

Einführung in die Statistik mit R

von Silke Ahlers
23. Oktober 2008

Institut für Mathematische Statistik
Westfälische Wilhelms-Universität Münster

Inhaltsverzeichnis

1	Begrifflichkeiten	1
1.1	Erhebungsarten	1
1.2	Merkmale und Stichproben	1
2	Der Einstieg	6
2.1	Datenstrukturen	7
2.1.1	Vektoren	7
2.1.2	Matrizen, Arrays und Tabellen	12
2.2	Funktionen	21
2.3	Exkursion zum Programmieren	22
2.4	Grafiken	24
2.5	Verteilungen	28
2.6	Pakete	30
3	Univariate Analyse	31
3.1	Darstellung univariater Datensätze	31
3.1.1	Darstellung qualitativer Merkmale	31
3.1.2	Darstellung quantitativer Merkmale	38
3.2	Beschreibung univariater Datensätze	43
3.2.1	Median und Mittelwert	43
3.2.2	Quantile	45
3.2.3	Maßzahlen für die Variabilität	46
3.2.4	Der Boxplot	48
4	Multivariate Datenanalyse	52
4.1	Zusammenhang zwischen quantitativen und qualitativen Merkmalen	52
4.2	Zusammenhang zwischen nominalskalierten Merkmalen	55
4.3	Zusammenhang zwischen quantitativen Merkmalen	61
5	Testtheorie	67
5.1	Mittelwertvergleiche (parametrische Verfahren)	68
5.2	Das Einstichprobenproblem	70
5.3	Der Zweistichprobenfall	82
5.4	Der k -Stichprobenfall ($k \geq 3$)	92
6	Varianzanalyse	98
6.1	Einfaktorielle Varianzanalyse	99

Inhaltsverzeichnis

6.2	Mehrfaktorielle Varianzanalyse	108
7	Regressionsanalyse	119
8	Schätzung und Simulation	130

1 Begrifflichkeiten

Bevor wir uns mit dem Programm R oder Beschreibung und Auswertung von Datensätzen beschäftigen, wollen wir in diesem Kapitel auf die in Praxis üblichen statistischen Begrifflichkeiten eingehen, insbesondere bezüglich der Art der Datengewinnung und der untersuchten Daten.

1.1 Erhebungsarten

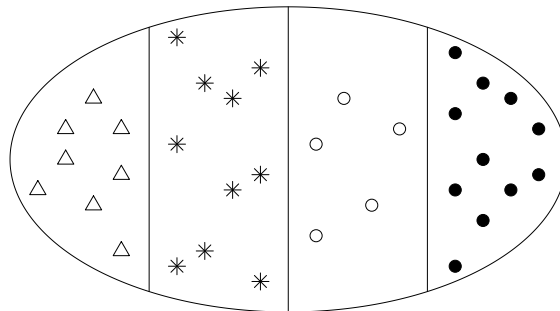
Zur Datengewinnung bieten sich uns mehrere Möglichkeiten. Wir können eine **Befragung**, eine **Beobachtung** oder ein **Experiment** durchführen, um Daten zu erheben. Bei den Befragungen haben wir zudem die Wahl zwischen einer **schriftlichen** oder einer **mündlichen** Variante, wobei bei der mündlichen Befragung zusätzlich unterschieden wird, ob sie direkt oder telefonisch stattfindet. Dabei können zu den gestellten Fragen die Antwortmöglichkeiten vorgegeben sein, in welchem Fall man von **geschlossenen Fragen** spricht, oder nicht, wobei die Frage dann als **offen** bezeichnet wird. Bei einer Befragung sollte die Antwortquote, der Quotient aus der Anzahl der gegebenen Antworten und der Anzahl der Befragten, hoch sein, damit man aus den so gewonnenen Daten Erkenntnisse (durch Rückschlüsse) gewinnen kann. Bei Befragungen sind die Antworten von der Wahrnehmung (und der Ehrlichkeit!) der Befragten abhängig, so dass sie nicht immer einen korrekten Eindruck der Sachlage vermitteln. Wenn man eine Beobachtung durchführt, sind diese individuellen Einflüsse ausgeschaltet. Werden die Beobachtungen zu äquidistanten Zeitpunkten erhoben, spricht man von einer **Zeitreihe**. Ein Nachteil bei einer Beobachtung ist allerdings, dass man keinerlei Einfluss auf die Rand- und Rahmenbedingungen hat. Bei einem Experiment hingegen kontrolliert der Durchführende sämtliche Randbedingungen, wodurch der Vergleich unterschiedlicher Eigenschaften oder Verfahren bzw. einer Eigenschaft unter leicht variierten Bedingungen möglich wird.

1.2 Merkmale und Stichproben

Die bei einer Datenerhebung auf bestimmte **Merkmale** untersuchten Personen bzw. Objekte nennt man **statistische Einheit** oder auch **Merkmalsträger**, während die Menge aller Merkmalsträger als **(Grund-)Gesamtheit** bezeichnet wird. Werden alle interessierenden Einheiten erfasst, so spricht man einer **Vollerhebung**, ansonsten von einer **Teilerhebung**. Oft ist es nicht möglich, eine Vollerhebung durchzuführen. Will man beispielsweise die Lebensdauer von Glühbirnen untersuchen, so würde eine Vollerhebung zur Zerstörung der gesamten Produktion führen. Man spricht in diesem Fall von einer **zerstörenden Prüfung**.

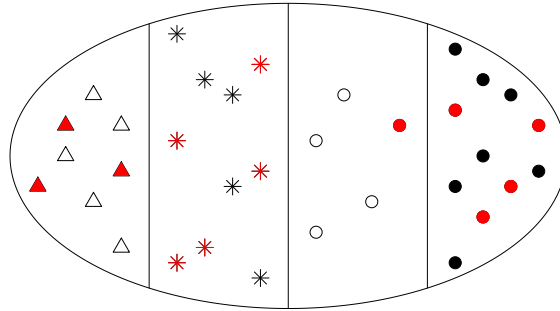
1 Begrifflichkeiten

Bei Teilerhebungen ist man in der Regel an der Verteilung der untersuchten Merkmale in der **Grundgesamtheit** und nicht nur in der untersuchten **Teilgesamtheit** interessiert. In der Statistik ist der Ausgangspunkt immer die vorliegende Stichprobe, die (fast immer) eine Erhebung einer Teilgesamtheit ist. Diese Teilgesamtheit sollte repräsentativ für die Grundgesamtheit sein. Im Falle keinerlei Vorkenntnisse über die Grundgesamtheit zieht man eine Zufallsstichprobe, wodurch gewährleistet sein soll, dass die Stichprobe repräsentativ für die Gesamtheit ist. Werden jedoch Elemente der Grundgesamtheit bei der Ziehung nicht berücksichtigt, spricht man von einer **verzerrten Stichprobe**. Dies kann durch bewussten Ausschluss dieser Elemente von der Ziehung, einen **Selektions-Bias**, geschehen (z.B. bei Befragungen via Internet) oder dadurch, dass (insbesondere unangenehme) Fragen unbeantwortet bleiben, was man **Nonresponse-Bias** nennt. Eine verzerrte Stichprobe erhält man häufig auch bei Befragungen auf freiwilliger Basis, da sich oft nur die beteiligen, die an der Frage besonders interessiert sind (z.B. regelmäßige Mensabesucher bei der Mensaaumfrage, Kranke bei Umfragen zu Arzt- und Krankenkassenleistungen,...), wodurch ein **Selfselection-Bias** entsteht. Man fasst nun die Stichprobe der Länge n als Realisierung von n Zufallsgrößen X_1, \dots, X_n auf und betrachtet dann Stichprobenfunktionen $g(X_1, \dots, X_n)$, häufig $g(X_1, \dots, X_n) = \frac{1}{n} \sum_{i=1}^n X_i =: \bar{X}$. Hat man bereits Informationen über die Grundgesamtheit, so sollte man diese bei der Ziehung der Stichproben berücksichtigen. Eine Möglichkeit ist die Ziehung einer **geschichteten Stichprobe**: Man unterteilt die Grundgesamtheit gemäß bekannter Merkmale (Eigenschaften) in disjunkte Gruppen, sogenannte **Schichten**, und zieht aus jeder Schicht eine (zu ihrer Größe proportionale) Stichprobe. Aus jeder dieser vier Schichten

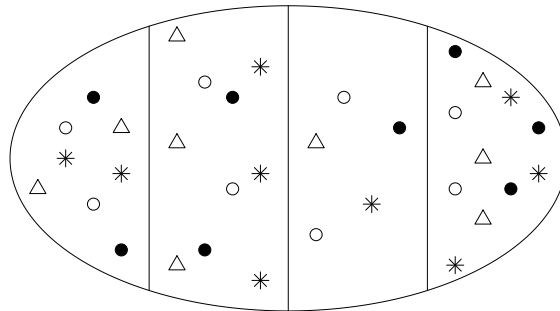


würde man nun eine kleine Stichprobe ziehen, und daraus eine Gesamtstichprobe zusammensetzen. Ein Beispiel für eine solche Auswahl zeigt die folgende Grafik, in der die ausgewählten Punkte, die zu der Stichprobe zusammengesetzt werden, rot eingefärbt sind. Bei einer geschichteten Stichprobe entspricht der Erwartungswert von \bar{X} gerade dem der Grundgesamtheit, die Varianz wird aber kleiner als die von \bar{X} bei einer Zufallsstichprobe (zumindest wenn die Streuung innerhalb der Schichten nicht so groß ist und die verschiedenen Schichten Unterschiede aufweisen). Dieses Phänomen nennt man den **Schichtungseffekt**.

1 Begrifflichkeiten



Bei Erhebungen wird häufig das **Klumpenverfahren** angewendet, das darin besteht, mehrere Personen oder Objekte zu einer Einheit, einem **Klumpen**, zusammenzufassen und bei der Stichprobe ganze Klumpen und nicht einzelne Personen oder Objekte auszuwählen. Die obere der beiden nun folgenden Grafiken zeigt eine solche Einteilung in Klumpen. Ungeachtet der Merkmalsausprägungen, die die Merkmalsträger aufweisen, werden diese in Klumpen eingeteilt, wobei die Klumpen nicht die gleiche Größe haben müssen. Aus diesen Klumpen wird nun zufällig einer (oder auch mehrere) ausgewählt



und als Stichprobe behandelt, in diesem Fall der zweite Klumpen in der Grundgesamtheit. Hier tritt nun der **Klumpeneffekt** auf: Der Erwartungswert von \bar{X} ist wieder derselbe, doch die Varianz von \bar{X} ist (normalerweise) größer als die von \bar{X} bei einer Zufallsstichprobe. Trotz dieser schlechteren Eigenschaft wird das Klumpenverfahren oft aus Kostengründen eingesetzt (mündliche Befragungen haushalts- und siedlungsweise).

Beispiel 1.1. Es soll das durchschnittliche Einkommen aller Erwerbstätigen in Deutschland bestimmt werden. Die Grundgesamtheit umfasst also genau alle Erwerbstätigen in Deutschland und **nicht** die Erwerbslosen, Rentner oder (Schul-, Klein-)Kinder. Bei einer Vollerhebung würde das Einkommen jedes einzelnen festgehalten, bei einer Teilerhebung

1 Begrifflichkeiten

Bei der Unterscheidung zwischen diesen Eigenschaften von Merkmalen spricht man auch vom **Skalenniveau** der Merkmale. Tritt in einer Stichprobe ein Wert mehrmals auf, spricht man von **Bindungen** (engl.: ties). Werte mit gleichem Betrag nennt man **Rangbindungen**.

- Beispiel 1.2.** a) Bei der Sonntagsfrage sollen die Befragten angeben, welche Partei sie wählen, wenn am nächsten Sonntag Bundestagswahl wäre. Das untersuchte Merkmal ist also „bevorzugte Partei“ und somit in Kategorien einzuteilen (für jede Partei eine, plus eine für Enthaltungen). Da zwischen den einzelnen Ausprägungen dieses Merkmals keine Ordnung herzustellen ist, liegt hier ein nominalskaliertes Merkmal vor.
- b) Schulnoten sind offensichtlich in Klassen eingeteilt, die geordnet sind. Es handelt sich in dem Fall also um ein ordinalskaliertes Merkmal. Allerdings ist es nicht intervallskaliert, da der Abstand zwischen den Noten 1 und 2 zwar der gleiche ist wie der zwischen den Noten 4 und 5, aber eine völlig andere Bedeutung besitzt. Auch können die Verhältnisse aus den Noten nicht sinnvoll gedeutet werden; wer eine vier bekommen hat, ist nicht viermal so schlecht wie jemand, der eine eins bekommen hat, da allein der Ausdruck „x-mal so schlecht“ bzw. „x-mal so gut“ keinen Sinn hat.
- c) Bei einem Physikversuch werden Schwingungsdauern eines Pendels gemessen. Hierbei handelt es sich offenbar um ein quantitatives (quasi-)stetiges Merkmal. (Durch die Messmethoden und -ungenauigkeiten der Messgeräte werden die Daten diskretisiert, die Feinheit der Instrumente dürfte aber eine Stetigkeitsannahme rechtfertigen...) Da hier sowohl die Differenzen der Werte („schwingt x sec länger“) als auch die Quotienten („schwingt x-mal so lange“) sinnvoll gedeutet werden können, ist dieses Merkmal metrisch skaliert.

◇

2 Der Einstieg

R bietet eine **interaktive Umgebung**, den **Befehlsmodus**, in dem man die Daten direkt eingeben und analysieren kann. Durch das **Bereitschaftszeichen** `>` wird angezeigt, dass eine Eingabe erwartet wird. Mit den Operatoren `+`, `-`, `*`, `/` und `^` können wir die Grundrechenarten und Potenzierungen durchführen, indem wir hinter `>` den zu berechnenden Ausdruck eingeben und mit der return-Taste den Befehl abschicken. Das Ergebnis gibt R in der nächsten Zeile aus. Bei Dezimalzahlen ist zu beachten, dass sie mit einem **Dezimalpunkt** geschrieben werden.

`>`
`+ - * / ^`

Beispiel 2.1. Addition, Subtraktion, Multiplikation und Division erhält man wie folgt:

```
> 2.1+2
[1] 4.1

> 2.1-2
[1] 0.1

> 2.1*2
[1] 4.2

> 2.1/2
[1] 1.05
```

Zum Potenzieren benutzen wir das Zeichen `^`, also berechnet man die Quadratwurzel von 2 durch

```
> 2^0.5
[1] 1.414214
```

wobei wir zum selben Ergebnis mit der Funktion `sqrt` (Abkürzung für `squareroot`) gelangen:

`sqrt`

```
> sqrt(2)
[1] 1.414214
```

◇

Das Programm R rundet (falls nötig) Zahlen standardmäßig auf sechs Dezimalstellen. Um auf weniger Stellen zu runden, verwendet man die Funktion `round`. Diese besitzt

`round`

ein **obligatorisches Argument** `x`, nämlich die Zahl, die gerundet werden soll, und das **optionale Argument** `digits`, das die Anzahl der Dezimalstellen angibt, auf die gerundet werden soll. Standardmäßig (also ohne Eingabe des optionalen Argumentes) wird in R auf ganze Zahlen gerundet.

Beispiel 2.2. Betrachten wir noch einmal die Quadratwurzel aus 2. Wie oben gesehen erhalten wir als Ergebnis eine Zahl mit sechs Dezimalstellen. Mit der Funktion `round` erhalten wir:

```
> round(sqrt(2))
[1] 1
```

bzw. beim Runden auf zwei Dezimalstellen

```
> round(x=sqrt(2), digits=2)
[1] 1.41
```

◇

Wie die Rundungsfunktion besitzt jede Funktion in R zwei Typen von Argumenten, zum einen die obligatorischen, die angegeben werden müssen, und zum anderen die optionalen, deren Eingabe nicht zwingend ist und die bei Weglassen auf einen Standardwert eingestellt werden. Die einzelnen Argumente werden bei der Eingabe in die Funktion durch ein Komma getrennt. Kennt man die Reihenfolge der Argumente im sogenannten Kopf der Funktion, so kann man einfach ihre Werte eingeben. Ist dies nicht der Fall, gibt man sie mit „Name = ·“ ein. (Man kann die Namen der Argumente durch Weglassen hinterer Buchstaben abkürzen, wenn sie dadurch eindeutig bleiben.) Im Beispiel würde es demnach auch ausreichen, `> round(sqrt(2), 2)` einzugeben.

Da bei Erhebungen häufig n Merkmalsträger, $n \in \mathbb{N}$, in Bezug auf p Merkmale untersucht werden, liegen uns zur Auswertung üblicherweise p Vektoren der Länge n oder eine $n \times p$ -Matrix vor. Daher beschäftigen wir uns im kommenden Abschnitt mit:

2.1 Datenstrukturen

Dabei werden wir neben Vektoren, Matrizen und Arrays auch Listen und Tabellen in R erzeugen bzw. in das Programm importieren. Doch zunächst zum einfachsten Fall:

2.1.1 Vektoren

Einen Vektor erzeugen wir in R mit der Funktion `c`. Wollen wir z.B. einen Vektor aus den Daten x_1, \dots, x_5 bilden, geben wir folgenden Befehl ein:

```
> c(x1, x2, x3, x4, x5)
[1] x1 x2 x3 x4 x5
```

2 Der Einstieg

wobei die untere Zeile die Ausgabe des Programms darstellt, die sofort nach Drücken der return-Taste erfolgt. Um mit einem Datensatz, wie in diesem Fall dem Vektor, weiterhin arbeiten zu können ohne ihn jedesmal neu eingeben zu müssen, weisen wir ihm eine Variable, also einen Namen, zu. Dies geschieht mit dem **Zuweisungsoperator** `<-`, der sich aus den beiden Zeichen `<` und `-` ergibt. Links von diesem Operator steht der Name, dem die Daten zugewiesen werden sollen. Wollen wir den obigen Vektor mit **Versuch** bezeichnen, geben wir also

```
> Versuch<-c(x1,x2,x3,x4,x5)
```

ein, und können den Vektor mit diesem Namen wieder aufrufen:

```
> Versuch
[1] x1 x2 x3 x4 x5
```

Diese Variable bleibt während der gesamten Sitzung im **Workspace** erhalten, es sei denn, wir löschen sie mit dem Befehl `rm` (Abkürzung von remove). Wollen wir die Sitzung beenden, verlassen wir den Workspace mit der Funktion `q()`, die als Argument `yes` zulässt, wenn wir den Workspace und insbesondere die von uns eingegebenen Daten und Variablen sichern wollen, bzw. `no`, wenn wir diesen nicht speichern wollen. Beide Argumente müssen dabei in Anführungszeichen gesetzt werden. Bei den Variablen gilt es zu beachten, dass R Groß- und Kleinschreibung unterscheidet. Wollen wir beispielsweise `versuch` aufrufen, erhalten wir

```
> versuch
Fehler: objekt versuch nicht gefunden
```

Hätten wir andererseits gern eine Liste der bisher definierten Variablen, erhalten wir eine solche mit Hilfe des Befehls `ls()`, bei dem die Klammern leer bleiben.

Wollen wir auf Komponenten eines Vektors zugreifen, **indizieren** wir diesen. Hierzu gibt man den Namen des Vektors gefolgt von eckigen Klammern ein, zwischen denen die Nummer der Komponente oder der Vektor mit den Nummern der Komponenten steht, auf die man zugreifen will. Ist dies die letzte Komponente des Vektors, sollten wir die Länge desselben kennen, die wir durch Aufruf der Funktion `length` erhalten.

Beispiel 2.3. Ein Comicsammler hat im vergangenen Jahr bei einem amerikanischen Händler zehn Comics gekauft und dafür folgende Preise in US-Dollar bezahlt:

22 30 16 25 27 23 18 29 31 26.

Wir geben diese Daten als Vektor unter dem Namen `comics` in R ein:

```
> comics<-c(22,30,16,25,27,23,18,29,31,26)
```

2 Der Einstieg

```
> comics
[1] 22 30 16 25 27 23 18 29 31 26
```

Wollen wir wissen, wie viel die ersten drei Comics gekostet haben, haben wir zwei Möglichkeiten, dies einzugeben:

```
> comics[c(1,2,3)]
[1] 22 30 16
```

```
> comics[1:3]
[1] 22 30 16
```

Mit dem Operator `:`, der in der vorletzten Zeile auftaucht, erzeugt man nämlich einen Vektor mit aufeinander folgenden natürlichen Zahlen, beginnend bei der Zahl, die links vom Operator steht, und auf- bzw. absteigend zu der Zahl rechts davon, je nachdem, ob diese kleiner oder größer als die erste ist. Den Preis der letzten beiden Comics erhalten wir durch

```
> comics[c(length(comics)-1,length(comics))]
[1] 31 26
```

◇

Wir können die Komponenten eines Vektors auch nach Bedingungen auswählen. Dazu kann man in R die Operatoren

```
==  gleich
!=  ungleich
<   kleiner
<=  kleiner oder gleich
>   größer
>=  größer oder gleich
```

benutzen. Beim Vergleich von zwei Zahlen gilt beispielsweise:

```
> 3<4
[1] TRUE
```

```
> 3>4
[1] FALSE
```

Man kann nun auch Vektoren mit Skalaren vergleichen, wobei das Ergebnis ein Vektor ist, dessen Komponenten angeben, ob die entsprechenden Komponenten des Ausgangsvektors die Bedingung erfüllen. Wir nehmen wieder den Vektor `comics` und wollen wissen, welche Komponenten größer oder gleich 25 sind:

2 Der Einstieg

```
> comics >= 25
[1] FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE
```

Man spricht von einem **logischen Vektor**. Wenn wir einen Vektor x mit einem logischen Vektor l indizieren, also $x[l]$, so werden in x alle Komponenten ausgesucht, die in l den Wert `TRUE` annehmen. Durch

```
> comics[comics>=25]
[1] 30 25 27 29 31 26
```

erhalten wir demnach die Preise der Comichefte, die mindestens 25 US-Dollar gekostet haben. Will man nicht den Wert dieser Komponenten wissen, sondern ihre Positionen innerhalb des Vektors x (in diesem Fall also `comics`), so erzeugt man den Vektor $1 : \text{length}(x)$ und indiziert diesen mit dem logischen Vektor l (hier: `comics >= 25`). Schneller zum gleichen Ergebnis gelangt man mit der Funktion `which`:

```
> which(comics>=25)
[1] 2 4 5 8 9 10
```

Mit den Funktionen `any` und `all` kann man überprüfen, ob einer oder alle Komponenten eines Vektors eine bestimmte Bedingung erfüllen.

```
> any(comics>30)
[1] TRUE
```

```
> all(comics>30)
[1] FALSE
```

Auf einen Vektor bzw. seine Komponenten kann man noch weitere einfache nützliche Funktionen anwenden. Mit dem Befehl `sum(x)` bildet man zum Beispiel die Summe aller Komponenten des Vektors x , mit den Funktionen `min(x)` und `max(x)` kann man dessen Minimum und Maximum bestimmen. Wollen wir die Komponenten der Größe nach ordnen, so erreichen wir das durch die Funktion `sort(x)`. Standardmäßig wird der Vektor x aufsteigend geordnet, wenn wir jedoch das optionale Argument `decreasing` auf den Wert `TRUE` setzen, wird der Vektor absteigend sortiert.

Beispiel 2.4. Schauen wir uns noch einmal die Ausgaben des Comicsammlers an. Insgesamt hat er

```
> sum(comics)
[1] 247
```

US-Dollar ausgegeben, wobei der günstigste und der teuerste Comic

2 Der Einstieg

```
> min(comics)
[1] 16
```

```
> max(comics)
[1] 31
```

US-Dollar gekostet haben. Der Größe nach sortiert erhält er diese Preislisten:

```
sort(comics)
[1] 16 18 22 23 25 26 27 29 30 31
```

```
sort(comics,decreasing=TRUE)
[1] 31 30 29 27 26 25 23 22 18 16
```

◇

Bis jetzt haben wir uns nur mit quantitativen Merkmalen beschäftigt. Wie aber gibt man Daten bei einem qualitativen Merkmal ein? In diesem Fall erzeugt man einen Vektor, dessen Komponenten Zeichenketten sind, wobei eine Zeichenkette eine Folge von Zeichen (z.B. Buchstaben) ist, die in Hochkommata stehen. Der Vektor, der dadurch in R erzeugt wird, enthält weiterhin die Hochkommata. Um diese zu beseitigen, wenden wir die Funktion `factor` an, die den aus Zeichenketten bestehenden Vektor zu einem **Faktor**, also einem qualitativen Merkmal macht. Auch solche Vektoren können wie numerische Vektoren indiziert werden.

`factor`

Beispiel 2.5. Bei einem Seminar mit zehn Teilnehmern ergab eine Erhebung des Geschlechts der Teilnehmer

w m w m w m m m w m.

In R geben wir daher folgenden Befehl ein:

```
> Geschlecht<-c("w","m","w","m","w","m","m","m","w","m")
> Geschlecht<-factor(Geschlecht)
> Geschlecht
[1] w m w m w m m m w m
Levels: m w
```

Diesen Vektor können wir wie vorhin eingeführt indizieren:

```
> Geschlecht[5:length(Geschlecht)]
[1] w m m m w m
Levels: m w

> Geschlecht[Geschlecht=="w"]
```

```
[1] w w w w
Levels: m w
```

```
> which(Geschlecht=="w")
[1] 1 3 5 9
```

◇

Wollen wir zwei (oder mehr) Bedingungen an einem Vektor überprüfen, können wir die Operatoren `&` und `|` benutzen. Der erste, `&`, nimmt genau dann den Wert `TRUE` an, wenn beide (alle) Bedingungen erfüllt sind, während der zweite, `|`, genau dann `TRUE` wird, wenn mindestens eine der Bedingungen erfüllt ist.

& |

2.1.2 Matrizen, Arrays und Tabellen

Bisher haben wir nur Datensätze zu einem Merkmal betrachtet, allerdings werden üblicherweise mehrere Merkmale gleichzeitig erhoben, die sich dann nicht mehr sinnvoll in einem Vektor darstellen lassen. In solchen Fällen werden wir eine Matrix erzeugen wollen. Dies geschieht in R mit der Funktion `matrix`, die durch

`matrix`

```
matrix(data,nrow,ncol,byrow(=F))
```

aufgerufen wird. Das erste Argument sind also die Daten, die in der Matrix stehen sollen. Diese geben wir in Form eines Vektors ein, der dann durch das Programm in die entsprechenden Zeilen oder Spalten zerlegt wird. Letzteres hängt vom Argument `byrow` (was ungefähr „nach Zeilen“ heißt) ab. Nimmt dieses den Wert `FALSE` (abgekürzt: `F`) an, wird die Matrix spaltenweise aufgefüllt. In diesem Fall sollte der Datenvektor aus einer Aneinanderreihung der gewünschten Spalten bestehen. Setzen wir das Argument `byrow` auf den Wert `TRUE` (oder abgekürzt `T`), wird die Matrix zeilenweise eingetragen, d.h. unser Datenvektor sollte sich aus den gewünschten Zeilen zusammensetzen. Gibt man das Argument `byrow` nicht ein, wird es durch das Programm standardmäßig auf den Wert `FALSE`, also auf spaltenweise Füllung der Matrix gesetzt. Die beiden verbleibenden Argumente `nrow` und `ncol` sind obligatorisch und geben die Anzahl der Zeilen (number of rows) und die Anzahl der Spalten (number of columns) an. (Das gerade Gesagte stimmt nicht ganz: Man muss mindestens eines der beiden Argumente eingeben. Gibt man nur eine Dimensionszahl ein errechnet R daraus und aus der Länge des Datenvektors die zweite Dimensionszahl, sofern die Länge des Vektors durch die eingegebene Zahl teilbar ist.)

Beispiel 2.6. Zwanzig Studenten und Studentinnen wurden unter anderem nach ihrer Körpergröße, ihrem Gewicht und ihrer Schuhgröße befragt. Dabei gaben sie folgende Werte (in cm bzw. kg) an:

2 Der Einstieg

Körpergröße	Gewicht	Schuhgröße
171	58	40
180	80	44
178	80	42
171	60	41
182	73	44
180	70	41
180	77	43
170	55	42
163	50	37
169	51	38
201	93	48
180	67	42
183	73	42
176	65	42
170	65	41
182	85	40
180	80	41
190	83	44
180	67	39
183	75	45

Wollen wir diese Werte in R als Matrix darstellen, gibt es zwei Möglichkeiten der Eingabe: Einerseits als eine spaltenweise aufgefüllte Matrix

```
> maße<-matrix(c(171,180,178,171,182,180,180,170,163,169,201,
                 180,183,176,170,182,180,190,180,183,58,80,80,
                 60,73,70,77,55,50,51,93,67,73,65,65,85,80,83,
                 67,75,40,44,42,41,44,41,43,42,37,38,48,42,42,
                 42,41,40,41,44,39,45),nrow=20,ncol=3)
```

oder andererseits zeilenweise gefüllt

```
> maße<-matrix(c(171,58,40,18,80,44,178,80,42,171,60,41,182,
                 73,44,180,70,41,180,77,43,170,55,42,163,50,
                 37,169,51,38,201,93,48,180,67,42,183,73,42,
                 176,65,42,170,65,41,182,85,40,180,80,41,190,
                 83,44,180,67,39),nrow=20,ncol=3,byrow=TRUE)
```

was zum selben Ergebnis führt:

```
> maße
```

2 Der Einstieg

	[, 1]	[, 2]	[, 3]
[1,]	171	58	40
[2,]	180	80	44
[3,]	178	80	42
[4,]	171	60	41
[5,]	182	73	44
[6,]	180	70	41
[7,]	180	77	43
[8,]	170	55	42
[9,]	163	50	37
[10,]	169	51	38
[11,]	201	93	48
[12,]	180	67	42
[13,]	183	73	42
[14,]	176	65	42
[15,]	170	65	41
[16,]	182	85	40
[17,]	180	80	41
[18,]	190	83	44
[19,]	180	67	39
[20,]	183	75	45

◇

Mit dem optionalen Argument `dimnames` können wir den Zeilen und Spalten Namen zuordnen. Da es bei einer Matrix zwei Dimensionen gibt, geben wir eine Liste aus zwei Vektoren ein, wobei der erste aus den Zeilennamen und der zweite aus den Spaltennamen besteht:

```
dimnames=list(c(„erste Zeile“,...,„letzte Zeile“),  
             c(„erste Spalte“,...,„letzte Spalte“))
```

Man kann auch nachträglich die Dimensionsnamen einer Matrix `x` ändern, indem man `dimnames` verwendet:

```
dimnames(x)<-list(c(„neue erste Zeile“,...,„neue letzte Zeile“),  
                c(„neue erste Spalte“,...,„neue letzte Spalte“))
```

Wollen wir aus einer Matrix ein bestimmtes Element auswählen, so können wir dies wie bei Vektoren durch Indizierung vornehmen, wobei zu beachten ist, dass eine Matrix ein zweidimensionales Objekt ist. Wir müssen also zwei Koordinaten `[x,y]` angeben, wovon `x` für die Zeilennummer und `y` für die Spaltennummer steht. Will man die ganze Zeile Nr. `x` auswählen, schreibt man `[x,]` und analog `[,y]`, wenn man nur die Spalte `y` auswählen möchte.

Wollen wir eine Matrix `A` transponieren, erreichen wir dies mit der Funktion `t`, also `t`

durch `t(A)`. Liegen uns zwei Matrizen `A` und `B` vor, die wir miteinander multiplizieren wollen, benutzen wir den Operator `%*%`.

`%*%`

Beispiel 2.7. Wir erzeugen zunächst zwei Matrizen durch

```
> A<-matrix(1:9,3,3)
> B<-matrix(9:1,3,3)
```

Nun soll die erste Matrix transponiert werden. Dies geschieht mit der Funktion `t`:

```
> A
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

> t(A)
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

Wir wollen die Matrizen `A` und `B` multiplizieren:

```
> A%*%B
      [,1] [,2] [,3]
[1,]   90   54   18
[2,]  114   69   24
[3,]  138   84   30

> B%*%A
      [,1] [,2] [,3]
[1,]   30   84  138
[2,]   24   69  114
[3,]   18   54   90
```

◇

Auf Matrizen kann man verschiedene Funktionen anwenden, so z.B. `sum`, `min` und `max`, um die Summe aller Matrixeinträge zu berechnen oder deren kleinstes bzw. größtes Element zu finden. Wollen wir diese (oder andere) Funktionen nur auf Zeilen oder Spalten anwenden, so können wir, wie oben beschrieben, die Matrix entsprechend indizieren und auf den so entstandenen Vektor die Funktion wirken lassen. Eine etwas kürzere Vorgehensweise bietet in einigen Fällen die Funktion `apply`, die drei obligatorische Argumente besitzt: Den Datensatz `X`, die Dimension `MARGIN` von `X`, auf die man die Funktion `FUN`, die als letztes Argument eingetragen wird, anwenden möchte. Der Befehl lautet also

`apply`

```
> apply(X,MARGIN,FUN)
```

Ist die anzuwendende Funktion die Addition `sum`, gelangt man noch schneller zum Ziel durch die beiden Funktionen `colSums` und `rowSums`, die die Summen der Spalten bzw. der Zeilen bilden.

`colSums`
`rowSums`

Beispiel 2.8. Schauen wir uns noch einmal die Körpermaße der Studenten an. Wenn wir wissen wollen, wie groß bzw. wie klein der größte bzw. der kleinste Student ist, haben wir mehrere Möglichkeiten:

```
> min(maße[,1])
[1] 163

> apply(maße,2,min)
[1] 163 50 37
> (apply(maße,2,min))[1]
[1] 163

> max(maße[,1])
[1] 201

> (apply(maße,2,max))[1]
[1] 201
```

Wir könnten uns auch für das arithmetische Mittel der einzelnen Merkmale interessieren. Auch dafür können wir uns verschiedene Befehle und Befehlsfolgen zu Nutze machen:

```
> sum.maße<-apply(maße,2,sum)
> sum.maße
[1] 3569 1407 836
> length(maße[,1])
[1] 20
> sum.maße/length(maße[,1])
[1] 178.45 70.35 41.80

> colSums/length(maße[,1])
[1] 178.45 70.35 41.80
```

Schneller geht das ganze allerdings mit der Funktion `mean`, die von den in sie eingesetzten Daten das arithmetische Mittel bildet. Wir könnten also auch

`mean`

```
> apply(maße,2,mean)
```

eingeben.

◇

Bei manchen Datensätzen bietet es sich an, eine Strukturierung mit mehr als zwei Dimensionen zu wählen. Dies kann zum Beispiel dann der Fall sein, wenn man mehrere Merkmale bei einem Datensatz berücksichtigen und untersuchen möchte. In dem eingebauten Datensatz `UCBAdmissions` z.B. sind die absoluten Häufigkeiten von Bewerbern für die Graduate School in Berkeley 1973 für die wichtigsten sechs Fachbereiche unter Berücksichtigung von Zulassung und Geschlecht. Dieser Datensatz besitzt drei Dimensionen. Wollen wir Daten als d-dimensionalen Datensatz, also als Array, in R eingeben, gehen wir ähnlich vor wie bei der Matrizenbildung. Mit `array(data,dim,dimnames)` können wir einen Array erzeugen, indem wir wie bei Matrizen die Daten als Vektor `data` eingeben und dann mit `dim=c(n1,...,nk)` spezifizieren, wie viele Dimensionen der Array besitzen soll (gegeben durch die Länge `k` von `dim`) und welchen Umfang die jeweilige Dimension haben soll (`ni` mit $1 \leq i \leq k$). Mit dem Argument `dimnames` können wir die Dimensionen benennen.

array

Liegen uns aber Datensätze mit qualitativen (und quantitativen) Merkmalen vor, können wir diese nicht mehr in Form einer Matrix oder eines Arrays eingeben, wie folgendes Beispiel zeigt:

Beispiel 2.9. Wir befragen zwei Frauen und einen Mann nach ihrem Alter und wollen eine Tabelle der Form

```
w 23
m 24
w 22
```

erstellen. Wenn wir dies mit der Funktion `matrix` versuchen, ergibt sich Folgendes:

```
> nur.so<-matrix(c("w","m","w",23,24,22),3,2)
> nur.so
  [,1] [,2]
[1,] "w" "23"
[2,] "m" "24"
[3,] "w" "22"
```

◇

Solche Datensätze geben wir in R als Datentabelle mit dem Befehl `data.frame` ein. In diesen speist man die angestrebten Spalten als Vektoren ein, denen man direkt Namen geben kann. Zeichenketten werden bei der Erzeugung einer Datentabelle automatisch durch das Programm in Faktoren umgewandelt. Wenn wir die Daten aus dem vorigen Beispiel als Tabelle darstellen wollen, gehen wir also folgendermaßen vor:

data.frame

```
> gesch.alter<-data.frame(Geschlecht=c("w","m","w"),Alter=c(23,24,22))
> gesch.alter
```

2 Der Einstieg

```
      Geschlecht Alter
1           w     23
2           m     24
3           w     22
```

Wie bei Matrizen kann man auch bei Datentabellen durch Indizierung auf einzelne oder mehrere Elemente des Datensatzes zurückgreifen:

```
> gesch.alter[2,2]
[1] 24

> gesch.alter[3,]
      Geschlecht Alter
3           w     22

> gesch.alter[,1]
[1] w m w
Levels: m w
```

Datentabellen sind Listen, auf deren Komponenten wir entweder mit einer doppelten eckigen Klammer `[[.]]` oder mit `Name.der.Liste$Name.der.Komponente` zugreifen. Die Indizierung mit einfachen eckigen Klammern liefert uns eine gestutzte Tabelle:

```
> gesch.alter[2]
      Alter
1       23
2       24
3       22

> gesch.alter[[2]]
[1] 23 24 22

> gesch.alter$Alter
[1] 23 24 22
```

Diese Eingabe ist ein wenig umständlich, insbesondere dann, wenn man auf viele verschiedene Komponenten zurückgreifen will. Die Funktion `attach` hilft, dieses Problem zu umgehen. Man setzt den Namen einer Datentabelle als Argument in diese Funktion ein und kann dann auf die in der Datentabelle enthaltenen Variablen unter ihrem Namen zugreifen, ohne den Namen der Datentabelle mit eingeben zu müssen. Diese Zugriffsmöglichkeit bleibt für die betreffende Datentabelle während der gesamten Sitzung erhalten, es sei denn, man hebt sie durch die Funktion `detach` wieder auf.

`attach`

`detach`

