data, it would necessarily take values outside the interval $[0, 1]$ on subsets of the domain; interpretation of such values would be difficult. In fact, the preferred interpretation of output arising from the fitting of models to such data is that of probability. That is, useful models can provide answers to questions such as, “What is the probability of success at a given target speed?” Since probabilities must lie within the interval $[0, 1]$, we must consider models based on nonlinear functions.

There are many functions which have values in $[0, 1]$. For example, the absolute value of the sine function is a candidate. Such a function might be appropriate if there were oscillatory or periodic behaviour to be modelled, but often, the desired model behaviour is monotonic (either increasing or decreasing). For the current example, we might reasonably believe that the probability of success decreases as target speed increases.

A large class of increasing functions that have values in $[0, 1]$ is the class of probability distribution functions. Some of the many possibilities are plotted in Figure 6.8.

```
par(mar=c(4, 4, .1, .1))
curve(pnorm(x), from = -3.5, to = 3.5, ylab = "F(x)") # normal distribution function
curve(pexp(x), from = -3.5, to = 3.5, add = TRUE, lty = 2) # exponential
curve(punif(x), from = -3.5, to = 3.5, add = TRUE, lty = 3) # uniform
curve(pt(x, df=2), from = -3.5, to = 3.5, add = TRUE, lty = 4) # t(2)
```

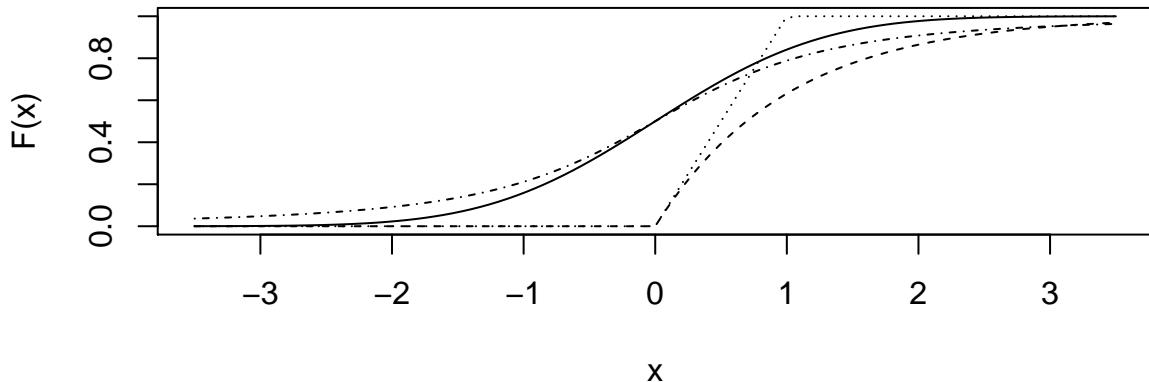


Figure 6.8: Some examples of cumulative probability distribution functions; increasing with values in $[0, 1]$. The solid curve is normal; the dashed curve is exponential; the dotted curve is uniform; the dashed-dotted curve is t on 2 degrees of freedom.

Fitting probit models with nonlinear least-squares

One of the functions pictured in Figure 6.8 is the normal cumulative distribution function. When this function is used to model the probabilities of binary outcomes, the term probit model is often used. In Figure 6.9, a number of independent observations have been simulated from such a model:

$$P('1') = P(Z < x)$$

where Z is a standard normal random variable (i.e. $\mu = 0$ and $\sigma = 1$). The `rbinom()` function is used to simulate these Bernoulli outcomes.

Since a general normal random variable might have a nonzero mean and a standard deviation that is not 1, we might consider estimating μ and σ from the simulated data. However, we are now choosing estimates of μ and σ so that the points plotted in Figure 6.9 are as close as possible to the resulting distribution function curve. This contrasts with the traditional approach to estimating the mean and variance by calculating the corresponding sample quantities. In fact, a way to approach the problem is, again, nonlinear least-squares, and our code below illustrates how we can do this with the `nls()` function, together with the `pnorm()` function.

```
n <- 25; x <- runif(n, min = -3, max = 3)
p <- pnorm(x); y <- rbinom(n, 1, p)
par(mar=c(4, 4, .1, .1)); plot(x, y)
y.nls <- nls(y ~ pnorm(x, mean=mu, sd = sigma), start = c(mu = 0, sigma = 1))
betahat <- coef(y.nls)
betahat

##      mu      sigma
## 0.0437 1.3247

curve(pnorm(x), add = TRUE)
curve(pnorm(x, mean = betahat[1], sd = betahat[2]), add = TRUE,
      lty = 2, col = "gray", lwd = 2)
```

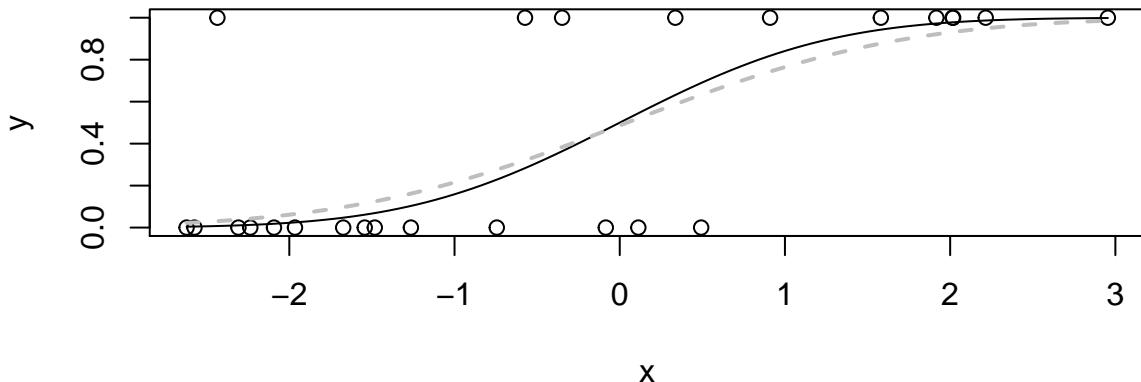


Figure 6.9: Simulated data from a probit model, together with fitted model. The truth is plotted as a solid black curve, and the estimate is overlaid as a dashed gray curve.

Decreasing functions

Returning to the surface-to-air missile data, we recall that the function we seek should be decreasing with x , not increasing. This can be handled by using the survivor function which is the distribution function subtracted from 1. Figure 6.10 demonstrates how this might work, again with simulated data. We now give the details of our simulation procedure as well as the steps taken to analyze the simulated data.

We first simulate the data. Execution of the following code gives random binary responses y at a set of 25 x -locations randomly distributed between -3 and 3:

```
n <- 25
x <- runif(n, min = - 3, max = 3)
p <- 1 - pnorm(x, mean = 0.1, sd = 1.2)
y <- rbinom(n, 1, p)
```

We have used a probit function with $\mu = 0.1$ and $\sigma = 1.2$ to generate the responses. In other words, the responses are binomially distributed with success probability $p(x)$ which is a cumulative normal distribution with mean 0.1 and variance 1.44.

We next use ordinary nonlinear least-squares to estimate μ and σ , as if we did not know the true values. The following code shows how to do this. Note that we need starting guesses for μ and σ .

```
y.nls <- nls(y ~ 1 - pnorm(x, mean=mu, sd = sigma),
               start = c(mu = 0, sigma = 1))
betahat <- coef(y.nls)
betahat

##      mu      sigma
## -0.2800  0.0688
```

The output gives us estimates of μ and σ which, unsurprisingly, differ somewhat from the true values. We can see the effect of the estimation error on the success probability estimates by comparing the estimated cumulative normal distribution curve with the true normal distribution curve. The code to produce the curves is as follows.

```
par(mar=c(4, 4, .1, .1))
plot(x, y)
curve(1 - pnorm(x), add = TRUE)
curve(1 - pnorm(x, mean = beta[1], sd = beta[2]), add = TRUE,
      lty = 2, col = "gray", lwd = 2)
```

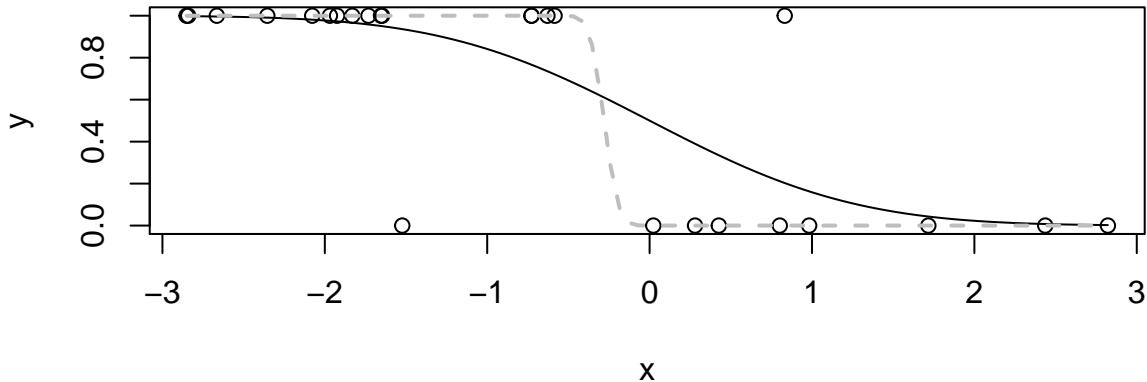


Figure 6.10: Simulated data from a probit model for a decreasing probability of success, together with fitted model. The truth is plotted as a solid black curve, and the estimate is overlaid as a dashed gray curve.

The need for weighting

As in the count case, it is unlikely that the variance of the response is constant, since the variance of a Bernoulli random variable with parameter p is $p(1 - p)$. Thus, an improvement to the nonlinear least-squares approaches outlined above would involve incorporating weights based on the inverse of $p(1 - p)$.

The following code illustrates how one might do this. In order to come up with the mean and standard deviation for the probit weights, trial and error was employed until a rough match was obtained between the estimates of μ and σ and the values used in the probit weights. The dashed curve in Figure 6.11 shows the resulting weighted least-squares estimate. It is not very different from the unweighted estimate, since both estimates are unbiased. The difference comes in looking at the coefficient standard error estimates; the unweighted least-squares standard error estimates are not valid, while the ones based on the weighted least-squares will be reasonable approximations to the truth.

```
par(mar=c(4, 4, .1, .1)); plot(x, y)
px <- 1 - pnorm(x, mean = -.18, sd = 1.34)
estweights <- 1 / (px * (1 - px))
y.nls <- nls(y ~ 1 - pnorm(x, mean=mu, sd = sigma),
               start = c(mu = 0, sigma = 1), weights = estweights)
betahat <- coef(y.nls)
betahat

##      mu    sigma
## -0.112 1.004

curve(1 - pnorm(x), add = TRUE)
curve(1 - pnorm(x, mean = betahat[1], sd = betahat[2]), add = TRUE,
      lty = 2, col = "gray", lwd = 2)
```

A probit model for the missile success data

Returning to the missile success/failure data, we note that the approximate center of the data is around $x = 350$, so this gives us an initial guess for μ (which turns out to be a pretty good guess ultimately). Guessing σ is somewhat less obvious, but calculating the sample standard deviation of the x values can work. In this case, we obtain a value near 112, so we round off to 100 as an initial guess for σ .

We use weighted least-squares, and again employ trial-and-error until we see a rough match between the values of μ and σ used in the weights and the final estimates.

The fitted probit curve is plotted in Figure 6.12. Note that the estimated probability of a success at a target speed of 200 is very high, but not quite 1.0. This is due to the influence of the failure at a target speed near 230.

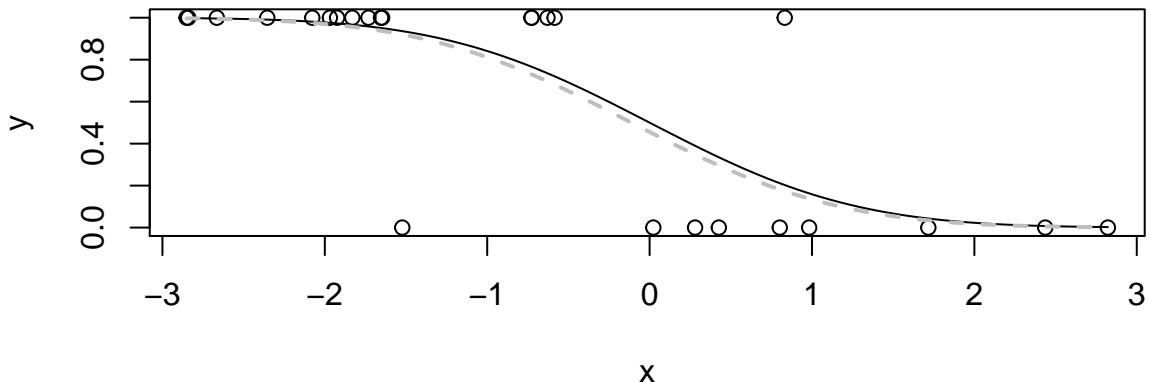


Figure 6.11: Simulated data from a probit model for a decreasing probability of success, together with fitted model based on weighted least-squares. The truth is plotted as a solid black curve, and the estimate is overlaid as a dashed gray curve.

Similarly, the success near 500 prevents the probability of success from hitting 0 at 500, though the estimated probability of success at that point is very low.

```
estweights <- 1 - pnorm(p13.1$x, mean = 350, sd = 99)
estweights <- 1/(estweights*(1-estweights))
y.nls <- nls(y ~ 1 - pnorm(x, mean=mu, sd = sigma),
               start = c(mu = 350, sigma = 100), weights = estweights, data = p13.1)
betahat <- coef(y.nls)
betahat

##      mu      sigma
## 350.5     99.4

par(mar=c(4, 4, .1, .1))
plot(p13.1, xlab = "target speed", ylab = "success/failure")
curve(1-pnorm(x, mean = betahat[1], sd = betahat[2]), add = TRUE)
```

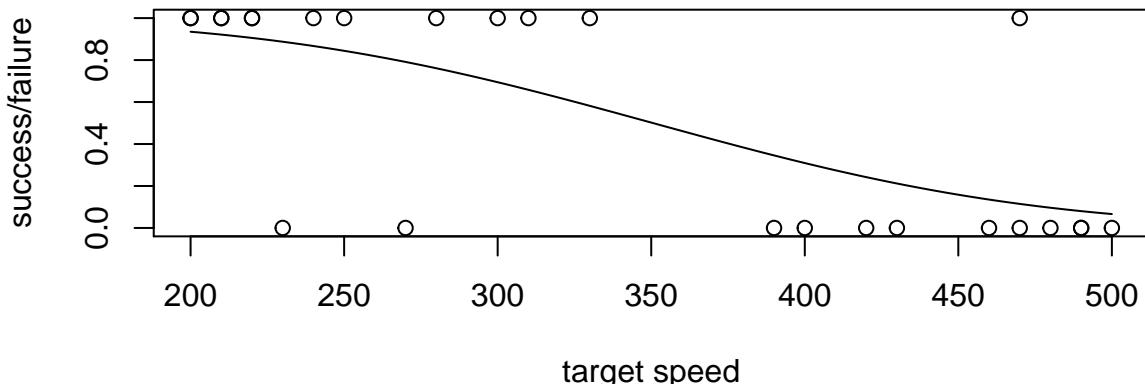


Figure 6.12: A probit model for the missile success data; the estimate is overlaid as a solid black curve.

The logistic and logit functions

Another function which has the behaviour pictured in Figure 6.8 is the logistic function. This function arises in a number of ways. Here we will obtain it as an inverse function to the logit function which we now develop.

Recall that probabilities or proportions p take values only in the interval $[0, 1]$, and this property renders linear models unsatisfactory. However, there is a way to transform p so that linear modelling again becomes an option. Thus, we seek a transformation of p which takes values on the entire real line.

First, note that if $p \in (0, 1)$, then the odds $p/(1 - p) \in (0, \infty)$. If we take the log odds, we obtain the logit of p :

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right).$$

It is easily shown that $\text{logit}(p)$ can take any value on $(-\infty, \infty)$. Therefore, straight line models are again a possibility under such a transformation. For example,

$$\text{logit}(p) = \beta_0 + \beta_1 x$$

where x is some covariate (such as target speed in the missile example).

Recalling the definition of an inverse function, and employing some algebra, we can invert the logit function to obtain the logistic function:

$$p(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}.$$

This is an increasing function of x and takes values in $(0, 1)$, so it is a valid probability distribution function.

The code below shows how to simulate the logistic model where $\beta_0 = 0$ and $\beta_1 = 1$ at a number of uniformly distributed points. Again, `rbinom()` is used to simulate the Bernoulli distributed binary outcomes.

```
n <- 25; x <- runif(n, min = - 3, max = 3)
p <- exp(x) / (1+exp(x))
y <- rbinom(n, 1, p)
par(mar=c(4, 4, .1, .1)); plot(x, y)
y.nls <- nls(y ~ exp(b0 + x*b1)/(1+ exp(b0 + x*b1)), start = c(b0 = 0, b1 = 1))
betahat <- coef(y.nls)
betahat

##      b0      b1
## -4.17 30.72

curve(exp(x) / (1+ exp(x)), add = TRUE)
curve(exp(betahat[1] + x*betahat[2]) / (1+ exp(betahat[1] + x*betahat[2])), 
      add = TRUE, lty = 2, col = "gray", lwd = 2)
```

The code also produces a plot as in Figure 6.13 which displays the successes and failures as open circles. The logistic function model is fit using nonlinear least-squares, and the resulting coefficient estimates are -4.172 and 30.722. The fitted logistic function is plotted as a dashed gray curve overlaying the true function which is represented by the solid curve.

As in the probit case, the variance of the responses is unlikely to be constant in this situation. Thus, a weighted least-squares approach is needed in order to improve upon the estimates and in order for the standard errors of the coefficients to be valid. The approach is very similar to that taken with the probit case and is left as an exercise.

Figure 6.14 shows the logistic model curve as obtained by unweighted nonlinear least-squares estimation. The code for fitting the model is below. Adding appropriate weights is an exercise for the reader.

```
y.nls <- nls(y ~ exp(b0 + b1*x) / (1+ exp(b0 + b1*x)),
start = c(b0 = 0, b1 = 0), data = p13.1)
betahat <- coef(y.nls); betahat
```

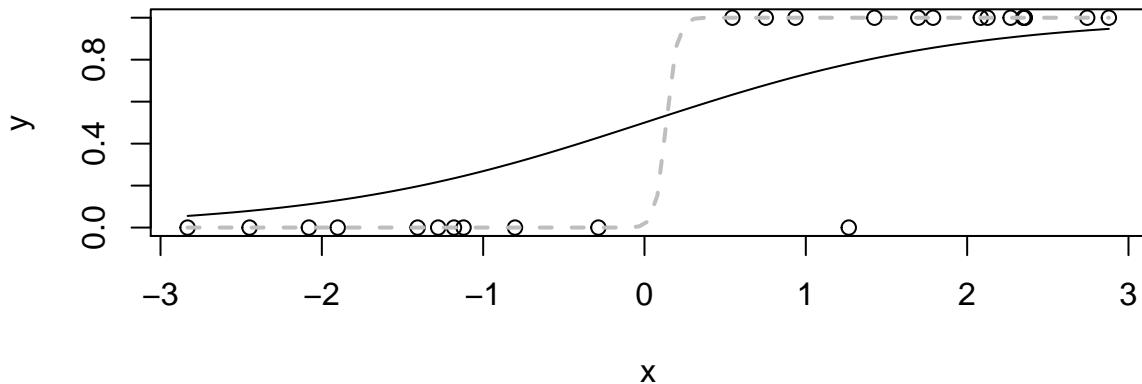


Figure 6.13: Simulated data from a logistic model, together with fitted model.

```

##      b0      b1
## 7.2519 -0.0209

par(mar=c(4, 4, 1, .1))
plot(p13.1, xlab = "target speed", ylab = "success/failure")
curve(exp(betahat[1] + x*betahat[2]) / (1+ exp(betahat[1] + x*betahat[2])), add = TRUE)

```

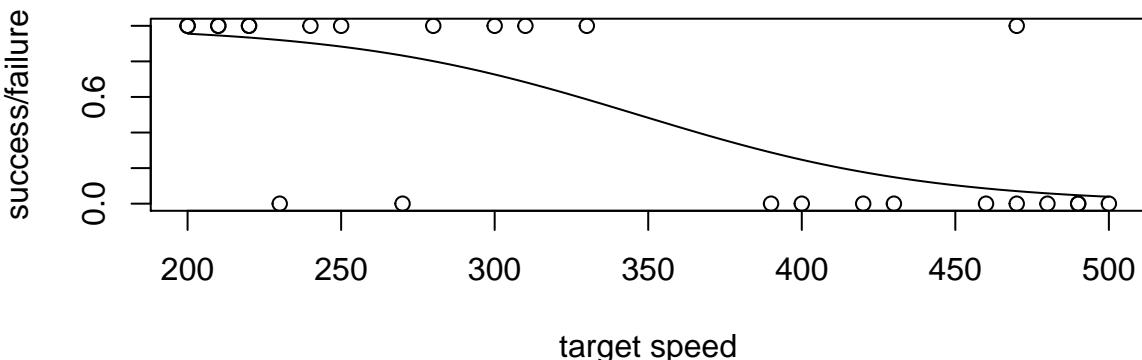


Figure 6.14: A logistic model for the missile success data; the estimate is overlaid as a solid black curve.

6.5 Exercises

- Consider the following data:

Catalyst	Temperature	Yield
present	10	15
present	-10	10
absent	10	5
absent	-10	5

- Fit a linear model (*without an intercept*) to this data, relating Yield to the other two variables. Write down the fitted model, defining all terms used.

- (b) Estimate the error variance.
- (c) Give a point estimate for the mean yield when Catalyst is present and Temperature is 5.
- (d) What is the estimated standard error for the above mean estimate?
2. Consider the data on height, weight and sex considered in this chapter.
- (a) Fit an analysis of covariance model to these data, relating weight to height and sex, including the interaction term between height and sex.
- (b) Is an interaction between sex and height needed?
- (c) Write out the design matrix for the model with interaction.
3. Consider the following data:
- | Catalyst | Pressure | Yield |
|----------|----------|-------|
| present | 20 | 15 |
| present | 10 | 10 |
| present | 5 | 5 |
| absent | 10 | 5 |
| absent | 0 | 5 |
- (a) Fit a linear model (*without an intercept*) to this data, relating Yield to the other two variables. Write down the fitted model, defining all terms used.
- (b) Write down equations for the two fitted lines, corresponding to whether Catalyst is present or absent. Are the estimated intercepts the same? Are the estimated slopes the same?
- (c) Estimate the error variance.
- (d) Calculate a 95% confidence interval for the coefficient of Pressure.
- (e) Is there evidence of a difference in mean yield depending on whether the catalyst is present or absent?
- (f) Calculate a 95% prediction interval for the yield when Catalyst is present and pressure is 8.
4. Consider the following data set:
- | i | j | x | y | gender | (g: M=0, F=1) |
|---|---|---|----|--------|---------------|
| 1 | 1 | 1 | 2 | M | |
| 1 | 2 | 2 | 4 | M | |
| 2 | 1 | 3 | 8 | F | |
| 2 | 2 | 4 | 15 | F | |
- (a) Find the model matrix for the following regression models:
- $y_{ij} = \beta_1 x_{ij} + \varepsilon_{ij}$
 - $y_{ij} = \beta_0 + \varepsilon_{ij}$
 - $y_{ij} = \beta_0 + \beta_1 g_{ij} + \varepsilon_{ij}$
 - $y_{ij} = \beta_0 + \beta_1 g_{ij} + \beta_2 x_{ij} + \varepsilon_{ij}$
- (b) For each of the above models, how many degrees of freedom would be available for estimating the error variance, if they were fit to the given data?
5. Fit the model with two parallel lines to the tooth growth data. What are the intercepts for the VC and OJ lines? What is the slope? Plot the data with the two lines overlaid.
6. Consider the data in airquality which relate to Ozone levels in New York. Construct a model which relates Ozone level to temperature and wind, taking Month into account, as a factor. Is there evidence that Month should be included in the model? Is there an interaction between Month and wind? between Month and temperature?

7. Recall the hypothesis test discussed in Section 6.1.1. The test statistic is a t-statistic. Can you set up a test for the same hypotheses using an F-statistic?
8. Consider the data in the `gasdata` data frame in the *MPV* package. The `dailyusage` column gives the average daily volume of natural gas used in a furnace each month from 2006 until 2011. Average daily temperature for each month is contained in the `temp` column. The columns `I1`, `I2`, `I3` are factors indicating the presence of added insulation as well as an upgrade to the furnace.
 - (a) Fit a simple regression model relating average daily gas usage to temperature. Obtain a scatter plot of the data together with an overlaid fitted line.
 - (b) Now add a term involving `I1` to the model. Test whether an interaction between temperature and `I1` is needed. Again, plot the data, together with the resulting two lines overlaid.
 - (c) Now add a term involving `I2` to the model. Test whether an interaction between temperature and `I2` is needed. Again, plot the data, together with the resulting three lines overlaid.
 - (d) Finally, add a term involving `I3` to the model. Test whether an interaction between temperature and `I3` is needed. Again, plot the data, together with the resulting four lines overlaid.
 - (e) Based on your analysis, would you say that the insulation and furnace upgrade are leading to decreased natural gas requirements?
9. Consider the R data frame `xy` given below.


```
> xy
  x1  x2   y
  0    0   0
  1    1   2
 -1   -1  -2
  1   -1   1
 -1    1   0
```

 - (a) Fit the model

$$y = \beta_1 x_1 + \beta_2 x_2 + \varepsilon$$
 where the usual assumptions are made about the error ε .
 - (b) Calculate a 95% confidence interval for β_1 and interpret.
 - (c) Compute variance inflation factors for $\hat{\beta}_1$ and $\hat{\beta}_2$. Is there evidence of multicollinearity?
10. The `cfseal` data frame is included in the *DAAG* package. It includes observations on several variables on cape fur seals, including age.
 - (a) Develop a predictive model for age, given the other variables.
 - (b) Check for multicollinearity, using variance inflation factors.
11. Consider the simulated logistic regression data of Section 6.4.4. Adjust the code for the nonlinear least-squares estimates of the model parameters so that appropriate weights are included. Apply the same technique to obtain an improved logistic model for the missile success data.
12. Consider the model of Exercise 7i in Chapter 2.
 - (a) Use the fact that the variance of the i th response is $\sigma^2(1 + \sigma_B^2 x_i^2 / \sigma^2)$ to show that weighted least-squares can be carried out using weights $w_i = \sqrt{1 + (\sigma_B / \sigma)^2 x_i^2}^{-1}$, and show that the resulting estimator for β_1 is $\tilde{\beta}_1 = \sum_{i=1}^n w_i^2 x_i y_i / \sum_{i=1}^n w_i^2 x_i^2$.
 - (b) Show that the variance of $\tilde{\beta}_1$ is $1 / \sum_{i=1}^n x_i^2 / (\sigma^2 + \sigma_B^2 x_i^2)$.

- (c) Using the parameter values for the simulation study of the Chapter 2, compare the true standard errors for $\hat{\beta}_1$ and $\hat{\beta}_1$. Which estimator is better?
- (d) Identify any practical difficulties which would be associated with using weighted least-squares in this context, and suggest strategies that could be used to overcome these difficulties.

13. The Michaelis-Menten model for chemical kinetics is a nonlinear regression model of the form

$$y = \frac{\alpha x}{x + \beta} + \varepsilon$$

where α and β are constants, and ε is noise which has mean 0 and variance σ^2 . In this exercise, we will fit this model to the untreated data in `Puromycin` which can be obtained as

```
puntreated <- subset(Puromycin, state=="untreated")
```

and where y is taken as the initial rate of the reaction (`rate`), and x is substrate concentration (`conc`).

- (a) In order to fit this nonlinear regression model, we need starting guesses for $\hat{\alpha}$ and $\hat{\beta}$. For this model, use the fact that
- $$\frac{1}{y} \doteq \frac{1}{\alpha} + \frac{\beta}{\alpha x}$$
- and regress $1/rate$ against $1/conc$ to get estimates of $1/\alpha$ and β/α .
- (b) Use the estimates obtained at the previous step as starting guesses in a call to `nls`.
- (c) Check the summary output for the fitted model to obtain standard error estimates for the coefficient estimates.
- (d) Plot the data and overlay the fitted Michaelis-Menten curve.

14. Repeat the steps of the previous question for the treated data in `Puromycin`.

15. Consider the tree height and age data in `tree.sample`.

- (a) Regress height against age, and check the diagnostic plots. Comment on the results.
- (b) Add a quadratic term in age to the model. Does this provide an improvement to the model? What issues remain?
- (c) The quadratic model you have just fit is of the form

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

where y is height, x is age, and ε is the “error” which appears to have a variance that increases with age. Suppose you could also obtain information on soil type, moisture conditions and drainage, amount of sunlight, growing degree days, and genetic characteristics of each tree. Which parts of the above model would need to be modified in order to incorporate all of this information? In particular, what would happen to ε ?

- (d) In fact, `tree.sample` is a stratified random sample of observations taken from the `Loblolly` data frame which consists of 6 measurements of age and height taken on 12 different trees. Run the following code to see the structure in the data set:

```
library(lattice)
xyplot(height ~ age, groups=Seed, data = Loblolly,
       type=c("p", "smooth"))
```

On the basis of this plot, offer an explanation for the changing variance that you (should have) observed in part (b)?

16. Fit a weighted quadratic regression model to the data in `tree.sample` using weights proportional to $1/\text{age}$. How do the coefficient estimates and standard error estimates compare with the unweighted result? Which ones do you think are more accurate?
17. Consider the regression model

$$y = \mathbf{X}\beta + \varepsilon$$

where ε is a normal random vector with mean 0 and variance-covariance matrix $\sigma^2\Sigma$. Set $y' = \Sigma^{-.5}y$, $X' = \Sigma^{-.5}X$ and $\varepsilon' = \Sigma^{-.5}\varepsilon$, and develop the generalized least-squares estimator for β by applying the QR decomposition approach to the model

$$y' = \mathbf{X}'\beta + \varepsilon'$$

Specifically,

- (a) provide the formula to calculate $\hat{\beta}$.
 - (b) provide the formula for the variance of $\hat{\beta}$, and identify the standard errors of the components of $\hat{\beta}$.
 - (c) provide a formula for the estimate of σ^2 .
 - (d) provide a formula for the F statistic which could be used to test the significance of the regression.
18. (a) Read the help file on the `lesions` data on $n = 66$ rat colons in the *MPV* package.
 (b) Obtain a scatter plot relating `ACF.Total` (y) to `T` (time, x_1) for each level of `INJ` (number of carcinogen injections, x_2) for the subset of the data where `SECT` is 5. Does a model that links $\log(E[y])$ to time and number of injections make sense? Suppose, for the i th rat,

$$\log(E[y_i]) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i}.$$

Use the following code to explore the data graphically to see if this model is suitable?

```
library(MPV)
lesions$T <- as.numeric(as.character(lesions$T)) # T was a factor
xypot(ACF.Total ~ T|INJ, data = subset(lesions, SECT==5))
```

In the following sequence of steps, we will fit this model to the fifth section data.

- (c) The model suggested in the previous part is mathematically equivalent to

$$E[y_i] = e^{(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i})} = e^{x_i^\top \beta}$$

but the latter has the advantage that the y_i 's are unbiased estimators for $E[y_i]$. We can use this fact as the basis for setting up least-squares estimators for the regression parameters:

$$L(\beta) = \sum_{i=1}^n (y_i - e^{(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i})})^2 = \sum_{i=1}^n (y_i - e^{x_i^\top \beta})^2$$

$$\hat{\beta} = \operatorname{argmin}_{\beta} L(\beta).$$

This is a nonlinear least-squares problem. We will use the Gauss-Newton method to solve it.

- (d) Expand $e^{x_i^\top \beta}$ in Taylor series about an initial guess b_0 (a 3-vector) to obtain the approximation

$$e^{x_i^\top \beta} \doteq e^{x_i^\top b_0} - x_i^\top b_0 e^{x_i^\top b_0} + e^{x_i^\top b_0} x_i^\top \beta$$

and deduce a suitable approximation for

$$y_i - e^{x_i^\top \beta}.$$

Recall that Taylor gives us

$$g(\beta) \doteq g(b_0) + g'(b_0)(\beta - b_0).$$

(Here, $g'(b_0)$ is the gradient or the vector of partial derivatives of g with respect to the components of β .)

The suitable approximation is

$$y_i - e^{x_i^\top \beta} \doteq y_i - e^{x_i^\top b_0} + x_i^\top b_0 e^{x_i^\top b_0} - e^{x_i^\top b_0} x_i^\top \beta.$$

- (e) Identify 'pseudodata' y'_i from the previous step so that you can re-write the above approximation as

$$y_i - e^{x_i^\top \beta} = y'_i - e^{x_i^\top b_0} x_i^\top \beta.$$

This can now be used in an ordinary least-squares calculation which minimizes

$$\sum_{i=1}^n (y'_i - e^{x_i^\top b_0} x_i^\top \beta)^2$$

with respect to the three-dimensional vector β .

Write out the form of the i th row of the model matrix \mathbf{X} for this problem.

- (f) Write out the R code required to convert the section 5 `lesions` data to the model matrix described in the previous part. Assume a starting vector $b_0 = [0.5 \ 1.0 \ 0.5]$.
- (g) Obtain the pseudodata y' .
- (h) Apply the QR decomposition to \mathbf{X} , and use this to estimate β .
- (i) Replace the initial guess `b0` with the estimate `beta` and re-calculate the pseudodata, model matrix and estimation. Repeat until the results stabilize.
19. (a) Repeat the preceding analysis of the `SECT=5` subset of the `lesions` data, using the `nls` function using the same starting value for β .
- (b) If we believe the counts follow a Poisson distribution, then the variance cannot be constant. Therefore, a weighted least-squares approach is needed, where the weights are the reciprocal of the variance, and where the variance is equal to the mean. To estimate the mean, you need to plug in the data and use your most recent coefficient estimates. Incorporate this change into your nonlinear least-squares estimate.
- (c) Repeat the procedure several times, updating the weights each time using the most recent values of the coefficient estimates. Stop when the coefficient estimates stop changing. (A while loop would be appropriate here, for example.)
- (d) Compare the result above with what you would have obtained using `glm` and the `poisson` family. That is, try code such as the following.

```
15.poisson <- glm(ACF.Total ~ T + INJ,
                    data = sect5data, family=poisson)
15.poisson
```

- (e) Using code as shown below, simulate new responses from the fitted Poisson model, and look at the summary output after re-fitting. Identify ways in which the output from the simulated data differ from the output from the original data.

```
ACF.sim <- simulate(15.poisson)$sim_1
ACFsim.poisson <- glm(ACF.sim ~ T + INJ,
                      data = sect5data, family=poisson)
summary(ACFsim.poisson)
```

Is there any reason why you might not believe that the Poisson model is adequate for the actual data?

20. Analyze the `SECT == 6` subset of the `lesions` data using the procedure outlined in the preceding question.

21. What strategies should be followed, if nonlinear least-squares (nls) does not converge? Should the number of iterations be increased? Or should we use a different starting guess?
22. Consider the `gasdata` models that you constructed in Exercise 16 of Chapter 5. Compute variance inflation factors for the reduced model. Is multicollinearity a problem with this model?

7

Theory for Likelihood

7.1 Likelihood properties

7.1.1 Invariance

Supppose y_1, y_2, \dots, y_n are observations from $f(Y; \theta)$ where θ is a parameter which governs the probability model f . The maximum likelihood estimator (MLE) $\hat{\theta}$ maximizes $f(Y; \theta)$ with respect to θ , by definition.

Now, suppose that $\gamma = g(\theta)$ for some known invertible function $g(\cdot)$. The quantity γ is a parameter and it is often possible to express the model $f(Y)$ in terms of γ instead of θ . This is an example of reparametrizing a model.

Set

$$L(\gamma) = f(Y; \theta)$$

with $\gamma = g(\theta)$ for all θ in the original parameter space for the model. Also, define $\hat{\gamma} = g(\hat{\theta})$. Then $\hat{\gamma}$ is the maximum likelihood estimator for γ since, for any γ ,

$$L(\gamma) = f(Y; \theta) \leq f(Y; \hat{\theta}) = L(\hat{\gamma}).$$

That is $\hat{\gamma}$ is a maximizer of the likelihood when the model is parametrized in terms of γ .

This property that says that a function of the MLE of θ is the maximum likelihood estimator of the function of θ is called the invariance property of maximum likelihood estimation. Even though it is a very simple property, it is very important, since it prevents ambiguities from occurring which might otherwise arise when changing scales or when taking logs or square roots of parameters. This kind of invariance property does not, in general, hold for other kinds of estimators.

Example – normal distribution: standard deviation versus variance

A notable example is the case of the standard deviation σ and the variance σ^2 , where the MLE for the standard deviation can be obtained by simply taking the square root of the MLE of the variance.

Suppose y is normally distributed with mean 0 and standard deviation σ . Under this parametrization, the probability density function is

$$f(y) = \frac{e^{-\frac{1}{2}y^2/\sigma^2}}{\sqrt{2\pi}\sigma}.$$

The likelihood function $L(\sigma) = f(y)$ is maximized at

$$\hat{\sigma} = |y|.$$

On the other hand, if we estimate the variance $\gamma = \sigma^2$, the probability density function becomes

$$f(y) = \frac{e^{-\frac{1}{2}y^2/\gamma}}{\sqrt{2\pi\gamma}}$$

where $\gamma > 0$ is assumed. Maximizing the likelihood as a function of γ , we obtain

$$\hat{\gamma} = y^2.$$

Note that the (positive) square root of this estimate is the standard deviation estimate in accord with the invariance principle.

Example – Poisson distribution: log rate versus rate

The rate λ for a Poisson distribution is required to be a positive real number, so it is sometimes more convenient to work with the logarithm of the rate, since that can assume any real value. If y_1, y_2, \dots, y_n are a collection of independent counts from a Poisson, λ distribution, the MLE for λ is $\hat{\lambda} = \bar{y}$ where \bar{y} is the average of the counts. By the invariance principle the MLE for $\log(\lambda)$ is $\log(\bar{y})$.

7.1.2 The derivative of the log likelihood

The expected value and variance of the derivatives of the log likelihood are often useful, particularly, in establishing large sample properties of maximum likelihood estimators.

We will consider a single observation from a probability density function with unknown parameter θ : $f(y; \theta)$. The log likelihood is given by

$$\ell(\theta) = \log f(y; \theta).$$

The expected score function

If $\ell(\theta)$ is a differentiable function of θ , we can show that under fairly general conditions, the expected value of the first derivative, often called the score function, is 0:

$$E \left[\frac{\partial \ell(\theta)}{\partial \theta} \right] = 0. \quad (7.1)$$

This can be demonstrated by noting that the left hand side of this identity is defined as

$$\int \frac{\partial \ell(\theta)}{\partial \theta} f(y; \theta) dy$$

which, by the definition of the log likelihood, can be written as

$$\int \frac{1}{f(y; \theta)} \frac{\partial f(y; \theta)}{\partial \theta} f(y; \theta) dy$$

which simplifies to

$$\int \frac{\partial f}{\partial \theta} dy = \frac{\partial}{\partial \theta} \int f(y; \theta) dy = 0.$$

The interchange of differentiation and integration depends on conditions on f as a function of y and θ , which are often, but not always, satisfied in practice.

The importance of the result at (7.1) cannot be understated: we expect the likelihood to be have a critical point at the true value of θ . We will see shortly that the critical point is often a maximizer.

Information, variance, and derivatives

The variance of the score function contains information about the potential for precise estimation of the parameter θ . When the second derivative of the log likelihood exists, it has a useful relation to this variance, as it contains information about the curvature of the log likelihood in the vicinity of the true value of θ . If the likelihood is flat, there will be less precision than if the likelihood is more curved. In other words, a larger second derivative, in magnitude, would give more precision. The concept is illustrated graphically in Figure 7.1.

The second derivative of the log likelihood function in the normal case is $-1/\sigma^2$, so smaller values of σ^2 correspond to increased curvature around the maximum; when σ is 0.5, there is more curvature than when $\sigma = 1$, and at the same time, there is more certainty about the estimate when σ is smaller. In the Poisson case, the second derivative of the log likelihood is $-y/\lambda^2$ when there is a single observation y , and the second derivative is $-n\bar{y}/\lambda^2$ when there are n independent observations. The curvature is higher when $n > 1$, and this corresponds to greater precision in estimation about the true value of λ .

The next two results summarize this notion mathematically:

$$E \left(\frac{\partial^2 \ell(\theta)}{\partial \theta^2} \right) = -E \left[\left(\frac{\partial \ell(\theta)}{\partial \theta} \right)^2 \right] \quad (7.2)$$

$$\text{Var} \left(\frac{\partial \ell(\theta)}{\partial \theta} \right) = E \left[\left(\frac{\partial \ell(\theta)}{\partial \theta} \right)^2 \right] \quad (7.3)$$

The first result can be proved in a similar manner to the result at (7.1) using the chain and product rules of differential calculus. The second result follows immediately from the (7.1) and (7.2). The variance of the first derivative of the log likelihood, evaluated at the true value of the parameter, is referred to as the Fisher Information.

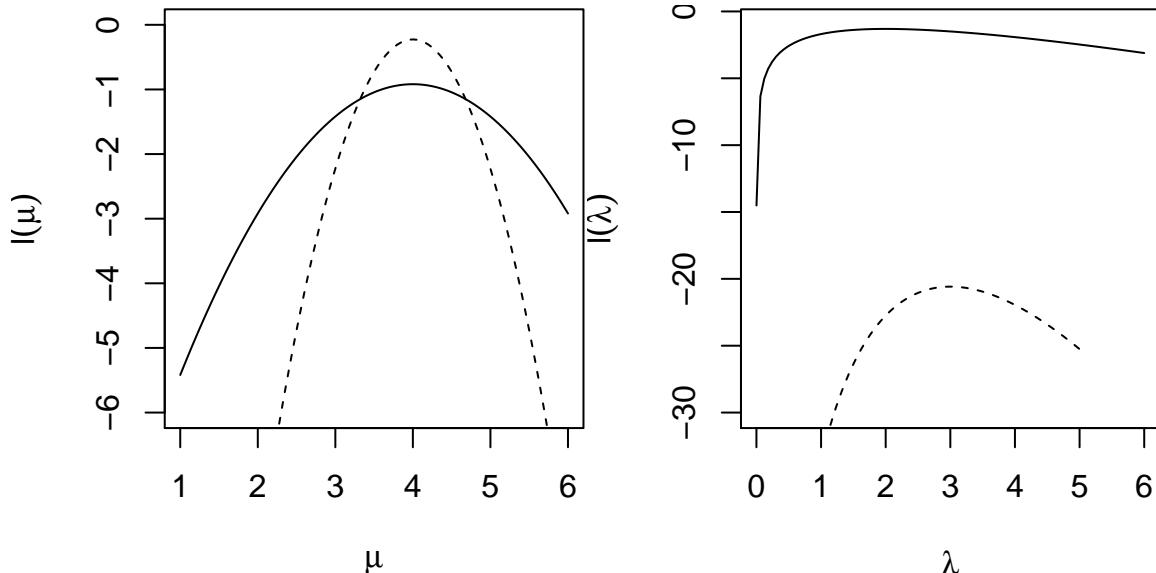


Figure 7.1: Curvature of the log likelihood function and precision of estimation. Left panel: log likelihood of the normal pdf at $y = 4$, for $\sigma^2 = 1$ (solid curve) and $\sigma^2 = 0.25$ (dashed curve). Right panel: log likelihood for the Poisson distribution with rate parameter λ with one observation at $y = 2$ (solid curve) and with 10 independent observations having come from a Poisson population with rate $\lambda = 2$.

The expected log likelihood is higher at the truth

Jensen's inequality, which can be found in the Appendix, can be used to demonstrate that

$$E[\ell(\theta)] \geq E[\ell(\theta')]$$

for all $\theta' \neq \theta$. Here, θ is assumed to be the true value of the parameter, and the expected value is taken with respect to the density function $f(x; \theta)$. The proof of this result is essentially the same as that given for Result 4 of Section C.4 where θ_0 is taken as the “true” value of the parameter.

7.1.3 Using ratios of likelihoods in statistical testing

Since the likelihood function is expected to take on larger values at parameter values near the true value, and smaller values elsewhere, it is possible to test whether a particular parameter value θ_0 is reasonable for a given data set. To do this, the likelihood is calculated at θ_0 as well as at the MLE. The likelihood at the MLE will be larger than the likelihood anywhere else, so the ratio of the two likelihood values should be a number that is at least one. How much larger this likelihood ratio statistic is than one gives a measure of evidence *against* the assumption that the true parameter value is θ_0 .

The likelihood ratio test of the null hypothesis that $\theta = \theta_0$ proceeds as follows:

1. Calculate the likelihood ratio: $R_L = L(\theta_{\text{MLE}})/L(\theta_0)$
2. Calculate the probability that such a likelihood ratio could exceed R_L for any other sample coming from a population for which the null hypothesis is true. This probability is the p-value for the test.

Since evidence against the null hypothesis increases with the value of the likelihood ratio, R_L , the p-value as defined above provides a corresponding measure of evidence. Small p-values are indicators of strong evidence against the null hypothesis.

Although the likelihood ratio is the basis for the test, it is often more convenient to work the log of the likelihood ratio. This becomes a difference of log likelihoods:

$$\log(L(\theta_{\text{MLE}})/L(\theta_0)) = \log(L(\theta_{\text{MLE}})) - \log(L(\theta_0)).$$

Example – impurity proportions

A possible model for proportions of impurity in samples is

$$f(x) = (\alpha + 1)x^\alpha, \quad x \in (0, 1), \quad \alpha > -1.$$

Interest centers on α . We know how to estimate it from one or more independent observations by maximizing the likelihood:

```
# mle:
alphahat <- function(x) {
  -1/mean(log(x)) - 1
}

# examples: x = .37; x = .9; x = .37, .9, .8
alphahat(.37)

## [1] 0.005780954

alphahat(.9)

## [1] 8.491222

alphahat(c(.37, .9, .8))

## [1] 1.267991
```

How do we test whether α differs from some possible baseline value α_0 . For example, $\alpha_0 = 0$ corresponds to the uniform distribution on $(0, 1)$, and we might want to quantify the evidence against the null hypothesis that α is 0. The likelihood ratio test offers a way to do this. The test statistic is constructed as

$$R_L = \frac{L(\alpha_{\text{mle}})}{L(\alpha_0)}.$$

For our problem, the likelihood ratio could be calculated as follows, assuming a single observation:

```
LR <- function(x, alpha) {
  # Here, alpha represents the null hypothesis parameter value.
  (-1/log(x)) * x^(-(1/log(x) + 1)) / ((alpha + 1) * x^alpha)
}

# examples: n = 1 only; null distribution is uniform
LR(.37, alpha = 0)

## [1] 1.000017

LR(.9, alpha = 0)

## [1] 3.879584
```

We are not yet in a position to say how much evidence we have against the null hypothesis in either of these cases, since we don't know the distribution of the test statistic. In this case, determining the distribution is somewhat challenging, but a parametric bootstrap can be employed here.

First, we can construct a random number generator for X values coming from the model pdf $f(x)$:

```
rX <- function(n, alpha) (runif(n))^^(1/(alpha + 1))

# examples:
rX(5, 0) # 2 uniform variates

## [1] 0.6247155 0.1264537 0.7330586 0.2403276 0.8458744

rX(3, 2) # 3 variates with alpha = 2

## [1] 0.8396079 0.9149839 0.6688381
```

Using such values, we can simulate random values of the likelihood ratio as follows:

```
rLR <- function(n, alpha) {
  x1 <- rX(n, alpha)
  LR(x1, alpha = 0)
}

# examples:
rLR(5, 0) # 5 likelihood ratio values (corresponding to n = 1); alpha = 0

## [1] 1.000050 9.093448 1.537902 1.014317 1.318422

rLR(3, 2) # 3 likelihood ratio values (corresponding to n = 1); alpha = 2

## [1] 15.689167 1.678737 1.892941
```

Simulating the distribution under the null hypothesis (uniform distribution) proceeds as follows:

```
y <- rLR(1000000, alpha = 0)
par(mar=c(4, 4, 1, .1))
plot(density(log(y)), xlim=c(-1, 6), main = "",
      xlab = expression(log(R[L]))) # this estimates the density from the
      # logs of the simulated data
rug(log(LR(.37, 0))), ticksize = .1 # this shows the evidence against the null
      # hypothesis if x = .37
rug(log(LR(.9, 0))), ticksize = .2 # this shows the evidence against the null
      # hypothesis if x = .9
abline(h = 0) # a horizontal base line
```

Figure 7.2 shows an estimate of the probability density function of the log of the likelihood ratios, for testing the hypothesis that $\alpha = 0$ based on a single observation. The p-values for the two tests correspond to the areas under the density curve to the right of the ticks for the rug plot. It is now clear that there is no evidence against the null hypothesis when $x = .37$ (shorter tick), but when $x = .9$, there is fairly strong evidence against the null hypothesis (that $\alpha = 0$).

7.1.4 Multiple observations and an asymptotic χ^2 distribution – a simulation example

If we have n independent observations, it can be shown that the log of the likelihood ratio considered above is related to a χ^2 distribution on 1 degree of freedom. (This corresponds to the number of parameters being tested, i.e. 1.) In fact, it can be shown mathematically that $2 \log(R_L)$ has a limiting distribution which is χ^2 on 1 degree of freedom as the sample size n increases, under the assumption that the null hypothesis is true, under fairly general conditions. (It is not always true.)

We will demonstrate this by simulation for Poisson data where the rate is homogeneous with $\lambda = \alpha = 1$.

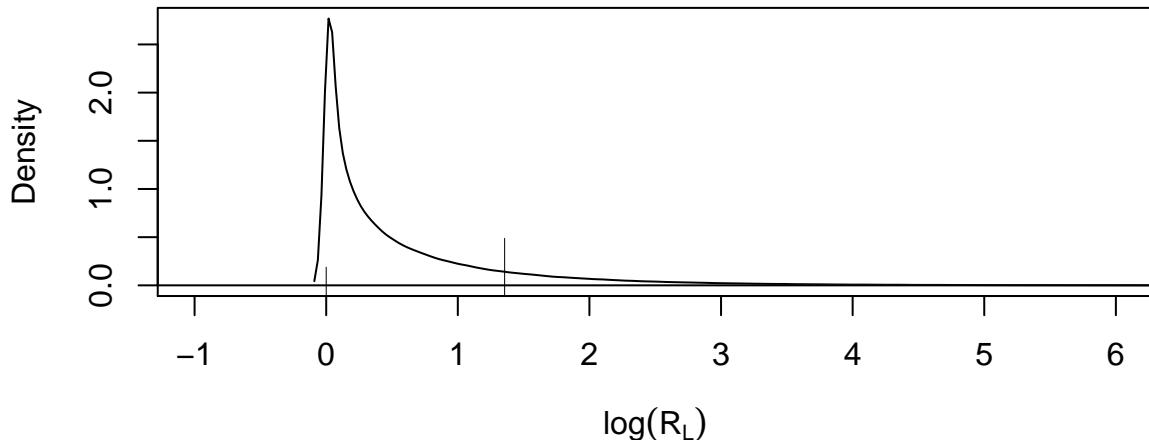


Figure 7.2: Density estimate of the log of the likelihood ratio based on parametric bootstrap resampling for samples of size 1 coming from the impurity proportion model. The rug values correspond to two different tests, one based on an observed measurement at $x = .37$ and the other (with the larger ticksize) based on an observed measurement at $x = .9$.

We will need new versions of the log likelihood and likelihood ratio functions in order to efficiently do the computations using the `sapply()` function in R. This function applies a given function to each column of an input data frame. In this case, we will create data frames consisting of $N = 1000$ Poisson $\alpha = 1$ samples (columns) of sizes 2, 5 and 100, respectively. The log of the likelihood ratio will be computed for each column.

```
# unnormalized log likelihood: for Poisson alpha distribution
ll <- function(alpha, x) {
  sum(log(alpha)*x - alpha)
}
```

In the above code, we have omitted the $\log(x!)$ term which would be expected in the log of the Poisson likelihood. Thus, we refer to the result as the unnormalized log likelihood. The $\log(x!)$ is not needed in our calculation of the log likelihood ratio because it is contained in both log likelihood expressions, and is thus cancelled out when the log likelihood ratio is computed, as in the next function.

```
# likelihood ratio (alpha is null value)
# Note that the MLE for alpha is the average of the sample values.
lr <- function(alpha, x) {
  ll(mean(x), x) - ll(alpha, x)
}

# random generator of likelihood ratios under the null alpha value
# based on N samples of size n
rlr <- function(N, n, alpha) {
  x1 <- data.frame(matrix(rpois(n*N, lambda = alpha), nrow=n))
  sapply(x1, function(x) lr(alpha, x))
}
```

The true value of α is taken to be the null hypothesis value, 1, for the simulations:

```
# compare observed density with chisquare on 1 df
par(mfrow=c(1, 3), mar=c(4, 4, .1, .1))
y <- 2*rlr(1000, n = 2, alpha = 1)
qqplot(y, rchisq(1000, df = 1), ylab="")
abline(0,1)
y <- 2*rlr(1000, n = 5, alpha = 1)
qqplot(y, rchisq(1000, df = 1), ylab="")
abline(0,1)
y <- 2*rlr(1000, n = 100, alpha = 1)
qqplot(y, rchisq(1000, df = 1), ylab="")
abline(0,1)
```

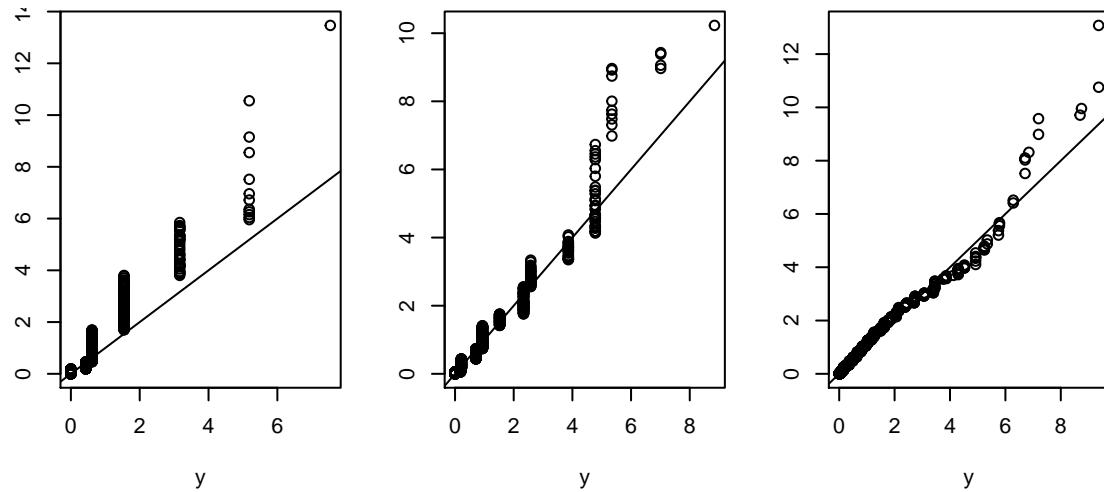


Figure 7.3: QQ-plots of log likelihood ratio multiplied by 2 versus the chi-square distribution with 1 degree of freedom, for Poisson $\lambda = 1$ data, with samples of size 2 (left panel), 5, (middle panel) and 100 (right panel).

Figure 7.3 displays single degree of freedom chi-square QQ-plots for $2 \log(R_L)$ for a samples of size 2, 5 and 100. The first two plots show the discrete nature of the distribution clearly, so the chi-square approximation is not accurate for these sample sizes (2 and 5), but the third plot shows points very close to a straight line, indicating that the distribution of the log of the likelihood ratio is very much like the distribution of random chi-square variables on 1 degree of freedom. Thus, we have a demonstration that the chi-square approximation is reasonable for this model as n gets large. The large sample distribution of twice the log likelihood ratio is approximately χ^2 on one degree of freedom, when the null hypothesis is true.

7.1.5 The distribution of the log likelihood ratio in the normal case

Showing that the log likelihood ratio has an asymptotic χ^2 distribution requires some effort in the general case. In the case of normally distributed observations, it is possible to obtain the result, exactly, relatively easily.

The log likelihood, as a function of the mean only, for independent observations x_1, x_2, \dots, x_n coming from a normal population with mean μ and variance σ^2 is

$$\ell(\mu) = -\frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \mu)^2 - \frac{n}{2} \log(2\pi) - \frac{n}{2} \log(\sigma^2).$$

When testing the null hypothesis that $\mu = \mu_0$, the log likelihood ratio is

$$\log(R_L) = \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \mu_0)^2 - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \bar{X})^2$$

since the MLE for μ is the sample average, \bar{x} . This expression simplifies to

$$\frac{n(\bar{X} - \mu_0)^2}{2\sigma^2}.$$

Thus,

$$2 \log(R_L) = \frac{n(\bar{X} - \mu_0)^2}{\sigma^2}$$

which has an exact χ^2 distribution on 1 degree of freedom for any sample size n .

7.2 The exponential family of distributions

A probability density function $f(x)$ is said to belong to the exponential family of distributions when

$$f(x) = e^{(x\theta - b(\theta))/a(\phi) + c(x, \phi)}$$

for some functions a, b and c . The parameter θ is referred to as the canonical parameter. The parameter ϕ is often referred to as the dispersion parameter.

Examples – normal and Poisson

Exponential family distributions are very common. The normal distribution is an example:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}.$$

The canonical parameter is $\theta = \mu$, $b(\theta) = \mu^2/2$, $\phi = \sigma^2$, $a(\phi) = \sigma^2$, and $c(x, \phi) = -x^2/(2\phi) - \log(\sqrt{2\pi\phi})$.

Another example is the Poisson distribution

$$p(x) = \frac{e^{-\lambda} \lambda^x}{x!}.$$

This time, the canonical parameter is $\theta = \log(\lambda)$, $b(\theta) = \lambda = e^\theta$, $\phi = 1$, $a(\phi) = 1$, and $c(x, \phi) = -\log(x!)$. Other examples include the Bernoulli distribution, binomial distribution, gamma distribution, and geometric distribution. Notable examples of distributions that are not members of the exponential family are the uniform distribution and other beta distributions.

7.2.1 Expectation and variance

The log likelihood of an exponential family distribution is

$$\ell(\theta) = \frac{x\theta - b(\theta)}{a(\phi)} + c(x, \phi).$$

Differentiating this with respect to θ results in

$$\frac{\partial \ell}{\partial \theta} = \frac{x - b'(\theta)}{a(\phi)}. \tag{7.4}$$

Taking expectations on both sides of this yields

$$E\left[\frac{\partial \ell}{\partial \theta}\right] = \frac{E[x] - b'(\theta)}{a(\phi)}. \tag{7.5}$$

Here, the expectation is assumed to be with respect to $f(x; \theta)$. We saw earlier in (7.1) that the expected value of the derivative of the log likelihood must be 0, so we are led to the identity

$$b'(\theta) = E[x].$$

The inverse of the function $b'(\theta)$ is referred to as the canonical link function since it links the canonical parameter θ with the expectation of the random variable in question.

Taking a second derivative of the log likelihood function with respect to θ yields

$$\frac{\partial^2 \ell}{\partial \theta^2} = \frac{-b''(\theta)}{a(\phi)}.$$

Taking expectations on both sides of this equation, and recalling

$$E\left[\frac{\partial^2 \ell}{\partial \theta^2}\right] = -E\left[\left(\frac{\partial \ell}{\partial \theta}\right)^2\right],$$

and (7.4), we see that

$$\text{Var}(x) = b''(\theta)a(\phi).$$

Briefly consider the Poisson distribution as an example here. We saw that $b(\theta) = e^\theta = \lambda$. Differentiating twice, with respect to θ , we have $b''(\theta) = e^\theta = \lambda$. Since $a(\phi) = 1$, we have $\text{Var}(x) = \lambda$, as is well known.

7.2.2 Maximum likelihood estimation within the exponential family

We begin by fitting an exponential family model $f(y; \theta)$ when there is a single response value y . The log likelihood is given by

$$\ell(\theta) = \log(f(y; \theta)) = \frac{y\theta - b(\theta)}{a(\phi)} + c(y, \phi).$$

The maximum likelihood estimator for θ can be obtained by differentiating with respect to θ

$$\ell'(\theta) = \frac{y - b'(\theta)}{a(\phi)}$$

and finding the solution of $\ell'(\theta) = 0$. In other words, we must solve the equation

$$b'(\theta) = y$$

for θ .

Example – homogeneous Poisson model

If y follows a Poisson distribution with mean λ , the log likelihood is given by

$$\ell(\lambda) = y \log(\lambda) - \lambda - \log(y!)$$

so we see that $\theta = \log(\lambda)$, $a(\phi) = 1$, $c(y, \phi) = -\log(y!)$ and $b(\theta) = \lambda = e^\theta$. In terms of θ , the log likelihood is expressed as

$$\ell(\theta) = y\theta - e^\theta - \log(y!).$$

Differentiating $\ell(\theta)$ with respect to θ , we see that we must solve

$$b'(\theta) = e^\theta = y$$

for θ :

$$\hat{\theta} = \log(y)$$

and by invariance of maximum likelihood estimation, we also have

$$\hat{\lambda} = e^{\hat{\theta}} = y.$$

Example – homogeneous Bernoulli model

If y follows a Bernoulli distribution with mean p , the log likelihood is given by

$$\ell(p) = y \log(p) + (1 - y) \log(1 - p) = y \text{logit}(p) + \log(1 - p)$$

so we see that $\theta = \text{logit}(p)$, $a(\phi) = 1$, $c(y, \phi) = 0$, and $b(\theta) = \log(1 + e^\theta)$. Differentiating ℓ as a function of θ with respect to θ , our estimating equation for θ is

$$b'(\theta) = \frac{e^\theta}{1 + e^\theta} = y$$

which does not have a very useful solution, since y only takes on values 0 or 1. More data points are needed in order to make any progress with this model.

7.2.3 Estimation from samples of observations

When we have multiple response values, y_1, y_2, \dots, y_n where each $y_j \sim f(y_j; \theta_j)$, we can set up the log likelihood for each data point as done earlier in this section, and we find that the maximum likelihood estimator for θ_j is

$$\hat{\theta}_j = b'^{-1}(y_j)$$

for $j = 1, 2, \dots, n$. This type of model for the response values is referred to as a saturated model. It is impossible to construct a more flexible model for the given data than this, using likelihood considerations. The saturated model is very flexible and allows the closest fit to the data. When this model does not fit the data well, the implication is that the assumed underlying probability model is incorrect.

Observe that we have not had to assume independence in order to do the fitting; this is analogous, and indeed generalizes what we observed earlier about fitting models via least-squares: independence is not needed to do the parameter estimation, but statistical inference is not possible without inducing some restrictions on the form of dependence in the data. Practical modeling also requires restrictions on the form of the model. We consider the most restrictive form next.

7.2.4 Models for independent and identically distributed observations

When there are independent and identically distributed responses y_1, y_2, \dots, y_n (i.e. a random sample) from a probability density function in the exponential family, governed by parameter θ , the log likelihood is given by

$$\ell(\theta) = \sum_{i=1}^n \log(f(y_i; \theta)) = \sum_{i=1}^n \frac{y_i \theta - b(\theta)}{a(\phi)} + \sum_{i=1}^n c(y_i, \phi).$$

The maximum likelihood estimator for θ can once again be obtained by differentiating with respect to θ

$$\ell'(\theta) = \sum_{i=1}^n \frac{y_i - b'(\theta)}{a(\phi)}$$

and finding the solution of $\ell'(\theta) = 0$. In other words, we must solve the equation

$$b'(\theta) = \frac{1}{n} \sum_{i=1}^n y_i = \bar{y}$$

for θ .