```
> attach(gesch.alter)
```

2 Der Einstieg

```
> Geschlecht
[1] w m w
Levels: m w

> Alter
[1] 23 24 22

> detach(gesch.alter)

> Geschlecht
Fehler: objekt Geschlecht nicht gefunden

> Alter
Fehler: objekt Alter nicht gefunden
```

Benutzt man bei einer Sitzung mehrere Datensätze und Datentabellen, auf die die Funktion `attach` angewendet wurde, kann es vorkommen, dass mehrere Variablen denselben Namen haben. In diesem Fall kann es Probleme beim Zugriff auf die Variablen geben, da R in der Regel die Variable auswählt, die zuerst unter diesem Namen eingeführt wurde, was nicht immer die ist, die man gerne aufrufen würde. In einem solchen Fall sollte man auf alle anderen Datentabellen als die gerade genutzte die Funktion `detach` anwenden und sonstige Variablen mit diesem Namen umbenennen. Letzteres gelingt mit Hilfe des Zuweisungsoperators: `neuer.Name<-alter.Name` und der Funktion `rm`, mit der man nach der Umbenennung die alte Variable entfernt. Gibt es beispielsweise zwei Variablen mit dem Namen `Alter`, macht man dieses:

```
> age<-Alter
> rm(Alter)
```

Die Variable, die als erstes mit `Alter` bezeichnet wurde, heißt nun `age` und kann unter diesem Namen aufgerufen werden, während man unter `Alter` nun die zweite erhält.

Mit Hilfe der Funktion `split(X,Y)` kann man aus einer Datentabelle eine Variable `X` auswählen und in Teilmengen aufspalten, die von der Ausprägung einer zweiten Variable `Y` abhängen. Im obigen Beispiel könnten wir die Variable `Alter` auswählen und unter dem Kriterium `Geschlecht` in zwei Mengen unterteilen: `split`

```
> attach(gesch.alter)
> split(Alter,Geschlecht)
$m
[1] 24

$w
[1] 23 22
```

2 Der Einstieg

Eine weitere Möglichkeit, aus einer Tabelle kleinere Datensätze auszuwählen, bietet die Funktion `subset` (Teilmenge), die man mit dem Befehl

`subset`

```
> subset(x,condition)
```

aufruft. Wollen wir bei der Auswertung der Körpermaße der Studenten speziell die Daten derer betrachten, die mindestens 1.80m groß sind, geben wir in R die Daten als Datentabelle (nicht als Matrix) ein, die wieder `maße` heißen soll, und erhalten mit der Funktion `subset`:

```
> subset(maße,Körpergröße>=180)
  Körpergröße Gewicht Schuhgröße
2          180      80          44
5          182      73          44
6          180      70          41
7          180      77          43
11         201      93          48
12         180      67          42
13         183      73          42
16         182      85          40
17         180      80          41
18         190      83          44
19         180      67          39
20         183      75          45
```

Interessieren wir uns bei diesem neuen Datensatz wiederum nur für die Schuhgröße, wählen wir diese durch das optionale Argument `select` in `subset` aus:

```
> subset(maße,Körpergröße>=180,select=Schuhgröße)
  Schuhgröße
2          44
5          44
6          41
7          43
11         48
12         42
13         42
16         40
17         41
18         44
19         39
20         45
```

Man kann in das Argument `condition` natürlich auch mehrere Bedingungen eingeben,

die man dann allerdings entsprechend mit den Operatoren `&` und `|` verknüpfen muss.

2.2 Funktionen

Häufig muss man kompliziertere Rechnungen an mehreren Werten oder Datensätzen durchführen und sie für die Durchführung in R durch eine lange Befehlsfolge eingeben. Um zu vermeiden, eine solche Folge mehrmals eingeben zu müssen, kann man sich in R selber eine entsprechende Funktion definieren. Dies geschieht mit diesem Befehl:

```
> name<-function(Argumente){Körper der Funktion return(Ergebnis)}           function
```

Die Vorgehensweise lässt sich an einem Beispiel leicht verdeutlichen:

Beispiel 2.10. In Deutschland werden Temperaturen üblicherweise in Grad Celsius angegeben, wohingegen es in den USA üblich ist, die Einheit Fahrenheit zu verwenden. Zwischen beiden Temperaturskalen herrscht folgender Zusammenhang:

$$y = \frac{9}{5}x + 32,$$

falls x die Gradzahl in Celsius und y die Gradzahl in Fahrenheit bezeichnet. Diese Beziehung wollen wir ausnutzen und als Funktion in R einspeisen, die Celsius-Werte in Fahrenheit umrechnet. Sie soll `fahrenheit` heißen und besitzt nur ein Argument, den Wert in Grad Celsius, der umgerechnet werden soll. Also programmieren wir:

```
> fahrenheit<-function(x){return(9/5*x+32)}
```

und in diese Funktion können wir nun beliebige reelle Werte einsetzen:

```
> fahrenheit(40)
[1] 104
> fahrenheit(232.7778)
[1] 451
```

◇

Im sogenannten Körper der Funktion stehen noch weitere Angaben zu den Argumenten und der Wirkungsweise der Funktion sowie Definitionen von Variablen, die in der Funktion benutzt werden. Häufig müssen in die Definition einer Funktion Fallunterscheidungen bzgl. des Outputs in Abhängigkeit von den eingesetzten Werten vorgenommen werden. Dazu verwendet man Anweisungen der Art

```
if(Bedingung){Befehlsfolge 1}else{Befehlsfolge 2}
```

Beispiel 2.11. Es soll die Indikatorfunktion auf dem abgeschlossenen Einheitsintervall programmiert werden. Also schreiben wir

```
> indikator<-function(x){if(x<0){return(0)}
                        else{if(x>1){return(0)}
                              else{return(1)}}}
```

Dieses Beispiel zeigt, wie Fallunterscheidungen zwischen mehr als zwei Fällen behandelt werden können. Da die Indikatorfunktion aber eigentlich nur zwischen zwei Fällen unterscheidet, kann man einfacher definieren:

```
> indikator<-function(x){if(x<0|x>1){return(0)}else{return(1)}}
```

◇

Der Befehl `return` vermittelt dem Programm, welchen Wert es in dem genannten Fall auswerfen soll. Konstruieren wir nun eine Funktion, die nicht für alle reellen Zahlen definiert ist, brauchen wir für die Werte außerhalb des Definitionsbereichs eine Fehlermeldung. In der `if-else` Befehlsfolge müssen wir diesen Fall berücksichtigen und geben bei `return` Folgendes ein:

```
if(Bedingung){return(cat(„Fehler: x liegt nicht im Definitionsbereich"))}
else{...}
```

Beispiel 2.12. Bei unserem ersten Beispiel zur Umrechnung von Grad-Celsius-Werten in Fahrenheit müssen wir noch eine Fallunterscheidung einfügen, da es keine Temperaturen unter $-273,16^{\circ}\text{C}=0^{\circ}\text{K}$ gibt. Unsere Funktion sollte also folgendermaßen aussehen:

```
fahrenheit<-function(x){if(x<-273,16){return(cat(„Nicht definiert!"))}
                        else{return(9/5*x+32)}}}
```

◇

Es gibt in R einen Editor, der die Eingabe von Funktionen erleichtert. Man ruft ihn mit dem Befehl `fix(Name der Funktion)` auf.

2.3 Exkursion zum Programmieren

Im vorigen Abschnitt haben wir schon die `if`-Aussage kennengelernt. Mit den Befehlen `for` und `while` kann man Schleifenprozesse programmieren, also solche Vorgänge, die vom Programm immer wiederholt werden, bis eine gewisse Bedingung erfüllt ist.

Mit der `for`-Aussage können wir festlegen, dass eine Operation eine bestimmte Anzahl von Malen wiederholt wird. Sie unterliegt folgender Syntax:

```
for(Name in Vektor){Befehle}.
```

for

Dadurch wird eine Variable, die Name heißt, schrittweise gleich den Elementen des

Vektors gesetzt. In jedem Schritt wird für den entsprechenden Wert des Vektors der zugehörige Befehl aus den geschweiften Klammern ausgeführt. Mit den geschweiften Klammern kann man die Befehle gruppieren, so dass R sie wie einen einzelnen Befehl behandelt. Gibt es nur einen Befehl auszuführen, sind die Klammern unnötig.

Beispiel 2.13. Wir wollen die ersten zwölf Fibonacci-Zahlen erzeugen und geben dazu ein:

```
> Fibonacci <- numeric(12)
> Fibonacci[1] <- Fibonacci[2] <- 1
> for (i in 3:12) Fibonacci[i] <- Fibonacci[i-2]+Fibonacci[i-1]
```

Durch den ersten Befehl erhalten wir einen numerischen Vektor der Länge zwölf bestehend aus Nullen, den wir mit `Fibonacci` benennen. Mit dem zweiten Befehl verändern wir die ersten beiden Einträge von `Fibonacci` zu Einsen. Der dritte Befehl definiert die restlichen zehn Einträge als Summe ihrer zwei unmittelbaren Vorgänger.

```
> Fibonacci
[1] 1 1 2 3 5 8 13 21 34 55 89 144
```

◇

Nun kann es sein, dass man einen Vorgang wiederholen möchte, aber im Vorfeld das Wiederholungsmuster noch nicht genau kennt, sondern nur weiß, dass die Wiederholungen stattfinden sollen, solange eine bestimmte Bedingung erfüllt ist. In einem solchen Fall kann man die `while`-Aussage verwenden, die folgende Syntax besitzt:

```
while(Bedingung){Aussagen} while
```

In jedem Schritt wird die Bedingung überprüft. Solange `TRUE` gilt, werden die Aussagen ausgeführt, tritt jedoch `FALSE` ein, wird die Schleife beendet.

Beispiel 2.14. Wir wollen alle Fibonacci-Zahlen auflisten, die kleiner als 300 sind. Da wir nicht ohne weiteres wissen, für wie viele dies zutrifft, können wir nicht mit `for` arbeiten, sondern greifen auf `while` zurück:

```
> Fib1 <- 1
> Fib2 <- 1
> Fibonacci <- Fib1
> while (Fib2<300){
  Fibonacci <- c(Fibonacci,Fib2)
  oldFib2 <- Fib2
  Fib2 <- Fib1+Fib2
  Fib1 <- oldFib2
}
```

die Zentrale Idee dabei steckt in den Zeilen vier und fünf: solange die letzte Fibonacci-Zahl kleiner als 300 ist, hängen wir sie an den wachsenden Vektor `Fibonacci`. Wir müssen also sicherstellen, dass `Fib2` immer die aktuelle Fibonacci-Zahl ist. Dazu werden immer die letzten beiden Zahlen festgehalten und addiert und entsprechend umbenannt. Dies geschieht in den letzten drei Zeilen. Damit das Ganze funktionieren kann, brauchen wir wieder Startwerte, die in den ersten drei Zeilen festgelegt werden.

◇

2.4 Grafiken

Im Wesentlichen gibt es zwei Befehle, mit denen man Grafiken und Graphen erstellen kann: `plot` und `curve`. Tatsächlich gibt es natürlich sehr viele, aber mit diesen beiden kann man auch schon eine ganze Menge anstellen...

In die Funktion `plot` werden Punkte bzw. daraus erzeugte Vektoren als Argumente eingesetzt, und (ohne Einstellung weiterer Parameter) zeichnet R ein Koordinatensystem, in dem diese Punkte eingetragen sind. Wollen wir die Punkte $(x_1, y_1), \dots, (x_n, y_n)$, $n \in \mathbb{N}$, darstellen, geben wir zwei Argumente in `plot` ein: Als erstes den Vektor mit sämtlichen x-Koordinaten und als zweites den Vektor der y-Koordinaten (in der richtigen Reihenfolge!!!). In der Grafik, die R nun ausgibt, sind diese Punkte mit einem Kreis als Symbol markiert. Mit dem Argument `pch` (Abkürzung für: plot character) mit Werten aus \mathbb{N}_0 können wir andere Symbole wählen.

Wollen wir jedoch nicht die Punkte darstellen, sondern sie durch Strecken verbunden sehen, setzen wir das Argument `type` auf "l" (Abkürzung für: line). Mit den Argumenten `xlab`, `ylab`, `main` (für label bzw. main title) werden die Achsen benannt und eine Überschrift über die Grafik gesetzt. Diese Beschriftungen werden jeweils als Zeichenketten eingegeben.

Beispiel 2.15. Wir wollen einen Streckenzug zwischen den Punkten

(2, 5) (5, 3) (9, 9) (13, 6)

zeichnen. Durch

```
> x<-c(2,5,9,13)
> y<-c(5,3,9,6)
> plot(x,y)
```

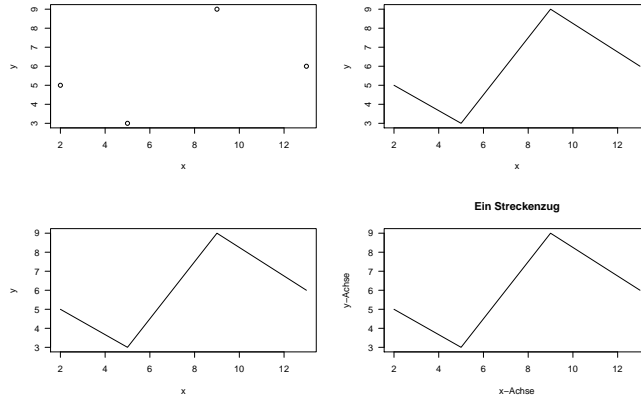
erhalten wir ein Koordinatensystem, in dem die Punkte eingetragen sind. Den Streckenzug bekommen wir durch

```
> plot(x,y,type="l")
```

2 Der Einstieg

und die Achsenbeschriftung und eine Überschrift mit

```
> plot(x,y,type="l",xlab="x-Achse",ylab="y-Achse",main="Ein Streckenzug")
```



◇

Man sieht, dass die Beschriftung der Achsen parallel zu den Achsen gewählt wird, was in den USA üblich und auch der Standard in R ist. Soll die Beschriftung der Ordinate orthogonal zur Achse (also waagrecht) sein, muss man

```
par(las=1)
```

eingeben, was hier vor dem Erstellen der dritten Grafik durchgeführt wurde. Diese Einstellung bleibt während der gesamten Sitzung erhalten, wenn man sie nicht aktiv umändert. `par` steht hier für „parameter“ und besitzt viele optionale Argumente.

Mit `bty` können wir z.B. den „box type“ einstellen. Standardmäßig zeichnet R eine geschlossene Box, deren linke und untere Seiten der Ordinate und Abszisse entsprechen. `bty` kann die Werte „o“, „l“, „7“, „u“ und „]“ annehmen und die resultierenden Boxen gleichen dem zugehörigen Großbuchstaben bzw. der Ziffer. Um also nur die beiden Achsen zu zeichnen, setzt man `bty="l"`.

`cex` (Abk. für: character expansion) gibt die Größe der Beschriftungen und Symbole in der Grafik an. Es gibt noch die Argumente `cex.axis`, `cex.lab`, `cex.main`, `cex.sub`, die (relativ zu `cex`) die Größe der Achsenbeschriftung, der Benennung der Achse, des Titels und des Untertitels festlegen.

Den Linientyp können wir mit dem Argument `lty` (Abk. für: line type) einstellen, das Werte von 0 bis 6 annehmen kann:

0	1	2	3	4	5	6
blank	solid	dashed	dotted	dotdash	longdash	twodash

`par`
`bty`

`cex`
`cex.axis`
`cex.lab`
`cex.sub`
`lty`

Man kann den Linientyp auch selber variieren, indem man bis zu vier Strichlängen und Pausen numerisch bestimmt. Näheres dazu findet man im Hilfetext.

Mit `lwd` (Abk. für: line width) können wir die Breite der zu zeichnenden Linien einstellen, die standardmäßig die Breite 1 haben.

`lwd`

Möchte man mehrere Grafiken auf einer Seite erstellen (und nicht jede auf einer eigenen Seite), kann man dies mit `mfrow` bzw. `mfcol` festlegen (matrix frame by row/column o.ä.). Hierbei muss man eingeben, in wie viele Zeilen und Spalten man die Seite aufteilen möchte:

`mfrow``mfcol`

```
> par(mfrow=c(nrow,ncol))
```

Auf der so eingerichteten Seite ist nun Platz für `nrow` mal `ncol` Grafiken, die in diesem Fall zeilenweise (`mfrow`) eingetragen werden.

Die mit `par` vorgenommenen Einstellungen bleiben die gesamte Sitzung über erhalten. Einige von den Parametern kann man aber auch als Argumente in `plot` eingeben, so dass sie nur für die damit erstellte Grafik festgelegt werden, so dass sich jede Grafik leicht individuell gestalten lässt. (Offensichtlich gilt dies nicht für die Argumente `mfrow` und `mfcol`.)

Mit der Funktion `seq(from,to,by)` wird ein Vektor erzeugt, der aus einer Folge von äquidistanten Werten besteht: `from` ist der erste Wert und es folgen Werte im Abstand `by` bis zum Punkt `to`. Ist letzterer nicht ein Vielfaches von `by` von `from` entfernt, wird der nächstkleinere Wert gewählt, für den dies gilt.

`seq`

Beispiel 2.16. Um den Vektor $(0, 5, 10, \dots, 95, 100)$ zu erzeugen, geben wir ein:

```
> seq(0,100,5)
```

Das gleiche Ergebnis erhalten wir mit

```
> seq(0,103,5)
```

nicht jedoch mit

```
> seq(-2,100,5)
```

da man damit den Vektor $(-2, 3, 8, \dots, 98)$ erhält.

◇

Anstatt den Abstand zwischen den einzelnen Einträgen des Vektors zu bestimmen, kann man auch die Länge des Vektors festlegen: `seq(from,to,length)`. Im obigen Beispiel hätten wir auch `> seq(0,100,length=21)` eingeben können, um zum gleichen Ergebnis zu gelangen.

Diese letzte Eigenschaft kann man sich zunutze machen, um mit der Funktion `plot` Kurven zu zeichnen. Angenommen, wir wollen die Funktion $y=f(x)$ auf dem Intervall $[a,b]$

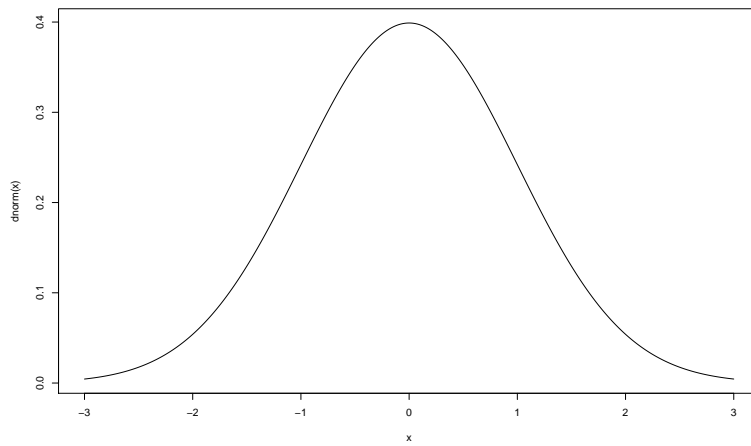
2 Der Einstieg

plotten. Dann erzeugen wir mit Hilfe von `seq(a,b,length=10000)` einen sehr langen Vektor `x`. Mit `plot(x,f(x),type="l")` wird dann zwar ein Streckenzug geplottet, der aber dank der hohen Anzahl von Punkten (wenn es die Funktion `f` zulässt) glatt aussieht.

Beispiel 2.17. Wir wollen die Dichte der Standardnormalverteilung auf dem Intervall `[-3,3]` zeichnen. Die Dichte erhält man in R durch den Befehl `dnorm`. Mit

```
> x<-seq(-3,3,length=10000)
> plot(x,dnorm(x),type="l")
```

bekommen wir diese Grafik:



◇

Mit der Funktion `curve(expr,from,to,...)` gelangt man schneller (und exakter) ans Ziel. Man gibt die zu zeichnende Funktion (in Abhängigkeit von `x`) als `expr` (Abk. für: expression) ein und bestimmt den zu zeichnenden Bereich mit `from` und `to`. In diese Funktion kann man die gleichen optionalen Argumente zur Parametereinstellung eingeben wie bei `plot`.

curve

Bis jetzt können wir also Punkte, Streckenzüge und Kurven grafisch darstellen. Häufig möchte man aber vergleichend mehrere Kurven in ein Diagramm eintragen, wozu wir in R die Funktion `lines` verwenden. Wenn wir mit `plot` oder `curve` eine Grafik erstellt haben, können wir mit `lines` noch weitere Linien eintragen. Sie wird genauso gehandhabt wie die Funktion `plot`, ist aber immer auf den Linienmodus gestellt. Punkte kann man mit `points` nachträglich in eine Grafik einzeichnen; mit `text` kann man in die Grafik weitere Beschriftungen (oder auch Symbole) eintragen. Hat man nun eine Grafik erstellt, in der mehrere Kurven oder Punktwolken dargestellt sind, sollte man der Übersichtlichkeit halber eine Legende einfügen. Dies geschieht mit

lines

points

`legend(x,y,c(„erster Name“,...,„letzter Name“),lty=c(...),col=c(...),...)` legend

wobei `x,y` die Koordinaten sind, an der die Legende eingefügt werden soll. Danach wird aufgezählt, wodurch die unterschiedlichen Kurven gekennzeichnet sind, z.B. durch einen bestimmten Linientyp oder eine Farbe. (Natürlich sollten nur die Merkmale als Argument eingegeben werden, die als Unterscheidungskriterium dienen.)

Eine Grafik in R beginnt nicht beim Minimum der Beobachtungen und endet nicht beim Maximum, sondern lässt an beiden Seiten noch ein wenig Platz, da sonst bei der Darstellung einer Punktwolke das Minimum und das Maximum nicht mehr sichtbar wären. Setzt man jedoch die Argumente `xaxis` und `yaxis` auf den Wert „i“ (Abk. für: axis bzw. initial), wird nur der tatsächliche Bereich geplottet. Mit `xlim=c(.,.)` und `ylim=c(.,.)` kann man den Bereich der Grafik von Hand festlegen.

`xaxis yaxis`
`xlim ylim`

2.5 Verteilungen

Im vorigen Abschnitt haben wir schon die Dichte einer Standardnormalverteilung gesehen, die in R durch `dnorm` aufgerufen wird. Dabei steht das `d` für „density“ und `norm` für „normal distribution“. Setzt man anstelle des `d` ein `p` oder ein `q` vor `norm`, erhält man die Verteilungsfunktion bzw. das Quantil an den jeweiligen Stellen, die man als Argumente in die so entstandenen Funktionen einsetzt. Diesem Prinzip unterliegen die meisten (gängigen) Verteilungen in R: Sie besitzen einen Namen, vor den man ein `d`, `p` oder `q` setzt und damit die Dichte, Verteilungsfunktion oder Quantilsfunktion erhält. In diese drei kann man einzelne Punkte einsetzen oder auch Vektoren und erhält entsprechend den Wert oder einen Vektor der zugehörigen Werte. In die Quantilsfunktion setzt man allerdings Wahrscheinlichkeiten ein!

`d p q`

Üblicherweise hängen Verteilungen von gewissen Parametern ab, die man in R mit eingeben muss. Für einige Verteilungen gibt es allerdings auch Standardwerte, auf die die Parameter automatisch gesetzt werden, wenn man nichts anderes eingibt. Dies gilt z.B. für die Normalverteilung. Diese wird über ihren Mittelwert und die Varianz definiert bzw. gibt man in R statt der Varianz die Standardabweichung (=Wurzel aus der Varianz) ein. Unterlässt man dies, werden automatisch 0 als Mittelwert und 1 als Standardabweichung gewählt. Hier eine Tabelle der wichtigsten Verteilungen und ihrer Parameter:

Verteilung	Name in R	Parameter in R	Parameter
$\beta(a, b)$	beta	shape1	a
		shape2	b
C(a, b)	cauchy	location(= 0)	a
		shape(= 1)	b
χ_n^2	chisq	df	n
Exponential	exp	rate(= 1)	λ

2 Der Einstieg

Verteilung	Name in R	Parameter in R	Parameter
F(m, n)	<code>f</code>	<code>df1</code> <code>df2</code>	m n
$\Gamma(n, \lambda)$	<code>gamma</code>	<code>shape</code> <code>rate(= 1)</code>	n λ
Gleich $R(a, b)$	<code>unif</code>	<code>min</code> <code>max</code>	a b
logistisch	<code>logis</code>	<code>location(= 0)</code> <code>scale(= 1)</code>	
lognormal	<code>lnorm</code>	<code>meanlog(= 0)</code> <code>sdlog(= 1)</code>	μ σ
$\mathcal{N}(\mu, \sigma^2)$	<code>norm</code>	<code>mean(= 0)</code> <code>sd(= 1)</code>	μ σ
t (k Freiheitsgr.)	<code>t</code>	<code>df</code>	k
$B(n, p)$	<code>binom</code>	<code>size</code> <code>prob</code>	n p
Geometrisch	<code>geom</code>	<code>prob</code>	p
Hypergeom.	<code>hyper</code>	<code>m</code> <code>n</code> <code>k</code>	M $N - M$ n
$NB(\alpha, \theta)$	<code>nbinom</code>	<code>size</code> <code>prob</code>	α θ
Poisson	<code>pois</code>	<code>lambda</code>	λ

Beispiel 2.18. Zur Veranschaulichung der Vorgehensweise hier ein paar Anwendungen: Wir wollen die Verteilungsfunktion der $B(10, 0.5)$ -Verteilung an der Stelle 5 ausrechnen:

```
> pbinom(5, size=10, prob=0.5)
[1] 0.6230469
```

oder die α -Quantile der $\mathcal{N}(2, 9)$ -Verteilung an den Stellen $\alpha=0.25, 0.5, 0.75$:

```
> qnorm(c(0.25, 0.5, 0.75), mean=2, sd=3)
[1] -0.02346925 2.00000000 4.02346925
```

◇

Setzt man den Buchstaben **r** (Abk. für: randomize) vor den Namen einer Verteilung, erhält man eine Funktion, die Zufallszahlen gemäß der entsprechenden Verteilung generiert. Die Argumente dieser Funktion sind die gewünschte Anzahl an Zufallszahlen und die Parameter der Verteilung. **r**

Haben wir Daten vorliegen, aus denen wir eine Zufallsstichprobe (also gemäß einer Laplaceverteilung) ziehen wollen, greifen wir auf die Funktion `sample(x, size, replace(=F))` (engl. für Zufallsstichprobe) zurück. Mit dieser Funktion wird aus dem Datenvektor **x** eine Zufallsstichprobe vom Umfang **size** gezogen. Das Argument `replace`, das standardmäßig auf `FALSE` steht, gibt dabei an, ob mit Zurücklegen gezogen wird. **sample**

2.6 Pakete

R ist ein offenes Programm, sodass es durch Funktionen, die von Benutzern erstellt wurden, erweitert werden kann. Diese Funktionen sind in Paketen enthalten. Um eine Funktion aus einem Paket benutzen zu können, muss man das Paket installieren und laden. Man installiert ein Paket, indem man auf den Schalter **Pakete** und danach auf den Schalter **Installiere Paket(e)** klickt (wenn man dies noch nicht vorher getan hat, muss man sich zunächst einen CRAN mirror aussuchen, also einen Ort, von dem man die Daten herunterladen möchte). Danach öffnet sich eine Liste mit Paketen, aus der man das entsprechende auswählt. Dazu muss eine Verbindung zum Internet vorhanden sein. Das Programm fragt schließlich, ob das Paket in eine library eingebunden werden soll, was man tun muss, um die Funktionen des Paketes verwenden zu können. Dies kann man auch über den Befehl `library(Name des Paketes)` erreichen. Man muss ein Paket nur einmal installieren, muss es aber bei jeder Sitzung einmal laden, wenn man es verwenden will. Dazu klickt man wieder auf **Pakete** und dann auf **Lade Paket**. Arbeitet man nicht mit RGui, sondern in einer Konsole ohne Menüleiste, kann man die Installation mit dem Befehl

```
> install.packages("Name des Paketes")
```

```
install.  
packages
```

vornehmen und dann das Paket mit folgendem Befehl laden:

```
> library("Name des Paketes")
```

```
library
```

3 Univariate Analyse

In diesem Kapitel werden wir uns nur mit Erhebungen *eines* Merkmals befassen. Die Auswertung solcher Erhebungen bezeichnet man als **univariate Datenanalyse**. Im Folgenden beschäftigen wir uns zunächst mit der Darstellung solcher Datensätze und gehen dann über zu ihrer Beschreibung.

3.1 Darstellung univariater Datensätze

Zunächst wollen wir uns damit beschäftigen, wie man einen Datensatz möglichst übersichtlich darstellen kann. Handelt es sich um einen eher kleinen Datensatz aus nur wenigen Beobachtungen bzw. Messwerten, so kann es schon ausreichen, diese einfach aufzuzählen. Mit wachsender Anzahl der Werte werden solche Aufzählungen aber sehr schnell unübersichtlich, weshalb man auf eine Darstellung mithilfe von Tabellen und Grafiken zurückgreift. Dabei hängt die Darstellungsweise davon ab, ob es sich um ein qualitatives oder ein quantitatives Merkmal handelt. Wir beschäftigen uns hier zuerst mit dem Fall eines qualitativen Merkmals, also eines Merkmals, das kategorial und weder Intervall- noch verhältnisskaliert ist.

3.1.1 Darstellung qualitativer Merkmale

Bei qualitativen Merkmalen unterscheiden wir zwischen nominalskalierten und ordinalskalierten Merkmalen. Beide sind in Kategorien unterteilt und erlauben fast dieselben Darstellungsarten, wobei bei ordinalskalierten Merkmalen beachtet werden muss, dass es eine Ordnungsstruktur zwischen den Klassen gibt.

Wir betrachten ein qualitatives Merkmal mit k Merkmalsausprägungen a_1, a_2, \dots, a_k . Normalerweise trifft auf jeden Merkmalsträger genau eine der Merkmalsausprägungen zu, doch bei Befragungen werden häufig Fragen gestellt, bei denen der Befragte mehrere Antworten ankreuzen kann. In diesem Fall spricht man von **Mehrfachantworten**. Bei Fragen mit Mehrfachantworten ordnet man jeder möglichen Antwort ein eigenes Merkmal mit den Merkmalsausprägungen **ja** und **nein** zu, wodurch sichergestellt wird, dass jeder Merkmalsträger bei jedem Merkmal genau eine der Merkmalsausprägungen aufweist. Merkmale mit genau zwei Ausprägungen nennt man **binär** oder **dichotom**.

Beispiel 3.1. Bei einer Fahrgastumfrage der Bahn sollen die Fahrgäste unter anderem angeben, zu welchem Zweck sie die Zuglinie nutzen, und können Mehrfachantworten geben:

3 Univariate Analyse

Fahrt zum Arbeitsplatz	[]	Einkauf/Shopping	[]
Fahrt zum Studium/Schule	[]	Urlaub	[]
Besuch von Familie/Freunden	[]	Sonstiges	[]

Zur Auswertung der Fragebögen wandelt man die sechs Merkmalsausprägungen **Fahrt zum Arbeitsplatz**, ..., **Sonstiges** in Merkmale um, die die Merkmalsausprägungen **ja** und **nein** besitzen, man erhält also:

	ja	nein
Fahrt zum Arbeitsplatz	[]	[]
Fahrt zum Studium/Schule	[]	[]
Besuch von Familie/Freunden	[]	[]
Einkauf/Shopping	[]	[]
Urlaub	[]	[]
Sonstiges	[]	[]

◇

Bei einer Befragung von n Merkmalsträgern erhalten wir für jede Merkmalsausprägung a_i , $1 \leq i \leq k$, die absolute Häufigkeit n_i , also die Anzahl der Merkmalsträger, die diese Merkmalsausprägung aufweisen, und die relative Häufigkeit

$$f_i := \frac{n_i}{n}.$$

Bei dichotomen Merkmalen reicht es völlig aus, die relative Häufigkeit einer der beiden Ausprägungen anzugeben, da dadurch natürlich die zweite eindeutig ist. In allen anderen Fällen empfiehlt es sich, alle relativen Häufigkeiten anzugeben. Dabei verwendet man üblicherweise Prozentangaben, da es im Allgemeinen schwieriger ist, Nachkommastellen zu vergleichen. Da schon bei wenigen Daten deren Angabe im Fließtext sehr unübersichtlich wird, greift man auf eine Darstellung in Tabellenform zurück. Eine Häufigkeitstabelle hat dabei gewöhnlich diese Form:

Merkmalsausprägung	absolute Häufigkeit	relative Häufigkeit
a_1	n_1	$100 \cdot f_1$
\vdots	\vdots	\vdots
a_k	n_k	$100 \cdot f_k$

Bei einer Präsentation der Daten in Tabellenform sollte man berücksichtigen, dass viele Dezimalstellen (wenn es sich nicht gerade um Nullen handelt) die Aufnahme der Daten erheblich erschweren. Daher sollte man die Daten runden, möglichst auf zwei effektive Stellen. Aus der Zahl 56489 wird dann 56000, aus 0.000216 wird 0.00022.

Die in der Häufigkeitstabelle enthaltenen Informationen sind noch leichter zu erfassen, wenn man sie grafisch darstellt, wobei den relativen Häufigkeiten Längen von Strecken oder Flächeninhalte zugeordnet werden. Dies kann mit einem **Kreis(sektoren)-** oder **Tortendiagramm** (engl.: pie chart) geschehen, aber auch in Form eines (**geordneten**) **Säulendiagrammes**. Eine solche Darstellung ist intuitiv leichter erfassbar als eine Tabelle, insbesondere in Bezug auf den Größenvergleich der relativen Häufigkeiten, die genauen Werte der Häufigkeiten gehen dabei allerdings schnell für den Betrachter verloren.

Bei nur wenigen Ausprägungsmöglichkeiten bietet sich eine Darstellung per Kreisdiagramm an, bei dem die den Ausprägungen zugeordneten Flächen proportional zur relativen Häufigkeit sind. In der einfachsten Form eines solchen Diagrammes sind die genauen Häufigkeitszahlen nicht angegeben, so dass durch diese Darstellung Informationen (für den Betrachter) verloren gehen, wohingegen bei einer Darstellung, die die konkreten Zahlen in das Diagramm einbindet, die Übersichtlichkeit leidet. Mit steigender Zahl der Ausprägungsmöglichkeiten verliert ein Kreisdiagramm weiter an Übersichtlichkeit, da dann die einzelnen Sektoren nur schwer miteinander verglichen werden können.

Um in R ein Kreisdiagramm zu erstellen, verwenden wir die Funktion `pie`, doch dazu müssen wir zunächst den darzustellenden Datensatz in eine geeignete Form bringen. Prinzipiell können wir auch einfach einen aus den relativen oder den absoluten Häufigkeiten bestehenden Vektor in die Funktion einsetzen, doch wir erhalten dann keine Beschriftung der einzelnen Sektoren, die dann einfach nur nummeriert werden. dadurch verlöre das Diagramm seine Aussagekraft und seinen Sinn. Wir brauchen also Häufigkeitstabellen, die in diesem Fall entweder nur die absoluten oder nur die relativen Häufigkeiten umfassen dürfen. Nun gibt es verschiedene Möglichkeiten, eine derartige Tabelle zu erstellen. Nehmen wir an, es liegt uns ein Faktor `x` vor, also ein Vektor bestehend aus Ausprägungen eines qualitativen Merkmals. Mit der Funktion `table` erhalten wir eine Tabelle mit den absoluten Häufigkeiten der auftretenden Faktoren. Durch die Befehlsfolge `table(x)/sum(table(x))` erhält man entsprechend eine Tabelle der relativen Häufigkeiten, die bei Multiplikation mit 100 die Prozentangaben liefert. Auf jede dieser Tabellen kann man nun die Funktion `pie` anwenden und erhält ein mit den Bezeichnungen der Faktoren beschriftetes Tortendiagramm.

Beispiel 3.2. Bei einer Umfrage kann eine Frage nur mit „ja“ (j) oder „nein“ (n) beantwortet werden. Die Daten werden als Vektor aus Zeichenketten eingegeben und in einen Faktor umgewandelt, und es ergibt sich

```
> ja.oder.nein
[1] j n n n j j n n n n
Levels: j n
```

Durch Anwendung der oben beschriebenen Vorgehensweise erhalten wir Tabellen der absoluten bzw. der relativen Häufigkeiten bzw. der prozentualen Anteile:

```
> jon<-table(ja.oder.nein)
> jon
ja.oder.nein
  j  n
3  7

> jon/sum(jon)
ja.oder.nein
```

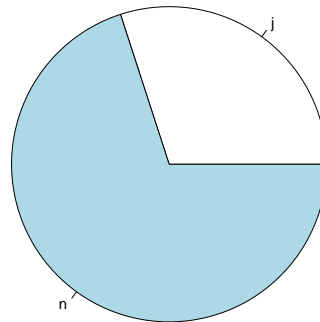
3 Univariate Analyse

```
j    n
0.3  0.7

> 100*jon/sum(jon)
ja.oder.nein
  j    n
30  70

> pie(jon)
```

liefert dann ein Tortendiagramm.



◇

Eine andere Möglichkeit zur Erzeugung einer Häufigkeitstabelle bietet sich, wenn man nicht den Datensatz der einzelnen Antworten, sondern schon die absoluten oder relativen Häufigkeiten vorliegen hat. Dann gibt man diese als (numerischen) Vektor `x` ein, dessen Komponenten mit Hilfe der Funktion `names` benannt werden können:

`names`

```
> names(x) <- c("erste Komponente", ..., "letzte Komponente")
```

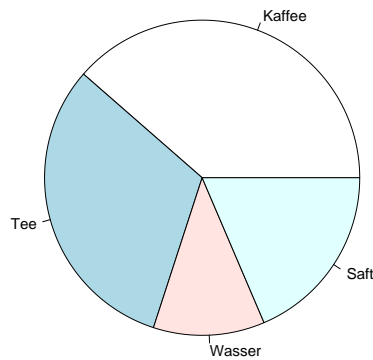
Beispiel 3.3. Bei einer Institutsfeier können die Anwesenden zwischen vier Getränkearten wählen: Es gibt Kaffee, Tee, Wasser und Saft. 27 Teilnehmer entscheiden sich für Kaffee, 22 für Tee, acht für Wasser und 13 für Saft. Aus diesen Daten bilden wir als erstes eine

3 Univariate Analyse

Häufigkeitstabelle:

```
> getränke<-c(27,22,8,13)
> names(getränke)<-c(„Kaffee“,„Tee“,„Wasser“,„Saft“)
> getränke
Kaffee Tee Wasser Saft
      27  22     8  13
```

Mit der Funktion `pie` gelangen wir dann zum gewünschten Kreisdiagramm.



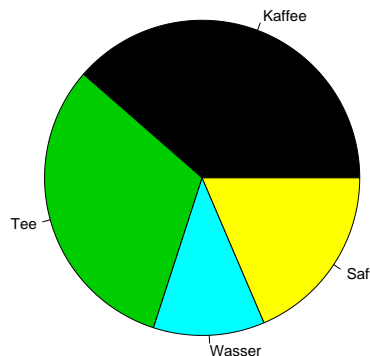
◇

Bisher haben wir in die Funktion `pie` nur das obligatorische Argument (den Datensatz) eingegeben. Dabei wählt das Programm R eigenständig für jeden der Kreissektoren eine unterschiedliche Farbe (meist Pastelltöne). Man kann allerdings den Sektoren durch das optionale Argument `col` (Abkürzung für colour) selbst Farben zuweisen, was manchmal durch den Inhalt des Datensatzes empfehlenswert wird. Will man zum Beispiel Wahlergebnisse als Kreissektorendiagramm wiedergeben, ist es vorteilhaft, die Sektoren in den entsprechenden Parteifarben einzufärben. Die einzelnen Farben sind durch Zahlen verschlüsselt. Setzt man das Argument `col` auf einen skalaren Wert, werden alle Sektoren mit demselben Farbton eingefärbt. Um zu erreichen, dass jeder Sektor eine eigene Farbe erhält, setzt man einen Vektor ein, der für jede Merkmalsausprägung genau eine Komponente besitzt, die den zur Ausprägung gehörigen Farbton angibt. Die Farbtöne sind durch die Zahlen 0 bis 8 verschlüsselt.

`col`

3 Univariate Analyse

Im obigen Beispiel könnten wir versuchen, die Farbe der Getränke in das Diagramm einfließen zu lassen. Das - zugegebenermaßen nicht gerade überwältigende - Beispiel sieht man auf der nächsten Seite. Dies kommt dadurch zustande, dass uns bei dieser Art der Farbwahl nur neun Farbtöne zur Verfügung stehen. Insgesamt können wesentlich mehr Farben gewählt werden, dazu muss man allerdings erst die entsprechenden Paletten (bzw. Pakete) öffnen (`rainbow`, `heat.colors`, `terrain.colors`, `rgb`).

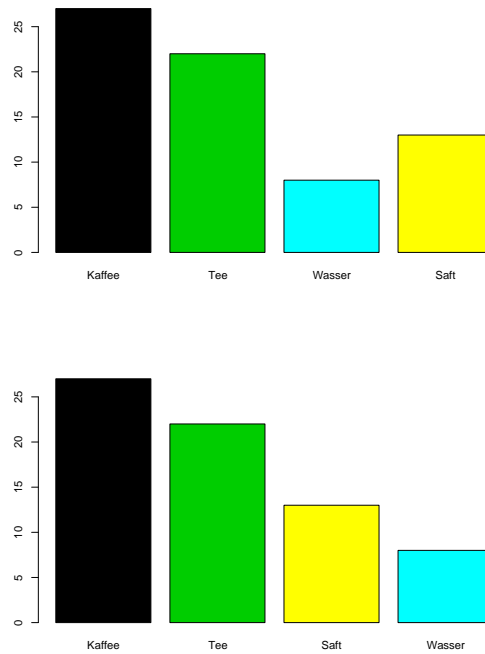


Bei vielen Merkmalsausprägungen trägt ein Tortendiagramm nicht mehr dazu bei, einen schnellen Zugang zu den Daten zu liefern. In einem solchen Fall greift man auf eine Darstellung per Säulendiagramm zurück das man durch die Funktion `barplot` bekommt. Genau wie beim Tortendiagramm erstellt man zuerst eine Häufigkeitstabelle und gibt diese dann als das obligatorische Argument ein. Auch hier kann man wieder durch das optionale Argument `col` die einzelnen Säulen farblich gestalten. Um eine größere Übersichtlichkeit zu erzielen, kann man (bei nominal-, nicht ordinalskalierten Merkmalen) die Ausprägungen gemäß ihrer Häufigkeit sortieren, wobei es sich empfiehlt, eine absteigende Reihenfolge zu wählen. (Leichtere Erfassung der Informationen der am häufigsten auftretenden Merkmale) Statt des Vektors `x` der Häufigkeiten setzt man in diesem Fall `sort(x, decreasing=T)` ein. Beim Beispiel der Institutsfeier erhalten wir damit das ungeordnete Säulendiagramm (oben) und das geordnete (unten).

`barplot`

Bei ordinalskalierten Merkmalsausprägungen macht es (dank der auf ihnen definierten Ordnungsstruktur) Sinn, **kumulierte Häufigkeiten** zu betrachten. Sind die a_i , $1 \leq i \leq k$, die (aufsteigend) geordneten Merkmalsausprägungen, erhalten wir die i -te absolute

3 Univariate Analyse



bzw. relative kumulierte Häufigkeit durch

$$\sum_{j=1}^i n_j \quad \text{bzw.} \quad \sum_{j=1}^i f_j.$$

Für ein ordinalskaliertes Merkmal besitzt die Häufigkeitstabelle im Allgemeinen eine weitere Spalte, die die kumulierten relativen Häufigkeiten enthält. Letztere erhält man in R durch Anwendung der Funktion `cumsum` (Abkürzung für: cumulated sum) auf die relativen Häufigkeiten. `cumsum`

Beispiel 3.4. Im Sommersemester 2004 haben 15 Studenten an der Klausur zur Statistik I teilgenommen. Dabei ergaben sich folgende Noten:

sehr gut	gut	befriedigend	ausreichend	mangelhaft
2	6	4	2	1

Diese Werte geben wir als Vektor ein und benennen die einzelnen Komponenten (wie im Beispiel mit den Getränken), bilden die relativen Häufigkeiten durch Division durch die Anzahl der Noten und wenden darauf die Funktion `cumsum` an:

```
> noten<-c(2,6,4,2,1)
> names(noten)<-c(„sehr gut“,„gut“,„befriedigend“,„ausreichend“,„mangelhaft“)
> noten
```

```

      sehr gut   gut   befriedigend   ausreichend   mangelhaft
            2     6             4             2             1
> Noten<-round(noten/sum(noten),2)
> Noten
      sehr gut   gut   befriedigend   ausreichend   mangelhaft
            0.13  0.40             0.27             0.13             0.07
> cumsum(Noten)
      sehrgut   gut   befriedigend   ausreichend   mangelhaft
            0.13  0.53             0.80             0.93             1.00

```

◇

3.1.2 Darstellung quantitativer Merkmale

Da quantitative Merkmale Zahlen als Ausprägungen besitzen und wir auf diese viele (Rechen-)Operationen anwenden können, gibt es viel mehr Möglichkeiten der Darstellung, wovon ein paar hier vorgestellt werden sollen. Bei diskreten Merkmalen können wir zunächst einmal dieselben Methoden anwenden wie bei ordinalskalierten Merkmalen (qualitativer Art), wohingegen stetige Merkmale eine andere Vorgehensweise verlangen. Diskrete Merkmale sind solche, die höchstens abzählbar viele Ausprägungen besitzen, doch in der Praxis werden nur solche Merkmale als diskret aufgefasst, die wenige Ausprägungen haben. Das Merkmal „Anzahl der Lebensjahre“ (sprich: Alter in Jahren) ist offensichtlich diskret, wird aber als stetig aufgefasst, da es viele Ausprägungen besitzt. Ausgangspunkt der Analyse quantitativer Merkmale ist die **Urliste**, die die Merkmalsausprägungen der einzelnen Merkmalsträger (in der Reihenfolge der Erhebung) auflistet. Bei einer Erhebung eines Merkmals bei n Merkmalsträgern erhalten wir also die Urliste

$$x_1, x_2, \dots, x_n,$$

wobei x_i die Ausprägung beim i -ten Merkmalsträger bezeichnet, $1 \leq i \leq n$.

Wir wollen uns zuerst mit diskreten Merkmalen beschäftigen. Im Allgemeinen ist die Urliste x_1, \dots, x_n recht unübersichtlich, weshalb wir den **geordneten Datensatz**

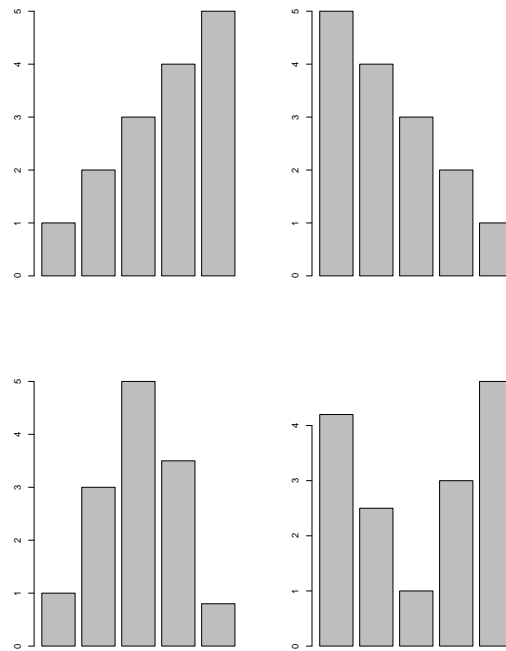
$$x_{(1)}, \dots, x_{(n)} \quad \text{mit} \quad x_{(1)} \leq \dots \leq x_{(n)}$$

bilden, was wir in R mit der Funktion **sort** erreichen. Wie bei qualitativen Merkmalen bestimmen wir die relativen und die kumulierten relativen Häufigkeiten und stellen sie in einer Tabelle zusammen. Ebenso können wir ein Säulendiagramm erzeugen, dürfen hier jedoch nicht mehr wie bei nominalskalierten Merkmalen die Ausprägungen nach ihrer Häufigkeit sortieren, da die Ausprägungen untereinander geordnet sind. Dadurch ergeben sich bei den Säulendiagrammen sehr unterschiedliche Konstellationen. Einige dieser Muster tragen spezielle Namen.

sort

Beim Merkmal in der Abbildung links oben werden die relativen Häufigkeiten mit wachsendem x immer größer. Große Ausprägungen treten also viel häufiger auf als kleine. Man spricht von einer **rechtssteilen** oder auch **linksschiefen** Verteilung. Beim

3 Univariate Analyse



Merkmal in der Abbildung oben rechts verhält es sich genau umgekehrt: Kleine Ausprägungen treten viel häufiger auf als große, und man spricht von einer **linkssteilen** oder **rechtsschiefen** Verteilung. Das Diagramm unten links in der Abbildung zeigt ein Merkmal, das sich beinahe symmetrisch zum Zentrum der Verteilung verhält, wobei die Merkmalsausprägungen beim Zentrum deutlich wahrscheinlicher sind als die am Rand. Diesen Fall nennt man eine **symmetrische** Verteilung. Im vierten Diagramm ist eine **zweigipflige** oder auch **bimodale** Verteilung dargestellt. Bei einer solchen Verteilung gibt es zwei Bereiche mit Ausprägungen hoher Wahrscheinlichkeit. Solche klaren Bilder wie hier ergeben sich in der Praxis - gerade bei kleineren Stichprobenumfängen - eher selten. Im Allgemeinen liefern die Säulendiagramme auch nur einen Eindruck der eigentlichen Verteilung des Merkmals in der Grundgesamtheit, der insbesondere bei kleinen Stichprobenumfängen verzerrt sein kann. Jedoch lassen sich Tendenzen häufig doch ganz gut ablesen.

Auch bei quantitativen Merkmalen können wir die kumulierten relativen Wahrscheinlichkeiten bestimmen, in diesem Fall sogar für jede reelle Zahl. Wir stellen uns unsere Stichprobe x_1, \dots, x_n als Realisation einer Zufallsgrößen $X = (X_1, \dots, X_n)$ vor, wobei die Komponenten X_i , $1 \leq i \leq n$, unabhängig seien und eine identische, uns unbekanntes Verteilung besitzen. Für ein beliebiges $t \in \mathbb{R}$ zählen wir, wie viele der Komponenten

kleiner gleich t sind, also:

$$F_n(t) := \frac{1}{n} \sum_{i=1}^n \mathbf{1}_{(-\infty, t]}(x_i).$$

Diese Funktion ist die **empirische Verteilungsfunktion**, die fast sicher gleichmäßig gegen die (unbekannte) Verteilungsfunktion der X_i , $1 \leq i \leq n$, konvergiert (s. Satz von Glivenko/Cantelli). Die empirische Verteilungsfunktion ist eine Treppenfunktion.

Die empirische Verteilungsfunktion können wir auch bei stetigen Merkmalen bestimmen, müssen aber insgesamt eine etwas andere Vorgehensweise zur Darstellung von Datensätzen wählen, wobei (wie schon erwähnt) in der Praxis auch diskrete Merkmale mit vielen Ausprägungen als stetig aufgefasst werden. Auch im stetigen Fall bildet man zunächst den geordneten Datensatz und unterteilt diesen dann in mehrere Klassen, bei denen es sich im Normalfall um links offene und rechts abgeschlossene Intervalle handelt. Dabei ist die Klasse mit den kleinsten Werten beidseitig abgeschlossen. Manchmal liegt es durch die Art der Werte nahe, eine andere Klassenstruktur zu wählen, wovon es sich dann in den meisten Fällen um rechts offene und links abgeschlossene Intervalle handelt. (Das mit den größten Werten ist dann beidseitig abgeschlossen). Wichtig ist, dass kein Wert in zwei Klassen liegen kann.

Für die so entstandenen Klassen kann man wieder Häufigkeitstabellen erstellen. Anstelle von Säulendiagrammen erzeugt man hier jedoch **Histogramme**. Bei einem Histogramm trägt man in einem rechtwinkligen Koordinatensystem, bei dem die Klassen auf der x-Achse liegen, über jeder Klasse ein Rechteck ab, dessen **Fläche** (und nicht die Höhe) gleich der relativen Häufigkeit ist. Die Höhe eines solchen Rechtecks ergibt sich also gerade aus dem Quotienten aus der relativen Häufigkeit und der Klassenbreite. Zu jeder reellen Zahl können wir somit eine Klasse und die Höhe des zugehörigen Rechtecks zuordnen. Die Funktion, die jeder Zahl die entsprechende Höhe zuordnet, heißt **empirische Dichtefunktion** \hat{f} und ist eine Treppenfunktion.

Da die Höhe der Rechtecke von der Klassenbreite abhängt, kann man nicht wie bei einem Säulendiagramm einfach die Höhen vergleichen, um die relativen Häufigkeiten zu vergleichen, es sei denn, jede Klasse weist die gleiche Breite auf. Bei solchen **äquidistanten** Klassen kann man wie bei den Säulendiagrammen auftretende Muster als schiefe / symmetrische / bimodale Verteilungen interpretieren.

Die Gestalt, der Informationsgehalt und die Übersichtlichkeit eines Histogramms hängen maßgeblich von der Gestalt und der Anzahl der Klassen ab. Für den Betrachter ist es wesentlich einfacher, Informationen aus einem Histogramm abzulesen, wenn äquidistante Klassen dargestellt sind. Sind dies allerdings sehr viele ist die Darstellung unübersichtlich und schwer zu lesen. Bei sehr wenigen Klassen wird die Gesamtstruktur hingegen sehr grob, und es gehen viele Informationen verloren. Außerdem ist es sinnvoll, runde Zahlen als Klassengrenzen zu wählen (oder dies zumindest bei der untersten Klasse zu tun), da der Betrachter solche schneller wahrnimmt und als „natürliche“ Einteilungen kennt.

Ein Problem tritt auf, wenn man ein Merkmal betrachtet, dessen Ausprägungen beliebig groß (beliebig klein) werden können, da man dann für die oberste (unterste) Klasse keine Obergrenze (Untergrenze) wählen kann. Nähme man ∞ ($-\infty$), hätte das zugehörige

Rechteck im Histogramm die Höhe 0. Für jeden anderen Wert als Grenze läge aber eine Verfälschung des Ergebnisses vor, da Daten, die außerhalb dieser Grenze lägen, missachtet würden. In einem solchen Fall kann also kein Histogramm gezeichnet werden. (Ein Beispiel wäre die Untersuchung von Jahreseinkommen...)

Es sei eine Erhebung zu einem Merkmal gegeben, bei der die Merkmalsausprägungen der Urliste zwischen den reellen Werten $a, b, a < b$, liegen. Wir bilden nun k Klassen, $k \in \mathbb{N}$, mit den (aufsteigenden) Grenzen x_0^*, \dots, x_k^* , wobei $x_0^* \leq a$ und $x_k^* \geq b$ gilt. Die i -te Klasse liegt also zwischen den Grenzen x_{i-1}^* und x_i^* . Der nächste Schritt ist, die absoluten Häufigkeiten der Klassen zu ermitteln. In R benutzen wir dazu die Funktion `cut`, mit der wir unseren Datensatz in die gewünschten Klassen einteilen können, und mit der Funktion `table`. Die Funktion `cut` verlangt zwei Argumente: den Datensatz `data` (also einen Vektor) und eine Anweisung (`breaks`), wo die Grenzen zu setzen sind, was in Form eines Vektors eingegeben wird. Möchte man zum Beispiel die Punkte 20,25,30,35,40 als Klassengrenzen setzen, gibt man Folgendes ein:

```
> cut(data,breaks=c(20,25,30,35,40))
```

Das Programm wählt automatisch rechts abgeschlossene Klassen. Sollen sie jedoch rechts offen und links abgeschlossen sein, müssen wir das optionale Argument `right` auf den Wert `FALSE` setzen. Mit dem Argument `include.lowest` können wir zusätzlich festlegen, ob die erste Klasse links abgeschlossen ist, wenn `right=TRUE` gilt, oder ob die oberste Klasse rechts abgeschlossen ist im Fall `right=FALSE`. Die Funktion `cut` liefert zu dem Datensatz `data` als Ergebnis einen genauso langen Vektor, der angibt, in welcher Klasse die entsprechende Komponente des Datensatzes `data` liegt.

Beispiel 3.5. Im Kindergarten werden einmal im Jahr die Kinder untersucht. Dabei werden ihre Reaktionen getestet, ihre Seh- und Hörfähigkeiten sowie die Sprachentwicklung überprüft und schließlich Körpergröße und -gewicht gemessen. Bei der diesjährigen Untersuchung wurden in der Schmetterlingsgruppe folgende Körpergrößen gemessen:

```
> größe<-c(103,105,106,98,87,104,116,104,112,96,101,109,99,113,107)
> klass.größe<-cut(größe,c(85,90,95,100,105,110,115,120),include.lowest=TRUE)
> klass.größe
 [1] (100,105] (100,105] (105,110] (95,100] [85,90] (100,105] (115,120]
 [8] (100,105] (110,115] (95,100] (100,105] (105,110] (95,100] (110,115]
[15] (105,110]
Levels: [85,90] (90,95] (95,100] (100,105] (105,110] (110,115] (115,120]
```

Wie schon zuvor kann nun mit der Funktion `table` die absoluten Häufigkeiten auflisten:

```
> abs.h.größe<-table(klass.größe)
> abs.h.größe
klass.größe
```

[85, 90]	(90, 95]	(95, 100]	(100, 105]	(105, 110]	(110, 115]	(115, 120]
1	0	3	5	3	2	1

◇

In diesem Beispiel haben wir äquidistante Klassengrenzen gewählt und sie als Vektor eingegeben. Mit der Funktion `seq` hätten wir uns ein wenig Arbeit sparen können, da es sich hier um äquidistante Klassengrenzen handelt. Im obigen Beispiel hätten wir also anstelle des Vektors `c(85, 90, 95, 100, 105, 110, 115, 120)` den Befehl

```
seq(85, 120, 5)
```

eingeben können, der den gleichen Vektor erzeugt hätte. Diese Funktion erweist sich besonders bei einer Einteilung in viele Klassen als nützlich.

Ein Histogramm erhalten wir nun mit der Funktion `hist`, die ähnliche Argumente fordert wie die Funktion `cut`. Das erste Argument ist der Datensatz, das zweite der Vektor der Klassengrenzen, dann kommen die Argumente `right` und `include.lowest`, sowie das Argument `freq` (Abkürzung für: frequency). Setzen wir dieses Argument auf `TRUE`, so werden die absoluten Häufigkeiten auf der Ordinate abgetragen, setzen wir es auf `FALSE`, entsprechen die Flächen der Rechtecke den relativen Häufigkeiten. Haben wir in die Funktion `hist` den Datensatz `data` eingetragen, erscheint `Histogram of data` als Überschrift über der Grafik. Mit dem Argument `main` können wir eine andere Überschrift als Zeichenkette eingeben bzw. keine Überschrift über das Diagramm setzen mit der leeren Zeichenkette “ “.

Für diskrete Merkmale haben wir bereits besprochen, wie man die empirische Verteilungsfunktion erhält. Diese wollen wir auch für stetige Merkmale bestimmen. Liegt die Urliste vor, erhalten wir die empirische Verteilungsfunktion wie im diskreten Fall. Liegen die Daten allerdings nicht als Urliste, sondern in klassierter Form vor, berechnet man zunächst die empirische Dichtefunktion \hat{f} . Mit Hilfe dieser Funktion, die jeder reellen Zahl x die Höhe des entsprechenden Rechtecks im Histogramm zuordnet, können wir nun die **approximierende empirische** oder auch **stetige empirische Verteilungsfunktion** F_n^* bestimmen. Diese ordnet jedem Wert $x \in \mathbb{R}$ den Wert der Fläche unter der empirischen Dichtefunktion bis zur Stelle x zu. Dies bedeutet nichts anderes, als dass man davon ausgeht, dass auf den Klassen jeweils eine Gleichverteilung gegeben ist. Es gilt also:

$$F_n^*(x) = \begin{cases} 0, & \text{für } x \leq x_0^*, \\ \sum_{j=1}^{i-1} f_j + (x - x_{i-1}^*)\hat{f}(x), & \text{für } x_{i-1}^* < x \leq x_i^*, \quad i = 1, \dots, k, \\ 1, & \text{für } x \geq x_k^*. \end{cases}$$

Die Funktion `hist` erwartet als Argument die Urliste, doch oft liegen die Daten nur in Form einer Häufigkeitstabelle mit klassierten Daten vor. Mit Hilfe der Funktion `rep` (Abkürzung für: replicate) können wir eine Urliste „fälschen“. Diese Funktion besitzt zwei Argumente. Das erste ist ein Vektor $x = (x_1, \dots, x_k)$ der Länge k mit Zahlen als Einträgen, der wiederholt werden soll. Wie oft dies geschehen soll, gibt das zweite

Argument an. Es kann eine natürliche Zahl n sein, was bewirkt, dass durch die n -fache Aneinanderreihung von x ein neuer Vektor erzeugt wird. Wir können auch einen Vektor $y = (y_1, \dots, y_k)$ einsetzen, der bewirkt, dass ein Vektor erzeugt wird, indem y_1 -mal x_1 , dann y_2 -mal x_2 usw. und schließlich y_k -mal x_k eingetragen wird. Die gefälschte Urliste erhalten wir dadurch, dass wir aus jeder Klasse einen Wert (z.B. die untere Klassengrenze oder die Klassenmitte) auswählen, daraus einen Vektor bilden und ihn als erstes Argument in `rep` einsetzen, wobei das zweite Argument dann ein Vektor mit den zugehörigen absoluten Häufigkeiten ist.

Beispiel 3.6. Wir können in die Funktion `rep` als Argumente eine Zahl und eine Zahl, einen Vektor und eine Zahl sowie zwei gleich lange Vektoren eingeben. Drei einfache Beispiele verdeutlichen die Wirkungsweise der Funktion:

```
> rep(4,5)
[1] 4 4 4 4 4

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

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

◇

3.2 Beschreibung univariater Datensätze

Im letzten Abschnitt haben wir Merkmale dargestellt, um die zu Grunde liegende Verteilung und ihre Eigenschaften zu verdeutlichen bzw. herauszufinden. Wir können eine Verteilung und ihre Eigenschaften aber auch beschreiben statt sie darzustellen und benötigen dazu gewisse Maßzahlen. Man untersucht in diesem Zusammenhang, welches Merkmal bzw. welche Ausprägung am häufigsten auftritt, wo das Zentrum der Verteilung liegt und wie sehr die Beobachtungen um das Zentrum streuen; man interessiert sich also für Mittelwerte, Varianzen bzw. ähnliche Maßzahlen und das Merkmal mit der größten relativen Häufigkeit.

Die Merkmalsausprägung, die am häufigsten auftritt, nennt man **Modus**, der aber nicht eindeutig sein muss, da mehrere Ausprägungen die gleiche (höchste) relative Häufigkeit aufweisen können und da bei klassierten Daten alle Werte der entsprechenden Klasse im Modus enthalten sind.

3.2.1 Median und Mittelwert

Der **Median** und der **Mittelwert** sind Maßzahlen für die Lage einer Verteilung und beschreiben, wo sich das sogenannte Zentrum der Verteilung befindet. Neben dem üblichen Median und arithmetischen Mittel greifen wir auch auf **getrimmte** und **gewichtete** Varianten zurück, bei denen man am Rande der Beobachtung liegende Werte nicht

berücksichtigt bzw. jedem Wert ein bestimmtes Gewicht und somit individuellen Einfluss verleiht. Mit Hilfe dieser Methoden kann man eine Verfälschung durch Ausreißer vermindern bzw. vermeiden.

Da wir diese Maßzahlen offensichtlich nicht für nominalskalierte Merkmale bilden können, befassen wir uns im Folgenden nur mit Merkmalen, die mindestens ordinalskaliert sind.

Den Median wollen wir mit $x_{0.5}$ bezeichnen. Bei metrischen Merkmalen und einer Stichprobe der Länge n erhalten wir den Median durch

$$x_{0.5} = \begin{cases} x_{(\frac{n+1}{2})}, & \text{falls } n \text{ ungerade,} \\ \frac{x_{(n/2)} + x_{(1+n/2)}}{2}, & \text{falls } n \text{ gerade,} \end{cases}$$

wobei $x_{(1)}, \dots, x_{(n)}$ die aufsteigend geordnete Stichprobe bezeichnet.

Bei einem ordinalskalierten Merkmal mit geordneter Stichprobe $x_{(1)}, \dots, x_{(n)}$ ist der Median die Ausprägung, bei der mindestens 50% der Beobachtungen größer gleich und mindestens 50% der Beobachtungen kleiner gleich sind. In diesem Fall (oder auch bei einem diskreten metrischen Merkmal) kann man den Median direkt aus der Häufigkeitstabelle mit den kumulierten relativen Häufigkeiten ablesen bzw. auch aus der empirischen Verteilungsfunktion bestimmen. In R erhält man den Median durch Eingabe von:

```
> median(data) median
```

Fehlen in dem Datensatz Beobachtungen, setzt man zusätzlich das Argument `na.rm` auf `TRUE` (Abk. für: not available bzw. remove), da sonst als Ausgabe lediglich `NA` erscheint. Gleiches gilt bei anderen Berechnungen mit Datensätzen, in denen Antworten fehlen. `na.rm`

Bei stetigen Merkmalen (bzw. bei diskreten mit sehr vielen Ausprägungen) kann man sowohl aus der Urliste als auch aus den klassierten Daten den Median bestimmen. Bei letzterer Methode gehen allerdings durch die Klassenbildung Informationen verloren, so dass man i.A. nicht den exakten Wert für den Median erhält. Zudem hängt in diesem Fall der Wert des Medians von der Wahl der Klassen ab.

Eine andere Maßzahl für die Lage der Verteilung ist der Mittelwert. Das einfachste Beispiel ist das arithmetische Mittel. Liegt die Urliste vor, kann man das arithmetische Mittel auf gewohnte Weise berechnen. Bei klassierten Daten muss man jedoch zunächst die Klassenmitten bestimmen und multipliziert diese dann mit den relativen Häufigkeiten. Die Summe der so entstehenden Werte entspricht dem Mittel. (Bei diskreten Werten kann man das Mittel ebenfalls durch Summation über alle Produkte aus Merkmalsausprägung und zugehöriger relativer Häufigkeit berechnen.) In R erhält man durch die Eingabe `mean(data)` den Mittelwert, wobei man auch hier im Fall fehlender Werte `na.rm` benutzt. `mean`

Während der Median durch einen Ausreißer kaum beeinflusst wird, ist der Mittelwert sehr **ausreißerempfindlich**. Größen, die nicht ausreißerempfindlich sind, bezeichnen wir als **robust** (also auch den Median).

Beispiel 3.7. Zwei Physiker führen Messungen zum selben Aufbau durch und erhalten die geordneten Messreihen M1 und M2 mit

M1 1 1 1 1.5 1.5 2 2 2
 M2 1 1 1 1.5 1.5 2 2 8

Messreihe M2 enthält offensichtlich einen Ausreißer. Bei beiden Messreihen liegt der Median bei 1.5, als Mittelwerte ergeben sich jedoch 1.5 bei M1 und 2.25 bei M2.

◇

In Abhängigkeit von der Verteilung ergibt sich folgendes Verhältnis zwischen Mittelwert \bar{x} und Median $x_{0.5}$:

rechtsschief $\bar{x} > x_{0.5}$
 symmetrisch $\bar{x} \approx x_{0.5}$
 linksschief $\bar{x} < x_{0.5}$

Wir können den Mittelwert durch **Trimmen** weniger ausreißerempfindlich gestalten. Dabei wird vor der Bildung des Mittelwerts von beiden Rändern der geordneten Stichprobe ein vorher festgelegter Anteil der Beobachtungen entfernt (Der Median ist ein Spezialfall eines getrimmten Mittelwertes.), so dass Ausreißer nicht mehr beachtet werden. Ein getrimmter Mittelwert ist daher eine robuste Größe.

Zu jedem $\alpha \in [0, \frac{1}{2}]$ können wir den getrimmten Mittelwert durch

$$\bar{x}_\alpha = \frac{1}{n - 2 \lfloor n\alpha \rfloor} \sum_{i=1+\lfloor n\alpha \rfloor}^{n-\lfloor n\alpha \rfloor} x_{(i)}$$

berechnen, wobei $\lfloor \cdot \rfloor$ die Gaußklammer bezeichne. Diesen α -getrimmten Mittelwert erhält man in R durch die Eingabe von

`mean(data, trim= α)`

Der getrimmte Mittelwert ist ein Spezialfall des gewichteten Mittelwerts. In diesem Fall ordnen wir jedem Stichprobenwert eine Zahl aus dem Einheitsintervall, das Gewicht, zu, womit wir jedem einzelnen Wert gezielt Einfluss auf den Mittelwert entziehen bzw. zuordnen können. Der gewichtete Mittelwert \bar{x}_w ist definiert durch

$$\bar{x}_w := \sum_{i=1}^n w_i x_i,$$

wobei $\sum_{i=1}^n w_i = 1$ gelten muss.

3.2.2 Quantile

Wir wollen bei gegebener geordneter Stichprobe $(x_{(1)}, \dots, x_{(n)})$ das p -Quantil x_p , $p \in (0, 1)$, bestimmen. Dann muss gelten: Mindestens $p \cdot 100$ Prozent der Beobachtungen sind

3 Univariate Analyse

$\leq x_p$ und mindestens $(1-p)100$ Prozent $\geq x_p$. Um x_p zu finden, berechnen wir $k = np$. Ist k eine natürliche Zahl, gilt

$$x_p = \frac{x_{(k)} + x_{(k+1)}}{2},$$

ist k jedoch keine natürliche Zahl, gilt

$$x_p = x_{(\lfloor k \rfloor + 1)}.$$

Bei klassierten Daten können wir Quantile approximativ mit Hilfe der empirischen Verteilungsfunktion bestimmen:

$$x_p = x_{i-1}^* + \frac{p - F_n^*(x_{i-1}^*)}{f_i} (x_i^* - x_{i-1}^*),$$

falls $F_n^*(x_{i-1}^*) \leq p < F_n^*(x_i^*)$, wobei x_0^*, \dots, x_n^* wie zuvor die Klassengrenzen bezeichnen sollen.

Quantile erzeugt man in R mit der Funktion `quantile`:

`quantile`

`quantile(x, probs, type)`

in die man neben der Stichprobe `x` einen Vektor mit den Wahrscheinlichkeiten einsetzt, zu denen man die Quantile bestimmt haben möchte. Das Argument `type` gibt an, welchen Quantiltyp wir bestimmt haben wollen. Standardmäßig steht dieses Argument auf 7, die oben beschriebene Vorgehensweise entspricht jedoch Typ 2. Diese ist etwas intuitiver, die in R eingestellte Methode des Typs 7 kann natürlich genauso verwendet werden. Näheres zu den einzelnen Methoden findet man im Hilfetext der Funktion `quantile`. Die Quantile $x_{0.25}$ und $x_{0.75}$ werden unteres (bzw. erstes) und oberes (bzw. drittes) Quartil genannt und werden uns später nochmal begegnen.

`type`

3.2.3 Maßzahlen für die Variabilität

Um die Streuung der Beobachtungen ums Zentrum der Verteilung einstufen zu können, brauchen wir auch dafür Maßzahlen. Man kann zum Beispiel die mittlere absolute Abweichung $\frac{1}{n} \sum_{i=1}^n |x_i - \bar{x}|$ betrachten, oder auch die mittlere quadratische Abweichung

$$d^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2,$$

die größeren Abweichungen mehr Gewicht verleiht. Man kann d^2 auch so darstellen:

$$d^2 = \bar{x^2} - \bar{x}^2 \quad \text{mit} \quad \bar{x^2} = \frac{1}{n} \sum_{i=1}^n x_i^2,$$

3 Univariate Analyse

verwendet man jedoch diese Formel in R, erhält man (wegen Rundungsfehlern, insbesondere bei großen x_i) teilweise sehr ungenaue Ergebnisse. In der Regel betrachtet man aber statt d^2 die Stichprobenvarianz

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2,$$

die, falls die x_i , $i = 1 \dots n$, Realisierungen unabhängiger, identisch verteilter Zufallsgrößen X_i (mit unbekannter Verteilung) sind, ein erwartungstreuer Schätzer für die Varianz $\text{Var}(X_i)$ ist. Da in der Literatur häufig auch die mittlere quadratische Abweichung als Stichprobenvarianz bezeichnet wird, sollte man sich immer vergewissern, welche Größe (d^2 oder s^2) damit gemeint ist. Mit dem Befehl

`var(x)`

`var`

gelangen wir zur Stichprobenvarianz s^2 der Stichprobe \mathbf{x} . Zwischen d^2 und s^2 besteht offensichtlich der lineare Zusammenhang

$$s^2 = \frac{n}{n-1} d^2.$$

Wir wollen hier die Quadratwurzel aus der Stichprobenvarianz als Standardabweichung bezeichnen.

Bei Vorliegen einer Häufigkeitstabelle (statt einer Urliste) mit diskreten Merkmalen können wir mit Hilfe der absoluten Häufigkeiten eine künstliche Urliste generieren und aus dieser mit `var` die Stichprobenvarianz bestimmen. Sind nur die relativen Häufigkeiten gegeben, können wir damit keine Urliste erstellen, da wir nicht die Stichprobenlänge kennen! Aus solchen Daten können wir aber zumindest \bar{x} und $\overline{x^2}$ berechnen und sie dann in die (ungenaue) Formel

$$d^2 = \overline{x^2} - \bar{x}^2$$

einsetzen. Wir berechnen \bar{x} und $\overline{x^2}$ durch

$$\begin{aligned} \bar{x} &= \sum_{i=1}^k a_i f_i \quad (= \frac{1}{n} \sum_{i=1}^k a_i n_i), \\ \overline{x^2} &= \sum_{i=1}^k a_i^2 f_i \quad (= \frac{1}{n} \sum_{i=1}^k a_i^2 n_i). \end{aligned}$$

Bei einem stetigen Merkmal mit klassierten Daten muss man bei Vorliegen einer Häufigkeitstabelle zunächst die Klassenmitten m_i bestimmen und geht dann genauso vor, wie oben beschrieben, allerdings mit m_i statt a_i . Die Klassenmitten sind hier wichtig, da wir implizit von einer Gleichverteilung der Merkmalsausprägungen innerhalb der Klassen ausgehen.

Eine weitere Maßzahl für die Streuung ist die Spannweite (engl.: range) R , die als Differenz aus der größten und kleinsten Beobachtung definiert ist:

$$R(x) = x_{(n)} - x_{(1)}.$$

Die Spannweite hat als Maßzahl für die Streuung den Nachteil, dass sie nur von den Extremen abhängt und folglich sehr ausreißerempfindlich ist.

Wir erhalten sie in R durch eine Kombination von zwei Befehlen: `range` und `diff.range`. `range` hat als Ergebnis einen Zweiertupel, bestehend aus dem Minimum und dem Maximum des in `range` eingesetzten Vektors. Die Funktion `diff` bildet die Differenzen aufeinander folgender Komponenten eines Vektors:

```
> diff(c(1,1,2,3,5,8))
[1] 0 1 1 2 3
```

die Spannweite (Differenz zwischen Maximum und Minimum eines Vektors) erhalten wir demnach durch:

```
> diff(range(x))
```

Eine andere, weniger ausreißerempfindliche Maßzahl ist der Interquartilsabstand (engl.: inter quartile range) IQR , der die Differenz zwischen dem dritten und dem ersten Quartil angibt:

$$IQR(x) = x_{0.75} - x_{0.25}.$$

In R müssen wir dazu erst die Quartile erzeugen und darauf dann die Funktion `diff` anwenden:

```
> diff(quantile(x,probs=c(0.25,0.75))).
```

3.2.4 Der Boxplot

Mit einem Boxplot kann man die Verteilung eines quantitativen Merkmals einfach und effizient veranschaulichen. Im Wesentlichen stellt der Boxplot fünf Maßzahlen der Verteilung dar. Diese fünf Zahlen nennt man auch die Fünf-Zahlen-Zusammenfassung der Verteilung. Es handelt sich bei ihnen um:

- das Minimum $x_{(1)}$,
- das unter Quartil $x_{0.25}$,
- den Median $x_{0.5}$,
- das obere Quartil $x_{0.75}$,
- das Maximum $x_{(n)}$.

Diese fünf Zahlen werden in einem Boxplot grafisch dargestellt. Vom unteren Quartil bis zum oberen Quartil wird eine Box gezeichnet, in der Median als Linie eingetragen

wird. Zwischen den Extremen und der Box wird eine Linie gezeichnet, die an den Extremen durch eine Querlinie begrenzt wird. Diese Querlinien nennt man die Zäune (engl.: whiskers). Durch die fünf Werte werden vier Bereiche abgegrenzt, in denen sich jeweils 25% der Beobachtungen befinden. Auf dieser Tatsache beruht auch die Interpretation der Grafik. Der Median dient als Maßzahl für die Lage des Zentrums der Verteilung, die Breite der Box, also der IQR, verdeutlicht die Streubreite der Verteilung. Die Lage der Medianlinie innerhalb der Box erlaubt dann Vermutungen darüber, ob eine symmetrische (Mitte), rechtsschiefe (beim unteren Quartil) oder linksschiefe (beim oberen Quartil) Verteilung vorliegt. Die Linien zu den Zäunen kennzeichnen den Bereich vom Minimum bis zum untersten Quartil bzw. vom oberen Quartil bis zum Maximum. Wenn aber die Stichprobe Ausreißer enthält, wenn also Werte sehr klein oder groß werden, kennzeichnet der Zaun nicht mehr das Minimum und das Maximum, sondern vielmehr den kleinsten und den größten Wert, der sich noch innerhalb des eineinhalbfachen IQR von der Box befindet. Weicht ein Punkt mehr als $1.5 \cdot \text{IQR}$ von beiden Quartilen ab, wird er außerhalb des Zaunes gezeichnet und mit einem o oder * (o.Ä.) gekennzeichnet.

Beispiel 3.8. Ein Professor macht in seinem Seminar immer mal wieder Scherze. Ein am Seminarstoff nicht besonders interessierter Student achtet aus Langeweile darauf, wie viele der Teilnehmer (aus welchem Grund auch immer) über die Scherze lachen. Er erhält:

1 5 8 9 9 9 10 15.

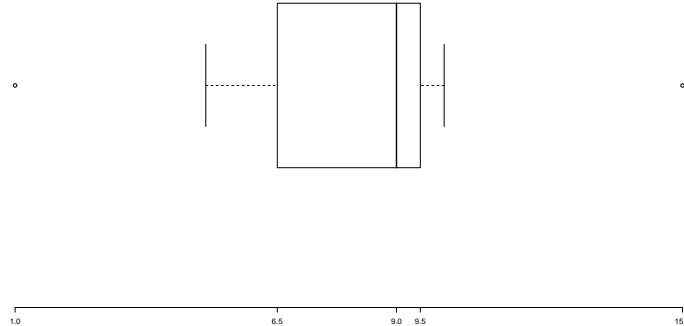
Minimum und Maximum sind also 1 und 15, der Median liegt bei 9, die beiden Quartile bei 6.5 und 9.5. Demnach beträgt der Interquartilsabstand $9.5 - 6.5 = 3$, der eineinhalbfache IQR als 4.5. Der Abstand des Minimums vom unteren Quartil beträgt $6.5 - 1 = 5.5$ und ist somit größer als 4.5. Daher wird das Minimum im Boxplot als Ausreißer gekennzeichnet, der Zaun wird nur bis zum zweitkleinsten Wert 5 gezeichnet, der nur 1.5 vom unteren Quartil entfernt ist. Das Maximum ist ebenfalls ein Ausreißer und wird als solcher markiert, da es ebenfalls um 5.5 vom oberen Quartil abweicht. Der obere Zaun wird daher bis zur zweitgrößten Zahl 10 gezeichnet, und man erhält den folgenden Boxplot:

◇

Ein Boxplot besitzt - ebenso wie ein Histogramm - Vor- und Nachteile. Man kann anhand der Medianlinie erkennen, ob die Verteilung symmetrisch ist (bzw. sein könnte), und Ausreißer werden auch klar gekennzeichnet. Man kann jedoch die Verteilungsart nicht genauer bestimmen. Es könnte beispielsweise eine mehrgipflige Verteilung vorliegen, was am Boxplot allerdings nicht zu erkennen wäre. Die wiederum käme durch ein Histogramm gut heraus, das auch Symmetrie und Ausreißer erkennen lässt (wenn vorhanden), wobei dieses jedoch stark von der Wahl der Klassengrenzen abhängt. Wählt man nur wenige Klassen, lässt sich auch aus einem Histogramm nicht sehr viel ablesen. Um die Vorteile beider Darstellungen nutzen zu können, setzt man beide in eine Grafik.

In R gehen wir folgendermaßen vor: Die Fünf-Zahlen-Zusammenfassung bekommen wir mit der Funktion `fivenum`, mit der Funktion `summary` erhalten wir neben diesen fünf

`fivenum`
`summary`



Zahlen auch das arithmetische Mittel. Dabei kann es sein, dass `fivenum` und `summary` leicht unterschiedliche Werte für die Quartile ausgeben, was auf unterschiedliche Berechnungsarten für die Quartile zurückzuführen ist. Im obigen Beispiel erhalten wir

```
> fivenum(c(1,5,8,9,9,9,10,15))
[1] 1.0 6.5 9.0 9.5 15.0
```

```
> summary(c(1,5,8,9,9,9,10,15))
  Min. 1st Qu. Median Mean 3rd Qu.  Max.
  1.00   7.25   9.00  8.25   9.25  15.00
```

Den Boxplot einer Stichprobe erzeugen wir mit Hilfe der Funktion `boxplot`. Dieser wird `boxplot` standardmäßig senkrecht gezeichnet und in eine Achsenbox eingetragen. Mit dem Argument `axes(=FALSE)` können wir diese vermeiden und mit `axis` nachträglich welche hinzufügen. Soll der Boxplot waagrecht gezeichnet werden, geben wir noch `horizontal=TRUE` `axis` in den `boxplot`-Befehl ein. `horizontal`

Beispiel 3.9. Im vorigen Beispiel ergibt also

```
> boxplot(c(1,5,8,9,9,9,10,15),horizontal=TRUE,axes=FALSE)
> axis(1,at=fivenum(c(1,5,8,9,9,9,10,15)))
```

einen waagerechten Boxplot mit der bei den fünf Zahlen beschrifteten x-Achse. Das erste Argument in `axis` gibt an, welche Achse gezeichnet werden soll, das zweite, wo diese beschriftet werden soll. Die x-Achse, also die untere Seite der Achsenbox, wird durch die 1 gekennzeichnet, die linke durch 2, die obere durch 3 und schließlich die rechte Seite durch 4.

◇

3 Univariate Analyse

Man kann auch mehrere Boxplots in einem Diagramm erstellen, indem man mehrere (nicht notwendig gleich lange) Datensätze in die Funktion `boxplot` einsetzt. Einen Boxplot und ein Histogramm in einer Abbildung erhält man mit dem Befehl `simple.hist.and.boxplot` aus dem Paket `UsingR`.

```
simple.hist.  
and.boxplot
```

4 Multivariate Datenanalyse

Mit Hilfe statistischer Verfahren kann man untersuchen, ob zwischen mehreren Merkmalen Abhängigkeiten bestehen. Wir wollen uns in diesem Abschnitt allerdings nur mit der Untersuchung von zwei Merkmalen befassen. (Bei mehreren Merkmalen werden andere Methoden notwendig, die in den Kapiteln über Testen und über die Regressions- oder Varianzanalyse zur Sprache kommen.)

Sind beide untersuchten Merkmale quantitativ, werden wir die Abhängigkeitsstruktur mit einer Maßzahl beschreiben. Ist jedoch mindestens eins der beiden Merkmale qualitativ, können wir alle Merkmalsträger gemäß ihrer Merkmalsausprägung in Gruppen einteilen.

4.1 Zusammenhang zwischen quantitativen und qualitativen Merkmalen

Wir unterteilen unsere Stichprobe gemäß den Ausprägungen des qualitativen Merkmals in Gruppen. Innerhalb dieser Gruppen können wir die üblichen Maßzahlen bilden und diese dann untereinander vergleichen, wodurch wir einen Einblick in die Verteilung des quantitativen Merkmals in Abhängigkeit von der Ausprägung des qualitativen Merkmals erhalten. Es liegen $c \in \mathbb{N}$ Gruppen vor, die n_1, \dots, n_c Elemente enthalten, wobei $n_1 + \dots + n_c = N$ gelten muss, wenn N Merkmalsträger befragt wurden. Das j -te Element in der i -ten Gruppe, $1 \leq i \leq c$, $1 \leq j \leq n_i$, bezeichnen wir mit y_{ij} . Für jede Gruppe bestimmen wir nun die Fünf-Zahlen-Zusammenfassung (oder mit `summary` auch die fünf Zahlen plus den Mittelwert) und zeichnen die Boxplots nebeneinander in ein Diagramm.

summary

Beispiel 4.1. Wir betrachten wieder den Datensatz `Weiterbildung` und interessieren uns dabei für die beiden Merkmale `Alter` und `Geschlecht`. Also unterteilen wir die Altersdaten gemäß der Ausprägung des Merkmals `Geschlecht` in zwei Gruppen:

```
> attach(weiterbildung)
> alter.w<-Alter[Geschlecht=="w"]
> alter.m<-Alter[Geschlecht=="m"]

> summary(alter.w)
  Min. 1stQu. Median  Mean 3rdQu.  Max.
23.00 25.00 28.00 27.77 29.00 38.00

> summary(alter.m)
  Min. 1stQu. Median  Mean 3rdQu.  Max.
23.00 26.75 29.50 30.50 34.00 38.00
```

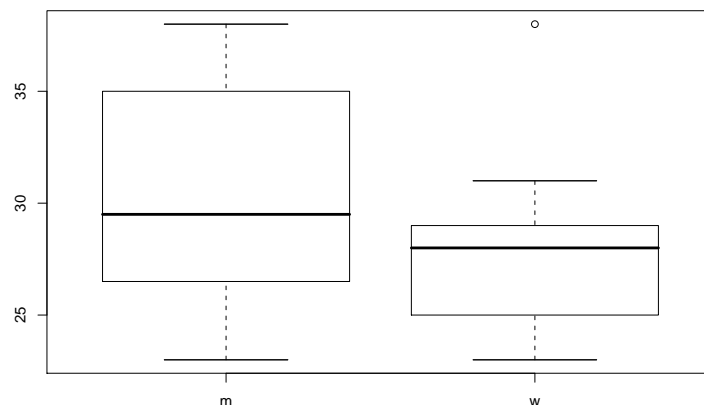
4 Multivariate Datenanalyse

```
> var(alter.w)
[1] 16.52564
> var(alter.m)
[1] 24.27273
```

Anhand dieser Zahlen können wir nun die Verteilung des Merkmals Alter in Abhängigkeit vom Geschlecht untersuchen. Es zeigt sich, dass die Extrema in beiden Gruppen gleich sind, sämtliche anderen Werte sind allerdings in der Gruppe der Männer höher. Bei den Frauen treten also mehr kleine Alterszahlen auf als bei den Männern, wo die Alterszahlen relativ gleichmäßig in dem ganzen Intervall gestreut zu sein scheinen. Am dritten Quantil der Daten der Frauen lässt sich ablesen, dass in dieser Gruppe 75% der Beobachtungen zwischen 23 und 29 liegen, während sich die restlichen über den Bereich von 30 bis 38 verteilen. Insgesamt ist die Streuung innerhalb der Gruppe der Frauen daher auch nicht so hoch wie bei den Männern, wie die Stichprobenvarianzen zeigen. Nun wollen wir noch die Boxplots der beiden Gruppen zeichnen:

```
> boxplot(alter.m,alter.w,names=c("m","w"))
```

was folgende Grafik ergibt:



◇

Liegen die Daten nur klassiert und in Form einer Häufigkeitstabelle vor, kann man Minimum und Maximum der Stichprobe nicht bestimmen und somit keinen Boxplot zeichnen, es sei denn, man verzichtet auf die Zäune und zeichnet lediglich die Box mit der Medianlinie. Für eine solche Grafik bestimmt man den Median M sowie das erste und dritte Quartil ($1.Qu$ und $3.Qu$) der klassierten Daten. Setzt man dann den Vektor

`c(1.Qu,1.Qu,M,3.Qu,3.Qu)` in die Funktion `boxplot` ein, erhält man die entsprechende Grafik. `boxplot`

Bei Vorliegen genau zweier Gruppen kann man statt der vergleichenden Boxplots zwei senkrechte Histogramme „Rücken an Rücken“ zeichnen, wie man es von einer Bevölkerungspyramide kennt, wozu wir die Funktion `histbackback` aus dem Paket `Hmisc` benutzen. `histbackback`

Im obigen Beispiel haben wir einige Schritte benötigt, um die beiden Boxplots zu erstellen. Mit den Funktionen `split`, `lapply` und `sapply` können wir das Vorgehen etwas abkürzen. `split` haben wir bereits kennengelernt. In diese Funktion setzen wir als erstes Argument den zu unterteilenden Datensatz und als zweites den Faktor ein, gemäß dessen die Unterteilung vorgenommen werden soll, also: `split`
`lapply`
`sapply`

```
split(weiterbildung,Geschlecht)
```

Die beiden Funktionen `lapply` und `sapply` werden durch

```
lapply(x,FUN)
sapply(x,FUN)
```

aufgerufen, wobei `x` eine Liste (also eine Datentabelle o.Ä.) und `FUN` eine Funktion ist, die auf `x` angewendet werden soll. `lapply` liefert dann eine Liste, deren i -ter Eintrag das Resultat enthält, das man bei Anwendung von `FUN` auf die i -te Komponente von `x` erhält. `sapply` gibt einen Vektor als Ergebnis, wenn `FUN` skalare Werte annimmt, und eine Matrix, wenn das Ergebnis von `FUN` ein Vektor ist. Dabei ist der i -te Eintrag des Vektors bzw. die i -te Zeile der Matrix, das Resultat von `FUN` angewendet auf die i -te Komponente von `x`.

Beispiel 4.2. Wenn wir also im Weiterbildungsdatensatz das Alter getrennt nach dem Geschlecht untersuchen und in einem Boxplot darstellen wollen, können wir folgendermaßen vorgehen: Die Aufspaltung erhalten wir mit

```
> attach(weiterbildung)
> split(Alter,Geschlecht)
$m
...

$w
...
```

die Stichprobenvarianz oder die Zusammenfassung bekommen wir durch Eingabe von

```
> lapply(split(Alter,Geschlecht),var)
```

```

$m
24.27273

$w
16.52564

> sapply(split(Alter,Geschlecht),var)
  m      w
24.27273 16.52564

> lapply(split(Alter,Geschlecht),summary)
$m
Min. 1st Qu. Median Mean 3rd Qu. Max.
...   ...     ...     ...   ...     ...

$w
Min. 1st Qu. Median Mean 3rd Qu. Max.
...   ...     ...     ...   ...     ...

> sapply(split(Alter,Geschlecht),summary)
      m      w
Min.   ...   ...
1st Qu. ...   ...
Median ...   ...
Mean   ...   ...
3rd Qu. ...   ...
Max.   ...   ...

```

Zu den Boxplots und den Histogrammen gelangen wir mit den Befehlen

```

> boxplot(split(Alter,Geschlecht),names=c("m","w"))
> histbackback(split(Alter,Geschlecht),brks=seq(20,40,5),names=c("m","w"))

```

wobei für die Funktion `histbackback` zunächst das Paket `Hmisc` geladen werden muss.

◇

4.2 Zusammenhang zwischen nominalskalierten Merkmalen

Wir betrachten nun zwei nominalskalierte Merkmale A und B mit den Ausprägungen a_1, \dots, a_I und b_1, \dots, b_J , $I, J \in \mathbb{N}$, und halten für jede Kombination von a_i und b_j die absolute Häufigkeit des gemeinsamen Auftretens mit Hilfe einer Kontingenztabelle fest, die im Allgemeinen folgenden Aufbau besitzt:

4 Multivariate Datenanalyse

	b_1	b_2	\dots	b_J	
a_1	n_{11}	n_{12}	\dots	n_{1J}	$n_{1\cdot}$
a_2	n_{21}	n_{22}	\dots	n_{2J}	$n_{2\cdot}$
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
a_I	n_{I1}	n_{I2}	\dots	n_{IJ}	$n_{I\cdot}$
	$n_{\cdot 1}$	$n_{\cdot 2}$	\dots	$n_{\cdot J}$	

wobei $n_{i\cdot} = \sum_{j=1}^J n_{ij}$ die absolute Häufigkeit des Merkmals a_i und entsprechend $n_{\cdot j}$ die absolute Häufigkeit des Merkmals b_j ist, die wir jeweils als Randhäufigkeiten bezeichnen. Bei Division durch die Gesamtzahl der Beobachtungen $\sum_{i,j} n_{ij}$ erhalten wir die relativen Häufigkeiten, bei Division durch die Randhäufigkeiten die bedingten relativen Häufigkeiten (im ersten Fall von b_j bedingt unter a_i , im zweiten umgekehrt):

$$h_{b_j|a_i} = \frac{n_{ij}}{n_{i\cdot}}$$

$$h_{a_i|b_j} = \frac{n_{ij}}{n_{\cdot j}}$$

Eine Kontingenztabelle der absoluten Häufigkeiten erhalten wir in R mit der Funktion `table`, in die wir einen Vektor, eine Tabelle o.Ä. einsetzen, die aus Faktoren bestehen (oder aus Werten, die als Faktoren interpretiert werden können). Liegen uns die Daten bereits als Matrix (mit den Häufigkeiten als Einträgen) vor, fassen wir diese als Kontingenztabelle auf. Zu einer Tabelle der relativen Häufigkeiten gelangen wir mit Hilfe von `prop.table`, wo wir als Argument eine Kontingenztabelle bzw. eine Matrix der absoluten Häufigkeiten einsetzen. Die bedingten relativen Häufigkeiten erhalten wir, indem wir in `prop.table` als zusätzliches Argument `margin` einsetzen. Bei `margin=1` bekommen wir die unter den a_i , $1 \leq i \leq I$, bedingten relativen Häufigkeiten, bei `margin=2` die unter b_j , $1 \leq j \leq J$, bedingten.

`table`

`prop.table`

`margin`

Beispiel 4.3. Es sei ein einfacher Datensatz aus zwei nominalskalierten Merkmalen gegeben:

```
versuch<-data.frame(Geschlecht=c("w","w","w","w","m","m","m","m"),
                    Antwort=c("j","j","n","j","n","n","j","j"))
```

Die entsprechenden Kontingenztabellen der absoluten, relativen und bedingten Häufigkeiten erhalten wir durch

```
table(versuch)
      Antwort
Geschlecht j  n
      m  2  2
      w  3  1

prop.table(table(versuch))
```

```

      Antwort
Geschlecht  j      n
      m 0.250 0.250
      w 0.375 0.125

```

```
prop.table(table(versuch),1)
```

```

      Antwort
Geschlecht  j      n
      m 0.50 0.50
      w 0.75 0.25

```

```
prop.table(table(versuch),2)
```

```

      Antwort
Geschlecht  j      n
      m 0.4000000 0.6666667
      w 0.6000000 0.3333333

```

Die zweitletzte Tabelle enthält dabei an der Position (i, j) die relative Häufigkeit von b_j bedingt unter a_i , die letzte Tabelle die von a_i bedingt unter b_j .

◇

Nun wollen wir diese in den Tabellen enthaltenen Daten auch grafisch darstellen, da sie dann wesentlich eingänglicher sind. Dazu gibt es mehrere Möglichkeiten: Wir können ein vergleichendes Säulendiagramm erstellen, wozu wir in `barplot` einfach die entsprechende Tabelle als Argument eingeben, und erhalten dann in einer Grafik mehrere Säulendiagramme nebeneinander. Dabei werden die Säulen aufeinander gezeichnet. Sollen sie wie üblich nebeneinander gezeichnet werden, setzt man das optionale Argument `beside` auf `TRUE`. Mit `legend=TRUE` wird eine passende Legende eingefügt. Wenn sich die Grafik zu unübersichtlich gestaltet, kann man dies zumindest zum Teil dadurch verbessern, dass man das erste Säulendiagramm der Größe der Säulen nach ordnet und in den übrigen Säulendiagrammen diese Reihenfolge beibehält, wozu man die Funktion `order` benutzen kann. Man geht dann z.B. folgendermaßen vor (für genauere Informationen siehe Hilfetext der Funktion)

`beside`
`legend`

`order`

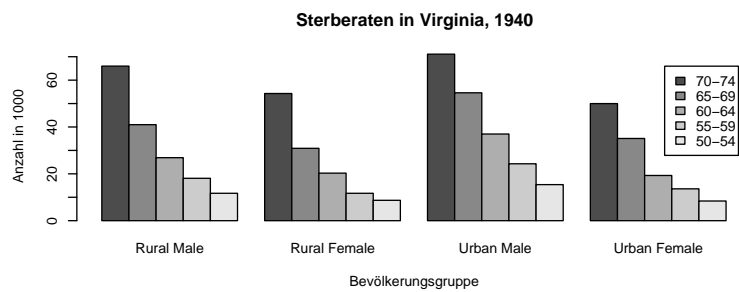
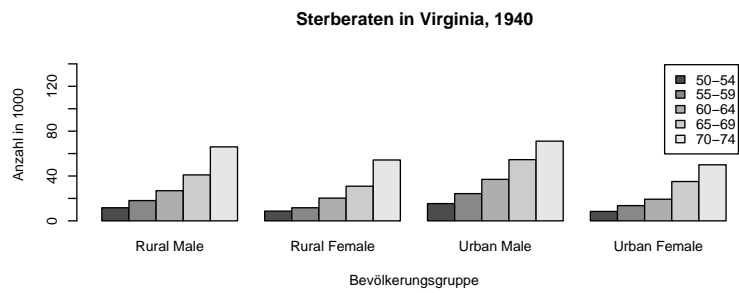
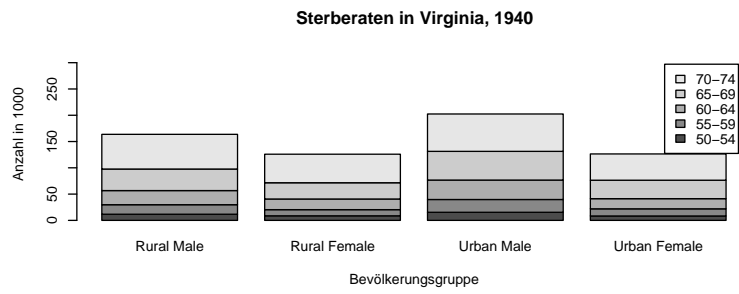
```
barplot(Tabelle[order(prop.table(Tabelle)[,1],decreasing=T),],beside=T)
```

und erhält eine Grafik, bei der das erste Säulendiagramm absteigend sortiert ist. Die Ergebnisse, die man mit dem (in sich schon sortierten Datensatz `VADeaths` erhält, zeigen die folgenden Grafiken. Dabei enthält der verwendete Datensatz die Sterberaten im US-Bundesstaat Virginia im Jahre 1940, aufgeteilt nach Geschlecht und Art der Wohngegend (Stadt oder Land). Für diese Grafiken wurden die folgenden Befehle benutzt:

```
> barplot(VADeaths,legend=T,main="Sterberaten in Virginia, 1940",
          ylim=c(0,320),ylab="Anzahl in 1000",xlab="Bevölkerungsgruppe")
```

```
> barplot(VADeaths,legend=T,main="Sterberaten in Virginia, 1940",
```

4 Multivariate Datenanalyse



```
ylim=c(0,150),ylab="Anzahl in 1000",
xlab="Bevölkerungsgruppe",beside=T)
```

```
> barplot(VADeaths[order(prop.table(VADeaths)[,1],decreasing=T),],
  beside=T,main="Sterberaten in Virginia, 1940",
  ylab="Anzahl in 1000",xlab="Bevölkerungsgruppe",legend=T)
```

Eine andere Möglichkeit der Darstellung ist mit Hilfe eines Mosaikplots. Bei einem Mosaikplot wird für jede Zelle einer Kontingenztabelle ein Rechteck gezeichnet, dessen Fläche proportional zu der in der Zelle eingetragenen absoluten Häufigkeit ist. Die Zeilen der Kontingenztabelle werden dabei zu einer Spalte im Mosaikplot, deren Breite proportional zur Zeilensumme in der Tabelle ist, so dass man die Größe der verschiedenen Gruppen miteinander vergleichen kann, ohne jedoch konkrete Werte ablesen zu können. Diese Darstellung erreichen wir, indem wir eine Matrix oder einen Array der absoluten Häufigkeiten in die Funktion `mosaicplot` einsetzen. Für den Datensatz `VADeaths` ergeben sich dann mit

mosaicplot

```
> mosaicplot(VADeaths)
> mosaicplot(t(VADeaths))
```

die folgenden Grafiken: Die Beschriftung wird hier automatisch vorgenommen, kann aber natürlich auch manuell angepasst werden. Setzt man 3- oder mehrdimensionale Arrays ein, werden die Zeilen und Spalten weiter unterteilt und die Grafik kann etwas unübersichtlich werden. Benutzt man bei Arrays mit drei oder mehr Dimensionen die Funktion `mosaic` aus dem Paket `vcd`, erhält man auch einen Mosaikplot des Datensatzes, der allerdings klarer strukturiert, daher übersichtlicher und folglich vorzuziehen ist.

mosaic
vcd

Eine letzte Darstellungsform, auf die wir hier eingehen wollen, ist die des Profildiagrammes. Auf der x-Achse werden dabei die Ausprägungen b_j , $1 \leq j \leq J$, des Merkmals B äquidistant abgetragen. Über jeder solchen Ausprägung b_j werden dann die bedingten relativen Häufigkeiten $h_{b_j|a_i}$, $1 \leq i \leq I$, eingetragen. Für jedes $1 \leq i \leq I$ werden die relativen Häufigkeiten durch Linien verbunden, d.h. auf einem Streckenzug liegen also $h_{b_1|a_i}$, $h_{b_2|a_i}$, \dots , $h_{b_J|a_i}$. Für `VADeaths` sieht das so aus: Diese Grafik erhalten wir mit der Funktion

```
interaction.plot(x.factor,trace.factor,response,xlab,trace.label,...)
```

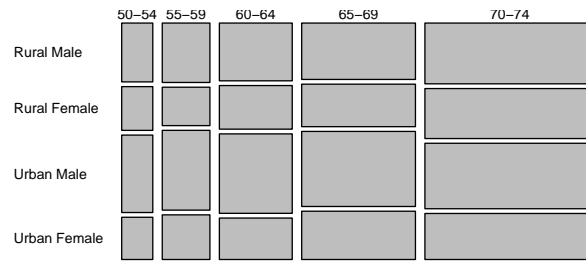
interaction.
plot

wobei wir unsere Daten als `response` einsetzen, dafür allerdings zuerst noch aufbereiten müssen, während der `x.factor` jeder Komponente unserer Daten die entsprechende Ausprägung b_j von B zuordnet und `trace.factor` die von A. Mit `xlab` und `trace.label` können wir dann die Achsen beschriften.

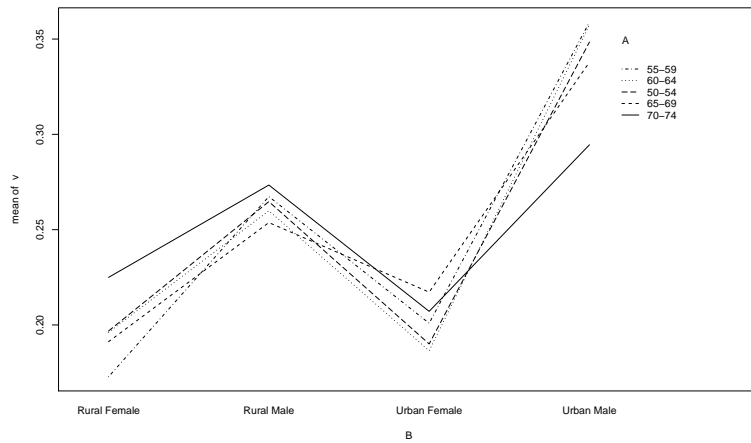
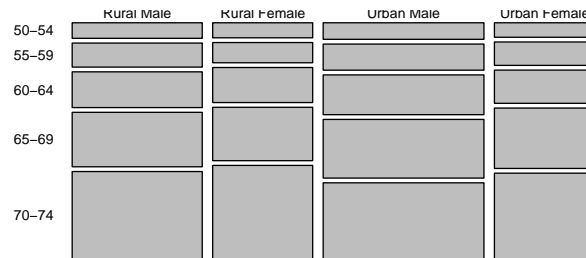
Es sei eine Kontingenztabelle K gegeben, die wir in einem solchen Profildiagramm darstellen wollen. Zuerst müssen wir die entsprechenden bedingten relativen Häufigkeiten $h_{b_j|a_i}$ erzeugen, und speichern diese dann nicht als Tabelle, sondern als Vektor:

4 Multivariate Datenanalyse

VADeaths



t(VADeaths)



```
> v<-as.vector(prop.table(K,margin=1))
```

Nun müssen wir noch den `x.factor` (wir werden ihn unter dem Namen `B` speichern, da es sich um die Ausprägungen des Merkmals `B` handelt) und das `trace.label`, das wir mit `A` bezeichnen werden, erstellen. Der Vektor `v` ist eine Aneinanderreihung der Spalten der Kontingenztabelle, und wir wollen nun Vektoren erstellen, die jeder Komponente von `v` die zugehörige Ausprägung von `A` bzw. `B` zuordnen. Der Einfachheit halber beginnen wir mit `A`. Zu jeder Spalte der Kontingenztabelle gehört der Vektor (a_1, \dots, a_I) der ja gerade den Zeilennamen entspricht. Um also `A` zu erzeugen, müssen wir den Vektor der Zeilennamen J -mal aneinanderhängen, da wir J Spalten vorliegen haben, also:

```
> as<-dim(K)[2]           #Anzahl der Spalten
> az<-dim(K)[1]          #Anzahl der Zeilen

> A<-factor(rep(dimnames(K)[[1]],as))
```

`B` ergibt sich nun dadurch, dass wir spaltenweise in der Kontingenztabelle die Ausprägungen von `B` durchgehen. In der ersten Spalte steht immer nur b_1 , in der zweiten immer b_2 usw., und das jeweils I -mal. Jeder Spaltenname der Tabelle wird also az -mal eingetragen. Der entsprechende Befehl lautet also:

```
> B<-factor(rep(dimnames(K)[[2]],rep(az,as)))
```

Das Diagramm wird nun mit diesem Befehl erzeugt:

```
> interaction.plot(B,A,v)
```

4.3 Zusammenhang zwischen quantitativen Merkmalen

In diesem letzten Abschnitt des Kapitels wollen wir uns mit der Zusammenhangsanalyse bei zwei quantitativen Merkmalen beschäftigen. Das ist natürlich nur dann sinnvoll, wenn es sich bei den beiden Stichproben um gepaarte Stichproben handelt, d.h. wenn jeweils an einem Merkmalsträger zwei Merkmale erhoben wurden bzw. wenn ein Merkmal jeweils unter zwei verschiedenen Bedingungen gemessen wurde. (Wenn die Stichproben unabhängig voneinander zu unterschiedlichen Merkmalen gezogen wurden, dürfte sich kein Zusammenhang erkennen lassen...) Einen bildlichen Eindruck des Zusammenhangs erhält man durch ein Streudiagramm (einen Scatterplot) der Daten. Liegen uns also zwei gepaarte Stichproben (x_1, \dots, x_n) , (y_1, \dots, y_n) mit einem $n \in \mathbb{N}$ vor, erzeugen wir das Streudiagramm einfach durch

```
> plot(c(x1, ..., xn), c(y1, ..., yn))
```

Liegen die Punkte in dem Streudiagramm ungefähr auf einer Geraden, liegt ein linearer Zusammenhang vor, der je nach Steigung der Geraden positiv oder negativ ist. Ähnelt das Streudiagramm einer Kurve, liegt zwischen den Merkmalen ein funktionaler (aber nicht linearer) Zusammenhang vor. Zeigt das Diagramm mehrere Punktwolken, deutet dies auf einen merkmalsbezogenen Zusammenhang hin, der eine Einteilung der Merkmalsträger in Gruppen erlaubt. Ist jedoch keinerlei Systematik in der Grafik erkennbar, deutet dies daraufhin, dass kein Zusammenhang zwischen den Merkmalen vorliegt. Häufig kann es bei der Auswertung eines solchen Diagrammes hilfreich sein, eine Gerade in die Grafik einzufügen. Hat man schon (in diesem Fall mit `plot`) eine Zeichnung erstellt, kann man mit der Funktion `abline` eine Gerade in das bestehende Diagramm einzeichnen. Die durch $y = a + bx$ definierte Gerade erhält man mit

`abline`

```
> abline(a,b)
```

eine vertikale (`v`) oder horizontale (`h`) Gerade an der Stelle `u` erhält man mit

`v h`

```
> abline(v=u)
```

```
> abline(h=u)
```

Bei einem Datensatz mit mehr als zwei quantitativen Merkmalen kann man mit Hilfe der Funktion `pairs` eine Grafik erstellen, in die alle möglichen Streudiagramme zweier Merkmale eingebaut sind.

`pairs`

Beispiel 4.4. In R sind Edgar Andersens Irisdaten in dem Datensatz `iris` eingebaut, die zu drei verschiedenen Irissorten Messungen von vier verschiedenen Blütenblattkennzahlen enthalten. Wenn wir diese vier Größen auf paarweisen Zusammenhang grafisch untersuchen wollen, geben wir diesen Befehl ein:

```
> pairs(iris[,1:4])
```

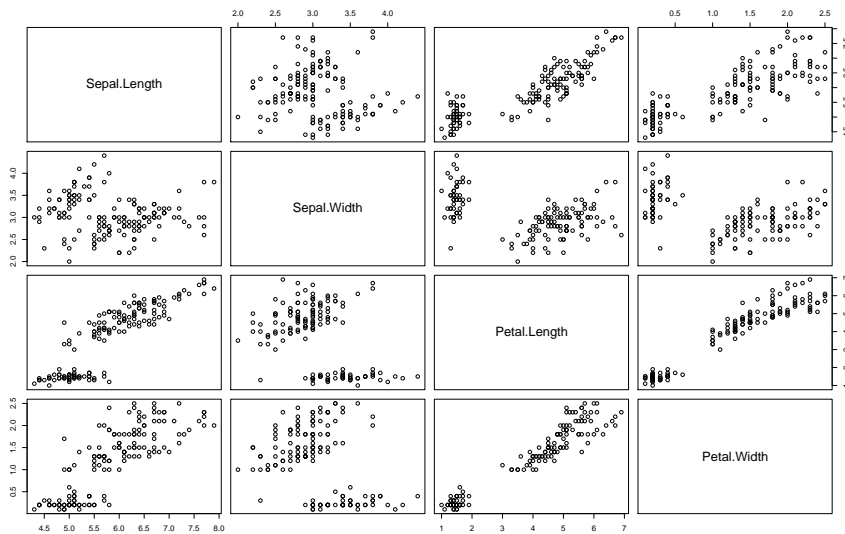
und erhalten diese Grafik: Man sieht, dass bei einigen Paarungen offenbar ein linearer Zusammenhang besteht und dass bei allen Diagrammen mehr oder weniger deutlich zwei Punktwolken entstanden sind. Man sollte also die Merkmalsträger noch einmal in Gruppen unterteilen und dann weiter untersuchen. Eine mögliche Unterteilung wäre z.B. gemäß `Petal.Length` in diejenigen, die dort einen Wert ≤ 2.5 haben, und die, deren Wert dort größer ist. Bei diesem Datensatz bietet es sich allerdings auch an, für jede Spezies gesondert die vier Merkmale auf Zusammenhänge zu untersuchen.

◇

Eine besondere Form des Streudiagrammes ist der **QQ-Plot** (QQ für Quantil-Quantil), bei dem die aufsteigend geordneten Datensätze gegeneinander geplottet werden. Dies erledigt die Funktion `qqplot`, in die wir die ungeordneten Vektoren einsetzen. Zeigt das Diagramm annähernd die Ursprungsgerade, sind die beiden untersuchten Merkmale ungefähr identisch verteilt, zeigt sich eine andere Gerade, geht die eine Verteilung

`qqplot`

4 Multivariate Datenanalyse



(approximativ) durch eine lineare Transformation aus der anderen hervor. Bilden die Punkte irgendeine Kurve, kann man lediglich festhalten, welche Verteilung in welchem Wertebereich dominiert. Will man speziell einen Datensatz (in Form eines Vektors) auf Normalverteilung überprüfen, muss man dazu nicht erst einen zweiten Vektor gemäß einer Normalverteilung erzeugen und beide dann in `qqplot` einsetzen, man kann stattdessen nur den Datenvektor in die Funktion `qqnorm` eingeben, die dann diese Schritte automatisch vornimmt. Entspricht der eingesetzte Vektor einer Normalverteilung, sollte sich in dem **Normal-QQ-Plot** eine Gerade ergeben (am besten mit Steigung 1). Weichen die gezeichneten Punkte allerdings stark von einer Geraden ab (die man mit der Funktion `qqline(Datensatz)` in die bestehende Grafik einfügen kann), sollte man eher davon ausgehen, dass eine Normalverteilungsannahme nicht gerechtfertigt ist. Auf die QQ-Plots und insbesondere die Normal-QQ-Plots werden wir im Rahmen der Testtheorie, der Regressions- sowie der Varianzanalyse noch einmal näher eingehen, da dort für einige Verfahren Normalverteilungen vorausgesetzt werden.