Example – homogeneous Poisson model

For n independent observations from a Poisson distribution with mean λ , the log likelihood is given by

$$\ell(\lambda) = \sum_{i=1}^n y_i \log(\lambda) - n\lambda - \sum_{i=1}^n \log(y_i!).$$

Again, we take $\theta = \log(\lambda)$, and $b(\theta) = e^\theta$. In terms of θ , the log likelihood becomes

$$\ell(\theta) = \sum_{i=1}^n y_i \theta - ne^\theta - \sum_{i=1}^n \log(y_i!).$$

Differentiating this with respect to θ , we must solve

$$ne^\theta = \sum_{i=1}^n y_i$$

for θ :

$$\hat{\theta} = \log\left(\sum_{i=1}^n y_i/n\right) = \log(\bar{y})$$

and by invariance of maximum likelihood estimation, we also have

$$\hat{\lambda} = e^{\hat{\theta}} = \bar{y}.$$

Example – homogeneous Bernoulli model

If the responses follow a Bernoulli distribution with mean p , the log likelihood is given by

$$\ell(p) = \sum_{i=1}^n y_i \log(p) + \sum_{i=1}^n (1 - y_i) \log(1 - p) = \sum_{i=1}^n y_i \text{logit}(p) + n \log(1 - p)$$

and with $\theta = \text{logit}(p)$, the log likelihood can be expressed in terms of theta as

$$\ell(\theta) = \sum_{i=1}^n y_i \theta + n \log(1 - p(\theta))$$

where $p(\theta) = e^\theta / (1 + e^\theta)$. Differentiating the log likelihood with respect to θ , our estimating equation for θ is

$$\frac{e^\theta}{1 + e^\theta} = \frac{1}{n} \sum_{i=1}^n y_i = \bar{y}$$

which is the proportion of 1's in the observed sample.

7.2.5 Existence and uniqueness of the maximum likelihood estimator

When modelling within the exponential family of distributions, it is relatively straightforward to show that maximum likelihood estimation leads to a single parameter estimate.

When the log likelihood is written as

$$\ell(\theta) = \frac{y\theta - b(\theta)}{a(\phi)} + c(y, \phi)$$

with $a(\phi) > 0$ for all ϕ , and $b(\theta)$ is a smooth enough function of θ , we can differentiate the log likelihood with respect to θ . Setting this derivative to 0, we readily see that θ should be the solution of

$$b'(\theta) = y.$$

To see that the solution of this equation must be a maximizer, we differentiate the log likelihood one more time, obtaining

$$-b''(\theta)/a(\phi).$$

However, we have already observed that $a(\phi) > 0$, and we saw earlier that

$$\text{Var}(X) = b''(\theta)a(\phi).$$

Therefore $b''(\theta) \geq 0$, since variances are never negative. This means that the second derivative of the log likelihood is negative. The solution to the optimization problem is a maximizer. (The log likelihood is a concave function.)

In fact, the maximizer will be unique whenever the variance of X is strictly positive, using a property of strictly concave functions. See Appendix C.3.1.

7.3 Exercises

1. (a) Suppose X_1, X_2, \dots, X_n are independent random variables taken from a population governed by the probability mass function

$$p(x) = \begin{cases} \theta, & x = 0, 1 \\ 1 - 2\theta, & x = 2 \end{cases} \quad (0 \leq \theta \leq 0.5)$$

- i. Find the maximum likelihood estimator for θ .
 - ii. Show that the maximum likelihood estimator for θ is unbiased.
 - iii. Find the standard error of the maximum likelihood estimator.
 - iv. Suppose the sample 0, 2, 1, 1, 1 has been observed. Calculate the maximum likelihood estimate of θ and estimate its standard error.
 - (b) Simulate a random sample of size 5 from the distribution described in the previous question, using $\theta = .1$. Estimate θ using the maximum likelihood estimator.
 - (c) Repeat the preceding question 100 times, and plot a histogram of the resulting estimates.
 - (d) Repeat the preceding question, but using samples of size 25. Compare the resulting histograms, and relate to the corresponding standard errors.
 - (e) Repeat questions (b), (c), and (d), but using a Poisson distribution with $\lambda = 3$.
2. Suppose $g(x)$ is a differentiable decreasing function with inverse $h(x)$, and X has a Poisson distribution with mean λ . Show that the maximum likelihood estimator for $g(\lambda)$ is $g(\hat{\lambda})$.
3. Refer to the previous question. Find the maximum likelihood estimator for $P(X > 0)$. Is this estimator unbiased?
4. Numbers of defects were counted in a random sample of 12 electrical appliances:

0 2 0 1 3 0 2 1 2 1 0 1

Use a Poisson distribution to model this data set:

$$P(X = x) = \frac{\lambda^x e^{-\lambda}}{x!}, \quad x = 0, 1, 2, \dots$$

- (a) Find the maximum likelihood estimate for λ .
- (b) Find the maximum likelihood estimate for the probability that the next appliance will have 4 defects.
- (c) Supposing that the sample size is large enough for the central limit theorem to apply, calculate an approximate 95% confidence interval for the true mean number of defects in this population of appliances.

5. Assuming that integration with respect to x and differentiation with respect to θ can be interchanged, show that (7.2) holds when the second derivative of the log likelihood exists.
6. Suppose you have two independent observations X_1 and X_2 from Poisson distributions with means λ and 2λ , respectively. Set $\hat{\lambda} = (X_1 + X_2)/3$.
 - (a) Show that $\hat{\lambda}$ is an unbiased estimator for λ .
 - (b) Find the standard error of $\hat{\lambda}$. Compare with the standard error of another unbiased estimator $X_2 - X_1$.
 - (c) Is $\hat{\lambda}$ the maximum likelihood estimator?
7. Show that the normal distribution is a member of the exponential family.
8. Show that the Poisson distribution is a member of the exponential family.
9. Show that the Bernoulli distribution is a member of the exponential family.
10. Show that the exponential distribution is a member of the exponential family.
11. Show that the geometric distribution is a member of the exponential family.
12. Show that the uniform distribution on the interval $[0, \theta]$ is not a member of the exponential family.
13. Calculate the first derivative of the log likelihood for the uniform distribution on the interval $[0, \theta]$ and find its expectation. Why does your result not contradict the theory that says such expected values are often 0?
14. Find the log likelihood ratio for testing the hypothesis that $p = p_0$ for n independent Bernoulli random variables having parameter p . Write R code to simulate the log likelihood ratio $\log(R_L)$ in the case where $p_0 = 0.5$, and verify that the distribution of $2 \log(R_L)$ is approximately χ^2 on 1 degree of freedom for large n . Conduct the simulation using $n = 10, 100$ and $n = 1000$.
15. **Cramer-Rao Inequality.** Suppose a measurement X is to be taken from a population governed by the density function $f(x; \theta)$, where θ is an unknown parameter, which could possibly be any real value. Suppose $f(x; \theta)$ has two continuous partial derivatives with respect to θ . Also, suppose an unbiased estimator for θ can be computed using the formula $\hat{\theta} = g(X)$.
 - (a) Show that

$$\frac{\partial \log(f)}{\partial \theta} = \frac{1}{f} \frac{\partial f}{\partial \theta}$$
 and

$$\frac{\partial^2 \log(f)}{\partial \theta^2} = - \left(\frac{\partial \log(f)}{\partial \theta} \right)^2 + \frac{1}{f} \frac{\partial^2 f}{\partial \theta^2}$$
 and conclude that

$$E \left[\frac{\partial^2 \log(f)}{\partial \theta^2} \right] = -E \left[\left(\frac{\partial \log(f)}{\partial \theta} \right)^2 \right].$$
 (You may interchange the order of integration and differentiation here.)
 - (b) Write down the Cauchy Schwarz inequality in integral form.
 - (c) Show that

$$V(g(X)) E \left[\left(\frac{\partial}{\partial \theta} \log(f(X)) \right)^2 \right] \geq 1.$$
 - (d) Deduce that

$$V(g(X)) = V(\hat{\theta}) \geq \frac{1}{E \left[\left(\frac{\partial}{\partial \theta} \log(f(X)) \right)^2 \right]}.$$

16. Suppose you have one observation X from an exponential distribution with mean β . Show that the maximum likelihood estimator for β has the minimum possible variance among all unbiased estimators for λ .
17. Suppose X is a normal random variable with parameters μ and σ . Show that the maximum likelihood estimator for μ has the minimum possible variance among all unbiased estimators for μ .

8

Generalized Linear Models - MLE with Covariates

8.1 Reducing model complexity through modelling with covariates

Modelling a set of observations with probability density functions where the unknown parameters are estimated by maximum likelihood estimation is often too restrictive when the goal is prediction. We saw in the previous chapter that a much more flexible model is the saturated model. The drawback of the saturated model is that the number of parameters to estimate is the same as the number of data points, so it is too complex to be useful. The saturated model is an example of extreme over-fitting while modelling an entire set of data with a single probability density function might be viewed as extreme under-fitting. Often, the best model for a given set of data will be found somewhere between these two extremes.

Inclusion of covariate information allows us to reduce the number of parameters in the model, while providing some predictive power. There is reduction in complexity but at a cost: the resulting model is likely to be less flexible, and the fit to the data will be worse than for the saturated model. We also need to combine the data values in a way that gives rise to good estimators for the remaining model parameters. At a minimum, we seek estimators with

1. low bias.
2. small variance (i.e. we seek efficiency).
3. consistency.

Ultimately, we will see that the log likelihood based on the joint distribution of the y 's provides a way to do this, but we will need to impose an independence assumption in order for that to be justified. The following example shows that independence is not actually required to produce an estimator, but independence provides a simple mechanism for demonstrating that such an estimator has the required properties listed above. In fact, the next sequence of examples illustrate how maximum likelihood estimation in the exponential family is just a disguised form of weighted nonlinear least-squares estimation.

8.1.1 A normal model via best linear unbiased estimation

Consider the regression through the origin model

$$y_j = \beta_1 x_j + \varepsilon_j$$

where ε_j is normally distributed with mean 0 and variance σ^2 . In other words, $y_j \sim N(\beta_1 x_j, \sigma^2)$, and this distribution is a member of the exponential family with canonical parameter $\theta_j = \beta_1 x_j$, with $b(\theta_j) = \frac{1}{2}\theta_j^2$. Thus, $b'(\theta_j) = \theta_j$. Fitting the saturated model is easy: $\hat{\theta}_j = y_j$, for $j = 1, 2, \dots, n$. Using the invariance property of maximum likelihood estimation and the modelling assumption $\theta_j = \beta_1 x_j$, we can see that maximum likelihood estimators for β_1 are $\hat{\beta}_1 = \bar{y}_j / \bar{x}_j$ for $j = 1, 2, \dots, n$.

A simple way to combine the above estimates into a single estimate would be to take the average. This would give us an estimator of the form

$$\hat{\beta}_1 = \sum_{j=1}^n \frac{1}{nx_j} y_j,$$

but then we might immediately ask why the coefficient of y_j is $\frac{1}{nx_j}$. Can we do better? The answer is “yes”. We will now find a good estimator of β_1 which is a linear combination of the y 's:

$$\hat{\beta}_1 = \sum_{j=1}^n a_j y_j = a^\top y \tag{8.1}$$

which is unbiased, and has the smallest possible variance among all such estimators. We first see that

$$E[\hat{\beta}_1] = \sum_{j=1}^n a_j E[y_j] = \beta_1 \sum_{j=1}^n a_j x_j = \beta_1$$

provided that $\sum_{j=1}^n a_j x_j = 1$. This is the condition required to obtain an unbiased estimator for β_1 . Note that we did not require an independence assumption yet.

To obtain a linear unbiased estimator with smallest possible variance, we now impose an independence assumption on the responses.¹ Then we can see that

$$\text{Var}(\hat{\beta}_1) = \sum_{j=1}^n a_j^2 \text{Var}(y_j) = \sigma^2 \sum_{j=1}^n a_j^2.$$

Here, we have used the assumption that all of the responses have the same variance, σ^2 . In order to obtain the smallest possible variance for the estimator of β_1 , we must solve the optimization problem:

$$\min_a a^\top a : \quad a^\top x = 1.$$

where a represents the n -vector of a -coefficients and x is the n -vector of covariate values. Lagrange multipliers can be used to solve this problem. The Lagrangian is

$$L = a^\top a + \lambda(a^\top x - 1).$$

Differentiating with respect to a and setting to 0, we have

$$2a + \lambda x = 0. \tag{8.2}$$

Pre-multiplying by a^\top and using the constraint, we obtain

$$2a^\top a + \lambda = 0$$

which tells us that $\lambda = -2a^\top a$.

Equation (8.2) can be used again:

$$2a = -\lambda x.$$

The squared norm of each side is

$$4a^\top a = \lambda^2 x^\top x$$

which, when λ is replaced by the expression given at the end of previous paragraph, gives

$$4a^\top a = 4(a^\top a)^2 x^\top x.$$

Simplifying this tells us that

$$a^\top a = \frac{1}{x^\top x}.$$

This allows us to express λ fully in terms of x : $\lambda = \frac{-2}{x^\top x}$. Plugging this into (8.2) and solving for a leads to

$$a = \frac{x}{x^\top x}$$

and plugging this into (8.1) gives

$$\hat{\beta}_1 = \frac{\sum_{j=1}^n x_j y_j}{\sum_{j=1}^n x_j^2}. \tag{8.3}$$

Recall from our initial discussions of simple linear regression that this estimator is the least-squares estimator for the slope in the regression through the origin model. What we have proved here is a version of the Gauss-Markov theorem: least-squares estimation leads to best linear unbiased estimation in simple regression with independent observations having constant variance. Note that the adjective ‘best’ has a specific meaning here: that is, minimum variance.

¹Independence is used here for mathematical simplicity. Other restrictions on the dependence could be imposed, leading to different results.

8.1.2 The normal model via maximum likelihood

We now show that maximum likelihood estimation for the regression through the origin model with independent normal errors gives rise to the same estimation scheme as linear least-squares. To this end, note that the log likelihood for a single observation $y_j \sim N(\beta_1 x_j, \sigma^2)$ is

$$\ell_j(\beta_1) = \frac{y_j \theta_j - b(\theta_j)}{a(\phi)} + c(y_j, \phi) = \frac{y_j x_j \beta_1 - b(\beta_1 x_j)}{a(\phi)} + c(y_j, \phi)$$

and for n independent observations, the log likelihood is

$$\ell(\beta_1) = \sum_{j=1}^n \left\{ \frac{y_j x_j \beta_1 - b(\beta_1 x_j)}{a(\phi)} + c(y_j, \phi) \right\}.$$

Differentiating this with respect to β_1 yields

$$\sum_{j=1}^n \frac{y_j x_j - b'(\beta_1 x_j) x_j}{a(\phi)}$$

and noting that $b'(\beta_1 x_j) = \beta_1 x_j$ for the normal model leads to the estimating equation

$$\sum_{j=1}^n (y_j x_j - \beta_1 x_j^2) = 0.$$

Solving for β_1 gives us the least-squares estimator as at (8.3).

8.1.3 Example – Poisson regression through the origin

This time, we suppose y_j has a Poisson distribution with parameter $\theta_j = x_j \beta$. The log likelihood expressed as an exponential family model is

$$\ell_j = y_j \beta x_j - b(\beta x_j) + c(y_j, \phi)$$

where $a(\phi) = 1$, and $b(\theta_j) = e^{\theta_j}$. Differentiating with respect to β gives the estimating equation

$$y_j x_j - b'(\beta x_j) x_j = y_j x_j - e^{\beta x_j} x_j = 0. \quad (8.4)$$

The MLE of β is then $\hat{\beta} = \frac{\log(y_j)}{x_j}$.

If we have observations y_1, y_2, \dots, y_n , we are again confronted with the question as to how to combine estimates of the above form to obtain a single good estimate. Taking the cue from the previous (normal error) example where the least-squares estimator was best possible, we re-consider the estimating equations (8.4) from the stand-point of least-squares. However, because the equation is nonlinear and because the variance of y_j is nonconstant (i.e. $\text{Var}(y_j) = e^{\beta x_j}$) we must use weighted nonlinear least-squares. Recall that the weights are the inverses of the variances.

From (8.4), we have

$$y_j = e^{\beta x_j}$$

so the weighted least-squares problem is to minimize

$$\sum_{j=1}^n (y_j - e^{\beta x_j})^2 e^{-\beta x_j}$$

with respect to β .

As will be seen in the numerical example that follows, we will need to assume a value of β in order to calculate the weights. Therefore, we need to practically solve a sequence of minimizations of the form

$$\sum_{j=1}^n (y_j - e^{\beta x_j})^2 e^{-\beta^* x_j}$$

where β^* is fixed at the previously obtained estimate. The nonlinear least-squares problem then proceeds by solving the estimating equation

$$\sum_{j=1}^n (y_j - e^{\beta x_j}) x_j e^{(\beta - \beta^*) x_j} = 0 \quad (8.5)$$

which is obtained by differentiating the objective function with respect to β (but not β^*).

Example - numerical calculation based on simulated data

This problem can be solved using the `nls()` function in R. We illustrate on 10 simulated observations where y_j is Poisson distributed with mean e^{2x_j} . Note that the weights depend on the true value of β , so we need to use a starting guess for the weights as well as for the nonlinear least-squares iteration.

The following code uses a starting guess of $\beta^* = 1$.

```
x <- runif(10)
betatrue <- 2
betastar <- 1
y <- rpois(10, lambda = exp(betatrue*x))
xy <- data.frame(x, y)
y.nls <- nls(y ~ exp(beta*x), start = list(beta = betastar),
  weights = exp(-betastar*x), data = xy)
coef(y.nls)

##      beta
## 2.091801
```

Now, we need to update the weights using the estimate of β , 2.091801, as a starting value:

```
y.nls <- nls(y ~ exp(beta*x), start = list(beta = coef(y.nls)),
  weights = exp(-coef(y.nls)*x), data = xy)
coef(y.nls)

##      beta
## 2.099352
```

Note that the resulting estimate is quite close to the first one. Normally, we would run this procedure, updating the weights each time, until we achieve convergence to within a given tolerance. One more run with the latest value of β gives convergence to the number of digits that we are currently working with:

```
y.nls <- nls(y ~ exp(beta*x), start = list(beta = coef(y.nls)),
  weights = exp(-coef(y.nls)*x), data = xy)
coef(y.nls)

##      beta
## 2.099411
```

Note that the estimate of β is 2.0994112 which is quite close to the true value, $\beta = 2$.

The above technique of iterating the weighted nonlinear least-squares with updated weights at each stage is called iteratively reweighted least-squares or IRLS.

8.1.4 Poisson regression through the origin via maximum likelihood

It is important to note that the nonlinear least-squares analysis above did not rest on an assumption of independence. Of course, it will be difficult to conduct statistical tests on the regression coefficient. If we assume independence, the likelihood approach will lead to the same estimation procedure as we obtained in the previous section.

The log likelihood based on the n observations becomes

$$\ell(\beta) = \sum_{j=1}^n y_j \beta x_j - \sum_{j=1}^n b(\beta x_j) + \sum_{j=1}^n c(y_j, \phi)$$

Differentiating this with respect to β gives

$$\frac{\partial \ell}{\partial \beta} = \sum_{j=1}^n (y_j x_j - b'(\beta x_j) x_j)$$

which, when the Poisson assumption is invoked, i.e. $b'(\beta x_j) = e^{\beta x_j}$, gives the estimating equation for the MLE for β as

$$\frac{\partial \ell}{\partial \beta} = \sum_{j=1}^n x_j (y_j - e^{\beta x_j}) = 0.$$

Compare this with the nonlinear regression estimating equation at (8.5) where there is an additional term of the form $e^{(\beta-\beta^*)x_j}$. At each stage of the iteration, β is estimated using weighted nonlinear least-squares to be $\hat{\beta}$, using β^* as the starting guess, so at convergence $\beta^* = \hat{\beta}$. Thus, the term has no effect on the resulting estimate of β , apart from the number of iterations required for convergence. Thus, the maximum likelihood estimator is numerically equivalent to the weighted nonlinear least-squares estimator, and iteratively re-weighted least-squares can be used to do the calculation.

Example - revisiting the simulated data with the `glm()` function

Using the `glm()` function, with the `poisson` family, we can estimate the Poisson regression through the origin model for the simulated data as follows:

```
y.glm <- glm(y ~ x - 1, family = poisson)
coef(y.glm)

##           x
## 2.099411
```

Note that we needed to use `- 1` in order to remove the intercept. The value of the estimate is identical to the one obtained earlier, using weighted nonlinear least-squares. This is not an accident, since both were based on the same algorithm, iteratively reweighted least-squares.

8.1.5 Why does the algorithm work without specifying the correct weights at the start?

Recall from the discussion of weighted least-squares and nonlinear least-squares that the weights are needed to increase the efficiency of the estimators. That is, to ensure that their standard errors are as small as possible. The unweighted estimators are still consistent, so the use of approximate weights at each step of the iteration is justified.

The weights will be more accurate at each step, so the end result is the same as if the weights had been optimized at each step, or if the weights had been chosen correctly to begin with.

8.2 Iteratively reweighted least-squares for MLE in the exponential family

8.2.1 Implementation when the link function is canonical

We are given n independent observations on a response variable y that follows an exponential family distribution, together with corresponding observations on k predictor variables x_1, x_2, \dots, x_k . As usual we have

$$\ell_j(\theta_j) = \frac{y_j \theta_j - b(\theta_j)}{a(\phi)} + c(y_j, \phi)$$

and because we are using the canonical link function, we have

$$\theta_j = \mathbf{X}_j \beta$$

where \mathbf{X}_j denotes the j th row of the design matrix \mathbf{X} . The vector β contains the regression coefficients. With this setup, it is sufficient to minimize the nonlinear least-squares objective function

$$\sum_{j=1}^n (y_j - b'(\theta_j))^2 / \text{Var}(y_j)$$

with respect to the coefficients of β . We have seen earlier that $\text{Var}(y_j) = b''(\theta_j)a(\phi)$, so the objective function can be re-written as and simplified to

$$\sum_{j=1}^n (y_j - b'(\mathbf{X}_j \beta))^2 / b''(\mathbf{X}_j \beta).$$

If we knew the denominator weights, this would be an ordinary weighted nonlinear least-squares problem. As argued earlier, we can try fairly arbitrary weights as a starting guess, and then solve a succession of weighted nonlinear least-squares problems where the weights are updated at each stage.

Thus, at stage m , we assume that we have an estimate $\hat{\beta}^*$. To obtain the $(m + 1)$ st updated estimate of $\hat{\beta}$, we apply nonlinear least-squares to minimize

$$\sum_{j=1}^n (y_j - b'(\mathbf{X}_j \beta))^2 / b''(\mathbf{X}_j \beta^*)$$

with respect to β . When the minimizer is within a prescribed tolerance of β^* , we declare convergence, and return the current value of β as the estimate.

In the next example, we apply the approach described above directly to the problem of estimating the intercept and slope for logistic regression. In other examples, we will bypass all of the details with a call to the `glm()` function. Being able to program the problems using `nls()` provides the user with more flexibility than is currently available in the `glm()` function.

Example – R code for binary data with logit link

For a binomial random variable y_j having mean $\mu_j = p_j$, the log likelihood, written in the form of the exponential family, is

$$\ell_j(\theta_j) = y_j \theta_j - b(\theta_j)$$

where $\theta_j = \log(p_j/(1 - p_j))$, and $b(\theta_j) = \log(1 + e^{\theta_j})$. That is, the logit function is the canonical link function. Differentiating twice, we have

$$b'(\theta_j) = \frac{e^{\theta_j}}{1 + e^{\theta_j}}$$

and

$$b''(\theta_j) = \frac{e^{\theta_j}}{(1 + e^{\theta_j})^2}.$$

With the model where $\theta_j = \beta_0 + \beta_1 x_j$, we have

$$b'(\theta_j) = e^{\beta_0 + \beta_1 x_j} / (1 + e^{\beta_0 + \beta_1 x_j})$$

and

$$b''(\theta_j) = e^{\beta_0 + \beta_1 x_j} / (1 + e^{\beta_0 + \beta_1 x_j})^2$$

The weighted nonlinear least-squares objective function to be solved iteratively is

$$\sum_{j=1}^n (y_j - e^{\beta_0 + \beta_1 x_j} / (1 + e^{\beta_0 + \beta_1 x_j}))^2 / b''(\beta_0^* + x_j \beta_1^*).$$

The following function implements one step of the iteration:

```
logitstep <- function(x, y, bstar) {
  logit.nls <- nls(y ~ exp(b0 + b1*x) / (1+exp(b0+b1*x)),
    start = c(b0 = bstar[1], b1 = bstar[2]),
    weights = (1+exp(bstar[1]+bstar[2]*x)) ^ 2 / exp(bstar[1] + bstar[2]*x))
  coef(logit.nls)
}
```

To test the function, we consider 20 observations of simulated data from a model where $\text{logit}(p) = 1 + 2x$, where the x 's are uniformly distributed on the interval $(-3, 2)$:

```
x <- runif(20, min = -3, max = 2)
y <- rbinom(20, size = 1, prob = exp(1 + 2*x) / (1 + exp(1 + 2*x)))
```

Running one iteration of the nonlinear least-squares with a starting value of $(0, 0)$ gives

```
betastar <- c(0, 0)
betaout <- logitstep(x, y, bstar = betastar)
betaout

##          b0          b1
## 0.2571402 1.8038932
```

Running a second iteration, with the output from the previous iteration as the starting value, we have

```
betastar <- as.numeric(betaout)
betaout <- logitstep(x, y, bstar = betastar)
betaout

##          b0          b1
## 0.425671 1.827530
```

The following code could be used to iterate the method until the updated estimate is within 10^{-7} of the input estimate:

```
while (sqrt(sum((betaout - betastar) ^ 2)) > 1e-7) {
  betastar <- as.numeric(betaout)
  betaout <- logitstep(x, y, bstar = betastar)
}
betaout

##          b0          b1
## 0.4517607 1.8838329
```

The fitted model is $\text{logit}(\hat{p}) = 0.452 + 1.884x$. Compare this with the truth: $\text{logit}(p) = 1 + 2x$.

Example – car purchase data

The data in p13.5 (MPV package) are concerned with whether an individual purchases a new vehicle within 6 months, given current income (x_1) and the age of their oldest vehicle (x_2). The response variable y takes on the value 1 if the purchase takes place. To ensure that the scales of the variables are similar, it is useful to consider income in thousands of dollars:

```
p13.5$inc1000 <- p13.5$x1/1000
```

```
summary(p13.5)
```

```
##          x1            x2            y      inc1000
##  Min.   :27000   Min.   :1.00   Min.   :0.0   Min.   :27.00
##  1st Qu.:37375   1st Qu.:2.00   1st Qu.:0.0   1st Qu.:37.38
##  Median :46500   Median :3.50   Median :0.5   Median :46.50
##  Mean    :47275   Mean    :3.75   Mean    :0.5   Mean    :47.27
##  3rd Qu.:53500   3rd Qu.:5.00   3rd Qu.:1.0   3rd Qu.:53.50
##  Max.    :75000   Max.    :9.00   Max.    :1.0   Max.    :75.00
```

Since the response is binary, we could try a logistic model. We could modify the code from the previous example, but we will instead demonstrate the use of the `glm()` function here:

```
y.logit <- glm(y ~ inc1000 + x2, data = p13.5, family = binomial(link = "logit"))
summary(y.logit)

##
## Call:
## glm(formula = y ~ inc1000 + x2, family = binomial(link = "logit"),
##      data = p13.5)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.5635  -0.8045  -0.1397   0.9535   1.7915
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.04706   4.67423 -1.508   0.132
## inc1000     0.07382   0.06371  1.159   0.247
## x2          0.98789   0.52737  1.873   0.061
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 27.726 on 19 degrees of freedom
## Residual deviance: 21.082 on 17 degrees of freedom
## AIC: 27.082
##
## Number of Fisher Scoring iterations: 5
```

The fitted model is as follows:

$$\widehat{\log}(p/(1-p)) = -7.047 + 0.074x_1 + 0.988x_2.$$

8.2.2 Implementation when the link is not canonical

When the function linking the mean of the responses to the covariates is not the canonical link function, the same general technique applies. The only difference is that when applying each nonlinear least-squares step, the inverse of the link function is used in place of the inverse of the derivative of $b(\theta)$. The m th step of the IRLS algorithm requires minimizing

$$\sum_{j=1}^n (y_j - g^{-1}(\mathbf{X}_j \beta))^2 / b''(\theta_j^*) \quad (8.6)$$

where $\theta_j^* = b'^{-1}(\mu_j^*) = b'^{-1}(g^{-1}(X_j\beta^*))$. The equation to solve at this step is

$$\sum_{j=1}^n \frac{(y_j - g^{-1}(X_j\beta))X_j}{b''(\theta_j^*)g'(g^{-1}(X_j\beta^*))} = 0.$$

This equation is obtained by differentiating the function at (8.6) with respect to β or by differentiating the log likelihood directly.

Example – exponential distribution with log link

The canonical link for the exponential distribution is the reciprocal, but often, a log link makes more sense:

$$g(\mu_j) = \log(\mu_j) = X_j\beta$$

where X_j is the j th row of the design matrix. Recalling that $\theta_j = -1/\mu_j$, $b(\theta_j) = -\log(-\theta_j)$, and so on, we can see that

$$b'(\theta_j) = \mu_j = e^{X_j\beta}$$

and

$$b''(\theta_j) = \mu_j^2 = e^{2X_j\beta}.$$

Therefore the weighted nonlinear least-squares objective function to be minimized at each step of IRLS is

$$\sum_{j=1}^n (y_j - e^{X_j\beta})^2 e^{-2X_j\beta}.$$

Given the previous estimate at β^* , the updated estimate of β is calculated by solving the nonlinear equation

$$\sum_{j=1}^n (y_j - e^{X_j\beta})X_j e^{-X_j\beta^*} = 0.$$

In the next example, we exhibit R code that shows the precise steps required in every step of the computation of the MLE using IRLS for exponential data using the identity link function.

Example – windspeed data and raw R code for fitting the exponential model

Recall the Winnipeg windspeed data, `windWin80` (*MPV* package) where interest centers on predicting noon hour windspeed from the previous midnight's value. We had fit an exponential model to the 5/3 power of the noon hour responses, relating the exponential rate to the reciprocal of the linear model in the midnight windspeed.

The following code can be used to fit a generalized model where the response is assumed to be exponentially distributed. The reciprocal link function is the canonical link, but the identity link function is being used here, since we require that the mean be linearly related to the covariates.

```
glm.exp <- function(X, y, beta.old) {
  n <- length(y)
  X <- cbind(rep(1, n), X)
  p <- length(beta.old)
  for (i in 1:10) {
    W <- diag(as.vector(1 / (X %*% as.matrix(beta.old))^2))
    Xprime <- sqrt(W) %*% X
    yprime <- sqrt(W) %*% y
    X.QR <- qr(Xprime)
    Q <- qr.Q(X.QR, complete = TRUE)
    R <- qr.R(X.QR, complete = TRUE)
    Q1 <- Q[, 1:p]
    U <- R[1:p, 1:p]
```

```

beta <- backsolve(U, t(Q1) %*% yprime)
beta.old <- beta
}
beta
}

```

Applying the code to the data set gives values for the coefficients which are very similar to what we obtained earlier:

```

windFit <- glm.exp(windWin80$h0, windWin80$h12^1.66, c(1, 1))
windFit

##          [,1]
## [1,] 108.140731
## [2,] 5.297837

```

The predicted windspeeds (raised to the power 5/3) at noon are modelled as exponential random variables with mean

$$\mu = 108.1 + 5.3h_0.$$

Example - car purchase, `glm()` function and a probit model

An example of a link function which is not canonical for binomial data is the probit function. We apply a probit model to the car purchase data which was modeled earlier with a logit model.

```

y.probit <- glm(y ~ inc1000 + x2, data = p13.5, family = binomial(link = "probit"))
summary(y.probit)

##
## Call:
## glm(formula = y ~ inc1000 + x2, family = binomial(link = "probit"),
##      data = p13.5)
##
## Deviance Residuals:
##       Min      1Q   Median      3Q     Max
## -1.5640  -0.8053  -0.1545   0.9542   1.8009
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.34969   2.72227 -1.598   0.1101
## inc1000     0.04559   0.03787  1.204   0.2286
## x2          0.60992   0.29966  2.035   0.0418
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 27.726 on 19 degrees of freedom
## Residual deviance: 20.954 on 17 degrees of freedom
## AIC: 26.954
##
## Number of Fisher Scoring iterations: 6

```

The fitted model is as follows:

$$\widehat{\text{probit}}(p) = -4.35 + 0.046x_1 + 0.61x_2.$$

From this model, and given values of x_1 and x_2 , one can estimate the probability that $y = 1$ from the probability that a standard normal random variable takes on a value less than $-4.35 + 0.046x_1 + 0.61x_2$. For example, someone with an annual salary of \$50000 and a 7-year old car would be inclined, according to the fitted model, to buy a new car within the next 6 months with probability

```
estProb <- pnorm(coef(y.probit) %*% c(1, 50, 7))
round(estProb, 3) # round reduces number of displayed digits
##      [,1]
## [1,] 0.986
```

Example - complementary log-log link

The complementary log-log function is another popular link function for binomial responses:

```
y.cloglog <- glm(y ~ inc1000 + x2, data = p13.5, family = binomial(link = "cloglog"))
summary(y.cloglog)

##
## Call:
## glm(formula = y ~ inc1000 + x2, family = binomial(link = "cloglog"),
##      data = p13.5)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.5079   -0.8076   -0.2252    0.9941    1.7706
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.73696   3.31011 -1.733   0.0831
## inc1000     0.05680   0.04412  1.288   0.1979
## x2          0.72191   0.35150  2.054   0.0400
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 27.726 on 19 degrees of freedom
## Residual deviance: 20.855 on 17 degrees of freedom
## AIC: 26.855
##
## Number of Fisher Scoring iterations: 7
```

This relates the mean response p to the covariates through the function

$$\log(-\log(1-p)) = \beta_0 + \beta_1 x_1 + \beta_2 x_2.$$

8.3 Statistical inference for generalized linear models

For a particular model (which we will call the reduced model), the deviance is defined as

$$D = 2\phi(\ell_{\text{saturated}} - \ell_{\text{reduced model}}).$$

Recalling that the log likelihood for the saturated model would be the maximum possible value for the data and the model family, the deviance then measures the decrease in the likelihood that accrues from considering a simpler

model (i.e. the reduced model). Note that the deviance is a log likelihood ratio for a test where one would be checking whether the saturated model is more appropriate than the reduced model.

The null deviance is the deviance when the reduced model is the simplest possible model, i.e. where there are no covariates, only an intercept. In the case of a normally distributed response, the null deviance is equivalent to the (corrected) total sum of squares. The residual deviance is the deviance when the reduced model is a particular model involving covariates. For the case of normally distributed responses, the residual deviance is equivalent to the residual sum of squares.

The residual deviance and the null deviance can be extracted from the model object directly using the `summary()` function:

```
summary(y.logit)$deviance # residual deviance for the logit model on p13.5 data
## [1] 21.08152
summary(y.logit)$null.deviance
## [1] 27.72589
```

8.3.1 The scaled deviance and the chi-squared distribution

When ϕ is known, or can be estimated, the scaled deviance can be calculated as

$$D^* = D/\phi.$$

For the normal model, the scaled deviance follows a χ^2 distribution. For example, the scaled deviance for a linear model with p parameters has a χ^2 distribution on $n - p$ degrees of freedom.

It is possible to show that the scaled deviances for some non-normal models have approximate χ^2 distributions. For example, in the Poisson family, with p regression coefficients, we might expect the scaled deviance for to have an approximate χ^2 distribution on $n - p$ degrees of freedom. Because the expected value of a χ^2 random variable is equal to its degrees of freedom, a basic model check consists of comparing the residual deviance to its degrees of freedom; if the residual deviance is much greater than its degrees of freedom, it is unlikely that the model is adequate for the data.

For the logistic model, the asymptotic approximation of the scaled deviance to the χ^2 distribution on $n - p$ degrees of freedom is not valid.

Example - Poisson regression for cigarette data

For the cigarette butt data of Section B.3.2 (see also Section 8.4.4), we can fit a Poisson regression model to relate expect count to distance, and the residual deviance as follows:

```
cig.glm <- glm(count ~ distance, data = cigbutts, family = poisson)
summary(cig.glm)$deviance # the scaled deviance since phi is taken as 1
## [1] 9.003466
```

Here, we have fit a model with $p = 2$ parameters and there are 15 observations. Thus, the scaled deviance is slightly less than expected (i.e. $n - p = 13$).

Example - missile success-failure data

We consider the data of p13.1 from the *MPV* package here, concerning successes and failures of missile firing. A logistic regression model is fit as follows:

```
missile.glm <- glm(y ~ x, data = p13.1, family = binomial)
summary(missile.glm)$deviance
## [1] 20.36366
```

Here, we have fit a logistic regression model with $p = 2$ parameters and there are 25 observations. There is a relatively close match between the scaled deviance here and the residual degrees of freedom, but this is somewhat coincidental, as shown in the next example, involving simulated data.

Example - simulated Bernoulli data

Here we simulate 100 observations from the model

$$p(x) = \frac{e^{x+2}}{1 + e^{x+2}}$$

where the x values are randomly uniformly distributed on the interval $(0, 1)$:

```
n <- 100
x <- runif(100)
y <- rbinom(n, 1, exp(x+2) / (1+exp(x+2)))
sim.df <- data.frame(x = x, y = y); rm(x, y)
```

Fitting the logistic model (to estimate the slope and intercept parameters) is as follows:

```
sim.glm <- glm(y ~ x, data = sim.df, family = binomial)
```

If the asymptotic χ^2 distribution on $n - p = 98$ degrees of freedom was reasonable, we would expect the scaled deviance to be near 98. Instead it is calculated as

```
summary(sim.glm)$deviance
## [1] 19.35108
```

This shows clearly that the asymptotics do not work for the Bernoulli case.

8.3.2 Analysis of deviance is a new form of ANOVA

It is also possible to set up approximate tests and confidence intervals for generalized linear models in much the same way as for normal regression. Recall that for normal linear regression, partial F -tests were set up by essentially comparing deviances of a full model (having p coefficients, say) and a reduced model (having r coefficients, say). Analysis of variance tests based on χ^2 statistics on $p - r$ degrees of freedom and resulting F statistics on $p - r$ and $n - p$ could be used to test the null hypothesis that the reduced model is true versus the alternative that some or all of the other coefficients were nonzero.

Example - does income matter when deciding to buy a car?

Earlier, we fit a model to predict whether an individual with a given income and a car of a certain age would be likely to purchase a car within 6 months, using the p13.5 data set. In order to test if income should be added in to a model that already contains the other variable, we need to fit

```
y.logit2 <- glm(y ~ x2, data = p13.5, family = binomial(link = "logit"))
```

Then we conduct the approximate partial F-test, using the `anova()` function:

```
anova(y.logit2, y.logit)

## Analysis of Deviance Table
##
## Model 1: y ~ x2
## Model 2: y ~ inc1000 + x2
##   Resid. Df Resid. Dev Df Deviance
## 1       18    22.563
## 2       17    21.081  1    1.4818
```

We see that this is actually an Analysis of Deviance, not ANOVA. Note that there is no p -value associated with this test, so we must make a judgement call, or use some other technique to weigh the evidence.

8.3.3 Assessing the weight of evidence using the bootstrap

The following procedure will often provide usable p -values for analysis of deviance tests. In what follows, we compare two models for a given dataset where y is the response and there are $p - 1$ predictor variables. The full model contains all variables plus an intercept. We will denote its residual deviance by D_1 , and its approximate expected value will be $n - p$. The reduced model contains only $p_0 - 1$ of the predictor variables plus an intercept. We will denote its residual deviance by D_0 , and its approximate expected value is $n - (p_0 - 1)$ when the reduced model is the true model. The analysis of deviance statistic is

$$F^* = \frac{(D_0 - D_1)/(p - p_0)}{D_1/(n - p)}$$

which is approximately distributed as F on $p - p_0$ and $n - p$ degrees of freedom.

Rather than comparing with an F-distribution, we recommend a parametric bootstrap. The procedure to follow is:

1. Calculate D_0 , D_1 and F^* .
2. Simulate R datasets according to the reduced model. Each dataset should have the same number of observations as the original dataset.
3. For each simulated dataset, compute
 - (a) D_0 . Call it D_0^* .
 - (b) D_1 . Call it D_1^* .
 - (c) $F^{**} = \frac{(D_0^* - D_1^*)/(p - p_0)}{D_1^*/(n - p)}$
4. Calculate the proportion of F^{**} values that exceed F^* . This is the bootstrap estimate of the p -value for the analysis of deviance test.

You may need an estimate of ϕ for this purpose, but the `glm()` output includes this if it is needed. To do the simulation, you only need to simulate the response values; the covariate values can be left unchanged. The `simulate()` function can often be used, simplifying the coding requirements.

Example – car data

We repeat the analysis of deviance on the car purchase problem of p13.5 using the bootstrap procedure. We first collect all of the information for the test for the original data:

```

y.reduced <- glm(y ~ x2, data = p13.5, family = binomial(link = "logit"))
y.full <- glm(y ~ x1 + x2, data = p13.5, family = binomial(link = "logit"))
D0 <- summary(y.reduced)$deviance      # Step 1
D1 <- summary(y.full)$deviance
p <- 3
p0 <- 2
n <- nrow(p13.5)
Fstar <- (D0-D1)*(n-p+1) / (D1*(p-p0))

```

Next, we use the `simulate()` function to carry out the bootstrap resampling procedure $R = 999$ times:

```

R <- 999
Fstarstars <- numeric(R) # this will hold the bootstrap test statistics
p13.5star <- p13.5 # this will hold the resampled data
for (i in 1:R) {
  p13.5star$y <- simulate(y.reduced)$sim_1 # Step 2
  ystar.reduced <- glm(y ~ x2, data = p13.5star, family = binomial(link = "logit"))
  ystar.full <- glm(y ~ x1 + x2, data = p13.5star, family = binomial(link = "logit"))
  D0star <- summary(ystar.reduced)$deviance # Step 3
  D1star <- summary(ystar.full)$deviance
  Fstarstar <- (D0star-D1star)*(n-p+1) / (D1star*(p-p0))
  Fstarstars[i] <- Fstarstar
}
pvalue <- mean(Fstarstars > Fstar) # Step 4
pvalue

```

[1] 0.2932933

The computed p-value is large, indicating that the income variable does not seem to be needed in this model.

If you run the above code, you will likely see warning messages about convergence failure in some of the `glm()` calls. This is due to some of the resampled data sets not having enough 0's or enough 1's and is a reflection of the relatively small sample size. To reassure oneself about the quality of the produced p-value, one could double the size of R and re-run the simulation in order to ensure that a large number of resamples are being used.

8.3.4 Inference on the regression coefficients

As in the case of normal linear regression, standard errors can be calculated for the parameter estimates. These are given in the `glm` output. The diagonal elements of the Fisher information matrix are approximations to the coefficient variances. The Fisher information matrix is the inverse of the second derivative of the log likelihood.

Recall that the second derivative is related to how much curvature is in a function. If the likelihood function is flat near the parameter estimate, then the second derivative will be small, so its inverse will be large leading to a large variance.

If the likelihood function has a fairly sharp peak at the parameter estimate, then the second derivative will be large, so its inverse will be small leading to a small variance.

Another way to calculate standard errors is to use bootstrap resampling. The basic idea is to treat the sample estimates as if they were the population parameters, and to simulate from this “population”. Parameter estimates can be recalculated from the simulated data. Doing this repeatedly gives a set of parameter estimates which should resemble the sampling distribution of the estimator. We can calculate the standard deviation of this set of ‘estimates’ and that will be an approximation to the standard error.

8.4 Residuals

As in normal linear regression, diagnostic procedures are important in assessing the appropriateness of generalized linear models. Unfortunately, residuals, which play an important role in normal regression, are sometimes difficult to interpret in generalized linear models.

A number of different types of residuals have been proposed.

8.4.1 Raw residuals

For a model in which $\mu_j = E[y_j]$, we could define the raw residuals as

$$y_j - \hat{\mu}_j.$$

Here, $\hat{\mu}_j$ is determined from $g^{-1}(\mathbf{X}_j \hat{\beta})$.

8.4.2 Pearson residuals

These residuals are essentially the same as the studentized residuals in normal regression:

$$r_j = \frac{y_j - \hat{\mu}_j}{\sqrt{\text{Var}(\hat{\mu}_j)}}.$$

These will usually have a mean of 0 and a variance of ϕ if the model is correct.

Unfortunately, for many generalized linear models, these residuals will not be symmetric about 0, so patterns are difficult to interpret. They are of very limited use for binary regression.

8.4.3 Deviance residuals

Recall that the deviance is defined as the 2ϕ times the difference of two log likelihoods, one for the saturated model and one for the model of interest:

$$D = 2\phi(\ell_{\text{saturated}} - \ell(\hat{\beta})).$$

Since the log likelihoods are sums consisting of n terms which we could denote as d_j . That is, we can write

$$D = \sum_{j=1}^n d_j.$$

Alternatively, we can think of d_j as

$$d_j = 2\phi(\ell_j(\theta_j) - \ell_j(\mathbf{X}_j \hat{\beta})).$$

Noting that d_j must be positive, the j th deviance residual is defined as

$$\hat{\varepsilon}_j = \sqrt{d_j} \text{sign}(y_j - \hat{\mu}_j).$$

We might expect these residuals to look roughly like normal variables with mean 0 and variance ϕ .

8.4.4 A bootstrap approach to model-checking

The following procedure is often an effective technique for checking on the appropriateness of a generalized model. It has a similar motivation to the deviance residuals; in other words, it is based on the idea that the likelihood could be small or very small, if the peak of the fitted probability distribution does not correspond to where the data are. The approach taken here is that each term in the log likelihood sum should be reasonably large. Any terms that are very small are indicators of observations that are failing to be modeled appropriately.

We illustrate the technique with two simulated data sets containing 50 observations. In the first case, the simulated response is Poisson with $\theta_j = 3 - .2x_j$, for a simulated covariate x . The canonical link function is

assumed, and a Poisson model is fit to the data. Using the fitted model, 10 data sets are simulated and the log likelihood contributions for each of the 50 x values are calculated. Side-by-side boxplots of the resulting log likelihood contributions are shown in Figure 8.1. Also plotted, as large red dots are the log likelihood contributions for the original data.

```
n <- 50
x <- runif(n, min = 0, max = 10)
theta <- 3 - .2*x

lambda <- exp(theta)
y <- rpois(n, lambda = lambda)
y.glm <- glm(y ~ x, family = poisson)
a <- coef(y.glm)[1]
b <- coef(y.glm)[2]
llikelihoodValues <- log(dpois(y, lambda = exp(a + b*x)))

simloglike <- function(x, y.glm, N=10) {
  ll <- matrix(0, ncol=length(x), nrow=N)
  for (j in 1:N) {
    y <- simulate(y.glm)$sim_1
    sim.glm <- glm(y ~ x, family = poisson)
    a <- coef(sim.glm)[1]
    b <- coef(sim.glm)[2]
    loglikelihoodValues <- log(dpois(y, lambda = exp(a + b*x)))
    ll[j, ] <- loglikelihoodValues
  }
  ll <- data.frame(ll)
  #names(ll) <- as.character(x)
  names(ll) <- ""
  ll
}
```

```
simlike <- simloglike(x, y.glm)
par(mar=c(1, 4, .1, .1))
boxplot(simlike, ylim=range(c(llikelihoodValues, range(simlike)) ))
points(llikelihoodValues, pch=16, col=2)
```

The red dots often appear near the top of the plot, which is where we would like them to be, for a well-fit model. The likelihood contributions should be large, and this plot shows that they are almost always so. The rare exceptions - and we should expect a few - are not outside the range of the resampled log likelihood contributions.

A similar simulation was run with an incorrect model. This time, the number of 0's has been inflated by multiplying the correct Poisson value by a random Bernoulli random variable. When multiplied by 1, the value remains unchanged, of course, but when multiplied by 0, the Poisson value may change to 0 from a nonzero value. 5% of the values are affected in this way. The resulting boxplots are pictured in Figure 8.2. This time, some of the red dots are appearing outside the range of the resampled boxplots. The diagnostic has correctly identified a problem with the fitted model.

```
n <- 50
x <- runif(n, min = 0, max = 10)
theta <- 3 - .2*x
lambda <- exp(theta)
y <- (rpois(n, lambda = lambda))*rbinom(n, 1, prob=.95)
```

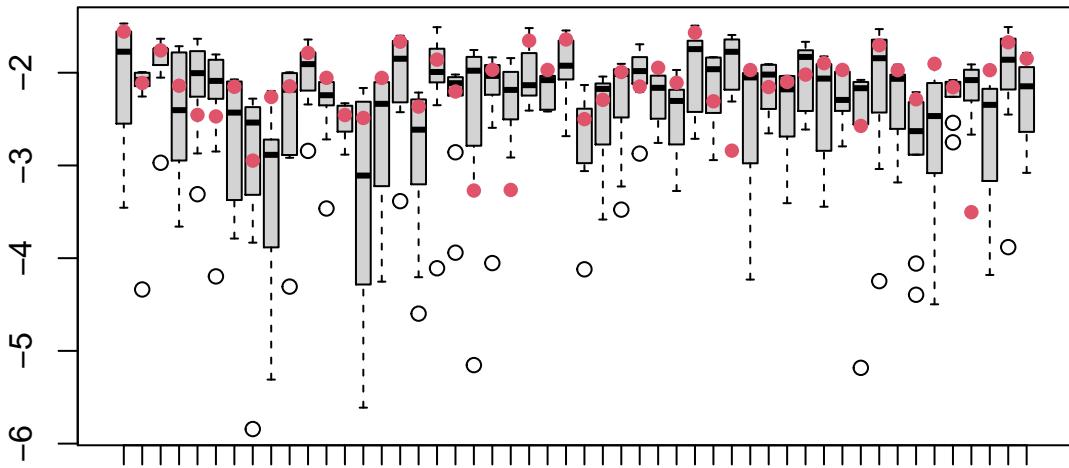


Figure 8.1: Distributions of observed likelihood contributions compared with resampled likelihood contributions. The model is correct in this case.

```
y.glm <- glm(y ~ x, family = poisson)
a <- coef(y.glm)[1]
b <- coef(y.glm)[2]
llikelihoodValues <- log(dpois(y, lambda = exp(a + b*x)))
```

```
simlike <- simloglike(x, y.glm)
par(mar=c(1, 4, .1, .1))
boxplot(simlike, ylim=range(c(llikelihoodValues, range(simlike))), )
points(llikelihoodValues, pch=16, col=2)
```

Example – cigarette butts data

The appropriate way to model the cigarette butt count data (interest would center on how the distribution of cigarette butts changes as a function of distance from the smoking area) is to use Poisson regression. First, plot the data to visualize the relationship, as in Figure 8.3.

```
par(mfrow=c(1, 2), mar=c(4, 4, .1, .1)) # create a 1x2 layout of plots
plot(count ~ distance, data = cigbutts)
plot(log(count+1) ~ distance, data = cigbutts)
```

The left panel of the figure shows a clear rapid decrease at short distances followed by an almost constant pattern (at or near 0) at larger distances. The second panel shows the counts on the log scale (note that 1 was added to the counts, so that the log is always defined). On the log scale, the relationship is almost linear.

It might be tempting to fit a line to the transformed data, using least-squares:

```
cig.lm <- lm(log(count+1) ~ distance, data = cigbutts)
cig.lm

## 
## Call:
## lm(formula = log(count + 1) ~ distance, data = cigbutts)
##
```

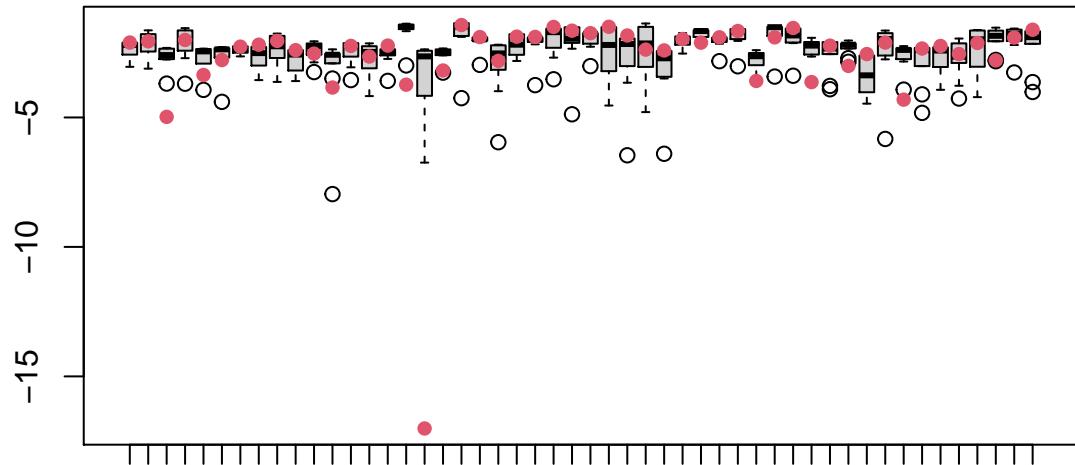


Figure 8.2: Distributions of observed likelihood contributions compared with resampled likelihood contributions. The model is incorrect in this case.

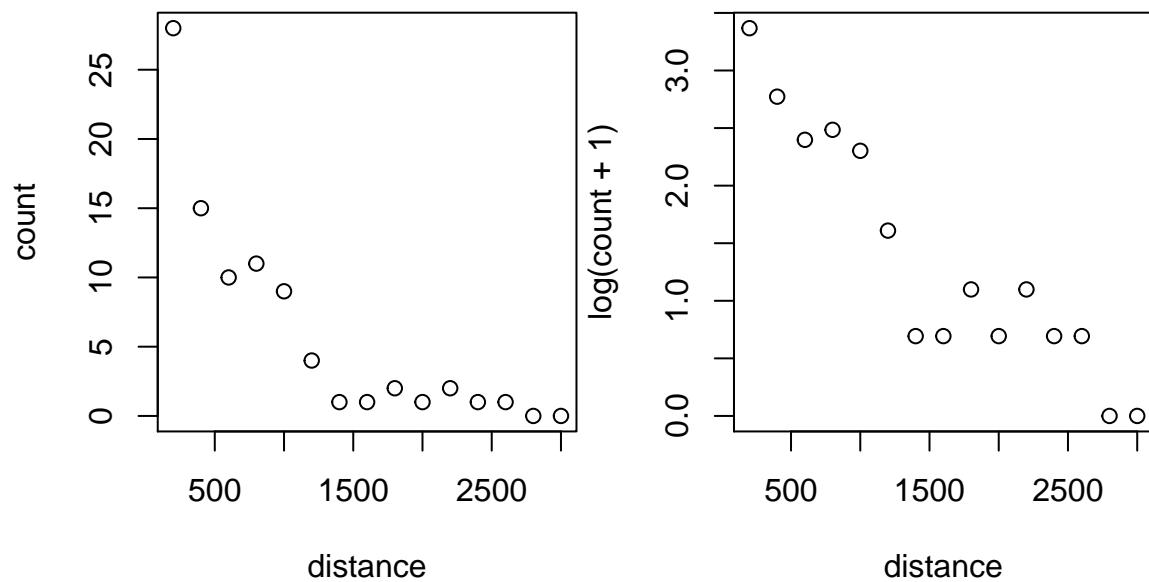


Figure 8.3: Numbers of cigarette butts observed at various distances from a designated smoking area.

```
## Coefficients:
## (Intercept)      distance
##     3.114218    -0.001088
```

but because of the nature of the data (that is, it is in the form of counts), a Poisson distribution is more appropriate as a model than the normal distribution, which would have justified the use of least-squares. The more appropriate approach then is to model the counts directly using the `glm` function with a distribution for the counts:

```
cig.glm <- glm(count ~ distance, data = cigbutts, family = poisson)
cig.glm

##
## Call: glm(formula = count ~ distance, family = poisson, data = cigbutts)
##
## Coefficients:
## (Intercept)      distance
##     3.553514    -0.001696
##
## Degrees of Freedom: 14 Total (i.e. Null); 13 Residual
## Null Deviance: 122.5
## Residual Deviance: 9.003 AIC: 53.69
```

By default, the log of the expected count is related to a linear function of distance in this model. We can compare the least-squares and generalized linear model fits on the original data as in Figure 8.4.

```
par(mar=c(4, 4, .1, .1))
plot(count ~ distance, data = cigbutts)
a0 <- coef(cig.lm)[1]
a1 <- coef(cig.lm)[2]
curve(exp(a0+a1*x)-1, from = 0, to = 3000, add = TRUE)
b0 <- coef(cig.glm)[1]
b1 <- coef(cig.glm)[2]
curve(exp(b0+b1*x), from = 0, to = 3000, add = TRUE, lty=2)
```

This example highlights the critical importance of using the correct technique when analyzing this kind of data. The Poisson generalized linear model is substantially more accurate, because it correctly models all aspects of the data distribution.

Model-checking using boxplots of log likelihood contributions from each observation for observed and from simulated data:

```
cig.glm <- glm(formula = count ~ distance, data = cigbutts, family = poisson)
attach(cigbutts)
llikelihoodValues <- log(dpois(count, lambda = fitted(cig.glm)))
detach(cigbutts)

simlike <- simloglike(cigbutts$distance, cig.glm)
par(mar=c(1, 4, .1, .1))
boxplot(simlike, ylim=range(c(llikelihoodValues, range(simlike)) ))
points(llikelihoodValues, pch=16, col=2)
```

Figure 8.5 shows the boxplots of the log likelihood contributions compared with what would be obtained from simulations of the correct model. The red dots again correspond to the original data, and only one of the twenty falls outside of the range of its corresponding boxplot, but it easily falls in the range of all of the simulated log likelihood contributions. There does not seem to be a problem with the way this model is fitting the data.

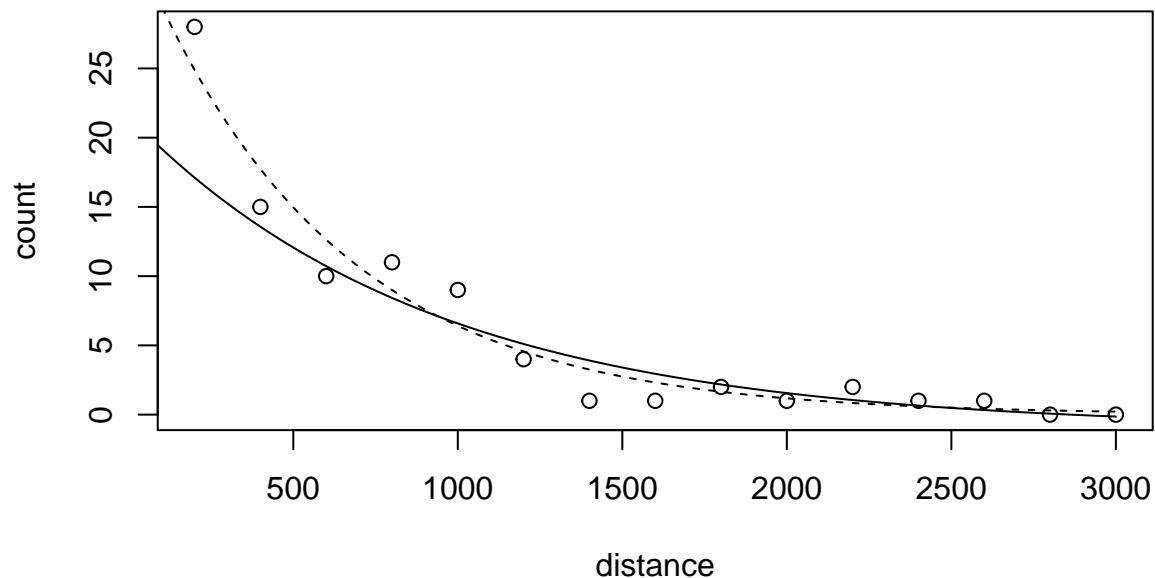


Figure 8.4: Comparison of linear and generalized linear model fits to the cigarette butts data set. The solid curve is from the least-squares fit and the dashed curve is from the Poisson GLM. Which one do you think is more accurate?

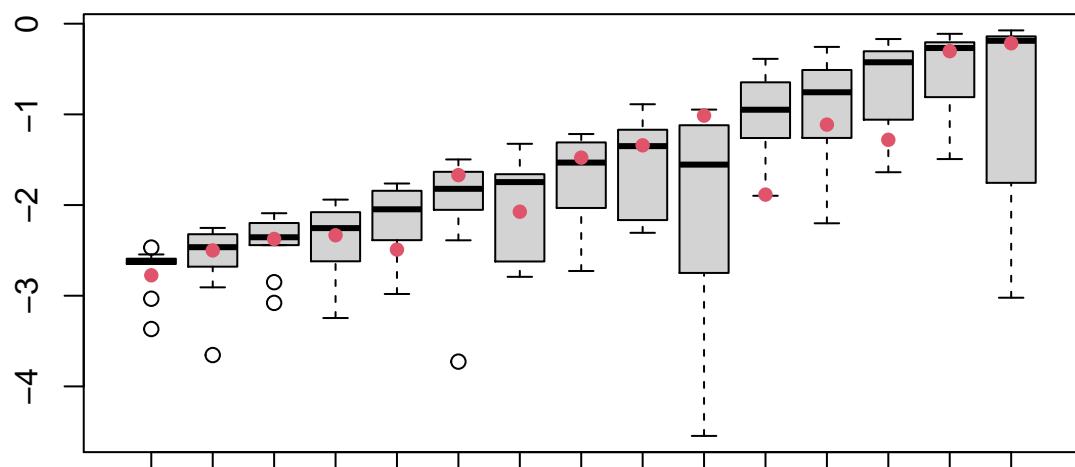


Figure 8.5: Distributions of observed likelihood contributions compared with resampled likelihood contributions for the cigarette butts data.

8.5 Overdispersion and quasilielihood

The binomial and Poisson assumptions can be very restrictive: the mean must equal the variance in the Poisson case, while the mean and variance of the Bernoulli distribution are both related through p .

Often, the variance will be much larger than the mean in count data; this is called overdispersion. Quasi-likelihood methods have been developed to handle this problem. In `glm()`, use `family=quasipoisson`.

Until now, the dispersion parameter ϕ has been taken to be 1. We have been assuming that there is no clustering in the data which would have possibly led to overdispersion: the case where the variance exceeds what would be expected under a binomial model or Poisson model. If there is a belief that clustering is occurring (not likely in this example), the `quasibinomial` family or `quasipoisson` family should be used instead.

Large sample properties of estimators such as consistency and asymptotic normality continue to hold. Hypothesis testing results using the F distribution remain approximately true. Test results will be more conservative when using `quasipoisson` in place of the `poisson` family.

8.5.1 Example – cigarette butts data

Although the data don't indicate a need for the use of quasilielihood in the case of the cigarette butts, we illustrate the use of the family here:

```
cig.quasi <- glm(formula = count ~ distance, data = cigbutts, family = quasipoisson)
summary(cig.quasi)

##
## Call:
## glm(formula = count ~ distance, family = quasipoisson, data = cigbutts)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.4653   -0.7173   -0.1668    0.6281    1.0751
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.5535143  0.1403755  25.31 1.91e-12
## distance    -0.0016956  0.0001625 -10.43 1.10e-07
##
## (Dispersion parameter for quasipoisson family taken to be 0.6545345)
##
## Null deviance: 122.4728 on 14 degrees of freedom
## Residual deviance:  9.0035 on 13 degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

If you compare the standard error estimates with those obtained for the `poisson` family, you will find these ones to be slightly larger.

8.5.2 A somewhat more complicated example

The data that we conclude this chapter with were collected at the University of Manitoba: precancerous lesions in the colons of 66 rats:

```
source("lesions.R")
```

```
summary(lesions)

## # T           INJ        SECT       RAT      ACF.Total     ACF.total.mult
## 1:132   Min.    :1.000  1:66    1     :54  Min.    : 0.00  Min.    : 0.0
## 2:144   1st Qu.:1.000  2:66    2     :54  1st Qu.: 14.00 1st Qu.: 31.0
## 3:120   Median  :2.000  3:66    3     :54  Median  : 28.00  Median  : 65.5
##          Mean    :2.333  4:66    4     :54  Mean    : 38.23  Mean    :105.4
##          3rd Qu.:4.000  5:66    5     :54  3rd Qu.: 53.25 3rd Qu.:140.8
##          Max.    :4.000  6:66    6     :54  Max.    :213.00  Max.    :719.0
##                               (Other):72
## # id
## # Min.  : 1.0
## # 1st Qu.:17.0
## # Median :33.5
## # Mean   :33.5
## # 3rd Qu.:50.0
## # Max.   :66.0
## #
```

The rats had been subjected to differing numbers of injections of carcinogen (INJ: 1, 2 or 4 doses) and sacrificed at 6, 12 or 18 weeks (T). Counts of lesions (called ACF) were taken on the colon walls, divided up into 6 sections (SECT) beginning at the proximal end and ending at the distal end. Interest centers on how the numbers and area of the lesions grow with time and number of doses, and in the different regions of the colon.

Counting lesions in some of the rats

We will begin with some visualizations of the data in order to gain some insights before beginning the formal modelling process. We start by looking at data from just a sample of 9 of the rats, each one corresponding to a unique injection-time factor level combination. Figure 8.6 shows how the total numbers of lesions in the rats varies with number of injections, by time, and section. The total number increases with time but more rapidly as the number of injections increases. The pattern of increase is not always the same among the colon sections.