`qqnorm`

`qqline`

Neben diesen grafischen Darstellungen des Zusammenhangs zweier Merkmale können wir auch auf bestimmte Maßzahlen für den Zusammenhang zurückgreifen, sogenannte Korrelationskoeffizienten. Man betrachtet dabei zum Beispiel für die beiden Merkmale für alle $1 \leq i \leq n$ das Produkt aus den Differenzen der Werte der beiden Merkmale x_i und y_i und ihren Mittelwerten \bar{x} und \bar{y} (bezogen auf den gesamten zugehörigen Vektor) und bildet schließlich das arithmetische Mittel all dieser Produkte:

$$d_{x,y} := \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$$

$d_{x,y}$ heißt die **empirische Kovarianz** und kann auch durch

$$d_{x,y} = \overline{xy} - \bar{x}\bar{y}$$

dargestellt werden, wobei

$$\overline{xy} := \frac{1}{n} \sum_{i=1}^n x_i y_i$$

gilt. Bei dieser Formel ist allerdings zu beachten, dass sie bei der Benutzung in R zu Rundungsungenauigkeiten führen kann, die dann das Ergebnis (erheblich) verfälschen können. Wir erhalten die empirische Kovarianz in R durch

`> var(x, y)`

`var`

Sie ist allerdings wegen ihrer starken Abhängigkeit von der Wahl der Maßeinheiten der betrachteten Merkmale noch kein gutes Maß für den Zusammenhang. Durch Normierung wird sie jedoch zu einem solchen und heißt dann **Korrelationskoeffizient von Bravais-Pearson** $r_{x,y}$:

$$\begin{aligned} r_{x,y} &:= \frac{d_{x,y}}{\sqrt{d_{x,x}d_{y,y}}} \\ &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}}. \end{aligned}$$

Diesen Koeffizienten erhalten wir in R durch Eingabe von

`> cor(x, y)`

`cor`

Liegt der Korrelationskoeffizient von Bravais-Pearson nah bei 1 bzw. bei -1, besteht zwischen den Merkmalen ein positiver bzw. negativer linearer Zusammenhang. Bei Werten nah an 0 kann man schließen, dass kein linearer Zusammenhang zwischen den Merkmalen vorliegt, allerdings können durchaus andere Zusammenhänge bestehen. Der Korrelationskoeffizient von Bravais-Pearson ist offensichtlich ausreißerempfindlich. Eine robuste Größe erhalten wir, wenn wir statt der Beobachtungen x_i und y_i ihre Ränge in die obige Formel einsetzen. Dies ist der **Rangkorrelationskoeffizient von Spearman** r_s . Bevor diesen bilden können, müssen wir zunächst klären, was die Ränge der Beobachtungen sind. Der Rang r_i der Beobachtung x_i (der Rang s_i der Beobachtung y_i) gibt an, an welcher Stelle x_i in der geordneten Stichprobe steht. Sind Beobachtungen identisch, bilden wir Durchschnittsränge.

Beispiel 4.5. Der Einfachheit halber betrachten wir hier eine Stichprobe x mit sechs Einträgen:

Beobachtung	x_1	x_2	x_3	x_4	x_5	x_6
Wert	4	1	4	2	4	2

In der geordneten Stichprobe käme der kleinste Wert an die erste Stelle, und diesen Wert nimmt in diesem Fall x_2 an. Daher gilt $r_2 = 1$. Der zweitkleinste Wert, der in der Stichprobe vorkommt, ist die 2. Diese wird von x_4 und x_6 angenommen, die damit in der geordneten Stichprobe die Plätze 2 und 3 belegen würden. Für diese beiden

4 Multivariate Datenanalyse

Beobachtungen müssen wir nun einen Durchschnittsrang bilden, also

$$r_4 = r_6 = \frac{2+3}{2} = 2,5.$$

Schließlich bleibt nur noch der Wert 4, der dreimal in der Stichprobe vorkommt, weshalb die restlichen Ränge wie folgt berechnet werden:

$$r_1 = r_3 = r_5 = \frac{4+5+6}{3} = 5.$$

◇

Der Rangkorrelationskoeffizient von Spearman sieht demnach so aus:

$$r_s = \frac{\sum_{i=1}^n (r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\sum_{i=1}^n (r_i - \bar{r})^2 \sum_{i=1}^n (s_i - \bar{s})^2}},$$

wobei \bar{r} und \bar{s} die arithmetischen Mittel der Ränge seien. Durch die Bildung der Ränge kann dieser Koeffizient natürlich keine Maßzahl für den linearen Zusammenhang zwischen den Merkmalen sein, doch mit seiner Hilfe kann man die Merkmale auf einen monotonen Zusammenhang untersuchen. Werte bei 1 deuten auf einen (streng) monoton wachsenden Zusammenhang hin, Werte bei -1 auf einen (streng) monoton fallenden. Den Koeffizienten bekommen wir in R, indem wir zuerst mit Hilfe der Funktion `rank` die Ränge der Beobachtungen erstellen und diese dann (wie beim Korrelationskoeffizienten von Bravais-Pearson) in die Funktion `cor` eingeben:

```
> cor(rank(x),rank(y))
```

Eine weitere mögliche Vorgehensweise ist, das optionale Argument `method` zu benutzen, wo man den Namen als Zeichenkette eingibt, also `method="spearman"`.

Bei der Interpretation von Korrelationen muss man Vorsicht walten lassen. Ist die Korrelation hoch, wird oft unterstellt, dass die eine Größe die andere beeinflusst. Dies muss nicht stimmen, da es sein kann, dass beide Größen von einer dritten (uns unbekannt) Größe beeinflusst werden. Haben wir drei Merkmale X,Y,Z vorliegen und wollen kontrollieren, ob Merkmal X den Zusammenhang zwischen Y und Z bedingt, bieten sich uns zwei Vorgehensweisen: Zum einen können wir nur Personen / Objekte betrachten, die die gleiche Merkmalsausprägung des Merkmals X besitzen, und bestimmen für diese den Zusammenhang zwischen Y und Z. Zum anderen können wir den Einfluss des Merkmals X auf die Merkmale Y und Z statistisch bereinigen und den Zusammenhang zwischen letzteren beiden durch die Berechnung des **partiellen Korrelationskoeffizienten** $r_{YZ.X}$ bestimmen:

$$r_{YZ.X} := \frac{r_{YZ} - r_{XY}r_{XZ}}{\sqrt{(1 - r_{XY}^2)(1 - r_{XZ}^2)}}.$$

In R ist eine entsprechende Funktion zur Berechnung des partiellen Korrelationskoeffizienten unter dem Namen `pcor` im Paket `ggm` eingebaut, und durch

`pcor`
`ggm`

4 Multivariate Datenanalyse

```
> pcor(c(Y,Z,X),var(cbind(Z,X,Y)))
```

erhalten wir $r_{YZ.X}$.

5 Testtheorie

In der Praxis wird üblicherweise die Nullhypothese H_0 gegen die Gegenhypothese H_1 getestet.

Dabei unterscheiden wir mehrere Fälle, an die wir dann das Verfahren entsprechend anpassen:

a) Einstichprobenfall:

Es wird ein einziges Merkmal X auf Grund einer einfachen Zufallsstichprobe (X_1, \dots, X_n) bzgl. der Kenngrößen seiner Verteilung (Mittelwert, Median, Varianz, Verteilungsfunktion) getestet.

b) Zweistichprobenfall:

In diesem Fall wird ein Merkmal unter zwei Bedingungen untersucht oder man betrachtet zwei Merkmale, die am selben Merkmalsträger erhoben werden:

Man kann zwei unabhängige Zufallsstichproben $(X_{1,1}, \dots, X_{1,n_1}), (X_{2,1}, \dots, X_{2,n_2})$ mit $n_1, n_2 \in \mathbb{N}$ untersuchen, wobei sich die Randbedingungen bei der Entnahme von Stichprobe 1 und Stichprobe 2 nur in einer Randbedingung unterscheiden.

Man kann ein Merkmal unter zwei verschiedenen Bedingungen am selben Merkmalsträger untersuchen, wodurch man verbundene Stichproben (matched pairs) erhält: $(X_{1,1}, X_{1,2}), \dots, (X_{n,1}, X_{n,2})$. (Beispiel: Blutdruckuntersuchung bei Patienten vor und nach Verabreichung eines Medikamentes.)

Ebenso kann man zwei Merkmale X und Y am selben Merkmalsträger (unter jeweils gleichen Bedingungen) untersuchen, wodurch man wieder zwei verbundene einfache Stichproben bekommt: $(X_1, Y_1), \dots, (X_n, Y_n)$.

c) k-Stichprobenfall ($k \geq 3$):

Hierbei untersucht man ein Merkmal unter k verschiedenen Bedingungen oder k Merkmale am selben Merkmalsträger.

Zur Messung eines Merkmals unter k verschiedenen Bedingungen werden k unabhängige einfache Zufallsstichproben $(X_{1,1}, \dots, X_{1,n_1}), \dots, (X_{k,1}, \dots, X_{k,n_k})$ (aus disjunkten Teilgesamtheiten) gezogen. Dabei soll ein globaler Test die Unterschiede zwischen allen Gruppen simultan aufdecken. Besteht ein signifikanter Unterschied zwischen den Gruppen, führt man sogenannte post-hoc multiple Tests durch, um aufzudecken, welche Gruppen sich signifikant unterscheiden.

Man kann auch zufällig n Merkmalsträger aus der Grundgesamtheit auswählen und an jedem Merkmalsträger X_i , $1 \leq i \leq n$, das Merkmal unter k verschiedenen Bedingungen messen. Eine weitere Möglichkeit besteht darin, eine Zufallsstichprobe von $n \cdot k$ Merkmalsträgern aus der Grundgesamtheit auszuwählen und sie in n Blöcke der Länge k einzuteilen. Dabei wird an jedem Merkmalsträger das Merkmal nur einmal erhoben: $(X_{1,1}, \dots, X_{1,n}), \dots, (X_{k,1}, \dots, X_{k,n})$.

Wir untersuchen parametrische Testverfahren (d.h. Tests auf den Parameter θ mit $\theta \in \mathbb{R}^n$) und nicht parametrische Testverfahren ($\theta \notin \mathbb{R}^n \forall n \in \mathbb{N}$). Bei parametrischen Verfahren setzen wir generell voraus, dass die Daten metrisches Messniveau und eine endliche Varianz besitzen. Dabei kann man Mittelwertvergleiche anstellen, auf Varianzhomogenität testen, Varianzanalyse betreiben oder Korrelationstests durchführen. Da Mittelwertvergleiche häufige Testobjekte sind, gibt es dazu nun eine Übersicht.

5.1 Mittelwertvergleiche (parametrische Verfahren)

In der parametrischen Testtheorie können wir im Wesentlichen drei verschiedene Arten von Hypothesen-Paarungen testen, wobei es sich um einen zweiseitigen und zwei einseitige Testprobleme handelt:

- a) $H_0: \vartheta = \vartheta_0$ gegen $H_1: \vartheta \neq \vartheta_0$
- b) $H_0: \vartheta \leq \vartheta_0$ gegen $H_1: \vartheta > \vartheta_0$
- c) $H_0: \vartheta \geq \vartheta_0$ gegen $H_1: \vartheta < \vartheta_0$

Den zweiseitigen Fall $\vartheta \in [c_1, c_2]$ mit einem beliebigen können wir hier nicht behandeln; R bietet bei den Testfunktionen keine Möglichkeit, einen solchen Fall einzugeben.

Die Prüfgröße, also die Teststatistik, die üblicherweise verwendet wird, ist

$$T(x) = \frac{\hat{\vartheta}(x) - \vartheta_0}{\hat{\sigma}_{\hat{\vartheta}(x)}}$$

wobei $\hat{\vartheta}(x)$ ein Schätzer für ϑ sei und $\hat{\sigma}_{\hat{\vartheta}(x)}$ ein Schätzer für die Varianz von $\hat{\vartheta}$. T besitze im folgenden die Verteilung q ; q_α bezeichne dabei für $\alpha \in (0, 1)$ das α -Quantil. Bei Betrachtung dieser Teststatistik, des standardisierten Schätzers $\hat{\vartheta}(x)$, ergeben sich für q (in den meisten Fällen) gut bekannte und entsprechend gut vertafelte Verteilungen, für die sich die Quantile leichter finden lassen. Dank diverser Computerprogramme (u.a. R) ist man auf solche Vertafelungen nicht mehr angewiesen...

Für eine solche Teststatistik T können wir zu beliebigem Niveau $1 - \alpha$, $\alpha \in (0, 1)$, ein Konfidenzintervall $C(x)$ bestimmen, also ein solches Intervall, das mit Wahrscheinlichkeit $1 - \alpha$ den wahren Parameter ϑ_0 überdeckt. Diese Intervalle hängen natürlich vom vorgegebenen Testproblem ab. Zu den oben unterschiedenen Fällen ergeben sich folgende Intervalle:

- a) $[\hat{\vartheta}(x) - q_{1-\frac{\alpha}{2}}\hat{\sigma}_{\hat{\vartheta}(x)}, \hat{\vartheta}(x) - q_{\frac{\alpha}{2}}\hat{\sigma}_{\hat{\vartheta}(x)}]$
- b) $[\hat{\vartheta}(x) - q_{1-\alpha}\hat{\sigma}_{\hat{\vartheta}(x)}, \infty)$
- c) $(-\infty, \hat{\vartheta}(x) - q_{\alpha}\hat{\sigma}_{\hat{\vartheta}(x)}]$

Entsprechend ergeben sich für die drei Fälle auch unterschiedliche Ablehnungsbereiche der Nullhypothese. Wir lehnen die Nullhypothese H_0 ab, wenn $\vartheta_0 \notin C(x)$ gilt. In den drei Fällen besetzt dies:

- a) Annahme von H_1 , falls $T < q_{\frac{\alpha}{2}}$ oder $T > q_{1-\frac{\alpha}{2}}$
- b) Annahme von H_1 , falls $T > q_{1-\alpha}$
- c) Annahme von H_1 , falls $T < q_{\alpha}$

Bei solchen Entscheidungen ist zu beachten, dass wir auf Grund unserer Stichprobe nur die Nullhypothese ablehnen können bzw. sie nicht ablehnen können. Mithilfe der Stichprobe kann man das Vorliegen der Nullhypothese nicht beweisen, sich also auch nicht für die Nullhypothese entscheiden.

Um herauszufinden, ob die Nullhypothese abgelehnt werden kann oder nicht, kann man sich am sogenannten p-Wert orientieren. Dies ist das kleinste Niveau $\tilde{\alpha}$ zu dem man bei Vorliegen der Stichprobe x und Prüfgrößenwert $T(x) = t$ die Nullhypothese noch ablehnen kann. Mit fallendem Niveau α wächst das Konfidenzintervall $C(x)$ und somit der Bereich der Stichprobenwerte, für die wir H_0 nicht ablehnen. Wir erhalten

- a) $\tilde{\alpha} = \mathbb{P}_{\vartheta_0}(|T| \geq |t|)$, falls q symmetrisch zu 0 ist
- b) $\tilde{\alpha} = \mathbb{P}_{\vartheta_0}(T \geq t)$
- c) $\tilde{\alpha} = \mathbb{P}_{\vartheta_0}(T \leq t)$

Führen wir also einen Test aus und erhalten den p-Wert $\tilde{\alpha}$ (der bei Testauswertungen in R immer mitgeliefert wird), vergleichen wir danach das Niveau α , zu dem getestet wurde, mit diesem Wert. Ist $\alpha < \tilde{\alpha}$ können wir die Nullhypothese H_0 nicht ablehnen. Gilt $\alpha \geq \tilde{\alpha}$ nehmen wir die Gegenhypothese H_1 an.

Beispiel 5.1. Wir testen zwei Stichproben x und y mit einer Teststatistik T zu den Niveaus $\alpha_1 < \alpha_2$ für das zweiseitige Testproblem

$$H_0 : \vartheta = \vartheta_0 \text{ gegen } H_1 : \vartheta \neq \vartheta_0.$$

Zu den Stichproben seien die p-Werte p_x und p_y mit

$$\alpha_1 < p_y < \alpha_2 < p_x$$

gegeben. Welche Entscheidungen sind nun zu treffen? Für die Stichprobe x sind beide Niveaus α_1 und α_2 kleiner als der p-Wert p_x , d.h. wir können für beide die Nullhypothese nicht ablehnen. Bei der Stichprobe y gilt $\alpha_1 < p_y < \alpha_2$, weshalb wir zum Niveau α_1 H_0 nicht ablehnen können, zum Niveau α_2 aber schon. Zum Niveau α_2 ist also die Gegenhypothese H_1 statistisch gesichert.

◇

5.2 Das Einstichprobenproblem

Haben wir eine einfache Stichprobe vorliegen, können wir mithilfe von Tests zwei Fragebereiche bearbeiten: Wir können die Verteilung des Merkmals auf ihre Lageparameter (also Mittelwert und Median) untersuchen oder testen, ob das Merkmal in der Grundgesamtheit eine bestimmte Verteilung besitzt. Beim Testen auf Lageparameter handelt es sich um parametrische Tests, schließlich sind diese Parameter $\in \mathbb{R}^d$ mit einem geeigneten $d \geq 1$. Bei der Untersuchung auf Vorliegen einer bestimmten Verteilung bewegt man sich allerdings im Bereich der nichtparametrischen Statistik, da Verteilungen nunmal für kein $d \geq 1$ Element von \mathbb{R}^d sind. Wir beginnen mit den Tests auf den Lageparameter.

Wenn die Stichprobe $\mathcal{N}(\mu, \sigma^2)$ -verteilt sein könnte mit unbekanntem μ und σ^2 , kann man einen **t-Test** anwenden, um auf die Lage des Mittelwerts μ zu testen. Um zu überprüfen, ob eine Normalverteilungsannahme gerechtfertigt ist, kann man verschiedene grafische Methoden anwenden: Man kann Histogramme, Boxplots oder QQ-Normal-Plots heranziehen, um sich einen Eindruck der Verteilung zu verschaffen, wobei es sich häufig anbietet, mehrere der Methoden anzuwenden. Bei einer Stichprobenlänge ≥ 30 reicht schon die Annahme, dass die X_i unabhängig und identisch verteilt sind mit Erwartungswert μ und Varianz σ^2 , da sich dann der zentrale Grenzwertsatz anwenden lässt.

Das Testproblem stellt sich nun durch die Hypothesen

$$H_0 : \mu = \mu_0 \quad \text{gegen} \quad H_1 : \mu \neq \mu_0$$

(bzw. die entsprechenden einseitigen Fälle) dar. Wir betrachten, wie im vorhergehenden Abschnitt eingeführt, die Prüfgröße $T(x)$, wobei wir hier das arithmetische Mittel

$$\hat{\mu}(x) = \bar{x} := \frac{1}{n} \sum_{i=1}^n x_i$$

als erwartungstreuen Schätzer für den Mittelwert μ verwenden sowie

$$\hat{\sigma}_{\hat{\mu}^2(x)} = \frac{s}{\sqrt{n}}$$

setzen, wobei die Stichprobenvarianz

$$s^2 := \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

wegen ihrer Erwartungstreue für die Varianz der X_i eingesetzt wird. Die Teststatistik sieht damit folgendermaßen aus:

$$T(x) = \frac{\sqrt{n}(\bar{x} - \mu_0)}{s}$$

Diese Statistik besitzt unter der Nullhypothese $\mu = \mu_0$ eine t-Verteilung mit $n - 1$ Freiheitsgraden t_{n-1} bzw. ist approximativ $\mathcal{N}(0, 1)$ -verteilt.

5 Testtheorie

Als Konfidenzintervall erhält man z.B. $[\bar{x} - t_{n-1, 1-\frac{\alpha}{2}} \cdot \frac{s}{\sqrt{n}}, \bar{x} + t_{n-1, 1-\frac{\alpha}{2}} \cdot \frac{s}{\sqrt{n}}]$ (entsprechend für die einseitigen Fälle), und daher als Ablehnungsbereich für die Nullhypothese $H_0 \{|T| > t_{n-1, 1-\frac{\alpha}{2}}\}$. Der p-Wert berechnet sich dann zu $2 \cdot t_{n-1}(|T(x)|, \infty)$.

Wollen wir in R einen t-Test durchführen (nachdem wir uns von der Normalverteilung überzeugt haben!), benutzen wir dazu die Funktion `t.test`, die folgendermaßen benutzt wird: `t.test`

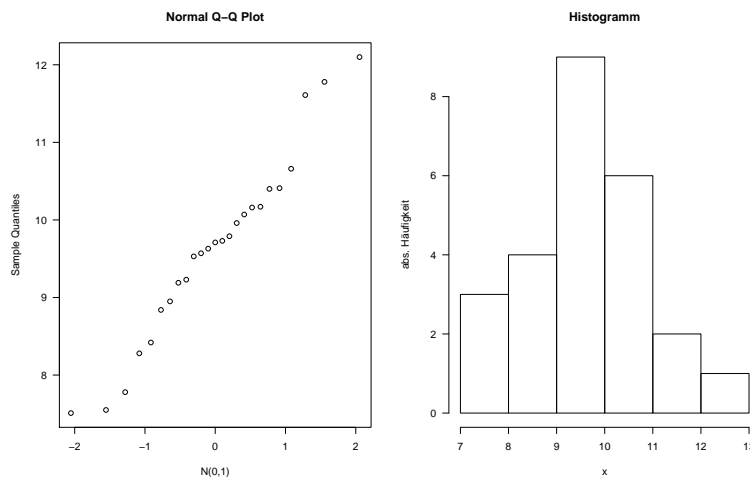
```
t.test(x, alternative=c("two.sided", "less", "greater"), mu=0, conf.level=0.95)
```

Dabei wird die Stichprobe als Argument `x` eingesetzt, mit `alternative` wählt man aus, welches Testproblem behandelt werden soll, indem man die Gegenhypothese (die Alternative) modelliert. `alternative` steht standardmäßig auf „two.sided“. Das nächste Argument `mu` gibt an, welcher der kritische Parameterwert sein soll, das μ_0 . Auch hier gibt es einen Standardwert, die 0. Schließlich wird noch das (Konfidenz-)Niveau $1 - \alpha$ gefordert, das bei Nichteingabe auf 95% gesetzt wird. Man kann noch weitere Argumente eingeben, einige davon werden im Fall einer Mehrfachstichprobe (Untersuchung mehrerer Merkmale) gebraucht. `alternative`
`mu`

Beispiel 5.2. Es sei die Stichprobe

$$x = (10.40, 11.78, 10.66, 12.10, 7.51, 9.57, 9.71, 8.28, 9.53, 10.17, 10.41, 9.19, 7.78, 9.63, 7.55, 9.73, 9.96, 10.07, 9.79, 11.61, 10.16, 8.84, 9.23, 8.42, 8.95)$$

Der Normal-QQ-Plot und auch das Histogramm zeigen ein für eine Normalverteilung typisches Bild: Wir wollen folgende Hypothesen mithilfe des t-Tests zu den Niveaus 90%



und 95% testen:

$$H_0 : \mu_0 \geq 10 \text{ gegen } H_1 : \mu_0 < 10.$$

Wir geben also Folgendes ein:

```
t.test(x,alternative="less",mu=10,conf.level=0.9)
```

```
One Sample t-test
```

```
data: x
t = -1.4898, df = 24, p-value = 0.07466
alternative hypothesis: true mean is less than 10
98 percent confidence interval:
-Inf 10.16420
sample estimates:
mean of x
  9.6412
```

Als Ausgabe des Tests erhält man eine Liste mit den wichtigsten Werten, wobei in der Überschrift angegeben wird, um welchen Stichprobenfall es sich handelt. In der ersten Zeile stehen der Wert der Prüfgröße, die eine t-Verteilung mit $df=24$ Freiheitsgraden besitzt, wie auch der p-Wert. Dieser ist größer als 5% (=100%-95%), d.h. wir können die Nullhypothese nicht ablehnen. Als nächstes wird die Gegenhypothese angegeben, wodurch das Testproblem vollständig bestimmt ist. Danach wird das zugehörige Konfidenzintervall aufgelistet. (Da 95% die Standardeinstellung für das Niveau ist, hätten wir dies in obigen Befehl nicht eingeben brauchen.) Den Abschluss bildet das arithmetische Mittel der Stichprobe als erwartungstreuer Schätzer für den Mittelwert der Normalverteilung. Wir sehen, dass dieser Wert tatsächlich kleiner ist als 10, dennoch können wir bei dieser Stichprobe H_0 nicht ablehnen und uns für $H_1: \mu_0 < 10$ entscheiden!

Für das zweite anstehende Testproblem geben wir ein:

```
t.test(x,alternative="two.sided",mu=9,conf.level=0.9)
```

```
One Sample t-test
```

```
data: x
t = 2.6623, df = 24, p-value = 0.01363
alternative hypothesis: true mean is not equal to 9
90 percent confidence interval:
  9.229145 10.053255
sample estimates:
mean of x
  9.6412
```

Hier können wir die Nullhypothese verwerfen, denn der p-Wert ist kleiner als 10%. Außerdem ist der Wert der Teststatistik größer als der p-Wert und das betrachtete $\mu = 9$ liegt nicht im Konfidenzintervall. Jede dieser drei Tatsachen reicht zur Begründung der

Entscheidung.

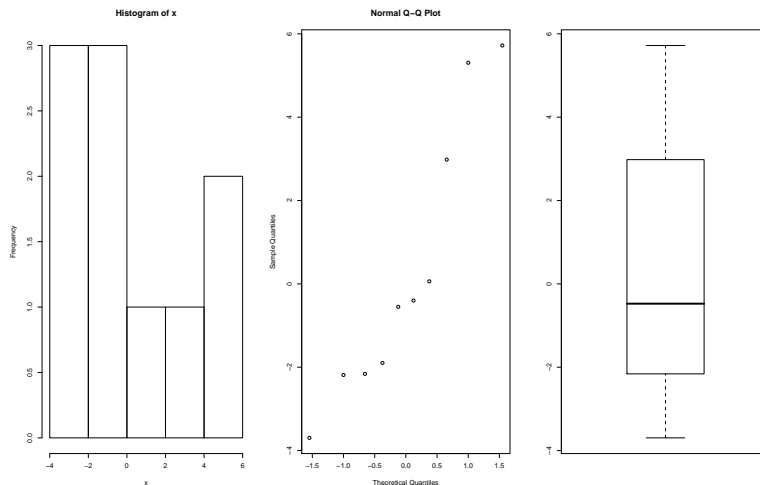
◇

Wenn man eine gegebene Stichprobe auf Normalverteilung untersucht, um die Voraussetzung für die Anwendung eines t-Tests zu überprüfen, kann es insbesondere bei kleinen Stichprobenumfängen sein, dass die Grafiken oder die Maßzahlen einen falschen Eindruck vermitteln. So kann z.B. eine Normalverteilung vorliegen, doch die Grafiken zur Stichprobe sehen gar nicht danach aus.

Beispiel 5.3. Es sei eine Normalverteilung mit Mittelwert 1 und Varianz 9 gegeben. Wir „ziehen“ eine Stichprobe, indem wir gemäß dieser Verteilung Zufallszahlen generieren. Die Stichprobe habe die Länge 10. Wir schauen uns die zugehörigen Grafiken an:

```
x<-rnorm(10,mean=1,sd=3)
hist(x)
qqnorm(x)
boxplot(x)
```

Weder Histogramm, noch Normal-QQ-Plot, noch Boxplot lassen eine Normalverteilung



annahme zu, obwohl tatsächlich eine Normalverteilung gegeben ist.

◇

Im Fall kleiner Stichproben ist man auf weitere Informationen bzw. theoretische Überlegungen angewiesen, die helfen, die richtige Verteilung zu finden. Führen z.B. Physiker Versuche durch, so ist im Allgemeinen durch die theoretischen Vorüberlegungen schon bekannt, welche Verteilung das untersuchte Merkmal besitzen sollte. Ansonsten geht man häufig von der Annahme aus, dass die auftretenden Messfehler einer Normalverteilung entspringen und dass somit deterministische Messgrößen ebenfalls normalverteilt sind.

(Sind die Messgrößen nicht deterministisch, sondern unterliegen einer gewissen Verteilung, ergibt sich die Verteilung des Merkmals als Linearkombination der Verteilungen der Messgrößen und der Normalverteilung der Messfehler...)

In den Fällen, in denen wir nicht von einer Normalverteilung ausgehen können, müssen wir auf andere Tests als den t-Test zurückgreifen, um die Lage der Verteilung zu untersuchen. Es gibt Tests auf den Wert des Medians, von denen wir zwei hier behandeln wollen. Beide basieren auf der Überlegung, dass (ca.) 50% der Beobachtungen größer und ebenso viele kleiner als der tatsächliche Median M_0 sind. Wenn der Wert M_0 nicht in der Stichprobe auftritt, sollte dementsprechend die Anzahl der Beobachtungen, die größer als M_0 sind, $B(n, 0.5)$ -verteilt sein. Tritt der Wert doch auf, können wir keine genaue Verteilung zuordnen. Deshalb wird die Stichprobe auf die Werte reduziert, die von M_0 verschieden sind, dann ist die Verteilungsannahme wieder gerechtfertigt, allerdings mit einer veränderten Stichprobenlänge $n.n\text{eu} < n$. Die Teststatistik ist in diesem Fall also folgende:

$$T(x) = \sum_{i=1}^n \mathbf{1}_{(0, \infty)}(x_i - M_0)$$

bzw. läuft die Summation nur über $\{1, \dots, n.n\text{eu}\}$, wenn wir Werte aus der Stichprobe entfernen mussten. Den Test, der auf der Auswertung dieser Statistik beruht, nennt man **Vorzeichentest**. Den Wert der Statistik bei gegebener Stichprobe x erhalten wir durch:

```
sum(x>M0)
```

```
bzw. durch
x.n\text{eu}<-x[x!=M0]
```

Den Test führt man mit der Funktion `binom.test` durch:

```
binom.test
```

```
binom.test(x,n,p=0.5,alternative=c("two.sided","less","greater"),
           conf.level=0.95)
```

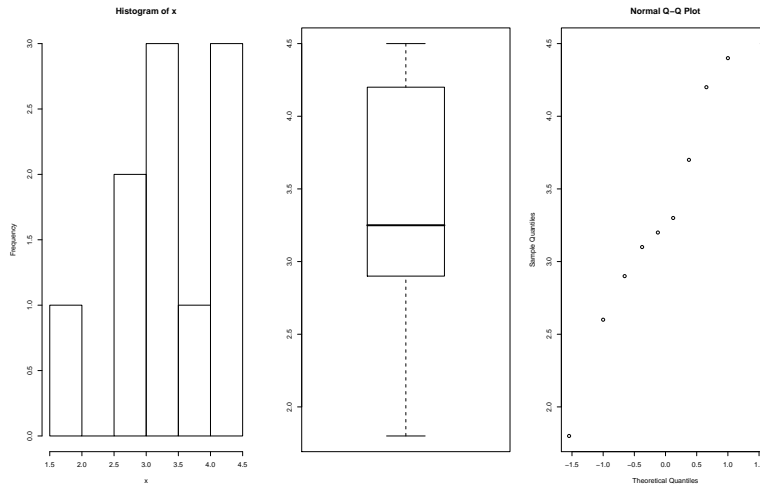
Dabei wird für x der Wert der Prüfgröße eingesetzt, nicht die Stichprobe! Wollen wir zum Niveau 95% testen, ob $M_0 = 5$ gilt, geben wir

```
binom.test(sum(x>5),length(x))
```

ein. Erfolgswahrscheinlichkeit, Alternative und Niveau müssen nicht eingegeben werden, da die Standardeinstellungen gerade `p=0.5`, `"two.sided"` und `0.95` sind.

Beispiel 5.4. Es sei die Stichprobe $x=(3.7,4.2,1.8,3.1,4.4,4.5,2.6,2.9,3.3,3.2)$ gegeben. Die grafische Auswertung zeigt, dass eine Normalverteilungsannahme nicht unbedingt gerechtfertigt ist. Daher greifen wir hier auf einen Vorzeichentest und nicht einen t-Test zurück. Nun soll überprüft werden, ob der Median M_0 signifikant größer als 3.3 ist, d.h.

5 Testtheorie



wir untersuchen das Testproblem

$$H_0 : M_0 \leq 3.3 \quad \text{gegen} \quad H_1 : M_0 > 3.3.$$

Es kann nur die Gegenhypothese signifikant (zu einem bestimmten Niveau) sein, da wir nur für sie und nicht für die Nullhypothese eine Entscheidung treffen können. Die Nullhypothese können wir höchstens ablehnen bzw. nicht ablehnen. Wenn wir uns bei einem gegebenen Testproblem für die Alternative entscheiden, ist diese signifikant. Bei dem hier vorliegenden Testproblem sei wieder $\alpha = 0.05$ gegeben, also das Niveau $1 - \alpha = 0.95$. Um den Vorzeichentest durchführen zu können, müssen wir zunächst die Stichprobe vom Wert 3.3 bereinigen, können dann den Wert der Teststatistik bestimmen und diesen in den Test einsetzen:

```
x.new<-x[x!=3.3]
y<-sum(x.new>3.3)
length(x.new)
[1] 9
binom.test(y,9,alternative="greater")
```

Exact binomial test

```
data: y and 9
number of successes = 4, number of trials = 9, p-value = 0.7461
alternative hypothesis: true probability of success is greater than 0.5
95 percent confidence interval:
0.1687505 1.0000000
sample estimates:
probability of success
0.4444444
```

Statt zuerst den Wert der Teststatistik und die Stichprobenlänge zu bestimmen, hätten wir diese direkt in den Test einsetzen können:

```
binom.test((x.new>3.3),length(x.new),alternative="greater")
```

Das Ergebnis wäre dasselbe gewesen. Aus der Testauswertung entnehmen wir den p-Wert 0.7461, der größer als 0.05 ist. Die Nullhypothese kann also nicht abgelehnt werden, der Median ist also zum Niveau 0.95 nicht signifikant größer als 3.3. Da wir hier zunächst die Stichprobe bereinigen mussten, handelt es sich in diesem Fall um einen konditionalen Vorzeichentest.

◇

Für hinreichend umfangreiche Stichproben ($n \geq 25$) kann die Binomialverteilung der Teststatistik durch die Normalverteilung $\mathcal{N}(\frac{n}{2}, \frac{n}{4})$ approximiert werden. In diesem Fall kann man die standardisierte Teststatistik

$$Z(x) = \frac{T(x) - \frac{n}{2}}{\frac{1}{2}\sqrt{n}} \sim \mathcal{N}(0, 1)$$

betrachten. Diese Approximation ist in dem Fall interessant, wo man keinen Computer zur Verfügung hat und auf Vertafelungen von Verteilungen zurückgreifen muss.

Beim Vorzeichentest werden nur sehr wenige der in der Stichprobe enthaltenen Informationen verwendet, nämlich nur die, ob die Werte größer oder kleiner als der vermutete Median sind. Wenn wir davon ausgehen können, dass unserer Stichprobe eine symmetrische (und stetige) Verteilung zugrundeliegt, hat es auch Sinn, die Ränge der Stichprobenwerte zur Auswertung heranzuziehen. Man kann also statt zu zählen, wie viele Stichprobenwerte größer als M_0 sind, deren Ränge addieren und diese Summe in Abhängigkeit von der Stichprobenlänge n auswerten. Wir bilden also zunächst die Ränge der Abstände der Beobachtungen von M_0 , bezeichnet mit

$$R(|x_i - M_0|).$$

(Wie Ränge gebildet werden haben wir bereits im Zusammenhang mit dem Rangkorrelationskoeffizienten von Spearman gesehen.) Diese Ränge benutzen wir nun zur Berechnung der oben angesprochenen Teststatistik:

$$W^+(x) = \sum_{i=1}^n \mathbf{1}_{(0,\infty)}(x_i - M_0) R(|x_i - M_0|).$$

Die Verteilung dieser Statistik W^+ (keine „typische“ Verteilung) lässt sich in Abhängigkeit von n berechnen, was allerdings zu einigem Aufwand führen kann. Für hinreichend große n (also ≥ 30) kann man jedoch mit der Normalapproximation arbeiten und wählt die

Statistik

$$Z = \frac{W^+ - \frac{n(n+1)}{4}}{\sqrt{\frac{n(n+1)(2n+1)}{24}}} \sim \mathcal{N}(0, 1)$$

als Prüfgröße. Glücklicherweise übernimmt diese ganze Arbeit die Funktion `wilcox.test` für uns, mit der wir den oben beschriebenen Test, den **Wilcoxon-Vorzeichen-Rangtest** `wilcox.test` durchführen und die wir folgendermaßen verwenden.

```
wilcox.test(x, alternative=c("two.sided", "less", "greater"), mu=0,
            conf.level=0,95, ...)
```

Das erste Argument `x` ist einfach die gegebene Stichprobe, `mu` steht für den zu testenden kritischen Wert des Medians M_0 . Mit den zusätzlichen Argumenten `exact` und `correct` kann man einstellen, ob die exakte Verteilung von W^+ bestimmt werden soll und nicht die approximative von Z , bzw. ob eine Stetigkeitskorrektur verwendet werden soll oder nicht. Letzteres rührt daher, dass die (exakte) Verteilung der Teststatistik nicht unbedingt stetig sein muss, man allerdings in seinen Annahmen für die Verwendung des Tests von einer stetigen Verteilung des untersuchten Merkmals in der Grundgesamtheit ausgegangen ist. `exact`
`correct`

Ein Problem bei der Anwendung des Tests ergibt sich, wenn in der Stichprobe mehrere Werte mit demselben Abstand von M_0 auftreten, wenn also sogenannte Rangbindungen auftreten. Liegen sogar genau dieselben Werte vor, spricht man von Bindungen (engl.: ties). In solchen Fällen sind die zugehörigen Ränge Durchschnittsränge, wodurch der Wert der Prüfgröße verfälscht wird. Ist die Stichprobe lang genug, dass man auf die Normalapproximation zurückgreifen kann, kann bei der Statistik Z eine entsprechende Korrektur in Form einer Varianzkorrektur vorgenommen werden:

$$\text{Var}(W^+) = \frac{n(n+1)(2n+1)}{24} - \frac{1}{48} \sum_{j=1}^r (b_j^3 - b_j),$$

wobei r die Anzahl der verschiedenen Werte sei, die mehrfach auftreten, und b_j die Anzahl der Beobachtungen in der j -ten Bindungsgruppe. Eine Bindungsgruppe besteht dabei gerade aus den Beobachtungen, die denselben Wert haben; hier liegen also r Bindungsgruppen vor.

Bei der Durchführung eines Wilcoxon-Vorzeichen-Rangtests weist `R` gegebenenfalls auf vorhandenen Bindungen hin ebenso auf die mögliche Ungenauigkeit des p -Werts. Die exakten Werte erhält man dann mithilfe der Funktion `wilcox.exact` aus dem Paket `exactRankTests`.

Der Rest dieses Abschnitts beschäftigt sich mit zwei Verfahren der nichtparametrischen Statistik, mit denen man untersuchen kann, ob eine bestimmte Verteilung P_0 vorliegt. (Der Parameterraum ist also der Raum aller Verteilungen, der offenbar keine Teilmenge eines \mathbb{R}^d , $d \in \mathbb{N}$ ist. Daher also nichtparametrische Statistik.) Die Hypothesen, die wir hier testen, sind also

$$H_0 : P = P_0 \quad \text{gegen} \quad H_1 : P \neq P_0$$

wobei P die vorliegende unbekannte Verteilung und P_0 die vermutete vollständig bekannte Verteilung des betrachteten Merkmals sei. (Natürlich können hier keine einseitigen Fälle betrachtet werden.) Als erstes Verfahren sehen wir uns den χ^2 -**Anpassungstest** an, bei dem die absoluten Häufigkeiten der einzelnen Ausprägungen bzw. der Klassen von Ausprägungen mit den absoluten Häufigkeiten verglichen werden, die bei unter P_0 zu erwarten wären. Bei gegebener Stichprobe $x = (x_1, \dots, x_n)$ können wir im diskreten Fall die absoluten Häufigkeiten der Ausprägungen a_1, \dots, a_k ohne weiteres bestimmen, im stetigen Fall müssen wir zunächst noch k disjunkte Klassen K_1, \dots, K_k einteilen und können dann die absoluten Häufigkeiten bestimmen. Die Wahl der Klassengrenzen ist hier wieder einer gewissen Willkür unterlegen, wodurch sich bei unterschiedlichen Einteilungen recht unterschiedliche Resultate ergeben können. Bei der Einteilung ist zu beachten, dass die Klassen nicht zu klein sein sollten. Die erwartete Anzahl der Werte in diesen Klassen sollte immer mindestens eins sein und bei mindestens 80% der Klassen mindestens bei fünf liegen. Ansonsten wird der Test sehr ungenau und damit für uns wertlos. Diese erwarteten Häufigkeiten werden mit den tatsächlichen verglichen. Bei großen Abweichungen entscheidet man sich dann also für die Gegenhypothese einer von P_0 verschiedenen Verteilung.

Klasse/Ausprägung	1	2	...	k	\sum
abs. Häufigkeiten	n_1	n_2	...	n_k	n

wobei die n_i die absoluten Häufigkeiten der Ausprägungen bzw. der Klassen seien. Wir definieren nun

$$\pi_i := \begin{cases} P_0(K_i) & \hat{=} \text{Wahrscheinlichkeit der Klasse } K_i \text{ unter } P_0, \\ P_0(a_i) & \hat{=} \text{Wahrscheinlichkeit der Ausprägung } a_i \text{ unter } P_0, \end{cases}$$

und

$$e_i := n_i \pi_i,$$

was der theoretisch unter P_0 erwarteten Klassen- bzw. Ausprägungshäufigkeit entspricht. Wie oben schon erwähnt muss $e_i \geq 1$ für alle $1 \leq i \leq k$ und $e_i \geq 5$ für mindestens 80% der $1 \leq i \leq k$ gelten, da sonst keine fundierten Schlüsse aus den Daten gezogen werden können. Um nun ein Maß für die Abweichungen der tatsächlichen von den erwarteten Häufigkeiten zu haben, betrachten wir wie üblich eine Summe der gewichteten quadratischen Abweichungen:

$$T(x) = \sum_{i=1}^k \frac{(n_i - e_i)^2}{e_i}.$$

Wir bilden wir hier nicht wie sonst das arithmetische Mittel der quadratischen Abweichungen (Gewichtung also jeweils $\frac{1}{n}$) gebildet, sondern wir gewichten jeden Summanden mit $\frac{1}{e_i}$ (mit dem zugehörigen i), da es Sinn hat, die Abweichungen in Relation zur Klassengröße zu betrachten.

Beispiel 5.5. Wir betrachten hier beispielhaft zwei Klassen und die jeweiligen Häufigkeiten. Es gelte

$$\begin{aligned}n_1 &= 5, \\e_1 &= 10, \\n_2 &= 205, \\e_2 &= 200.\end{aligned}$$

Bei der ersten und der zweiten Klasse ergibt sich jeweils 5 als Abweichung, doch bei der ersten Klasse muss sie als deutlich schwerwiegender aufgefasst werden, da die Abweichung 50% der erwarteten Anzahl entspricht. Die Gewichtungen $\frac{1}{e_1} = \frac{1}{10}$ und $\frac{1}{e_2} = \frac{1}{200}$ sind daher sinnvoll.

◇

Betrachtet man die Zufallsvariablen

$$N_i := |\{l : X_l \in K_i\}|,$$

so gilt unter P_0

$$(N_1, \dots, N_k) \sim M(n, \pi_1, \dots, \pi_k).$$

Des weiteren gilt asymptotisch

$$T(x) \sim \chi_{k-1-r}^2,$$

wobei r die Anzahl der zu schätzenden Parameter ist, die benötigt werden, um P_0 vollständig zu charakterisieren. Wird der Wert der Statistik T , also die Summe der gewichteten Abweichungen zu groß, lehnen wir die Nullhypothese, dass P_0 die wahre Verteilung ist, ab, genauer gesagt: Ist $T(x) \geq \chi_{k-1-r, 1-\alpha}^2$ bei einem Test zum Niveau $1 - \alpha$, entscheiden wir uns für $P \neq P_0$.

In R verwenden wir die Funktion `chisq.test` zur Durchführung eines χ^2 -Anpassungstests. Dabei gibt man folgendes ein:

```
chisq.test(x, correct=T, p=rep(1/length(x), length(x)), rescale.p=F, ...)      chisq.test
```

wobei x der Vektor der absoluten Häufigkeiten der Klassen bzw. Ausprägungen ist und p der Vektor der Vektor der Wahrscheinlichkeiten der Klassen unter P_0 . Standardmäßig wird jeder Klasse dieselbe Wahrscheinlichkeit zugeordnet, doch wir können natürlich jeden beliebigen Vektor an Wahrscheinlichkeiten hier einsetzen, insbesondere also auch (π_1, \dots, π_k) . Eine andere Möglichkeit besteht darin, den Vektor der erwarteten absoluten Häufigkeiten für p einzusetzen und mit `rescale.p=T` zu einem Wahrscheinlichkeitsvektor reskalieren zu lassen.

Beispiel 5.6. Ein Würfel soll auf Unverfälschtheit überprüft werden und wird dazu 1893-mal geworfen. Es ergeben sich folgende Häufigkeiten:

```
x<-c(347,289,321,309,330,297)
```

Die Nullhypothese ist, dass der Würfel unverfälscht ist, d.h. dass eine Laplace-Verteilung auf $\{1, \dots, 6\}$ vorliegt. Die Wahrscheinlichkeiten der einzelnen Augenzahlen sind unter dieser Verteilung alle ein Sechstel, die erwarteten absoluten Häufigkeiten sind jeweils $\frac{1893}{6} = 315.5$. Wir haben nun mehrere Möglichkeiten den Test durchzuführen:

```
chisq.test(x,p=rep(315.5,6),rescale.p=T)
chisq.test(x,p=rep(1/6,6))
chisq.test(x)
```

liefern alle dasselbe Ergebnis, nämlich

Chi-squared test for given probabilities

```
data: x
X-squared = 7.3518, df = 5, p-value = 0.1958
```

Der p-Wert zeigt, dass die Laplace-Verteilungsannahme zu allen Niveaus $1 - \alpha > 0.8042$ nicht abgelehnt werden kann.

◇

Das zweite Verfahren zur Überprüfung, ob eine gewisse Verteilung vorliegt, ist der **Kolmogorov-Smirnov-Test**, der auf dem Vergleich der empirischen Verteilungsfunktion mit der Verteilungsfunktion der vermuteten Verteilung beruht. Dazu muss allerdings vorausgesetzt werden, dass das Merkmal metrisch skaliert ist, um die entsprechenden Funktionen bilden und vergleichen zu können. Die theoretische Grundlage für den Vergleich ist der Satz von Glivenko/Cantelli, der besagt, dass die empirischen Verteilungsfunktionen bei gegen unendlich wachsender Stichprobenlänge fast sicher gleichmäßig gegen die Verteilungsfunktion konvergieren. Wir müssen hier zur Anwendung des Tests zusätzlich voraussetzen, dass die Verteilungsfunktion stetig ist, so dass theoretisch keine Bindungen auftreten. Das zugehörige Testproblem lautet hier:

$$H_0 : F = F_0 \quad \text{gegen} \quad H_1 : \exists t \in \mathbb{R} : F \neq F_0,$$

wobei F die zum untersuchten Merkmal gehörige unbekannte Verteilungsfunktion und F_0 die vermutete, stetige, vollständig spezifizierte Verteilungsfunktion sind. Wir betrachten die Prüfgröße

$$T_n(x) = \sqrt{n} \sup_{t \in \mathbb{R}} |F_0(t) - F_n(t|x)|$$

mit der empirischen Verteilungsfunktion $F_n(t|x)$ von $x = (x_1, \dots, x_n)$ an der Stelle t . Den Test führen wir in R mit der Funktion `ks.test` durch:

`ks.test`

```
ks.test(x,y,...,alternative=c("two.sided","less","greater"),exact=NULL)
```

Die Stichprobe wird als Argument `x` eingesetzt, als `y` eine Zeichenkette, die eine Verteilungsfunktion benennt (z.B. „pnorm“, wenn man überprüfen möchte, ob eine Normalverteilung gegeben ist). An die Stelle der `...` werden die Parameter eingesetzt, die die in `y` genannte Verteilung vollständig bestimmen (z.B. `mean` und `sd` für die Normalverteilung).

Beispiel 5.7. Wir betrachten wieder den Datensatz `iris`, insbesondere das Merkmal `Sepal.Length` der Spezies `versicolor`. Durch die grafische Auswertung dieser Daten liegt nahe, dass es sich hier um eine Normalverteilung mit Mittelwert bei 6 und geringer Streuung handelt. Wir schätzen also den Mittelwert und die Varianz durch das Stichprobenmittel und die Stichprobenvarianz und wählen diese als Parameter der mutmaßlich vorliegenden Normalverteilung:

```
mean(Sepal.Length[Species=="versicolor"])
[1] 5.936
var(Sepal.Length[Species=="versicolor"])
[1] 0.2664327
ks.test(Sepal.Length[Species=="versicolor"],pnorm,mean=5.936,
        sd=sqrt(0.2664327))
```

One-sample Kolmogorov-Smirnov test

```
data: Sepal.Length[Species == „versicolor“]
D = 0.0962, p-value = 0.7434
alternative hypothesis: two-sided
```

Warning message:

```
In ks.test(Sepal.Length[Species == „versicolor“], pnorm, mean = 5.936, :
cannot compute correct p-values with ties
```

Das Ergebnis ist also, dass man die Nullhypothese, dass eine Normalverteilung mit Mittelwert 5.936 und Varianz ungefähr 0.27 vorliegt, zu den meisten üblichen Niveaus nicht ablehnen kann. Zudem erhalten wir eine Warnung, dass in der Stichprobe Bindungen vorliegen und daher der p-Wert ungenau sein kann.

◇

Hierzu sind noch ein paar Anmerkungen zu machen: Die Teststatistik ist auch bei auftretenden Bindungen noch wohldefiniert, kann allerdings die Auswertung verfälschen, da wir eigentlich von einer stetigen Verteilungsfunktion ausgehen, bei der Bindungen nicht auftreten sollten. Ist die hypothetische Verteilungsfunktion nicht-stetig, ist der Kolmogorov-Smirnov-Test konservativ, d.h. er hält länger als geboten an der Nullhypothese fest. Müssen (wie im Beispiel) Parameter zur Spezifizierung aus den Daten

geschätzt werden, ist der Test im Allgemeinen auch konservativ. Testet man auf Normalverteilung wird daher häufig eine entsprechende Korrektur, die Korrektur nach **Liliefors** durchgeführt.

5.3 Der Zweistichprobenfall

Im Wesentlichen gibt es beim Zweistichprobenproblem zwei Fälle zu unterscheiden: Wir erheben ein Merkmal in zwei unabhängigen Stichproben, wobei sich die Randbedingungen der beiden Stichproben nur in einer Bedingung unterscheiden, d.h. das Merkmal wird unter zwei verschiedenen Bedingungen untersucht. Wir können aber auch zwei Erhebungen am selben Merkmalsträger vornehmen, wodurch wir zwei verbundene einfache Stichproben erhalten. In diesem Fall reden wir von **gepaarten** Stichproben, wobei sich um Erhebungen zu zwei verschiedenen Merkmalen oder zu einem Merkmal unter zwei verschiedenen Bedingungen handeln kann.

Wie auch im Einstichprobenfall können wir einen t-Test anwenden, wobei hier zusätzlich zwischen den Fällen unabhängiger und verbundener Stichproben unterschieden werden muss. Beginnen wir mit dem Fall zweier unabhängiger Stichproben $(X_{1,1}, \dots, X_{1,n_1})$, $(X_{2,1}, \dots, X_{2,n_2})$. Auch hier müssen wir wieder voraussetzen, dass Normalverteilungen vorliegen, dass also für alle $1 \leq i \leq n_1$ bzw. n_2 gilt:

$$\begin{aligned} X_{1,i} &\sim \mathcal{N}(\mu_1, \sigma_1^2), \\ X_{2,i} &\sim \mathcal{N}(\mu_2, \sigma_2^2). \end{aligned}$$

Eine alternative Voraussetzung wäre wieder, dass die $X_{i,j}$, $1 \leq j \leq n_i$, iid mit beliebiger Verteilung sind und geeigneten Momentenbedingungen genügen, so dass der zentrale Grenzwertsatz anwendbar ist, und dass $n_i \geq 30$ gilt.

Beim Zweistichprobenproblem vergleichen wir die Verteilungen der beiden gegebenen Stichproben miteinander, indem wir als Parameter ϑ die Differenz der beiden unbekannt Mittelwerte $\mu_1 - \mu_2$ betrachten. Die Testprobleme beschäftigen sich folglich mit der Frage, was die wahre Differenz ϑ_0 zwischen den Mittelwerten ist:

- a) $H_0: \vartheta_0 = \mu_0$ gegen $H_1: \vartheta_0 \neq \mu_0$,
- b) $H_0: \vartheta_0 \leq \mu_0$ gegen $H_1: \vartheta_0 > \mu_0$,
- c) $H_0: \vartheta_0 \geq \mu_0$ gegen $H_1: \vartheta_0 < \mu_0$.

Die Prüfgröße sieht hier folgendermaßen aus:

$$T(x_1, x_2) = \frac{\hat{\vartheta}(x_1, x_2) - \vartheta_0}{\hat{\sigma}_{\hat{\vartheta}(x_1, x_2)}}$$

mit

$$\hat{\vartheta}(x_1, x_2) = \bar{x}_1 - \bar{x}_2, \quad \bar{x}_i := \frac{1}{n_i} \sum_{j=1}^{n_i} x_{ij}$$

und einem Schätzer $\hat{\sigma}_{\hat{\vartheta}(x_1, x_2)}$ für die Standardabweichung des Schätzers $\hat{\vartheta}(x_1, x_2)$. Hier müssen wir zwischen dem homoskedastischen Fall, d.h. $\sigma_1 = \sigma_2$, und dem heteroskedastischen Fall, d.h. $\sigma_1 \neq \sigma_2$, unterscheiden. Im homoskedastischen Fall gilt

$$\hat{\sigma}_{\hat{\vartheta}(x_1, x_2)} = \sqrt{\left(\frac{1}{n_1} + \frac{1}{n_2}\right) \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}}$$

mit

$$s_i^2 = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_i)^2.$$

Im heteroskedastischen Fall gilt

$$\hat{\sigma}_{\hat{\vartheta}(x_1, x_2)} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}},$$

allerdings besitzt der durch die Prüfgröße T angegebene Test keine gleichmäßigen Optimalitätseigenschaften (Behrens-Fisher-Problem). Die Schätzer $\hat{\sigma}_{\hat{\vartheta}(x_1, x_2)}$ des homoskedastischen und des heteroskedastischen Falles stimmen übrigens überein, wenn $n_1 = n_2$ gilt. Im homoskedastischen Fall besitzt die Prüfgröße $T(x_1, x_2)$ eine $t_{n_1+n_2-2}$ -Verteilung, im heteroskedastischen Fall eine t_k -Verteilung mit

$$k = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{1}{n_1-1} \left(\frac{s_1^2}{n_1}\right)^2 + \frac{1}{n_2-1} \left(\frac{s_2^2}{n_2}\right)^2}.$$

Liegen uns zwei verbundene Stichproben $(X_1, Y_1), \dots, (X_n, Y_n)$ statt unabhängiger Stichproben vor, können wir auch einen t-Test durchführen, wofür sich die Teststatistik und ihre Verteilung allerdings ändern. Voraussetzung ist auch hier das Vorliegen von Normalverteilungen:

$$\begin{aligned} X_i &\sim \mathcal{N}(\mu_x, \sigma_x^2), \\ Y_i &\sim \mathcal{N}(\mu_y, \sigma_y^2) \end{aligned}$$

mit Mittelwerten $\mu_x, \mu_y \in \mathbb{R}$ und Varianzen $\sigma_x^2, \sigma_y^2 > 0$. Wie im Fall unverbundener Stichproben wird auch hier die Differenz der Mittelwerte $\vartheta = \mu_x - \mu_y$ betrachtet. Als erwartungstreuen Schätzer für die Differenz der beiden Stichproben verwendet man

$$\hat{\vartheta}(x, y) = \bar{x} - \bar{y}$$

mit den Stichprobenmitteln \bar{x} und \bar{y} . Zur Schätzung der Streuung von $\hat{\vartheta}(x, y)$ benutzt man

$$\hat{\sigma}_{\hat{\vartheta}(x, y)} = s_{x-y} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n ((x_i - y_i) - (\bar{x} - \bar{y}))^2},$$

also die Wurzel aus der Stichprobenvarianz von $(x_1 - y_1, \dots, x_n - y_n)$, die ein erwartungstreuer Schätzer für die Standardabweichung dieser einfachen Stichprobe ist. Die Prüfgröße T ist nun unter H_0 t_{n-1} -verteilt.

Beide t-Tests dienen dem Vergleich der Verteilungen von zwei Stichproben. Bei unabhängigen Stichproben wird untersucht, ob die gleiche Verteilung vorliegen könnte bzw. um wieviel die Verteilungen verschoben sein könnten, indem die Mittelwerte miteinander verglichen werden. Dazu verwendet man

```
t.test(x,y,alternative=c("two.sided","less","greater"),mu=0,paired=FALSE,
      var.equal=FALSE,conf.level=0.95,...)
```

wobei `paired` angibt, ob es sich um gepaarte Stichproben handelt (StandardEinstellung: `FALSE`), und mit `var.equal` eingegeben werden kann, ob der homoskedastische oder heteroskedastische Fall betrachtet werden soll (Standard: heteroskedastisch). Bei gepaarten Stichproben geht man genauso vor, setzt dann aber `paired` natürlich auf `TRUE`. Hier kann man auch so vorgehen, dass man die Differenz der beiden Stichproben in einen Einstichproben-t-Test einsetzt; beide Vorgehensweisen führen zum selben Ergebnis.

Beispiel 5.8. Bei dem Datensatz `iris` kann man die Verteilungen der beiden Merkmale `Sepal.Length` und `Sepal.Width` auf ihren Mittelwert untersuchen bzw. für eines der beiden Merkmale die Mittelwerte von zwei verschiedenen Spezies vergleichen. Betrachtet man `Sepal.Length` für die Spezies `setosa` und `versicolor`, liegen unverbundene Stichproben vor (wovon man sich mittels eines Streudiagrammes überzeugen kann). Schaut man sich die Histogramme der beiden anscheinend normalverteilten Größen an, sieht man, dass der Mittelwert bei der Spezies `setosa` wohl kleiner sein wird als der von `versicolor`. Entsprechend gestaltet sich dann die Gegenhypothese beim t-Test:

```
attach(iris)
sl<-Sepal.Length
sw<-Sepal.Width
sp<-Species
detach(iris)
t.test(sl[sp=="setosa"],sl[sp=="versicolor"],alternative="less")
```

Hier ergibt sich ein sehr kleiner p-Wert; zu allen üblichen Niveaus (90%-99%) wird die Hypothese abgelehnt. Wir können auch auf den Unterschied zwischen den Mittelwerten testen. Eine mögliche Fragestellung wäre hier, ob der Mittelwert bei der Spezies `setosa` um mehr eins kleiner ist als bei `versicolor`. Der entsprechende Test ist dann:

```
t.test(sl[sp=="setosa"],sl[sp=="versicolor"],mu=-1,alternative="less")
```

Nun stellen wir einen Mittelwertvergleich für die Merkmale `Sepal.Length` und `Sepal.Width` bei der Spezies `setosa` an. Da die beiden Merkmale jeweils am selben Merkmalsträger erhoben wurden, handelt es sich hier um eine gepaarte Stichprobe. Grafische Methoden

oder auch die Bildung des Stichprobenmittels zeigen, dass der Mittelwert bei `Sepal.Width` um ca. 1.5 kleiner sein dürfte als der von `Sepal.Length`. Wir könnten also (zum Niveau 95%) testen, ob dies die tatsächliche Differenz ist:

```
t.test(sl[sp=="setosa"],sw[sp=="setosa"],mu=1.5,paired=T)
```

Der p-Wert ist 0.04164, wir entscheiden uns demnach für die Gegenhypothese, dass die Differenz nicht gleich 1.5 ist.

◇

Ist die Annahme einer Normalverteilung nicht gerechtfertigt, kann man auch im Zweistichprobenproblem Tests auf den Median durchführen und diesen als Lageparameter für die Verteilung betrachten. Medianvergleiche können bei zwei gepaarten Stichproben angestellt werden, wozu man die Differenzen $D_i = x_i - y_i$ der Stichproben bildet und für die dadurch entstehende Stichprobe D den Vorzeichen- oder den Wilcoxon-Vorzeichen-Rangtest ausführt. Die Hypothesen werden dabei für den Median der Stichprobe D , also die Differenz der Mediane der Einzelstichproben, formuliert, die Ausführung ist dann genau so, wie im Einstichprobenfall vorgestellt.

Wenn zwei verbundene quantitative Stichproben gegeben sind, können sie auf Zusammenhänge untersucht werden. Hier gibt es zwei Tests, die auf den Korrelationskoeffizienten von Pearson und Spearman beruhen. Wenn man davon ausgehen kann, dass eine 2-dimensionale Normalverteilung zugrundeliegt, wenn also

$$(X_i, Y_i) \sim \mathcal{N}_2(\mu, \Sigma)$$

für jedes $1 \leq i \leq n$ mit Mittelwertvektor μ und Kovarianzmatrix Σ gilt, kann der **Pearson-Korrelationstest** zur Zusammenhangsanalyse verwendet werden. Dabei gestalten sich die Testprobleme um die (lineare) Korrelation $\rho_{x,y}$ zwischen den Stichproben x und y :

- a) H_0 : x und y sind unkorreliert, d.h. $\rho_{x,y} = 0$ gegen H_1 : x und y sind korreliert, d.h. $\rho_{x,y} \neq 0$.
- b) H_0 : x und y sind nicht-negativ korreliert, d.h. $\rho_{x,y} \geq 0$ gegen H_1 : x und y sind negativ korreliert, d.h. $\rho_{x,y} < 0$.
- c) H_0 : x und y sind nicht-positiv korreliert, d.h. $\rho_{x,y} \leq 0$ gegen H_1 : x und y sind positiv korreliert, d.h. $\rho_{x,y} > 0$.

Als Schätzer für die Korrelation wird der Korrelationskoeffizient von (Bravais-)Pearson $r_{x,y}$ verwendet; die Teststatistik ergibt sich als Quotient aus $r_{x,y}$ und einem erwartungstreuen Schätzer für dessen Standardabweichung

$$T(x, y) = \frac{r_{x,y}}{\sqrt{\frac{1-r_{x,y}^2}{n-2}}}$$

und ist unter H_0 t_{n-2} -verteilt.

Bevor man diesen Test anwenden kann, muss überprüft werden, ob die Normalverteilungsannahme gerechtfertigt ist. Zeigt sich mittels grafischer Methoden, dass beide Stichproben normalverteilt sind (mit eventuell unterschiedlichen Parametern), ist die Annahme begründet. Anderenfalls muss man auf andere Tests, z.B. den **Spearman-Rang-Korrelationstest** zurückgreifen. Bei diesem Test muss vorausgesetzt werden, dass die Verteilungen der X_i und Y_i stetig sind, damit theoretisch keine Bindungen auftreten, die zu Durchschnittsrängen und infolgedessen zu einer verfälschten Teststatistik führen würden, die auf dem Rang-Korrelationskoeffizienten von Spearman r_S aufbaut. Die Testprobleme sind die gleichen wie beim Pearson-Korrelationstests, nur mit dem Unterschied, dass hier nicht speziell auf linearen, sondern allgemein auf monotonen Zusammenhang getestet wird. Als Teststatistik wird für Stichproben mit einer Länge $n \leq 10$ einfach die Summe der quadrierten Rangdifferenzen

$$D^2 = \sum_{i=1}^n (R(x_i) - R(y_i))^2$$

verwendet, die verteilungsfrei ist. Bei größeren Stichproben wird die Teststatistik

$$T(x, y) = \frac{r_S}{\sqrt{\frac{1-r_S^2}{n-2}}}$$

Für hinreichend großes n gilt approximativ $T \sim t_{n-2}$.

Beide Tests führt man in R mit der Funktion `cor.test` aus:

`cor.test`

```
cor.test(x, y, alternative=c("two.sided", "less", "greater"),
        method=c("pearson", "kendall", "spearman"), ...)
```

Je nachdem, welchen Test wir ausführen wollen, wählen wir für `method` die passende Zeichenkette aus, also `method="spearman"` für den Rang-Korrelationstest von Spearman. Die Standardeinstellung ist `"pearson"`.

Beispiel 5.9. Wollen wir einen t-Test durchführen und haben uns schon vom Vorliegen von Normalverteilungen überzeugt, können wir mithilfe des Pearson-Korrelationstests entscheiden, ob der t-Test besser für gepaarte oder unverbundene Stichproben durchgeführt werden sollte. Wir greifen auf das vorhergehende Beispiel zurück und überprüfen jetzt mit dem Korrelationstest, ob unsere Überlegungen zur Abhängigkeit der Stichproben in den verschiedenen Fällen richtig waren.

```
cor.test(sl[sp=="setosa"], sl[sp=="versicolor"])
```

Der p-Wert ist hier 0.5767; die Nullhypothese der Unabhängigkeit kann also zu den gängigen Niveaus nicht abgelehnt werden. Dies deckt sich mit der Überlegung im vorigen Beispiel, dass die Stichproben unabhängig sind.

```
cor.test(sl[sp=="setosa"],sw[sp=="setosa"])
```

Der p-Wert ist mit $6.71e-10$ sehr klein. Hier entscheiden wir uns also zu praktisch jedem Niveau für die Gegenhypothese der Abhängigkeit. Auch dies deckt sich mit den Überlegungen im Beispiel zum t-Test. Hier ist allerdings zu beachten, dass die (lineare) Korrelation nicht dasselbe ist wie (Un-)Abhängigkeit.

◇

Wollen wir zwei Merkmale auf Unabhängigkeit untersuchen, können wir den χ^2 -Unabhängigkeitstest für gepaarte Stichproben benutzen. Die Hypothesen sind also

$$H_0 : X, Y \text{ unabhängig} \quad \text{gegen} \quad H_1 : X, Y \text{ abhängig.}$$

Man unterteilt nun die n Beobachtungspaare (x_i, y_i) in $k \cdot l$ disjunkte Klassen $A_i \times B_j$, $1 \leq i \leq k$, $1 \leq j \leq l$, und vergleichen dann die tatsächlichen absoluten Häufigkeiten der Klassen mit denen, die bei Unabhängigkeit von X und Y zu erwarten wären.

$X \setminus Y$	B_1	B_2	\dots	B_l	\sum
A_1	n_{11}	n_{12}	\dots	n_{1l}	$n_{1.}$
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
A_k	n_{k1}	n_{k2}	\dots	n_{kl}	$n_{k.}$
\sum	$n_{.1}$	$n_{.2}$	\dots	$n_{.l}$	n

Dabei ist n_{ij} die Anzahl der Stichprobenpaare in der Klasse $A_i \times B_j$. Wir definieren nun die Wahrscheinlichkeit, in einer bestimmten Klasse $A_i \times B_j$ zu landen:

$$\pi_{ij} = P(X \in A_i, Y \in B_j).$$

Auch hier (wie beim χ^2 -Anpassungstest) müssen gewisse Bedingungen durch die erwarteten Anzahlen erfüllt werden, damit eine solide Grundlage für Schlüsse gegeben ist: Für jede Klasse $A_i \times B_j$ sollte die erwartete Häufigkeit $e_{ij} := n\pi_{ij}$ größer oder gleich eins sein und für mindestens 80% der Klassen sollten mindestens fünf Treffer zu erwarten sein. Wenn wir jetzt die Randwahrscheinlichkeiten

$$\begin{aligned} \pi_{i.} &:= P(X \in A_i), \\ \pi_{.j} &:= P(Y \in B_j) \end{aligned}$$

betrachten, können die Hypothesen folgendermaßen formuliert werden:

$$H_0 : \forall i, j : \pi_{ij} = \pi_{i.} \pi_{.j} \quad \text{gegen} \quad H_1 : \exists i, j : \pi_{ij} \neq \pi_{i.} \pi_{.j}$$

Wir können die Parameter $\pi_{i.}$ und $\pi_{.j}$ nur schätzen, und zwar verwenden wir dazu die Randhäufigkeiten $n_{i.}$ und $n_{.j}$ und dividieren diese durch die Gesamtanzahl n . Die zu

erwartenden Anzahlen haben wir bereits über π_{ij} ausgedrückt. Jetzt können wir die Nullhypothese und die soeben angesprochene Schätzung verwenden und erhalten

$$\begin{aligned} e_{ij} &= n\pi_{ij} = n\pi_{i.}\pi_{.j} \\ &= n \frac{n_{i.}}{n} \frac{n_{.j}}{n} = \frac{n_{i.}n_{.j}}{n} \end{aligned}$$

Wie schon beim χ^2 -Anpassungstest wird nun als Prüfgröße für den Test die Summe der quadrierten Abweichungen der tatsächlichen von den erwarteten Häufigkeiten, gewichtet durch die erwartete Häufigkeit, gebildet:

$$T(x, y) = \sum_{i=1}^k \sum_{j=1}^l \frac{(n_{ij} - e_{ij})^2}{e_{ij}}.$$

Unter H_0 besitzt diese Statistik eine χ^2 -Verteilung mit $(k-1) \cdot (l-1) = kl - 1 - (k-1 + l-1)$ Freiheitsgraden, wobei $k-1+l-1$ Freiheitsgrade durch das Schätzen der Parameter $\pi_{i.}$ bzw. $\pi_{.j}$ verloren gehen.

Die Funktion in R, mit der dieser test durchgeführt wird, ist dieselbe wie bei χ^2 -Anpassungstest, nämlich `chisq.test`. In diesem Fall wird jedoch nicht ein einzelner Vektor x eingesetzt, sondern stattdessen eine Kontingenztabelle oder zwei gleichlange Vektoren x und y , aus denen dann eine Kontingenztabelle berechnet wird.

Beispiel 5.10. Wir greifen nochmal auf das `iris`-Beispiel zurück und benutzen den χ^2 -Unabhängigkeitstest, um zu überprüfen, ob verbundene bzw. unverbundene Stichproben vorliegen.

```
chisq.test(sl[sp=="setosa"],sl[sp=="versicolor"])
chisq.test(sl[sp=="setosa"],sw[sp=="setosa"])
```

Hier ergibt sich im Wesentlichen das gleiche Ergebnis wie bei den Pearson-Korrelationstests, allerdings mit leicht verschiedenen p-Werten. Das liegt daran, dass der Korrelationstest von Pearson für Normalverteilungen genauer ist als der χ^2 -Unabhängigkeitstest und nur auf Korrelation und eben nicht auf Unabhängigkeit untersucht. Letzterer ist allerdings auf alle Verteilungsarten (und nicht nur Normalverteilungen) anwendbar.

◇

Beispiel 5.11. Wir wollen beim Datensatz `VADeaths` untersuchen, ob die beiden Merkmale `Sterbealter` und `Bevölkerungsgruppe` (`Rural Male`, `Urban Male`, `Rural Female`, `Urban Female`) unabhängig voneinander sind. Dabei liegt der Datensatz bereits in Form einer Kontingenztabelle vor, so dass man ihn direkt in den χ^2 -Unabhängigkeitstest einsetzen kann:

```
chisq.test(VADeaths)
p-value = 0.996
```

Die Nullhypothese der Unabhängigkeit kann also nicht abgelehnt werden.

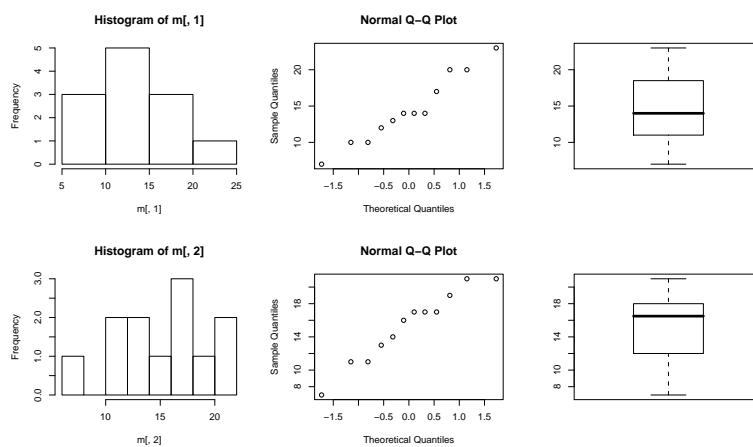
◇

Beispiel 5.12. Im Datensatz `InsectSprays` sind die Daten zu Anwendungen von sechs verschiedenen Insektenschutzmitteln enthalten. Dabei wurden auf Testfeldern bzw. bestimmten Parzellen die dort angebaute Pflanzen mit jeweils einem der Insektenschutzmittel behandelt (pro Mittel zwölf Anbauflächen) und danach wurde gezählt, wie viele Insekten sich trotz der Mittel in den Feldern aufhielten. Wir interessieren uns nun für die Frage, ob Mittel A signifikant besser ist als Mittel B.

Um überhaupt angemessene Tests auswählen zu können betrachten wir die Verteilungen der beiden Mittel:

```
par(mfrow=c(2,3))
attach(InsectSprays)
m<-matrix(count,ncol=6)
A<-m[,1]
B<-m[,2]
hist(A)
qqnorm(A)
boxplot(A)
hist(B)
qqnorm(B)
boxplot(B)
```

Die Grafiken (zumindest die Histogramme und QQ-Plots) lassen eine Normalverteilungs-

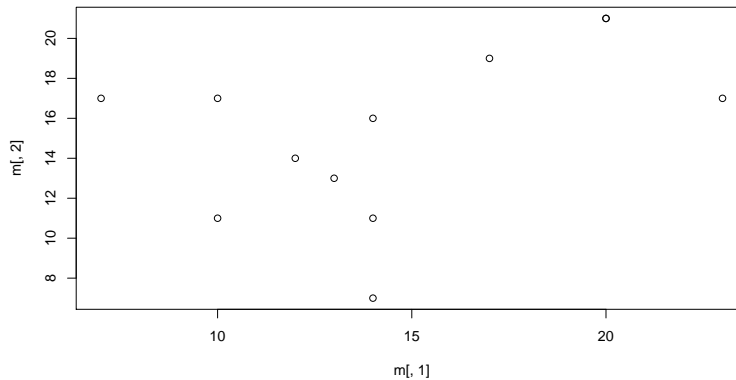


annahme gerechtfertigt scheinen, die leicht verzerrten Boxplots können auf den geringen Stichprobenumfang zurückgeführt werden. Bei einer Normalverteilungsannahme können wir nun die Mittelwerte der Verteilungen von A und B miteinander vergleichen, indem wir einen t-Test durchführen. Dazu müssen wir uns allerdings zunächst vergewissern, ob es sich hier um verbundene oder unverbundene Stichproben handelt. Da gibt es nun mehrere Möglichkeiten: wir können ein Streudiagramm erstellen, einen Korrelationstest durchführen und den χ^2 -Unabhängigkeitstest anwenden. Zur Sicherheit führen wir alle

drei Methoden durch:

```
plot(A,B)
cor.test(A,B)
p-value = 0.1503
chisq.test(A,B)
p-value = 0.1551
```

Im Streudiagramm lässt sich kein Muster erkennen und der Pearson-Korrelationstest



(Normalverteilungen!) und der χ^2 -Unabhängigkeitstest liefern beide einen p-Wert von ca. 15%, was dafür spricht, die Nullhypothese der Unabhängigkeit (zu den gängigen Niveaus) nicht abzulehnen. Wir werden also einen t-Test für unverbundene Stichproben durchführen. dazu brauchen wir allerdings das konkrete Testproblem. Wenn A signifikant besser sein soll als B, müssen wir dies gerade als Gegenhypothese formulieren, da nur diese signifikant sein kann. In diesem Fall ist es so, dass ein Mittel besser ist als ein anderes, wenn nach der Anwendung weniger Insekten auf dem Feld zu finden sind. Folglich ist das Testproblem

$$H_0 : \mu_A \geq \mu_B \quad \text{gegen} \quad H_1 : \mu_A < \mu_B$$

mit den unbekanntem Erwartungswerten μ_A und μ_B der beiden Stichproben. Der t-Test ist also

```
t.test(A,B,alternative="less")
p-value = 0.3273
```

Das Insektenschutzmittel A ist also nicht signifikant besser als das Mittel B. (Bei dem Befehl in R muss nicht speziell eingegeben werden, dass ungepaarte Stichproben vorliegen, da dies der Standardeinstellung entspricht.)