```
barchart(ACF.Total ~ T|INJ, groups=SECT, data =
  subset(lesions, RAT==4), stack=TRUE, auto.key=TRUE,
  ylab="lesion counts")
```

Section 5 has a lot of lesions. And it seems that the number of lesions does not change from injection dose level 2 to dose level 4.

Measuring the area taken up by lesions in those rats

The total ACF multiplicity is related to the total area occupied by the lesions in the colons of the rats. Figure 8.7 displays a bar plot of the total ACF multiplicities in each of 6 sections of the sample of 9 rats. The same general patterns of increase that were seen in the ACF counts appear here as well. In addition, Section 3 develops large lesions. And it seems that the size of the lesions increases from dose level 2 to dose level 4.

```
barchart(ACF.total.mult ~ T|INJ, groups=SECT, data =
  subset(lesions, RAT==4), stack=TRUE, auto.key=TRUE,
  ylab="lesion counts")
```

Visualizing the numbers of lesions in all of the rats

Figures 8.8 shows how the numbers of ACF lesions vary by section within each of the 66 rats, nested within each of the 9 injection-time factor level combinations. Figure 8.9 is set up the same but for the ACF multiplicities.

```
barchart(ACF.Total ~ RAT|paste(T, INJ), groups=SECT,
data = lesions, stack=FALSE, ylab="lesion counts",
xlab = "rat")
```

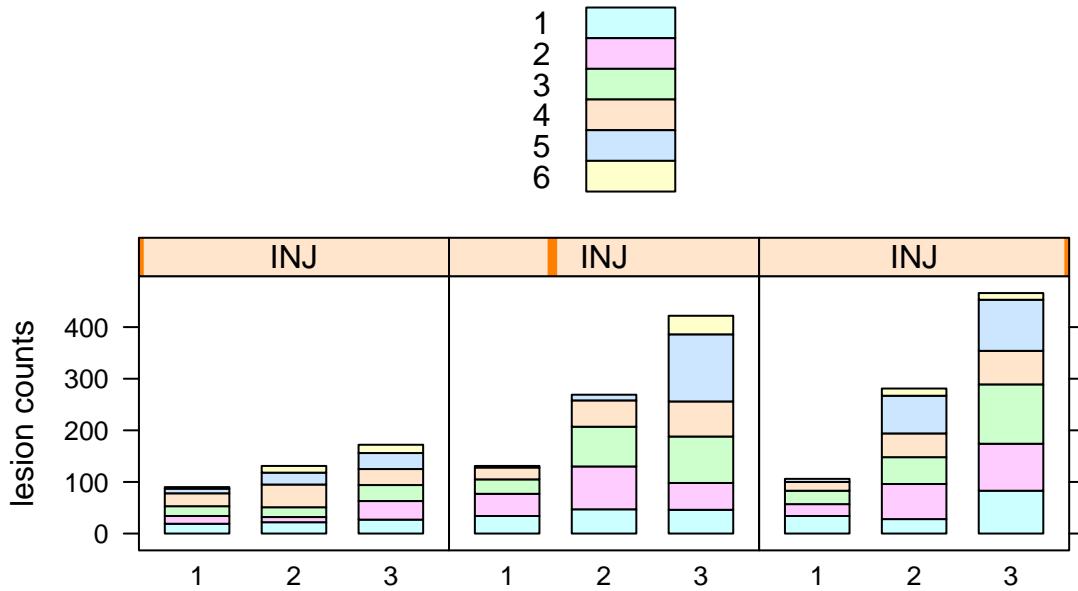


Figure 8.6: Bar plots of total ACF counts in 6 sections of colons (colour-coded according to the legend) of 66 mice with varying numbers of carcinogen injections, varying incubation times (1, 2, 3).

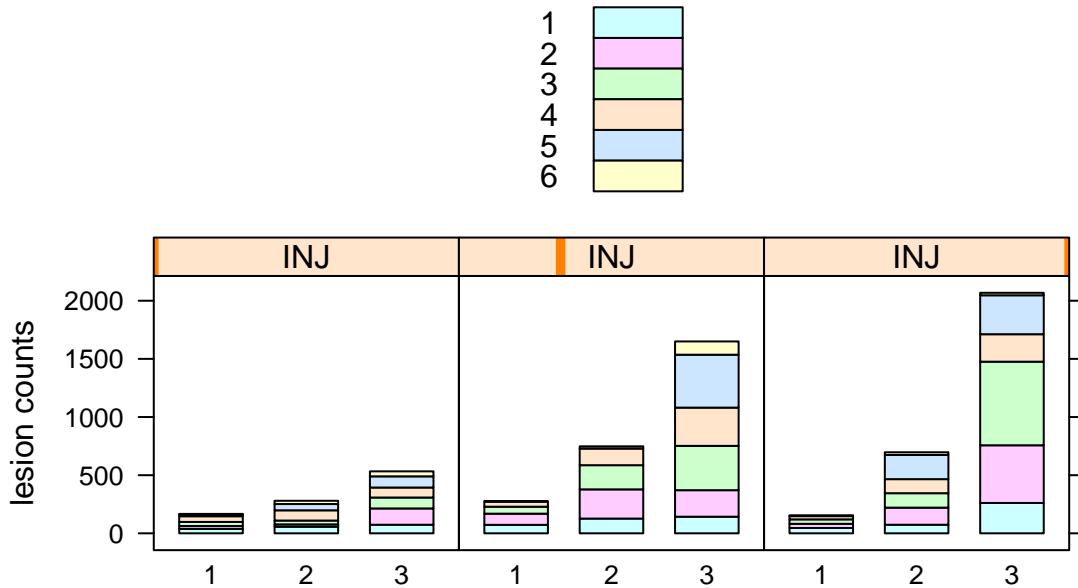


Figure 8.7: Bar plots of total ACF multiplicities in 6 sections of colons (colour-coded according to the legend) of 66 mice with varying numbers of carcinogen injections, varying incubation times (1, 2, 3).

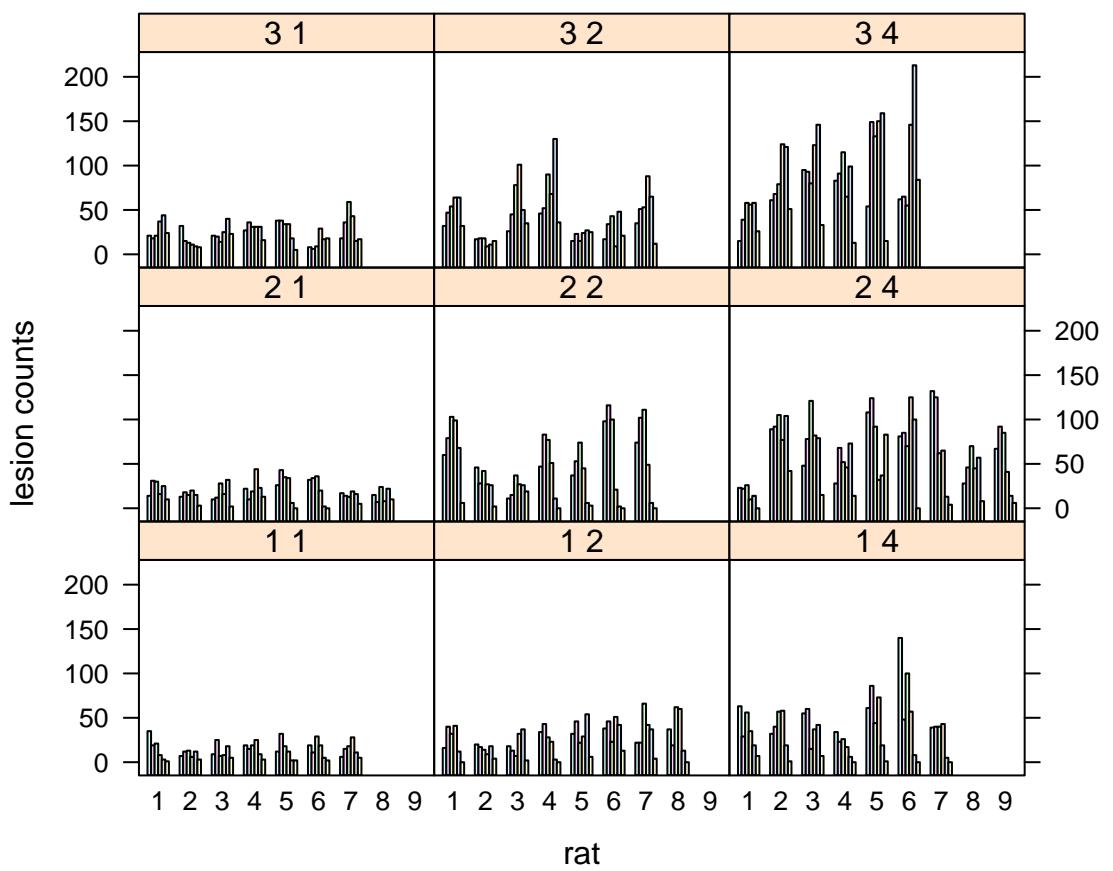


Figure 8.8: Bar plots of the total ACF counts for all 66 rats where panel (i, j) contains the data for rats subjected to j injections for $6i$ weeks. Within each panel, individual bar plots of the total ACF counts for the six colon sections are displayed for each of varying numbers of rats (7, 8, or 9).

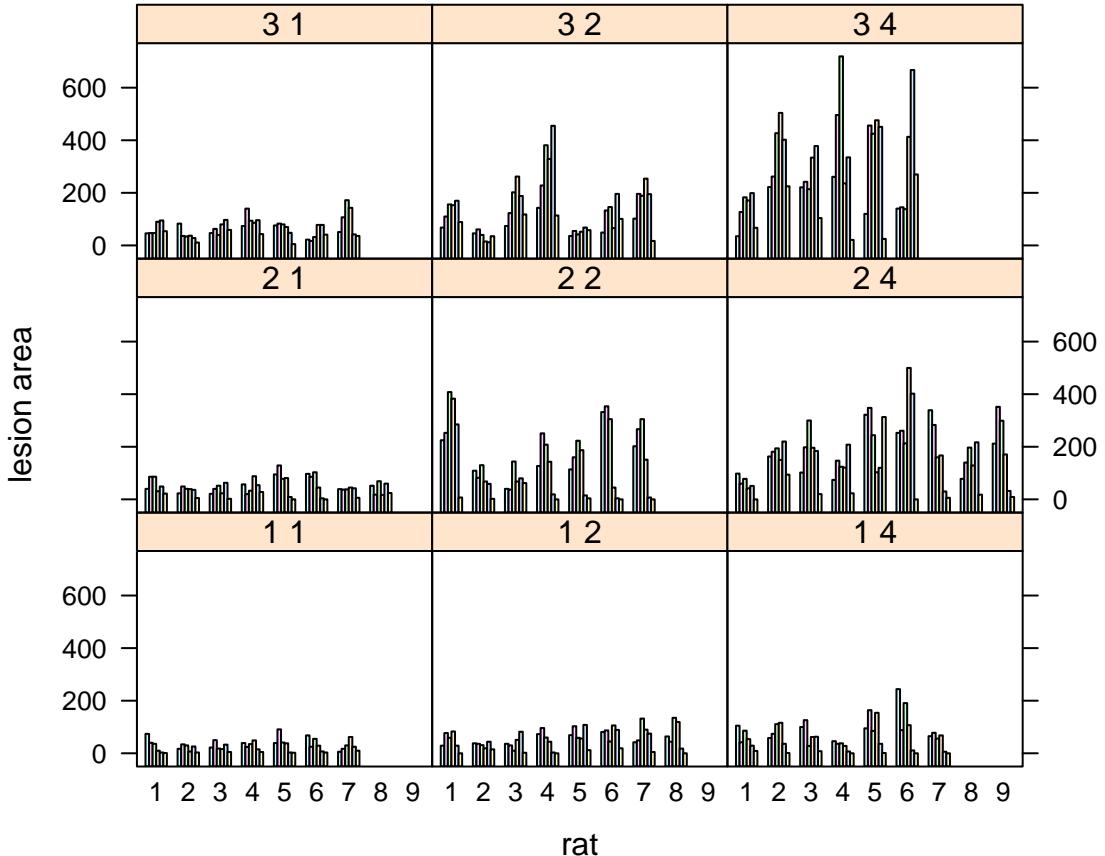


Figure 8.9: Bar plots of the total ACF multiplicities for each of the 66 rats where panel (i, j) contains the data for rats subjected to j injections for $6i$ weeks. Within each panel, individual bar plots of the total ACF multiplicities for the six colon sections are displayed for each of varying numbers of rats (7, 8, or 9).

Viewing the lesion area in all rats

```
barchart(ACF.total.mult ~ RAT|paste(T, INJ), groups=SECT,
data = lesions, stack=FALSE, ylab="lesion area",
xlab="rat")
```

In Figure 8.8, we see that there is a tendency for more ACF to concentrate in the middle sections of the colon, and this tendency increases as the number of injections and incubation times increase. We also see that increasing number of injections tendencies to have more of an effect on total numbers than increasing incubation time, though both factors appear to play a role. More-or-less the same observations can be stated for the multiplicities, except that incubation time seems to have just as much of an effect on the development of the ACF multiplicities as number of injections.

Modelling the ACF data

We start with modelling the total number of ACF as a function of time and injection, for each rat. (That is, we will sum over all colon sections in each rat.)

```
ACF.All <- aggregate(ACF.Total ~ id + INJ + T, FUN=sum, data = lesions)
```

A Poisson GLM is a reasonable first guess for such data. We include an interaction term for injection and time effects:

```

ACF.poisson <- glm(ACF.Total ~ INJ * T, data = ACF.All,
  family=poisson)
summary(ACF.poisson)

##
## Call:
## glm(formula = ACF.Total ~ INJ * T, family = poisson, data = ACF.All)
##
## Deviance Residuals:
##       Min        1Q     Median        3Q       Max
## -17.2309   -4.2434   -0.9751    4.2298   13.0646
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 4.30032   0.04199 102.423 < 2e-16
## INJ         0.28459   0.01386  20.530 < 2e-16
## T2          0.39760   0.05320   7.474 7.80e-14
## T3          0.34349   0.05344   6.427 1.30e-10
## INJ:T2      0.02167   0.01724   1.257   0.209
## INJ:T3      0.11758   0.01741   6.754 1.44e-11
##
## (Dispersion parameter for poisson family taken to be 1)
##
## Null deviance: 5894.4 on 65 degrees of freedom
## Residual deviance: 2049.2 on 60 degrees of freedom
## AIC: 2528
##
## Number of Fisher Scoring iterations: 4

```

Note that the residual deviance is much larger than its degrees of freedom. This is a strong hint that the model is very inaccurate. Thus, we cannot trust the p-values for the effects at all. The simplest alternative is to use the quasipoisson family instead of poisson. This relaxes the Poisson assumption and allows us to handle overdispersion. The model is no longer interpretable in the same way, however, since it is not based on a real probability model.

```

ACF.qp <- glm(ACF.Total ~ INJ * T, data = ACF.All, family=quasipoisson)
summary(ACF.qp)

##
## Call:
## glm(formula = ACF.Total ~ INJ * T, family = quasipoisson, data = ACF.All)
##
## Deviance Residuals:
##       Min        1Q     Median        3Q       Max
## -17.2309   -4.2434   -0.9751    4.2298   13.0646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.30032   0.24178 17.786 < 2e-16
## INJ         0.28459   0.07983  3.565 0.000721
## T2          0.39760   0.30635  1.298 0.199306
## T3          0.34349   0.30775  1.116 0.268813
## INJ:T2      0.02167   0.09928  0.218 0.827981
## INJ:T3      0.11758   0.10026  1.173 0.245511
##
## (Dispersion parameter for quasipoisson family taken to be 33.1603)
##
## Null deviance: 5894.4 on 65 degrees of freedom
## 
```

```
## Residual deviance: 2049.2 on 60 degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

One thing that this quasilielihood approach tells us is that the interaction effect is not significant. Fit the model again, without the interaction term. Note the effect on the INJ and T coefficients. The standard errors on the coefficients are smaller than with the interaction term. Because of the multiple testing (2 tests), we should demand that the p-values be less than about .025 in order to say that we have strong evidence of effects. The required code to omit the interaction term is as follows.

```
ACF.qp <- glm(ACF.Total ~ INJ + T, data = ACF.All, family=quasipoisson)
summary(ACF.qp)

##
## Call:
## glm(formula = ACF.Total ~ INJ + T, family = quasipoisson, data = ACF.All)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -17.7800  -4.1140  -0.6099   3.6721  13.5608
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.14947   0.14540 28.538 < 2e-16
## INJ         0.33844   0.03733  9.066 5.72e-13
## T2          0.45345   0.12422  3.651 0.000539
## T3          0.67487   0.12467  5.413 1.06e-06
##
## (Dispersion parameter for quasipoisson family taken to be 32.98798)
##
## Null deviance: 5894.4 on 65 degrees of freedom
## Residual deviance: 2110.9 on 62 degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

An alternative to quasilielihood here is the negative binomial model. This is a model for counts which allows for overdispersion. In fact, the distribution arises from a form of clustering. The MASS package has a function that fits generalized linear models, based on the negative binomial distribution:

```
library(MASS)
ACF.nb <- glm.nb(ACF.Total ~ INJ + T, data = ACF.All)
summary(ACF.nb)

##
## Call:
## glm.nb(formula = ACF.Total ~ INJ + T, data = ACF.All, init.theta = 7.19002744,
##        link = log)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -3.0685  -0.7880  -0.0612   0.4765   2.2565
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 4.12938   0.11996 34.424 < 2e-16
## INJ         0.35607   0.03751  9.493 < 2e-16
```

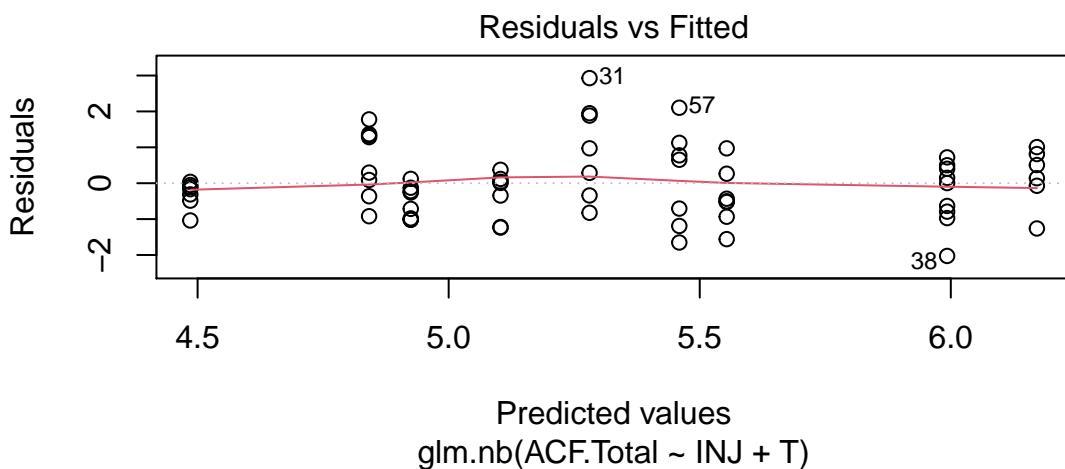
```

## T2          0.43902    0.11259   3.899 9.65e-05
## T3          0.61777    0.11766   5.251 1.52e-07
##
## (Dispersion parameter for Negative Binomial(7.19) family taken to be 1)
##
## Null deviance: 184.98 on 65 degrees of freedom
## Residual deviance: 67.16 on 62 degrees of freedom
## AIC: 763.56
##
## Number of Fisher Scoring iterations: 1
##
##
##           Theta:  7.19
##           Std. Err.: 1.27
##
## 2 x log-likelihood: -753.564

```

Look at the residual deviance value. The residual deviance is in line with the residual degrees of freedom, supporting the model – of course, we cannot really conclude much about the model until we examine graphical diagnostics. If the model is appropriate, we can now conclude quite safely that the time and injection effects are significant. We can inspect the residual plot produced with the following code.

```
plot(ACF.nb, which=1)
```



No real trend in the mean residual level is apparent, but there is a pattern to the variation in the residuals which would have to be examined more closely, say, by simulating from this model. Observation 38 is being flagged as an outlier.

Analyzing the section effects requires more sophisticated machinery

If we want to study an effect due to colon section, we need to use a mixed-effects model.² The *lme4* package has a function that allows us to do this with a negative binomial distribution. We need to include a random coefficient for section, since we have repeated measurements for the different sections, within each rat (identified by *id*).

```
library(lme4)
```

In order to reduce the number of parameters to be estimated, we convert *T* and *SECT* to numeric variables. This is done in the first lines of codes below.

²You may want to omit this section until you have read Chapter 10.

```

lesions$T <- as.numeric(lesions$T)
lesions$SECT <- as.numeric(lesions$SECT)
ACF.nb <- glmer.nb(ACF.Total ~ INJ + T + SECT + (SECT|id), data = lesions)
summary(ACF.nb)

```

If you run the above code, you will encounter a failure to converge. We can adjust for this by centering the variables. This is not foolproof, but it often works, and it does so in the implementation that follows

```

ACF.nb <- glmer.nb(ACF.Total ~ I(INJ-2) + I(T-2) + I(SECT-3) + (I(SECT-3)|id),
  data = lesions)
summary(ACF.nb)

## Generalized linear mixed model fit by maximum likelihood (Laplace
## Approximation) [glmerMod]
## Family: Negative Binomial(2.2982)  ( log )
## Formula: ACF.Total ~ I(INJ - 2) + I(T - 2) + I(SECT - 3) + (I(SECT - 3) | 
##     id)
## Data: lesions
##
##      AIC      BIC      logLik deviance df.resid
## 3508.6   3540.5   -1746.3    3492.6     388
##
## Scaled residuals:
##      Min      1Q Median      3Q      Max
## -1.4752 -0.6469 -0.1754  0.5814  2.7614
##
## Random effects:
## Groups Name        Variance Std.Dev. Corr
## id     (Intercept) 0.06267  0.2503
##          I(SECT - 3) 0.01574  0.1255  0.00
## Number of obs: 396, groups: id, 66
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 3.39030   0.05203 65.156 < 2e-16
## I(INJ - 2)  0.34884   0.03822  9.127 < 2e-16
## I(T - 2)    0.33506   0.06918  4.844 1.28e-06
## I(SECT - 3) -0.19570   0.03011 -6.499 8.07e-11
##
## Correlation of Fixed Effects:
##           (Intr) I(INJ-2) I(T-2)
## I(INJ - 2) -0.258
## I(T - 2)    0.041 -0.020
## I(SECT - 3) -0.155 -0.006  0.091

```

This model accounts for the dependency in the ACF counts within each rat colon, allowing us a way to more accurately assess the standard errors for injection and time effects, as well as section effects. We see clear evidence of increase in mean ACF counts for both injection and time, while there is a significant decrease in mean ACF count as one proceeds along the colon from the 1st section to the 6th section. The standard deviation of the section effect is 0.12 and the model intercept standard deviation is 0.25.

The ACF multiplicities can be analyzed in the same way, using code such as what follows. The details are left to the reader to pursue.

```

ACF.nb <- glmer.nb(ACF.total.mult ~ I(INJ-2) + I(T-2) + I(SECT-3) + (I(SECT-3)|id),
  data = ACF.All)
summary(ACF.nb)

```

A summary of the results

We have found evidence that the injected carcinogen is associated with the development and growth of the aberrant crypt foci (ACF). The number of ACF produced depends upon location within the colon (i.e. whether close to the anus or whether close to the small intestine). The number of ACF increases with time, but not so much when there has only been a single injection of the carcinogen.

8.6 Exercises

1. Refer to the missile success-failure data in `p13.1` of the *MPV* package. (See Section 8.3.1.) Estimate the logit of the probability of missile success at a speed of 400 knots. Calculate the probability of missile success. (For this, you can either use the logistic formula, or the `predict()` function in R, using the correct `type`.)
2. The `p13.2` data frame in the *MPV* package has 20 observations on home ownership as it relates to family income. Fit a logistic regression model to the data and use the output to
 - (a) identify the logit of the probability of home ownership as a linear function of family income.
 - (b) determine if the logistic model is reasonable.
 - (c) estimate the probability that a family with an income of \$40000 owns their home.
3. The data in `HairEyeColor[, , 2]` concern hair and eye color for a sample of females. Conduct a test to see if hair and eye color are associated.
4. Suppose p is an element of the interval $(0, 1)$ and define the *logit* function as

$$g(p) = \log\left(\frac{p}{1-p}\right).$$

- (a) What is the range of the function $g(p)$.
- (b) Using algebra, show that the inverse function of $g(p)$ is given by the *logistic* function

$$\ell(g) = \frac{e^g}{1 + e^g}$$

and determine the range of this new function.

- (c) Define $p(x) = \ell(\beta_0 + \beta_1 x)$, for constants β_0 and β_1 , and for $x \in (-5, 5)$. Plot the graph of this function for the following cases:
 - i. $\beta_0 = 0, \beta_1 = 1$
 - ii. $\beta_0 = 0, \beta_1 = 5$
 - iii. $\beta_0 = 0, \beta_1 = -3$
 - iv. $\beta_0 = 3, \beta_1 = -3$
 - v. $\beta_0 = -3, \beta_1 = -3$
 - vi. $\beta_0 = 3, \beta_1 = 1$

Use the following procedure to do the plotting:

- i. Assign numeric values to `beta0` and `beta1`.
 - ii. Construct a function `g` which takes `x` as an argument and returns the value `beta0 + beta1 * x`.
 - iii. Construct a function `l` which takes `g` as an argument and returns the value `exp(g) / (1+exp(g))`.
 - iv. Construct a function `p` which takes `x` as an argument and returns the value `l(g(x))`.
 - v. Now, apply the `curve()` function to the function `p()` on the appropriate interval, re-assigning new values to `beta0` and `beta1` as needed for each plot. Use `par(mfrow=c(3, 2))` to produce a 3×2 layout of plots on the page.
5. Suppose that you are confronted with a set of data consisting of three columns, headed by an `x`, a `y` and an `n`, such as

	x	y	n
1	80	4	6
2	81	7	10
3	82	2	5
4	83	4	10
5	84	1	8

- (a) Suppose the entries of the y column are realizations of independent binomial random variables, with parameters which depend on the corresponding entries of the x and n columns. In other words, y_i is from a binomial distribution with parameters $p(x_i)$ (defined in the previous question) and n_i . For example, the first entry of the above table was from a binomial distribution having $n = 6$ and $p = p(80)$.
- Find the mean and variance of y_i in terms of β_0 , β_1 and x_i .
 - Which of the regression assumptions (i.e. independence, normality, constant variance) do not hold for this data set.
 - Use the binomial model to write out the log likelihood for such a data set. Maximizing this with respect to β_0 and β_1 gives estimators for β_0 and β_1 . (Don't worry. You do not have to try to maximize this likelihood yourself. It must be done numerically, and R can do this for you.)
[Hint: write down the likelihood first as a function of p . Then replace p by $p(x_i)$ and then use the functional form of $p(x)$ to finish the job.]

- (b) Use the following code to read in the above data frame:

```
xyn <- data.frame(x=c(80, 81, 82, 83, 84), y=c(4, 7, 2, 4, 1), n=c(6, 10, 5, 10, 8))
```

- Use the `glm()` function in R to estimate the parameters of the model

$$g(p) = \beta_0 + \beta_1 x,$$

maximizing the likelihood for this data set:

```
xyn.glm <- glm(I(y/n) ~ x, family=binomial, weights = n, data=xyn)
```

Use the output from the `summary()` function to write down the fitted model. Are the coefficients significantly different from 0?

- Create a scatterplot of the observed proportions, y/n , against x , and overlay it with a curve which passes through the predicted values coming from the fitted model:

```
lines(predict(xyn.glm) ~ x, data=xyn)
```

6. Referring to the previous question, suppose the values in the n column of the data frame came from a Poisson distribution where the mean λ is actually a function of x , that is,

$$E[n_i] = \lambda(x_i)$$

for some function $\lambda(x)$. Suppose also that

$$\log(\lambda(x)) = \beta_0 + \beta_1 x.$$

- Write down the log likelihood as a function of β_0 and β_1 , for given data x and n . Again, first write it out as a function of λ , and then replace occurrences of λ with $\lambda(x_i)$ and then replace this with an appropriate function of $\beta_0 + \beta_1 x_i$.

- (b) The likelihood can be maximized numerically using

```
xyn.glm2 <- glm(n ~ x, family=poisson, data=xyn)
```

Use the `summary()` function to write out the fitted model, and note whether the coefficients differ from 0.

- Use the `predict()` and `lines()` functions to produce a scatterplot of n versus x with a curve passing through the fitted values overlaid.

7. Write out the weighted nonlinear least-squares objective function that would be minimized in each of the following situations where it is assumed that the mean μ of a given response variable y is to be related to a single covariate x , according to a model of the form $g(\mu_j) = \beta_0 + \beta_1 x_j$.

- Bernoulli response with a log link function.
- Exponential response with a square root link function.
- Poisson response with an identity link function.
- Poisson response with a square root link function.

8. Suppose w is related to p through the complementary log-log function. That is

$$w = g(p) = \log(-\log(1-p)).$$

Find the inverse function. That is, express p as a function of w .

9. Test out the bootstrap diagnostic procedure of Section 8.4.4 on the following “incorrect” models:

(a) `y <- (rpois(n, lambda = lambda))^2`
(b) `y <- (rpois(n, lambda = lambda))*rbinom(n, 1, prob=.8)`
(c) `y <- (rpois(n, lambda = lambda))*rbinom(n, 1, prob=.99)`

Does the procedure correctly identify problems with the fitted model in these cases?

10. Further analyze the data in `lesions` by modelling `ACF.total.mult` as it relates to incubation time and number of carcinogen injections. Try the quasilielihood approach and the negative binomial approaches that were applied in the text to the `ACF.Total` responses. In particular, check for an interaction between incubation time and number of injections. There was not such an interaction in the case of the total ACF counts, but is there such an interaction when we consider ACF multiplicities?

9

Splines and Spline Regression

Given data of the form $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, we seek an estimate of the regression function $g(x)$ satisfying the model

$$y = g(x) + \varepsilon$$

where the noise term satisfies the usual conditions assumed for simple linear regression.

The discovery that piecewise polynomials or splines could be used in place of polynomials occurred in the early twentieth century. Splines have since become one of the most popular ways of approximating nonlinear functions.

In this chapter, we will describe some of the basic properties of splines, describing two bases. We will also discuss how to estimate coefficients of a spline using least-squares regression. We close this chapter with a discussion of smoothing splines, penalized splines and additive models.

9.1 Basic properties of splines

Splines are essentially piecewise polynomials. They are usually taken to be as smooth as possible, though this is not an absolute requirement. In this subsection, we will describe how splines can be viewed as linear combinations of truncated power functions and appropriate monomials. We will then describe the B-spline basis which is a more convenient basis for computing splines.

9.1.1 Truncated power functions

Let t be any real number. Then we can define a p th degree truncated power function as

$$(x - t)_+^p = (x - t)^p I_{\{x > t\}}(x)$$

As a function of x , this function takes on the value 0 to the left of t , and it takes on the value $(x - t)^p$ to the right of t . The number t is called a *knot*.

A function for computing truncated powers in R is as follows.

```
tr.pwr <- function (x, knot, degree=3) {
  (x > knot) * (x - knot)^degree
}
```

Figure 9.1 shows two examples of truncated power functions. The graph on the left is of the linear truncated power function with a knot at 0.25

$$T_1(x) = (x - 0.25)_+.$$

The graph on the right is of the cubic truncated power function with a knot at 0.4

$$T_2(x) = (x - 0.4)_+^3.$$

Notice that both graphs are continuous at the respective knots, but the cubic function is smoother. Code for producing the graphs is:

```
curve(tr.pwr(x, knot=0.25, degree=1), ylab=expression(T[1](x)))
curve(tr.pwr(x, knot=0.4), ylab=expression(T[2](x)))
```

The truncated power function is a basic example of a spline. The space of spline functions of a given degree having a specified set of knots is spanned by a set of truncated power functions. In other words, the truncated power functions can play the role of basis functions for the space of splines.

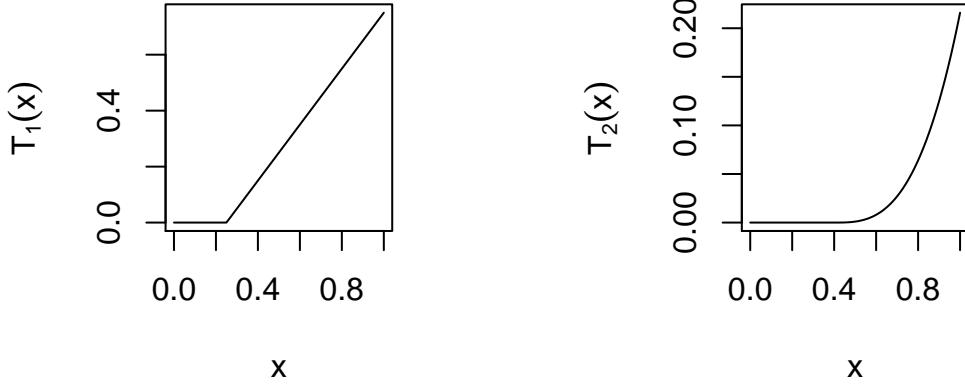


Figure 9.1: Truncated linear and truncated cubic functions with knots at 0.25 and 0.4, respectively.

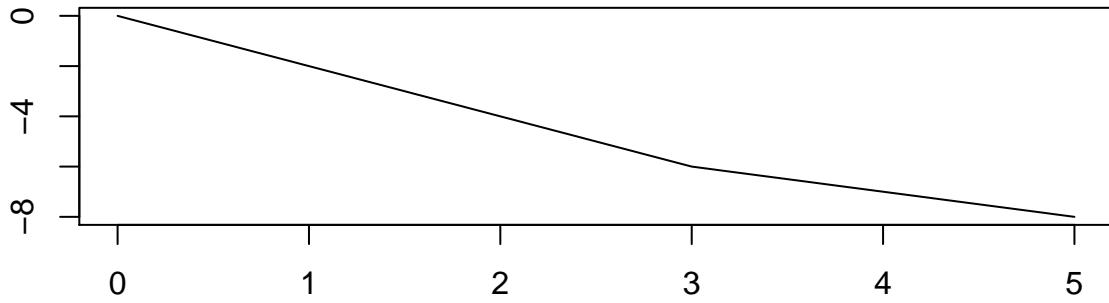


Figure 9.2: An example of a linear spline with one knot, at $x = 3$.

9.1.2 Spline functions

Linear combinations of polynomials with truncated power functions of the same degree are piecewise polynomials, hence, splines. Here are a few examples:

$$s(x) = -2x + (x - 3)_+$$

Figure 9.2 shows the graph of this linear spline function with a single knot. Notice that the slope changes at $x = 3$ from -2 to -1 , because of the effect of the truncated function.

$$s(x) = 1 - 1.2x + x^2 - .5x^3 + (x - 2)_+^3 + (x - 5)_+^3$$

Figure 9.3 shows the graph of this cubic spline function with two knots. Notice how this function is clearly not cubic, but it is very smooth. In fact, it has continuous second derivatives everywhere.

9.1.3 The truncated power basis for splines

For simplicity, let us consider a general p th degree spline with a single knot at t . Let $P(x)$ denote an arbitrary p th degree polynomial

$$P(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_p x^p.$$

Then

$$S(x) = P(x) + \beta_{p+1}(x - t)_+^p \quad (9.1)$$

takes on the value $P(x)$ for any $x \leq t$, and it takes on the value $P(x) + \beta_{p+1}(x - t)^p$ for any $x > t$. Thus, restricted to each region, the function is a p th degree polynomial. As a whole, this function is a p th degree piecewise polynomial; there are two pieces. Note that the pieces have been joined in such a way as to ensure that the resulting function is as smooth as possible.

We require $p + 2$ coefficients to specify this piecewise polynomial. This is one more coefficient than the number needed to specify a p th degree polynomial, and this is a result of the addition of the truncated power function specified by the knot at t . In

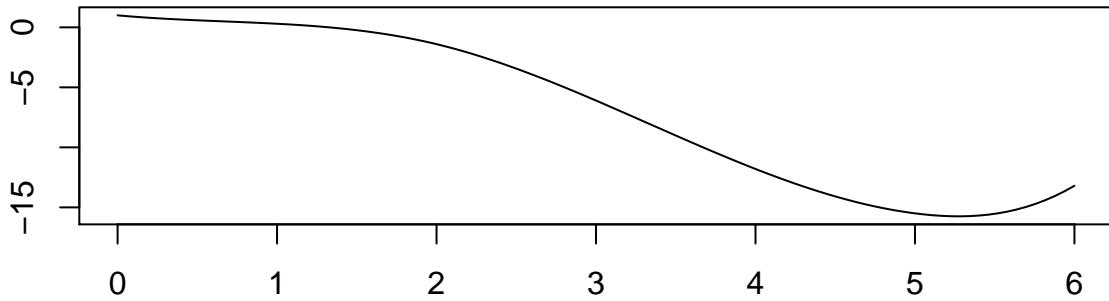


Figure 9.3: An example of a cubic spline with two knots, at $x = 2$ and $x = 5$.

general, we may add k truncated power functions specified by knots at t_1, t_2, \dots, t_k , respectively, each multiplied by different coefficients. Since there are $p + k + 1$ such coefficients, we have $p + k + 1$ degrees of freedom in the specification of such a spline.

This discussion indicates that we can represent any piecewise polynomial of degree p , possessing p continuous derivatives, in the following way:

$$S(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \dots + \beta_p x^p + \beta_{p+1}(x - t_1)_+^p + \dots + \beta_{p+k}(x - t_k)_+^p \quad (9.2)$$

This can be rigorously proven. See de Boor (1978), for example. Display (9.2) says that any piecewise polynomial can be expressed as a linear combination of truncated power functions and polynomials of degree p . In other words,

$$\{1, x, x^2, \dots, x^p, (x - t_1)_+^p, (x - t_2)_+^p, \dots, (x - t_k)_+^p\}$$

is a basis for the linear space of p th degree splines possessing knots at t_1, t_2, \dots, t_k . This basis is referred to as the truncated power basis.

9.1.4 Exercises

1. Evaluate the following at $x = 3$ using the specified knots t .

- (a) $(x - t)_+^2$, with $t = 7$.¹
- (b) $(x - t)_+^2$, with $t = 1$.²
- (c) $(x - t)_+^3$, with $t = 1$.³
- (d) $(x - t)_+^3$, with $t = -1$.⁴
- (e) $(x - t)_+^0$, with $t = 1$.⁵

2. Express the function

$$f(x) = \begin{cases} 2x + 5, & x < 3 \\ 4x - 1, & x \geq 3 \end{cases}$$

as a linear combination of the truncated power basis functions $\{1, x, (x - 3)_+\}$.⁶

3. Express the function

$$f(x) = 3 - 2x + x^2 + 7(x - 5)_+^2$$

in terms of two polynomials, the first defined on the domain where $x < 5$ and the second defined on the domain where $x \geq 5$.⁷

¹0

²4

³8

⁴64

⁵1

⁶ $f(x) = 5 + 2x + 2(x - 3)_+$

⁷When $x < 3$, $f(x) = 3 - 2x + x^2$; when $x \geq 3$, $f(x) = 178 - 72x + 8x^2$.

4. Use the `curve()` function to plot the quadratic spline function $f(x) = 3 + 2x - x^2 + 2(x - 1)_+^2 - 3(x - 3)_+^2$ on the interval $[0, 5]$.⁸ You can use code such as the following to construct the relevant R function:

```
f <- function(x) {
  3 + 2*x - x^2 + 2*tr.pwr(x, 1, 2) - 3*tr.pwr(x, 3, 2)
}
```

5. Plot the cubic spline function $f(x) = 8 - x + x^2 + 0.5x^3 - 2(x - 1.5)_+^3 - 3(x - 7)_+^3$ on the interval $[0, 10]$.⁹

9.2 Least-squares splines

By adding a noise term to (9.2), we can obtain a spline regression model relating a response $y = S(x) + \varepsilon$ to the predictor x . The model to be fit is of the form

$$y_j = \beta_0 + \beta_1 x_j + \cdots + \beta_p x_j^p + \beta_{p+1}(x_j - t_1)_+^p + \cdots + \beta_{p+k}(x_j - t_k)_+^p + \varepsilon_j, \quad j = 1, 2, \dots, n$$

where the noise terms $\{\varepsilon_j\}$ satisfy the usual conditions (mean 0, independence, and constant variance). In vector-matrix form, we may write

$$\mathbf{y} = \mathbf{T}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \quad (9.3)$$

where \mathbf{T} is an $n \times (p + k + 1)$ matrix whose first $p + 1$ columns correspond to the model matrix for p th degree polynomial regression, and whose $(j, p + 1 + i)$ element is $(x_j - t_i)_+^p$.

Applying least-squares to (9.3), we see that

$$\hat{\boldsymbol{\beta}} = (\mathbf{U})^{-1} \mathbf{Q}_1^T \mathbf{y} \quad (9.4)$$

where $\mathbf{T} = \mathbf{Q}_1 \mathbf{U}$, using the QR decomposition. The hat matrix or influence matrix is as in ordinary multiple regression: $\mathbf{H} = \mathbf{Q}_1 \mathbf{Q}_1^T$. Thus, all of the usual linear regression technology is at our disposal here, including standard error estimates for coefficients and confidence and prediction intervals. Even regression diagnostics are applicable.

9.2.1 Example: kiwifruit resistance

The `fruitohms` data frame in the `DAAG` package furnishes an example where a nonlinear regression function may be appropriate.

We first show how one might fit a linear spline function with a few knots. Determining the range of the predictor variable is the first step: among other things, this tells us the domain of the knots as well as the gridpoints where the fitted spline function will be evaluated.

```
library(DAAG)
x <- fruitohms$juice
y <- fruitohms$ohms
range(x) # we should place a few knots inside this interval
## [1] 4 60
```

Although choosing equally spaced knots does not necessarily give the most accurate spline estimates, we will do so here, to illustrate the method. The knots for the first estimate are $\{10, 30, 50\}$.

```
knots <- seq(10, 50, 20)
```

The next step is to construct the design matrix. For the linear spline, we need a column of ones, the predictor values, and the truncated powers associated with each of the 3 knots, evaluated at the predictor values. The `outer()` function can be used to construct a matrix whose (i, j) th entry is the truncated linear function with the j th knot evaluated at x_i .

⁸`curve(f(x), 0, 5)`
`f <- function(x) {`
 `8 - x + x^2 + 0.5x^3 + 2*tr.pwr(x, 1.5, 3) - 3*tr.pwr(x, 7, 3)`
`}`

⁹`curve(f(x), 0, 10)`

```
n <- length(x)
T <- cbind(rep(1,n), x)
T <- cbind(T, outer(x, knots, tr.pwr, degree=1))
```

We can use the `lm()` function to perform the least-squares calculations. Usually, we are not interested in the summary statistics when doing this kind of smoothing, but the result is shown here for completeness. Note that since the design matrix includes a column of ones, the `lm()` function was invoked without intercept. That is, the model formula was $y \sim T - 1$ instead of the usual $y \sim T$. Alternatively, we could construct the design matrix T without including the column of ones, and then use the model formula $y \sim T$ in the call to `lm()`.

```
kiwi.lsp <- lm(y ~ T - 1)
summary(kiwi.lsp)

##
## Call:
## lm(formula = y ~ T - 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3363.5  -597.1   -16.1    482.0   2849.1
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## T          4132.95    1019.98   4.052 8.93e-05
## Tx         347.24     111.04   3.127  0.0022
## T          -533.38    118.96  -4.484 1.66e-05
## T          138.54     31.75   4.364 2.68e-05
## T          104.87     53.94   1.944  0.0541
##
## Residual standard error: 944.2 on 123 degrees of freedom
## Multiple R-squared:  0.9618, Adjusted R-squared:  0.9603
## F-statistic: 619.9 on 5 and 123 DF, p-value: < 2.2e-16
```

The fitted model is

$$\hat{E}[y] = 4132.95 + 347.24x - 533.38(x - 10)_+ + 138.54(x - 30)_+ + 104.87(x - 50)_+$$

Usually, there is not much interest in the fitted model coefficient values when smoothing, since the terms in the model are not very interpretable, beyond indicating that at the knots, the regression function slope has changed.

In the present case, the p -values for most coefficients are small. Although care should be exercised in interpreting these results, they seem to suggest that the knot choice may be reasonable, although there is no suggestion that enough knots have been selected.

Plotting the regression function estimate is usually of higher priority. To do this, a set of gridpoints corresponding to the range of the predictor must be selected. At these gridpoints, the fitted model is evaluated. Figure 9.4 shows the data with the overlaid linear spline. The required code is as follows.

```
gridpoints <- seq(4, 60, .5)
T <- outer(gridpoints, knots, tr.pwr, degree=1)
T <- cbind(rep(1,length(gridpoints)), gridpoints, T)
ohms.predicted <- T %*% coef(kiwi.lsp)
```

Observe that the result is continuous but not smooth, since a broken line is being used.

Figure 9.5 shows the standard diagnostic plots for the linear spline estimate of the regression function for the kiwi resistance data. Recall that these plots can be obtained using the following code.

```
par(mfrow=c(2,2))
plot(kiwi.lsp)
```

There is a slight indication of nonlinearity in the first residual plot which may be suggesting that more knots are needed.

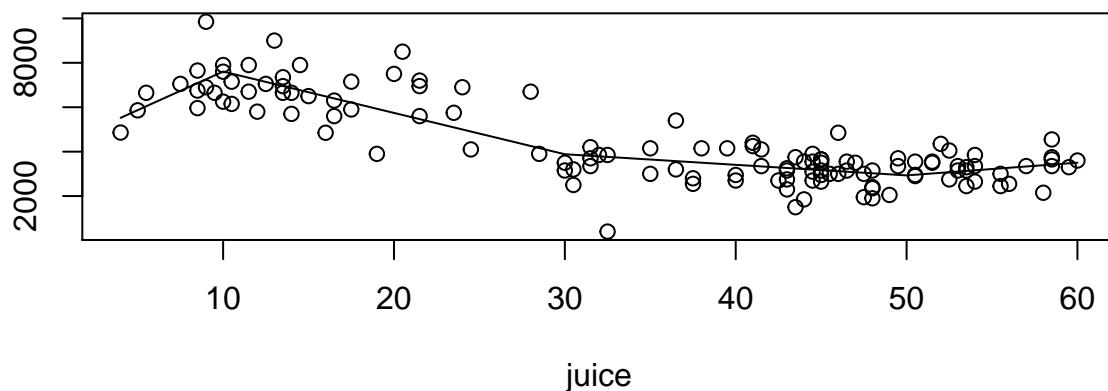


Figure 9.4: The kiwi resistance data with a piecewise linear spline overlaid. The spline has knots at 10, 30 and 50.

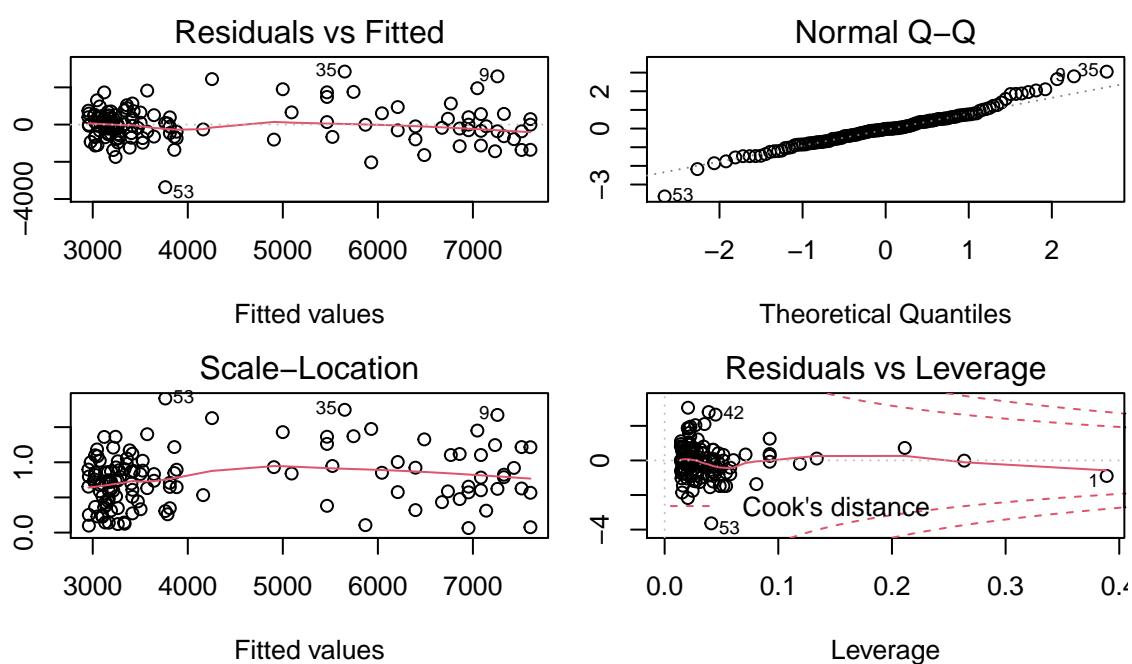


Figure 9.5: The standard diagnostic plots for the linear spline fit to the kiwi resistance data.

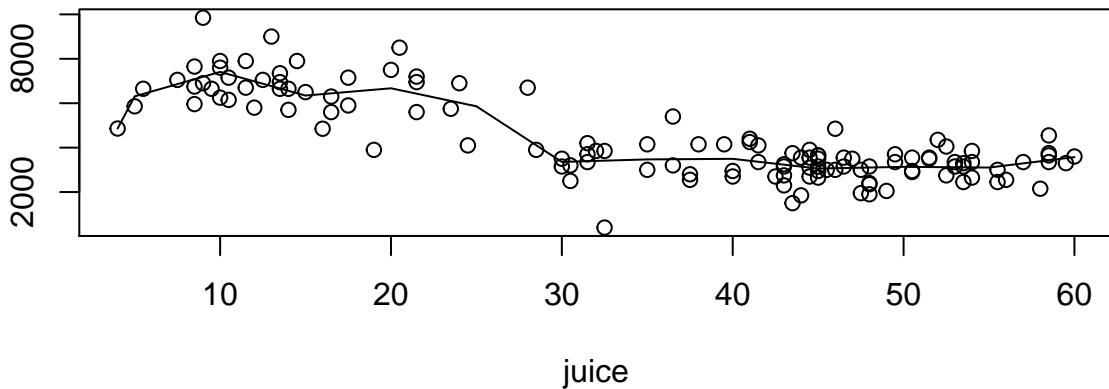


Figure 9.6: Linear spline estimate of the regression function for the kiwi resistance data using many equally-spaced knots.

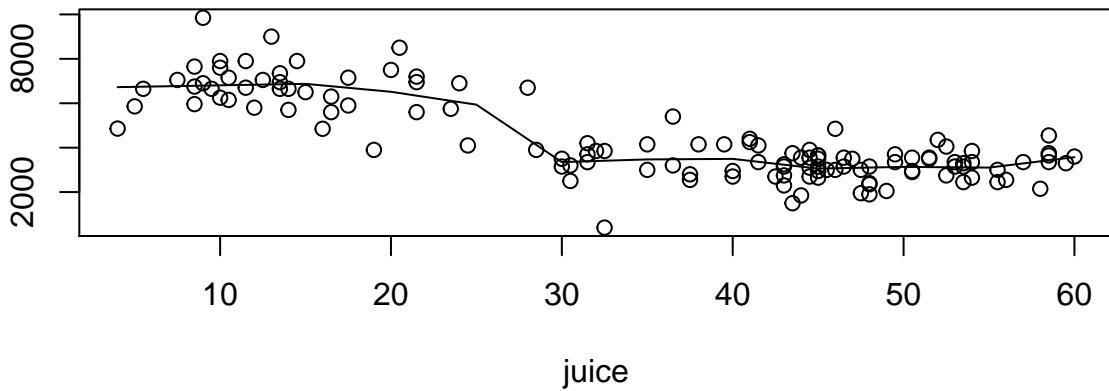


Figure 9.7: Linear spline estimate of the regression function for the kiwi resistance data using many equally-spaced knots but without knots at $x = 5$ and $x = 10$.

9.2.2 Using more knots

Substantially more knots are used in our next estimate. Figure 9.6 shows the fitted curve overlaying the data, after applying the following code.

```

knots <- seq(5, 55, 5) # knot locations: 5, 10, ..., 55
X <- outer(x, knots, tr.pwr, degree=1) # linear spline columns
X <- cbind(x, X) # appending column for linear term
kiwi.lsp <- lm(y ~ X) #
X <- outer(gridpoints, knots, tr.pwr, degree=1)
X <- cbind(rep(1,length(gridpoints)), gridpoints, X)
# predicted resistance values at gridpoints:
ohms.predicted <- X%*%coef(kiwi.lsp)

```

Of course, the result is still not smooth, but more features of the data are reflected in this fit. Note, however, that the rapid increase in resistance for low juice content may be an artifact of the choice of knots rather than a true reflection of the underlying science. In fact, it is possible that there is some confounding in the data. In these kinds of experiments, it is not unusual to begin taking measurements at the low juice concentrations and increase the concentration at each trial; it is also not unusual for more experimental errors to occur when the first few measurements have been taken than at later times.

Figure 9.7 shows what happens when the knots at $x = 5$ and $x = 10$ have been removed. The result is somewhat more believable than the previously obtained estimates.

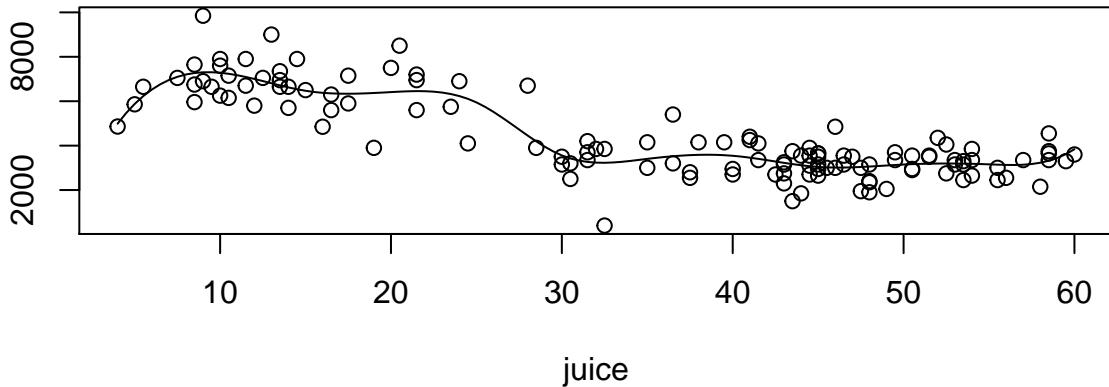


Figure 9.8: Cubic spline estimate of the regression function for the kiwi resistance data using many equally-spaced knots.

9.2.3 A cubic spline estimate

Figure 9.8 shows an example of a cubic spline fit to the kiwi resistance data, according to the following code:

```

knots <- seq(15, 55, 5)
X <- outer(x, knots, tr.pwr, degree=3)
X <- cbind(x, x^2, x^3, X)
kiwi.csp <- lm(y ~ X)
X <- outer(gridpoints, knots, tr.pwr, degree=3)
X <- cbind(rep(1,length(gridpoints)), gridpoints,
           gridpoints^2, gridpoints^3, X)
# predicted resistance values at gridpoints:
ohms.predicted <- X%*%coef(kiwi.csp)

```

Again, we see the effect of the knots on the left side of the figure. We also observe that, although the regression is smoother than for the linear spline, there are some spurious increases in resistance for increasing juice content. Knot selection is clearly an issue. Of course, it is also possible to use weighted regression; small weights for the first few observations may be appropriate here.

9.2.4 Numerical inaccuracies involving the truncated power basis

In addition to the problem of knot selection, a major difficulty is the poor conditioning of the truncated power basis which will result in inaccuracies in the calculation of $\hat{\beta}$.

If the QR decomposition is not used, the normal equations for the least-squares problem lead to

$$\hat{\beta} = (\mathbf{T}^\top \mathbf{T})^{-1} \mathbf{T}^\top \mathbf{y}$$

which is mathematically equivalent to the form given at (9.4), but which is numerically much harder to work with.

The condition number of a matrix gives a good indication as to whether it will lead to numerical inaccuracies. It is defined as the ratio of the maximum to the minimum eigenvalue in absolute value of the matrix, and can be obtained in R using `kappa()`. For example, the eigenvalues of the identity matrix are all 1, so the condition number is 1. For the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & .999 \\ .999 & 1 \end{bmatrix}$$

the condition number can be computed as

```

A <- matrix(c(1, .999, .999, 1), nrow=2)
kappa(A)

## [1] 1999.999

```

This number is fairly large. This indicates that the matrix is almost singular (which is also clear, by inspection). For the cubic spline estimate computed earlier, we compute the condition number as follows.

```

knots <- seq(15, 55, 5)
T <- outer(x, knots, tr.pwr, degree=3)
T <- cbind(rep(1,n), x, x^2, x^3, T)
kappa(t(T) %*% T)
## [1] 1.145472e+13

```

This condition number is enormous, indicating severe ill-conditioning. There is nothing particularly special about this example. The knots are not unusually close together. Rather, this is typical of the behaviour of the truncated power basis. It is for this reason that the B-spline basis was originally introduced.

9.2.5 Exercises

1. Consider the data set

x	y
0	5
1	7
2	4
3	2
4	-3
5	-1

Write out the model matrix for linear spline regression, assuming that a single knot is used at $x = 2$.¹⁰

2. For the data of the previous question, write out the model matrix for quadratic spline regression, assuming that a single knot is used at $x = 2$.¹¹
3. The data `cfseal` consists of observations on taken on a sample of 30 Cape Fur Seals that died accidentally due to commercial fishing. Interest centers on predicting age, given the other variables.

- (a) Write down the **first three rows** of the model matrix which would be used to fit the linear spline model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 (x_1 - 100)_+^2 + \beta_4 (x_1 - 200)_+^2 + \beta_5 x_2 + \varepsilon$$

where y represents age, x_1 represents right kidney weight, x_2 represents the weight of the intestines and the error term, ε , satisfies the usual assumptions.

- (b) Use the `lm()` function and `tr.pwr()` function to fit the above quadratic spline model.

¹⁰

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 2 & 0 \\ 1 & 3 & 1 \\ 1 & 4 & 2 \\ 1 & 5 & 3 \end{bmatrix}$$

¹¹

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 2 & 4 & 0 & 0 \\ 1 & 3 & 9 & 1 & 1 \\ 1 & 4 & 16 & 2 & 4 \\ 1 & 5 & 25 & 3 & 9 \end{bmatrix}$$

9.3 B-splines

The normal equations associated with the truncated power basis are highly ill-conditioned. In the polynomial context, this problem is avoided by considering orthogonal polynomials. Although it is not orthogonal, there is a basis which is reasonably well-conditioned: the B-spline basis.

Let us first consider the piecewise constant functions (0th degree splines). These are normally not of much practical use. However, they provide us with a starting point to build up the B-spline basis.

The truncated power basis for the piecewise constant functions on an interval $[a, b]$ having knots at $t_1, t_2, \dots, t_k \in [a, b]$ is

$$\{1, (x - t_1)_+^0, \dots, (x - t_k)_+^0\}.$$

Note that these basis functions overlap a lot. That is, they are often 0 together or 1 together. This will lead to the ill-conditioning when applying least-squares mentioned earlier. A more natural basis for piecewise constant functions is the following

$$\{1_{x \in (a, t_1)}, 1_{x \in (t_1, t_2)}(x), 1_{x \in (t_2, t_3)}(x), \dots, 1_{x \in (t_{k-1}, t_k)}(x), 1_{x \in (t_k, b)}(x)\}$$

To see that this is a basis for piecewise constant functions on $[a, b]$ with jumps at t_1, \dots, t_k , note that any such function can be written as

$$S_0(x) = \beta_0 1_{x \in (a, t_1)} + \sum_{j=1}^{k-1} \beta_j 1_{x \in (t_k, t_{k+1})} + \beta_k 1_{x \in (t_k, b)}.$$

Note that if we try to fit such a function to data of the form $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, we require at least one x value between any pair of knots (which include a and b , the so-called ‘Boundary knots’) in order to obtain a full-rank model matrix. This is required for invertibility of the normal equations. Note, as well, that the columns of the model matrix are orthogonal which implies that this least-squares problem is now very well conditioned. The reason for the orthogonality is that the regions where the functions are non-zero do not overlap; each basis function is non-zero only on a single interval of the form $[t_i, t_{i+1}]$. Finally, observe that the size of this basis is the same as the truncated power basis, so we are requiring the same numbers of degrees of freedom.

Moving to the piecewise linear case, we note first that the piecewise constant B-splines could be obtained from their truncated counterparts by differencing. Two levels of differencing applied to the truncated linear functions on $[a, b]$ with knots at t_1, \dots, t_k gives a set of continuous piecewise linear functions $B_{i,1}(x)$ which are non-zero on intervals of the form $[t_i, t_{i+2}]$. These functions are called “hat” or “chapeau” functions. They are not orthogonal, but since $B_{i,1}(x)$ is nonzero only with $B_{i-1,1}(x)$ and $B_{i+1,1}(x)$, the resulting model matrix used in computing the least-squares regression estimates is well-conditioned.

9.3.1 Example: changing linear spline bases

Suppose

$$S(x) = \beta_0 + \beta_1(x - t_1) + \beta_{11}(x - t_1)_+$$

for $x \in [t_0, t_2]$, assuming $t_0 < t_1 < t_2$. We now want to show that $S(x)$ can be written as a linear combination of three B-splines, based on knots at t_0, t_1 and t_2 :

$$\begin{aligned} B_0(x) &= 1 - \frac{x - t_0}{t_1 - t_0} + \frac{(x - t_1)_+}{t_1 - t_0} \\ B_1(x) &= \frac{x - t_0}{t_1 - t_0} + (x - t_1)_+ \frac{t_0 - t_2}{(t_0 - t_1)(t_1 - t_2)} \\ B_2(x) &= \frac{(x - t_1)_+}{t_2 - t_1} \end{aligned}$$

These three functions are plotted on the interval $[0.3, 1.5]$ in Figure 9.9 using $t_0 = 0.3, t_1 = 0.7$ and $t_2 = 1.5$. $B_0(x)$ is the red dashed curve, $B_1(x)$ is the solid black curve and $B_2(x)$ is the dashed blue curve. Code for producing the curves is as follows.

```
t0 <- 0.3; t1 <- 0.7; t2 <- 1.5
B0 <- function(x) 1 - (x-t0)/(t1-t0) + tr.pwr(x, t1, 1)/(t1-t0)
B1 <- function(x) (x-t0)/(t1-t0) + tr.pwr(x, t1, 1)*(t0-t2)/((t0-t1)*(t1-t2))
B2 <- function(x) tr.pwr(x,t1,1)/(t2-t1)
```

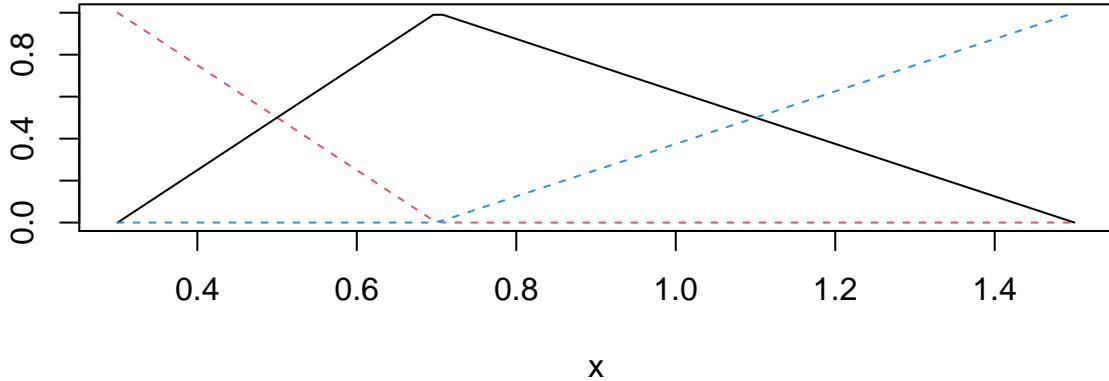


Figure 9.9: The three linear B-spline functions associated with knots at 0.3, 0.7 and 1.5.