◇

Im Zweistichprobenfall stellt sich natürlich auch die Frage, ob beide Stichproben aus derselben Verteilung hervorgegangen sein könnten. Eine Möglichkeit zur Überprüfung wäre die Anwendung des Kolmogorov-Smirnov-Tests, der bei zwei gegebenen Stichproben die zugehörigen empirischen Verteilungsfunktionen miteinander vergleicht. Das zugrundeliegende Prinzip ist hier auch wieder eine Anwendung des Satzes von Glivenko-Cantelli. Dabei wird vorausgesetzt, dass beide untersuchten Merkmale metrisches Messniveau besitzen und die $X_{i,1}, \dots, X_{i,n_i}$, $i = 1, 2$, unabhängig und identisch verteilt sind wie X_i , $i = 1, 2$, wobei die unbekanntes Verteilungsfunktionen F von X_1 und G von X_2 stetig sein sollen. (Wegen der Stetigkeit treten theoretisch keine Bindungen auf, durch die der Wert der Teststatistik verfälscht würde.) Das Testproblem gestaltet sich hier also zu

$$H_0 : F = G \quad \text{gegen} \quad H_1 : \exists t \in \mathbb{R} : F(t) \neq G(t)$$

bzw. entsprechend für die einseitigen Fälle. Die Prüfgröße ist hier

$$K_{n_1, n_2}(x_1, x_2) = \sup_{t \in \mathbb{R}} |F_{n_1, x_1}(t) - G_{n_2, x_2}(t)|$$

mit den empirischen Verteilungsfunktionen F_{n_1, x_1} der Stichprobe x_1 und G_{n_2, x_2} von x_2 . Wie oben schon erwähnt, setzen wir Stetigkeit der Verteilungsfunktionen voraus, um Bindungen zu vermeiden. Die Teststatistik bleibt bei Bindungen zwar auch wohldefiniert, doch der Test wird dann sehr konservativ, entscheidet also in weniger Fällen als angebracht für die Gegenhypothese. Zur Anwendung des Tests in R benutzt man wie im Einstichprobenfall die Funktion `ks.test`, in die nun beide Stichproben als Argumente eingesetzt werden:

```
ks.test(x, y, alternative=c("two.sided", "less", "greater"), exact = NULL)
```

Ein anderer Test zum Vergleich der Verteilungsfunktionen ist der **Wilcoxon-Rangsummen-Test** bzw. der **U-Test von Mann-Whitney**. Dabei bildet man die Ränge der Beobachtungen bezüglich der Gesamtstichprobe, also die Ränge von

$$(x_{1,1}, \dots, x_{1,n_1}, x_{2,1}, \dots, x_{2,n_2}).$$

Gehen beide Stichproben auf dieselbe Verteilung zurück, sollte die Summe der Ränge von x_1 ca. den Anteil $\frac{n_1}{n_1+n_2}$ der Summe aller Ränge ausmachen. Da bei Bindungen Durchschnittsränge gebildet werden, kann die Rangsumme von x_1 in einem solchen Fall einen falschen Eindruck vermitteln. Um dies zu vermeiden, setzen wir voraus, dass die $X_{i,1}, \dots, X_{i,n_i}$ unabhängig und identisch verteilt sind wie X_i mit stetigen Verteilungsfunktionen F_i , $i = 1, 2$. Die Prüfgröße des Wilcoxon-Rangsummen-Tests ist also

$$W(x_1, x_2) = \sum_{i=1}^{n_1} R(x_{1,i}).$$

Man kann jedoch auch die Statistik

$$U(x_1, x_2) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \mathbf{1}_{(x_{2,j}, \infty)}(x_{1,i})$$

betrachten, die zum U-Test von Mann-Whitney führt. Zwischen beiden Statistiken gilt folgende Beziehung:

$$U = W - \frac{n_1(n_1 + 1)}{2}.$$

Beide ergeben also im Prinzip denselben Test. Außerdem hängen beide Statistiken nicht von den Verteilungsfunktionen F_1 und F_2 ab, sind also verteilungsfrei. Wenn die beiden Verteilungsfunktionen durch Verschiebungen auseinander hervorgehen, wenn also eine Translationsfamilie vorliegt, besitzen diese Tests bestmögliche Güte. Anderenfalls sind sie immerhin noch unverfälscht, im Allgemeinen aber zu konservativ. Die Durchführung in R läuft über die Funktion `wilcox.test`, in die beide Stichproben eingesetzt werden:

```
wilcox.test(x,y,alternative = c("two.sided","less","greater"),mu = 0,
           paired = FALSE, exact = NULL, correct = TRUE,conf.int = FALSE,
           conf.level = 0.95, ...)
```

5.4 Der k -Stichprobenfall ($k \geq 3$)

Bei k Stichproben kann man natürlich die Einzelstichproben oder auch je zwei Stichproben mit den bereits vorgestellten Verfahren auf bestimmte Maßzahlen, Zusammenhänge, Unabhängigkeit und Verteilungen untersuchen. Üblicherweise führt man hier jedoch zunächst **globale Tests** auf diese Kenngrößen und Beziehungen durch, die alle k Stichproben einbeziehen und auf Unterschiede im Hinblick auf diese Kriterien untersuchen. Hat ein solcher globaler Test signifikante Unterschiede zwischen den Stichproben aufgedeckt, kann man danach sogenannte **post hoc multiple Tests** durchführen, die dann aufzeigen, zwischen welchen Stichproben die Unterschiede bestehen. Häufige Test- und Analyseverfahren sind im k -Stichprobenfall solche, die auf der Zerlegung der Stichprobenvarianzen beruhen und daher Varianzanalysen genannt werden. Diese werden wir im nächsten Kapitel gesondert behandeln.

Der erste globale Test, den wir uns anschauen werden (und der auch in den Bereich der Varianzanalyse fällt), ist der **Test auf Varianzheterogenität von Levene** oder kurz der **Levene-Test**, mit dem k unabhängige Stichproben auf Gleichheit der Varianzen überprüft werden. Dies ist insbesondere dann von Interesse, wenn wir Tests anwenden wollen, die für den heteroskedastischen und den homoskedastischen Fall unterschiedliche Ergebnisse liefern bzw. nur in einem der beiden Fälle vorgenommen werden können. (Bei der Varianzanalyse setzt man z.B. Homoskedastizität voraus.)

Beim Levene-Test gehen wir wie üblich davon aus, dass die einzelnen Stichproben Realisierungen von unabhängigen und identisch verteilten Zufallsgrößen sind, also $X_{i,1}, \dots, X_{i,n_i}$ iid wie X_i verteilt mit Varianz σ_i^2 für alle $1 \leq i \leq k$. Weiter setzen wir voraus, dass jede Stichprobe mindestens die Länge 10 hat. Die Hypothesen sind hier:

$$H_0 : \sigma_1^2 = \dots = \sigma_k^2 = \sigma^2 \quad \text{gegen} \quad H_1 : \exists i, j : \sigma_i^2 \neq \sigma_j^2$$

Die Idee bei diesem Test ist, dass man sich anschaut, wie stark die Streuung zwischen den einzelnen Stichproben ist, da dies ein erwartungstreuer Schätzer für die Varianz ist,

wenn die Varianzen alle gleich sind, und dies aber durch die Streuung innerhalb der Stichproben relativiert. Sind die Varianzen groß, kann auch leicht eine größere Streuung zwischen den Stichproben entstehen, sind sie jedoch klein, sollten auch die Unterschiede zwischen einzelnen Stichproben auch nicht besonders groß werden. Um überhaupt die Streuungen der einzelnen Stichproben miteinander vergleichen zu können, ziehen wir von den Stichproben die (getrimmten) Mittelwerte oder die Mediane ab und erhalten die modifizierten Beobachtungen $y_{ij} = |x_{ij} - \tilde{x}_i|$ mit den (getrimmten) Mittelwerten bzw. Medianen \tilde{x}_i . Die einzelnen oben angesprochenen Streuungen erhalten wir, indem wir die Gesamtstichprobenvarianz zerlegen:

$$SQT = SQE + SQR$$

mit :

$$SQT = \sum_{i=1}^k \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_{\bullet\bullet})^2, \text{ (sum of squares total)}$$

$$SQE = \sum_{i=1}^k n_i (\bar{y}_{i\bullet} - \bar{y}_{\bullet\bullet})^2, \text{ (sum of squares explained)}$$

$$SQR = \sum_{i=1}^k (n_i - 1) s_i^2, \text{ (sum of squares residual)}$$

wobei $\bar{y}_{i\bullet}$ und s_i^2 das Stichprobenmittel bzw. die Stichprobenvarianz der i -ten Stichprobe bezeichnen und $\bar{y}_{\bullet\bullet}$ das Gesamtmittel. SQE gibt dabei die Streuung zwischen den Stichproben an, SQR die innerhalb der Stichproben. Weiter ist $\frac{SQR}{n-k}$ ($n = \sum_{i=1}^k n_i$) unter H_0 ein erwartungstreuer Schätzer für die Varianzen ebenso wie $\frac{SQE}{k-1}$. Außerdem gilt bei Vorliegen von H_0 :

$$\frac{SQR}{n-k} \sim \chi_{n-k}^2,$$

$$\frac{SQE}{k-1} \sim \chi_{k-1}^2.$$

Daher besitzt die Prüfgröße

$$F = \frac{\frac{1}{k-1} SQE}{\frac{1}{n-k} SQR}$$

unter H_0 eine $F_{k-1, n-k}$ -Verteilung mit $k-1$ und $n-k$ Freiheitsgraden. Wird die Streuung zwischen den Stichproben im Vergleich zu den Streuungen innerhalb der Stichproben zu groß, lehnen wir also die Nullhypothese der Homoskedastizität ab. Den Test können wir in R mit der Funktion `levene.test` aus dem Paket `lawstat` durchführen:

`levene.test`

```
levene.test(y,group,option=c("mean","median","trim.mean"),
            trim.alpha = 0.25)
```

Das erste Argument `y` ist der Datensatz als Vektor (aus einer Tabelle zum Beispiel leicht mit der Funktion `as.vector` zu erstellen), das zweite Argument ist ein Faktor,

also ein Vektor eines qualitativen Merkmals, der zu jedem Element aus y angibt, welcher Stichprobe es angehört. Mit dem Argument `option` kann angegeben werden, wie die Stichprobe modifiziert werden soll: Mithilfe des Mittelwertes, was die klassische Methode wäre, des Medians oder eines getrimmten Mittelwertes, wofür das Argument `trim.alpha` wichtig wird. Die beiden letzten Methoden liefern einen robusten Test.

Beispiel 5.13. Wir können bei dem Datensatz `iris` überprüfen, ob die Varianzen der durch die Spezies vorgegebenen drei Stichproben für die einzelnen Blattkenngrößen dieselben Varianzen besitzen. Wir schauen uns zunächst einmal die Varianzen der Kenngrößen unterteilt nach Spezies an, um einen ersten Eindruck zu gewinnen:

```
sapply(split(Sepal.Length,Species),var)
  setosa versicolor virginica
0.1242490  0.2664327  0.4043429

sapply(split(Sepal.Width,Species),var)
  setosa versicolor virginica
0.14368980  0.09846939  0.10400408

sapply(split(Petal.Length,Species),var)
  setosa versicolor virginica
0.03015918  0.22081633  0.30458776

sapply(split(Petal.Width,Species),var)
  setosa versicolor virginica
0.01110612  0.03910612  0.07543265
```

Wir sehen, dass sich die Varianzen (in Relation zu ihrer Höhe) bei drei der Größen stark unterscheiden: `Sepal.Length`, `Petal.Length` und `Petal.Width`. Bei diesen dreien ist also eher zu erwarten, dass die Nullhypothese der Varianzgleichheit abgelehnt werden muss. Führen wir nun den Test durch. Einzusetzen ist dabei der Vektor eines Blattmerkmals als der Datenvektor und der Vektor `Species` als Vektor `group`, da durch ihn die Stichprobenzugehörigkeit bestimmt wird. Da zu jeder `Species` 50 Werte erhoben wurden, sind die Einzelstichproben auch lang genug für den Test.

```
levene.test(Sepal.Length,Species)
```

```
Classical Levene's test based on the absolute
deviations from the mean
```

```
data: Sepal.Length
Test Statistic = 7.3811, p-value = 0.0008818
```

Der p-Wert ist (wie erwartet) sehr klein. Natürlich muss man nicht vorher die Varianzen bestimmen und miteinander vergleichen, das sollte hier nur der Anschaulichkeit dienen. Bei den anderen drei Größen ergibt sich:

```
levene.test(Sepal.Width,Species)
p-value = 0.5498
```

```
levene.test(Petal.Length,Species)
p-value = 1.216e-08
```

```
levene.test(Petal.Width,Species)
p-value = 2.733e-08
```

◇

Beispiel 5.14. Der Datensatz `ToothGrowth` enthält die Daten der Zahn­längen von Meerschweinchen, die gezielt Vitamin C bekamen. Wir wollen uns nun anschauen, ob die Streuung der Werte bei unterschiedlichen Dosierungen des Vitamins verschiedene Werte annimmt. Wir geben also folgendes ein:

```
attach(ToothGrowth)
levene.test(len,dose)
```

```
Classical Levene's test based on the absolute
deviations from the mean
```

```
data: len
Test Statistic = 0.7328, p-value = 0.4850
```

Hier lässt sich die Nullhypothese der Varianzgleichheit zu den typischen Niveaus (zwischen 90% und 99%) nicht ablehnen.

◇

Wenn mehrere unabhängige Stichproben gegeben sind, kann man natürlich auch diese auf Verteilungsgleichheit untersuchen, wozu man eine Verallgemeinerung des Wilcoxon-Rangsummen-Tests verwenden kann, den **Kruskal-Wallis-Test**. Bei diesem muss vorausgesetzt werden, dass die $X_{i,1}, \dots, X_{i,n_i}$ unabhängig und identisch verteilt sind wie X_i mit stetigen Verteilungsfunktionen F_i , $1 \leq i \leq k$. Die Hypothesen sind

$$H_0 : F_1 = \dots, F_k \quad \text{gegen} \quad H_1 : \exists i, j : F_i \neq F_j.$$

Man bildet nun die Rangsummen der einzelnen Stichproben bzgl. aller Beobachtungen

$$R_i(x_1, \dots, x_k) = \sum_{j=1}^{n_i} R(x_{ij})$$

mit

$$E(R_i) = \frac{n_i(n+1)}{2} \quad \text{und} \quad \text{Var}(R_i) = \frac{n_i n(n+1)}{12},$$

wobei $n = \sum_{i=1}^k n_i$ die Gesamtanzahl aller Beobachtungen sei. Wenn die einzelnen Stichproben verteilungsgleich sind, sollten die Rangsummen R_i ungefähr den Anteil $\frac{n_i}{n}$ an der Gesamtsumme haben. Die zu betrachtende Prüfgröße ist daher die Summe der quadrierten standardisierten Rangsummen

$$H(x) = \sum_{i=1}^k \frac{12}{n_i n(n+1)} \left(R_i(x_1, \dots, x_k) - \frac{n_i(n+1)}{2} \right)^2,$$

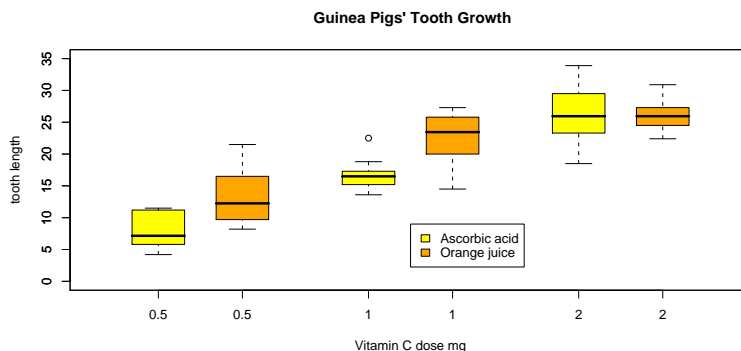
die unter H_0 verteilungsfrei ist, allerdings für den Fall, dass $n_i > 5$ für alle $1 \leq i \leq k$ gilt, asymptotisch χ_{k-1}^2 -verteilt ist (nach dem zentralen Grenzwertsatz und der Definition der χ^2 -Verteilung). Man erhält diesen Test in R mit der Funktion `kruskal.test`, die folgendermaßen verwendet wird:

`kruskal.test`

```
kruskal.test(x, g, ...)
```

wobei - wie beim Levene-Test - der Datensatz als Vektor `x` und ein Faktor zur Kennzeichnung der Stichproben als Vektor `g` eingesetzt werden.

Beispiel 5.15. Bleiben wir bei `ToothGrowth` wie im vorigen Beispiel. Hier können wir nun mithilfe des Kruskal-Wallis-Tests überprüfen, ob bezüglich der durch die Dosierungen vorgegebenen Gruppen Verteilungsgleichheit vorliegt. Bei der grafischen Umsetzung des Datensatzes (z.B. mit Boxplots, wie in den Figuren schon geschehen) sieht man, dass hier keine Verteilungsgleichheit gegeben sein kann. Da die Zahnlänge ein stetiges



Merkmal ist, kann der Test auch angewendet werden. Dabei ist der Vektor `dose` wieder derjenige, der die Gruppen definiert, und wird daher als zweites Argument eingegeben.

```
kruskal.test(len, dose)
```

Kruskal-Wallis rank sum test

```
data: len and dose
```

```
Kruskal-Wallis chi-squared = 40.6689, df = 2, p-value = 1.475e-09
```

Der Test zeigt genau das Ergebnis, das wir erwartet haben: Die Nullhypothese ist praktisch zu allen Niveaus nicht haltbar.

◇

Im k -Stichprobenfall können wir auch mithilfe eines χ^2 -Testes überprüfen, ob die Stichproben verteilungsgleich sind. Der Test verläuft nach denselben Prinzipien wie schon der χ^2 -Anpassungstest und der χ^2 -Unabhängigkeitstest. Der Test, den wir uns hier anschauen, heißt χ^2 -Homogenitätstest. Hier bilden wir l disjunkte Klassen K_1, \dots, K_l , die alle Stichproben x_1, \dots, x_k überdecken und erstellen für diese Einteilung eine Kontingenztabelle

$x_i \setminus K_l$	K_1	K_2	\dots	K_l	Σ
x_1	n_{11}	n_{12}	\dots	n_{1l}	$n_{1\cdot}$
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
x_k	n_{k1}	n_{k2}	\dots	n_{kl}	$n_{k\cdot}$
Σ	$n_{\cdot 1}$	$n_{\cdot 2}$	\dots	$n_{\cdot l}$	n

Dabei ist n_{ij} wie schon vorher die Anzahl der Stichprobenwerte von x_i , die in der Klasse K_j liegen. Man untersucht nun wieder, die Abweichungen der tatsächlichen Häufigkeiten n_{ij} der einzelnen Stichproben-Klassen-Kombinationen von den erwarteten Häufigkeiten e_{ij} , die wieder durch $e_{ij} = n\pi_{ij}$ gebildet werden mit den Wahrscheinlichkeiten

$$\pi_{ij} = P_0(X_i \in K_j)$$

unter der Nullhypothese, dass alle Stichproben die Verteilung P_0 besitzen. Die Prüfgröße gestaltet sich hier zu

$$T(x_1, \dots, x_k) = \sum_{i=1}^k \sum_{j=1}^l \frac{(n_{ij} - e_{ij})^2}{e_{ij}},$$

und diese ist unter der Nullhypothese asymptotisch $\chi_{k \cdot l - k - (l-1)}^2$ -verteilt. Damit diese Asymptotik gelten kann, müssen wir wie schon zuvor voraussetzen, dass die erwarteten Besetzungszahlen e_{ij} alle mindestens 1 betragen und mindestens 80% von ihnen größer oder gleich 5 sind. Um den Test durchzuführen, wendet man auch hier die Funktion `chisq.test` auf die Kontingenztabelle an.

6 Varianzanalyse

Bei der Varianzanalyse geht es darum, für eine metrische Zielvariable X zu untersuchen, ob eine oder mehrere qualitative oder quantitative Größen A, B, C, D, \dots Einfluss auf den Mittelwert der Verteilung von X haben. Dabei werden häufig quantitative Merkmale kategorisiert, d.h. man misst die Zielvariable nur für bestimmte Werte einer oder mehrerer Einflussgrößen.

Beispiel 6.1. Wir haben den Datensatz `ToothGrowth` bereits in den Übungen kennengelernt. Dabei handelt es sich um eine Versuchsreihe, die zeigen sollte, ob und wie das Zahnwachstum bei Meerschweinchen von der Vitamin-C-Zufuhr abhängt. Dabei wurde das Augenmerk auf zwei Einflussfaktoren gelenkt: Die Art der Vitamin-C-Zufuhr und die Höhe der Dosierung. Bei dem ersten Merkmal handelt es sich offensichtlich um ein nominalskaliertes Merkmal, das hier nur die Ausprägungen VC (Verabreichung von Vitamin C in Reinform) und OJ (Zufuhr mithilfe von Orangensaft) annimmt. Das zweite Merkmal ist eigentlich ein stetiges metrisches Merkmal, doch es wurden immer nur bestimmte Dosen verabreicht: 0.5mg, 1.0mg und 2.0mg.

◇

Um nun analysieren zu können, welchen Einfluss die Größen A, B, C, D, \dots auf X haben, erstellt man folgendes Modell:

$$\begin{aligned} X &= f(A, B, C, D) = \mu + \varepsilon \\ \text{mit } \mu &= E_{\vartheta}(X) = g_{A,B,C,D}(\vartheta) \\ \text{und } E_{\vartheta}(\varepsilon) &= 0 \end{aligned}$$

Die Annahme ist also, dass wir die Zielvariable X vollständig durch die Einflussgrößen erklären können. Weiter können wir X durch den Erwartungswert plus gewisse Abweichungen ε darstellen, die natürlich den Erwartungswert 0 haben müssen. Üblicherweise gehen wir davon aus, dass diese Fehler normalverteilt sind, wodurch dann auch unsere Zielgröße normalverteilt sein muss. Diese Annahmen der Normalverteilung gehören zur **klassischen Methode nach Fisher**, die auf einem linearen Modell beruht, d.h. es wird weiterhin angenommen, dass die Größen A, B, C, D nur linearen Einfluss auf den Mittelwert (über die folglich lineare Funktion $g_{A,B,C,D}$) ausüben. Die Vorstellung dahinter ist, dass X (durch die Werte von A, B, C, D) eigentlich deterministisch ist, weshalb sich immer der konkrete Wert μ ergeben sollte. Doch es schleichen sich üblicherweise Abweichungen und Unsicherheiten durch die Messvorgänge ein, die als normalverteilt angenommen werden, so dass kleine Abweichungen nicht unwahrscheinlich sind und größere Abweichungen nur selten eintreten. Daraus ergibt sich die Fehlervariable ε .

Ein lineares Modell basiert dabei auf der Kleinste-Quadrate-Methode, die die Summe der Fehlerquadrate, also der quadrierten Abweichungen der tatsächlichen Werte von denen, die gemäß des Modells vorliegen sollten, bildet. Je kleiner die Summe der Fehlerquadrate ist, desto besser ist die Anpassung des Modells an die Wirklichkeit.

Um eine Grundlage für Rückschlüsse zu haben, wird die Zielvariable mehrmals unter den verschiedenen Bedingungen, die sich aus den möglichen Werten der Einflussgrößen ergeben, gemessen. Die so entstandene Stichprobe (x_1, \dots, x_n) wird als Realisierung von unabhängigen und identisch wie X verteilten Zufallsgrößen

$$X_i = \mu + \varepsilon_i$$

aufgefasst, wobei die Fehler ε_i homoskedastisch seien, d.h. dieselbe Varianz besitzen sollen. Der Vorteil bei diesem Modell ist, dass hier ein **exakter** F -Test zur Verfügung steht, während es bei verallgemeinerten linearen Modellen nur asymptotische Testverfahren gibt.

Man unterscheidet bei der Varianzanalyse die Verfahren nach der Anzahl der erklärenden Variablen. Gehen wir also von einem Einflussfaktor aus, sprechen wir von einfaktorieller Varianzanalyse, bei zwei Faktoren von zweifaktorieller Varianzanalyse, usw. Wenn die Einflussgrößen metrisch sind, spricht man Kovariaten und entsprechend der Kovariatenanalyse.

6.1 Einfaktorielle Varianzanalyse

Es soll untersucht werden, ob sich die Verteilung eines Merkmals X in Abhängigkeit von einem anderen Merkmal A ändert. Dazu werden k Stichproben

$$\begin{aligned} x_1 &= (x_{1,1}, \dots, x_{1,n_1}) \\ &\vdots \\ x_k &= (x_{k,1}, \dots, x_{k,n_k}) \end{aligned}$$

erhoben, wobei in jeder Stichprobe eine andere Ausprägung des Einflussmerkmals vorliegt (und zwar genau eine!). Dadurch werden hier auch Kovariaten (metrische Merkmale) zu Faktoren. Da wir hier die Fisher-Methode anwenden wollen, müssen wir voraussetzen, dass die Stichproben x_1, \dots, x_k aus Normalverteilungen $\mathcal{N}(\mu_i, \sigma^2)$ hervorgehen. Bevor wir also mit einer Analyse starten können, müssen wir uns mithilfe von grafischen Methoden oder Tests auf die Verteilung vergewissern, dass die Normalverteilungsannahme gerechtfertigt ist. Bei der Varianzanalyse wird dann ein Mittelwertvergleich angestellt, wobei die Hypothesen folgende sind:

$$H_0 : \mu_1 = \dots = \mu_k \quad \text{gegen} \quad H_1 : \exists i, j : \mu_i \neq \mu_j.$$

Eine Prüfgröße dafür erhalten wir, indem wir die Streuung der Mittelwerte (also die Streuung zwischen den Stichproben) betrachten, allerdings in Relation zur Streuung innerhalb der Stichproben. Eine gewisse Streuung zwischen den Stichproben kann noch

akzeptabel sein und nicht gegen die Nullhypothese sprechen, wenn in den einzelnen Stichproben die Werte schon stark gestreut sind. Liegt in den einzelnen Stichproben jedoch nur eine geringe Streuung vor, fällt jede Abweichung zwischen den Mittelwerten gleich viel stärker ins Gewicht und spricht eher gegen die Nullhypothese. Dazu zerlegen wir die einzelnen Beobachtungen folgendermaßen:

$$x_{ij} = \mu + (\mu_i - \mu) + (x_{ij} - \mu_i),$$

wobei μ das Gesamtmittel aller Beobachtungen sei. Man zerlegt eine Beobachtung in drei Summanden: Der erste ist das Gesamtmittel, der zweite Summand gibt an, wie das Stichprobenmittel vom Gesamtmittel abweicht, und der dritte schließlich, wie sich der Wert der Beobachtung vom Stichprobenmittel unterscheidet. Dieser letzte Term ist ungefähr das, was die Fehler ε_i beschreiben, die eine $\mathcal{N}(0, \sigma^2)$ -Verteilung besitzen. Beim zweiten Term macht sich dagegen der Einfluss der zweiten Größe A bemerkbar und man könnte diesen Term als eine Realisierung a_i des Faktors A auffassen. Mit diesem Ansatz kann man nun die Varianz der Gesamtstichprobe zerlegen:

$$\begin{aligned} SQT &= SQE + SQR \\ \text{mit : } SQT &= \sum_{i=1}^k \sum_{j=1}^{n_i} (x_{ij} - \mu)^2, \text{ (Gesamtstreuung),} \\ SQE &= \sum_{i=1}^k n_i (\mu_i - \mu)^2, \text{ (Streuung zwischen den Stichproben),} \\ SQR &= \sum_{i=1}^k (n_i - 1) s_i^2, \text{ (Streuung innerhalb der Stichproben),} \end{aligned}$$

wobei wieder s_i^2 die Stichprobenvarianz der i -ten Stichprobe sei. Ein Teil der Gesamtstreuung erklärt sich also über die Einteilung der Gesamtstichprobe in k Stichproben, also kleinere Gruppen. Die Frage ist, wieviel dadurch erklärt werden kann. Wenn die Einteilung der Gruppen rein zufällig war, es also keine Abhängigkeit der Zielvariablen vom Einflussfaktor gibt, sollten die beiden Summanden jeweils einen Anteil an der Gesamtstreuung haben, der ihren Freiheitsgraden entspricht. Bei der Streuung zwischen den Gruppen ist dies $k - 1$, bei der Streuung innerhalb der Gruppen $n - k$, wenn $n = \sum_{i=1}^k n_i$ die Gesamtanzahl an Beobachtungen ist. Die Freiheitsgrade ergeben sich daraus, dass man im ersten Fall nur $k - 1$ der k Stichprobenmittel „frei“ wählen kann, da das k -te durch das Gesamtmittel und die übrigen Stichprobenmittel dann schon festgelegt ist. Ebenso sind im zweiten Fall in jeder Stichprobe alle Beobachtungswerte bis auf einen frei wählbar, da der letzte durch die anderen und das Stichprobenmittel bestimmt ist. Da k Gruppen vorliegen, ergeben sich bei insgesamt n Beobachtungen also $n - k$ Freiheitsgrade. Wenn wir nun aus den beiden Quadratsummen die mittleren Quadratsummen berechnen, erhalten wir zwei Schätzer für die Varianz σ^2 , die unter der Nullhypothese

6 Varianzanalyse

erwartungstreu sind. Weiter gilt

$$\begin{aligned} \frac{SQE}{\sigma^2} &\stackrel{H_0}{\sim} \chi_{k-1}^2, \\ \frac{SQR}{\sigma^2} &\sim \chi_{n-k}^2, \\ F &:= \frac{\frac{1}{k-1}SQE}{\frac{1}{n-k}SQR} \stackrel{H_0}{\sim} F_{k-1, n-k}, \end{aligned}$$

d.h. wenn die Nullhypothese vorliegt, besitzt der betrachtete Quotient F eine F -Verteilung mit $k - 1$ und $n - k$ Freiheitsgraden. Idealerweise wäre der Wert dieser Statistik 1, doch es werden sich auch bei Vorliegen von H_0 Abweichungen von diesem Wert ergeben. Grundsätzlich gilt aber, dass sich die Mittelwerte mehr ähneln, wenn der Wert der Teststatistik klein ist. Der konkrete Test ist also einseitig, obwohl sich das Testproblem uns als ein zweiseitiges präsentiert.

Das Vorgehen in R baut darauf auf, dass wir zunächst ein lineares Modell erstellen und anschließend für dieses Modell eine Varianzanalyse durchführen. Das lineare Modell ist im einfachsten Fall

$$X = A + \varepsilon,$$

wir können X aber auch auf andere Abhängigkeiten vom Faktor A überprüfen, z.B.:

$$\begin{aligned} X &= A^{-1} + \varepsilon, \\ X &= \log(A) + \varepsilon, \\ X &= c^A + \varepsilon, \end{aligned}$$

usw. Es gibt mehrere Funktionen, mit denen man ein solches Modell aufstellen kann, z.B. `lm` (Abkürzung für engl.: linear model), `glm` (Abkürzung für engl.: general linear model), `lme` (abkürzung für engl.: linear mixed effects) und insbesondere `aov` (Abkürzung für engl.: analysis of variance). Mit den zuerst genannten Funktionen kann man nur Modelle erstellen, die letzte der Funktionen (`aov`) ist auch direkt für die Varianzanalyse zuständig. Die Ausgabe beinhaltet die Quadratsummen (`Residuals` kennzeichnet die Werte der Streuung innerhalb der Gruppen, der Name des Faktors steht vor der Quadratsumme der Streuungen zwischen den Stichproben) und die zugehörigen Freiheitsgrade. Wenn man noch `summary` auf sie anwendet, erhält man die Auswertung der Teststatistik, die darüber hinaus die gemittelten Quadratsummen, den Wert der Statistik und den p-Wert enthält. Hinter dem p-Wert ist die Signifikanz durch Sternchen gekennzeichnet: Drei Sternchen bedeuten höchste Signifikanz (Niveau $\geq 99.9\%$), zwei Sternchen hohe Signifikanz (Niveau zwischen 99.0% und 99.9%), ein Sternchen einfache Signifikanz (Niveau zwischen 95% und 99%), ein Punkt steht für schwache Signifikanz zu Niveaus zwischen 90% und 95% . Signifikanz bedeutet hier, dass der Faktor A tatsächlich Einfluss auf die Zielvariable X hat. Dieselben Ergebnisse liefert die Funktion `anova` (ebenfalls Abkürzung für engl.: analysis of variance), wenn man sie auf ein bereits erstelltes Modell anwendet.

`lm glm aov`

`anova`

6 Varianzanalyse

Wenn wir eine Funktion verwenden, um $X = A + \varepsilon$ zu modellieren, setzen wir in diese Funktion $X \sim A$ ein, also beispielsweise:

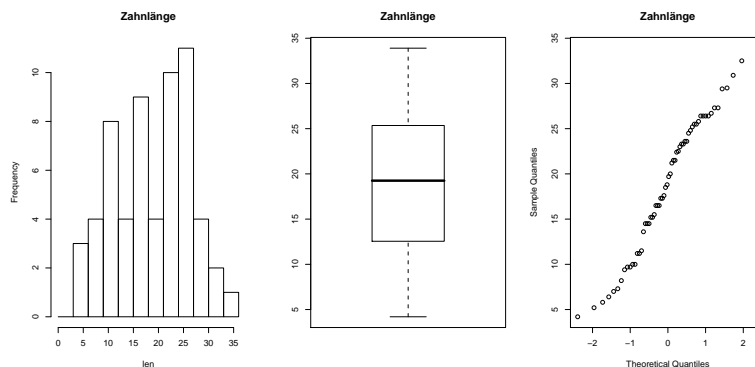
```
lm(X~A)
aov(X~A)
```

Dabei wird die Tilde gelesen als „in Abhängigkeit von“ oder „wird erklärt durch“. Natürlich kann man auch kompliziertere Modelle aufstellen, z.B. durch funktionale Zusammenhänge:

```
lm(X~log(A))
glm(X~1/A)
aov(X~c^A)
```

Häufig ergeben sich solche Ansätze, wenn man Streudiagramme erzeugt und funktionale Zusammenhänge erkennen kann. Bei vielen Datensätzen werden metrische Merkmale zu Faktoren umgewandelt, indem die gesamte Untersuchung gezielt nur für bestimmte Werte durchgeführt wird, so dass sich die Darstellung mithilfe eines Streudiagrammes lohnen kann. Im Falle wirklicher Faktoren (also qualitativer Merkmale) hat eine solche Darstellung nur wenig Sinn. Kompliziert wird das Erstellen eines passenden Modelles auch erst, wenn man Varianzanalyse bezüglich mehrerer Faktoren betreibt, wo auch die Wechselwirkungen zwischen den Faktoren Einfluss auf die Zielvariable haben können.

Beispiel 6.2. Bleiben wir einmal beim Datensatz `ToothGrowth`. Hier wurde die Zahnlänge der Meerschweinchen mit Blick auf den Einfluss von Vitamin C untersucht. Dabei wurden zwei Einflüsse berücksichtigt: Die Art der Vitamin-C-Zufuhr und die Dosierung. Wir interessieren uns nun dafür, welchen Einfluss die Darreichungsform auf das Zahnwachstum hat. Als erstes müssen wir uns vergewissern, dass die Zahnlänge ein normalverteiltes Merkmal ist, bevor wir mit der Varianzanalyse beginnen können:



```
hist(len,seq(0,36,3),include.lowest=T,main=„Zahnlänge“)
boxplot(len,main=„Zahnlänge“)
```

6 Varianzanalyse

```
qqnorm(len,main=„Zahnlänge“)
```

Hier ist eine Normalverteilungsannahme offenbar gerechtfertigt. Wir erstellen also nun das Modell $\text{len} = a + b_1 \cdot \text{OJ} + b_2 \cdot \text{VC}$:

```
lm(len~supp)
```

Call:

```
lm(formula = len ~ supp)
```

Coefficients:

```
(Intercept)  suppVC  
    20.66    -3.70
```

Die Ausgabe der Funktion `lm` besteht aus einer Tabelle, die zu jeder Ausprägung des Faktors (bis auf die erste) angibt, mit welchem Koeffizienten sie linear in die Zielvariable eingeht. Außerdem wird der y-Achsenabschnitt (`Intercept`) der linearen Anpassung der Zielvariable durch den Faktor angegeben. Aus diesem und anderen angegebenen Koeffizienten sowie den Werten der Zielvariable `len` kann man den Koeffizienten b_1 der ersten Ausprägung des Faktors berechnen. Unser Modell ist hier also:

```
len=20.66+b1*OJ-3.70*VC
```

Dieses Modell setzen wir nun zur Varianzanalyse in die Funktion `anova` ein:

```
anova(lm(len~supp))
```

Analysis of Variance Table

Response: len

	Df	Sum Sq	Mean Sq	F value	Pr(> F)
supp	1	205.4	205.4	3.6683	0.06039 .
Residuals	58	3246.9	56.0		

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Wenn wir nun die mittleren Streuungen miteinander vergleichen, sehen wir, dass die Streuung zwischen den Gruppen ungefähr viermal so groß ist wie die innerhalb der Gruppen. Daher ist der Wert der Teststatistik nicht klein und wir erhalten eine Signifikanz zum Niveau 90%. Zum gleichen Ergebnis kommen wir mit der Funktion `aov` und mit `summary`. Mit `aov` erhalten wir nur die Varianztafel, die die Streuungswerte umfasst, bei der Anwendung von `summary` kommt die Testauswertung hinzu. außerdem ist es häufig nützlich, das Modell als Variable zu speichern, so dass man sich Schreibarbeit erspart.

6 Varianzanalyse

```
Zähne<-lm(len~supp)
```

```
aov(Zähne)
```

```
Call:
```

```
  aov(formula = Zähne)
```

```
Terms:
```

	supp	Residuals
Sum of Squares	205.350	3246.859
Deg. of Freedom	1	58

```
Residual standard error: 7.482
```

```
Estimated effects may be unbalanced
```

```
summary(aov(Zähne))
```

	Df	Sum Sq	Mean Sq	F value	Pr(> F)
supp	1	205.4	205.4	3.6683	0.06039 .
Residuals	58	3246.9	56.0		

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

◇

Wenn (anders als im gerade betrachteten Beispiel) noch kein Faktor vorliegt, der die verschiedenen Stichproben bzw. Gruppen einteilt, und nur die k Stichproben x_1, \dots, x_k zum selben Merkmal gegeben sind, erzeugen wir aus den Stichproben den Vektor der Zielvariablen und einen entsprechenden Faktor, der kennzeichnet, aus welcher Stichprobe die einzelnen Werte stammen:

```
Zielvar<-c(x1, ..., xk)
```

```
Faktor<-factor(rep(1:k, c(n1, ..., nk)))
```

Beispiel 6.3. Beim Datensatz `VADeaths` könnten wir uns fragen, ob die Bevölkerungsgruppe (definiert durch Geschlecht und Umfeld) signifikanten Einfluss auf die Sterberaten hat. Zunächst überzeugen wir uns davon, dass es sich um ein normalverteiltes Merkmal handelt:

```
vad<-as.vector(VADeaths)
```

```
hist(vad)
```

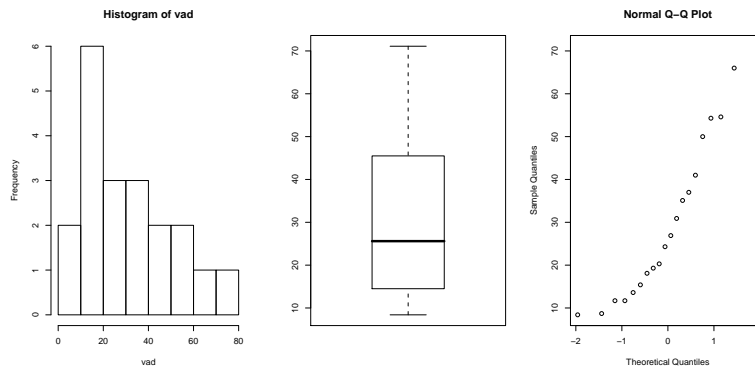
```
boxplot(vad)
```

```
qqnorm(vad)
```

Es könnte sich hier um eine Normalverteilung handeln, daher beginnen wir nun mit der Varianzanalyse:

```
bev<-factor(rep(1:4, rep(5, 4)))
```

6 Varianzanalyse



```
modell<-aov(vad~bev)
summary(modell)
Pr(>F) = 0.5876
```

Es lässt sich also kein signifikanter Einfluss feststellen, was hier auch zu erwarten war, haben wir doch im vorigen Kapitel mit dem χ^2 -Unabhängigkeitstest die Nullhypothese der Unabhängigkeit der beiden Merkmale nicht ablehnen können.