```
curve(B0(x), 0.3, 1.5, col=2, lty=2)
curve(B1(x), 0.3, 1.5, add=TRUE, col=1)
curve(B2(x), 0.3, 1.5, add=TRUE, col=4, lty=2)
```

The problem of changing the basis is equivalent to asking whether $S(x)$ can be expressed as a linear combination of $B_0(x)$, $B_1(x)$ and $B_2(x)$:

$$S(x) = A_0 B_0(x) + A_1 B_1(x) + A_2 B_2(x).$$

Using the definition of the B-splines and from the definition of $S(x)$, we match coefficients of 1, $(x - t_1)$ and $(x - t_1)_+$ in order to determine A_0 , A_1 and A_2 .

$$\begin{aligned} \beta_0 &= A_0 \\ \beta_1 &= \frac{A_0}{t_1 - t_0} + \frac{A_1}{t_1 - t_0} \\ \beta_2 &= \frac{A_0}{t_1 - t_0} + \frac{A_1(t_0 - t_2)}{(t_0 - t_1)(t_1 - t_2)} + \frac{A_2}{t_2 - t_1} \end{aligned}$$

Solving this system of equations for A_0 , A_1 and A_2 results in

$$A_0 = \beta_0$$

$$A_1 = \beta_1(t_1 - t_0) - \beta_0$$

and

$$A_2 = \beta_2(t_2 - t_1) + \beta_1(t_2 - t_0) - \beta_0 \left(1 + 2 \frac{t_2 - t_1}{t_0 - t_1} \right).$$

This means that any linear spline with a knot at t_1 can be expressed as a linear combination of a constant, linear and truncated linear function or as a linear combination of the three B-spline functions.

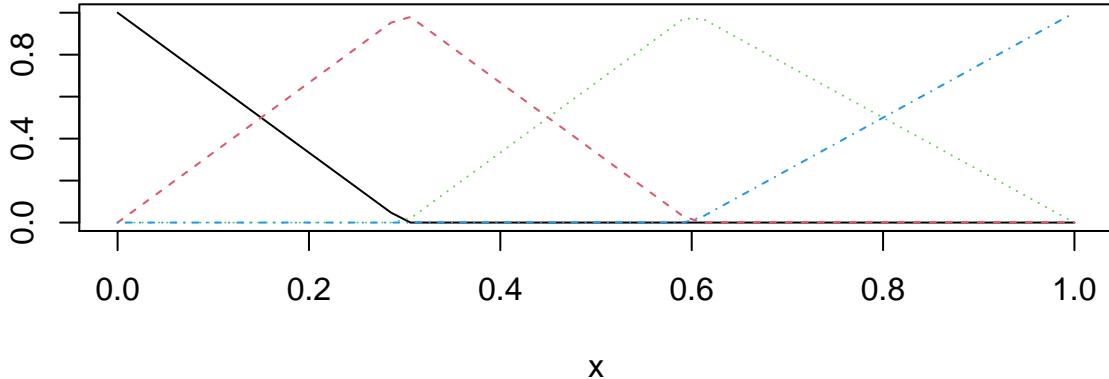
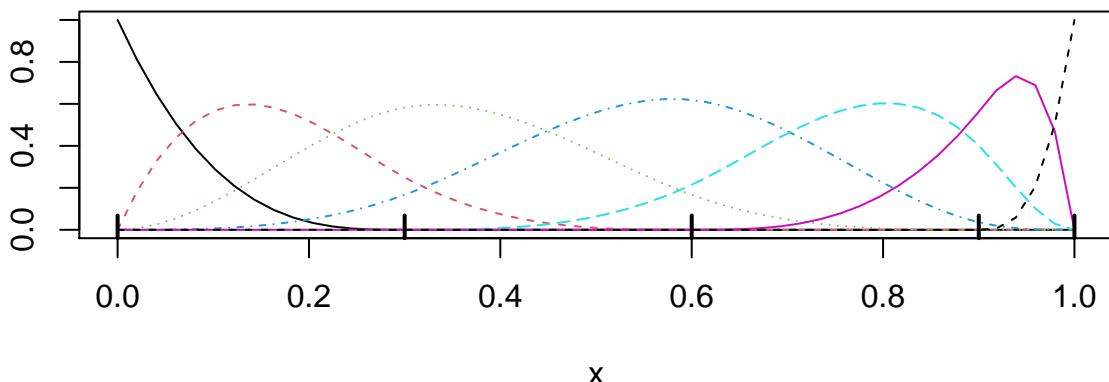
9.3.2 Calculating B-spline values

The more common way of computing the linear B-spline sequence is through the formula

$$B_{i,1}(x) = \frac{(x - t_i)}{t_{i+1} - t_i} 1_{x \in [t_i, t_{i+1})}(x) + \frac{(t_{i+2} - x)}{t_{i+2} - t_{i+1}} 1_{x \in [t_{i+1}, t_{i+2})}(x), \quad i = -1, 2, \dots, k \quad (9.5)$$

This formula is implemented in the `b_s()` function in the `splines` package.

By taking $t_{-1} = t_0 = a$ and $t_{k+1} = b$, we can use the above formula to compute $B_{-1,1}(x)$ and $B_{0,1}(x)$, $B_{k-1,1}(x)$ and $B_{k,1}(x)$, for $x \in [a, b]$, using the convention that $0 \times \infty = 0$. In total, this gives us $k + 2$ basis functions for continuous piecewise linear splines. For the case of knots at .3 and .6 and boundary knots at 0 and 1, these functions are plotted in Figure 9.10. To obtain the plot, type

Figure 9.10: Linear B-splines on $(0, 1)$ corresponding to knots at .3 and .6.Figure 9.11: Cubic B-splines on $(0, 1)$ corresponding to knots at .3, .6 and .9.

```
library(splines)
x <- seq(0, 1, length=50)
matplot(x, bs(x, knots=c(.3, .6), intercept=TRUE, degree=1) [, 1:4], type="l")
```

Note that, with two knots, there are 4 basis functions. The middle two have the “hat” shape, while the first and last are “incomplete” hats. Because the B-splines are nonzero on at most the union of two neighbouring inter-knot subintervals, some pairs of B-splines are mutually orthogonal.

Higher degree B-splines can be obtained, recursively, using an extension of the formula given at (9.5). That is, the p th degree B-spline functions are evaluated from the $p - 1$ st degree B-spline functions using

$$B_{i,p}(x) = \frac{(x - t_i)}{t_{i+p} - t_i} B_{i,p-1}(x) + \frac{(t_{i+p+1} - x)}{t_{i+p+1} - t_{i+1}} B_{i+1,p-1}(x), \quad i = -p, -p + 1, \dots, k. \quad (9.6)$$

Here, $t_0 = t_{-1} = t_{-2} = \dots = t_{-p} = a$ and $t_k = b$.

Figure 9.11 illustrates the 7 (i.e. $p + k + 1$) cubic B-splines on $[0, 1]$ having knots at .3, .6 and .9. The knot locations have been highlighted using the `rug()` function. This plot is obtained using

```
matplot(x, bs(x, knots=c(.3, .6, .9), intercept=TRUE, degree=3) [, 1:7], type="l")
rug(c(0, .3, .6, .9, 1), lwd=2, ticksize=.1)
```

From this figure, it can be seen that the i th cubic B-spline is nonzero only on the interval $[t_i, t_{i+4}]$. In general, the i th p degree B-spline is nonzero only on the interval $[t_i, t_{i+p+1}]$. This property ensures that the i th and $i + j + 1$ st B-splines are orthogonal, for $j \geq p$. B-splines whose supports overlap are linearly independent.

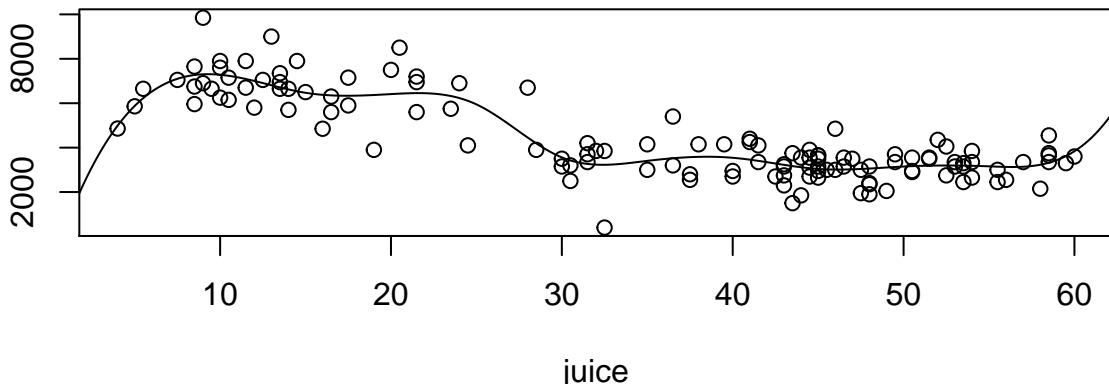


Figure 9.12: Cubic spline estimate of the regression function for the kiwi resistance data using many equally-spaced knots, and using the B-spline basis.

9.3.3 Regression with B-splines

Using the B-spline basis, we can re-formulate the regression model as

$$y_j = \sum_{i=0}^{p+k} \beta_i B_{i,p}(x_j) + \varepsilon_j \quad (9.7)$$

or in vector-matrix form

$$\mathbf{y} = \mathbf{B}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

where the (j, i) element of \mathbf{B} is $B_{i,p}(x_j)$. The least-squares estimate of $\boldsymbol{\beta}$ can be written as

$$\hat{\boldsymbol{\beta}} = (\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{y}.$$

The orthogonality of the B-splines which are far enough apart results in a banded matrix $\mathbf{B}^T \mathbf{B}$ which has better conditioning properties than the matrix $\mathbf{T}^T \mathbf{T}$. The bandedness property allows for the use of more efficient numerical techniques when computing $\hat{\boldsymbol{\beta}}$. Again, all of the usual regression techniques are available. The only drawback with this form of spline regression is that the coefficients are uninterpretable, and the B-splines are less intuitive than the truncated power functions.

9.3.4 Re-visiting the kiwifruit resistance data

Code to re-fit the cubic spline using the same knots as before, but using B-splines, is as follows.

```
kiwi.lm <- lm(ohms ~ bs(juice, knots=seq(15, 55, 5), Boundary.knots=c(0, 65),
  degree=3, intercept=TRUE)-1, data=fruityohms)
```

Again, interest is in plotting the resulting estimate as an overlay on the scatterplot.

```
x.grid <- seq(0, 65, length=401)
y.grid <- predict(kiwi.lm, newdata=data.frame(juice=x.grid))
```

Notice how the same knots and boundary knots are needed to construct the predicted values. Figure 9.12 shows the result of plotting the estimated cubic spline, interpolating the values of y .grid at the gridpoints stored in juice.

9.3.5 Exercises

- Refer to the data in Exercise 1 of Section 9.2. Use the `bs` function in the *splines* package to evaluate the model matrix for the linear spline regression model with a single knot at $x = 2$. Use boundary knots at -1 and 6.¹²

¹²`matrix(bs(x, degree=1, knots=c(2), Boundary.knots=c(-1, 6), intercept=TRUE)), nrow=6)`

2. Refer to the data in Exercise 1 of Section 9.2. Use the `bs` function in the `splines` package to evaluate the model matrix for the quadratic spline regression model with a single knot at $x = 2$.¹³
3. What happens to the matrices in the previous two exercises, if you use `intercept=FALSE`?¹⁴
4. Consider the `geophones` data set in the `DAAG` package.
 - (a) Use the `bs()` and `matplot()` functions to plot three linear B-splines with knots at 25 and 65, evaluated at the values of `distance`, using the default values of `Boundary.knots`.¹⁵
 - (b) Perform the linear spline regression using the B-spline basis obtained in the previous question, and overlay the fitted values (joined together with a broken line).¹⁶
 - (c) Plot the quadratic B-splines corresponding to knots 25, 45, 65, 75 and 78. Use Boundary knots of 0 and 100.¹⁷
 - (d) Perform the quadratic spline regression using the B-spline basis obtained in the previous question, and overlay the fitted values (joined together with a broken line).¹⁸
5. Re-fit the quadratic spline to the Cape Fur Seal data from Exercise 3 of Section 9.2, using B-splines.

9.4 Statistical inference

9.4.1 Pointwise confidence intervals

The spline regression approach allows us to calculate pointwise 95% confidence bands for the true regression function. We can assert, for a given value of x , with 95% confidence, that the true regression function $E[Y|x]$ lies within the bands at x . Code to compute the bands is as follows.

```
y.grid <- predict(kiwi.lm, newdata= data.frame(juice=x.grid), interval="confidence")
```

Figure 9.13 shows the plotted regression function estimate, together with the dashed confidence bands. Code for producing the plot is shown on the figure as well.

```
par(mar=c(4, 3, .1, .1))
plot(fruitohms)
lines(y.grid[,1] ~ x.grid)
lines(y.grid[,2] ~ x.grid, col=2, lty=2, lwd=2)
lines(y.grid[,3] ~ x.grid, col=2, lty=2, lwd=2)
```

The graph tells us, for example, that the expected resistance at 40% juice concentration is between about 3000 ohms and 4500 ohms.

9.4.2 Pointwise prediction intervals

The spline regression approach also allows us to calculate pointwise 95% prediction bands for individual responses. We can assert, for a given value of x , with 95% probability, that a future value of Y lies within the bands at x . Code to compute the bands is as follows.

¹³`matrix((bs(x, degree=2, knots=c(2), Boundary.knots=c(-1, 6), intercept=TRUE)), nrow=6)`

¹⁴The result is a matrix with one fewer column. A slightly different basis is used which does not include the constant term. When using such a matrix in the `lm()` function, an intercept is added unless -1 is employed in the model formula.

¹⁵`x <- geophones$distance`
`x.1 <- bs(x, knots=c(25, 65), degree=1, intercept=TRUE)`

¹⁶`matplot(geophones$distance, x.1, type="l")`
`geo.lm <- lm(thickness ~ x.1 - 1, data=geophones)`

¹⁷`plot(geophones)`
`lines(geophones$distance, predict(geo.lm))`
`x.2 <- bs(x, knots=c(25, 45, 65, 75, 78), degree=2, intercept=TRUE,`

¹⁸`Boundary.knots=c(0, 100))`
`matplot(geophones$distance, x.2, type="l", xlim=c(0,100))`
`geo.lm <- lm(thickness ~ x.2-1, data=geophones)`

¹⁹`plot(geophones)`
`lines(geophones$distance, predict(geo.lm))`

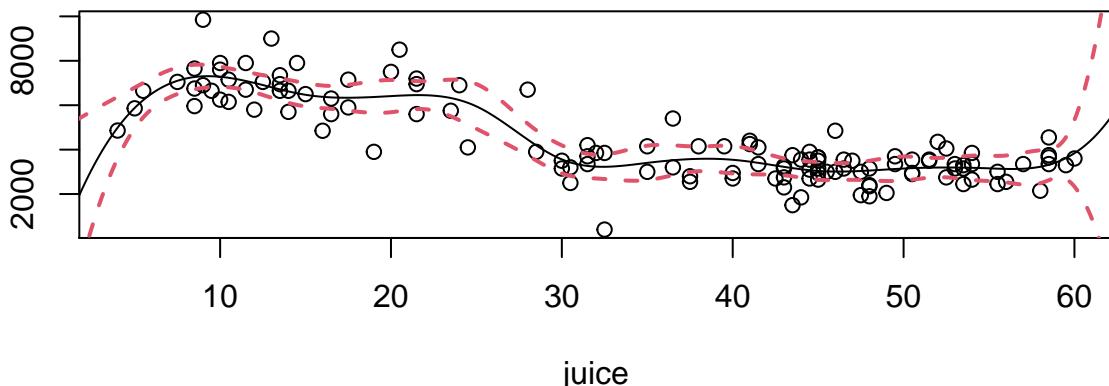


Figure 9.13: Cubic spline estimate of the regression function, together with pointwise 95% confidence interval bands (dashed curves).

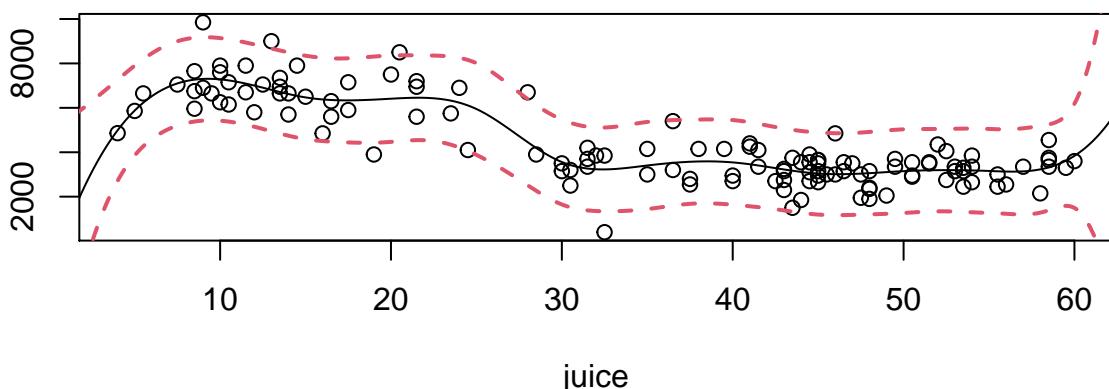


Figure 9.14: Cubic spline estimate of the regression function, together with pointwise 95% prediction interval bands (dashed curves).

```
y.grid <- predict(kiwi.lm, newdata= data.frame(juice=x.grid), interval="prediction")
```

Figure 9.14 shows the plotted regression function estimate, together with the dashed prediction bands. Code for producing the plot is shown on the figure as well.

```
par(mar=c(4, 3, .1, .1))
plot(fruitohms)
lines(y.grid[,1] ~ x.grid)
lines(y.grid[,2] ~ x.grid, col=2, lty=2, lwd=2)
lines(y.grid[,3] ~ x.grid, col=2, lty=2, lwd=2)
```

The graph tells us, for example, that a future value of resistance at 40% juice concentration is between about 1500 ohms and 5500 ohms. As usual, predicting a future value of a response involves much more uncertainty than estimating the mean value of a response.

9.5 Conditioning of the B-spline basis

Recall the cubic spline estimate for the kiwifruit resistance problem in which the truncated power basis led to a highly ill-conditioned design matrix. Using the B-spline basis with the same knots leads to a substantial improvement as can be seen in what follows.

```
B <- bs(fruitohms$juice, knots=seq(15, 55, 5),
Boundary.knots=c(0, 65), degree=3, intercept=TRUE)
kappa(t(B) %*% B)

## [1] 4375.27
```

Although the resulting value is still not very small, it is much less than the extreme value observed when using the truncated power basis.

9.5.1 Exercises

1. Consider the `geophones` data in the *DAAG* package. Fit a linear spline model to the data with knots at 25, 65 and 75. Use boundary knots of 0 and 100. Plot the data, together with predictions at distances in the sequence from 5 through 95, with the corresponding upper and lower 95% confidence bands.¹⁹
2. Repeat the previous question, but use a cubic spline model with knots at 25, 45, 65, 75 and 78.²⁰

9.6 Knot selection and regularization

We have been assuming that the knots are known. In general, they are unknown, and they must be chosen. Badly chosen knots can result in bad approximations. Because the spline regression problem can be formulated as an ordinary regression problem with a transformed predictor, it is possible to apply variable selection techniques such as backward selection to choose a set of knots. The usual approach is to start with a set of knots located at a subset of the order statistics of the predictor. Then backward selection is applied, using the truncated power basis form of the model. Each time a basis function is eliminated, the corresponding knot is eliminated. The method has serious drawbacks, notably the ill-conditioning of the basis as mentioned earlier.

Figure 9.15 exhibits an example of a least-squares spline with automatically generated knots, applied to a data set consisting of titanium measurements. To obtain Figure 9.15, type

```
attach(titanium)
y.lm <- lm(g ~ bs(temperature, knots=c(755, 835, 905, 975),
Boundary.knots=c(550, 1100)))
plot(titanium)
lines(temperature, predict(y.lm))
```

A version of backward selection was used to generate these knots; the stopping rule used was similar to the Akaike Information Criterion (AIC). Although this least-squares spline fit to these data is better than what could be obtained using polynomial regression, it is unsatisfactory in many ways. The flat regions are not modelled smoothly enough, and the peak is cut off.

A substantial improvement can be obtained by manually selecting additional knots, and removing some of the automatically generated knots. In particular, we can render the peak more effectively by adding an additional knot in its vicinity. Adjusting the knot that was already there improves the fit as well. The plot in Figure 9.16 can be generated using

¹⁹
geo.lm <- lm(thickness ~ bs(distance, knots=c(25, 65, 75), degree=1, intercept=TRUE,
Boundary.knots=c(0,100))-1, data=geophones)
plot(thickness ~ distance, data=geophones)
lines(5:95, predict(geo.lm, newdata=data.frame(distance=5:95)))
Pointwise Confidence Bands:
lines(5:95, predict(geo.lm, newdata=data.frame(distance=5:95), interval="confidence") [, "lwr"], lty=2)
lines(5:95, predict(geo.lm, newdata=data.frame(distance=5:95), interval="confidence") [, "upr"], lty=2)
²⁰
geo.lm <- lm(thickness ~ bs(distance, knots=c(25, 45, 65, 75, 78),
degree=3, intercept=TRUE, Boundary.knots=c(0,100))-1, data= geophones)
plot(thickness ~ distance, data=geophones)
lines(5:95, predict(geo.lm, newdata=data.frame(distance=5:95)))
lines(5:95, predict(geo.lm, newdata=data.frame(distance=5:95), interval="confidence") [, "lwr"], lty = 2)
lines(5:95, predict(geo.lm, newdata=data.frame(distance=5:95), interval="confidence") [, "upr"], lty = 2)

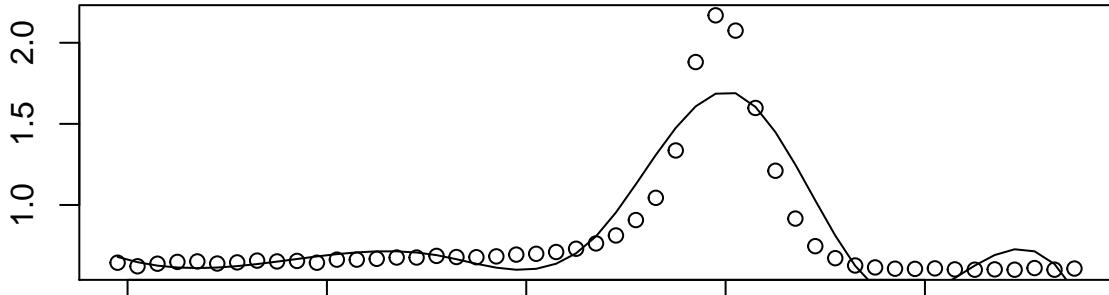


Figure 9.15: A least-squares spline fit to the titanium heat data using automatically generated knots. The knots used were 755, 835, 905, and 975.

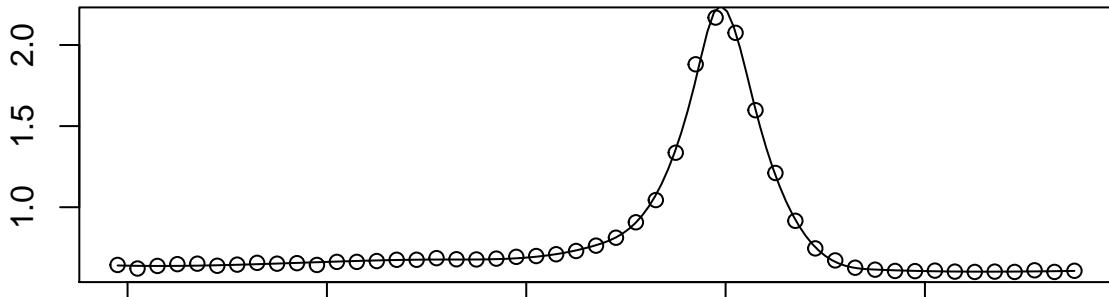


Figure 9.16: A least-squares spline fit to the titanium heat data using manually-selected knots.

```
y.lm <- lm(g ~ bs(temperature, knots=c(755, 835, 885, 895, 915, 975),
Boundary.knots=c(550, 1100)))
plot(titanium)
lines(spline(temperature, predict(y.lm)))
detach(titanium)
```

9.6.1 Smoothing splines

One way around the problem of choosing knots is to use lots of them. A result analogous to the Weierstrass approximation theorem says that any sufficiently smooth function can be approximated arbitrarily well by spline functions with enough knots.

The use of large numbers of knots alone is not sufficient to avoid trouble, since we will over-fit the data if the number of knots k is taken so large that $p + k + 1 > n$. In that case, we would have no degrees of freedom left for estimating the residual variance. A standard way of coping with the former problem is to apply a penalty term to the least-squares problem. One requires that the resulting spline regression estimate has low curvature as measured by the square of the second derivative.

More precisely, one may try to minimize (for a given constant λ)

$$\sum_{j=1}^n (y_j - S(x_j))^2 + \lambda \int_a^b (S''(x))^2 dx$$

over the set of all functions $S(x)$ which are twice continuously differentiable. The addition of a penalty term constrains the problem and is another example of regularization. This concept came up in another guise, when we considered the LASSO. The solution to this minimization problem has been shown to be a cubic spline which is surprisingly easy to calculate.²¹ Thus,

²¹The B-spline coefficients for this spline can be obtained from an expression of the form

$$\hat{\beta} = (\mathbf{B}^\top \mathbf{B} + \lambda \mathbf{D}^\top \mathbf{D})^{-1} \mathbf{B}^\top \mathbf{y}$$

where \mathbf{B} is the matrix used for least-squares regression splines and \mathbf{D} is a matrix that arises in the calculation involving the squared second

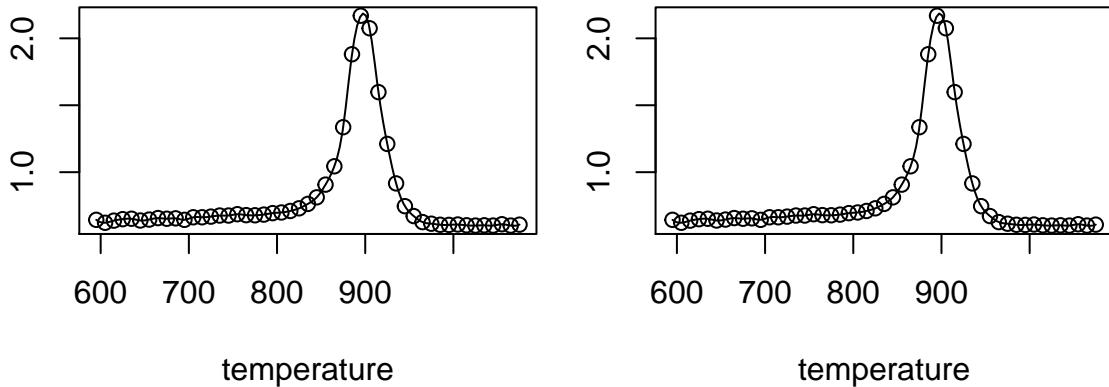


Figure 9.17: A smoothing spline fit to the titanium heat data using the GCV-optimal smoothing parameter (left panel) and CV-optimal smoothing parameter (right panel).

the problem of choosing a set of knots is replaced by selecting a value for the smoothing parameter λ . Note that if λ is small, the solution will be a cubic spline which almost interpolates the data; increasing values of λ render increasingly smooth approximations.

The usual way of choosing λ is by cross-validation. The ordinary cross-validation choice of λ minimizes

$$\text{CV}(\lambda) = \sum_{j=1}^n (y_j - \hat{S}_{\lambda,(j)}(x_j))^2$$

where $\hat{S}_{\lambda,(j)}(x)$ is the smoothing spline obtained using parameter λ , using all data but the j th observation. Note that the CV function is similar in spirit to the PRESS statistic, but it is much more computationally intensive, since the regression must be computed for each value of j . Recall that PRESS could be computed from one regression, upon exploiting the diagonal of the influence matrix. If one pursues this idea in the choice of smoothing parameter, one obtains the so-called generalized cross-validation criterion which turns out to have very similar behaviour, and sometimes superior behaviour, to ordinary cross-validation but with lower computational requirements. The generalized cross-validation calculation is carried out using the trace of a matrix which plays the role of the influence matrix.²²

The left panel of Figure 9.17 exhibits a plot which can be obtained using

```
plot(titanium)
lines(spline(smooth.spline(temperature, g)))
```

This invokes the smoothing spline with smoothing parameter selected by generalized cross-validation. To use ordinary cross-validation, include the argument `cv=TRUE`. This is shown in the right panel of Figure 9.17. In this example, the results are not very different from each other.

In order to see what the value of the smoothing parameter (λ) is, the object created by `smooth.spline()` can be inspected directly..

```
smooth.spline(temperature, g)

## Call:
## smooth.spline(x = temperature, y = g)
##
## Smoothing Parameter  spar= -1.332258  lambda= 1.543412e-18 (22 iterations)
## Equivalent Degrees of Freedom (Df): 49.00001
```

derivatives of the spline. Details can be found in de Boor (1978). It is sufficient to note here that this approach has similarities with ridge regression, and that the estimated regression is a linear function of the responses.

²²The influence matrix is called the smoother matrix in this context, and for smoothing splines, it is of the form

$$\mathbf{H}(\lambda) = (\mathbf{B}^\top \mathbf{B} + \lambda \mathbf{D}^\top \mathbf{D})^{-1} \mathbf{B}^\top.$$

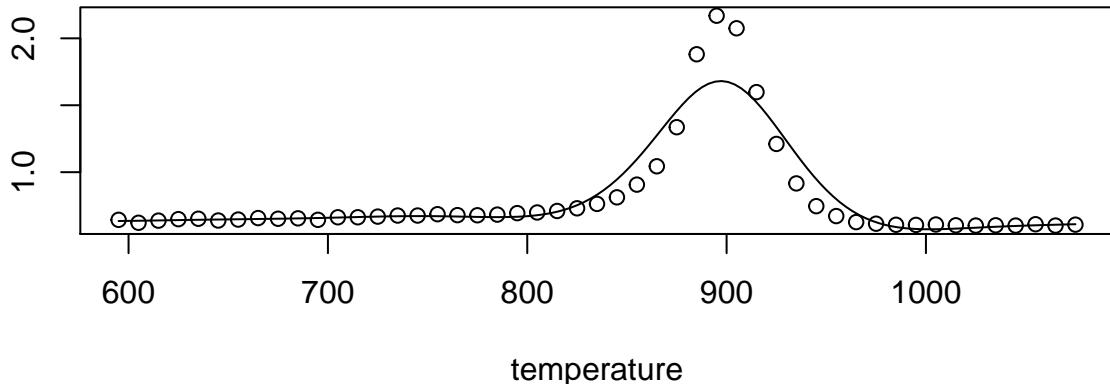


Figure 9.18: A smoothing spline fit to the titanium heat data using the GCV-optimal smoothing parameter (left panel) and CV-optimal smoothing parameter (right panel).

```
## Penalized Criterion (RSS): 1.556956e-25
## GCV: 2.900278e-13

smooth.spline(temperature, g, cv=TRUE)

## Call:
## smooth.spline(x = temperature, y = g, cv = TRUE)
##
## Smoothing Parameter  spar= -1.126165  lambda= 4.757534e-17 (17 iterations)
## Equivalent Degrees of Freedom (Df): 49
## Penalized Criterion (RSS): 1.477955e-22
## PRESS(l.o.o. CV): 0.0005578431
```

From this output, it can be seen that the λ values are different, but both are very small. The effective degrees of freedom for the fits are near 49, which means that the smoothing splines are almost interpolating the data. Contrast this with the cubic regression spline of Figure 9.16 which was based on 6 knots. The number of degrees of freedom required for that fit (i.e. parameters to be estimated) was only 10 ($= 6 + 3 + 1$). Thus, the regression spline might be preferred. The following code produces a smoothing spline estimate based on a smoothing parameter chosen so that the effective degrees of freedom is 10:

```
plot(titanium)
lines(spline(smooth.spline(temperature, g, df=10)))
detach(titanium)
```

The result is plotted in Figure 9.18. Clearly, the smoothing spline has failed here.

This is often the case: avoiding the knot selection problem by using the smoothing spline does not necessarily give a better result than could be obtained by carefully selecting knots.

9.6.2 Penalized splines

The smoothing spline is an example of the use of penalized least-squares. In this case, regression functions which have large second derivatives are penalized, that is, the objective function to be minimized has something positive added to it. Regression functions which have small second derivatives are penalized less, but they may fail to fit the data as well. Thus, the penalty approach seeks a compromise between fitting the data and satisfying a constraint (in this case, a small second derivative).

Eilers and Marx (1996) generalized this idea by observing that other forms of penalties may be used. The penalized splines of Eilers and Marx (1996) are based on a similar idea to the smoothing spline, with two innovations. The penalty term is no longer in terms of a second derivative, but a second divided difference, and the number of knots can be specified. (For smoothing splines, the number of knots is effectively equal to the number of observations.)

Recall that the least-squares spline problem is to find coefficients $\beta_0, \beta_1, \dots, \beta_{k+p}$ to minimize

$$\sum_{j=1}^n (y_j - S(x_j))^2$$

where $S(x) = \sum_{i=0}^{k+p} \beta_i B_{i,p}(x)$. Eilers and Marx (1996) advocate the use of k equally spaced knots, instead of the order statistics of the predictor variable. Note that the number of knots must be chosen somehow, but this is simpler than choosing knot locations.

The smoothness of a spline is related to its B-spline coefficients; thus, Eilers and Marx (1996) replace the second derivative penalty for the smoothing spline with ℓ th order differences of B-spline coefficients. These are defined as follows:

$$\begin{aligned}\Delta\beta_i &= \beta_i - \beta_{i-1} \\ \Delta^\ell\beta_i &= \Delta(\Delta^{\ell-1}\beta_i), \quad \ell > 1.\end{aligned}$$

For example,

$$\Delta^2\beta_i = \Delta(\Delta\beta_i) = \beta_i - 2\beta_{i-1} + \beta_{i-2}$$

By using a penalty of the form $\sum_{i=\ell+1}^k (\Delta^\ell\beta_i)^2$, the smoothness of the resulting regression function estimate can be controlled.

The penalized least-squares problem is then to choose $\beta_0, \beta_1, \dots, \beta_{k+p}$ to minimize

$$\sum_{j=1}^n (y_j - S(x_j))^2 + \lambda \sum_{i=\ell+1}^{k+p} (\Delta^\ell\beta_i)^2 \tag{9.8}$$

for $\lambda > 0$.

In vector-matrix form, this objective function becomes

$$(\mathbf{y} - \mathbf{B}\beta)^\top(\mathbf{y} - \mathbf{B}\beta) + \lambda\beta^\top\mathbf{D}^\top\mathbf{D}\beta \tag{9.9}$$

where \mathbf{D} is a banded matrix whose entries correspond to the difference penalty. For example, when $\ell = 1$,

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 \cdots 0 & 0 \\ -1 & 1 & 0 \cdots 0 & 0 \\ 0 & -1 & 1 \cdots 0 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 \cdots -1 & 1 \end{bmatrix}$$

When $\ell = 2$ (a common choice), the typical row has the nonzero elements 1, -2, 1.

Differentiating (9.9) with respect to β gives

$$-(\mathbf{y} - \mathbf{B}\mathbf{B}^\top)\mathbf{B} + \lambda\beta^\top\mathbf{D}^\top\mathbf{D} = \mathbf{0}$$

or

$$\mathbf{B}^\top\mathbf{y}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{D}^\top\mathbf{D})\beta.$$

The matrix on the right will usually well-conditioned in practice. The B-spline coefficients can thus be estimated as

$$\hat{\beta} = (\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{D}^\top\mathbf{D})^{-1}\mathbf{B}^\top\mathbf{y}.$$

The fitted values of the p-spline estimate can be calculated from

$$\hat{\mathbf{y}} = \mathbf{B}\hat{\beta}.$$

Thus, we can see the form of the influence or ‘hat’ matrix:

$$\mathbf{H} = \mathbf{B}(\mathbf{B}^\top\mathbf{B} + \lambda\mathbf{D}^\top\mathbf{D})^{-1}\mathbf{B}^\top.$$

This is useful in choosing the smoothing parameter λ . We minimize the ordinary cross-validation function or the generalized cross-validation function. These functions are computed as follows:

$$\text{CV}(\lambda) = \sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{1 - h_{ii}} \right)^2$$

$$\text{GCV}(\lambda) = \sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{n - \text{tr}(\mathbf{H})} \right)^2.$$

Eilers and Marx (1996) assert that, in practice, the difference between these two quantities is small.

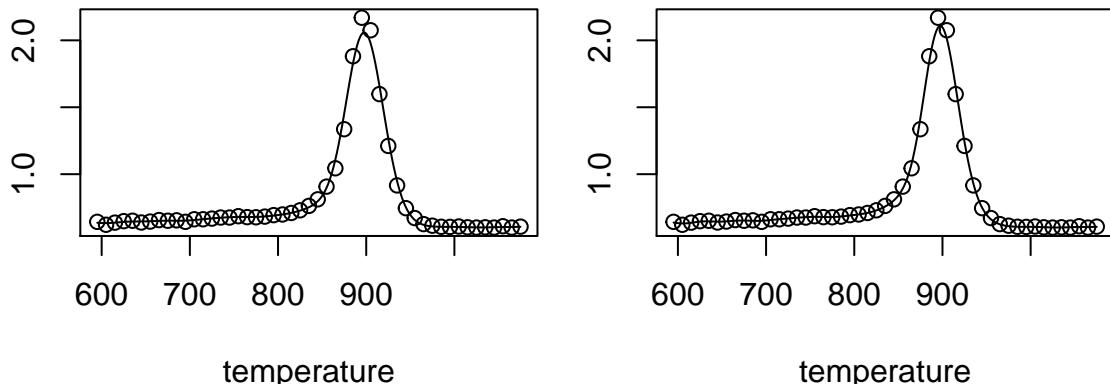


Figure 9.19: Penalized spline fits to the titanium heat data.

Modelling the titanium heat data again

To obtain the plot, code from Eilers and Marx (1996) was used (this is reproduced, with minor variations, in the Appendix to this chapter).

The function `psplinereg()` has a number of arguments in addition to those corresponding to the predictor and response data vectors. Boundary knots are specified by `x1` and `x2` and are taken as the minimum and maximum of the predictor values, by default. The number of knots is specified in terms of the number of intervals `ndx` which separate the knots (the knots are assumed to be equally spaced). The degree of the B-splines is specified by the `bdeg` parameter, and `pord` specifies the order of differencing to be used in the penalty.

The smoothing parameter is chosen by one of three methods: GCV, AIC, and AICC. AICC is an improvement on the AIC which works very well for penalized splines. The `psplinereg()` function evaluates these three criteria at each of a set of values specified by the user in the `LAMBDA` parameter. The method of choosing the optimal λ value is selected according to the `opt` argument. In Figure 9.19, we see penalized cubic spline fits to the titanium heat data, using 80 knots (note that this is more than the number of observations). In the left panel, the default (AICC) is used to choose λ and in the right panel, GCV is used.

```
attach(titanium)
titan.psp <- psplinereg(temperature, g, ndx=80,
                        LAMBDA=seq(.01, 1, length=20))
titan.pspGCV <- psplinereg(temperature, g, ndx=80,
                            LAMBDA=seq(.01, 1, length=20), opt="GCV")
detach(titanium)
```

Again, we can see how many effective degrees of freedom are used in these model fits. This is summarized by the trace of H (why?).

```
titan.psp$trH[which.min(titan.psp$AICC)]
## [1] 24.24057

titan.pspGCV$trH[which.min(titan.psp$gcv)]
## [1] 39.83968
```

The AICC fit uses fewer degrees of freedom than GCV, but also fails to reach the peak. Arguably, the rest of the fit is superior; that is, it is not interpolating the data in the flat areas. In both cases, we have not succeeded in obtaining a model which uses fewer degrees of freedom than the cubic regression spline with 6 manually chosen knots.

Linear penalized splines

Relatively smooth results can be obtained using linear P-splines, when the number of knots is large. For the example below, we obtain similar results to what we obtained for the cubic P-spline. However, this time, we use order 4 for the roughness penalty instead of order 2.

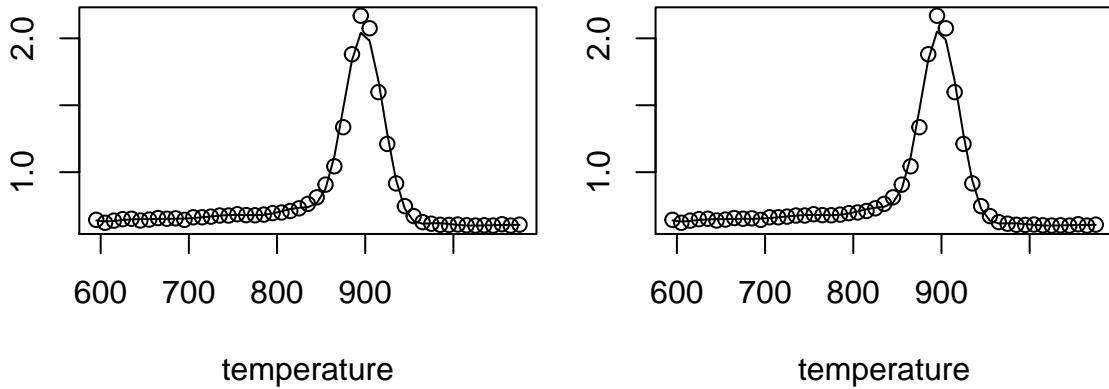


Figure 9.20: Linear penalized spline fits to the titanium heat data.

```
attach(titanium)
titan.linpss <- psplinereg(temperature, g, ndx=50,
                           LAMBDA=seq(.01,1,length=20), bdeg=1, pord=4)
titan.linpssGCV <- psplinereg(temperature, g, ndx=50,
                               LAMBDA=seq(.01,1,length=20), opt="GCV", bdeg=1, pord=4)
detach(titanium)
```

```
titan.linpss$trH[which.min(titan.linpss$AICC) ]
## [1] 25.21031

titan.linpssGCV$trH[which.min(titan.linpss$gcv) ]
## [1] 36.5976
```

Example: modelling the kiwi resistance

Applying the linear P-spline with the 4th order roughness penalty to the kiwi resistance data gives the results shown in Figure 9.21, using 80 equally spaced knots. The AICC criterion uses far more degrees of freedom than necessary, while the GCV fit is excellent, apart from the anomaly associated with the low values of juice content.

```
attach(fruitohms)
kiwi.lsp <- psplinereg(juice, ohms, ndx=80,
                        LAMBDA=seq(.01,1,length=20), bdeg=1, pord=4)
kiwi.lspGCV <- psplinereg(juice, ohms, ndx=20,
                            LAMBDA=seq(.01,1,length=80), opt="GCV", bdeg=1, pord=4)
detach(fruitohms)
```

9.7 Multiple predictors

The previous sections of this chapter were concerned primarily with the case of a single predictor variable x . Extensions of spline and penalized spline methods to multiple predictor variables is possible.

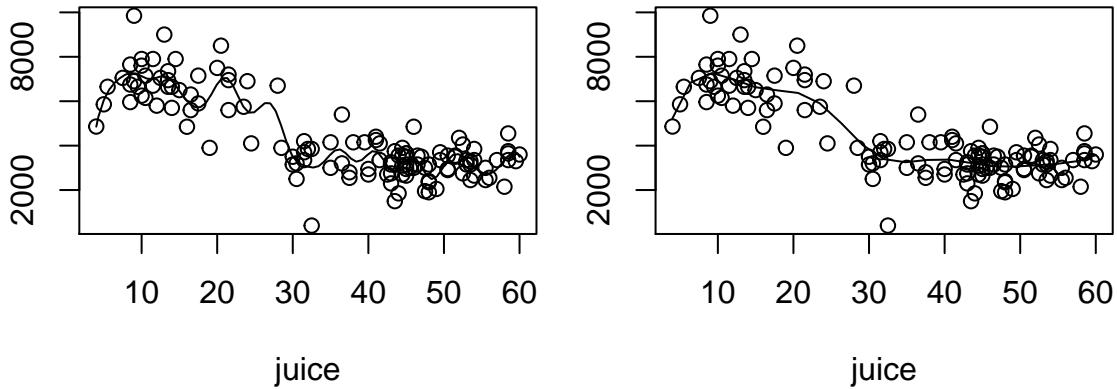


Figure 9.21: Linear penalized spline fits to the kiwi resistance data.

9.7.1 Spline regression

Extensions of the regression spline are straightforward in principle. For example, if x and w are predictors and y is the response, we might consider a model of the form

$$y = \sum_{j=1}^{p_1} \beta_j B_j(x) + \sum_{j=p_1+1}^{p_2} \beta_j B_j(w) + \varepsilon \quad (9.10)$$

where the $B_j(\cdot)$'s denote B-spline functions. Knots have to be chosen along both the x and w axes; alternatively, one could choose equally spaced knots. Interaction terms of the form $\sum_{j \neq k} \beta_{jk} B_j(x)B_k(w)$ could also be included. Note that the number of parameters to estimate quickly increases with the number of knots. This is an example of the so-called ‘curse of dimensionality’.

As an example, we consider data on ore bearing samples listed in the book of Green and Silverman (1994). The response variable is `width`, and this is measured at coordinates given by `t1` and `t2`. The following lines can be used to fit these data.

```
library(splines)
source("orebearing.R")
ore.lm <- lm(width ~ bs(t1, df=5, degree=3) +
               bs(t2, df=5, degree=3), data=orebearing)
```

The usual functions such as `plot()` and `predict()` may be used to analyze the output.

Note that we are assuming no interaction between `t1` and `t2`, an unrealistic assumption for such data. To check this assumption, we can construct a conditioning plot. This plot is pictured in Figure 9.22. The plot clearly shows the need for a model which includes an interaction term.

```
library(lattice)
t2Range <- equal.count(orebearing$t2, number=3)
xyplot(residuals(ore.lm) ~ t1 | t2Range,
       data=orebearing, layout=c(3,1), type=c("p", "smooth"))
```

9.7.2 Additive models

The model (9.10) can be viewed as a simple example of an additive model. It can be easily fit using the `lm()` function. When working with kernel regression, it is also possible to use additive models such as

$$y = \beta_0 + g_1(x) + g_2(w) + \varepsilon.$$

Fitting these kinds of models requires the use of the backfitting algorithm which is as follows:

1. Set $\widehat{\beta}_0 = \bar{y}$.

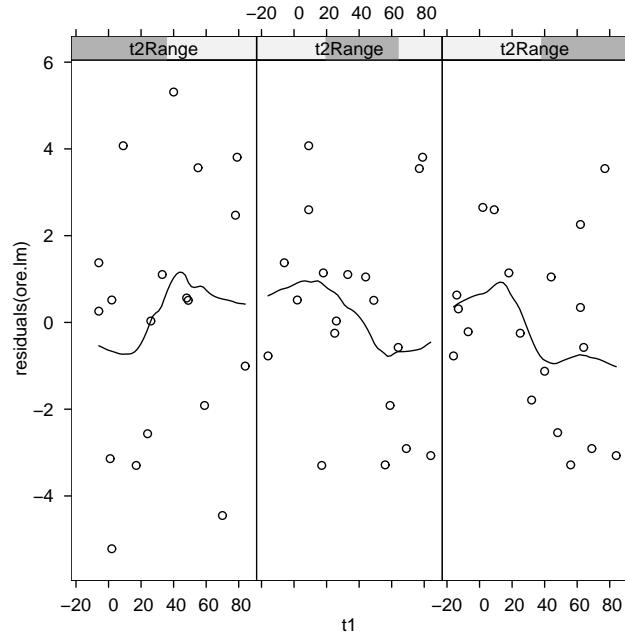


Figure 9.22: Conditioning plots for spline regression applied to the ore bearing data.

2. Obtain an initial estimate of $\hat{g}_2(w)$ by applying univariate nonparametric regression estimation to $\{(w_1, y_1), (w_2, y_2), \dots, (w_n, y_n)\}$.
3. Do the following until convergence:
 - (a) Set $z_{1i} = y_i - \hat{\beta}_0 - \hat{g}_2(w_i)$, for $i = 1, 2, \dots, n$.
 - (b) Estimate $g_1(x)$ by applying univariate nonparametric regression estimation to $\{(x_1, z_{11}), (x_2, z_{12}), \dots, (x_n, z_{1n})\}$.
 - (c) Set $z_{2i} = y_i - \hat{\beta}_0 - \hat{g}_1(x_i)$, for $i = 1, 2, \dots, n$.
 - (d) Estimate $g_2(w)$ from $\{(w_1, z_{21}), (w_2, z_{22}), \dots, (w_n, z_{2n})\}$.

This back-fitting algorithm is an example of the Gauss-Seidel iteration for solving large linear systems of equations. Since Gauss-Seidel is convergent, the back-fitting algorithm is also convergent.

Note that any smoothing method can be used to do the univariate nonparametric regression at each stage. The `gam()` function in the `mgcv` package (Wood, 2017) does additive modelling²³ using smoothing splines at each iteration. The following code demonstrates the elementary usage of the `gam()` function for the ore bearing data. Figure 9.23 is similar to the conditioning plots obtained earlier, and Figure 9.24 demonstrates the basic usage of the `visgam()` function to obtain a perspective plot of the fitted surface.

```
library(mgcv)
ore.gam <- gam(width ~ s(t1) + s(t2), data=orebearing)
plot(ore.gam, pages=1)    # this invokes plot.gam() which
                          # differs from plot.lm()
visgam(ore.gam)          # produces a perspective plot
```

A 2-term additive model

We aim to fit the model

$$y = f_1(x) + f_2(z) + \varepsilon$$

to trivariate data, having two predictors and a single response.

²³Actually, the function does more: *generalized* additive models. For more on this, see the next section of this chapter.

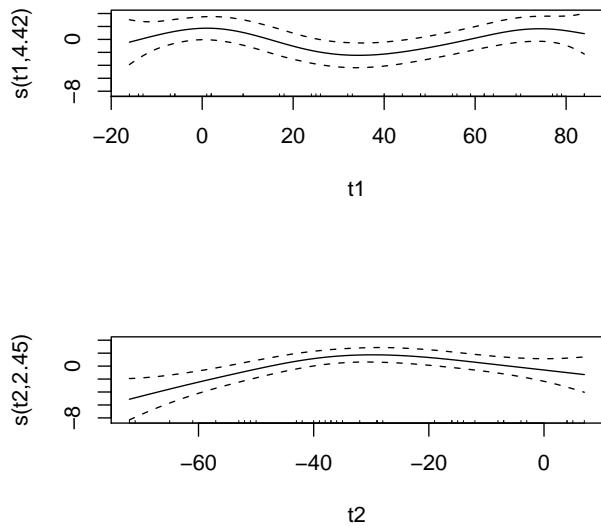


Figure 9.23: Conditional plots for the ore bearing data using the `plot.gam()` function.

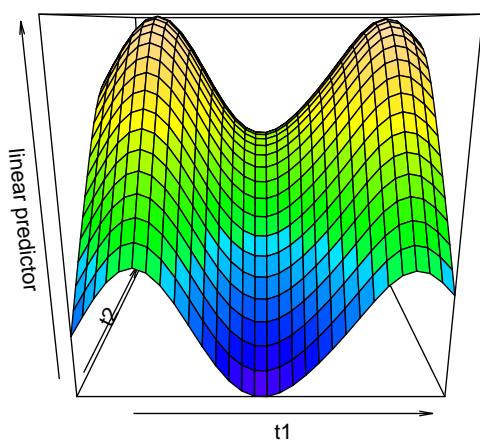


Figure 9.24: A perspective plot of an additive model fit to the ore bearing data.

The technique is to fit a sum of penalized splines, one as a function of x and the other as a function of z . Only one intercept can be used.

The penalized objective function is

$$\|y - [X \ Z] \begin{bmatrix} \beta_X \\ \beta_Z \end{bmatrix}\|^2 + \lambda_X^2 \sum_{j=2}^{d+kx+1} \beta_{X,j}^2 + \lambda_Z^2 \sum_{j=2}^{d+kz+1} \beta_{Z,j}^2.$$

Here X and Z are design matrices corresponding to d -degree splines with kx and kz knots. (Again, note that there is no column of ones in the Z matrix.) β_X and β_Z are the corresponding parameter vectors, and λ_X and λ_Z are the smoothing parameters. These would be chosen by cross-validation or generalized cross-validation.

Again, the actual fitting can be done using `lm()` applied to the augmented problem:

$$\left\| \begin{bmatrix} y \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} X & Z \\ \lambda D_X & 0 \\ 0 & \lambda D_Z \end{bmatrix} \begin{bmatrix} \beta_X \\ \beta_Z \end{bmatrix} \right\|^2.$$

Here, D_X and D_Z are diagonal matrices with 1's and 0's on the diagonal. The 0 vectors under the y are of length $kx+d-1$ and $kz+d-1$, respectively. The 0 matrices are of dimension $(kx+d-1) \times (kz+d)$ and $(kz+d-1) \times (kx+d+1)$, respectively.

9.7.3 A practical overview of generalized additive models

As we have seen, the `gam()` function in the R Package *mgcv* (Wood, 2017) can be used to fit additive models of the form

$$E[y|x_1, x_2, \dots, x_m] = s_1(x_1) + s_2(x_2) + \dots + s_m(x_m).$$

However, we have only considered situations where ordinary least-squares is appropriate. This is the default, but there is scope for other distributional families as we saw for generalized linear models.

```
gam(formula, family=gaussian(), data=list(), ...)
```

Other families include `binomial`, `poisson` and `Gamma`, as well as a much larger list of possibilities outlined in the documentation for `family.mgcv`.

The `gam` formula involves smooth terms which need not be univariate. The default is the thin-plate spline (`tp`) which is a multivariate extension of the smoothing spline:

```
s(x1, x2, ..., k=10, bs="tp")
```

Here, `k` is the dimension of the basis used to represent the smooth terms. The maximum possible degrees of freedom allowed for each model term is $k-1$ which is one less than the basis dimension. The loss of one degree of freedom is due to an identifiability constraint on each smooth term. See Wood (2017) for more details on this point.

The `bs` argument indicates the basis to use in the smooth terms.

- “tp”: thin plate regression splines, default
- “ts”: as “tp” but with a small ridge penalty added to the smoothing penalty so that the whole term can be shrunk to zero
- “cr”: cubic regression splines
- “cs”: a shrinkage version of “cr”
- “cc”: cyclic cubic regression splines, penalized “cr” whose ends match
- “ps”: P-splines

9.7.4 GCV and UBRE

By default, the `gam` function will estimate effective degrees of freedom from the data, by minimizing the generalized cross-validation (GCV) or the Un-Biased Risk Estimator (UBRE) score. The formula for GCV used here, with smoother matrix \mathbf{A} , is

$$\nu_g = \frac{n\|\mathbf{y} - \mathbf{Ay}\|^2}{[n - \gamma \text{Tr}(\mathbf{A})]^2}$$

while the formula for UBRE is

$$\nu_u = \frac{1}{n} \|\mathbf{y} - \mathbf{Ay}\|^2 + \frac{2}{n} \sigma^2 \gamma \text{Tr}(\mathbf{A}) - \sigma^2$$

The argument *scale* controls whether to use UBRE or GCV:

- *scale* > 0: known scale, UBRE used;
- *scale* < 0: unknown scale, GCV used;
- *scale* = 0: UBRE for Poisson or binomial, otherwise GCV

9.7.5 Plots for gam

Diagnostic plots can be obtained using `gam.check()`. This function provides a QQ-plot for the residuals, as well as a plot of the residuals versus the fitted values on the linear scale (i.e. on the scale of the link function applied to the expected response, not the response itself), a histogram of the residuals and a plot of the responses versus the fitted values.

The generic `plot()` function can be used to visualize the smooth terms in the `gam` model:

```
plot(gam.object)
```

Solid lines or curves represent the estimated effects, and the dashed lines (or shaded regions) are roughly like 95% confidence limits but are actually Bayesian credible intervals. These are the 2.5% and 97.5% percentiles of the posterior cumulative distribution of the response, after placing prior multivariate normal distributions on the regression coefficients. Again, Wood (2017) provides much more detail.

Example

Consider the `litters` data set in the *DAAG* package. Use the `gam()` function to model the relationship between brain weight and body weight, accounting for litter size.

First, we fit the linear model using `lm()` and check the residuals to see if there are any problems with this model. The diagnostic plots for this model are provided in Figure 9.25. There is a hint of nonlinearity and nonnormality, so we might not have obtained the most appropriate model for these data.

```
library(DAAG)
litters.lm <- lm(brainwt ~ bodywt + lsize, data = litters)
par(mfrow=c(2, 2))
plot(litters.lm)
```

An allometric growth model might be more appropriate. That is, it may make more sense to work on the log scale. We can model `log(brainwt)` as a function of `log(bodywt)`, taking `lsize` into account. To account for possible nonlinearity, we include a smooth function of `lsize` in a call to `gam()`. The diagnostic plots are pictured in Figure 9.26. The QQ-plot suggests some mild outliers at the extremes, so there may still be a better model than this. Figure 9.27 allows for visualization of the smooth function of litter size. It is pretty clearly nonlinear.

```
library(mgcv)
par(mfrow=c(2, 2))
litters.log <- gam(log(brainwt) ~ log(bodywt) + s(lsize), data = litters)
gam.check(litters.log)
```

```
##
## Method: GCV Optimizer: magic
## Smoothing parameter selection converged after 6 iterations.
## The RMS GCV score gradient at convergence was 5.586775e-08 .
## The Hessian was positive definite.
## Model rank = 11 / 11
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'    edf  k-index p-value
## s(lsize) 9.00  3.14     1.17     0.74
```

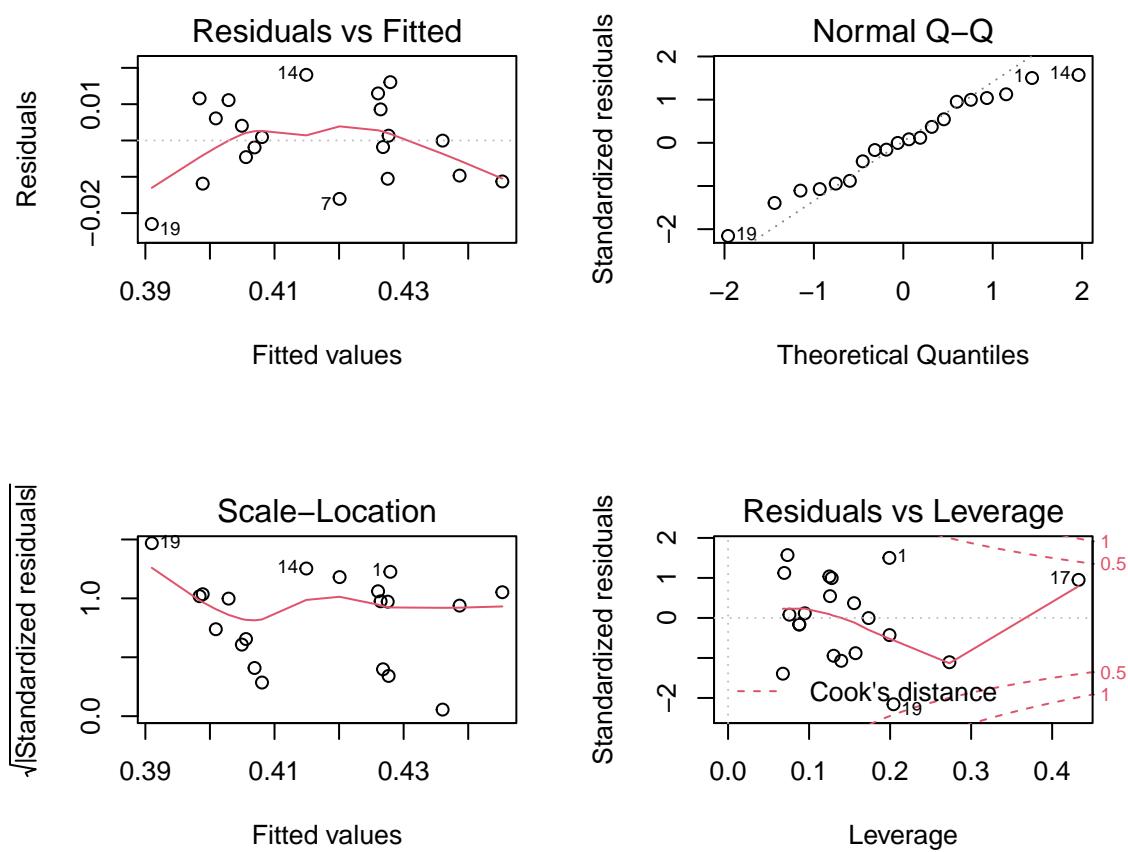


Figure 9.25: Diagnostic plots for linear model fit to the litters data set.

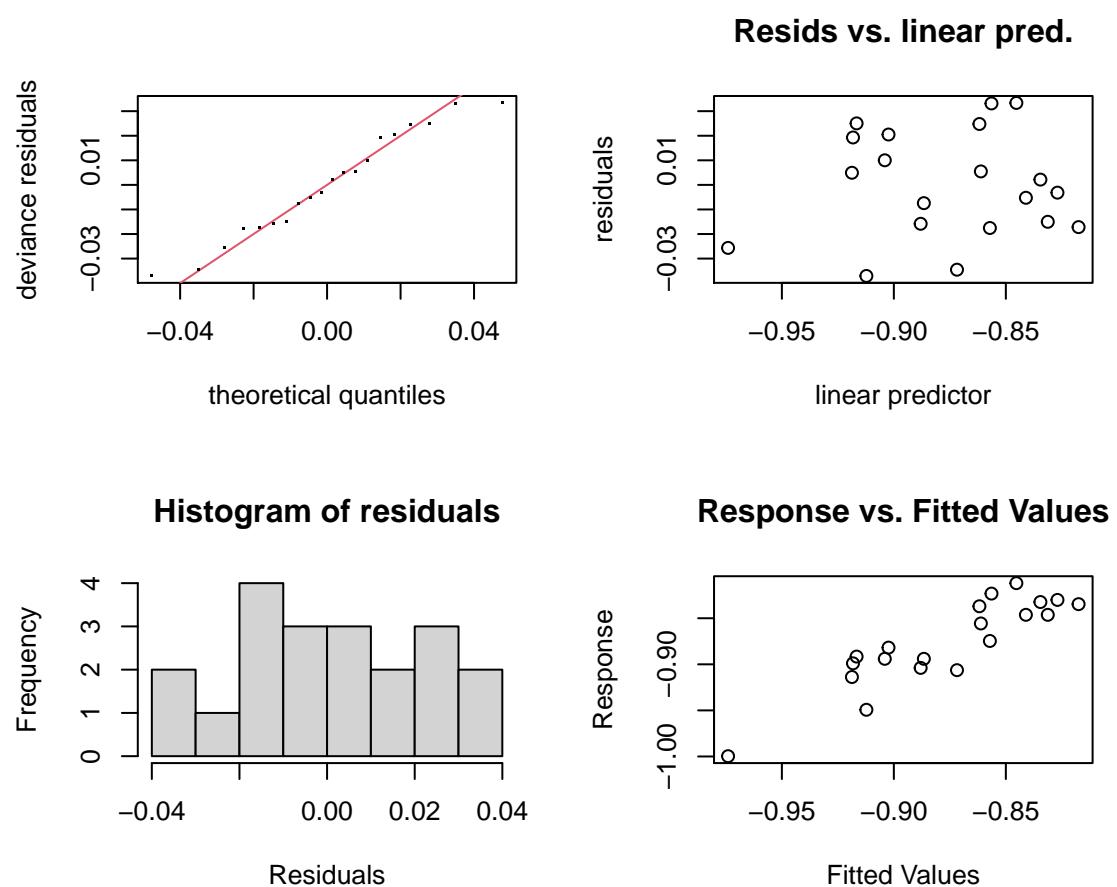


Figure 9.26: Diagnostic plots for allometric growth model fit to the litters data set.

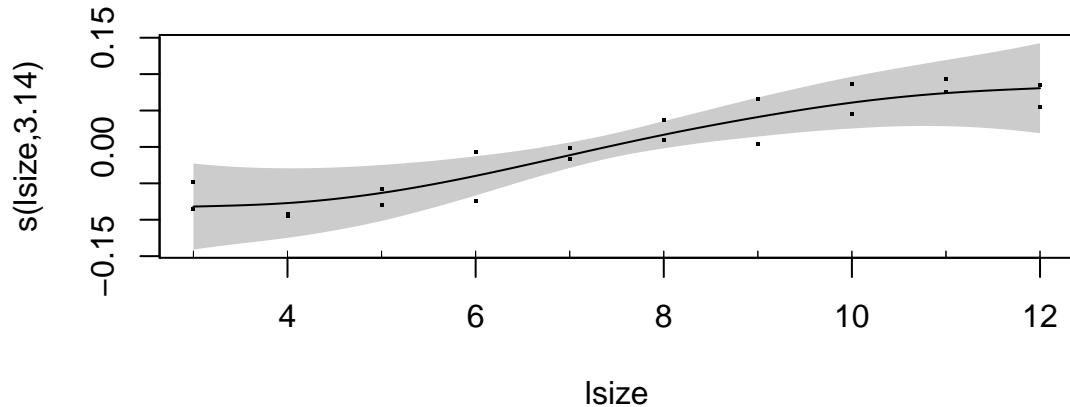


Figure 9.27: Smooth function of litter size in allometric growth model fit to the litters data set.

```
plot(litters.log, residuals=TRUE, cex=2, shade=TRUE)
```

The summary output from this model can be obtained in the usual way:

```
summary(litters.log)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(brainwt) ~ log(bodywt) + s(lsize)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.9525    0.2186 -8.933 2.33e-07
## log(bodywt)  0.5295    0.1075  4.925 0.000188
##
## Approximate significance of smooth terms:
##          edf Ref.df   F p-value
## s(lsize) 3.142 3.896 3.149 0.0487
##
## R-sq.(adj) = 0.729 Deviance explained = 78.8%
## GCV = 0.00079732 Scale est. = 0.00059235 n = 20
```

According to the summary output, the `bodywt` and `lsize` terms are highly and marginally significant, respectively, and the coefficient of `bodywt` is near 0.5. This implies that the brain weight is roughly proportional to the square root of the body weight.

It might be more appropriate to assume gamma distributed brain weights, since they must be positive. We can re-fit the model on the original scale for `brainwt` and `sqrt(bodywt)`, and with a smooth function term for `lsize`. The diagnostics for this model are plotted in Figure 9.28 and we do not see much of a difference from the previous model.

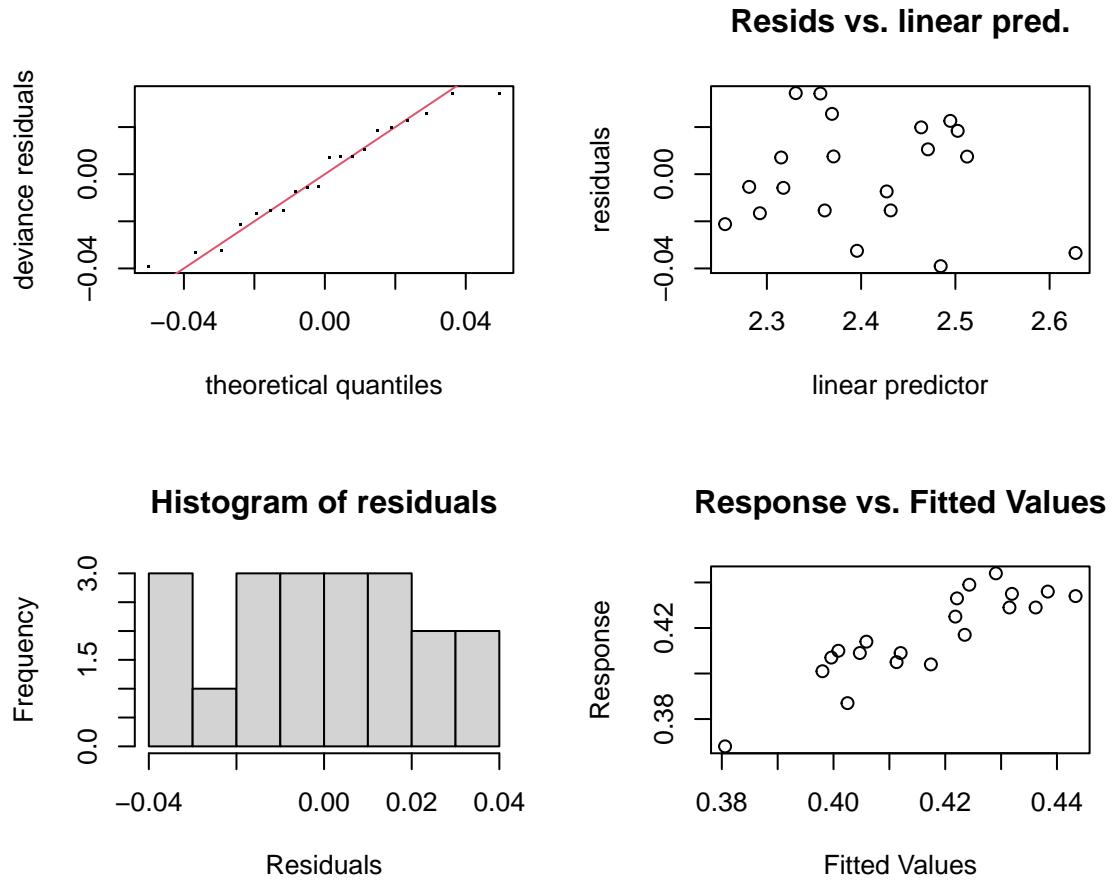


Figure 9.28: Diagnostic plots for Gamma model fit to the litters data set.

```
litters.Gamma<- gam(brainwt ~ sqrt(bodywt) + s(lsize), family = Gamma, data = litters)
par(mfrow=c(2,2))
gam.check(litters.Gamma)
```

```
##
## Method: GCV   Optimizer: outer newton
## full convergence after 5 iterations.
## Gradient range [-1.946197e-15, -1.946197e-15]
## (score 0.0008722104 & scale 0.0006440292).
## Hessian positive definite, eigenvalue range [5.394561e-05, 5.394561e-05].
## Model rank = 11 / 11
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'    edf  k-index p-value
## s(lsize) 9.00  3.21     1.19    0.75
```

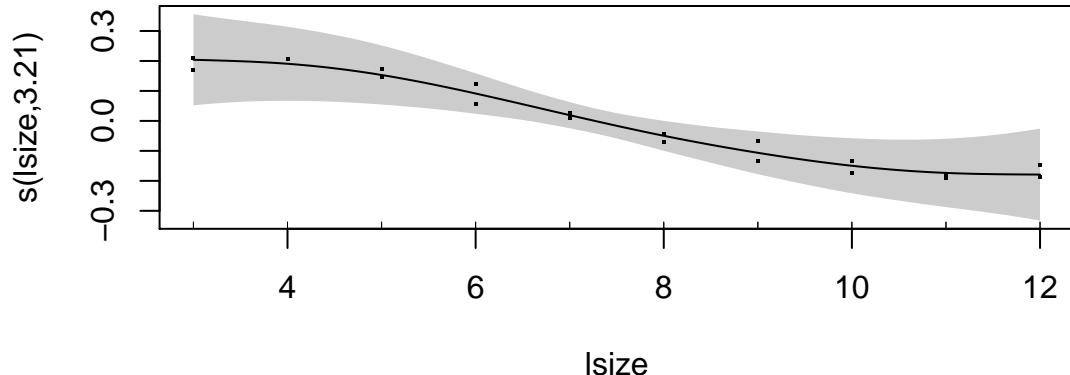


Figure 9.29: Visualizing the smooth function of litter size for the Gamma model fit to the litters data set.

```
plot(litters.Gamma, residuals=TRUE, cex=2, shade=TRUE)
```

Note that the default link function for `Gamma()` is the inverse while the link function for the Gaussian model is the identity function. This is why the smooth function of litter size pictured in Figure 9.29 decreases and the corresponding function pictured in Figure 9.27 increases.

The summary output from this model can again be obtained:

```
summary(litters.Gamma)

##
## Family: Gamma
## Link function: inverse
##
## Formula:
## brainwt ~ sqrt(bodywt) + s(lsize)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.9540    0.5553   8.921 2.45e-07
## sqrt(bodywt) -0.9199    0.1999  -4.601 0.000359
##
## Approximate significance of smooth terms:
##          edf Ref.df    F p-value
## s(lsize) 3.208 3.984 2.788 0.0663
##
## R-sq.(adj) =  0.699   Deviance explained = 76.7%
## GCV = 0.00087221  Scale est. = 0.00064403 n = 20
```

This time, the `bodywt` is significant with a negative coefficient, meaning that the inverse of brain weight is negatively related to the square root of the body weight. The smooth function of litter size is very marginally significant now. More data would really be required before one could say conclusively whether or not litter size really plays a role here.

9.7.6 Exercises

- Consider the data set

```
##      x  y
## 1 -2 -1
## 2 -1  1
## 3  0  2
## 4  1  4
## 5  2 -2
```

and the model

$$y = \beta_0 + \beta_1 x + \beta_2(x - 1)_+ + \varepsilon$$

where $\varepsilon \sim N(0, \sigma^2)$ and the observations are mutually independent.

- Estimate all of the parameters of the model (including σ^2).
- Calculate a 95% confidence interval for β_2 . Is there sufficient evidence to conclude that the knot at $x = 1$ is required?
- Assuming a boundary of -2 and 2, obtain formulae for three linear B -splines associated with a knot at $x = 1$. Sketch their graphs.
- Use the B -splines obtained in exercise 2 to fit the data of exercise 1. Compare the estimate of σ^2 with that obtained earlier.
- Assume that $B_{0,1}(x) \equiv 0$. Obtain formulae for the four quadratic B -splines associated with a knot at $x = 1$. Check that these functions are all differentiable at -2, 1 and 2. Sketch graphs of these functions.
- Check your results to exercises 2 and 4 by using the `matplot()` and `bs()` functions.
- Use the `gam()` function in the `mgcv` package to model the `geophones` data in the `DAAG` package.

- Try the default first:

```
geo.gam <- gam(thickness ~ s(distance), data=geophones)
plot(geo.gam)
points(geophones$distance, geophones$thickness-mean(geophones$thickness))
```

- Control the number of degrees of freedom in the `s()` term by using the `k` parameter. For example, see what happens with `s(distance, k=4)` and `s(distance, k=40)`.
- Now, set the knots explicitly as in

```
geo.gam <- gam(thickness ~ s(distance, k=7), knots=list(distance =
c(30, 35, 40, 50, 73, 76, 80)), data=geophones)
```

- Analyze the `earthquake` data in the `MPV` package, which describe earthquakes occurring from 1964 through 1985 having magnitude at least 5.8. The `depth` variable measures the focal depth, and `latitude` and `longitude` measure the geographic coordinates.
 - Use the `gam()` function in the `mgcv` package to construct a contour map which shows how depth changes with latitude and longitude. (You may use the default settings.) Interpret the results – are there locations on the planet where earthquakes tend to be at a greater depth?
 - Investigate the residuals for your fitted model. In a short paragraph, discuss any problems that you identify. If there are extreme residuals, identify them on your contour map, and give a basic interpretation of what you see.

9.8 Quantile regression - yet another approach to predictive modelling

The `earthquake` dataset contains measurements of latitude, longitude, focal depth and magnitude for all earthquakes having magnitude greater than 5.8 between 1964 and 1985.

The goal of this brief account is to use R. Koenker's (2020) `quantreg` package to fit a nonparametric model relating focal depth to latitude and longitude. The result is a set of surfaces, each corresponding to a given quantile, which can be plotted as a contour plot or a perspective plot. Thus, for example, we can see where the 10th percentile of the focal depth distribution is very deep within the Earth's crust.

9.8.1 Required libraries and mapping functions

The following code is needed in order to read in the data, load the required quantile regression package and some mapping packages. It also contains a map plotting function which allows us to center the world map on the Pacific Ocean which is the location of much of the earthquake activity.

```
library(maps)
library(quantreg)
library(tripack)
library(MatrixModels)
library(akima)
plot.map<- function(database,center,...){
  Obj <- map(database,...,plot=F)
  coord <- cbind(Obj[[1]],Obj[[2]])
  # split up the coordinates
  id <- rle(!is.na(coord[,1]))
  id <- matrix(c(1,cumsum(id$lengths)),ncol=2,byrow=T)
  polygons <- apply(id,1,function(i){coord[i[1]:i[2],]}) 
  # split up polygons that differ too much
  polygons <- lapply(polygons,function(x){
    x[,1] <- x[,1] + center
    x[,1] <- ifelse(x[,1]>180,x[,1]-360,x[,1])
    if(sum(diff(x[,1])>300,na.rm=T) >0){
      id <- x[,1] < 0
      x <- rbind(x[id,],c(NA,NA),x[!id,])
    }
    x
  })
  # reconstruct the object
  polygons <- do.call(rbind,polygons)
  Obj[[1]] <- polygons[,1]
  Obj[[2]] <- polygons[,2]
  map(Obj,...)
}
```

9.8.2 Implementation of the quantile regression function

The `rq()` function in the *quantreg* package allows us to fit linear quantile regression models. Its usage is very similar to the `lm()` function. A formula is needed, as well as a reference to a data frame. Many other optional arguments are available as well, including one for the `tau` parameter which specifies the particular quantile value. The default value is 0.5 for median regression. Consult the vignette *rq.pdf* for much more detailed information.²⁴

Since the earthquake focal depth surface is expected to be nonlinear, we would prefer to use a nonparametric regression method. The `rqss()` function can be used to compute something very much like a quantile version of a smoothing spline. A smooth term is included in the model formula using the `qss()` function. This function computes the smooth and is governed by a smoothing parameter λ .

For median regression of the earthquake data, we try three different values: a small value which undersmooths, a large value which oversmooths and a moderate value which is just about right. Figures 9.30, 9.31 and 9.32 show the output for these three model fits.

```
# median regression:
earthquake.rqss <- rqss(depth ~ qss(cbind(longitude, latitude),
  lambda=.15), data=earthquake)
plot(earthquake.rqss)
```

²⁴
library(quantreg)
vignette("rq")

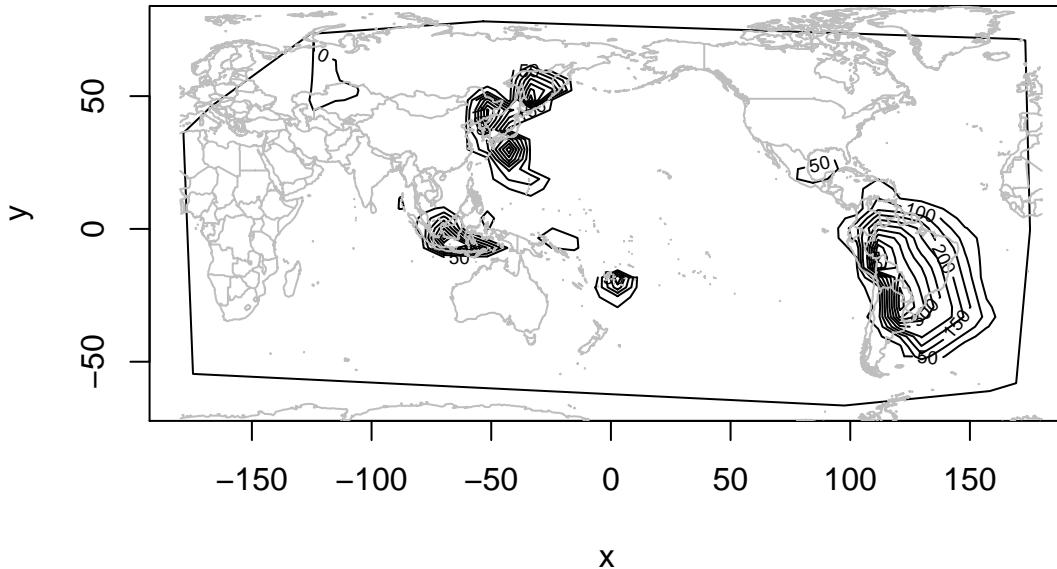


Figure 9.30: Visualization of median for earthquake focal depth, using lambda = .15.

```
plot.map("world", center=180, col="gray", mar=c(0,0,0,0),
         wrap=TRUE, add=TRUE)
```

```
earthquake.rqss <- rqss(depth ~ qss(cbind(longitude, latitude),
                           lambda=.6), data=earthquake)
plot(earthquake.rqss)
plot.map("world", center=180, col="gray", mar=c(0,0,0,0),
         wrap=TRUE, add=TRUE)
```

```
earthquake.rqss <- rqss(depth ~ qss(cbind(longitude, latitude),
                           lambda=.3), data=earthquake)
plot(earthquake.rqss)
plot.map("world", center=180, col="gray", mar=c(0,0,0,0),
         wrap=TRUE, add=TRUE)
```

These plots all show that there are various locations on the planet where the median focal depth for earthquakes is very low. By setting the `tau` parameter to 0.1, we can find the 10th percentile of the distribution of focal depths (as a function of latitude and longitude) as pictured in Figure 9.33.

```
# 10th percentile regression:
earthquake.rqss <- rqss(depth ~ qss(cbind(longitude, latitude),
                           lambda=.3), data=earthquake, tau=.1)
plot(earthquake.rqss)
plot.map("world", center=180, col="gray", mar=c(0,0,0,0),
         wrap=TRUE, add=TRUE)
```

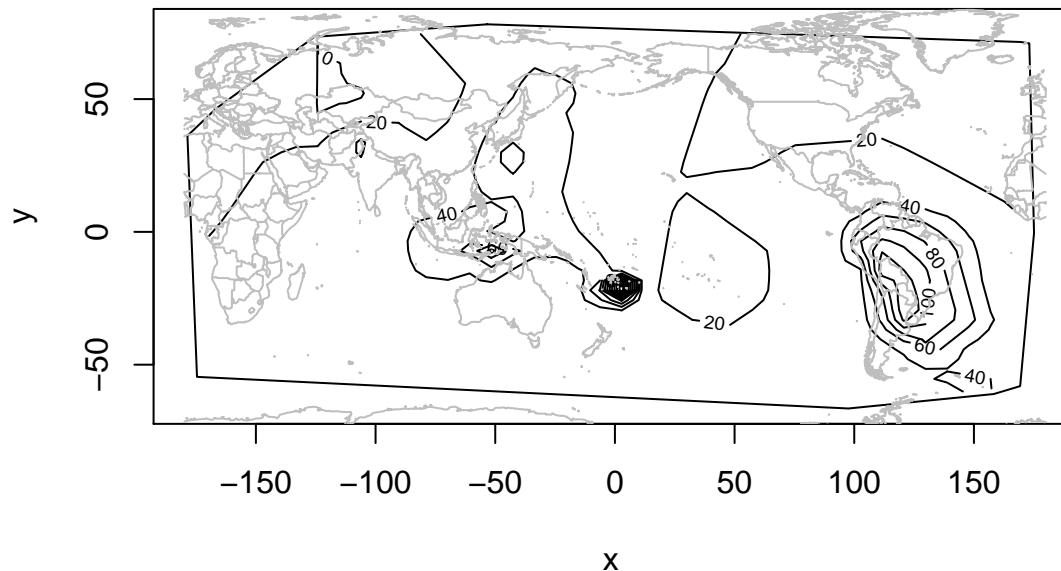


Figure 9.31: Visualization of median for earthquake focal depth, using $\lambda = .6$.

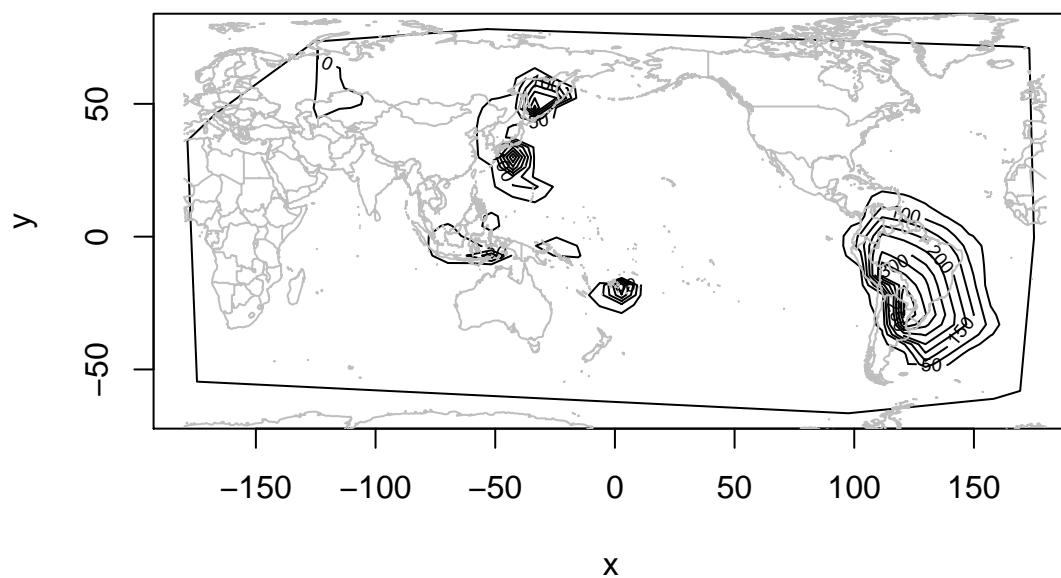


Figure 9.32: Visualization of median for earthquake focal depth, using $\lambda = .3$.

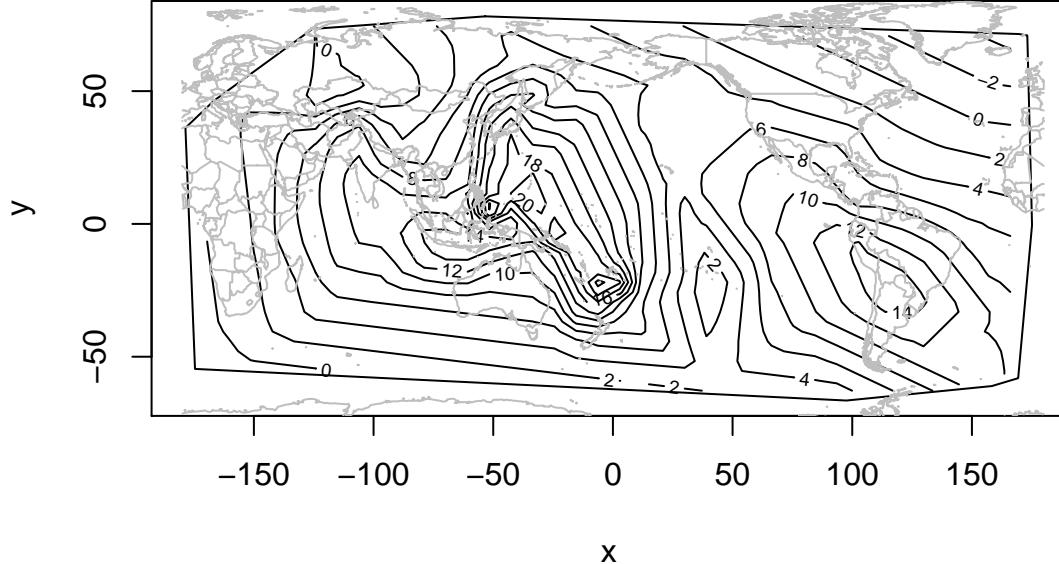


Figure 9.33: Visualization of 10th percentiles for earthquake focal depth.

The same general pattern of focal depth emerges, though there are some differences in the details. Similarly, we can use `tau = 0.9` to obtain the 90th percentile surface as pictured in Figure 9.34

```
# 90th percentile regression:
earthquake.rqss <- rqss(depth ~ qss(cbind(longitude, latitude),
  lambda=.3), data=earthquake, tau=.9)
plot(earthquake.rqss)
plot.map("world", center=180, col="gray", mar=c(0,0,0,0),
  wrap=TRUE, add=TRUE)
```

This plot shows that the 90th percentile of the focal distance distribution can be extremely low. Again, the general pattern is the same as for the other percentiles, but there are differences in the details.

9.9 Appendix - Penalized Spline Code (Eilers and Marx, 1996)

```
psplinereg <- function (x,y,LAMBDA,xl=min(x),xr=max(x),ndx=20,bdeg=3,pord=2,opt="AICC") {
# Eilers and Marx P-spline functions
# x, y = data
# lambda = smoothing parameter (optimize by gcv)
# ndx = number of intervals divided by the knots (see line 13 of Sect. 7)
# bdeg = b-spline degree
# pord = order of difference penalty
  require(splines)
  m <- length(y)
  scale <- (xr - xl)/length(x)
  B <- bspline(x, xl-scale, xr+scale, ndx, bdeg)
```

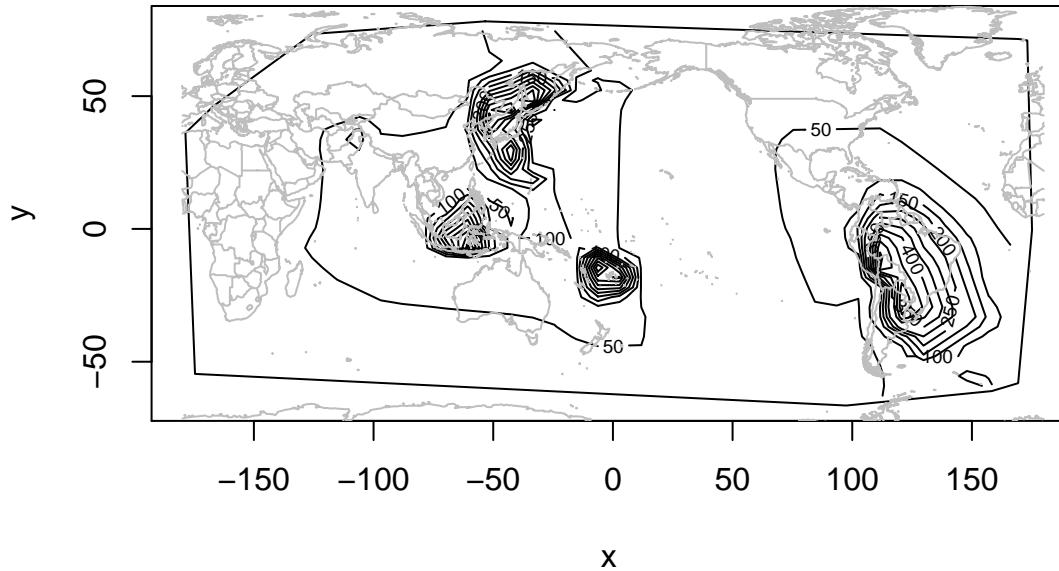


Figure 9.34: Visualization of 90th percentiles for earthquake focal depth.

```

D <- diag(ncol(B))
for (k in 1:pord) D <- diff(D)
yHAT <- NULL
trH <- NULL
gcv <- NULL
SSE <- NULL
for (lambda in LAMBDA) {
  a <- solve(t(B) %*% B + lambda*t(D) %*% D, t(B) %*% y)
  yhat <- B %*% a
  s <- sum((y-yhat)^2)
  Q <- solve(t(B) %*% B + lambda*t(D) %*% D)
  tmp <- sum(diag(Q %*% (t(B) %*% B)))
  gcv <- c(gcv, s/(nrow(B) - tmp)^2)
  trH <- c(trH, tmp)
  yHAT <- cbind(yHAT, yhat)
  SSE <- c(SSE, s)
}
sigma0 <- sqrt(m*min(gcv))
AIC <- SSE/sigma0^2 + 2*trH
AICC <- log(SSE/(m-trH)) + 1 + 2*(trH + 1)/(m - trH - 2)
if (opt=="GCV") {
  y <- yHAT[,gcv==min(gcv)]
} else {
  if (opt == "AIC") {
    y <- yHAT[,AIC==min(AIC)]
  } else {
    y <- yHAT[,AICC==min(AICC)]
  }
}
```

```
        }
    }
    list(x=x, y=y, gcv=gcv, trH=trH, AIC=AIC, AICC=AICC)
}

bspline <- function(x, xl, xr, ndx, bdeg) {
  dx <- (xr-xl)/ndx
  knots <- seq(xl - bdeg*dx, xr+bdeg*dx, by=dx)
  B <- spline.des(knots, x, bdeg+1, 0*x)$design
  B
}
```

10

An Introduction to Mixed-Effects Models

Multilevel models or linear mixed-effects models provide a modern approach to predictive modelling when there is a lot of structure in the data.

R packages that do this kind of modelling well are the nonlinear mixed-effects (`nlme`) package and the linear mixed-effects (S Version 4) (`lme4`) packages of Pinheiro and Bates:

```
library(nlme)
library(lme4)
```

When applying standard multiple regression methodology, it is assumed that the observations are independent. This assumption is often violated, because of natural groupings that occur within the data. Hierarchical or multi-level models can be used to handle such data. In this chapter, we will introduce the basic ideas behind handling mixed-effects modelling.

10.1 A one-factor random effects model

The conclusions to be drawn for a one-factor random effects model are identical to those to be drawn from a one-factor fixed effects model. However, it is instructive to go through the proper procedure.

10.1.1 Fitting the model

We begin with a simple example concerning a number of vibration measurements that were taken for 30 motors. Each motor had one of 5 different types of bearings installed. There were 6 measurements for each type of bearing.

```
library(MPV) # contains motor data frame
motor

##   Brand 1 Brand 2 Brand 3 Brand 4 Brand 5
## 1    13.1    16.3    13.7    15.7    13.5
## 2    15.0    15.7    13.9    13.7    13.4
## 3    14.0    17.2    12.4    14.4    13.2
## 4    14.4    14.9    13.8    16.0    12.7
## 5    14.0    14.4    14.9    13.9    13.4
## 6    11.6    17.2    13.3    14.7    12.3
```

Interest centers on whether there are differences in the mean vibration between brands. We might solve this problem using a one-factor analysis of variance, i.e. the `aov()` function. We need to `stack` the data frame first to convert to case-by-variable format

```
motors <- stack(motor)
motors

##   values     ind
## 1    13.1 Brand 1
## 2    15.0 Brand 1
## 3    14.0 Brand 1
## 4    14.4 Brand 1
## 5    14.0 Brand 1
## 6    11.6 Brand 1
## 7    16.3 Brand 2
## 8    15.7 Brand 2
```

```
## 9    17.2 Brand 2
## 10   14.9 Brand 2
## 11   14.4 Brand 2
## 12   17.2 Brand 2
## 13   13.7 Brand 3
## 14   13.9 Brand 3
## 15   12.4 Brand 3
## 16   13.8 Brand 3
## 17   14.9 Brand 3
## 18   13.3 Brand 3
## 19   15.7 Brand 4
## 20   13.7 Brand 4
## 21   14.4 Brand 4
## 22   16.0 Brand 4
## 23   13.9 Brand 4
## 24   14.7 Brand 4
## 25   13.5 Brand 5
## 26   13.4 Brand 5
## 27   13.2 Brand 5
## 28   12.7 Brand 5
## 29   13.4 Brand 5
## 30   12.3 Brand 5
```

```
names(motors) <- c("vibration", "bearingbrand")
summary(aov(vibration ~ bearingbrand, data= motors))

##           Df Sum Sq Mean Sq F value    Pr(>F)
## bearingbrand  4 30.86   7.714   8.444 0.000187 ***
## Residuals     25 22.84   0.914
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

However, since the bearing brands used could be considered to be a random sample from a larger population of brands, we should really consider brand as a random effect. The appropriate model is

$$y_{ij} = \beta_0 + B_i + \varepsilon_{ij}$$

where y_{ij} represents the vibration of the j th motor with the i th bearing brand. The constant β_0 is the mean vibration for such motors for the entire population of bearing brands. The random variable B_i represents the difference in vibration due to using the i th bearing brand. We assume that the B_i are independent normal random variables with mean 0 and variance σ_B^2 , and independent of the noise ε_{ij} which are also normal with mean 0 and variance σ^2 .

The `aov()` function handles basic random effects. For this example, we can treat the mean effects for the different bearing brands as random variables by using the `Error()` function inside the `aov()` call.

```
summary(aov(vibration ~ Error(bearingbrand), data= motors))

##
## Error: bearingbrand
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Residuals  4 30.86   7.714
##
## Error: Within
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Residuals 25 22.84   0.9135
```

For one-factor models, the calculations lead to the same numerical values; it is the interpretation of the model that is different. For the random-effects model, the treatment means are interpreted as random variables with a variance estimated by

the Mean-Squared-Error (here, 7.714) while for the fixed-effects model, the treatment means are not random, and the Mean-Squared-Error (here, .9135) is the variance of the noise.

Since $y_{ij} = \beta_0 + B_i + \varepsilon_{ij}$, the variance of \bar{y}_i is $\sigma_B^2 + \sigma^2/n$, where n is the number of measurements in the i th treatment group. (In this case, $n = 6$.) Therefore, the expected value of the MSR must be $n\sigma_B^2 + \sigma^2$. From the output, we estimate σ^2 to be 0.914 and $E[\text{MSR}]$ to be 7.714. We can solve for $\hat{\sigma}_B^2$ in

$$n\hat{\sigma}_B^2 + \hat{\sigma}^2 = \text{MSR}$$

to get $\hat{\sigma}_B^2 = 1.13$. Hence, $\hat{\sigma}_B = 1.065$.

We could also use the function `lme()` in the `nlme` package. This function will fit linear mixed-effects models using restricted maximum likelihood estimation (REML).

```
library(nlme)
motors.lme <- lme(fixed=vibration ~ 1, data=motors, random = ~1|bearingbrand)
```

The response is `vibration`. The `fixed` argument identifies the intercept (1) as the fixed effect. There is a single random effect for each unique value given by the variable `bearingbrand`.

```
summary(motors.lme)

## Linear mixed-effects model fit by REML
## Data: motors
##      AIC      BIC logLik
## 97.61081 101.7127 -45.8054
##
## Random effects:
## Formula: ~1 | bearingbrand
##          (Intercept) Residual
## StdDev:    1.064605 0.9557894
##
## Fixed effects: vibration ~ 1
##                  Value Std.Error DF t-value p-value
## (Intercept) 14.22333 0.5070777 25 28.04961     0
##
## Standardized Within-Group Residuals:
##      Min        Q1        Med        Q3        Max
## -2.24660837 -0.64349806  0.04942164  0.50877201  1.52176368
##
## Number of Observations: 30
## Number of Groups: 5
```

The output tells us that $\hat{\beta}_0 = 14.2$, $\hat{\sigma}_B = 1.06$, and $\hat{\sigma} = .956$. Compare these values with what was derived from the `aov` output. The log of the restricted likelihood is -45.8 at the estimate.

10.1.2 Assessing the model

Fitted values (i.e. estimates of the treatment means, in this case) are obtained as follows:

```
coef(motors.lme)

##             (Intercept)
## Brand 1      13.74728
## Brand 2      15.74551
## Brand 3      13.73259
## Brand 4      14.67294
## Brand 5      13.21834
```

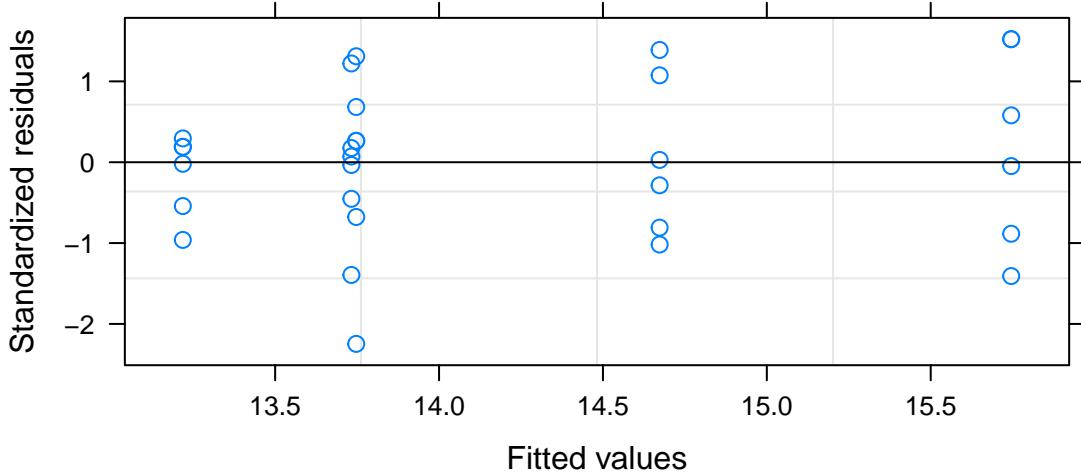


Figure 10.1: A plot of the standardized residuals versus the fitted values for the motors data.

These are estimates of $\beta_0 + b_i$, for $i = 1, 2, \dots, 5$.

Residuals are then calculated from $y_{ij} - \hat{\beta}_0 - \hat{b}_i$. A simple way of standardizing is to divide by the residual standard deviation. The `plot.lme()` function plots these standardized residuals against the fitted values:

```
plot(motors.lme)
```

On plots like the one in Figure 10.1, we would usually look for patterns of increasing or decreasing variance. There is a hint of an increase, though the third group does not fit this pattern. One possibility would be to repeat the analysis on the square root scale.

10.1.3 Confidence intervals

We have estimated three parameters: β_0 , σ_b , and σ . Confidence intervals for these respective parameters can be obtained from the `intervals()` function:

```
intervals(motors.lme)

## Approximate 95% confidence intervals
##
## Fixed effects:
##           lower     est.     upper
## (Intercept) 13.17899 14.22333 15.26768
## attr(),"label")
## [1] "Fixed effects:"
##
## Random Effects:
##   Level: bearingbrand
##           lower     est.     upper
## sd((Intercept)) 0.4820061 1.064605 2.351388
##
## Within-group standard error:
##           lower     est.     upper
## 0.7246515 0.9557894 1.2606520
```

10.2 Randomized block designs

The calorie content of six different brands of orange juice were determined by three different machines. The numbers below are the determination in calories per 6 fluid ounces. We are interested in knowing whether the caloric content differs for the different brands, but we also would like to take into account differences in the machines' ability to measure caloric content.

```
obj <- read.table("obj.txt", header = TRUE)
obj

##   MACHINE A   B   C   D   E   F
## 1       M1 89  97  92 105 100  91
## 2       M1 94  96  94 101 103  92
## 3       M2 92 101  94 110 100  95
## 4       M2 90 100  98 106 104  99
## 5       M3 90  98  94 109  99  94
## 6       M3 94  92  96 107  97  98
```

To set up the data set so that we can apply `lme()`, do the following:

```
obj.df <- stack(obj[,-1])
obj.df$machine <- rep(obj$MACHINE, 6)
names(obj.df) <- c("calories", "brand", "machine")
```

We are interested in differences between the six brands, so we treat `brand` as a fixed factor. We would like to be able to generalize our results about machines to the larger population of such machines, so `machine` is a random factor.

The model we will fit is

$$y_{ijk} = \beta_0 + \beta_i + B_j + \varepsilon_{ijk}, \quad i = 1, 2, \dots, 6; j = 1, 2, 3; k = 1, 2$$

where $\text{Var}(B) = \sigma_B^2$ and $\text{Var}(\varepsilon) = \sigma^2$. To fit the model, we try:

```
obj.lme <- lme(calories ~ brand, data = obj.df, random = ~ 1 | machine)
summary(obj.lme)

## Linear mixed-effects model fit by REML
## Data: obj.df
##      AIC      BIC      logLik
## 170.5107 181.7203 -77.25535
##
## Random effects:
## Formula: ~1 | machine
##          (Intercept) Residual
## StdDev:    1.272415 2.536246
##
## Fixed effects: calories ~ brand
##                Value Std.Error DF t-value p-value
## (Intercept) 91.50000 1.269555 28 72.07249 0.0000
## brandB      5.83333 1.464302 28  3.98370 0.0004
## brandC      3.16667 1.464302 28  2.16258 0.0393
## brandD     14.83333 1.464302 28 10.12997 0.0000
## brandE      9.00000 1.464302 28  6.14627 0.0000
## brandF      3.33333 1.464302 28  2.27640 0.0307
##
## Correlation:
##          (Intr) brandB brandC brandD brandE
## brandB -0.577
## brandC -0.577  0.500
## brandD -0.577  0.500  0.500
## brandE -0.577  0.500  0.500  0.500
## brandF -0.577  0.500  0.500  0.500  0.500
```

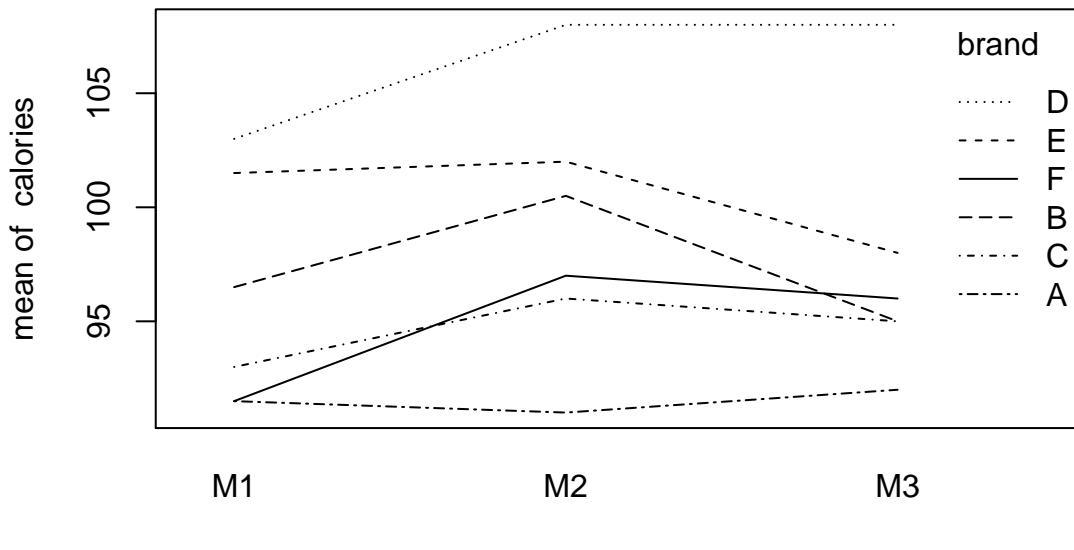


Figure 10.2: An interaction plot for energy measurements in calories for six brands of orange juice using three different machines.

```
## 
## Standardized Within-Group Residuals:
##      Min       Q1       Med       Q3      Max
## -2.0452492 -0.6062125 -0.1225348  0.8699338  1.3888857
## 
## Number of Observations: 36
## Number of Groups: 3
```

From the random effects output, we see that $\hat{\sigma}_B = 1.27$ and $\hat{\sigma} = 2.54$. Brand A is the baseline in this case so that $\hat{\beta}_1 = 91.5$. The estimates of β_2 through β_6 can be obtained by adding the `brand*` output from the Fixed effects slot to the (Intercept) value as in

```
fixef(oj.lme)[-1] + rep(fixef(oj.lme)[1], 5)

##   brandB   brandC   brandD   brandE   brandF
## 97.33333 94.66667 106.33333 100.50000 94.83333
```

Because the experiment has been replicated, we may also test for interaction effects between the blocking factor (`machine`) and the treatment (`brand`). Figure 10.2 displays such a plot for the orange juice data obtained from:

```
interaction.plot(machine, brand, calories)
```

If there are no interactions present, the curves should be roughly parallel. The plot in Figure 10.2 is not inconsistent with this.

10.3 Modelling growth curves

Simple regression can be applied to the estimation of a growth curve when the data are all taken from a single individual. Random effects models give an effective way of handling growth curves for several individuals.

10.3.1 An analysis of covariance model

The following data were collected at the University Hospital in London, Ontario. They concern measurements of tumours growing in three different mice at different times following injection of a known carcinogen. Because we would expect an allometric growth model to be appropriate for modelling volume as a function of time, we will work on the log scale.

```
tumours <- read.table("tumours.txt", header=TRUE)
tumours

##   mouse time volume  logtime  logvolume
## 1      8    11   0.07 2.397895 -2.65926004
## 2      8    13   0.18 2.564949 -1.71479843
## 3      8    15   0.27 2.708050 -1.30933332
## 4      8    19   0.61 2.944439 -0.49429632
## 5      8    21   1.29 3.044522  0.25464222
## 6      8    23   1.60 3.135494  0.47000363
## 7     10    11   0.70 2.397895 -0.35667494
## 8     10    13   0.27 2.564949 -1.30933332
## 9     10    15   0.52 2.708050 -0.65392647
## 10    10    19   0.80 2.944439 -0.22314355
## 11    10    21   1.15 3.044522  0.13976194
## 12    10    23   1.77 3.135494  0.57097955
## 13    11    11   0.14 2.397895 -1.96611286
## 14    11    13   0.22 2.564949 -1.51412773
## 15    11    16   0.52 2.772589 -0.65392647
## 16    11    19   1.02 2.944439  0.01980263
## 17    11    21   1.77 3.044522  0.57097955
## 18    11    23   2.35 3.135494  0.85441533
```

The allometric growth model is

$$\log v = \beta_0 + \beta_1 \log t + \varepsilon$$

where v denotes volume and t denotes time. The usual assumptions are made about the error term. This model would be appropriate if the measurements were taken on a single mouse. Here, we have measurements on three mice. We would not expect the slope and intercept to be the same for each mouse, since there are very complex interactions that take place within a biological system. The slope and intercept values for each mouse should be viewed as randomly sampled from a larger population. Instead of performing three separate regression analyses, we can gain efficiency by pooling the information, using a mixed-effects model.

First, we will fit a model in which the mice share a common slope, but where the intercept is allowed to vary between mice. Thus, we wish to fit

$$\log v_{ij} = \beta_0 + B_i + \beta_1 x_{ij} + \varepsilon_{ij}$$

where $\beta_0 + B_i$ is the intercept for the i th mouse. B_i is assumed to have a normal distribution with mean 0 and variance σ_B^2 .

To fit this model using `lme()`, type the following

```
tumours.lme <- lme(logvolume ~ logtime, data=tumours, random = ~1|mouse)
summary(tumours.lme)

## Linear mixed-effects model fit by REML
## Data: tumours
##      AIC      BIC      logLik
## 31.9761 35.06646 -11.98805
##
## Random effects:
## Formula: ~1 | mouse
##             (Intercept) Residual
## StdDev:  0.2548067 0.43292
##
## Fixed effects: logvolume ~ logtime
##                 Value Std.Error DF   t-value p-value
```

```

## (Intercept) -9.681728 1.1009687 14 -8.793826      0
## logtime      3.256588 0.3875802 14  8.402357      0
## Correlation:
##          (Intr)
## logtime -0.987
##
## Standardized Within-Group Residuals:
##      Min       Q1       Med       Q3       Max
## -1.2817179 -0.4705203 -0.3271385  0.3314163  3.0958873
##
## Number of Observations: 18
## Number of Groups: 3

```

From this output, we can see that $\hat{\sigma}_B = .25$, and $\hat{\sigma} = .43$. The overall intercept is -9.68 and the estimated slope is 3.25. We can update this model to include a random slope using the `update` function:

```

tumours.lme1 <- update(tumours.lme, random = ~ logtime|mouse)
anova(tumours.lme, tumours.lme1)

##           Model df     AIC     BIC   logLik   Test  L.Ratio
## tumours.lme     1 4 31.97610 35.06646 -11.988050
## tumours.lme1    2 6 27.82263 32.45816 -7.911316 1 vs 2 8.153469
##             p-value
## tumours.lme
## tumours.lme1  0.017

```

The new model is

$$\log v_{ij} = \beta_0 + B_i + \beta_1 x_{ij} + B'_i x_{ij} + \varepsilon_{ij}.$$

From the output, we see that there is evidence that the slope is different from mouse to mouse. We can obtain the parameter estimates using the `summary()` function.

```

summary(tumours.lme1)

## Linear mixed-effects model fit by REML
## Data: tumours
##        AIC     BIC   logLik
## 27.82263 32.45816 -7.911316
##
## Random effects:
## Formula: ~logtime | mouse
## Structure: General positive-definite, Log-Cholesky parametrization
##            StdDev   Corr
## (Intercept) 3.8839748 (Intr)
## logtime     1.2952657 -0.999
## Residual    0.2960641
##
## Fixed effects: logvolume ~ logtime
##                 Value Std.Error DF t-value p-value
## (Intercept) -9.698554 2.3633229 14 -4.103778 0.0011
## logtime      3.262055 0.7934118 14  4.111428 0.0011
## Correlation:
##          (Intr)
## logtime -0.999
##
## Standardized Within-Group Residuals:
##      Min       Q1       Med       Q3       Max
## -1.93118858 -0.41226469 -0.01650751  0.35302167  2.34274267

```

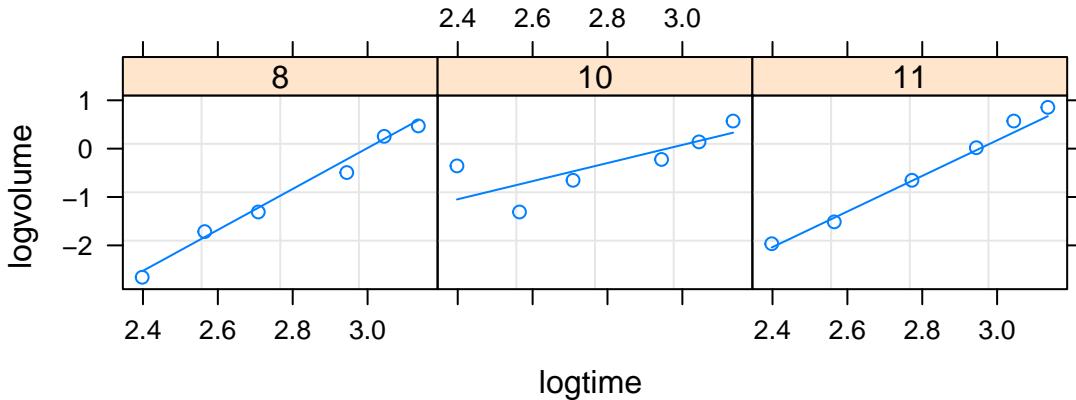


Figure 10.3: The original mouse tumour volume data with the fitted growth curves overlaid. The growth curves were estimated using restricted maximum likelihood for an analysis of covariance model.

```
## 
## Number of Observations: 18
## Number of Groups: 3
```

10.3.2 Visualizing the fitted data

The `augPred()` function in `nlme` can be used to obtain predicted values of the response. This function is easiest to apply when the data have been grouped using the `groupedData()` function:¹

```
tumours <- groupedData(logvolume ~ logtime | mouse, data=tumours)
```

The effect of this operation is associate the simple regression formula with each group as indicated by the `mouse` variable. Figure 10.3 shows the fitted model together with the original data. The following command produces the plot:

```
plot(augPred(tumours.lme1), grid=TRUE)
```

From the plot, we can see that the growth curves for each mouse have different intercepts and slopes. The 8th and 11th curves fit the data very closely, but the 10th curve exhibits an outlier. Upon further investigation, it was noted that the first volume observation for mouse 10 should have been recorded as 0.14 instead of 0.7. The following lines give the corrected analysis:

```
tumours$volume[7] <- .14
tumours$logvolume[7] <- log(.14)
tumours.lme10 <- lme(logvolume ~ logtime, data=tumours, random =
~1|mouse)
tumours.lme1 <- lme(logvolume ~ logtime, data=tumours, random =
~logtime|mouse)
anova(tumours.lme10, tumours.lme1)
```

¹The `augPred()` function computes predicted values for data with a grouped data structure. The first argument is an object which can be produced, for example, by `lme()`. The second argument `primary` is used to specify a covariate at which the predictions are to be based; if none is specified, then the `augPred()` uses the grouping structure of the data frame used to generate the object given in the first argument. In our example, the grouping structure of `tumours` is not specified until the `groupedData()` function is applied to it.

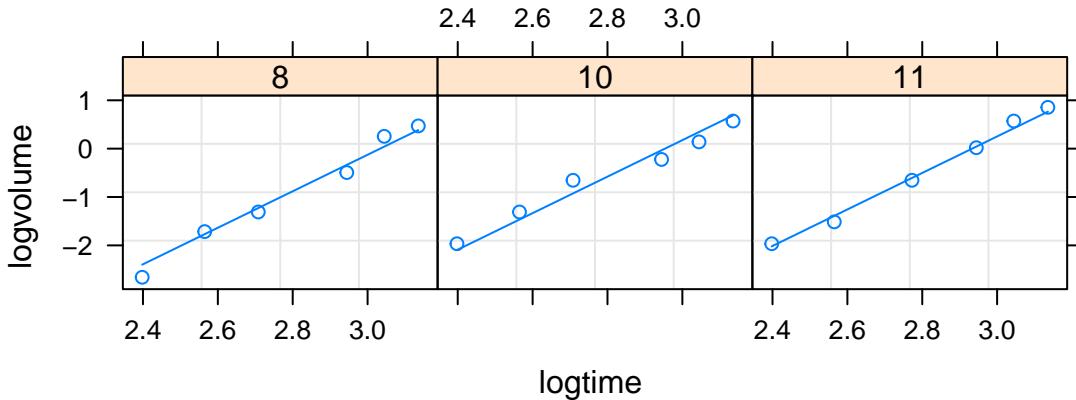


Figure 10.4: The corrected mouse tumour volume data with the fitted growth curves overlaid. The growth curves were estimated using restricted maximum likelihood for a model with random intercept but fixed slope.

```
##               Model df      AIC      BIC    logLik   Test L.Ratio
## tumours.lme10     1  4 4.583811 7.674166 1.708094
## tumours.lme1     2  6 4.243070 8.878603 3.878465 1 vs 2 4.340741
##               p-value
## tumours.lme10
## tumours.lme1  0.1141
```

There is no longer overwhelming evidence for differing slopes. Figure 10.4 displays the fitted data.

```
plot(augPred(tumours.lme10), grid=TRUE)
```

We can read off the coefficient estimates for the three lines as follows:

```
coef(tumours.lme10)

##      (Intercept)  logtime
## 8     -11.44784 3.773825
## 10    -11.14563 3.773825
## 11    -11.06911 3.773825
```

Multilevel modelling of the orange trees

In the orange tree data frame, we have multiple measurements of circumference and age for several trees. We will use `lmer()` in the `lme4` package to fit simultaneous linear regressions relating age to circumference, for each tree.

We need to standardize first in order for the software to run: subtract off each variable mean and divide by the variable standard deviation. Then run the software. Because we have centered the data, we do not include intercept terms.

```
Orange$Age <- (Orange$age -
  mean(Orange$age)) / sd(Orange$age)
Orange$Circ <- (Orange$circumference - mean(Orange$circumference)) /
  sd(Orange$circumference)
Orange$tree <- factor(Orange$Tree, ordered = FALSE) # remove ordering of factor
Orange.lmer <- lmer(Circ ~ Age + (Age-1|tree)-1, data = Orange)
```

```
summary(Orange.lmer)

## Linear mixed model fit by REML ['lmerMod']
## Formula: Circ ~ Age + (Age - 1 | tree) - 1
##   Data: Orange
##
## REML criterion at convergence: 37.7
##
## Scaled residuals:
##    Min     1Q Median     3Q    Max
## -1.7029 -0.7426 -0.2130  0.7225  1.8704
##
## Random effects:
##   Groups   Name Variance Std.Dev.
##   tree     Age   0.02339  0.1529
##   Residual      0.14677  0.3831
## Number of obs: 35, groups: tree, 5
##
## Fixed effects:
##   Estimate Std. Error t value
##   Age     0.91352   0.09484  9.632
```

The coefficient of Age appears have a mean near 0.9, with a variance of 0.023. The error variance is estimated at 0.147.

10.4 Exercises

1. For the orange juice energy data, we might also model brand as a random factor. Use the `lmer()` function to re-do the analysis. How does this new approach affect the fixed factor estimates and standard errors?
2. Can you use `aov()` to analyze the model that we fit for the orange juice energy data? If so,
 - (a) derive an estimate of the standard deviation of the machine effect from the various mean squares that are supplied in the ANOVA table.
 - (b) assume brand is also a random factor, and use `Error(brand+machine)` in your call to `aov()`. You can informally check the F ratios for brand and for machine by dividing by the residual Mean Square for Within. Are there machine effects as well as brand effects?
 - (c) what happens if you use `Error(brand*machine)` in the call to `aov()`? Is the interaction between brand and machine significant?
3. Use `lme()` to re-do the analysis of the `Orange` data. Do you need to standardize the data beforehand?
4. Experiment with the following code, trying different sets of simulated and/or real data.

```
# Loading the nlme package
library(nlme)

# Simulate some regression data
n <- 50
x <- sort(runif(n))
y <- sin(x*7) + rnorm(n, sd=.05)

# Fitting a penalized spline to the simulated data using lme
y <- y[order(x)] # sort data in ascending x order
x <- sort(x)
n <- length(y)
knots <- x[seq(2,n,2)]
Z <- outer(x, knots, function(x,y) abs(x-y))
xy <- data.frame(x=x, y=y)
```

```

xy$r <- rep(1,n)
xy <- data.frame(xy, Z)
xy.new <- groupedData(y ~ xy[,-c(2,3)]|r, data=xy)
xy.lme <- lme(y ~ 1 + x, random=pdIdent(as.formula(paste(~
  paste(names(xy.new[-c(1,2,3)]), collapse="+"), -1))), data=xy.new)
xy.newdata <- data.frame(x=seq(min(x),max(x),length=401),
  outer(seq(min(x), max(x), length=401), knots, function(x,y) abs(x-y)))
xy.newdata$r <- rep(1,401)
xy.pspline <- list(x=xy.newdata$x, y=predict(xy.lme, newdata=xy.newdata))

# Plotting the output
plot(x,y)
lines(xy.pspline)

```

5. Consider the data in the data frame `burnRate` contained in the `sharpData` package. This gives the proportion of cylinders of wood that burn in a given time under various wind speed and other atmospheric conditions. The objective is to relate `proportionBurned` to `densityRatio`.

- (a) On physical grounds, it can be argued that the proportion burned should be 0 when the density ratio is 0. Fit a simple regression through the origin model to the data, and identify any deficiencies in the fitted model.
- (b) As we have seen in earlier chapters, when dealing with proportions, a transformation such as a logit or probit is more appropriate. Construct a new variable called `logitBurned` by taking the logit transformation of `proportionBurned`. Then fit a simple regression model relating this new variable to `densityRatio`. Again, identify the deficiencies in the model.
- (c) Construct the following *lattice* plot:

```

xyplot(proportionBurned ~ densityRatio|species, groups=diameter,
       data = burnRate, col=1:2, pch=16, type=c("p", "smooth"), span=1.25)

```

What does this plot tell you about how the experiment was run and the variation in `proportionBurned` relates to other variables that were recorded?

- (d) In this part, we will use the `lmer()` function in the `lme4` package to fit a model that captures the behaviour displayed in the above plot and that reflects the fact that the diameters and species used in the experiment can be viewed as a random sample from a larger collection of possible diameters and tree species. Use the logit of the proportion burned as the response variable. The following code provides one possible model to consider.

```

burn.lmer <- lmer(log(proportionBurned/(1-proportionBurned)) ~
  densityRatio - 1 + (densityRatio-1|species:diameter), data = burnRate)

```

Write out the model equation, run the code, and identify the mean and variance estimates of the various components of the model.

- (e) Construct synthetic plots (similar to those discussed by Brillinger et al, 2006) which are based on data simulated from the fitted model and which can be used to compare with the actual data (via the graph constructed at (c)). Possible code to be run several times is:

```

y <- simulate(burn.lmer)$sim_1
xyplot(exp(y)/(exp(y)+1) ~ densityRatio|species, groups=diameter,
       data = burnRate, col=1:2, pch=16, type=c("p", "smooth"), span=1.25)

```

Contrast the patterns that you see on such plots with the patterns seen in the plot from part (c).

- (f) A plot that ignores the contributions of species and diameter is constructed from:

```

xyplot(exp(y)/(exp(y)+1) ~ densityRatio, data = burnRate,
       col=1:2, pch=16, type=c("p", "smooth"), span=1.25)

```

Compare such synthetic plots with the corresponding plot for the actual data. Do you think the model is capturing the variation in the data appropriately?

- (g) Construct synthetic plots as in parts (e) and (f) for the model you developed in part (a). Do these plots indicate any failings of the model?

A

Probability Concepts

A.1 Basic features of a probability model

A.1.1 Properties of density functions

Probability density is always nonnegative, and the area under the density curve is exactly 1.0. That is,

$$f(x) \geq 0, \text{ for all } x$$

and

$$\int_{-\infty}^{\infty} f(x)dx = 1.$$

All probability density functions have these two properties.

A.1.2 Calculation of probabilities using the probability density function

The probability that a random variable X with density function $f(x)$ takes a value in an interval $[a, b]$ is calculated as

$$P(a \leq X \leq b) = \int_a^b f(x)dx.$$

Such probabilities are also expressed in terms of the cumulative distribution function:

$$F(y) = P(X \leq y) = \int_{-\infty}^y f(x)dx.$$

Note that the probability density function can be recovered from the cumulative distribution function by differentiation:

$$f(x) = F'(x).$$

Example: probabilities of large error proportions

For example, we may interested in knowing whether an observed error proportion v is unusually large, using the model $f(x) = (\alpha + 1)x^\alpha$ that we have discussed earlier. We can check this by calculating the probability that the error proportion X exceeds v . If we know that the value of α is -0.5 , then

$$P(X > v) = \int_v^{\infty} f(x)dx = \int_v^1 0.5x^{-0.5}dx = 1 - v^{0.5}.$$

Note that we are assuming $v \in (0, 1)$ here. If $v \geq 1$, the probability would be 0.

The cumulative distribution function is

$$F(y) = \int_0^y 0.5x^{-0.5}dx = y^{0.5} \quad y \in [0, 1].$$

A.1.3 Uniform distribution

A simple example of a probability density function is the uniform distribution:

$$f(x) = \frac{1}{b-a}, \quad a < x < b$$

and $f(x) = 0$, otherwise. The parameters a and b specify the range of the distribution. An important special case is the uniform(0,1) distribution for which $a = 0$ and $b = 1$; in this case, the density function takes the value 1 on the interval $(0, 1)$ and is 0, elsewhere.

The `dunif()` function in R can be used to calculate uniform density function values. For example,

```
U <- dunif(3.5, min = 1, max = 6)
```

evaluates the uniform density at $x = 3.5$, when $a = 1$ and $b = 6$: $f(x) = 0.2$.

A.1.4 Exponential distribution

As another example of a probability density function, consider the exponential distribution, often taken as a very simple model for the lifetime distribution of, say, an electronic component:

$$f(x) = \lambda e^{-\lambda x}, \quad x > 0$$

and $f(x) = 0$, otherwise. The parameter λ is often interpreted as the failure rate.

The `dexp()` function in R can be used to calculate exponential density function values. For example, if the failure rate is 0.1 per hour, the density at 6 hours can be calculated using

```
dexp(6, rate = 0.1)
## [1] 0.05488116
```

To calculate the probability density function values at the sequence of values from 0.8 through 1.7, at increments of 0.05, when the rate is 1.5, we could proceed as follows:

```
x <- seq(0.8, 1.7, by = 0.05)
dexp(x, rate = 1.5)

## [1] 0.4517913 0.4191465 0.3888604 0.3607627 0.3346952 0.3105113
## [7] 0.2880749 0.2672596 0.2479483 0.2300325 0.2134111 0.1979908
## [13] 0.1836846 0.1704122 0.1580988 0.1466752 0.1360769 0.1262445
## [19] 0.1171225
```

The result of the following code is shown in Figure A.1.

```
curve(dexp(x, rate = 0.1), from = -1, to = 70, ylab = "f(x)")
```

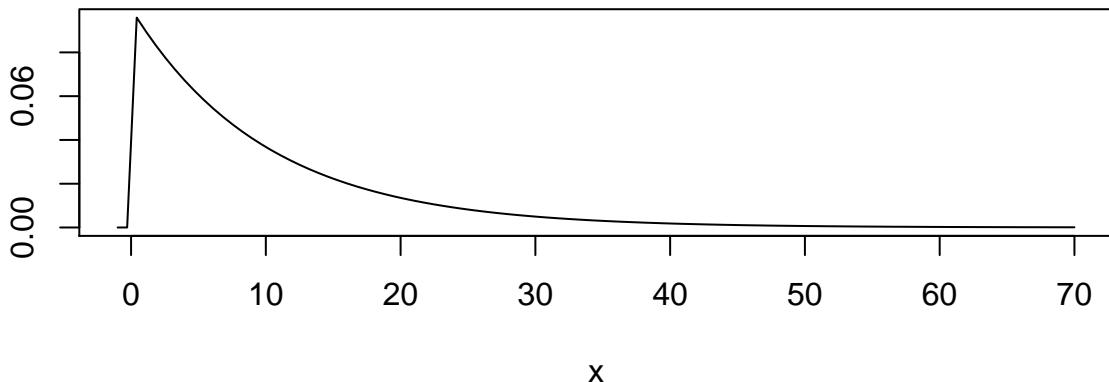


Figure A.1: Graph of exponential probability density function from $x = -1$ to $x = 70$, when failure rate is 0.1.

A.2 Simulation of random variates

A.2.1 Simulating and transforming uniform random numbers

The `runif()` function can be used to simulate samples of independent uniformly distributed numbers, as displayed in Figure A.2.

```
U <- runif(300)
hist(U)
```

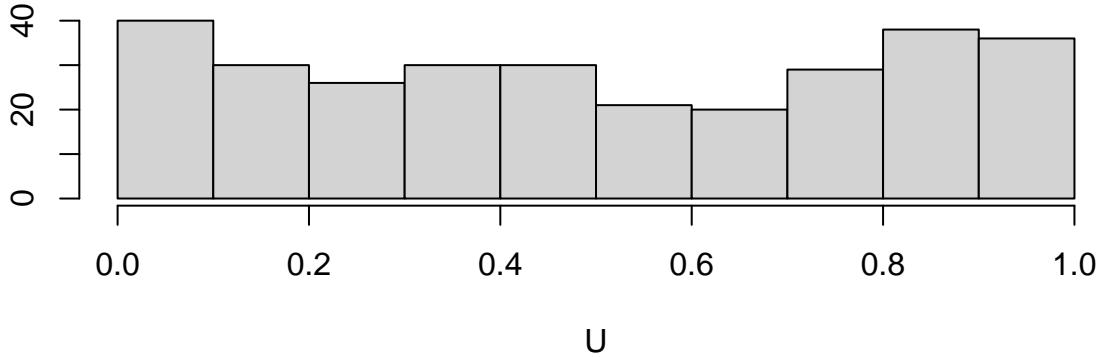


Figure A.2: Histogram of sample of 300 independent uniform numbers on (0,1).

We can convert uniform random¹ numbers so that they follow different kinds of distributions. A popular method of simulation is to invert the target probability distribution function. In other words, suppose the distribution function of interest is $F(x)$, and we have a uniform(0,1) numbers, U .

Our goal is to construct a value X for which, for every possible value of x ,

$$P(X \leq x) = F(x),$$

but since

$$P(U \leq F(x)) = F(x)$$

(note that $F(x) \in (0, 1)$ and U is uniform on $(0, 1)$), we should choose X so that $F(X) = U$. In other words, X is the U th quantile of the $F(x)$ distribution.

The exponential case serves as a very simple example. The code to compute the exponential quantiles corresponding to the uniform random numbers generated earlier is below, and the histogram of the resulting exponential variates is shown in Figure A.3.

```
X <- qexp(U)
hist(X)
```

For the error proportion model corresponding to $\alpha = -0.5$, recall that

$$F(y) = y^{0.5}, \quad y \in (0, 1)$$

so the simulated number of errors, Y , can be generated from

$$F(Y) = U \text{ or}$$

$$\sqrt{Y} = U$$

which means $Y = U^2$ should work as below:

```
Y <- U^2
hist(Y)
```

Compare the histogram in Figure A.4 with the black density curve plotted in Figure B.2; the latter is the graph of the error probability density function when $\alpha = -0.5$.

¹Actually, these numbers are not really random, because of the way in which they are produced; they are often referred to as pseudorandom numbers.

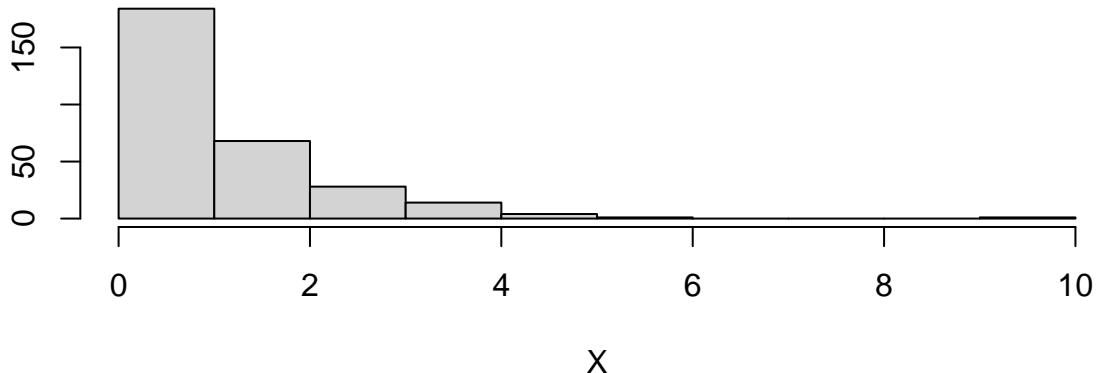


Figure A.3: Histogram of sample of 300 independent standard exponential numbers obtained as the exponential quantiles corresponding to random uniform numbers generated in Figure A.2.

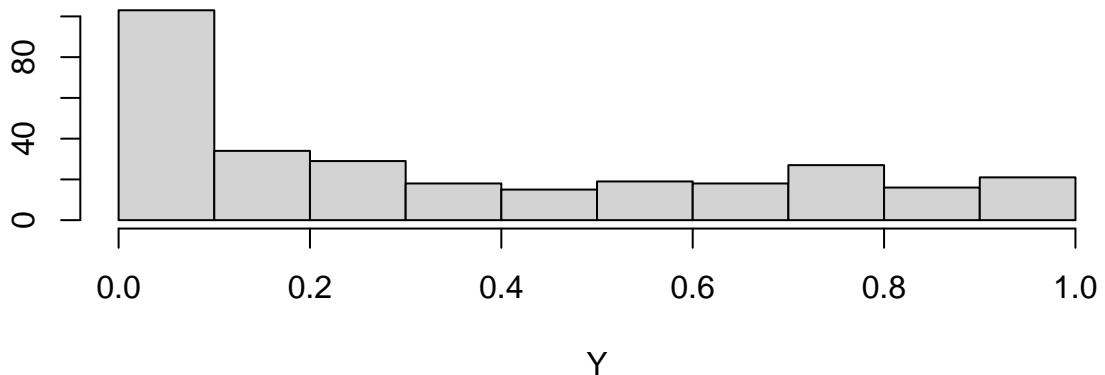


Figure A.4: Histogram of 300 simulated independent error proportions obtained as the quantiles corresponding to random uniform numbers generated in Figure A.2.

A.2.2 Built-in simulation functions

We can simulate samples of independent exponential random variates using the `rexp()` function. For example, we can simulate 5 such values as follows:

```
expSample <- rexp(5, rate = 0.1)
expSample

## [1] 17.2221729 2.7197404 1.3067737 13.2966618 0.4638834
```

By simulating a large sample and plotting a histogram of the result, we can gain more insight into the probability density function plotted in Figure A.1. To this end, we simulate 600 exponential variates with rate 0.1, and plot their relative frequency histogram. The result is shown in Figure A.5.

```
SimulatedSample <- rexp(600, rate = 0.1)
hist(SimulatedSample, breaks=seq(0, 70, 5), freq = FALSE)
```

The first argument of `hist()` is the data (contained in the object `SimulatedSample` here). The remaining arguments are optional, but in this case, they allow us to control where the histogram bins are drawn and whether the raw frequencies within each bin are calculated (`freq = TRUE`) or whether the corresponding proportions are (`freq = FALSE`).

The basic shape of the histogram in Figure A.5 matches the shape of the corresponding probability density function plotted in Figure A.1. The observed proportion of simulated exponential values occurring in the interval $[0, 5]$ is largest, and the observed proportions exponentially decrease in each of the successive intervals.

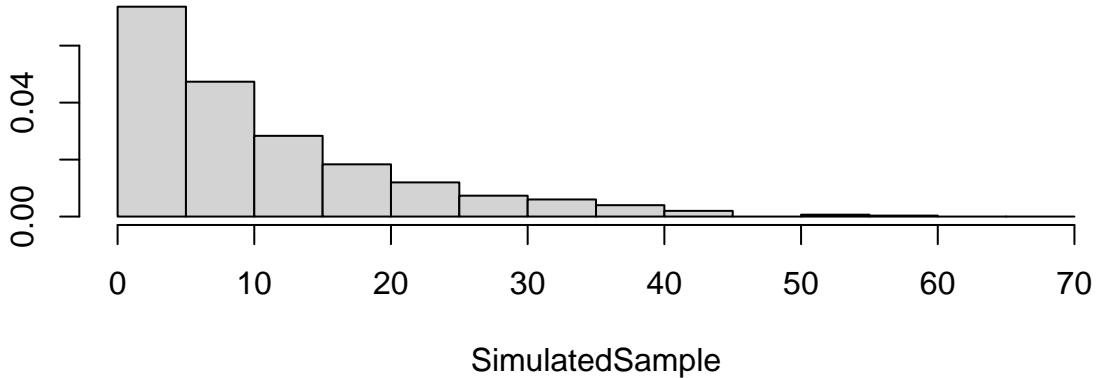


Figure A.5: Relative frequency histogram of simulated sample of 600 exponential variates with failure rate 0.1.

A.3 Expected value

The expectation operator $E[\cdot]$ is a linear operator on random variables. The expectation of a single (continuous) random variable X can be written as

$$E[X] = \int_{-\infty}^{\infty} xf(x)dx$$

where $f(x)$ is the probability density function of X . We say $E[X]$ is the mean of X . The expected value gives us a single number that, at least in a rough sense, conveys a typical value for the random variable. It is sometimes called a measure of location, since it specifies the location of the distribution along the real axis.

For the density function $f(x) = (\alpha + 1)x^\alpha$, for $x \in [0, 1]$, we have

$$E[X] = \int_0^1 x(\alpha + 1)x^\alpha dx = \frac{\alpha + 1}{\alpha + 2}. \quad (\text{A.1})$$

For the specific case where $\alpha = -0.5$, this gives $E[X] = 1/3$. In other words, the expected error proportion is 1/3 under this model. A commonly used alternate notation for the mean of a distribution is μ , the Greek letter which roughly translates to the letter “m”.

Other types of expected value can be calculated by the appropriate integration. For continuous functions $g(x)$, we have

$$E[g(X)] = \int_{-\infty}^{\infty} g(x)f(x)dx.$$

When a is a nonrandom constant, and $g(x) = ax$, we have

$$E[aX] = \int_{-\infty}^{\infty} axf(x)dx = a \int_{-\infty}^{\infty} xf(x)dx = aE[X].$$

For example, if $\alpha = -0.7$, and $a = 10000$, the above formula tells us that the expected number of incorrect pixels in an image consisting of 10000 pixels is

$$10000 \times 0.231 = 2310.$$

Similarly, it can be shown that

$$E[X + a] = E[X] + a.$$

When $g(x) = x^2$, and the probability density function is as above, we have

$$E[X^2] = \int_0^1 x^2(\alpha + 1)x^\alpha dx = \frac{\alpha + 1}{\alpha + 3}.$$

A.3.1 Variance

A feature of a distribution which is every bit as important as its location is its scale, or a measure of the degree of variability of the distribution. The variance is one way to measure the variability of a random variable. Denoting the mean of X by μ , we have

$$V(X) = E[(X - \mu)^2] = \int_{-\infty}^{\infty} (x - \mu)^2 f(x)dx.$$

An algebraically equivalent expression is

$$V(X) = E[X^2] - \mu^2.$$

For the error proportion distribution $f(x) = (\alpha + 1)x^\alpha$, the variance is

$$V(X) = \frac{\alpha + 1}{\alpha + 3} - \frac{(\alpha + 1)^2}{(\alpha + 2)^2} = \frac{\alpha + 1}{(\alpha + 3)(\alpha + 2)^2}.$$

A small value of $V(X)$ implies that there is more certainty about the value of X ; it will tend to take values close to μ when $V(X)$ is very small. The distribution will be more spread out when $V(X)$ is large.

The standard deviation is the square root of the variance. Both quantities summarize the spread or variability in a probability distribution. Note also that

$$\text{Var}(aX) = a^2 \text{Var}(X) \quad (\text{A.2})$$

for any nonrandom constant a , and

$$\text{Var}(X + a) = \text{Var}(X). \quad (\text{A.3})$$

A.3.2 Calculating the mean and variance from a sample

When confronted with a sample of measurements x_1, x_2, \dots, x_n , we can calculate the sample mean by taking the average of the sample values:

$$\bar{x} = \frac{1}{n} \sum_{j=1}^n x_j.$$

The sample variance is calculated as

$$s^2 = \frac{1}{n-1} \sum_{j=1}^n (x_j - \bar{x})^2.$$

The sample standard deviation is the square root of this: s . The reader can verify that for the sample contained in `expSample`, the sample mean, sample variance, and sample standard deviation are, respectively,

```
mean(expSample)
## [1] 7.001846
var(expSample)
## [1] 59.39875
sd(expSample)
## [1] 7.707059
```

where we have also demonstrated how the calculations could be carried out in R.

A.3.3 Probability calculations

Probabilities are calculated by integration, as in the probability that X_1 lies in the interval (a, b) and X_2 lies in the interval (c, d) is

$$P(X_1 \in (a, b), X_2 \in (c, d)) = \int_a^b \int_c^d f(x_1, x_2) dx_1 dx_2.$$

Suppose that when there are error proportion measurements coming from 2 images, the joint probability density function for these measurements is

$$f(x_1, x_2) = (\alpha + 1)^2 (x_1 x_2)^\alpha, \quad x_1, x_2 \in (0, 1),$$

and 0, otherwise. Using calculus, it can be shown that

$$\int_0^1 \int_0^1 (\alpha + 1)^2 (x_1 x_2)^\alpha dx_1 dx_2 = 1.$$

The probability that both error proportions exceed 0.75 is

$$\int_{0.75}^1 \int_{0.75}^1 (\alpha + 1)^2 (x_1 x_2)^\alpha dx_1 dx_2 = (1 - 0.75^{\alpha+1})^2.$$

A.3.4 Expectation and covariance

Expected values are also calculated using double integrals:

$$E[g(X_1, X_2)] = \int \int g(x_1, x_2) f(x_1, x_2) dx_1 dx_2$$

where $g()$ is a continuous function of 2 variables, and the integrals are assumed to be taken over the support of the function $f()$ (i.e. wherever it is strictly positive). For the error proportion example,

$$E[X_1] = \int_0^1 \int_0^1 x_1 (\alpha + 1)^2 (x_1 x_2)^\alpha dx_1 dx_2 = \frac{\alpha + 1}{\alpha + 2}.$$

It is no coincidence that this is the same as the expected value of a single error proportion measurement calculated at (A.1). An identical calculation shows that $E[X_2] = \frac{\alpha+1}{\alpha+2}$. Also,

$$E[X_1 X_2] = \int_0^1 \int_0^1 x_1 x_2 (\alpha + 1)^2 (x_1 x_2)^\alpha dx_1 dx_2 = \frac{(\alpha + 1)^2}{(\alpha + 2)^2}.$$

The covariance of two random variables gives a measure of their association, and can be calculated using

$$\text{Cov}(X_1, X_2) = E[X_1 X_2] - E[X_1] E[X_2].$$

A nonzero value implies that the distributions of values of the pair of random variables will have (at least a vague) linear relationship. If the covariance is positive, high values of X_1 will tend to be associated with high values of X_2 and low values of X_1 will be associated with low values of X_2 . When the covariance is negative, the opposite holds: high values of X_1 tend to occur with low values of X_2 , and so on. When the covariance is 0, the slope of any line relating the distribution of values of X_1 to X_2 is 0.

In the case considered earlier, we see that $\text{Cov}(X_1, X_2) = 0$ which means that the two random variables do not have either a positive or negative linear association.

Dependent exponential random variables

For another example, consider random variables with the following joint probability density function

$$f(x_1, x_2) = \frac{\lambda}{x_1} e^{-\lambda x_1 - x_2/x_1}, \quad x_1, x_2 \geq 0$$

and 0, otherwise. We can see that X_1 and X_2 are positively associated as follows.

$$\begin{aligned} E[X_1] &= \lambda \int_0^\infty \int_0^\infty e^{-\lambda x_1 - x_2/x_1} dx_1 dx_2 = \lambda \int_0^\infty x_1 e^{-\lambda x_1} dx_1 = \frac{1}{\lambda}. \\ E[X_2] &= \lambda \int_0^\infty \int_0^\infty \frac{x_2}{x_1} e^{-\lambda x_1 - x_2/x_1} dx_1 dx_2 = \frac{1}{\lambda}, \end{aligned}$$

and

$$E[X_1 X_2] = \lambda \int_0^\infty \int_0^\infty \frac{x_1 x_2}{x_1} e^{-\lambda x_1 - x_2/x_1} dx_1 dx_2 = \frac{2}{\lambda^2}.$$

To compute these integrals, it is necessary to use integration-by-parts several times. The final observation is that

$$\text{Cov}(X_1, X_2) = \frac{2}{\lambda^2} - \frac{1}{\lambda^2} = \frac{1}{\lambda^2}.$$

This value is positive which means that X_1 and X_2 are positively related.

A.3.5 Correlation

The magnitude of the covariance, when nonzero, depends on the scale of the variables. For example, if the covariance between X_1 and X_2 is 1.0, it can be shown that the covariance between $10X_1$ and $10X_2$ is 100. By dividing through by the standard deviations of both variables, we can remove this dependence on scale. The resulting measure is the correlation:

$$\text{Corr}(X_1, X_2) = \frac{\text{Cov}(X_1, X_2)}{\sqrt{\text{Var}(X_1) V(X_2)}}.$$

When the covariance is 0, the correlation will obviously be 0, so this is the case for the two error proportion measurements discussed earlier: we say these measurements are uncorrelated.

Positive covariances lead to correlations which are positive but no larger than 1. Negative covariances lead to correlations which are negative but no less than -1.

Calculation of covariance and correlation for a sample

For a sample $\{(x_{11}, x_{21}), (x_{12}, x_{22}), \dots, (x_{1n}, x_{2n})\}$, the sample covariance is given by

$$c = \frac{1}{n-1} \sum_{j=1}^n (x_{1j} - \bar{x}_1)(x_{2j} - \bar{x}_2).$$

The sample correlation is given by

$$r = \frac{c}{s_1 s_2}$$

where s_1 and s_2 are the sample standard deviations of the samples of x_1 's and x_2 's respectively.

As an example, re-consider the simulated sample of exponential random variates discussed in subsection A.2.2. We can simulate a second sample, independently of this one, using

```
expSample2 <- rexp(5, rate = 0.5)
expSample2

## [1] 2.28432095 0.21785216 1.04455107 0.20491794 0.06621054
```

It doesn't really matter what rate we use; we have used a different one from before to emphasize that we don't have to use the same one. The reader can then verify that the covariance between `expSample` and `expSample2` is

```
cov(expSample, expSample2)

## [1] 4.330471
```

and the sample correlation is

```
cor(expSample, expSample2)

## [1] 0.6017248
```

where we have illustrated the use of the R functions for these calculations.

Note that the theoretical values should be 0, but we are observing nonzero values. This is because we are dealing with samples of the distributed values, and not the complete population. A larger sample size would lead to values which are closer to 0, as in

```
largeSample1 <- rexp(500000, rate = 0.1)
largeSample2 <- rexp(500000, rate = 0.5)
cov(largeSample1, largeSample2)

## [1] 0.04839714

cor(largeSample1, largeSample2)

## [1] 0.002418173
```

A.3.6 Marginal distributions

Earlier, we saw that

$$P(X_1 \in (a, b), X_2 \in (c, d)) = \int_a^b \int_c^d f(x_1, x_2) dx_1 dx_2.$$

What if we are only interested in $P(X_1 \in (a, b))$? In this case, it doesn't make a difference to us what the value of X_2 is; it could be any element of $(-\infty, \infty)$. In other words, we could write

$$P(X_1 \in (a, b)) = P(X_1 \in (a, b), X_2 \in (-\infty, \infty))$$

but, in integral form, this becomes

$$P(X_1 \in (a, b)) = \int_a^b \int_{-\infty}^{\infty} f(x_1, x_2) dx_2 dx_1.$$

Recalling how we defined the probability density function for a single random variable, i.e.

$$P(X_1 \in (a, b)) = \int_a^b f(x) dx,$$

it necessarily follows that the probability density function for X_1 must be

$$f(x) = \int_{-\infty}^{\infty} f(x_1, x_2) dx_2.$$

Similar reasoning tells us that the probability density function for X_2 must be

$$f(x) = \int_{-\infty}^{\infty} f(x_1, x_2) dx_1.$$

To distinguish these probability density functions, if necessary, we write $f_{X_1}(x)$ as the probability density function of X_1 and $f_{X_2}(x)$ as the probability density function of X_2 . $f_{X_1}(x)$ and $f_{X_2}(x)$ are referred to as marginal density functions.

To summarize, when in the context of several random variables, the marginal density of a single one of the random variables can be obtained by integrating the joint density function over all possible values of all of the random variables apart from the one of interest.

Error proportions

The marginal probability density function for X_1 , the proportion of pixel errors observed in the first image, is

$$f_{X_1}(x_1) = \int_0^1 (\alpha + 1)^2 (x_1 x_2)^\alpha dx_2 = (\alpha + 1) x_1^\alpha.$$

Similarly,

$$f_{X_2}(x_2) = (\alpha + 1) x_2^\alpha.$$

Observe that the functional forms are the same for both X_1 and X_2 . In this case, we say X_1 and X_2 are identically distributed.

Dependent exponential random variables

When the joint density function is

$$f(x_1, x_2) = \frac{\lambda}{x_1} e^{-\lambda x_1 - x_2/x_1}, \quad x_1, x_2 \geq 0$$

we can obtain the marginal density function for X_1 by integrating over all possible values of x_2 (i.e. $x_2 > 0$):

$$f_{X_1}(x_1) = \int_0^{\infty} \frac{\lambda}{x_1} e^{-\lambda x_1 - x_2/x_1} dx_2 = \lambda e^{-\lambda x_1} \quad x_1 \geq 0$$

and 0, otherwise. We recognize this as the exponential density function.

Attempting to obtain the marginal density function for X_2 results in

$$f_{X_2}(x_2) = \int_0^{\infty} \frac{\lambda}{x_1} e^{-\lambda x_1 - x_2/x_1} dx_1$$

which can only be evaluated numerically. Clearly, X_2 does not have the same density function as X_1 , so they are not identically distributed.

A.3.7 Conditional density functions

Suppose we know the value of X_1 , and we would like to predict the value of X_2 , using this information. The joint probability density function tells us how X_1 and X_2 are related, so we might think that $f(x_1, x_2)$ is the probability density function of X_2 for each given value of X_1 , but this would be an incorrect interpretation. This is because

$$\int_{-\infty}^{\infty} f(x_1, x_2) dx_2 = f_{X_1}(x_1)$$

If $f(x_1, x_2)$ were a probability density function for X_2 , given X_1 , the above integral should evaluate to 1.

However, if we divide $f(x_1, x_2)$ by $f_{X_1}(x_1)$, we obtain a function of x_2 which integrates to 1:

$$\int_{-\infty}^{\infty} \frac{f(x_1, x_2)}{f_{X_1}(x_1)} dx_2 = \frac{f_{X_2|X_1}(x_1)}{f_{X_1}(x_1)} = 1.$$

It turns out that this gives a very useful predictive density function for X_2 , given knowledge of X_1 . We write

$$f_{X_2|X_1}(x_1, x_2) = \frac{f(x_1, x_2)}{f_{X_1}(x_1)}$$

as the conditional density function of X_2 given X_1 . This density function predicts the probability density of X_2 , when we know the value of X_1 .

Similar reasoning tells us that the predictive probability density of X_1 or its conditional density function is given by

$$f_{X_1|X_2}(x_1, x_2) = \frac{f(x_1, x_2)}{f_{X_2}(x_2)}$$

Error proportions

When

$$f(x_1, x_2) = (\alpha + 1)^2 (x_1 x_2)^\alpha, \quad x_1, x_2 \in (0, 1)$$

and

$$f_{X_1}(x_1) = (\alpha + 1)x_1^\alpha, \quad x_1 \in (0, 1)$$

it follows that

$$f_{X_2|X_1}(x_1, x_2) = (\alpha + 1)x_2^\alpha, \quad x_2 \in (0, 1).$$

Note that this conditional density function is the same as the marginal density of X_2 , $f_{X_2}(x_2)$. Thus, X_1 does not give us any information about X_2 . These random variables are independent.

Dependent exponential random variables

The situation is different when

$$f(x_1, x_2) = \frac{\lambda}{x_1} e^{-\lambda x_1 - x_2/x_1}, \quad x_1, x_2 \geq 0$$

and

$$f_{X_1}(x_1) = \lambda e^{-\lambda x_1}, \quad x_1 \geq 0.$$

The conditional density for X_2 , given X_1

$$f_{X_2|X_1}(x_1, x_2) = \frac{1}{x_1} e^{-x_2/x_1}, \quad x_2 \geq 0.$$

This is an exponential density function, but now the rate is $1/x_1$. This density function is not the same as the marginal density of X_2 . Thus, X_1 gives predictive information about X_2 . The two random variables are not independent.

A.3.8 Independence

We noted a case, earlier, where X_1 and X_2 are independent. In that case, X_1 provided no predictive information about X_2 ; the conditional density function of X_2 given X_1 was identical to the marginal density function of X_2 :

$$f_{X_2|X_1}(x_1, x_2) = f_{X_2}(x_2). \tag{A.4}$$

Multiplying both sides of this by $f_{X_1}(x_1)$ gives the joint density function on the left and the product of the marginal density function on the right. Random variables are independent if their joint distribution can be factored in this way. That is, X_1 and X_2 are independent, if their joint density is

$$f(x_1, x_2) = f_{X_1}(x_1)f_{X_2}(x_2) \tag{A.5}$$

where the two functions are the respective marginal densities of X_1 and X_2 .

Expected values of products of independent random variables

An easy consequence of (A.4) is that for independent X_1 and X_2 , we have

$$E[X_1 X_2] = E[X_1]E[X_2].$$

In fact, for a large class of functions, $f_1(x)$ and $f_2(x)$, we have the more general result that

$$E[f_1(X_1)f_2(X_2)] = E[f_1(X_1)]E[f_2(X_2)]$$

when X_1 and X_2 are independent. To see this result, note that, because of (A.4), the left hand side can be written as

$$\int \int f_1(x_1)f_2(x_2)f_{X_1}(x_1)f_{X_2}(x_2)dx_1dx_2$$

and the double integral separates into the iterated integral

$$\int f_1(x_1)f_{X_1}(x_1)dx_1 \int f_2(x_2)f_{X_2}(x_2)dx_2$$

which is the equivalent to the right hand side.

A.3.9 A Case of Dependence

Let's change the story a bit now. Suppose X_1 and X_2 are related. In particular, suppose X_1 is exponentially distributed with rate $\lambda = 1.5$ and X_2 is exponential with rate $1/X_1$. Here, we will apply the simulation-based graphical analysis described in Section 3.3.1.

The code to obtain Figure A.6 is as follows.

```
X1 <- rexp(10000, rate = 1.5)
X2 <- rexp(10000, rate = 1/X1)
plot(X2 ~ X1)
```

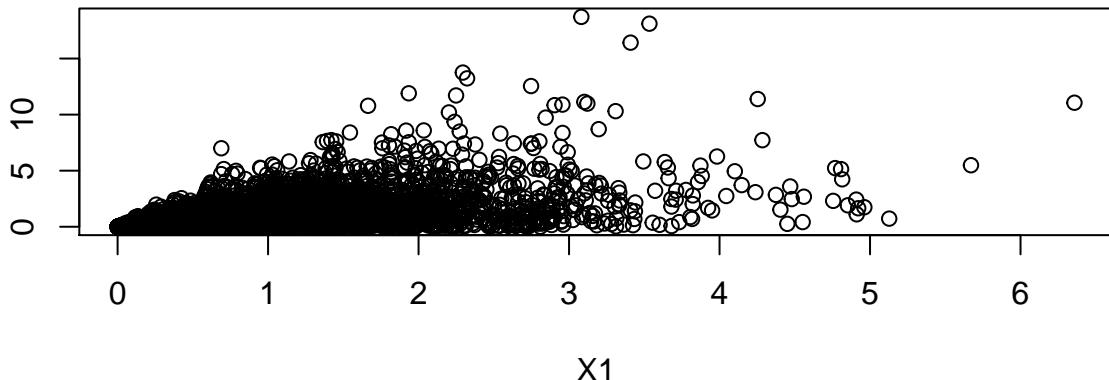


Figure A.6: A scatterplot of values taken from the distribution of X_2 against corresponding values taken from the distribution of X_1 , when X_1 and X_2 are dependent random variables.

In case it is not clear from Figure A.6 that the distribution of X_2 is changing, depending on the corresponding value of X_1 , we can make it more clear with Figure A.7 which is produced in exactly the same way as for Figure 3.7. That is,

```
par(mfrow=c(1, 2))
output <- sapply(split(X2, cut(X1, c(0, 1, 8))), hist, xlab="X2", main="", freq=FALSE)
```

This time, it can be seen that when X_1 is closer to 0, the distribution of X_2 is very different than when X_1 is farther from 0. X_2 is clearly dependent on X_1 .

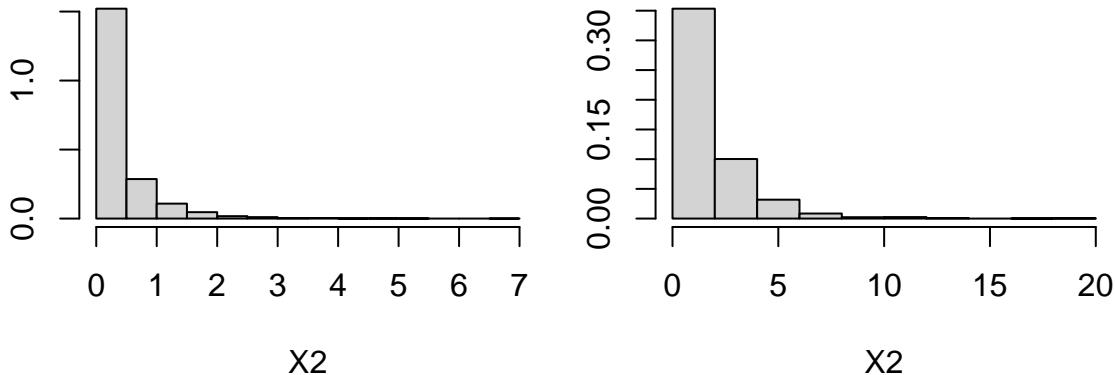


Figure A.7: Histograms of X_2 , depending on whether X_1 is less than or equal to 1 (left panel) or greater than 1 (right panel). This time, X_1 and X_2 are dependent.

Exercises

1. Suppose U is a uniform random variable on the interval $[0, 1]$.
 - (a) Find the probability density function of the random variable $X = \theta U$, where θ is a positive constant.
 - (b) Find the probability density function of the random variable $Y = U^\alpha$, where α is a positive constant.
 - (c) Find the probability density function of the random variable $Z = -\log(U)$.
2. (a) Use the result of 1c to explain how you would simulate 100 exponential random numbers, if you were given 100 uniform random numbers.
 - (b) Generate 100 random numbers using `runif(100)`. Convert these to 100 exponential random numbers using your idea. Plot the estimated density of your numbers. Find the sample mean.
 - (c) Now change those exponential numbers so that they come from a population with mean 5.
3. Simulate 1000 random numbers from a normal population with mean 0 and variance 1. Plot the estimated density of the squares of those numbers. Plot the density of the χ^2 distribution on 1 degree of freedom. Compare.
4. Simulate another set of 1000 normal random numbers, again with mean 0 and variance 1. Square these and add to the squares from the previous question. Plot the estimated density of the resulting vector. Plot the density of the χ^2 distribution on 2 degrees of freedom. Compare.
5. Perform the previous step one more time, but this time compare with a density of the χ^2 distribution on 3 degrees of freedom.
6. Simulate 1000 standard normal random variates Z and square them. Obtain the density plot of the result as well as the $\chi^2_{(1)}$ and $\chi^2_{(2)}$ QQ plots. (Use the `qqmath()` function from the *lattice* package. See the script file on the course webpage for examples of `qqmath` usage.) What is the distribution of Z^2 ?
7. Simulate 1000 exponential random variates with mean 2, and construct $\chi^2_{(1)}$ and $\chi^2_{(2)}$ QQ plots. What can you conclude about this exponential distribution?
8. Simulate 1000 gamma random variates with $\alpha = 2$ and $\beta = 12$. Standardize them by subtracting their mean and dividing by their standard deviation. Obtain the density plot for the result. Square the standardized values and construct a $\chi^2_{(1)}$ QQ plot. Can you conclude that squaring this kind of gamma random variable results in a $\chi^2_{(1)}$ random variable?
9. Repeat the previous exercise using $\alpha = 20$.
10. Run the code in the R script for the normal case to verify that the distribution of $(n - 1)S^2/\sigma^2$ is $\chi^2_{(n-1)}$, for $n = 2, 20, 200$.
11. Repeat the previous exercise for the exponential case and for the binomial case.
12. Use simulation to show that S^2 and \bar{X} are independent when based on random samples of size 5 from a normal population having mean 17 and variance 289.

13. Use simulation to decide whether S^2 and \bar{X} are approximately independent when based on random samples of size 5 from a Poisson population with mean 17. What about when the mean is 1.7?
14. Use simulation to study the distributions of

$$Z = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}}$$

$$T = \frac{\bar{X} - \mu}{S/\sqrt{n}}$$
- for the case where samples of size 5 are taken from normal population with mean 17 and variance 289. Check normal and t_4 QQ plots. Repeat for Poisson populations having means 17 and 1.7.
15. Suppose the time in hours until a light bulb burns out is modeled as an exponential random variable with failure rate 0.002. Find
 - (a) the probability that the light bulb lasts more than 600 hours.
 - (b) the expected lifetime for the light bulb.
 - (c) the variance of the lifetime distribution.
16. Use the `qnorm()` function to generate 300 standard normal random variates, using the 300 uniform random numbers.

B

Maximum Likelihood Estimation

Errors can occur in the production of two-dimensional medical images. A probability model for the proportions of such errors can be of use for quality assurance. For example, it is useful to know whether a machine is producing an unusually high proportion of errors.

An approximate model for the proportion of in an image that have been incorrectly classified is

$$f(x) = (\alpha + 1)x^\alpha, \quad 0 \leq x \leq 1$$

where α is an *unknown parameter*. The function $f(x)$ is an example of a probability density function. It completely characterizes the probability model. In Figure B.1 below, one of the many possibilities is sketched; the value of α has been taken as -0.75 .

The density function is highest at values of x that are most probable. In this case, we might expect to observe error proportions which are close to 0, and we would not expect to see many near 1.

```
alpha <- -0.75
curve((alpha+1)*x^alpha, ylab="f(x)")
```

B.1 Which parameter value is the most plausible?

In Chapter 2, we made brief reference to the use of maximum likelihood estimation in the context of simple linear regression to point out that it was equivalent to least-squares estimation when the data followed a normal distribution. Our approach was fairly “mechanical” in nature; for the student with little familiarity with the technique, the section may have been somewhat terse. In this chapter, we illustrate the *concept* of maximum likelihood estimation, a commonly used technique for using data to estimate unknown parameters which govern a probability model. We consider the problem of finding the unknown parameter α for the probability density function for the proportion of errors in an image:

$$f(x) = (\alpha + 1)x^\alpha.$$

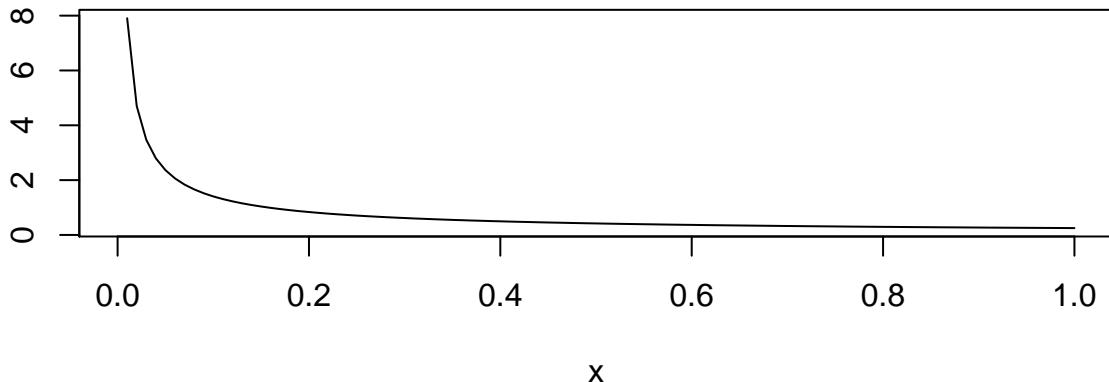


Figure B.1: Graph of a possible probability density function model for error proportions in images; the parameter $\alpha = -0.75$.

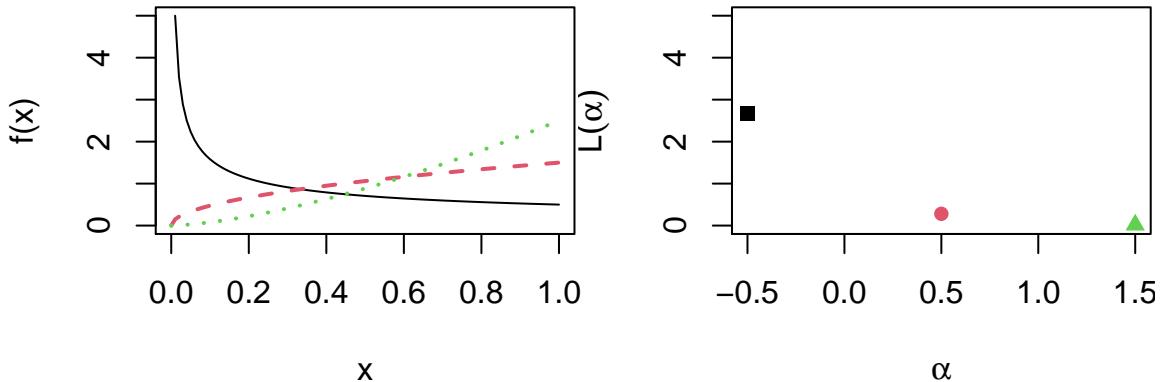


Figure B.2: Left Panel: Three possible models for image error proportions. Each curve is a graph of the function $f(x) = (\alpha + 1)x^\alpha$, where $\alpha = -0.5$ (black solid), $\alpha = 0.5$ (red dashed) and $\alpha = 1.5$ (green dotted). Right Panel: The likelihood function for the image error proportion problem, where there are 3 possible values for the unknown parameter.

Because α is unknown, it is impossible to know which probability density function is the correct one. By calculating the proportion of incorrect pixels in an image, we can *estimate* α by choosing the most likely or realistic value.

The maximum likelihood concept is based on probability density. Consider the left panel of Figure B.2 where three of the possible error distribution models have been plotted, each corresponding to a different value of α . When $\alpha = -0.5$, most error proportions would be close to 0 with a few larger ones. When $\alpha = .5$, there is a tendency for the proportions to be larger, and when $\alpha = 1.5$, most of the proportions are large than, and only a few would be near 0.

Suppose the proportion of errors in one image is $x = .035$. Unless we have taken an unusual measurement, the probability density function at our measurement should be high. Of the three possibilities graphed in the figure, the most likely is the one corresponding to $\alpha = -0.5$. If the three choices were the only possibilities available, we would say that the *maximum likelihood estimate* for α is -0.5 .

Note that our choice for the value of α was accomplished by comparing $f(0.035)$ for the three possibilities. In other words, we compared

$$f(0.035) = (\alpha + 1)(0.035)^\alpha$$

for $\alpha \in \{-0.5, 0.5, 1.5\}$ and chose the value of α which gave the largest value of $f(0.035)$. Notice that the expression for $f(0.035)$ is now a function of α , no longer of x (the value of x is now known to be 0.035).

B.1.1 The likelihood function

As a function of α , $f(0.035)$ gives us a way of deciding which value of α is most plausible or most likely. Therefore, it has been named the likelihood function. We would write

$$L(\alpha) = f(0.035) = (\alpha + 1)(0.035)^\alpha, \text{ for } \alpha \in \{-0.5, 0.5, 1.5\}$$

and $L(\alpha) = 0$ for all other values of α , since we are supposing that these other values are not possible. This likelihood function is graphed in the right panel of Figure B.2. Notice that there are only three points in this graph, since there are only three possible values for α . The value of α which maximizes this likelihood is now clearly seen to be -0.5 .

B.1.2 Using the fitted model

We can now use the fitted model to calculate estimates of the probability that the error proportion v for a new image as discussed earlier. Caution is warranted, however, since we have based our model on only a single observation. We will see later how to use a larger sample of independent observations to improve on the fitted model, but before that, we need to consider what happens when we allow more choice for the possible values of α .

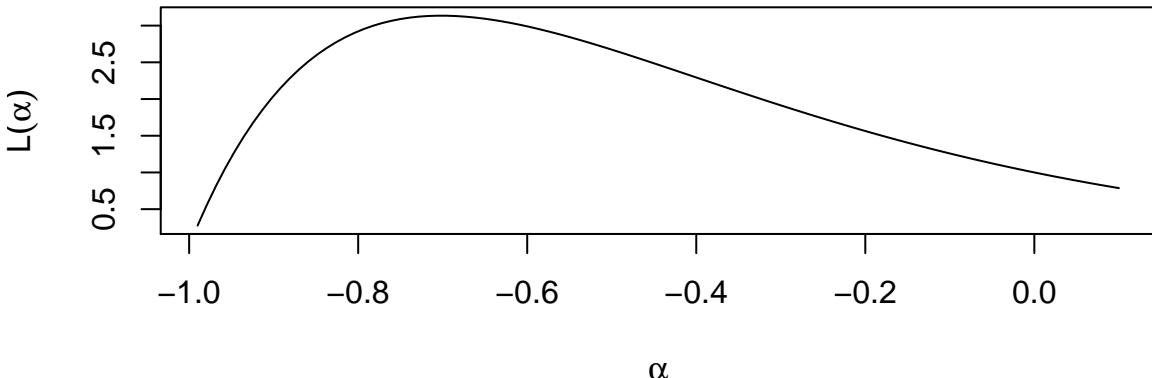


Figure B.3: The likelihood function for the image error proportion problem, where there are now an infinite number of possible values for the unknown parameter.

B.1.3 Maximizing the likelihood using calculus

In fact, α could take on any value greater than -1.0 and the probability density function above would still be valid. Therefore, we might be interested in the most plausible value of α from this infinitely large set. In other words, we need to see where

$$f(0.035) = (\alpha + 1)0.035^\alpha$$

is largest, or as we have seen, we are interested in maximizing the likelihood

$$L(\alpha) = (\alpha + 1)0.035^\alpha.$$

This likelihood function is sketched in Figure B.3.

We can *estimate* α by maximizing this likelihood. From the figure, it can be seen that the maximum value occurs near $\alpha = -0.7$. The exact value can be determined using calculus: differentiate $L(\alpha)$ and solve for α in $L'(\alpha) = 0$. That is, solve

$$\frac{1}{\alpha + 1} = -\log(0.035)$$

The solution is $\hat{\alpha} = -0.702$.

B.2 Modelling more than one random variable

We noted earlier that estimating α should be carried out using many measurements, not just one. We now consider probability models for more than one observation.

We start with 2 measurements and consider larger samples subsequently. Suppose x_1 and x_2 are error proportions observed in two medical images. Since there are 2 observations, the probability density function should be a function of 2 variables:

$$f(x_1, x_2).$$

In order for valid probabilities to be computed, it is necessary that this function be nonnegative, and its total (double) integral be 1.

B.2.1 Likelihood when there are 2 measurements

If images are constructed under similar conditions, the joint density function for error proportions from 2 independently obtained images, x_1 and x_2 , is

$$f(x_1, x_2) = f(x_1)f(x_2) = (\alpha + 1)^2(x_1x_2)^\alpha.$$

Figure B.4 exhibits contour plots of the joint density function corresponding to the cases where $\alpha = -0.25$ and $\alpha = -0.75$.

The first observed error proportion was $.035$. A second independent error proportion is $.038$. The data are represented by the ordered pair $(.035, .038)$ and are plotted as a solid black dot. Comparing the heights of the contours at the data point, corresponding to the two possible α values, we see that the probability density is much higher at the observed data when $\alpha = -0.75$. Thus, this value is the more likely of the two candidate values.

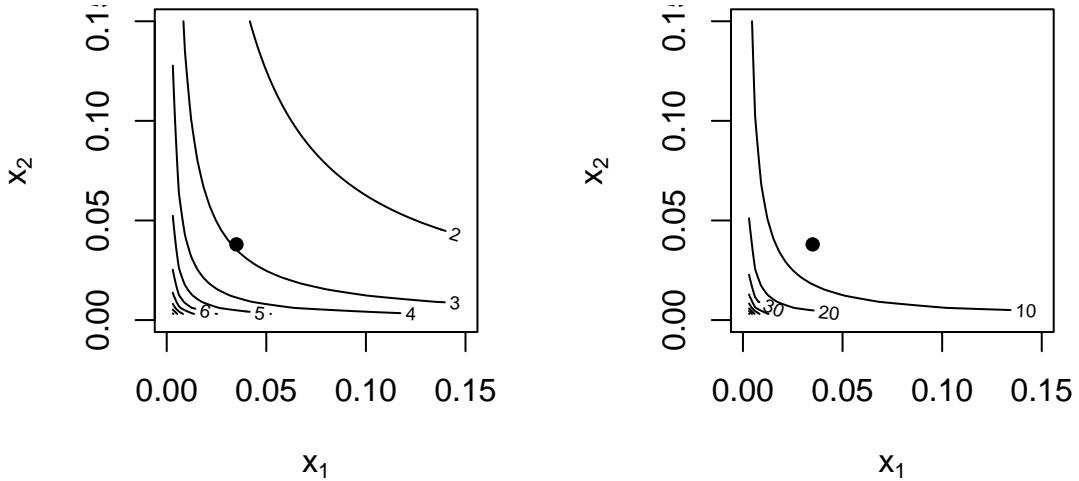


Figure B.4: Contour plots of joint density function for 2 image error proportions. Left Panel: $\alpha = -0.25$. Right Panel: $\alpha = -0.75$. The location of the observed data is denoted by the solid black dot in each panel.

```

x1 <- seq(0, .15, length=50)
x2 <- seq(0, .15, length=50)
par(mfrow=c(1,2), pty="s", mar=c(.4, .4, .1, .1))
alpha <- -0.25
f <- outer(x1, x2, function(x,y) (alpha + 1)^2 * (x*y)^(alpha))
contour(x1, x2, f, xlab=expression(x[1]), ylab=expression(x[2]))
points(.035, .038, pch=16, cex=1)
alpha <- -0.5
f <- outer(x1, x2, function(x,y) (alpha + 1)^2 * (x*y)^(alpha))
contour(x1, x2, f, xlab=expression(x[1]), ylab=expression(x[2]))
points(.035, .038, pch=16, cex=1)

```

Analogous to the case of a single measurement, we have identified the more likely value of α by choosing the value of α which makes $f(x_1, x_2) = f(.035)f(.038)$ larger.

Thus, in the case of 2 independent measurements, the likelihood function is given by

$$L(\alpha) = f_{X_1}(x_1)f_{X_2}(x_2).$$

For the error proportion modelling problem, we are assuming that the distributions of X_1 and X_2 are the same, so we have

$$L(\alpha) = f(.035)f(.038) = (\alpha + 1)^2(.035 \times .038)^\alpha.$$

We now assume that α can take any value larger than -1 . The likelihood function is plotted in the left panel of Figure B.5. The maximum likelihood estimate is the value of α that maximizes this function.

Equivalently, we can maximize the logarithm of $L(\alpha)$:

$$\log L(\alpha) = 2 \log(\alpha + 1) + \alpha \log(.035 * .038).$$

This log likelihood function is sketched in the right panel of Figure B.5. Note that the maximizers for the likelihood and log likelihood occur at the same place. Differentiating the above expression with respect to α and setting the result to 0, we see that the maximizer is $\hat{\alpha} = -.698$, so we might write

$$f(x) = 0.302x^{-.698}$$

as our probability model.

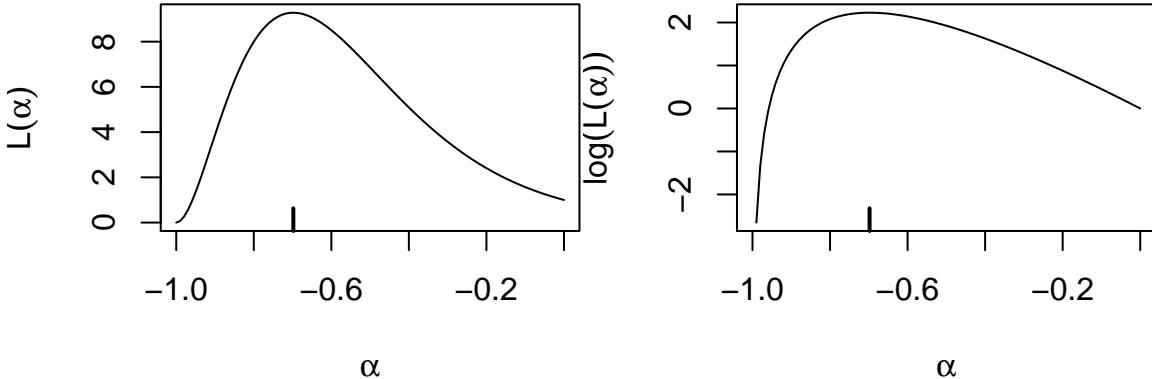


Figure B.5: Left Panel: Likelihood function for sample of size 2. Right Panel: The corresponding log likelihood. The location of the maximum in both cases is at $\alpha = -0.698$.

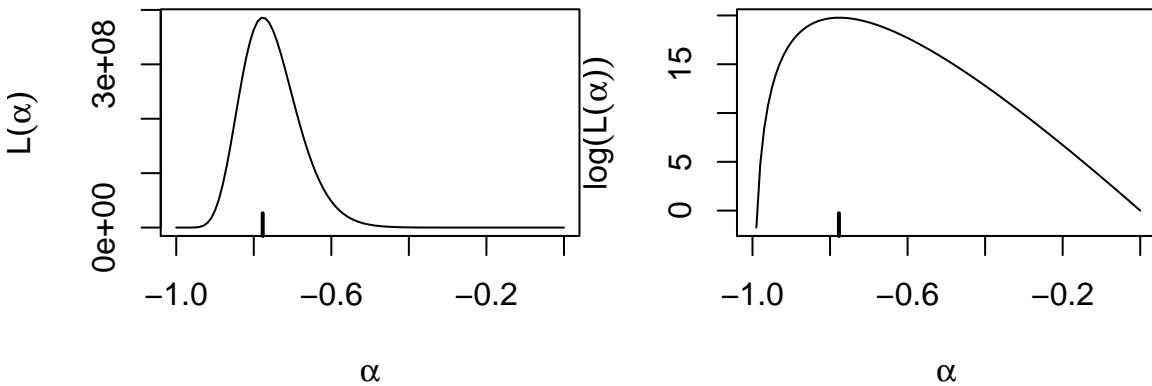


Figure B.6: Left Panel: Likelihood function for sample of size 10. Right Panel: The corresponding log likelihood. The location of the maximum in both cases is at $\alpha = -0.777$.

B.2.2 Likelihood with a larger sample

A larger sample of error measurements can be handled similarly.

For a sample of n independent random variables, the joint density would be

$$f_{X_1}(x_1)f_{X_2}(x_2) \cdots f_{X_n}(x_n).$$

If the marginal densities are all the same and depend on an unknown parameter θ , the likelihood function is

$$L(\theta) = f_X(x_1)f_X(x_2) \cdots f_X(x_n). \quad (\text{B.1})$$

Modelling the imaging pixel error proportions again

Here is a sample of 10 independent error proportions:

0.005 0.001 0.002 0.032 0.001 0.003 0.251 0.040 0.045 0.084

As before, we assume that the individual measurements come from the same distribution.

Using the assumption that the measurements are independent and identically distributed, the joint density evaluated at the measurements is

$$f(x_1, x_2, \dots, x_{10}) = f(x_1)f(x_2) \cdots f(x_{10}) = (\alpha + 1)^{10} (3.64 \times 10^{-20})^\alpha$$

Again, this is a function of α . The logarithm of this likelihood function is

$$\log L(\alpha) = 10 \log(\alpha + 1) - 44.76\alpha$$

The likelihood and log likelihood are displayed in Figure B.6. Notice that the likelihood function peak is much narrower in this figure than in Figure B.5 where the sample had been much smaller. Maximizing the log likelihood, we see that

$$\hat{\alpha} = -0.777.$$

B.2.3 The exponential distribution model

For exponentially distributed data x_1, x_2, \dots, x_n , with rate λ , the likelihood function is

$$L(\lambda) = \lambda^n e^{-\lambda \sum x_j}$$

and the log likelihood becomes

$$\log L = n \log(\lambda) - \lambda \sum_{j=1}^n x_j$$

It is a simple matter to maximize this, with respect to λ , to find that

$$\hat{\lambda} = \frac{1}{\bar{x}}.$$

Application to wind speed modeling

Observations of noon hour wind speed taken at the Winnipeg International Airport are listed in the second column of the MPV object `windWin80`. Squaring these observations yields a dataset for which the exponential distribution is a rough approximation as can be seen in the QQ-plot in Figure B.7 obtained from the following code.

```
library(MPV) # contains windWin80
windWin80sq <- windWin80$h12^2
lambdahat_sq <- 1/mean(windWin80sq)
qqplot(windWin80sq, qexp((1:366)/367, rate = lambdahat_sq),
       xlab = "Squared Windspeed", ylab = "Exponential quantiles")
abline(0,1)
lambdahatrounded <- round(lambdahat_sq, 5)
```

Note that, the maximum likelihood estimator for λ has been applied to the squared wind speed values. The value of the estimate obtained here is 0.00169. The QQ-plot indicates some trouble in the tail of the distribution, so the model is not completely accurate. An improved fit is given in Figure B.8 where a slightly different transformation, based on a power of 5/3 instead of 2, has been applied.

```
windWin80t <- windWin80$h12^1.66
lambdahat <- 1/mean(windWin80t)
qqplot(windWin80t, qexp((1:366)/367, rate = lambdahat),
       xlab = "Transformed Windspeed", ylab = "Exponential quantiles")
abline(0,1)

lambdahatrounded <- round(lambdahat, 5)
```

The estimated value of λ is 0.00535, under this model.

B.2.4 A Weibull model

In the previous section, we considered the square of the wind speed, but we could consider other possibilities. That is, if X represents a wind speed measurement, we could suppose $X = Y^\alpha$ for some unknown parameter α (which we will assume to be positive), and that Y is exponentially distributed with rate λ .

This means that $P(Y \leq y) = 1 - e^{-\lambda y}$ and

$$P(X \leq x) = P(Y^\alpha \leq x) = P(Y \leq x^{1/\alpha}) = 1 - e^{-\lambda x^{1/\alpha}}.$$

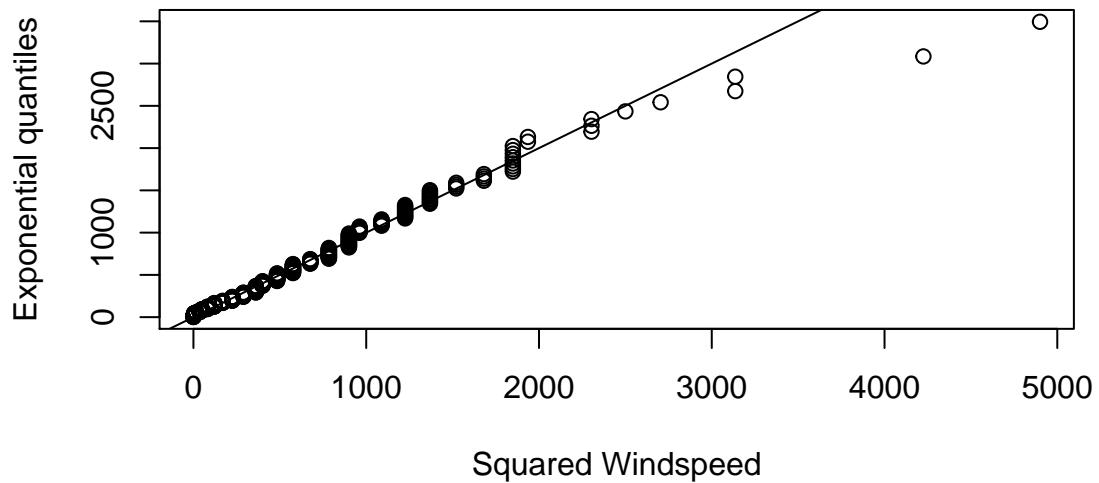


Figure B.7: Exponential QQ-plot of squared Winnipeg wind speed observations for 1980.

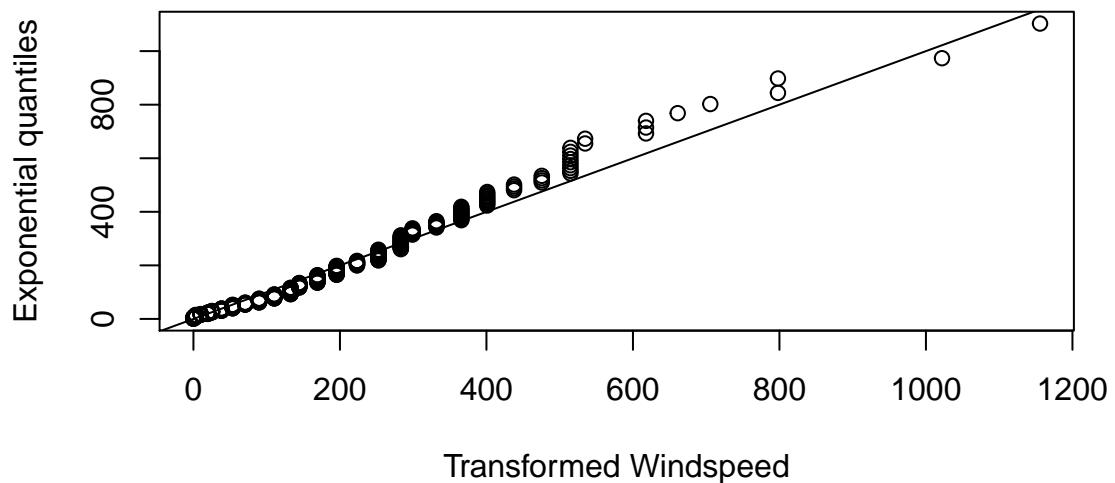


Figure B.8: Exponential QQ-plot of transformed Winnipeg wind speed observations for 1980.

Differentiating this distribution function with respect to x gives the probability density function:

$$f(x) = \frac{\lambda}{\alpha} x^{1/\alpha - 1} e^{-\lambda x^{1/\alpha}}.$$

Given a sample of independent measurements from this distribution, we can write down the log likelihood as

$$\log L = -n(\log(\lambda) - \log(\alpha)) + (1/\alpha - 1) \sum_{j=1}^n \log(x_j) - \lambda \sum_{j=1}^n x_j^{1/\alpha}.$$

We note the `fitdistr()` function in the MASS package which can be used to fit this distribution, but in the case of the wind speed data, there will be issues with the 0's. These need to be adjusted before the method can be applied. One possibility is to add 0.5 to all of those observations in order to avoid taking logarithms of 0.

B.3 Predictive modelling using the likelihood

We will see that the usefulness of generalized linear models comes in large part from their ability to relate response variables which follow particular probability distributions to one or more predictor variables, through exploitation of the likelihood framework.

B.3.1 Viewing model parameters as functions of covariates

Until now, we have viewed the parameters of probability models as fixed constants to be estimated. For example, in the wind speed example considered above, the parameter λ was taken to be constant. However, by viewing λ as a function of one or more covariates, we have the potential to use the probability model to make predictions based on values of these covariates. For example, if Y is an exponential random variable with parameter λ that depends on the value of another covariate x , the probability density function for Y becomes

$$f_Y(y) = \lambda(x) e^{-\lambda(x)y}$$

where $\lambda(x)$ is a known or presumed to be known function. Often this function is modelled in such a way that its inverse function, usually referred to as the link function, is a linear function of the covariate or covariates.

B.3.2 Poisson regression

We saw earlier that weighted nonlinear least-squares could be used to model the `cigbutts` data set (in the *MPV* package), noting that a Poisson model might be appropriate. The likelihood setup gives a more direct approach.

Recall that the Poisson distribution with mean λ is given by

$$P(X = x) = \frac{e^{-\lambda} \lambda^x}{x!}.$$

A popular choice of link function for the Poisson distribution model is the log link. A covariate x would be included in such a model through

$$\log(\lambda) = \beta_0 + \beta_1 x. \quad (\text{B.2})$$

This link function is modelled in such a way that recognizes that λ must be positive, while the right hand side of the above equation can take any real value.

Likelihood for the cigarette butts data

If X_i is the number of butts observed at the i th location, its expected value would be λ_i and the corresponding distance might be denoted as d_i . Therefore, the model for the i th observation is really

$$P(X_i = x_i) = \frac{e^{-\lambda_i} \lambda_i^{x_i}}{x_i!}.$$

From the discussion in this chapter, the likelihood function is the product of such probabilities (provided the observations are independent¹). The log likelihood is then the sum of the logs of the probabilities:

$$\log L = - \sum_{i=1}^n \lambda_i + \sum_{i=1}^n x_i \log \lambda_i - \sum_{i=1}^n \log x_i!.$$

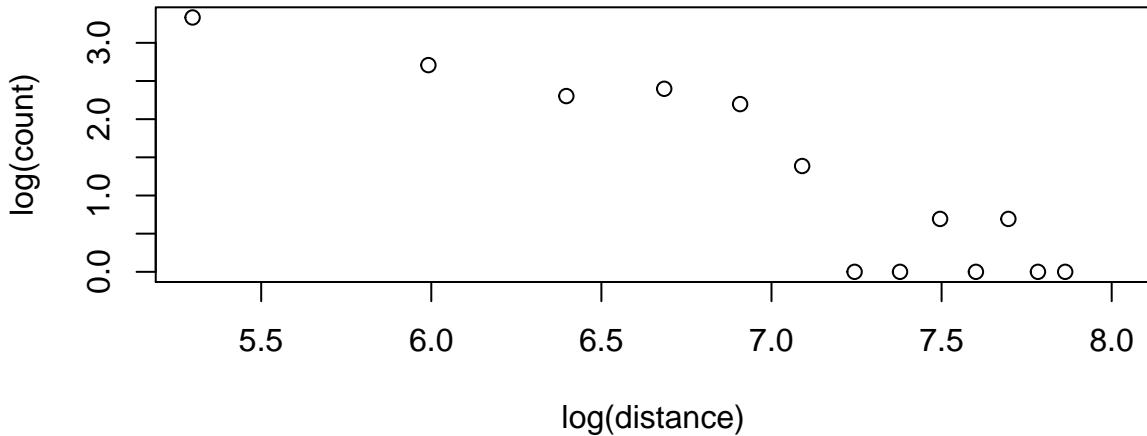


Figure B.9: Scatterplot of log of cigarette butt counts versus log of distance.

From the graph in the right panel of Figure B.9, we suspect a linear relation between the log of λ_i and d_i :

$$\log(\lambda_i) = \beta_0 + \beta_1 d_i$$

but let's suppose we know that $\beta_0 = 3.55$.² Then we could say that

$$\log(\lambda_i) = 3.55 + \beta_1 d_i$$

Plugging this into the log likelihood expression gives

$$\log L = - \sum_{i=1}^n e^{3.55 + \beta_1 d_i} + \sum_{i=1}^n x_i(3.55 + \beta_1 d_i) - \sum_{i=1}^n \log x_i!.$$

Recall that the x_i 's are the observed counts and the d_i 's are the distances, so we can plot this as a function of β :

```

x <- cigbutts$count
d <- cigbutts$distance
n <- length(x)
b <- seq(-.002, -.0015, length = 101)
logLterms <- outer(b, 1:n,
  function(b, i) -exp(3.55+ b*d[i]) + x[i]*(3.55+b*d[i]))
logL <- apply(logLterms, 1, sum)
par(mar=c(4,4, .1, .1))
plot(b, logL, xlab = expression(beta[1]), ylab = "log L", type = "l")

```

Note that we have ignored the term that does not involve β_1 , since we are interested in maximizing the likelihood as a function of β_1 .

B.3.3 A graphical approach to estimating two parameters

Later in this book, we will see that there is a method for automatically obtaining numerical estimates of parameter values for generalized linear models. In order to prevent that technique from taking on a “black-box” characteristic, it is perhaps helpful to first visualize what is being maximized when fitting a generalized linear model. In the case of two variables, contour plots of the likelihood surface are extremely valuable.

¹Do you think this is strictly true here?

²We will see how to estimate more than one parameter at a time later.

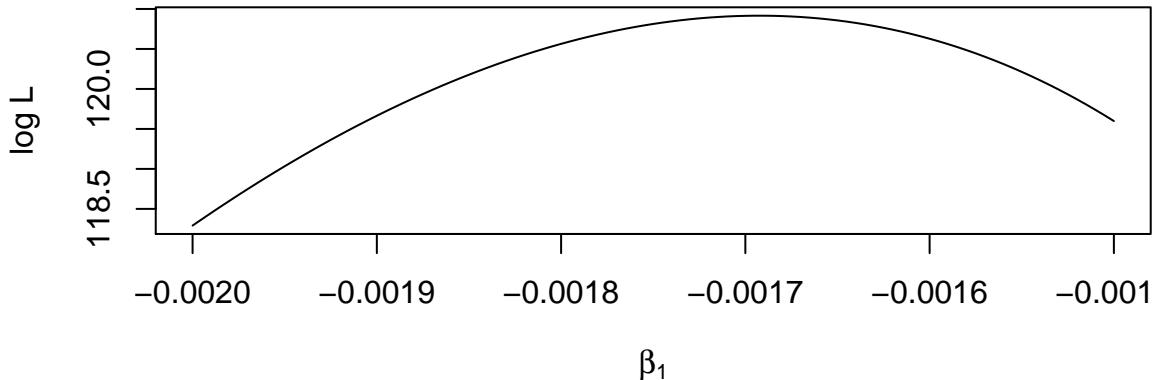


Figure B.10: Log likelihood curve for the cigarette butts example.

Predicting noon wind speed the previous midnight

The `windWin80` contains another column, called `h0`, containing the wind speed measurements at midnight, 12 hours before the noon wind speed measurements that we have been studying. A natural question to ask is: can we predict the future noon wind speed from the previous midnight wind speed?

A quick and simple approach is to use linear least-squares regression. The code below attempts to regress the transformed noon hour wind speed on to the previous midnight's wind speed.

```
wind.lm <- lm(h12^1.66 ~ h0, data = windWin80)
```

The estimated slope is 6.231 and the estimated intercept is 95.87.

On the other hand, we saw earlier that the noon hour wind speed measurements are not normally distributed, and that there is a simple transformation that renders them approximately exponentially distributed. Therefore, we might more reasonably try a likelihood approach.

To build a likelihood for a model where the distribution of noon hour wind speed depends on another covariate, we are confronted with the possibility that the wind speeds are not identically distributed. We postulate that the transformed noon wind speed is exponentially distributed with a rate parameter which depends on the previous midnight's wind speed. The particular relation must be guessed at, and a linear relation (in the mean) is a simple one to try:

$$\lambda = \frac{1}{\beta_0 + \beta_1 x}$$

where x represents the previous midnight's wind speed. Note that we are now attempting to estimate λ as a function of x , whereas previously, λ was assumed to be a constant. In order to estimate $\lambda(x)$, we must estimate β_0 and β_1 , whereas before, it was sufficient to estimate λ directly (by taking the reciprocal of the average of the h_{12} data points raised to the power 1.66).

With this form of λ (or more precisely, $\lambda(x)$), the marginal distribution of the transformed noon hour wind speed becomes

$$f(y|x) = \lambda(x) e^{-y\lambda(x)} = \frac{1}{\beta_0 + \beta_1 x} e^{-\frac{y}{\beta_0 + \beta_1 x}}$$

If we assume independence from noon hour to successive noon hour, the log likelihood becomes

$$\log L = - \sum_{j=1}^n \log(\beta_0 + \beta_1 x_j) - \sum_{j=1}^n \frac{y_j}{\beta_0 + \beta_1 x_j}.$$

In terms of h_0 and h_{12} , this would be written as

$$\log L = - \sum_{j=1}^n \log(\beta_0 + \beta_1 h_0 j) - \sum_{j=1}^n \frac{h_{12j}^{5/3}}{\beta_0 + \beta_1 x_j}.$$

We can code this in R as a function of β_0 and β_1 as follows:

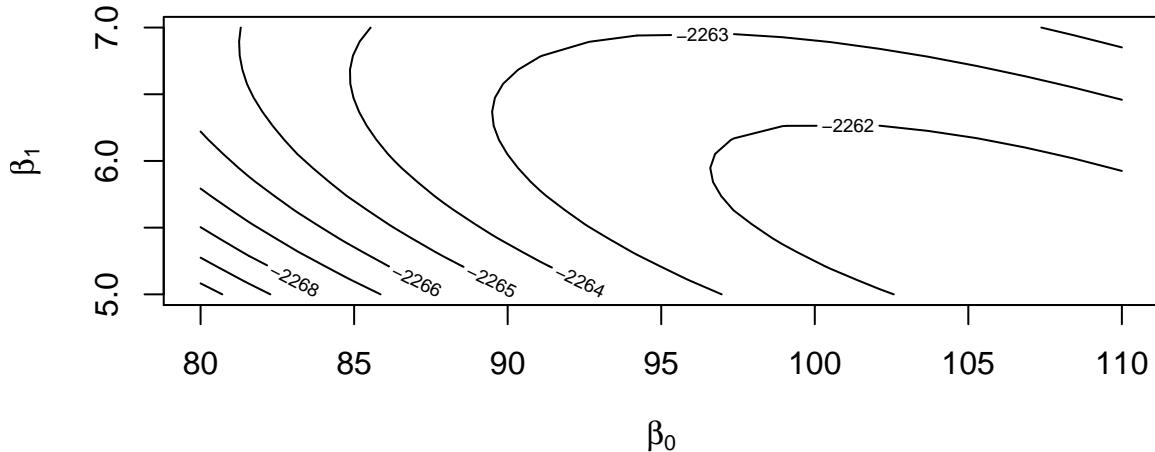


Figure B.11: Contour plot of log likelihood surface for transformed noon hour wind speed as it relates to midnight wind speed through a linear exponential regression.

```
windllhood <- function(b0, b1) {
  attach(windWin80)
  lambda <- 1/(b0 + b1*h0)
  ll <- sum(log(lambda)) - sum(lambda*h12^1.66)
  detach(windWin80)
  ll
}
```

Normally, we would use a numerical optimizer to maximize this function, but we will explore the likelihood surface numerically in this case. Since we have least-squares estimates of β_0 and β_1 , we can use these to guess rough ranges for the maximum likelihood estimates (80, 110) for β_0 and (5,7) for β_1 . We then compute the log likelihood at all combinations of a grid of these values.

```
llhood <- matrix(0, nrow=20, ncol=20)
beta0 <- seq(80, 110, length=20)
beta1 <- seq(5, 7, length=20)
for (i in 1:20) {
  for (j in 1:20) {
    llhood[i,j] <- windllhood(beta0[i], beta1[j])
  }
}
```

We can then obtain a contour plot of the likelihood surface using the `contour()` function:

```
par(mar=c(4, 4, 1, .1))
contour(beta0, beta1, llhood, xlab = expression(beta[0]),
        ylab=expression(beta[1]))
```

The contour plot is shown in Figure B.11. The contours approach a maximum in the vicinity of $\beta_0 \in (105, 110)$, and $\beta_1 \in (5, 7)$, so we can refine our grid, and re-plot the contours.

```
beta0 <- seq(105, 110, length=20)
beta1 <- seq(5, 5.5, length=20)
for (i in 1:20) {
  for (j in 1:20) {
```

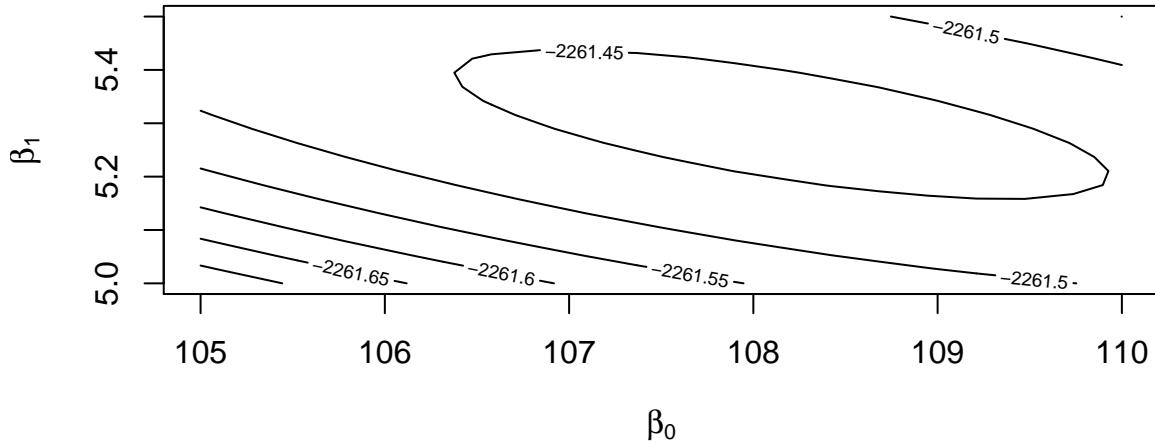


Figure B.12: Contour plot of log likelihood surface for transformed noon hour wind speed as it relates to midnight wind speed through a linear exponential regression.

```
    llhood[i,j] <- windllhood(beta0[i], beta1[j])
}
}
```

We can see in Figure B.12 that the values of β_0 and β_1 that maximize the log likelihood are near 108 and 5.3, respectively.

```
par(mar=c(4, 4, 1, .1))
contour(beta0, beta1, llhood, xlab = expression(beta[0]),
        ylab=expression(beta[1]))
```

Thus, the maximum likelihood approach leads us to the conclusion that, given the value of the wind speed at midnight x on the previous day, the distribution of the noon wind speed (raised to the power $5/3$) is distributed as an exponential random variable with mean

$$\mu = (5.3x + 108).$$

How well does the model fit the data?

Figure B.13 shows a scatter plot of the data with the overlaid fitted curve relating the noon day wind speeds to those of the previous midnights. We have also plotted a horizontal dashed line at the value of $1/\hat{\lambda}^{3/5}$, corresponding to a “null” model in which no dependency on h_0 is assumed, and for which the estimate of $1/\lambda$ had been obtained earlier by averaging the values of $h_{12}^{5/3}$.

```
par(mar=c(4, 4, .1, .1))
plot(h12 ~ h0, data = windWin80)
curve((5.3*x + 108)^(3/5), 0, 60, add = TRUE)
abline(h = 1/lambdahat^(3/5), lty = 2, lwd = 2) # lambdahat was the average of h12^1.66
```

The graph seems to indicate that the model might be providing somewhat more information about the mean behaviour, though it will not be a particularly useful predictive model, given the large amount of remaining unexplained variation.

In terms of checking model accuracy, a QQ-plot may be more useful anyway. Figure B.14 is based on a simulation of wind speeds from the fitted exponential distribution, conditional on the actual observed midnight wind speeds.

```
par(mar=c(4, 4, .1, .1))
with(windWin80, qqplot(h12,
                      rexp(length(h0), rate = 1/(5.3*h0 + 108))^(3/5), ylab = "theoretical quantiles"))
abline(0,1)
```

We see reasonable agreement between the quantile coordinate pairs and the 45° reference line, suggesting that the model is fitting the data adequately but not perfectly.

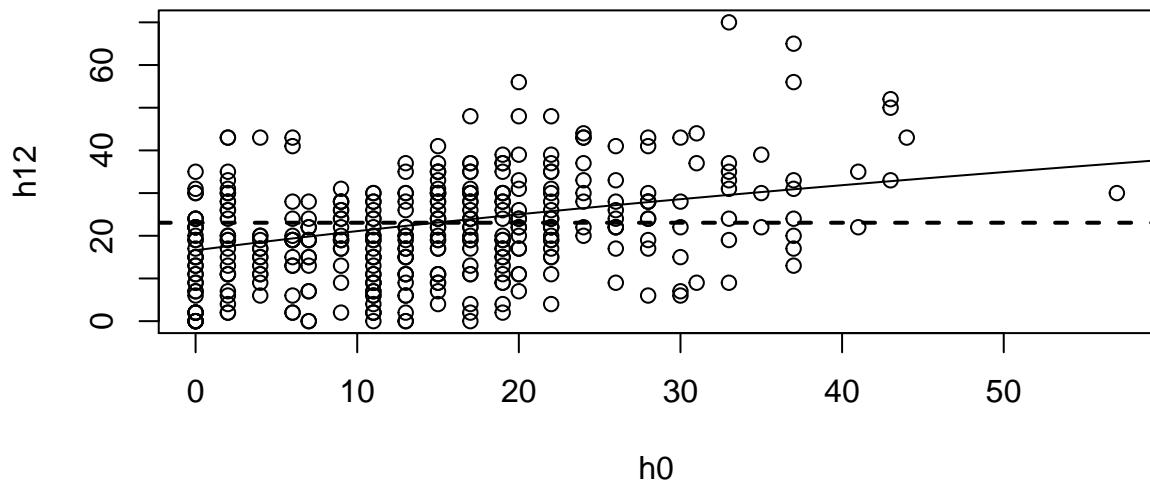


Figure B.13: Scatter plot of noon day wind speeds versus wind speeds from the previous midnight with overlaid Weibull regression model fitted by maximum likelihood (solid curve) and fitted null model (dashed line).

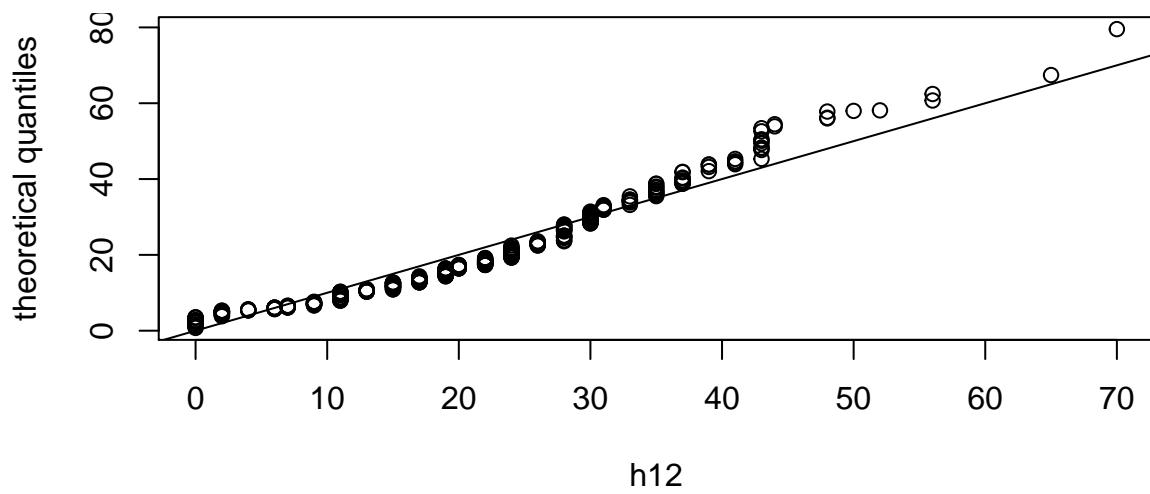


Figure B.14: Monte Carlo QQ-plot of the noon day wind speeds with a reference distribution of an exponential random variable raised to the power $3/5$, conditional on the previous midnight wind speeds.

B.4 Exercises

1. The reliability of an electric motor depends upon two critical components, having lifetimes X and Y (in hours) which can be modelled with the joint probability density function

$$f(x, y) = \begin{cases} \alpha x e^{-x(\alpha+y)}, & x \geq 0, y \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

Find the maximum likelihood estimator for α , given a single observation on X and a single observation on Y .

Suppose $X = 3$ hours and $Y = 2$ hours. What is the maximum likelihood estimate of α ?

2. Suppose X is a normally distributed measurement from a normal distribution with mean 7 and variance σ^2 :

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x-7)^2}$$

Find the maximum likelihood estimate for σ^2 , if the observed value of X is 3, and

- (a) σ^2 can take values in the set $\{1, 3, 7, 12\}$.
- (b) σ^2 can take any value in the set $\{\sigma^2 : \sigma^2 > 0\}$.

3. Consider a gamma distributed population with parameters $\alpha = 3$ and unknown β :

$$f(x) = \frac{x^2 e^{-x/\beta}}{2\beta^3}$$

Find the maximum likelihood estimate (or estimator) for β , under the following circumstances:

- (a) The parameter space is $\{1, 5, 7\}$, and a single measurement is observed to be 3.
 - (b) The parameter space is the set of all positive real numbers, and a single measurement is observed to be 3.
 - (c) The parameter space is the set of all positive real numbers, and n independent measurements are taken: x_1, x_2, \dots, x_n .
 - (d) Suppose the true value of β is 17 and $n = 289$. Calculate the standard error of the estimator you obtained in the previous question.
4. Suppose a light bulb lasts 600 hours. Find the maximum likelihood estimate of the failure rate λ for an exponential distribution model for the bulb's lifetime, assuming
- (a) $\lambda \in \{0.001, 0.002, 0.003\}$.
 - (b) $\lambda > 0$.
5. A population of measurements is governed by the density function $f(x) = ke^{-x} I_{0 < x < \alpha}$, where α is a positive real number, and k is the normalizing constant.
- (a) Find the maximum likelihood estimator for α .
 - (b) Calculate the maximum likelihood estimate of the expected value of the next independently selected measurement, if
 - i. you observe one measurement $x = 0.5$.
 - ii. you observe two independent measurements: $x_1 = 0.5, x_2 = 0.75$.
6. Apply the `fitdistr()` function with the "weibull" parameter to the Winnipeg wind speed data. What happens if you do this without adjusting the 0's? What happens if you replace the 0's with 0.5? 0.25?
- Note that the parametrization for the Weibull distribution is different than specified in the text. In particular, the shape parameter estimated by `fitdistr()` is the reciprocal of α , and the scale parameter is $1/\lambda^{1/\alpha}$.
7. Consider the least-squares regression of the transformed noon hour wind speed against the previous midnight's wind speed as discussed in Section B.3.3. Check the diagnostics for the fitted model. Which regression assumptions do you think are violated? Are the regression coefficient estimates biased? What are the estimated standard errors? Are these biased?
8. The Poisson regression model obtained in Section B.3.2 for the cigarette butts data set assumed an intercept of 3.55. By constructing contour plots analogous to those used in Section B.3.3, find the maximum likelihood estimates for both the slope and intercept.

9. An underlying population is assumed to follow a binomial distribution with parameters $n = 10$ and unknown p . It is known that p can only take on the values 0, .1, .5, and 1.
 - (a) What is the maximum likelihood estimate for p if you have obtained one measurement whose value is 2?
 - (b) What is the maximum likelihood estimate for p if you have obtained two independent measurements whose values are 2 and 4?
 - (c) What is the maximum likelihood estimate for p if you have obtained ten independent measurements: 1, 2, 1, 2, 1, 0, 3, 5, 3 and 6.
10. Repeat the above question for the case in which p is allowed to take on any value in $[0, 1]$.
11. Continue referring to question 1, but now suppose n is unknown.
 - (a) Estimate n and p under the conditions of question 1 (c).
 - (b) Estimate n and p under the conditions of question 2 (c).
 - (c) Is it possible to estimate n and p under the conditions of 1 (a) or 2 (a)?
12. Annual numbers of infant deaths in a pediatric cardiac unit in a western Canadian hospital averaged 3 per year during the early 1990's. In 1996, after the arrival of a new surgeon, the number of infant deaths in that unit was 15.
 - (a) What probability distribution would provide a reasonable model for the data coming from the early 1990's?
 - (b) Using maximum likelihood, estimate the parameter(s) for that model, and estimate the probability that more than 14 deaths would occur in any particular year. Does this result suggest anything about the new surgeon?
 - (c) Discuss limitations of your model and the conclusions drawn from your analysis.

C

Asymptotic Results for Statistics

Generalized linear models often involve distributions other than the normal distribution. In such situations, it is often necessary, or at least better, to use maximum likelihood estimation instead of least-squares.

When a large sample of independent observations is used to fit the model, it is possible to use asymptotic distribution approximations to derive confidence intervals and hypothesis tests for model parameters. The probability concepts discussed in this section are often useful in the derivation of those approximations. For some of the results of this section, the proofs are fairly straightforward and are provided; in other cases, the proofs take us beyond the mathematical level of this book, and empirical demonstrations using simulation are used to provide supporting evidence instead. Of course, the interested reader can refer to standard probability textbooks for the derivations of most of the results given here.

C.1 Convergence in distribution

Suppose that, for $j = 1, 2, \dots, n$, $F_j(x)$ is the distribution function of a random variable X_j , and $F(x)$ is the distribution function of the random variable X . We say that X_1, X_2, \dots, X_n converges in distribution to X if

$$\lim_{n \rightarrow \infty} F_n(x) = F(x)$$

at all points where $F(x)$ is continuous.

The central limit theorem provides an important instance of this kind of convergence. In the language of the foregoing definition, the basic result says that $F(x)$ is the standard normal probability distribution function (which is continuous everywhere), and $F_n(x)$ is the probability distribution function of

$$\frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma}$$

where \bar{X}_n is the sample mean of X_1, X_2, \dots, X_n , which are the first n elements of a sequence of independent and identically distributed (i.i.d.) random variables with common mean μ and common finite standard deviation σ . This result forms the basis of much of elementary statistical analysis.

What is a little less well known is that there are actually many central limit theorems, and they do not all involve averages of identically distributed random variables. The assumption of independence can also be relaxed to a degree, but caution is warranted in such circumstances. A very important application of the central limit theorem occurs in maximum likelihood estimation; under fairly general conditions, the distribution of a maximum likelihood estimator converges to a normal distribution as the sample size increases.

Example – maximum likelihood estimation of a shape parameter for a gamma probability model

The gamma distribution is a commonly used model for lifetime measurements or time-to-failure of electronic components. It is governed by two parameters, a shape α and a scale β . For simplicity, let's consider the probability density function where $\beta = 1$:

$$f(x) = \frac{x^{\alpha-1} e^{-x}}{\Gamma(\alpha)}, \quad x \geq 0$$

where $\Gamma(\cdot)$ denotes the gamma function. We aim to demonstrate that the maximum likelihood estimator for α has an approximate normal distribution with mean α when applied to a large sample of independent gamma random variables.

For this demonstration, we will demonstrate samples of n independent gamma random variables from the above probability density function using the `rgamma()` function where the scale argument `scale` is set to 1, and the shape argument `shape`

is set to a fixed value `alpha`. If x_1, x_2, \dots, x_n is such a random sample, the log likelihood is

$$\log L(\alpha) = \sum_{j=1}^n (\alpha - 1) \log(x_j) - \sum_{j=1}^n x_j - n \log(\Gamma(\alpha)).$$

The first derivative of the log of the gamma function is called the digamma function and can be evaluated in R using the `digamma()` function. Thus, we can differentiate the log likelihood with respect to α in order to find an estimating equation for α :

$$\frac{\partial \log L(\alpha)}{\partial \alpha} = \sum_{j=1}^n \log(x_j) - n \text{digamma}(\alpha).$$

Setting this derivative to 0 and solving for α gives the maximum likelihood estimator for α . However, a numerical method is needed to find this solution. A number of options are available, but Newton's method is fast and effective in this situation.

Recall that Newton's method can be used to solve $f(\alpha) = 0$, for differentiable $f(\cdot)$ using the iteration

$$\alpha_n = \alpha_{n-1} - \frac{f(\alpha_{n-1})}{f'(\alpha_{n-1})},$$

beginning the iteration with an initial guess α_0 .

The derivative of the digamma function is the trigamma function which is also programmed in R as the `trigamma()` function. The Newton's method iteration for this problem becomes

$$\alpha_n = \alpha_{n-1} - \frac{\sum_{j=1}^n \log(x_j) - n \text{digamma}(\alpha_{n-1})}{n \text{trigamma}(\alpha_{n-1})}$$

and this is easily implemented in R using a function such as the following.

```
alphahat <- function(x) {
  n <- length(x)
  alpha <- mean(x) # initial guess for alpha using moment estimation
  for (i in 1:8) { # Newton's method applied to mle problem
    alpha <- alpha - (digamma(alpha)*n - sum(log(x)))/trigamma(alpha)/n
  }
  alpha # maximum likelihood estimate
}
```

For this problem, the expected value of a gamma random variable with scale parameter 1 is α , so since the sample mean is an estimate of the population mean or expected value, we see that the sample mean is an estimate for α .¹ Thus, the sample mean provides a starting guess.

As an example, we simulate 10 independent gamma random variables, each with shape 3 and scale 1, and apply the Newton iteration to obtain the maximum likelihood estimate as follows.

```
alpha <- 3
x <- rgamma(4, shape = alpha, scale = 1)
alphahat(x) # mle for alpha
## [1] 2.933269
```

Figure C.1 shows a histogram and a normal QQ plot for the maximum likelihood estimates coming from 1000 such simulated samples. What we observe in the figure is that the distribution of the maximum likelihood estimate is not very close to normal for such a small sample. There is some skewness, and the tail behaviour is different than would be expected from a normal distribution, one being heavier than expected, and the other being lighter.

```
Nsims <- 1000
alphahats <- numeric(Nsims)
for (j in 1:Nsims) {
  x <- rgamma(4, shape = 3, scale = 1) # alpha = 3
```

¹This is an example of parameter estimation by the method of moments.

```

    alphahats[j] <- alphahat(x)
}
par(mfrow=c(1,2), mar=c(4, 4, 1, 1))
hist(alphahats - alpha, main = " ", xlab=" ")
qqnorm(alphahats - alpha)
qqline(alphahats - alpha)

```

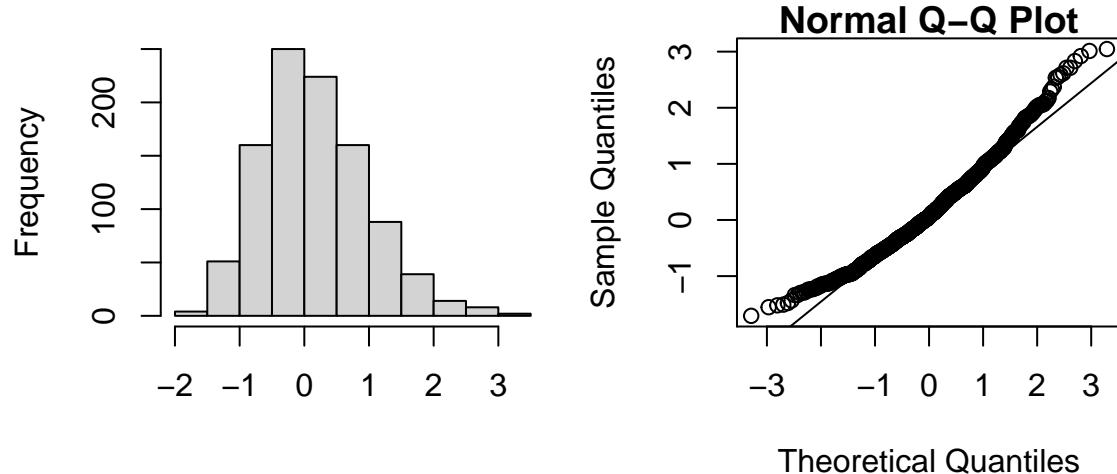


Figure C.1: Histogram and normal QQ plot of maximum likelihood estimates of shape parameter for gamma distribution with known scale parameter for 1000 simulated data sets of size 4.

On the other hand, if we simulate 1000 random samples of size 1000, we can obtain a histogram and normal QQ plot as in Figure C.2. With such a large sample, the central limit theorem effect is clear: the maximum likelihood estimator for α is asymptotically normal.

```

Nsims <- 1000
alphahats <- numeric(Nsims)
for (j in 1:Nsims) {
  x <- rgamma(1000, shape = 3, scale = 1) # alpha = 3
  alphahats[j] <- alphahat(x)
}
par(mfrow=c(1,2), mar=c(4, 4, 1, 1))
hist(alphahats - alpha, main = " ", xlab=" ")
qqnorm(alphahats - alpha)
qqline(alphahats - alpha)

```

C.2 Convergence in probability

We say that a sequence of random variables X_1, X_2, \dots, X_n converges in probability to a constant c , if for all $\varepsilon > 0$,

$$\lim_{n \rightarrow \infty} P \left(\left| \sum_{j=1}^n X_j - c \right| > \varepsilon \right) = 0.$$

The Law of large numbers is an important example of this concept.
Suppose X_1, X_2, \dots, X_n are uncorrelated random variables where

- $\mu_j = E[X_j]$.
- $\lim_{n \rightarrow \infty} \sum_{j=1}^n \mu_j = c$.

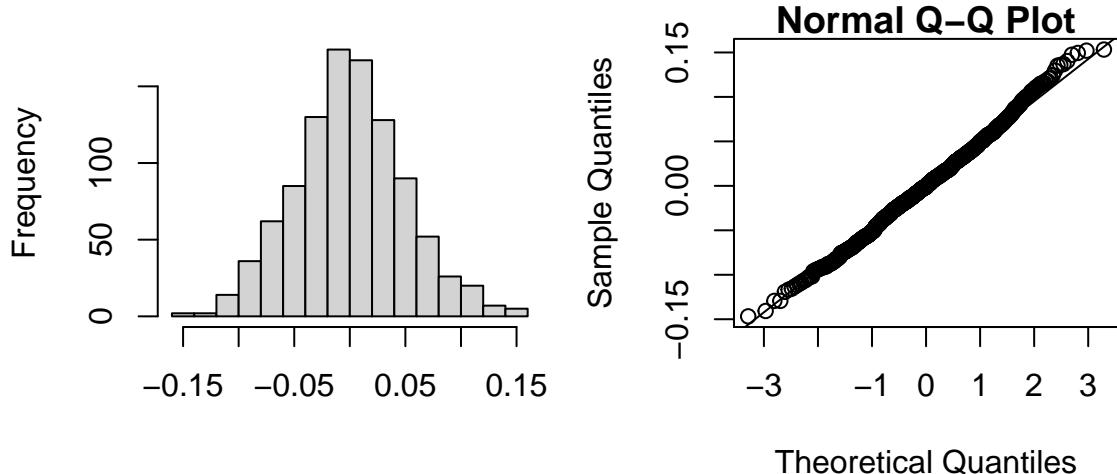


Figure C.2: Histogram and normal QQ plot of maximum likelihood estimates of shape parameter for gamma distribution with known scale parameter for 1000 simulated data sets of size 1000.

- $\lim_{n \rightarrow \infty} \text{Var}(\sum_{j=1}^n X_j) = 0$.

Under these conditions,

$$\lim_{n \rightarrow \infty} \sum_{j=1}^n X_j = c$$

in probability.

The proof of this statement follows from Chebyshev's inequality which states that, for any $\varepsilon > 0$,

$$P(|X - E[X]| > \varepsilon) \leq \frac{\text{Var}(X)}{\varepsilon^2}.$$

An important special case of the law of large numbers is concerned with the average of uncorrelated random variables having the same mean and variance. That is,

$$\lim_{n \rightarrow \infty} P\left(\left|\frac{1}{n} \sum_{j=1}^n X_j - \mu\right| > \varepsilon\right) = 0.$$

when $E[X_j] = \mu$ and $\text{Var}(X_j) = \sigma^2$, for all j .

As in the case of the central limit theorem, there are actually many laws of large numbers, where the assumptions of the basic result can be relaxed to some degree.

Example - consistency of the scale parameter estimate

We can demonstrate that the maximum likelihood estimator for α in the gamma model with $\beta = 1$ appears to converge in probability to the truth using simulations as before. This time, we consider a sequence of increasing sample sizes: 4, 20, 100 and 500, and we plot histograms of the absolute values of the errors in the estimates for 1000 simulated samples of each size. These plots appear in Figure C.3.

```
Nsims <- 1000
par(mfrow=c(2, 2), mar=c(5, 4, 1, 1))
for (n in c(4, 20, 100, 500)) {
  alphahats <- numeric(Nsims)
  for (j in 1:Nsims) {
    x <- rgamma(n, shape = 3, scale = 1)
    alphahats[j] <- alphahat(x)
  }
  hist(abs(alphahats - alpha), main = " ", xlab=" ", xlim = c(-1, 6),
       sub = paste("sample size: ", n))
}
```

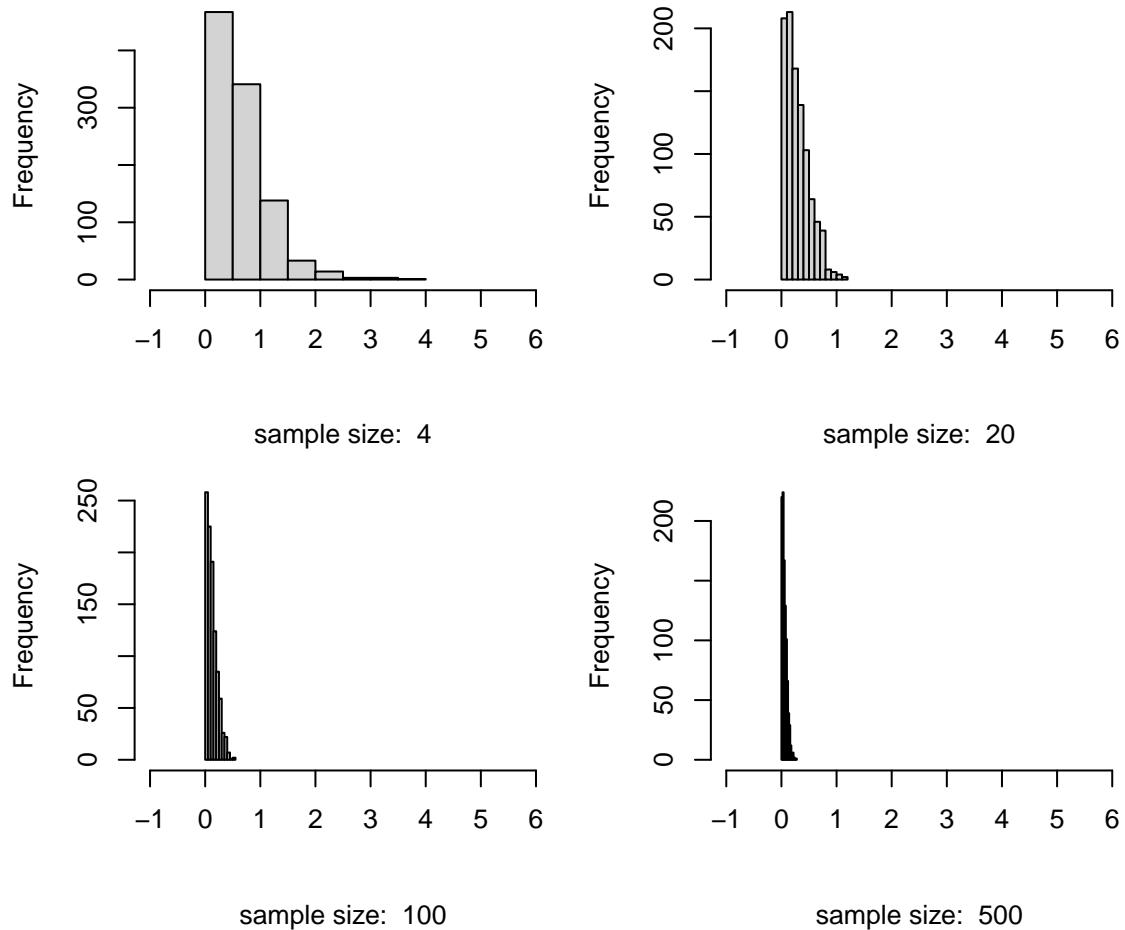


Figure C.3: Histograms of absolute deviations of the maximum likelihood estimates of the shape parameter for gamma distribution with known scale parameter for 1000 simulated data sets of sizes 4, 20, 100 and 500, respectively.

The figure provides an empirical demonstration of the convergence of the maximum likelihood estimate to α . As the sample size increases, it is clear that the probability that absolute value of the error will exceed 0.5, for example, decreases; when $n = 4$, this happens over half of the time, and for $n = 100$ it happens very occasionally, and when $n = 500$, the probability of such an occurrence is very close to 0.

C.2.1 Slutsky's theorem

Suppose A_n converges in probability to a constant a , suppose B_n converges in probability to another constant b and X_n converges in distribution to a random variable X , then $A_n X_n + B_n$ converges in distribution to $aX + b$.

C.3 Mathematical and statistical properties of concave functions

A function $f(x)$ is said to be concave, if, for $x_1 \neq x_2$,

$$f(\alpha x_1 + (1 - \alpha)x_2) \geq \alpha f(x_1) + (1 - \alpha)f(x_2), \quad (\text{C.1})$$

for any $\alpha \in [0, 1]$.

For smooth enough $f(x)$, concavity has a simple interpretation in terms of the second derivative, that is, $f''(x) \leq 0$.

C.3.1 Concave functions cannot have more than one maximum

Concave functions play an important role in optimization. In particular, suppose that a function $f(x)$ is *strictly* concave, meaning that the weak inequality at (C.1) is replaced by

$$f(\alpha x_1 + (1 - \alpha)x_2) > \alpha f(x_1) + (1 - \alpha)f(x_2). \quad (\text{C.2})$$

Such a function cannot have more than one maximizer. To see this, suppose that $x_1 \neq x_2$ both maximize a strictly concave function $f(x)$. This means that $f(x_1) = f(x_2)$ and all other values of $f(x)$ can be no larger than these values. However, plugging $\alpha = 1/2$ into (C.2) immediately leads to

$$f\left(\frac{x_1 + x_2}{2}\right) > \frac{1}{2}f(x_1) + \frac{1}{2}f(x_2) = f(x_1).$$

Therefore, we have found a value of x for which $f(x)$ exceeds the supposed maximum value, leading to a contradiction. There must be no more than 1 maximizer of a strictly concave function.

C.3.2 Jensen's inequality

Another useful result is Jensen's inequality. If $f(x)$ is a concave function, then

$$E[f(X)] \leq f(E[X]). \quad (\text{C.3})$$

The inequality becomes strict in the case where $f(x)$ is strictly concave, and where X is not constant.

Example – variances are positive

A simple special case of this inequality relates to the variance of a nonconstant random variable X . Note that $f(x) = -x^2$ is strictly concave. Therefore,

$$E[-X^2] < -(E[X])^2.$$

Multiplying through by -1 , we have

$$E[X^2] > (E[X])^2.$$

That is, $\text{Var}(X) > 0$.

Example – logarithms of random variables

Since the log function is strictly concave on the positive half-line, we have that, for any nonconstant positive-valued random variable X ,

$$E[\log(X)] > \log(E[X]). \quad (\text{C.4})$$

C.4 Consistency of maximum likelihood estimation

The following results are helpful in proving that maximum likelihood estimators are consistent, that is, they converge in probability to the true parameter values, as the sample size becomes infinitely large.

1. If A and B are events with probability exceeding $1 - \delta_A$ and $1 - \delta_B$, respectively, then

$$P(A \cap B) > 1 - \delta_A - \delta_B.$$

To see this, note that

$$P(A \cap B) = P(A) + P(B) - P(A \cup B) > 1 - \delta_A + 1 - \delta_B - P(A \cup B) \geq 1 - \delta_A - \delta_B.$$

since we know that $P(A \cup B) \leq 1$.

2. If A_1, A_2, \dots, A_m are events with probabilities exceeding $1 - \delta_1, 1 - \delta_2, \dots, 1 - \delta_m$, respectively, then

$$P(A_1 \cap A_2 \cap \dots \cap A_m) > 1 - (\delta_1 + \delta_2 + \dots + \delta_m).$$

This result can be proved by induction, using the first result repeatedly.

3. Suppose y_1, y_2, \dots, y_n are i.i.d. with common probability density function $f(y; \theta_0)$, and suppose $E[|\log(f(y_1; \theta))|]$ is finite. Then $\frac{1}{n} \sum_{j=1}^n \log(f(y_j; \theta))$ converges in probability to the expected value of $\log(f(y_1; \theta))$, where all expectations are taken with respect to $f(y; \theta_0)$.²

This result follows from the strong law of large numbers, which is beyond the scope of this book, but if finiteness of the variance of $\log(f(y_1; \theta))$ is also assumed, the weak law of large numbers based on Chebyshev's theorem applies. The central point to observe is that the quantity $\frac{1}{n} \sum_{j=1}^n \log(f(y_j; \theta))$ is the likelihood function of the data divided by the sample size, and this quantity is a sample average, an average of the contributions of each individual observation to the likelihood.

4. Under the conditions stated in point 3,

$$E[\log(f(y_1; \theta_0)) > E[\log(f(y_1; \theta))]],$$

and for all $\theta \neq \theta_0$.

Jensen's inequality can be used to obtain this result. The trick is to apply (C.4) to first see that

$$E \left[\log \left(\frac{f(y; \theta)}{f(y; \theta_0)} \right) \right] < \log \left(E \left[\frac{f(y; \theta)}{f(y; \theta_0)} \right] \right).$$

Because the expectations are with respect to $f(y; \theta_0)$, the right hand side simplifies to the log of $\int f(y; \theta) dy$ which is identically 0, since the complete integral of a probability density function is 1. The left hand side is mathematically equivalent to

$$E[\log(f(y_1; \theta)) - \log(f(y_1; \theta_0))]$$

so we can deduce that

$$E[\log(f(y_1; \theta))] - E[\log(f(y_1; \theta_0))] < 0$$

from which the result follows.

Theorem – consistency of maximum likelihood estimation.

Under the conditions stated in point 3,

$$\lim_{n \rightarrow \infty} P \left(\sum_{j=1}^n \log(f(y_j; \theta)) < \sum_{j=1}^n \log(f(y_j; \theta_0)) \right) = 1$$

for all $\theta \neq \theta_0$. This means that, as the sample size becomes infinitely large, the probability of arriving at the true value of the unknown parameter by maximizing the log likelihood approaches 1.

Proof

The proof will proceed by contradiction. Let $\varepsilon_1, \varepsilon_2, \delta_1, \delta_2, \delta_3$ be given positive constants and suppose that

$$\lim_{n \rightarrow \infty} P \left(\sum_{j=1}^n \log(f(y_j; \theta^*)) \geq \sum_{j=1}^n \log(f(y_j; \theta_0)) \right) = 1 \quad (\text{C.5})$$

for some $\theta^* \neq \theta_0$.

Define $\ell_j(\theta) = \log(f(y_j; \theta))$ and $\ell(\theta) = \sum_{j=1}^n \ell_j(\theta)$. That is, $\ell(\theta)$ is the log likelihood evaluated at θ and $\ell_j(\theta)$ is the contribution of the j th observation to the log likelihood.

For each sample size n , define events of the form

$$A_n = \left\{ \frac{1}{n} \ell(\theta^*) - E[\ell_1(\theta^*)] < \varepsilon_1 \right\},$$

$$B_n = \left\{ \frac{1}{n} \ell(\theta_0) - E[\ell_1(\theta_0)] > -\varepsilon_2 \right\},$$

and

$$C_n = \{\ell(\theta^*) - \ell(\theta_0) \geq \varepsilon_3\}.$$

By result 3, there exist N_1 and N_2 such that

$$P(A_n) > 1 - \delta_1 \text{ and } P(B_n) > 1 - \delta_2$$

²That is, the expected value is $\int \log(f(y_1; \theta)) f(y; \theta_0) dy$. Note that θ need not be equal to θ_0 here.

for all $n > N_1$ and $n > N_2$, respectively, and by assumption (C.5), there exists N_3 such that $P(C_n) > 1 - \delta_3$ for all $n > N_3$. By result 2, we have

$$P(A_n \cap B_n \cap C_n) > 1 - (\delta_1 + \delta_2 + \delta_3)$$

for all $n > \max\{N_1, N_2, N_3\}$.

Observe that

$$C_n = \{\ell(\theta^*) - nE[\ell_1(\theta_0)] \geq \ell(\theta_0) - nE[\ell_1(\theta_0)]\}$$

which can be re-written as

$$C_n = \left\{ \frac{1}{n} \ell(\theta^*) - E[\ell_1(\theta^*)] \geq \frac{1}{n} \ell(\theta_0) - E[\ell_1(\theta_0)] \right\}$$

Therefore,

$$\begin{aligned} P(A_n \cap B_n \cap C_n) &\leq P(A_n \cap B_n \cap \{E[\ell_1(\theta^*)] + \varepsilon_1 - E[\ell_1(\theta_0)] \geq E[\ell_1(\theta_0)] - \varepsilon_2 - E[\ell_1(\theta_0)]\}) \\ &\leq P(A_n \cap B_n \cap \{E[\ell_1(\theta^*)] - E[\ell_1(\theta_0)] \geq -\varepsilon_1 - \varepsilon_2\}) \\ &\leq P(E[\ell_1(\theta^*)] - E[\ell_1(\theta_0)] \geq -\varepsilon_1 - \varepsilon_2) > 1 - (\delta_1 + \delta_2 + \delta_3) \end{aligned}$$

for all $n > N$. Since $\varepsilon_1, \varepsilon_2, \delta_1, \delta_2, \delta_3$ could be made to be arbitrarily small, we are forced to conclude that

$$\lim_{n \rightarrow \infty} P(E[\ell_1(\theta^*)] \geq E[\ell_1(\theta_0)]) = 1.$$

This contradicts result 4(a), leading us to conclude that assumption (C.5) must be false and that result 4(b) must be true.

C.5 Maximum likelihood estimators are approximately normal in large samples

We saw empirical evidence of the asymptotic normality of maximum likelihood estimators in Section C.1. In this section, we will supply some of the mathematics to show why this is not an accident, and that the result is tied to a version of the central limit theorem. We will work in the context of a random sample y_1, y_2, \dots, y_n coming from a population governed by a probability density function $f(y; \theta)$.

The actual result requires the statement of regularity conditions which take us well outside the scope of this book, but the essence of the result is that, under conditions which are somewhat more restrictive than those used in the preceding section to prove consistency of maximum likelihood estimation, we can show that, as the sample size n increases, the distribution of the maximum likelihood estimator converges to that of a normal random variable with mean θ_0 (the true parameter value) and variance \mathcal{I}/n where

$$\mathcal{I} = \text{Var}\left(\frac{\partial}{\partial \theta} \log(f(y; \theta_0))\right).$$

\mathcal{I} is called the Fisher information. In other words, if $\hat{\theta}$ denotes the maximum likelihood estimator for the true parameter value θ_0 , then

$$\frac{\sqrt{n}(\hat{\theta} - \theta_0)}{\sqrt{\mathcal{I}}}$$

converges in distribution to a standard normal random variable.

We now give a rough sketch of the proof of this result, glossing over some of the technicalities. We let $\ell_j(\theta)$ denote the log likelihood associated with the j th observation, so that

$$\ell(\theta) = \sum_{j=1}^n \ell_j(\theta)$$

is the log likelihood for the entire sample. Assuming sufficient smoothness in the probability density function with respect to θ , we can expand the first derivative of the log likelihood evaluated at a value θ in Taylor series about θ_0 :

$$\frac{\partial \ell}{\partial \theta}(\theta) = \frac{\partial \ell}{\partial \theta}(\theta_0) + (\theta - \theta_0) \frac{\partial^2 \ell}{\partial \theta^2}(\theta) + \frac{(\theta - \theta_0)^2}{2} \frac{\partial^3 \ell}{\partial \theta^3}(\theta^*)$$

where θ^* is between θ and θ_0 . The left-hand-side of this expression is defined to be 0 for $\theta = \hat{\theta}$, so If $\hat{\theta}$ is close enough to θ_0 , and the third derivative of the log likelihood is bounded, we could argue that

$$(\hat{\theta} - \theta_0) \doteq \frac{\frac{\partial \ell}{\partial \theta}(\theta_0)}{\frac{\partial^2 \ell}{\partial \theta^2}(\theta_0)}. \quad (\text{C.6})$$

Because of the consistency of maximum likelihood estimation, if n is chosen to be large enough, $\hat{\theta} - \theta_0$ can indeed be made arbitrarily small with arbitrarily high probability so the approximation given by (C.6) is reasonable.

Multiplying the numerator of the right-hand-side of (C.6) by $1/\sqrt{n}$ gives

$$\frac{1}{\sqrt{n}} \frac{\partial \ell}{\partial \theta}(\theta_0) = \frac{1}{\sqrt{n}} \sum_{j=1}^n \frac{\partial \ell_j}{\partial \theta}(\theta_0)$$

where we are summing random variables whose expected value is $E[\frac{\partial \ell}{\partial \theta}(\theta_0)] = 0$ and whose variance is \mathcal{I} . Thus, by the central limit theorem, the quantity on the right-hand-side of (C.6) is converging in distribution to a normal random variable with mean 0 and variance \mathcal{I} .

Multiplying the denominator by $-1/n$ gives

$$-\frac{1}{n} \frac{\partial^2 \ell}{\partial \theta^2}(\theta_0) = -\frac{1}{n} \sum_{j=1}^n \frac{\partial^2 \ell_j}{\partial \theta^2}(\theta_0)$$

which converges in probability to the negative expected value of the second derivative of the log likelihood or equivalently, to the expected value of the square of the first derivative.³ In other words, we have convergence in probability to \mathcal{I} , the Fisher information which is the variance of the first derivative of the log likelihood.

Putting this all together, we can use Slutsky's theorem (see Section C.2.1) to see that

$$\sqrt{n}(\hat{\theta} - \theta_0) \doteq -\frac{\sqrt{n}/n}{1/n} \frac{\frac{\partial \ell}{\partial \theta}(\theta_0)}{\frac{\partial^2 \ell}{\partial \theta^2}(\theta_0)}.$$

converges in distribution to a normal random variable X with mean 0 and variance \mathcal{I} divided by \mathcal{I} . Note that X has variance \mathcal{I} , so X/\mathcal{I} has variance $\mathcal{I}/\mathcal{I}^2 = 1/\mathcal{I}$. Therefore, the asymptotic or large-sample distribution of $\sqrt{n}(\hat{\theta} - \theta_0)$ is normal with mean 0 and variance $1/\mathcal{I}$. This also means that $\hat{\theta}$ has a large sample distribution which is approximately normal with mean θ_0 and variance $\frac{1}{n\mathcal{I}}$.

Example – exponential distribution

If n independent observations have been taken from an exponential distribution with true parameter λ , we have seen that the MLE for λ is $1/\bar{y}$. The log likelihood function is

$$\log(\lambda) - \lambda y$$

so taking two derivatives leads to

$$\mathcal{I} = \frac{1}{\lambda^2}.$$

According to the theory just developed, $1/\bar{y}$ is approximately normal with mean λ and variance $\frac{1}{n\mathcal{I}} = \lambda^2/n$.

The following is an illustration of this result based on repeated simulations of samples of 1000 exponential variables with true parameter $\lambda_0 = 5$. The code computes the maximum likelihood estimates for each of the 1000 samples, and outputs the average of the MLEs as well as the variance. Normal QQ-plots of the MLEs are displayed in Figure C.4.

```
y <- matrix(rexp(1000*1000, rate = 5), nrow = 1000) # lambda = 5
y <- data.frame(y) # samples of size 1000
bary <- sapply(y, mean)
yMLE <- 1/bary
mean(yMLE)

## [1] 5.003844

var(yMLE)

## [1] 0.02513651

qqnorm(yMLE)
qqline(yMLE)
```

The average of the MLEs is very near the expected value of 5 while the variance should be near $5^2/1000 = .025$, and it is. The normal QQ-plot has a very clear linear pattern. This indicates that the maximum likelihood estimator is approximately normal for a sample of size 1000.

³See results (7.2) and (7.3) from Chapter 7 for more on this point.

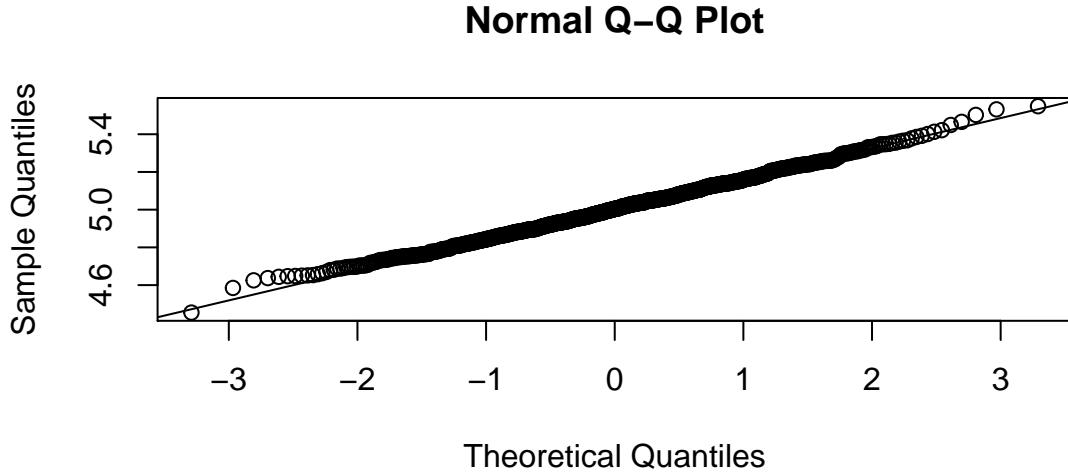


Figure C.4: Normal QQ-plot of the maximum likelihood estimates coming from 1000 large samples of exponential random variables with true parameter 5.

Exercises

- Suppose a sample of size n is taken from a normal population with mean 10 and variance 50. Let \bar{X} denote the average of the sample measurements. Find $P(7 < \bar{X} < 12)$ for the cases $n = 1, n = 2, n = 5, n = 25$ and $n = 100$.
- Repeat the previous question for a gamma population, obtaining approximate results in cases where the central limit theorem applies.
- The following code can be used to conduct 3 simulations. The first is a simulation of the distribution of a single uniform random variable. The second is the simulation of the average of 2 independent uniform random variables. The third is the simulation of the average of 4 independent uniform random variables.

```
x <- runif(10000)
qqnorm(x)
qqline(x)

xbar <- (runif(10000) + runif(10000))/2
qqnorm(xbar)
qqline(xbar)

xbar <- (runif(10000)+runif(10000)+runif(10000)+runif(10000))/4
qqnorm(xbar)
qqline(xbar)
```

- Run each simulation and decide whether the random variable is approximately normally distributed or not.
 - Construct code to simulate averages of 6, 10 and 15 independent uniforms. Is the normal approximation accurate in any of these cases?
 - Repeat the above simulations using `rexp` in place of `runif`. If the normal approximation does not look accurate for any of those sample sizes, try $n = 30$ as well.
- The probability density function for a gamma random variable with shape α and scale β is given by

$$f(x) = \frac{x^{\alpha-1} e^{-x/\beta}}{\Gamma(\alpha)\beta^\alpha}, \quad x \geq 0.$$

- (a) Find the expected value of X in terms of α and β .
 - (b) Deduce that the expected value of X is α when X is a gamma random variable with shape parameter α and scale parameter $\beta = 1$.
 - (c) Compare the expected value of X for the cases where $\beta > 1$, $\beta = 1$ and $\beta < 1$. Deduce why β is referred to as the *scale* parameter of the gamma distribution.
 - (d) Use the `curve()` function to plot the probability density function of gamma random variables having scale parameter 1 and shape parameters $\alpha = 0.5, 1, 2, 10$. Deduce why α is referred as the *shape* parameter of the gamma distribution.
5. (a) Show that the log likelihood function for independent observations x_1, x_2, \dots, x_n coming from a gamma distribution with parameters α and β leads to an estimating equation for α of the form

$$\frac{\partial \log L(\alpha)}{\partial \alpha} = \sum_{j=1}^n \log(x_j) - ndigamma(\alpha) - n \log(\alpha) + n \log(\bar{x})$$

and an estimating equation for β of the form

$$\beta = \frac{\alpha}{\bar{x}}.$$

- (b) Use the results obtained in (a) to modify the `alphahat()` function of Section C.1 so that it returns an estimate of α for this more general gamma model. Also, write a function called `betahat()` which takes the same arguments as `alphahat()` and which calls `alphahat()` once in order to provide an estimate of β .
 - (c) Using your version of the `alphahat()` re-run the two simulation studies of Section C.1 to verify that when estimating both α and β by maximum likelihood, the small sample estimate is not normally distributed, and the large sample estimate is approximately normally distributed.
 - (d) Modify the simulation study code used in (c) so that it employs `betahat()` instead of `alphahat()`. Observe that in small samples the distribution of the maximum likelihood estimate of β is not nearly normally distributed, but in large samples, a normal approximation is accurate.
 - (e) Modify the simulation study code used in Section C.2 to empirically demonstrate the consistency of the maximum likelihood estimator for β .
6. By finding the second derivative of $f(x) = -x^2$, show that $f(x)$ is strictly concave.
7. By finding the second derivative of $f(x) = \log(x)$, show that $f(x)$ is strictly concave on the positive half-line.
8. Show that if X is a random variable with mean μ , and ε is a positive number, then

$$P(|X - \mu| > \varepsilon) \leq \frac{E[|X - \mu|]}{\varepsilon}.$$

This is Markov's inequality.

- 9. Run the code demonstrating the asymptotic normality for the exponential distribution MLE using samples of 100, 50, 20, 10, 5 and 2. Based on your findings, how large a sample would you recommend before judging the MLE to be approximately normally distributed?
- 10. Refer to the previous question. Modify the code so that Poisson random variables are simulated instead of exponential variables. Verify that the MLE has an asymptotic normal distribution, and calculate the information in order to compare the variance of the simulated MLEs with the asymptotic variance.

D

Review of Linear Algebra

This appendix contains a review of linear algebra, covering the most elementary concepts which you might have learned about in an earlier course. In our review, we will consider the algebra of vectors and matrices. We will also review the matrix determinant, trace and eigenvalues and eigenvectors. We will also review linear spaces and their properties.

D.1 Vectors

Vectors are lists of numbers. The individual numbers are referred to as elements. The following are examples of column vectors:

$$\mathbf{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} -1 \\ 2 \\ 0 \end{bmatrix}, \mathbf{w} = \begin{bmatrix} 3 \\ 0.1 \\ -7.2 \\ 5 \end{bmatrix}, \mathbf{z} = \begin{bmatrix} 4 \\ 0 \\ 0 \end{bmatrix},$$

Row vectors are represented horizontally, and they are defined as the transpose of column vectors. We will use $^\top$ to denote the transpose operation. The following are examples of row vectors:

$$\mathbf{x}^\top = [1 \ 1], \mathbf{y}^\top = [-1 \ 2 \ 0]$$

The length of a vector is its number of elements. For example, \mathbf{z} has length 4, and \mathbf{y}^\top has length 3.

D.1.1 Unit Vectors

Unit vectors are vectors whose squared elements sum to 1. For example,

$$\mathbf{x} = \begin{bmatrix} 1/2 \\ \sqrt{3}/2 \end{bmatrix}$$

is a unit vector because $\frac{1}{2^2} + \frac{3}{2^2} = 1$. The vector

$$\mathbf{w} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

is not a unit vector because $1^2 + 1^2 \neq 1$.

D.1.2 Vector Algebra – Addition

Vectors having the same length can be added together, elementwise. For example, if

$$\mathbf{x} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \text{ and } \mathbf{y} = \begin{bmatrix} 5 \\ 9 \end{bmatrix},$$

then

$$\mathbf{x} + \mathbf{y} = \begin{bmatrix} 6 \\ 12 \end{bmatrix}.$$

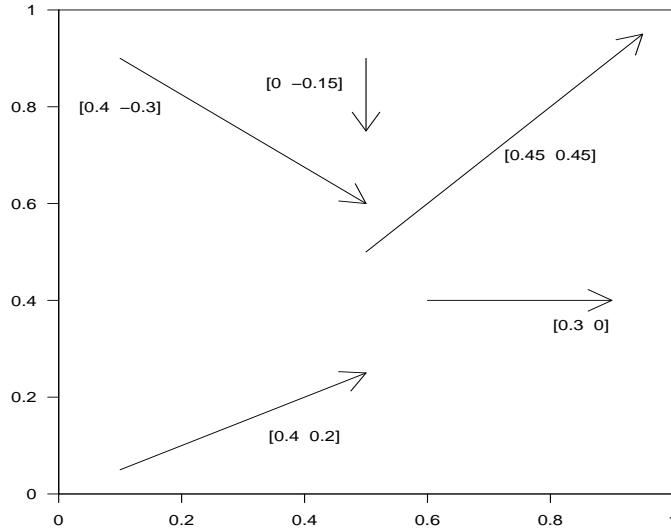


Figure D.1: Graphical display of vectors in 2 dimensions.

D.1.3 Scalar multiplication

A vector can be multiplied by a single number. The result is a vector of the same length whose elements are the products of the original elements with that single number. For example, if

$$\mathbf{x} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \text{ and } \mathbf{y} = \begin{bmatrix} 5 \\ 9 \\ -4 \end{bmatrix},$$

then

$$7\mathbf{x} = \begin{bmatrix} 7 \\ 21 \end{bmatrix} \text{ and } -2\mathbf{y} = \begin{bmatrix} -10 \\ -18 \\ 8 \end{bmatrix}.$$

D.1.4 Inner Product

The inner product (or dot product) is the usual way of multiplying vectors which have the same length. It is defined as the sum of the products of the corresponding elements. The result is a single number (a scalar).

The inner product of two vectors \mathbf{x} and \mathbf{y} is usually denoted by the product of the transpose of \mathbf{x} with \mathbf{y} :

$$\mathbf{x}^T \mathbf{y} = \sum_{i=1}^n x_i y_i.$$

Here, x_i denotes the i th element of the vector \mathbf{x} which is assumed to have length n . Another notation for the inner product is

$$\langle \mathbf{x}, \mathbf{y} \rangle.$$

Inner product – example

If

$$\mathbf{x} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \text{ and } \mathbf{y} = \begin{bmatrix} 5 \\ 9 \end{bmatrix},$$

then

$$\mathbf{x}^T \mathbf{y} = 5 + 27 = 32.$$

Note that

$$\mathbf{x}^T \mathbf{y} = \mathbf{y}^T \mathbf{x}.$$

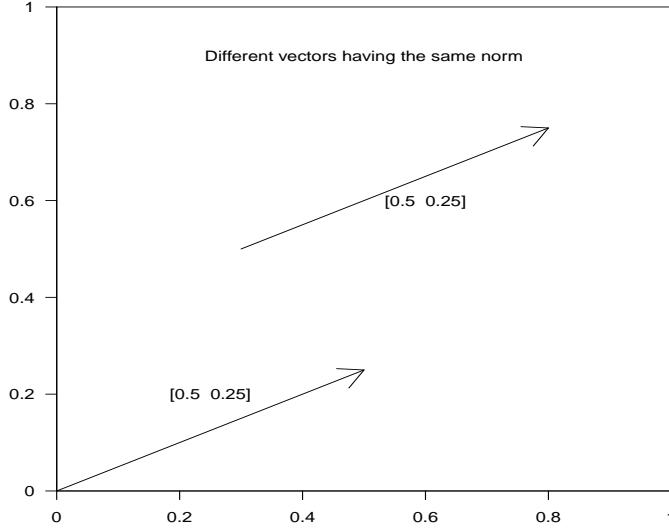


Figure D.2: The Norm is the Distance Between Vector Endpoints.

D.1.5 Orthogonal Vectors

Vectors are said to be orthogonal if their inner product is 0. In two dimensions, this occurs if the two vectors are perpendicular to each other. For example, if

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \text{ and } \mathbf{y} = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix},$$

then

$$\mathbf{x}^T \mathbf{y} = -1 + 1 = 0.$$

We conclude that \mathbf{x} and \mathbf{y} are orthogonal.

D.1.6 Vector Norm

The Euclidean norm of a vector \mathbf{x} , denoted by $\|\mathbf{x}\|$, is the square root of the inner product of the vector with itself:

$$\|\mathbf{x}\| = \sqrt{\mathbf{x}^T \mathbf{x}} = \sqrt{\sum_{i=1}^n x_i^2}.$$

For example, if

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix},$$

then

$$\mathbf{x}^T \mathbf{x} = 1 + 4 = 5$$

and

$$\|\mathbf{x}\| = \sqrt{5}.$$

Note that unit vectors have a norm of 1.

D.1.7 Projection

The projection of a vector \mathbf{x} onto a vector \mathbf{y} is defined as

$$\text{proj}_{\mathbf{y}}(\mathbf{x}) = \frac{\mathbf{y}^T \mathbf{x} \mathbf{y}}{\|\mathbf{y}\|^2}.$$

Note that if \mathbf{x} and \mathbf{y} are orthogonal,

$$\text{proj}_{\mathbf{y}}(\mathbf{x}) = \mathbf{0}.$$

If $\mathbf{x} = \mathbf{y}$, then

$$\text{proj}_{\mathbf{y}}(\mathbf{x}) = \mathbf{y}.$$

D.2 Matrices

Matrices are rectangular arrays of numbers. A matrix with m rows and n columns is said to be an $m \times n$ matrix. The numbers m and n are the dimensions of the matrix. For example,

$$\mathbf{A} = \begin{bmatrix} 3 & -4 \\ 5 & 0 \\ 1 & 2 \end{bmatrix}$$

is a 3×2 matrix, since it consists of 3 rows and 2 columns. The dimensions of A are 3 and 2. The (i, j) element of a matrix \mathbf{B} is denoted by $\mathbf{B}_{i,j}$ and is located in the i th row and j th column of \mathbf{B} . For example, $A_{2,1} = 5$.

D.2.1 Diagonal Matrices

A matrix \mathbf{A} is called a diagonal matrix if the only nonzero elements of \mathbf{A} are of the form $A_{i,i}$. These entries are the diagonal entries of \mathbf{A} . For example,

$$\mathbf{A} = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}$$

is a diagonal matrix.

D.2.2 Matrix Transpose

The matrix \mathbf{B} is said to be the transpose of a matrix \mathbf{A} if $\mathbf{B}_{j,i} = \mathbf{A}_{i,j}$ for all i and j . For example, if

$$\mathbf{A} = \begin{bmatrix} 3 & -4 \\ 5 & 0 \\ 1 & 2 \end{bmatrix}$$

then

$$\mathbf{B} = \mathbf{A}^T = \begin{bmatrix} 3 & 5 & 1 \\ -4 & 0 & 2 \end{bmatrix}$$

is the transpose of \mathbf{A} .

D.2.3 Symmetric Matrices

A matrix \mathbf{A} is said to be symmetric if $\mathbf{A}^T = \mathbf{A}$. If \mathbf{A} is symmetric, then $\mathbf{A}_{i,j} = \mathbf{A}_{j,i}$ for all i and j . For example,

$$\mathbf{A} = \begin{bmatrix} 3 & 5 \\ 5 & 2 \end{bmatrix}$$

is a symmetric matrix.

D.2.4 Scalar Multiplication

If \mathbf{A} is a matrix and b is a number, then a matrix \mathbf{C} can be defined as

$$\mathbf{C} = b\mathbf{A}$$

where the (i, j) element of \mathbf{C} is $\mathbf{C}_{i,j} = b\mathbf{A}_{i,j}$. In other words, each element of \mathbf{C} is the product of the corresponding element of \mathbf{A} multiplied by b . For example, if

$$\mathbf{A} = \begin{bmatrix} 3 & -4 \\ 5 & 0 \\ 1 & 2 \end{bmatrix}$$

and $b = -1$, then

$$\mathbf{C} = b\mathbf{A} = -\mathbf{A} = \begin{bmatrix} -3 & 4 \\ -5 & 0 \\ -1 & -2 \end{bmatrix}$$

D.2.5 Matrix Addition

If \mathbf{A} and \mathbf{B} are $m \times n$ matrices, then their sum can be defined as

$$\mathbf{C} = \mathbf{A} + \mathbf{B}$$

where the (i, j) element of \mathbf{C} is $\mathbf{C}_{i,j} = \mathbf{A}_{i,j} + \mathbf{B}_{i,j}$. This is element-wise addition. For example, if

$$\mathbf{A} = \begin{bmatrix} 3 & -4 \\ 5 & 0 \\ 1 & 2 \end{bmatrix} \text{ and } \mathbf{B} = \begin{bmatrix} 2 & 3 \\ 1 & 8 \\ -1 & -2 \end{bmatrix},$$

then

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} 5 & -1 \\ 6 & 8 \\ 0 & 0 \end{bmatrix}.$$

More addition examples

If \mathbf{P} is the 3×2 matrix whose elements are all 0's, then we say that $\mathbf{P} = \mathbf{0}$ and

$$\mathbf{A} + \mathbf{0} = \mathbf{A}$$

Also,

$$\mathbf{A} + (-\mathbf{A}) = \mathbf{0}.$$

The matrix $-\mathbf{A}$ is the additive inverse of \mathbf{A} . Note that there is sometimes also a multiplicative inverse which will be defined later. The additive inverse always exists, but the multiplicative inverse does not always exist. A matrix that has a multiplicative inverse is said to be invertible. A matrix that has no multiplicative inverse is said to be singular.

D.2.6 Matrix Multiplication

There are several ways that one could define matrix multiplication. Here, we will use the most important definition and the one that is almost always assumed when one talks about “multiplying matrices”.

Given an $\ell \times m$ matrix \mathbf{A} and an $m \times n$ matrix \mathbf{B} , then product \mathbf{AB} is an $\ell \times n$ matrix \mathbf{C} whose (i, j) element is the inner product of the i th row of \mathbf{A} with the j th column of \mathbf{B} :

$$\mathbf{C}_{i,j} = \sum_{k=1}^m \mathbf{A}_{i,k} \mathbf{B}_{k,j}.$$

Note that matrix multiplication cannot be defined in this way if the dimensions of \mathbf{A} and \mathbf{B} do not conform. That is, the number of columns of \mathbf{A} must match the number of rows of \mathbf{B} in order for \mathbf{AB} to be defined.

Example

$$\mathbf{A} = \begin{bmatrix} 3 & -4 \\ 5 & 0 \\ 1 & 2 \end{bmatrix} \text{ and } \mathbf{B} = \mathbf{A}^\top = \begin{bmatrix} 3 & 5 & 1 \\ -4 & 0 & 2 \end{bmatrix}$$

so

$$\mathbf{AB} = \begin{bmatrix} 25 & 15 & -5 \\ 15 & 25 & 5 \\ -5 & 5 & 5 \end{bmatrix}.$$

In this case, $\mathbf{B} = \mathbf{A}^\top$, so we have an example of the fact that \mathbf{AA}^\top is always a symmetric matrix.

D.2.7 Matrix Identity

A diagonal matrix whose diagonal entries are all 1 is called an identity matrix. It is usually denoted by the symbol \mathbf{I} . Sometimes, we will use the symbol \mathbf{I}_n to denote the $n \times n$ identity matrix. For example, the 3×3 identity matrix is given by

$$\mathbf{I}_3 = \mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The identity matrix has the property that multiplication with a matrix \mathbf{A} always returns the matrix \mathbf{A} :

$$\mathbf{IA} = \mathbf{A} = \mathbf{AI}.$$

D.2.8 Matrix Inverse

A matrix \mathbf{A} is said to have an inverse (or to be invertible) if

$$\mathbf{AB} = \mathbf{BA} = \mathbf{I}$$

for some matrix \mathbf{B} . \mathbf{B} is called the inverse of \mathbf{A} and

$$\mathbf{B} = \mathbf{A}^{-1}.$$

Check that if

$$\mathbf{A} = \begin{bmatrix} 3 & -4 \\ 1 & 2 \end{bmatrix},$$

then

$$\mathbf{A}^{-1} = \begin{bmatrix} 0.2 & 0.4 \\ -0.1 & 0.3 \end{bmatrix}.$$

D.2.9 Relations involving Transposes and Inverses

The inverse of \mathbf{P}^\top is given by

$$(\mathbf{P}^\top)^{-1} = (\mathbf{P}^{-1})^\top$$

provided that \mathbf{P} is invertible. Note that if \mathbf{P} is not invertible, then neither is \mathbf{P}^\top .

The transpose of $(\mathbf{P}_1 \mathbf{P}_2)$ is

$$(\mathbf{P}_1 \mathbf{P}_2)^\top = \mathbf{P}_2^\top \mathbf{P}_1^\top.$$

The inverse of $(\mathbf{P}_1 \mathbf{P}_2)$ is

$$(\mathbf{P}_1 \mathbf{P}_2)^{-1} = \mathbf{P}_2^{-1} \mathbf{P}_1^{-1}.$$

Note that in order for the inverse of $\mathbf{P}_1 \mathbf{P}_2$ to exist, \mathbf{P}_1 and \mathbf{P}_2 must both be invertible.

D.2.10 Orthogonal Matrices

A matrix \mathbf{A} is said to be orthogonal if $\mathbf{A}^\top = \mathbf{A}^{-1}$. The columns of an orthogonal matrix \mathbf{A} must all be unit vectors and must be mutually orthogonal.

D.2.11 Associativity and Commutativity

Matrices are associative and commutative under the addition operation:

$$\mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C}$$

$$\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$$

Matrices are associative under the multiplication operation, but they are not commutative in general

$$\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$$

$$\mathbf{AB} \neq \mathbf{BA}$$

Associativity only holds for matrices of finite dimensions; matrices with infinite dimension do not have the associativity property.

D.2.12 Idempotent Matrices

A matrix \mathbf{A} is said to be idempotent if $\mathbf{A}^2 = \mathbf{AA} = \mathbf{A}$. For example, check that \mathbf{P} is idempotent if

$$\mathbf{P} = \frac{1}{14} \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{bmatrix},$$

then \mathbf{P} is an idempotent matrix. That is, $\mathbf{P}^2 = \mathbf{P}$.

D.3 Solving Linear Equations

A very important application of linear algebra is in solving systems of linear equations. For example, we can represent the equations

$$\begin{aligned} 2x_2 + x_3 &= 1 \\ x_1 + x_3 &= 2 \\ -2x_2 + x_3 &= 3 \end{aligned}$$

as

$$\mathbf{Ax} = \mathbf{y}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 2 & 1 \\ 1 & 0 & 1 \\ 0 & -2 & 1 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

and solve them as

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{y} = \begin{bmatrix} -0.5 & 1 & -0.5 \\ 0.25 & 0 & -0.25 \\ 0.5 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.5 \\ 2.0 \end{bmatrix}$$

D.3.1 Gaussian elimination

The way to solve linear equations which is most commonly described in linear algebra courses is Gaussian elimination. Details on this method are described, for example, at

http://en.wikipedia.org/wiki/Gaussian_elimination

D.3.2 Trace

The trace of a matrix \mathbf{P} , denote $\text{Tr}(\mathbf{P})$, is the sum of the diagonal entries of \mathbf{P} . It has a number of important properties, including the linearity properties:

$$\text{Tr}(\mathbf{P}_1 + \mathbf{P}_2) = \text{Tr}(\mathbf{P}_1) + \text{Tr}(\mathbf{P}_2)$$

and

$$\text{Tr}(\alpha\mathbf{P}) = \alpha\text{Tr}(\mathbf{P}).$$

It also satisfies a commutativity property, even for non-commutative matrices:

$$\text{Tr}(\mathbf{P}_1\mathbf{P}_2) = \text{Tr}(\mathbf{P}_2\mathbf{P}_1)$$

To verify the last property, note that

$$\begin{aligned} \text{Tr}(\mathbf{P}_1\mathbf{P}_2) &= \sum_{i=1}^n (\mathbf{P}_1\mathbf{P}_2)_{ii} = \sum_{i=1}^n \sum_{j=1}^n (\mathbf{P}_1)_{ij}(\mathbf{P}_2)_{ji} \\ &= \sum_{j=1}^n \sum_{i=1}^n (\mathbf{P}_2)_{ji}(\mathbf{P}_1)_{ij} = \text{Tr}(\mathbf{P}_2\mathbf{P}_1). \end{aligned}$$

D.3.3 Determinant

The determinant of an $n \times n$ matrix \mathbf{A} is a scalar value that can be defined in a number of equivalent ways. For a 2×2 matrix with entries a_{ij} , for $i = 1, 2; j = 1, 2$, it can be calculated as $a_{11}a_{22} - a_{12}a_{21}$. Determinants for larger matrices can be calculated from smaller matrices recursively; we will not go into detail here. The `det()` function in R will perform such calculations efficiently.

D.3.4 Some Determinant Properties

$$\det(\mathbf{A}) \neq 0 \text{ if and only if } \mathbf{A}^{-1} \text{ exists.}$$

$$\det(\mathbf{P}_1 \mathbf{P}_2) = \det(\mathbf{P}_1) \det(\mathbf{P}_2).$$

$$\det(\alpha \mathbf{P}) = \alpha^n \det(\mathbf{P}).$$

$$\det(\mathbf{A}^T) = \det(\mathbf{A}).$$

$$\det(\mathbf{AB}) = \det(\mathbf{A}) \det(\mathbf{B}).$$

$$\det(\mathbf{A}^{-1}) = \frac{1}{\det(\mathbf{A})}.$$

$$\det(\exp(\mathbf{A})) = \exp \text{Tr}(\mathbf{A}))$$

$$\text{Tr}(\mathbf{A}) = \log(\det(\exp(\mathbf{A}))).$$

D.3.5 Eigenvalues and Eigenvectors

Consider an $n \times n$ matrix \mathbf{A} . If $\mathbf{Ax} = \lambda \mathbf{x}$, for some $\mathbf{x} \neq \mathbf{0}$ and a scalar λ , then \mathbf{x} is said to be an eigenvector of \mathbf{A} , and λ is an eigenvalue. \mathbf{A} has n eigenvalues (some may be repeated). If an eigenvalue λ is repeated k times, we say that it has algebraic multiplicity k .

Eigenvalues can be real or complex-valued.

Properties

If \mathbf{A} is symmetric, then its eigenvalues are all real, and it will have n linearly independent eigenvectors. The sum of the eigenvalues is equal to $\text{Tr}(\mathbf{A})$. The product of the eigenvalues is equal to $\det(\mathbf{A})$. Thus, \mathbf{A}^{-1} exists if and only if all eigenvalues of \mathbf{A} are nonzero.

If \mathbf{B} is an invertible matrix, then the eigenvalues of \mathbf{A} are identical to the eigenvalues of \mathbf{BAB}^{-1} . \mathbf{BAB}^{-1} is called a similarity transformation. If \mathbf{C} is a triangular matrix, the eigenvalues appear along the diagonal of \mathbf{C} .

D.3.6 Quadratic Forms

Suppose \mathbf{A} is a symmetric $n \times n$ matrix and \mathbf{x} is a vector of length n . Then

$$\mathbf{x}^T \mathbf{Ax}$$

is a quadratic form in \mathbf{x} . It can be written as a quadratic polynomial in the elements of \mathbf{x} . For example, if

$$\mathbf{A} = \begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix},$$

then

$$\mathbf{x}^T \mathbf{Ax} = 2x_1^2 + 6x_1x_2 + 4x_2^2.$$

D.3.7 Positive Definite Matrices

A symmetric matrix \mathbf{A} is said to be positive definite if its quadratic forms in \mathbf{x} are positive when $\mathbf{x} \neq \mathbf{0}$. In other words, \mathbf{A} is positive definite if and only if

$$\mathbf{x}^T \mathbf{A} \mathbf{x} > 0$$

whenever $\mathbf{x} \neq \mathbf{0}$. Completing the square for the previous example,

$$\mathbf{x}^T \mathbf{A} \mathbf{x} = 2(x_1 + 1.5x_2)^2 - .5x_2^2.$$

We can make the first term of right hand side identically 0 by taking $x_1 = -1.5x_2$ for any real x_2 , say $x_2 = 1$. That means that at $\mathbf{x} = [-1.5 \ 1]$,

$$\mathbf{x}^T \mathbf{A} \mathbf{x} = -.5(1)^2 = -.5 < 0.$$

Therefore, the matrix \mathbf{A} is not positive definite.

D.3.8 Exercises

1. By completing the square, verify that

$$\mathbf{B} = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$$

is positive definite.

2. Show that if λ is an eigenvalue of a positive definite matrix, then $\lambda > 0$.
3. Find the eigenvalues of the matrix \mathbf{A} used in the example, and verify that not all of them are positive. Verify that the eigenvalues of \mathbf{B} are all positive.
4. Set

$$\mathbf{C} = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{bmatrix}.$$

Is \mathbf{C} positive definite?

D.4 Linear Spaces

Vector spaces are sets of objects called vectors on which the operations of addition and scalar multiplication are defined, and in which there is a $\mathbf{0}$ vector. For each vector \mathbf{x} in a vector space, there is a corresponding vector called $-\mathbf{x}$ which satisfies the property:

$$\mathbf{x} + (-\mathbf{x}) = \mathbf{0}.$$

If a vector \mathbf{x} is multiplied by a scalar a , then $a\mathbf{x}$ is also a vector. If \mathbf{x} and \mathbf{y} are vectors, then $\mathbf{x} + \mathbf{y}$ is a vector. These last two properties are linearity properties: vector spaces are also called linear spaces.

A linear combination of vectors $\mathbf{x}_1, \dots, \mathbf{x}_k$ is any sum of the form

$$a_1 \mathbf{x}_1 + a_2 \mathbf{x}_2 + \dots + a_k \mathbf{x}_k$$

where a_1, \dots, a_k are scalars.

D.4.1 Subspaces

A subspace is a subset of a vector space which satisfies all of the axioms of a vector space. To check that a subset of a vector space is a subspace, it suffices to check that if \mathbf{x} and \mathbf{y} are members of the subset, then so is $a\mathbf{x} + b\mathbf{y}$, for all scalars a and b . For example, consider the set of all vectors of length 2 which are orthogonal to the vector $[1 \ 1]$. This is a subspace, since if \mathbf{x} and \mathbf{y} are both orthogonal to $[1 \ 1]$, then so is $a\mathbf{x} + b\mathbf{y}$, for any scalars a and b .

D.4.2 Span

A set of vectors is said to span a vector space if any vector \mathbf{x} in the vector space can be expressed as a linear combination of the spanning set. For example, consider the subspace of vectors orthogonal to the vector $[1 \ 1 \ 1]$. The vectors $[1 \ 0 \ -1]^T$ and $[2 \ -1 \ -1]^T$ span this subspace. To see this, let \mathbf{x} be any vector in this subspace, and note that $x_1 + x_2 + x_3 = 0$.

$$\mathbf{x} = a[1 \ 0 \ -1]^T + b[2 \ -1 \ -1]^T$$

where $b = -x_2$ and $a = x_2 - x_3 = x_1 + 2x_2$. The latter equality is consistent with $x_1 + x_2 + x_3 = 0$.

D.4.3 Linear Independence and Dependence

A set of k linearly independent vectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ is said to be linearly independent if the only solution to the system of equations

$$a_1\mathbf{x}_1 + a_2\mathbf{x}_2 + \cdots + a_k\mathbf{x}_k = \mathbf{0}$$

is $a_i = 0$, for $i = 1, 2, \dots, k$. For example, $[1 \ 0 \ -1]$ and $[2 \ -1 \ -1]$ are linearly independent. If the set of vectors is not linearly independent, then it is said to be linearly dependent.

D.4.4 Basis

A set of linear independent vectors is a basis for a vector space if it spans the space. For example, $\{[1 \ 0 \ -1], [2 \ -1 \ -1]\}$ is a basis for the subspace of vectors orthogonal to $[1 \ 1 \ 1]$.

D.4.5 Rank of a Matrix

The rank of a matrix \mathbf{A} is equal to the number of linearly independent columns of \mathbf{A} . It can be shown that the rank of an $n \times n$ matrix \mathbf{A} is equal to $n - k$ where k is the algebraic multiplicity of the 0 eigenvalue.

D.4.6 Gramm-Schmidt Orthogonalization

Given a set of k linearly independent vectors x_1, x_2, \dots, x_k , the Gramm-Schmidt orthogonalization procedure can be used to obtain a set of k orthogonal vectors $\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_k$ which spans the same space as that spanned by x_1, \dots, x_k .

The Gramm-Schmidt procedure is as follows:

$$\mathbf{y}_1 = \mathbf{x}_1, \quad \mathbf{z}_1 = \frac{\mathbf{y}_1}{\|\mathbf{y}_1\|} \quad (\text{D.1})$$

$$\mathbf{y}_2 = \mathbf{x}_2 - \text{proj}_{\mathbf{y}_1}(\mathbf{x}_2), \quad \mathbf{z}_2 = \frac{\mathbf{y}_2}{\|\mathbf{y}_2\|} \quad (\text{D.2})$$

$$\mathbf{y}_3 = \mathbf{x}_3 - \text{proj}_{\mathbf{y}_1}(\mathbf{x}_3) - \text{proj}_{\mathbf{y}_2}(\mathbf{x}_3), \quad \mathbf{z}_3 = \frac{\mathbf{y}_3}{\|\mathbf{y}_3\|} \quad (\text{D.3})$$

$$\vdots \quad \vdots \quad (\text{D.4})$$

$$\mathbf{y}_k = \mathbf{x}_k - \sum_{j=1}^{k-1} \text{proj}_{\mathbf{y}_j}(\mathbf{x}_k), \quad \mathbf{z}_k = \frac{\mathbf{y}_k}{\|\mathbf{y}_k\|}. \quad (\text{D.5})$$

D.4.7 Exercise: Analysis of an Idempotent Matrix

Suppose \mathbf{P} is a symmetric idempotent matrix (that is, $\mathbf{P}^T = \mathbf{P}$ and $\mathbf{P}^2 = \mathbf{P}$). The remaining subsections of the Appendix answer these questions:

1. Find the eigenvalues of \mathbf{P} : $\lambda_1, \lambda_2, \dots, \lambda_n$.
2. Show that the eigenvectors of \mathbf{P} span R^n .
3. Show that there is an $n \times n$ matrix \mathbf{x} for which $\mathbf{P} = \mathbf{X}\mathbf{D}\mathbf{X}^{-1}$ for some diagonal matrix \mathbf{D} .
4. Show that $\mathbf{P} = \mathbf{Z}\mathbf{D}\mathbf{Z}^T$ for an orthogonal matrix \mathbf{Z} and for a diagonal matrix \mathbf{D} .
5. Show that $\text{Tr}(\mathbf{P}) = \sum \lambda_i = \text{number of nonzero eigenvalues of } \mathbf{P}$.

Eigenvalues of \mathbf{P}

Find the eigenvalues of \mathbf{P} : $\lambda_1, \lambda_2, \dots, \lambda_n$. Suppose $\mathbf{x} \neq 0$ is an eigenvector for \mathbf{P} . Then

$$\mathbf{P}\mathbf{x} = \lambda\mathbf{x}$$

for some scalar λ . Premultiplying both sides by \mathbf{P} , we have

$$\mathbf{P}^2\mathbf{x} = \lambda\mathbf{P}\mathbf{x} = \lambda^2\mathbf{x}$$

but

$$\mathbf{P}^2\mathbf{x} = \mathbf{P}\mathbf{x} = \lambda\mathbf{x}$$

so $\lambda = \lambda^2$ which means that λ must be 0 or 1. Thus, all n eigenvalues of \mathbf{P} must be 0's and 1's.

The eigenvectors of P span R^n

Suppose $\mathbf{x} \in R^n$. Then $\mathbf{x} = \mathbf{Px} + (\mathbf{I} - \mathbf{P})\mathbf{x}$. Note that $\mathbf{P}(\mathbf{Px}) = \mathbf{Px}$, so \mathbf{Px} must be an eigenvector of \mathbf{P} (corresponding to the eigenvalue 1). Note also that $\mathbf{P}[(\mathbf{I} - \mathbf{P})\mathbf{x}] = \mathbf{0}$, so $(\mathbf{I} - \mathbf{P})\mathbf{x}$ is also an eigenvector of \mathbf{P} (corresponding to the eigenvalue 0). Thus any $\mathbf{x} \in R^n$ can be expressed as a sum (a special case of a linear combination) of eigenvectors of \mathbf{P} . Hence, the eigenvectors of \mathbf{P} must span R^n .

$\mathbf{P} = \mathbf{XDX}^{-1}$ for diagonal \mathbf{D} .

Show that there is an $n \times n$ matrix \mathbf{x} for which $\mathbf{P} = \mathbf{XDX}^{-1}$ for diagonal \mathbf{D} . Let \mathbf{x} be the matrix whose n columns are the n (linearly independent) eigenvectors of \mathbf{P} . Then $\mathbf{PX} = \mathbf{XD}$ for a diagonal matrix \mathbf{D} whose diagonal consists only of 0's and 1's (the eigenvalues of \mathbf{P}). Thus, $\mathbf{P} = \mathbf{XDX}^{-1}$.

$\mathbf{P} = \mathbf{XDX}^T$ for diagonal \mathbf{D} .

Show that there is an $n \times n$ matrix \mathbf{Z} for which $\mathbf{P} = \mathbf{ZDZ}^T$ for diagonal \mathbf{D} .

Suppose there are k linearly independent eigenvectors corresponding to the eigenvalue 1. These vectors span a k dimensional subspace of R^n . It is possible to linearly transform these vectors to form an orthogonal basis for this subspace (one way to do this is via the Gramm-Schmidt orthogonalization procedure). The $n - k$ eigenvectors corresponding to the 0 eigenvalue can be similarly transformed. It is an easy exercise to show that the eigenvectors corresponding to the 0 eigenvalue are orthogonal to the eigenvectors for the 1 eigenvalue. Therefore, we can use the entire set of orthogonalized eigenvectors as columns of a matrix \mathbf{Z} which will thus be orthogonal, i.e. $\mathbf{Z}^{-1} = \mathbf{Z}^T$. To finish off the argument, note that $\mathbf{PZ} = \mathbf{ZD}$, and argue as above.

The Trace of \mathbf{P} Equals the Number of Nonzero Eigenvalues

Show that $\text{Tr}(\mathbf{P}) = \sum \lambda_i =$ number of nonzero eigenvalues of \mathbf{P}

$$\text{Tr}(\mathbf{P}) = \text{Tr}(\mathbf{XDX}^T) = \text{Tr}(\mathbf{X}^T \mathbf{XD}) = \text{Tr}(\mathbf{D})$$

which is the total number of nonzero eigenvalues of \mathbf{P} .

E

The QR Decomposition of a Matrix

The theory and computation of linear models is greatly simplified when the QR decomposition is used. It can be shown that for any $n \times p$ matrix \mathbf{X} whose columns are linearly independent, there exists an $n \times n$ orthogonal matrix \mathbf{Q} and an invertible $n \times p$ upper triangular matrix R , such that

$$\mathbf{X} = \mathbf{QR}.$$

Orthogonality of \mathbf{Q} means that $\mathbf{Q}^\top \mathbf{Q} = \mathbf{I}$. In order for the columns of \mathbf{X} to be linearly independent, it is necessary that $p \leq n$. For our ultimate purpose, we will need to assume that $p < n$, in fact.

E.1 Householder transformations

Householder transformations are designed to transform any vector to another vector of the same length but where only the first entry is nonzero, and where the matrix of the transformation is orthogonal. In other words, if x is a vector of length n , the Householder transformation is given by a matrix \mathbf{H} such that

$$\mathbf{H}x = \alpha e_1 \quad (\text{E.1})$$

where $\mathbf{H}^\top \mathbf{H} = \mathbf{I}$, e_1 is the unit vector of length n having a ‘1’ as its first entry, and α is an appropriate scalar constant.

It is easy to see that $\alpha = \sqrt{x^\top x}$, by taking the norms of both sides of (E.1). That is,

$$x^\top \mathbf{H}^\top \mathbf{H} x = \alpha^2 e_1^\top e_1 = \alpha^2.$$

Orthogonality of \mathbf{H} tells us that $x^\top x = \alpha^2$, giving us the result.

To complete the specification of \mathbf{H} , we can proceed algebraically or geometrically.

E.2 Geometric derivation

Without loss of generality, consider x in two-dimensional space, as plotted in Figure E.1.

We want to find an orthogonal transformation \mathbf{H} such that $x = \alpha \mathbf{H} e_1$. That is

$$\mathbf{H}^\top x = \alpha e_1.$$

Since e_1 is the unit vector in the horizontal direction, and we already saw that $\alpha = \|x\|$, the vector αe_1 is as pictured in Figure E.2. By definition, it is exactly the same length as x .

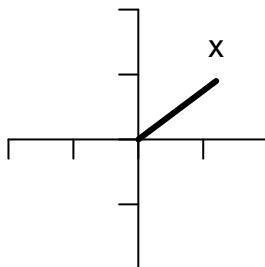


Figure E.1: An arbitrary two-dimensional vector x .

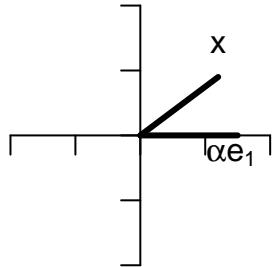


Figure E.2: An arbitrary two-dimensional vector x , and the result after transformation by the unknown orthogonal \mathbf{H} .

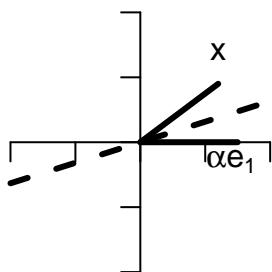


Figure E.3: An arbitrary two-dimensional vector x , and the result after transformation by the unknown orthogonal \mathbf{H} . The transformation is the result of reflection through the dashed line.

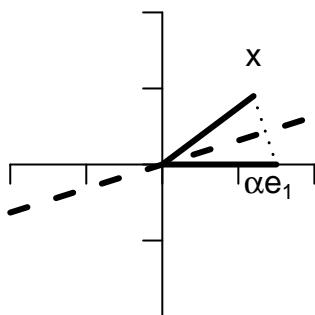


Figure E.4: An arbitrary two-dimensional vector x , and the result after transformation by the unknown orthogonal \mathbf{H} . The transformation is the result of reflection through the dashed line. The dotted line shows the vector $x - \alpha e_1$.

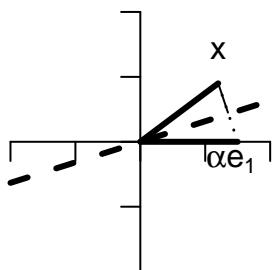


Figure E.5: An arbitrary two-dimensional vector x , and the result after transformation by the unknown orthogonal \mathbf{H} . The transformation is the result of reflection through the dashed line. The dotted line shows the vector $u = x - \alpha e_1$. The orthogonal projection of x onto u is indicated by the solid line segment joining x to the dashed line.

The Householder transformation is set up so that the vector αe_1 is the result of a *reflection* of x through a given line (or hyperplane, in higher dimensional space). In this case, the line is the one which bisects the angle between the vector x and the vector αe_1 as shown in Figure E.3.

Figure E.4 shows the vector $x - \alpha e_1$ as a dotted line segment. Note that it is orthogonal to the bisector of the angle between x and αe_1 .

Letting $u = x - \alpha e_1$, and $v = u/\|u\|$ (the unit vector in the direction of v), we can see the orthogonal projection of x onto v in Figure E.5. Now, recall from basic linear algebra that the orthogonal projection of a vector x onto a unit vector v is given by

$$(x^T v)v.$$

Therefore, $x - (x^T v)v$ must give us the vector parallel to the dashed line in the figure. Subtracting twice as much from x gives us the reflection through the dashed line. That is,

$$x - 2(x^T v)v = \alpha e_1.$$

Rearranging this equation, we have

$$x - 2v(v^T x) = x - 2(vv^T)x = \alpha e_1.$$

This is allowable, because $x^T v$ is a scalar. We conclude from this that

$$\mathbf{H} = \mathbf{I} - 2vv^T.$$

Evidently, \mathbf{H} is symmetric, and it is easy to verify that $\mathbf{H}^T \mathbf{H} = \mathbf{I}$, using the fact that $v^T v = 1$.

Summary of the Householder reflection algorithm

To find the orthogonal transformation \mathbf{H} needed to transform an arbitrary n -dimensional vector x to a multiple of the unit vector in the first coordinate direction, e_1 , follow these steps:

1. Set $\alpha = \sqrt{x^T x}$.
2. Set $u = x - \alpha e_1$, and $v = u/\sqrt{u^T u}$.
3. Set $\mathbf{H} = \mathbf{I} - 2vv^T$.

The resulting \mathbf{H} can be shown to satisfy the relation

$$\mathbf{H}x = \alpha e_1.$$

E.3 Algebraic derivation of Householder reflection of an arbitrary vector x

We seek a symmetric orthogonal transformation of the form

$$\mathbf{H} = \mathbf{I} - \beta ww^T \quad (\text{E.2})$$

where w is a unit vector, and such that

$$x = \mathbf{H}\alpha e_1. \quad (\text{E.3})$$

The orthogonality condition immediately implies that $\beta = 2$, since $w^T w = 1$. We can restate the requirement at (E.3) as

$$\mathbf{H}^T x = \alpha e_1.$$

Symmetry of \mathbf{H} gives

$$\mathbf{H}x = \alpha e_1.$$

Applying (E.2), we have

$$x - 2ww^T x = \alpha e_1.$$

Let $c = w^T x$, a scalar quantity which we will determine later. This allows us to write

$$x - 2cw = \alpha e_1 \quad (\text{E.4})$$

or

$$w = \frac{x - \alpha e_1}{2c}. \quad (\text{E.5})$$

To see what c is, pre-multiply equation (E.4) through by x^T :

$$x^T x - 2cx^T w = \alpha x^T e_1$$

which is equivalent to

$$\alpha^2 - 2c^2 = \alpha x_1$$

giving

$$c = \sqrt{\frac{\alpha^2 - \alpha x_1}{2}}.$$

To obtain the form we obtained from the geometric approach, note that the norm of $x - \alpha e_1$ is $\sqrt{2\alpha^2 - 2\alpha x_1}$ which is coincidentally $2c$. Thus, according to equation (E.5), w is a unit vector which has the same direction as the vector v obtained earlier; thus w and v coincide, and we have

$$\mathbf{H} = \mathbf{I} - 2vv^\top$$

where $v = u/\|u\|$, and $u = x - \alpha e_1$.

E.4 The QR decomposition of an arbitrary matrix

Given a matrix \mathbf{X} , having n rows and p linearly independent columns, we can immediately find an $n \times n$ symmetric orthogonal matrix \mathbf{H}_1 which has the property that

$$\mathbf{H}_1 x_1 = \alpha e_1$$

where x_1 denotes the first column of \mathbf{X} , and α denotes its norm. Thus,

$$\mathbf{H}_1 \mathbf{X}$$

is a matrix whose first column is nonzero only in its first entry. We do not care about the remaining columns at this point, other than to note that they remain linearly independent, after multiplication by an orthogonal matrix.

The next step of the QR decomposition procedure is to construct a second orthogonal matrix of the form

$$\mathbf{Q}_2 = \begin{bmatrix} 1 & 0^\top \\ 0 & \mathbf{H}_2 \end{bmatrix}$$

This time, \mathbf{H}_2 is an $(n-1) \times (n-1)$ Householder reflector which transforms the lower $n-1$ elements of the second column of $\mathbf{Q}_1 \mathbf{X}$. That is, if x_2 denotes the lower $n-1$ elements of the second column of $\mathbf{Q}_1 \mathbf{X}$,

$$\mathbf{H}_2 x_2 = \alpha_2 e_1$$

where α_2 is the norm of x_2 and e_1 now denotes the unit vector in the first coordinate direction, but now in $(n-1)$ -dimensional space. The effect of \mathbf{H}_2 on the lower $n-1$ elements of the first column of $\mathbf{Q}_1 \mathbf{X}$ is to leave them at 0. Thus, the first two columns of $\mathbf{Q}_2 \mathbf{Q}_1 \mathbf{X}$ will have the property we seek: only the top element of the first column, and the top two elements of the second column will be nonzero.

The remainder of the QR procedure is similar. Orthogonal matrices $\mathbf{Q}_3, \mathbf{Q}_4, \dots, \mathbf{Q}_p$ are constructed using progressively smaller Householder reflectors \mathbf{H}_k . At the final step, we would have

$$\mathbf{Q}_p \mathbf{Q}_{p-1} \cdots \mathbf{Q}_2 \mathbf{Q}_1 \mathbf{X} = \mathbf{R}$$

where \mathbf{R} is an $n \times p$ upper triangular matrix. We can also write this as

$$\mathbf{X} = \mathbf{Q}_1 \mathbf{Q}_2 \cdots \mathbf{Q}_p \mathbf{R}$$

so that

$$\mathbf{Q} = \mathbf{Q}_1 \mathbf{Q}_2 \cdots \mathbf{Q}_p$$

is the orthogonal transformation that we seek. Notice that this matrix is not necessarily symmetric, even though the individual Householder reflectors are.

E.5 Computational verification of the QR decomposition

Here, we outline the steps to verify that a given design matrix \mathbf{X} has a QR decomposition. For example, suppose

$$\mathbf{X} = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 1 & 5 \\ 1 & 0 & 3 \\ 1 & 1 & 1 \\ 1 & 2 & 0 \end{bmatrix}.$$

The matrices \mathbf{Q} and \mathbf{R} can be obtained as follows:

```

p <- 3
X <- matrix(c(rep(1,5),
               2,1,0,1,2,
               3,5,3,1,0),
               ncol=p)
X.QR <- qr(X)
Q <- qr.Q(X.QR, complete=TRUE)
Q

## [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] -0.4472136  0.4780914  0.3546497 -0.1821106 -0.6422522
## [2,] -0.4472136 -0.1195229  0.6698938  0.2617034  0.5181389
## [3,] -0.4472136 -0.7171372 -0.1182166 -0.5004816 -0.1457988
## [4,] -0.4472136 -0.1195229 -0.4334607  0.7392599 -0.2265413
## [5,] -0.4472136  0.4780914 -0.4728662 -0.3183710  0.4964534

R <- qr.R(X.QR, complete=TRUE)
R

## [,1]      [,2]      [,3]
## [1,] -2.236068 -2.683282 -5.366563
## [2,]  0.000000  1.673320 -1.434274
## [3,]  0.000000  0.000000  3.625308
## [4,]  0.000000  0.000000  0.000000
## [5,]  0.000000  0.000000  0.000000

```

We verify that $\mathbf{X} = \mathbf{QR}$ by computing the difference and rounding to 14 decimal places.

```

round(X - Q%*%R, digits = 14)

## [,1] [,2] [,3]
## [1,] 0 0 0
## [2,] 0 0 0
## [3,] 0 0 0
## [4,] 0 0 0
## [5,] 0 0 0

```

The orthogonality of \mathbf{Q} can be verified by comparing $\mathbf{Q}^T\mathbf{Q}$ with the identity matrix:

```

I <- diag(rep(1,5))
round(I - t(Q)%*%Q, 14)

## [,1] [,2] [,3] [,4] [,5]
## [1,] 0 0 0 0 0
## [2,] 0 0 0 0 0
## [3,] 0 0 0 0 0
## [4,] 0 0 0 0 0
## [5,] 0 0 0 0 0

```

The next step is to partition $\mathbf{Q} = [\mathbf{Q}_1 \ \mathbf{Q}_2]$, where \mathbf{Q}_1 consists of the first p columns of \mathbf{Q} . It should be easy to verify that

$$\mathbf{Q}_1^T \mathbf{Q}_1 = \mathbf{I}$$

$$\mathbf{Q}_1^T \mathbf{Q}_2 = 0$$

$$\mathbf{Q}_2^T \mathbf{Q}_1 = 0$$

and

$$\mathbf{Q}_2^T \mathbf{Q}_2 = \mathbf{I}.$$

Also,

$$\mathbf{X} = \mathbf{QR} = \mathbf{Q}_1 \mathbf{U}.$$

These properties can be verified for the numerical examples as follows:

```

Q1 <- Q[,1:p]
Q2 <- Q[,-(1:p)]
round(t(Q1) %*% Q1, 14)

##      [,1] [,2] [,3]
## [1,]     1     0     0
## [2,]     0     1     0
## [3,]     0     0     1

round(t(Q1) %*% Q2, 14)

##      [,1] [,2]
## [1,]     0     0
## [2,]     0     0
## [3,]     0     0

round(t(Q2) %*% Q1, 14)

##      [,1] [,2] [,3]
## [1,]     0     0     0
## [2,]     0     0     0

round(t(Q2) %*% Q2, 14)

##      [,1] [,2]
## [1,]     1     0
## [2,]     0     1

U <- R[1:p,]
round(Q1 %*% U, 14)

##      [,1] [,2] [,3]
## [1,]     1     2     3
## [2,]     1     1     5
## [3,]     1     0     3
## [4,]     1     1     1
## [5,]     1     2     0

```

Observe that the diagonal elements of \mathbf{U} are always nonzero when the columns of \mathbf{X} are linearly independent.

E.6 The distribution of the sample variance via the QR decomposition

In this section, we show that the sample variance is related to a χ^2 random variable on $n - 1$ degrees of freedom, when the underlying sample consists of n uncorrelated normal random variables with mean μ and variance σ^2 .

The model for the data that we will use is

$$Z = \mathbf{X}\mu + \varepsilon$$

where \mathbf{X} is an $n \times 1$ matrix consisting only of 1's. Z and ε are n -vectors, and μ is a scalar. When the elements of Z are uncorrelated normal random variables with common mean μ and common variance σ^2 , it follows that ε must have mean vector 0 and variance-covariance matrix $\sigma^2 I$.

Consider the least-squares problem of minimizing

$$(Z - \mathbf{X}\mu)^T(Z - \mathbf{X}\mu) = \sum_{j=1}^n (Z_j - \mu)^2$$

with respect to μ .

In the notation of the preceding section, we have $p = 1$, and we are assuming $n > 1$, so we can construct the QR decomposition of the one-column matrix \mathbf{X} .

Partition the matrix \mathbf{Q} as follows.

$$\mathbf{Q} = [\mathbf{Q}_1 \ \mathbf{Q}_2]$$

where \mathbf{Q}_1 is the first column and \mathbf{Q}_2 comprises the remaining $n - 1$ columns. Also, partition \mathbf{R} in a similar way:

$$\mathbf{R}^\top = [\mathbf{U} \ 0 \ 0 \ \cdots \ 0]$$

where \mathbf{U} is a 1×1 matrix (i.e. a single element), which must be nonzero. It then follows that

$$\begin{aligned} (Z - \mathbf{X}\mu)^\top (Z - \mathbf{X}\mu) &= (Z - \mathbf{Q}\mathbf{R}\mu)^\top (Z - \mathbf{Q}\mathbf{R}\mu) \\ &= (\mathbf{Q}^\top Z - \mathbf{R}\mu)^\top \mathbf{Q}^\top \mathbf{Q} (\mathbf{Q}^\top Z - \mathbf{R}\mu) = (\mathbf{Q}_1^\top Z - \mathbf{U}\mu)^\top (\mathbf{Q}_1^\top Z - \mathbf{U}\mu) + (\mathbf{Q}_2^\top Z)^\top (\mathbf{Q}_2^\top Z). \end{aligned}$$

The final part of this expression does not depend on μ so the minimizer of the entire expression is

$$\hat{\mu} = \mathbf{U}^{-1} \mathbf{Q}_1^\top Z.$$

On the other hand, applying calculus to the expression

$$\sum_{j=1}^n (Z_j - \mu)^2$$

we can see that

$$\hat{\mu} = \sum_{j=1}^n Z_j / n$$

must be the minimizer. There is only one minimizer so

$$\hat{\mu} = \mathbf{U}^{-1} \mathbf{Q}_1^\top Z = \sum_{j=1}^n Z_j / n = \bar{Z}.$$

From above, we have that

$$(Z - \mathbf{X}\hat{\mu})^\top (Z - \mathbf{X}\hat{\mu}) = (\mathbf{Q}_2^\top Z)^\top (\mathbf{Q}_2^\top Z) = Z^\top \mathbf{Q}_2 \mathbf{Q}_2^\top Z.$$

In other words,

$$(n - 1) S_Z^2 = \sum_{j=1}^n (Z_j - \bar{Z})^2 = Z^\top \mathbf{Q}_2 \mathbf{Q}_2^\top Z.$$

However, $\mathbf{Q}_2^\top Z = 0\mu + \mathbf{Q}_2^\top \varepsilon$ so $\mathbf{Q}_2^\top Z$ must be a vector of mean 0 normal random variables with variance-covariance matrix given by

$$\mathbf{Q}_2^\top \mathbf{Q}_2$$

but this is an $(n - 1) \times (n - 1)$ identity matrix by the definition of \mathbf{Q}_2 and the orthogonality of \mathbf{Q} . Therefore, $\mathbf{Q}_2^\top Z$ is a vector of uncorrelated normal random variables with mean 0 and variance σ^2 . Thus,

$$Z^\top \mathbf{Q}_2 \mathbf{Q}_2^\top Z / \sigma^2 = (n - 1) S_Z^2 / \sigma^2$$

must be a χ^2 random variable on $n - 1$ degrees of freedom. Furthermore, the elements of $\mathbf{Q}^\top Z$ are also uncorrelated normals, so $\mathbf{Q}_1^\top Z$ must be independent of $\mathbf{Q}_2^\top Z$, but this means that $\hat{\mu}$ must be independent of $\mathbf{Q}_2^\top Z$ as well, and hence it is independent of $Z^\top \mathbf{Q}_2 \mathbf{Q}_2^\top Z = \sum_{j=1}^n (Z_j - \bar{Z})^2 = (n - 1) S_Z^2$. In other words, \bar{Z} is independent of S_Z^2 .

Exercises

1. Find a Householder reflection matrix which maps the vector $x = [3 \ 4]^\top$ to a vector of the form $[a \ 0]^\top$.
2. Obtain the QR factorization of the matrix

$$\mathbf{X} = \begin{bmatrix} 3 & 10 \\ 4 & 0 \end{bmatrix}.$$

3. Find a Householder reflection matrix which maps the vector $x = [2 \ 3 \ 6]^\top$ to a vector of the form $[b \ 0 \ 0]^\top$.

4. Obtain the QR factorization of the matrix

$$\mathbf{X} = \begin{bmatrix} 2 & 10 \\ 3 & 0 \\ 6 & 0 \end{bmatrix}.$$

5. Repeat the previous four exercises using R, but don't use functions like `qr()`.
6. Suppose z is a vector of 5 independent standard normal random variables, and suppose \mathbf{Q} is an orthogonal 5×5 matrix. Let $y = \mathbf{Q}^T z$.
- (a) Identify the distribution of y .
 - (b) Are the elements of y independent?
 - (c) Identify the distribution of $y^T y$.
 - (d) Suppose \mathbf{Q}_1 is the matrix consisting of the first two columns of \mathbf{Q} , and \mathbf{Q}_2 is the matrix consisting of the remaining columns. Let $w = \mathbf{Q}_1^T z$ and let $v = \mathbf{Q}_2^T z$.
 - i. What is the length of the w vector? ... the v vector?
 - ii. The elements of w correspond to some of the elements of y . Which ones?
 - iii. Which elements of y correspond to the elements of v ?
 - iv. Are w and v independent? Why?
 - v. Identify the distributions of $w^T w$ and $v^T v$. Are these quantities independent?
 - vi. Identify the distribution of $\frac{3w^T w}{2v^T v}$.
7. Find the Householder reflector \mathbf{Q} which transforms the vector $x^T = [1 \ 1 \ 1 \ 1]$ to αe_1 . Determine \mathbf{Q}_1 and \mathbf{Q}_2 as in Section 2.7, and demonstrate explicitly the independence between \bar{Z} and S_Z^2 .

F

A Note on Nonlinear Least-Squares

This material supplements Chapter 6, providing more detail on the Gauss-Newton method for solving nonlinear least-squares problems. In some ways, this note provides a bit of an elaboration on the excellent treatment of Wood (2017).

The (unweighted) nonlinear least-squares problem is to find parameter estimates in a model of the form

$$y_i = g(x_i; \beta_0, \beta_1) + \varepsilon_i, \quad i = 1, 2, \dots, n$$

where the noise terms are uncorrelated with mean 0 and variance σ^2 . The function $g(x)$ is assumed known, except for the values of β_0 and β_1 that appear in the function. (We are assuming that there are only two unknown parameters here; the idea is easily extended to handle more parameters.) Examples of $g(x)$ could be

$$g(x) = \frac{1}{\beta_0 + \beta_1 x}$$

or

$$g(x) = \beta_0 e^{\beta_1 x^2}$$

and so on.

The nonlinear least-squares solution is often found using the Gauss-Newton iteration method. The essential idea is to replace the nonlinear regression problem with a sequence of linear regression problems, where it is hoped that the solutions become more and more accurate, until eventually, the solution of the linear regression problem cannot be distinguished from the solution to the nonlinear regression problem.

The key to the method is to expand the function $g(x_i; \beta_0, \beta_1)$ via Taylor Series in β_0 and β_1 about some initial guesses $\hat{\beta}_0^*$ and $\hat{\beta}_1^*$:

$$g(x_i; \beta_0, \beta_1) \doteq g(x_i; \hat{\beta}_0^*, \hat{\beta}_1^*) + (\beta_0 - \hat{\beta}_0^*) \frac{\partial g}{\partial \beta_0}(x_i, \hat{\beta}_0^*, \hat{\beta}_1^*) + (\beta_1 - \hat{\beta}_1^*) \frac{\partial g}{\partial \beta_1}(x_i, \hat{\beta}_0^*, \hat{\beta}_1^*).$$

For convenience, we collect all objects that we can compute in their entirety and collapse them into the response variable y , calling the modified response variable y' , and noting that it is not really appropriate data for regression anymore, we refer to it as *pseudodata*:

$$y'_i = y_i - g(x_i; \hat{\beta}_0^*, \hat{\beta}_1^*) + \hat{\beta}_0^* \frac{\partial g}{\partial \beta_0}(x_i, \hat{\beta}_0^*, \hat{\beta}_1^*) + \hat{\beta}_1^* \frac{\partial g}{\partial \beta_1}(x_i, \hat{\beta}_0^*, \hat{\beta}_1^*).$$

With the pseudodata defined this way, we have a modified regression model

$$y'_i = \beta_0 \frac{\partial g}{\partial \beta_0}(x_i, \hat{\beta}_0^*, \hat{\beta}_1^*) + \beta_1 \frac{\partial g}{\partial \beta_1}(x_i, \hat{\beta}_0^*, \hat{\beta}_1^*) + \varepsilon_i \tag{F.1}$$

and this model is now a *linear* regression model in β_0 and β_1 . If you look carefully, you will see that, apart from the noise, the only unknowns are β_0 and β_1 . Everything else can be computed.

The iteration proceeds as follows:

1. Choose starting guesses for $\hat{\beta}_0^*$ and $\hat{\beta}_1^*$.
2. Calculate the pseudodata responses and solve the linear regression problem at (F.1) for $\hat{\beta}_0$ and $\hat{\beta}_1$.
3. Reset the values of $\hat{\beta}_0^* = \hat{\beta}_0$ and $\hat{\beta}_1^* = \hat{\beta}_1$, and return to step 2, or STOP, if the estimates are close enough to the starting values.

Example - Poisson Regression

In the Poisson regression example involving the cigarette butt counts in the `cigbutts` data frame in the *MPV* package, we want to fit a model of the form

$$y_i = g(x; \beta_0, \beta_1) + \varepsilon_i = e^{\beta_0 + \beta_1 x_i} + \varepsilon_i.$$

The partial derivatives of g with respect to β_0 and β_1 are

$$e^{\beta_0 + \beta_1 x_i}$$

and

$$x_i e^{\beta_0 + \beta_1 x_i}$$

If we set $\hat{\beta}_0^*$ and $\hat{\beta}_1^*$ as the starting values for the parameters, then the pseudodata become

$$y'_i = y_i - (1 - \hat{\beta}_0^* - \hat{\beta}_1^* x_i) e^{\hat{\beta}_0^* + \hat{\beta}_1^* x_i}$$

and the current stage of the Gauss-Newton procedure solves the ordinary least-squares problem

$$y'_i = \beta_0 e^{\hat{\beta}_0^* + \hat{\beta}_1^* x_i} + \beta_1 x_i e^{\hat{\beta}_0^* + \hat{\beta}_1^* x_i} + \varepsilon_i.$$

To see that this is really ordinary linear regression note that you can make the transformations

$$v_i = e^{\hat{\beta}_0^* + \hat{\beta}_1^* x_i}$$

and

$$w_i = x_i e^{\hat{\beta}_0^* + \hat{\beta}_1^* x_i}$$

yielding the model

$$y'_i = \beta_0 v_i + \beta_1 w_i + \varepsilon_i.$$

Since the x_i 's are known and we have guesses for $\hat{\beta}_0^*$ and $\hat{\beta}_1^*$, the v_i 's and w_i 's are computable. Of course, after solving this regression problem, you simply re-solve it with the updated guesses: $\hat{\beta}_0^* = \hat{\beta}_0$ and $\hat{\beta}_1^* = \hat{\beta}_1$. Repeat this procedure until the estimates stop changing (i.e. convergence).

References

1. Braun, W.J. (2014). *MPV: Data Sets from Montgomery, Peck and Vining's Book*. R package version 1.35.
2. Braun, W.J., Loeppky, J., and Han, L. (2017) Naive variable selection. (preprint)
3. de Boor (1978) *A Practical Guide to Splines*. Springer, New York.
4. Brillinger D.R, Stewart B, Littnan C. (2006). Three months journeying of a Hawaiian monk seal. *Lecture Notes in Statistics*.
5. Canty, A. and Ripley, B. (2020). *boot: Bootstrap R (S-Plus) Functions*. R package version 1.3-25.
6. Davison, A. C. and Hinkley, D. V. (1997) *Bootstrap Methods and Their Applications*. Cambridge University Press, Cambridge. ISBN 0-521-57391-2
7. Eilers, P.H.C. and Marx, B.D. (1996) Flexible smoothing with B-spline and penalties (with discussion). *Statistical Science* **11** 89–121.
8. Green, P. and Silverman, B. (1994) *Nonparametric Regression and Generalized Linear Models*. Chapman and Hall, London.
9. Hastie, T. and Efron, B. (2013). lars: Least Angle Regression, Lasso and Forward Stagewise. R package version 1.2.
10. Hastie, T. and Tibshirani, R. (1990) *Generalized Additive Models*. Chapman and Hall, London.
11. Koenker, R. (2020). *quantreg: Quantile Regression*. R package version 5.75. <http://CRAN.R-project.org/package=quantreg>
12. McCullagh, P. and Nelder, J.A. (1989). *Generalized Linear Models* 2nd edition. Chapman & Hall/CRC, London.
13. Meloche, J. (1991) Estimation of a symmetric density. *The Canadian Journal of Statistics*. **19** 151–164.
14. Montgomery, D.C., Peck, E.A. and Vining, G.G. (2001) *Introduction to Linear Regression Analysis* 3rd edition, Wiley, New York.
15. Pinheiro J, Bates D, DebRoy S, Sarkar D, R Core Team (2020). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-149, <https://CRAN.R-project.org/package=nlme>.
16. R Development Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org/>.
17. Ruppert, D., Wand, M.P. and Carroll, R.J. (2003) *Semiparametric Regression*, Cambridge University Press.
18. Sarkar, D. (2008) *Lattice: Multivariate Data Visualization with R*. Springer, New York. ISBN 978-0-387-75968-5
19. Simonoff, J. (1996) *Smoothing Methods in Statistics*. Springer, New York.
20. Tibshirani, R. 1996. Regression Shrinkage and Selection via the lasso. *Journal of the Royal Statistical Society. Series B (methodological)* **58** 267–88.
21. Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S*. Fourth Edition. Springer, New York. ISBN 0-387-95457-0
22. Wood, S. (2017) *Generalized Additive Models: An Introduction with R* 2nd edition. Chapman & Hall/CRC, London.

Index

- F random variable, 44
 χ^2 , 43
root-MSE, 15
 p -value, 57
 t random variable, 45
MSE, 52
SSE, 12
- ACF, 86
additive error, 7
additive inverse, 284
additive models, 190
AIC, 61, 205
Akaike Information Criterion, 61
algebraic multiplicity, 287
allometric growth, 235
analysis of covariance, 112
ANOVA table, 22
`anova()`, 22
`aov()`, 229
asymptotic distribution, 269
asymptotic normality, 276
autocorrelation, 79, 84, 86
autocorrelation function, 86
average leverage, 97
- B-spline basis, 198
B-spline functions, 201
bar chart, 4
base packages, 1
basis functions, 190
Bayesian credible intervals, 216
Bernoulli, 131
binary data, 130
binary random variable, 2
Block designs, 8
blocking, 8
bootstrap, 171
Boundary knots, 199
box plot, 3
Box-Cox method, 90
Box-Ljung test, 87
`Box.test()`, 87
- canonical link function, 151
canonical parameter, 150, 157
case-by-variable format, 229
- case-resampling, 129
cases, 129
categorical variable, 2
Categorical variables, 106
central limit theorem, 269, 277
character vector, 112
characteristic function, 35
Chebyshev's inequality, 272
codings, 106
coefficient of determination, 23, 88
coefficient vector, 52
column, 280
column vectors, 280
concave function, 274
condition number, 197
conditional density function, 250
conditioning plot, 212
confidence bands, 203
`confint()`, 18
conform, 284
consistency, 277
continuous data, 2
`contour()`, 264
control, 123
`control()`, 130
convergence in probability, 277
Cook's D, 99
Cook's distance, 99
`cor()`, 24
correlation, 247
correlation coefficient, 23
count, 2
covariance, 247
covariates, 2, 48, 106
CRAN, 1
cross-validation, 61, 207
cubic B-spline, 201
cumulative distribution function, 241
`curve()`, 279
`cut()`, 38
CV, 79
- degree of freedom, 43
degrees of freedom, 12, 55
density curve, 43–45
dependent, 85
design matrix, 49, 80, 106, 107

design points, 11
designed experiment, 123
designed experiments, 3
determinant, 287
deterministic model, 4
deviance, 167
DFBETAS, 100
DFFITS, 102
diagnostic plots, 194, 216
diagnostics, 79
diagonal matrix, 283, 285
difference in β s, 100
difference in fitted values, 102
digamma, 270
dimensionless quantities, 118
dimensions, 283
dispersion parameter, 150
dummy variable, 116
Durbin-Watson test, 87

efficiency, 157
eigenvalue, 287
eigenvector, 287
element-wise addition, 284
elements, 280
empirical model, 5
error, 6, 10
error variance, 14, 15, 20
`Error()`, 230
estimating equation, 119, 270, 279
expectation, 245
explanatory variables, 2, 48
exponential distribution, 242
exponential family, 150, 151

factor, 107, 113
`factor()`, 110
factors, 8, 106
families, 215
Fisher Information, 144
Fisher information, 276, 277
fitted values, 11, 51
fixed design, 11, 50
fixed effects, 229
full model, 170

gamma function, 269
Gauss-Markov theorem, 158
GCV, 215
generalized additive models, 215
generalized cross-validation, 207
`ggplot2`, 1
gradient, 106
`graphics`, 1

hat diagonal, 80, 97
hat matrix, 51, 97, 193
`hat()`, 80

high leverage, 97
Householder transformations, 291

i.i.d., 269
idempotent, 286
identically, 249
identifiability constraint, 215
identity matrix, 285
indicator variable, 2
indicator variables, 111
indicator variables., 106
Influence, 99
influence matrix, 51, 72, 77, 193, 207
influential observation, 99
Information, 144
inner product, 281
`install.packages()`, 1
interaction plot, 234
interaction terms, 48
`intervals()`, 232
invariance property, 143
invertible, 284
IRLS, 160
iteratively reweighted least-squares, 160, 161

Jensen's inequality, 145, 274
joint distribution, 35

knots, 190
kurtosis, 77

lag plot, 85
Lagrange multipliers, 158
`lars`, 62
LASSO, 61
Law of large numbers, 271
least-squares, 7, 11–16, 23, 119, 193
leave-one-out cross-validation, 79
length of a vector, 280
`levels`, 111
leverage, 96, 97
likelihood function, 255
likelihood ratio statistic, 145
likelihood ratio test, 145
likely, 255
linear combination, 288
linear equations, 286
linear spaces, 288
linear spline function, 193
linear unbiased estimator, 158
linearity, 286
link, 261
`lm()`, 59
`lme4`, 229
location, 245
log link, 261
log odds, 135
log-linear model, 125

logistic, 135
 logit, 135
 machine learning, 79
 marginal, 249
 marginal probability distribution, 30
 MASS, 261
 MASS, 65
 matrix, 283
 matrix multiplication, 284
 maximum likelihood estimation, 254
 maximum likelihood estimator, 276
 mean, 245
 mean-squared-error, 12
 mechanistic model, 4
 method of moments, 270
 mixed-effects, 185
 MLE, 143
 model assessment, 77
 model complexity, 61
 model matrix, 49, 80
 model validation, 77
`model.matrix()`, 80
 most likely, 255
 MPV, 259
 MPV, 1
 MSE, 12
 multicollinearity, 62, 122
 multiple regression model, 48, 50
 multiple testing, 58
 multiplicative error, 7
 negative binomial, 184
 Newton's method, 270
`n1me`, 229
`nls()`, 126
 noise, 6, 10
 noise variance, 13
 nonlinear least-squares, 126
 nonparametric bootstrap, 129
 norm, 282
 normal distribution, 2
 normal equations, 119, 197, 199
 normality, 79
 null deviance, 168
 null hypothesis, 146
 numeric variable, 2
 object-oriented, 80, 99
 observational data, 2
 odds, 135
 orthogonal, 282
 orthogonal matrix, 34, 50, 291
 orthogonal polynomials, 122
 orthogonal transformation, 40
 outer product, 33
 over-fitting, 157
 overdispersion, 178, 183
 p-value, 145
 package, 1
 base, 1
 parametric bootstrap, 128
 parametrization, 111, 143
 Partial residual plots, 79
 penalized splines, 190
 piecewise polynomials, 190
 pixels, 254
 Poisson, 176
 polynomial regression, 48
 portmanteau test, 87
 positive definite, 288
 predicted value, 54
 predicted values, 11
 prediction bands, 203
 prediction error, 54
 prediction interval, 56
 predictor variables, 2, 48, 106
 PRESS residuals, 78
 probability density function, 241, 254
 probit, 131
 projection, 282
 pseudorandom numbers, 243
`pt()`, 18
 QQ-plot, 30
 QR decomposition, 193
`qt()`, 18
 quadratic form, 287
 random, 4
 random coefficient, 185
 random design, 11, 50
 random effects, 229
 random sample, 152
 random variable, 2
 random vector, 32
 Raw residuals, 77
 reduced model, 170
 regression coefficients, 11
 regression diagnostics, 79
 regression function, 11
 regression sum of squares, 20
 regression through the origin, 27
 regularization, 61, 206
 relative frequency histogram, 43–45
 reparametrizing, 143
 repeated measurements, 185
 residual deviance, 168
 residual standard error, 15, 23
 residuals, 12, 79
 response variable, 2, 48
 retrospective data, 2, 3
`r gamma()`, 269
 robust, 81
 row vectors, 280
 RStudio, 1

rstudio.com, 1
sample correlation, 248
sample covariance, 248
sample mean, 246
sample standard deviation, 246
sample variance, 246
sampling distribution, 171
`sapply()`, 38, 120, 148
saturated model, 152
scalar, 281
scale, 245
score, 144
serial correlation, 84
setup, 1
significance of regression, 59
simulation, 243, 244
singular, 284
skewness, 77
smooth terms, 215
smoother matrix, 207
smoothing splines, 190
spline regression, 193, 202
splines, 190
`split()`, 38
`stack`, 229
standard deviation, 246
standard error, 16, 53, 55, 109
standard errors, 16
standardized residuals, 77, 232
`stats`, 1
`stepAIC()`, 61
strong law of large numbers, 275
studentized residuals, 77
subscripts, *see* indices, *see* indices, *see* indices, *see* indices
sum of squares, 52
sum of squares for error, 12
`summary()`, 113
survivor function, 132
symmetric matrix, 283

Taylor expansion, 88
testing data, 79
trace, 97, 286
tractability, 5
training data, 79
transformation, 2
transpose, 280
transpose of a matrix, 283
treatment group, 111
triangular matrix, 287
trigamma, 270
truncated power basis, 192, 199
truncated power functions, 190
two-sample problem, 106

UBRE, 215
uncertainty, 7