◇

Wenn wir unsere Zielvariable auf Abhängigkeit von einer metrischen Einflussgröße Y untersuchen wollen, müssen wir diese zunächst in ein qualitatives Merkmal (also einen Faktor) umwandeln durch Klassierung. Anderenfalls würden wir eine Regressionsanalyse durchführen, die andere Auswertungsmethoden verwendet.

Beispiel 6.4. Wenn wir beim `iris`-Datensatz untersuchen wollen, ob `Sepal.Length` (von der Normalverteilung haben wir uns bereits überzeugt) von `Sepal.Width` beeinflusst wird, müssen wir `Sepal.Width` zuerst in einen Faktor umwandeln, ansonsten erhielten wir:

```
attach(iris)
aov(Sepal.Length~Sepal.Width)    #FALSCH!
Call:
  aov(formula = Sepal.Length ~ Sepal.Width)

              Sepal.Width  Residuals
Terms:  Sum of Squares      1.41224  100.75610
        Deg. of Freedom      1          148

Residual standard error: 0.8250966
Estimated effects may be unbalanced
```

Das man hier etwas falsch gemacht hat, sieht man sofort daran, dass bei der Streuung zwischen den Gruppen der Freiheitsgrad 1 ist. Das liegt natürlich daran, dass keine

6 Varianzanalyse

Gruppen eingeteilt sind. Infolgedessen können wir mit diesem Ansatz auch nicht untersuchen, ob signifikante Unterschiede zwischen den Gruppenmitteln auftreten! Der richtige Ansatz basiert auf der Einteilung der Einflussgröße in Gruppen, also der Umwandlung in einen Faktor:

```
range(Sepal.Width)
sw<-cut(Sepal.Width,seq(2,4.4,0.2),include.lowest=T)
aov(Sepal.Length~sw)          #RICHTIG!
Call:
  aov(formula = Sepal.Length ~ sw)

              sw Residuals
Terms: Sum of Squares  11.43742  90.73091
      Deg. of Freedom    11      138

Residual standard error: 0.8108455
Estimated effects may be unbalanced
```

Diese Varianztafel sieht schon sinnvoller aus und wir können die Auswertung vornehmen. Hier ist natürlich anzumerken, dass die Anzahl der Freiheitsgrade der Streuung zwischen den Gruppen unmittelbar von unserer (eher willkürlichen) Einteilung der Klassen abhängt.

```
summary(aov(Sepal.Length~sw))
      Df Sum Sq Mean Sq F value Pr(> F)
sw      11  11.437   1.040   1.5815  0.1106
Residuals 138  90.731   0.657
```

Wir sehen hier, dass die Gruppen, wie wir sie eingeteilt haben, keine Signifikanz besitzen. Zum Vergleich führen wir dasselbe noch einmal für eine andere Klasseneinteilung durch:

```
SW<-cut(Sepal.Width,seq(2,4.4,0.4),include.lowest=T)
summary(aov(Sepal.Length~SW))
Pr(>F) = 0.01440 *
```

Bei dieser Einteilung ergibt sich also eine Signifikanz, wodurch deutlich wird, wie sehr die Klasseneinteilung die Auswertung beeinflussen kann.

◇

Wenn man nun eine Varianzanalyse durchgeführt und die Signifikanz des Faktors erhalten hat, ist es interessant zu wissen, worin der Unterschied besteht. Daher wird ein Vergleich der verschiedenen Gruppen nötig. Da bei der Varianzanalyse der Einfluss von Faktoren auf den Mittelwert der Zielvariable untersucht wird, muss bei dem Vergleich der Gruppen auch das Augenmerk auf die Mittelwerte gelenkt werden. Um herauszu-

finden, welche Mittelwerte sich unterscheiden und dadurch die Signifikanz des Faktors herbeiführen, kann man paarweise für alle Paarungen von Gruppen vergleichende t-Tests durchführen. Dazu gibt es eine Funktion namens `pairwise.t.test`, die diese Tests für alle möglichen Gruppenpaarungen ausführt. Die „Bedienung“ dieser Funktion ist wie bei `kruskal.test` oder `levene.test`: Man setzt als erstes die Zielvariable ein und als zweites den Faktor, der angibt, zu welcher Stichprobe die einzelnen Werte gehören. Bei diesem Test werden die p-Werte der einzelnen Paarungen nachträglich an die Situation des Zweistichprobentests mit dem entsprechend auf diese beiden Stichproben reduzierten Datensatz angepasst. Dazu gibt es verschiedene Methoden, die man mit dem Argument `p.adjust.method` bzw. `p.adjust` vornehmen kann. Die Standardmethode ist dabei die von Holm. Insgesamt stehen uns acht solche Methoden zur Verfügung, Details finden sich im Hilfetext. Mit dem Argument `pool.sd` können wir einstellen, ob die gepoolte Stichprobenvarianz als Schätzer für die Varianz σ^2 (und somit der Standardabweichung σ) aller Stichproben genommen werden soll. Geht man davon aus, dass nicht alle Varianzen der Stichproben gleich sind, dass also der heteroskedastische Fall vorliegt, kann man nicht mehr die klassische Varianzanalyse durchführen. Allerdings bietet sich mit dem paarweisen t-Test die Möglichkeit, auch bei Heteroskedastizität die Stichproben paarweise auf Mittelwertgleichheit zu überprüfen, indem man keine gepoolte Stichprobenvarianz verwendet.

`pairwise.t.test`

Beispiel 6.5. Wir greifen noch einmal auf das vorhergehende Beispiel zurück und untersuchen, zwischen welchen Gruppen sich die signifikanten Unterschiede ergeben. Wir haben im obigen Beispiel nur Signifikanz bei der Varianzanalyse für die Einteilung von `Sepal.Width` in Klassen der Länge 0.4 erreicht, also mit `SW`. Nun untersuchen wir paarweise, welche Gruppenmittelwerte sich signifikant unterscheiden. Zunächst gehen wir davon aus, dass die Verteilungen, die den Gruppen zugrundeliegen, alle dieselbe Varianz besitzen. Daher verwenden wir `pool.sd=TRUE`, was die Standardeinstellung ist.

```
pairwise.t.test(Sepal.Length,SW)
```

```
Pairwise comparisons using t tests with pooled SD
```

```
data: Sepal.Length and SW
```

	[2, 2.4]	(2.4, 2.8]	(2.8, 3.2]	(3.2, 3.6]	(3.6, 4]
(2.4, 2.8]	0.519	—	—	—	—
(2.8, 3.2]	0.669	1.000	—	—	—
(3.2, 3.6]	1.000	0.040	0.044	—	—
(3.6, 4]	1.000	1.000	1.000	1.000	—
(4, 4.4]	1.000	1.000	1.000	1.000	1.000

Hier sehen wir, dass bei zwei Paarungen signifikante Abweichungen der Mittelwerte festgestellt wurden: Bei `(3.2,3.6]` und `(2.4,2.8]` bzw. bei `(3.2,3.6]` und `(2.8,3.2]`. Hierbei sind wir von der Annahme ausgegangen, dass in allen durch `SW` eingeteilten Gruppen dieselbe Varianz vorliegt. Schauen wir uns einmal an, welche Werte die Stich-

probenvarianzen annehmen:

```
sapply(split(Sepal.Length, SW), var)
      [2, 2.4]  (2.4, 2.8]  (2.8, 3.2]  (3.2, 3.6]  (3.6, 4]  (4, 4.4]
0.34890909  0.40351587  0.85745480  0.50554233  0.97181818  0.06333333
```

Hier zeigen sich schon deutliche Unterschiede zwischen den Stichprobenvarianzen. Diesen ersten Eindruck erweitern wir jetzt noch, indem wir den test auf Varianzheterogenität von Levene anwenden:

```
levene.test(Sepal.Length, SW)
p-value = 0.01092
```

Es liegen also signifikante Unterschiede zwischen den Varianzen vor. Daher untersuchen wir noch einmal die Gruppen paarweise auf Mittelwertunterschiede, verwenden dabei aber eine Teststatistik, die nicht auf der gepoolten Stichprobenvarianz aufbaut (die ja nur im homoskedastischen Fall Sinn macht).

```
pairwise.t.test(Sepal.Length, SW, pool.sd=F)
```

Pairwise comparisons using t tests with non-pooled SD

data: Sepal.Length and SW

	[2, 2.4]	(2.4, 2.8]	(2.8, 3.2]	(3.2, 3.6]	(3.6, 4]
(2.4, 2.8]	0.168	—	—	—	—
(2.8, 3.2]	0.308	1.000	—	—	—
(3.2, 3.6]	1.000	0.010	0.043	—	—
(3.6, 4]	1.000	1.000	1.000	1.000	—
(4, 4.4]	1.000	0.284	0.329	1.000	1.000

Hier ergeben sich bei einigen Paarungen kleinere p-Werte als zuvor, allerdings gibt es auch hier keine weiteren Paarungen mit signifikanten Mittelwertunterschieden. Dennoch sieht man, dass es eine Rolle spielt, ob sich die Varianzen der einzelnen Gruppen unterscheiden. Es kann unter Umständen vorkommen, dass die Varianzanalyse bei Annahme der Homoskedastizität eine Signifikanz aufzeigt, die eigentlich nicht gegeben ist, was sich dann mithilfe des paarweisen t-Tests mit ungepoolter Stichprobenvarianz herausstellt.

◇

6.2 Mehrfaktorielle Varianzanalyse

Wir schauen uns der Einfachheit halber zunächst nur den Fall zweier Faktoren an. Die anderen Fälle werden im Grunde analog behandelt, so dass wir auf die Theorie nicht weiter eingehen werden.

6 Varianzanalyse

Will man untersuchen, ob die Zielvariable X von zwei verschiedenen Faktoren A und B beeinflusst wird, sollten beide Faktoren gleichberechtigt sein. Das bedeutet, dass jede der I Stufen von A mit jeder der J Stufen von B gekreuzt wird und dass für jede dieser Faktorkombinationen K Messwerte erhoben werden. Insgesamt erhält man $l := IJ$ abhängige Stichproben der Länge K und somit $n := IJK$ Messwerte. Jeder Messwert x_{ijk} wird dabei als Realisierung einer Zufallsgröße X_{ijk} aufgefasst, für die folgendes gilt:

$$\begin{aligned} X_{ijk} &= \mu_{ij} + \varepsilon_{ijk} \\ \mu_{ij} &= \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij}, \\ \text{mit :} \quad \mu_{ij} &= \sum_{k=1}^K x_{ijk}, \\ \varepsilon_{ijk} &\sim \mathcal{N}(0, \sigma^2) \end{aligned}$$

und den Haupteffekten α_i und β_j der Faktoren A und B in den Stufen i bzw. j sowie dem Wechselwirkungseffekt der Faktorkombination (i, j) von A und B . Man geht also wieder davon aus, dass die aus den l unterschiedlichen Faktorkombinationen hervorgehenden l Stichproben einer Normalverteilung unterliegen, wobei der jeweilige Stichprobenmittelwert μ_{ij} aus dem Gesamtmittel μ durch die Einflüsse der Faktorstufen α_i und β_j bzw. durch den Einfluss derer Interaktion $(\alpha\beta)_{ij}$ entsteht. Für diese drei Effekte gilt, dass sie sich im Mittel aufheben, also

$$\sum_{i=1}^I \alpha_i = \sum_{j=1}^J \beta_j = 0 = \sum_{i=1}^I (\alpha\beta)_{ij} = \sum_{j=1}^J (\alpha\beta)_{ij}.$$

Ist der Wechselwirkungsterm $(\alpha\beta)_{ij}$ positiv (negativ), heißt das, dass die beiden Faktoren A und B in der Stufenkombination (i, j) einen größeren (kleineren) Einfluss auf den Mittelwert von X ausüben als die Summe ihrer Einzeleinflüsse.

Nun ergeben sich mehrere Testprobleme durch die einzelne Betrachtung der Faktoren und die Betrachtung ihrer Interaktion, wobei die entsprechenden Nullhypothesen folgende sind:

$$\begin{aligned} H_A &: \alpha_1 = \dots = \alpha_I = 0 \hat{=} \text{Faktor } A \text{ hat keinen Mittelwerteinfluss auf } X. \\ H_B &: \beta_1 = \dots = \beta_J = 0 \hat{=} \text{Faktor } B \text{ hat keinen Mittelwerteinfluss auf } X. \\ H_{AB} &: \forall i, j : (\alpha\beta)_{ij} = 0 \hat{=} A \text{ und } B \text{ wirken unabhängig auf } X. \end{aligned}$$

Die Auswertung der Daten erfolgt ähnlich wie im einfaktoriellen Fall, indem die Quadratsummen der Abweichungen der Gruppenmittel der durch die Faktorstufen bestimmten Gruppen vom Gesamtmittel \bar{x} in Relation zur Streuung innerhalb der Faktorkombina-

6 Varianzanalyse

tionen betrachtet werden. Die Summen der Quadrate sind folgende:

$$\begin{aligned}
 SQA &= JK \sum_{i=1}^I (\bar{x}_{i\bullet} - \bar{x})^2, \\
 SQB &= IK \sum_{j=1}^J (\bar{x}_{\bullet j} - \bar{x})^2, \\
 SQAB &= K \sum_{i=1}^I \sum_{j=1}^J (\bar{x}_{ij} - \bar{x}_{i\bullet} - \bar{x}_{\bullet j} + \bar{x})^2, \\
 SQR &= \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K (x_{ijk} - \bar{x}_{ij})^2, \\
 SQT &= SQA + SQB + SQAB + SQR
 \end{aligned}$$

mit den arithmetischen Mitteln

$$\begin{aligned}
 \bar{x} &:= \frac{1}{n} \sum_{i,j,k} x_{ijk}, \\
 \bar{x}_{i\bullet} &:= \frac{1}{JK} \sum_{j,k} x_{ijk}, \\
 \bar{x}_{\bullet j} &:= \frac{1}{IK} \sum_{i,k} x_{ijk}, \\
 \bar{x}_{ij} &:= \frac{1}{K} \sum_k x_{ijk}
 \end{aligned}$$

der Werte der Gesamtstichprobe, der Werte der Faktorstufe i von A bzw. j von B und der Werte der Faktorkombination (i, j) . Wie bei der einfaktorischen Varianzanalyse erhält man die entsprechenden Prüfgrößen für die Testprobleme, wenn man aus den gemittelten Summen der Quadrate der Faktoren bzw. der Wechselwirkung und der Streuung innerhalb der Faktorkombinationen Quotienten bildet:

$$\begin{aligned}
 F_A &= \frac{\frac{1}{I-1} SQA}{\frac{1}{n-IJ} SQR} \stackrel{H_A}{\sim} F_{I-1, n-IJ}, \\
 F_B &= \frac{\frac{1}{J-1} SQB}{\frac{1}{n-IJ} SQR} \stackrel{H_B}{\sim} F_{J-1, n-IJ}, \\
 F_{AB} &= \frac{\frac{1}{(I-1)(J-1)} SQAB}{\frac{1}{n-IJ} SQR} \stackrel{H_{AB}}{\sim} F_{(I-1)(J-1), n-IJ}.
 \end{aligned}$$

Mithilfe dieser Prüfgrößen wird getestet, ob sich durch die Faktoren und ihre Interaktion signifikante Mittelwertunterschiede ergeben. Insgesamt wurde wieder vorausgesetzt, dass

6 Varianzanalyse

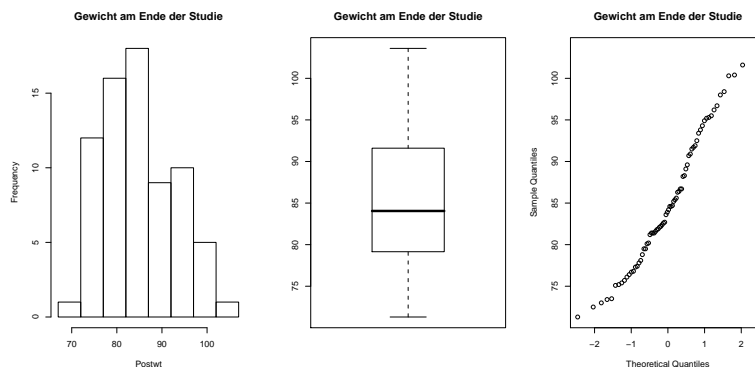
in allen Stichproben dieselbe Varianz vorliegen soll. Dies kann man mit dem Varianzheterogenitätstest von Levene überprüfen. Sollte sich bei diesem Test ein signifikanter Unterschied zwischen den Varianzen ergeben, führt man in der Praxis dennoch eine Varianzanalyse durch, in dem Fall allerdings zu Niveau $1 - \alpha = 0.99$.

Im Fall von $k \geq 3$ Faktoren wird analog verfahren: Es werden wieder die entsprechenden Summen der Streuungen gebildet, gemittelt und durch die gemittelte Streuung der einzelnen Faktorkombinationen dividiert, woraus eine F -verteilte Teststatistik resultiert.

In R ähnelt das Vorgehen stark dem bei der einfaktoriellen Varianzanalyse: Wir erstellen als erstes mithilfe einer modellbildenden Funktion (z.B. `aov`) ein Modell der Zielvariablen X in Abhängigkeit von den Faktoren. Da manchmal die gemessenen Faktoren in keinem linearen, sondern anderen funktionalen Zusammenhang mit der Zielvariable stehen können, lohnt es sich unter Umständen, mithilfe von grafischen Methoden (z.B. Streudiagrammen) einen Überblick über die Daten und mögliche Zusammenhänge zu gewinnen und dann den entsprechend modifizierten Faktor in das Modell eingehen zu lassen. Die Interaktion zwischen zwei Faktoren A und B gibt man in die modellbildenden Funktionen als $A : B$ ein, $A + B + A : B$ kann man dabei abkürzend durch $A * B$ eingeben. Weitere Modellierungen von Faktorenzusammenstellungen werden im Hilfetext zu `formula` erläutert.

`formula`

Beispiel 6.6. Der Datensatz `anorexia` aus dem Paket `MASS` enthält die Daten der Gewichtsveränderung junger Anorexie-Patientinnen bezogen auf das Gewicht zu Beginn der Studie (`Prewt`) und das am Ende der Studie (`Postwt`). Während der Verlaufszeit dieser Studie wurden die Patientinnen nach verschiedenen Methoden (`Treat`) therapiert. Wir untersuchen nun das Gewicht der Patientinnen am Ende der Studie in Abhängigkeit von der Therapie und dem Gewicht zu Beginn der Studie. Der Boxplot deutet zwar auf



eine leicht linkssteile Verteilung hin, Histogramm und Normal-QQ-Plot sehen aber denen einer Normalverteilung recht ähnlich, weshalb wir hier von einer solchen ausgehen. Nun gehen wir zur Varianzanalyse über. Als erstes müssen wir dafür sorgen, dass aus dem Vektor `Prewt` ein Faktor wird. Danach können wir verschiedene Modelle aufstellen:

6 Varianzanalyse

```
range(Prewt)
[1] 70.0 94.9
prew<-cut(seq(70,95,5),include.lowest=T)

behandlung<-aov(Postwt~Treat)
gewicht<-aov(Postwt~prew)
beh.und.gew<-aov(Postwt~Treat+prew)
beh.gew.interaktion<-aov(Postwt~Treat*prew)
```

Wenn wir nun eine varianzanalytische Auswertung dieser vier Modelle vornehmen, ergeben sich für die Faktoren folgende p-Werte:

```
anova(behandlung)
Treat 0.0004443 ***
```

```
anova(gewicht)
prew 0.0035 **
```

```
anova(beh.und.gew)
Treat 9.386e-05 ***
prew 0.001147 **
```

```
anova(beh.gew.interaktion)
Treat 7.367e-05 ***
prew 0.0009043 ***
Treat:prew 0.2049314
```

Wir sehen hier, dass die p-Werte bei der einfaktoriellen Analyse am größten sind, bei der zweifaktoriellen Analyse, bei der nur die Haupteffekte betrachtet wurden, ergaben sich schon kleinere p-Werte und bei der Betrachtung der Haupteffekte der beiden Faktoren und ihrer Wechselwirkung erhält man die kleinsten p-Werte und somit auch die höchsten Signifikanzen. Die Frage ist nun allerdings, ob das letzte Modell denn auch dasjenige ist, durch das die Zielvariable am besten beschrieben wird.

◇

Dieses gerade im Beispiel aufgetauchte Problem tritt eigentlich bei jeder merhfaktoriellen Varianzanalyse auf: Woher sollen wir wissen, welches Modell die Situation wirklich am besten beschreibt? Insbesondere wenn man sehr viele Faktoren vorliegen hat, bei denen es zu überprüfen gilt, ob sie signifikanten Mittelwerteeinfluss auf die Zielvariable haben, kann man kaum jede mögliche Zusammenstellung von Faktoren und Wechselwirkungen testen und schließlich vergleichen, wobei ja zusätzlich nicht ganz klar ist, wie so ein Vergleich vonstatten gehen sollte.

Nun gibt es ein paar Funktionen in R mit denen man ein Ausgangsmodell sukzessive erweitern bzw. verkleinern kann und die gleichzeitig die Güte der Anpassung jedes Modells überprüfen. Diese Überprüfung basiert auf Akaikes Informationskriterium, das mithilfe der Loglikelihoodfunktion die durch das Modell erklärten Werte mit den tatsächlichen Werten der Zielvariablen vergleicht. eine gute Anpassung des Modells an die Variable ist erreicht, wenn der AIC-Wert (**A**kaike's **I**nformation **C**riterion) niedrig wird. Demnach müssten wir bei einem Datensatz, den wir mithilfe einer Varianzanalyse untersuchen wollen, für jedes durch die Daten mögliche Modell den AIC berechnen und diese dann miteinander vergleichen. Das klingt nach sehr viel Arbeit, und das wäre es auch, wenn wir nicht - wie oben angedeutet - auf Funktionen zurückgreifen könnten, die automatisch ein Ausgangsmodell schrittweise erweitern oder verkleinern und jeweils den AIC berechnen. Ein solches Anfangsmodell erhalten wir, indem wir für jeden Faktor einzeln eine ANOVA durchführen, denjenigen Faktor F mit dem kleinsten p-Wert, also der höchsten Signifikanz auswählen und schließlich das Modell

```
anfangsmodell<-aov(Zielvariable~F)
```

erstellen. Mit `add1` kann nun schrittweise überprüft werden, wie sich die Güte des Modells ändert, wenn man jeweils einen einzelnen Faktor hinzufügt, wobei hier natürlich gilt: Je kleiner der AIC-Wert wird, desto besser. Der Faktor, bei dem der kleinste Wert steht, sollte dem Modell hinzugefügt werden, wenn der AIC-Wert kleiner als der des Ausgangsmodells ist (sonst hätten wir ja keine Verbesserung erreicht). Das geschieht dann mit der Funktion `update`.

add1

update

Beispiel 6.7. Wir bleiben beim obigen Beispiel des Datensatzes `anorexia` aus `MASS`. Wir haben im vorigen Beispiel schon gesehen, dass der Faktor `Treat` höhere Signifikanz für die Zielvariable `Postwt` besitzt als der andere Faktor `prew`. Unser Anfangsmodell wird dementsprechend

```
behandlung<-aov(Postwt~Treat)
```

Nun wenden wir `add1` an, um zu sehen, was beim Hinzufügen einzelner anderer Faktoren geschieht. (Da wir hier nur einen weiteren Faktor haben, werden sich hier keine besonders spannenden Ergebnisse zeigen.)

```
add1(behandlung, .~.+prew)
```

```
Single term additions
```

```
Model:
```

```
Postwt ~ Treat
      Df Sum of Sq    RSS   AIC
< none >
prew   4    882.1 2783.0 277.1
```

Der AIC verbessert sich also um ungefähr zwölf Punkte, wenn wir den Faktor `prew`

hinzunehmen. Dabei haben wir in die Funktion als erstes das bestehende Modell eingesetzt, das erweitert werden soll, und danach angegeben, nach welcher Formel dies geschehen soll. Der Ausdruck `.~.` steht dabei für das vorhandene Modell (das erste Argument von `add1`), der Rest der Formel gibt vor, wie die Erweiterung mit welchen Größen stattfinden soll. Hier werden allerdings Wechselwirkungen zwischen den Faktoren nicht berücksichtigt. Diese Funktion kann uns also eigentlich auch nur weiterhelfen, wenn wir die Interaktionen zwischen den Faktoren nicht berücksichtigen wollen. Jetzt nehmen wir die Modellerweiterung vor, die uns die Auswertung von `add1` nahegelegt hat:

```
neues.modell<-update(behandlung, .~.+prew)
```

Unser `neues.modell` entspricht gerade dem Modell `beh.und.gew` aus dem vorigen Beispiel. Die Funktion `update` wird gerade dann wichtig, wenn man ein schon bestehendes umfangreiches Modell erweitern möchte, da man sich damit sehr viel Tipparbeit ersparen und das Einschleichen von Fehlern vermeiden kann.

◇

Mit der Funktion `drop1` kann man aus einem großen Modell, das alle oder sehr viele Faktoren und Wechselwirkungen der Faktoren umfasst, betrachten, wie sich der AIC ändert, wenn man jeweils einen Faktor aus dem Modell herausnimmt. Wird der AIC größer, heißt das, dass das Modell mit diesem Faktor besser ist als ohne. Wird der AIC umgekehrt bei der Entfernung eines Faktors kleiner, bedeutet dies, dass die Anpassung des Modells bei Nicht-Berücksichtigung dieses Faktors besser ist. Mit dieser Funktion kann man genau wie mit `add1` nur die Auswirkungen der einzelnen Faktoren auf ein bestehendes Modell überprüfen. Man müsste diese Funktionen dementsprechend sukzessive auf die jeweils mit `update` aufgebesserten Modelle anwenden, bis sich keine Verbesserung mehr erreichen lässt. Die Funktion `step` nimmt uns die Arbeit ab und führt diese ganzen Schritte selbsttätig aus, wobei hier auch die Interaktionseffekte der Faktoren einbezogen werden. In der Ausgabe erscheinen die unterschiedlichen Modelle mit dem dazugehörigen AIC-Wert und als letztes das Modell mit dem kleinsten AIC-Wert sowie die zugehörige Varianztabelle, wenn wir das Modell mit `aov` erstellt haben. Die gerade eingeführten Funktionen `add1`, `drop1`, `update` und `step` können auch alle auf Objekte angewendet werden, die mit `lm` erzeugt wurden, hier empfiehlt sich jedoch immer die Verwendung der Modellfunktion `aov`, da wir eine Varianzanalyse durchführen wollen und so schon die Varianztafeln erhalten bzw. mit dem zusätzlichen Argument `test="F"` die Testwerte bekommen.

drop1

step

Beispiel 6.8. Zurück zum vorigen Beispiel. Wir wählen wieder dasselbe Anfangsmodell und lassen nun mit `step` das beste Modell suchen:

```
step(behandlung, .~.*prew)
Start: AIC=288.96
Postwt ~ Treat
```

6 Varianzanalyse

```

          Df Sum of Sq   RSS   AIC
+prew    4     882.1 2783.0 277.1
< none >          3665.1 289.0
-Treat    2     919.0 4584.0 301.1

Step: AIC=277.13
Postwt ~ Treat + prew

          Df Sum of Sq   RSS   AIC
< none >          2783.0 277.1
+Treat : prew    8     464.6 2318.4 280.0
-prew           4     882.1 3665.1 289.0
-Treat          2     857.4 3640.4 292.5
Call:
  aov(formula = Postwt ~ Treat + prew)

Terms:
          Treat      prew Residuals
Sum of Squares 918.9869 882.0969 2782.9606
Deg. of Freedom      2        4        65

Residual standard error: 6.5433
Estimated effects may be unbalanced

```

Hier zeigt sich nun, dass wir die beste Anpassung erreichen, wenn wir nur die Haupteffekte der beiden Faktoren **Treat** und **prew** beachten, die Wechselwirkung der beiden jedoch nicht. Für dieses Modell wird dann die Varianzanalyse durchgeführt, was wir bereits im ersten Beispiel getan haben. Dort haben wir gesehen, dass der Faktor der Behandlungsmethode **Treat** einen höchst signifikanten und der Faktor des Gewichts zu Beginn der Studie **prew** hoch signifikanten Einfluss auf den Mittelwert des Gewichts zum Ende der Studie **Postwt** haben.

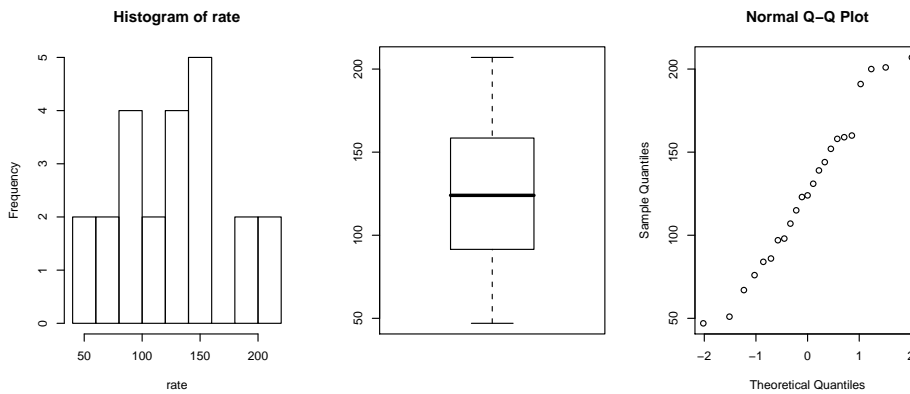
◇

Beispiel 6.9. Der Puromycin-Datensatz enthält Messungen zur Reaktionsschnelligkeit einer enzymatischen Reaktion von Zellen, die zuvor nicht bzw. mit unterschiedlichen Konzentrationen von Puromycin behandelt wurden. Es soll untersucht werden, ob die Reaktionsrate von der Behandlung oder der Dosierung abhängt. Die grafische Untersuchung zeigt, dass eine Normalverteilungsannahme getroffen werden kann. Bevor wir nun mit einer Varianzanalyse starten können, müssen wir aus dem Vektor **conc** der Konzentration einen Faktor erstellen. Dabei können wir z.B. die einzelnen Dosierungen einfach als Stufen des Faktors wählen, also

```
konz<-factor(conc)
```

Der Vektor **state**, der angibt, ob die Zellen behandelt wurden oder nicht, ist bereits

6 Varianzanalyse



ein Faktor; der Analyse steht also nichts mehr im Wege. Als erstes suchen wir den Faktor mit dem größten Mittelwerteinfluss auf die Reaktionsrate `rate` und bauen damit das Ausgangsmodell auf, das dann schrittweise verbessert wird.

```
anova(aov(rate~state))
Pr(>F) = 0.1221
anova(aov(rate~konz))
Pr(>F) = 3.304e-07 ***
modell.rate<-aov(rate~konz)
step(modell.rate,.~.*state)
```

In der Ausgabe der letzten Zeile erscheinen nun die durch `step` schrittweise verbesserten Modelle mit dem besten zum Schluss:

```
Step: AIC=112.84
rate ~ konz + state + konz:state

      Df Sum of Sq    RSS    AIC
< none >                1094.50  112.84
-konz:state    5    1204.35  2298.85  119.91
Call:
  aov(formula = rate ~ konz + state + konz:state)
```

```
Terms:
      konz    state konz:state Residuals
Sum of Squares  43604.80  3761.65    1204.35    1094.50
Deg. of Freedom      5      1      5      11
```

```
Residual standard error: 9.974969
Estimated effects may be unbalanced
```

Das beste Modell erhalten wir demnach, wenn wir die haupteffekte der beiden Fak-

6 Varianzanalyse

toren und deren Wechselwirkungseffekt berücksichtigen. Dafür ergibt sich nun

```
anova(aov(rate~konz+state+konz:state))
```

Analysis of Variance Table

Response: rate

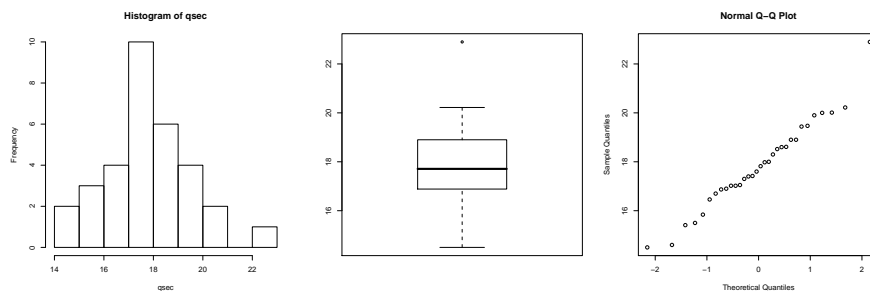
	Df	Sum Sq	Mean Sq	F value	Pr(> F)	
konz	5	43605	8721	87.6478	1.758e - 08	***
state	1	3762	3762	37.8055	7.222e - 05	***
konz : state	5	1204	241	2.4208	0.103	
Residuals	11	1094	99			

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

In diesem Modell sind beide Faktoren `konz` und `state` höchst signifikant für die Reaktionsrate, die Interaktion der beiden kann bei einem p-Wert von mehr als 10% nicht als signifikant bezeichnet werden.

◇

Beispiel 6.10. Der Datensatz `mtcars` enthält charakteristische Kenngrößen von 32 Automobilen aus den Jahren 1973 und 1974. Eine davon ist die Zeit, die benötigt wird, um aus dem Stand heraus eine Viertelmeile zurückzulegen (`qsec`). Wird diese Größe im Mittel von den anderen beeinflusst? Und wenn ja, wie stark? Zuerst überprüfen wir, ob eine Normalverteilung vorliegen könnte, dann wandeln wir die metrischen Größen in Faktoren um und untersuchen schließlich, welcher der Faktoren den größten Mittelwert-einfluss besitzt. Die Einteilung der Stufen der Faktoren sollte möglichst so geschehen, dass sie ungefähr gleich stark besetzt sind, um eine annähernde Gleichberechtigung der Faktoren zu erreichen.



```
miles<-cut(mpg,seq(10,35,5))
zyl<-factor(cyl)
hub<-cut(displ,seq(70,490,70))
ps<-cut(hp,c(seq(50,200,50),350))
umdr<-cut(drat,c(2.75,3,3.15,3.75,4,5))
gew<-cut(wt,c(seq(1.5,3,0.5),3.2,3.5,3.8,4.3,5.5))
```

6 Varianzanalyse

```

vs<-factor(vs)
sch<-factor(am)
gang<-factor(gear)
gas<-factor(carb)

anova(aov(qsec~miles)) 0.04382 *
anova(aov(qsec~zyl)) 0.001955 **
anova(aov(qsec~hub)) 0.03286 *
anova(aov(qsec~ps)) 0.000616 ***
anova(aov(qsec~umdr)) 0.6144
anova(aov(qsec~gew)) 0.2472
anova(aov(qsec~vs)) 1.030e-06 ***
anova(aov(qsec~sch)) 0.2057
anova(aov(qsec~gang)) 0.0005897 ***
anova(aov(qsec~gas)) 0.006064 **

```

Die Größe `vs` besitzt hier den größten Mittelwerteinfluss, wir wählen sie also als Grundlage für unser Modell und erweitern dieses entsprechend und werten das beste Modell aus.

```

autos<-aov(qsec~vs)
step(autos, .~. * miles * zyl * hub * ps * umdr * gew * sch * gang * gas)
anova(aov(formula = qsec ~ ps + gew + umdr + gas + miles + gew:umdr))
Analysis of Variance Table

```

```

Response: qsec

```

	Df	Sum Sq	Mean Sq	F value	Pr(> F)	
ps	3	45.061	15.020	167.4085	1.958e-05	***
gew	7	29.403	4.200	46.8157	0.0002928	***
umdr	4	0.471	0.118	1.3132	0.3788069	
gas	5	13.757	2.751	30.6647	0.0009305	***
miles	4	4.329	1.082	12.0615	0.0088231	**
gew : umdr	3	5.519	1.840	20.5040	0.0030705	**
Residuals	5	0.449	0.090			

Wir sehen, dass die Anzahl der Pferdestärken, das Gewicht des Wagens und die Anzahl der Vergaser(?) höchste Signifikanz für `qsec` besitzen und der Verbrauch (`miles`) und die Wechselwirkung zwischen Gewicht und Umdrehungszahl der Hinterachse hohe Signifikanz. Weiter fällt auf, dass das Merkmal, auf dem unser Ausgangsmodell aufgebaut war, in diesem Modell nicht mehr einbezogen wird.

◇

7 Regressionsanalyse

In diesem Kapitel schauen wir uns ein von der Systematik her ähnliches Problem an, wie im vorigen, das allerdings einen völlig anderen Auswertungsansatz vornimmt. Wir wollen hier eine Zielgröße durch eine oder mehrere (metrische) Ausgangsvariablen erklären. Der einfachste Fall ist der, dass wir die Zielvariable Y linear durch eine Ausgangsvariable X erklären wollen. Bei einem gegebenen Datensatz mit Wertepaaren $(x_1, y_1), \dots, (x_n, y_n)$ zu X und Y machen wir für jedes $1 \leq i \leq n$ den Ansatz

$$y_i = \alpha + \beta x_i + \varepsilon_i,$$

wobei die ε_i , $1 \leq i \leq n$, üblicherweise als unabhängig und identisch $\mathcal{N}(0, \sigma^2)$ -verteilt angenommen werden. (Wenn wir ein Modell erstellt haben, müssen wir überprüfen, ob die Residuen ε_i auch tatsächlich normalverteilt sind.) Wir unterstellen bei diesem Modell also, dass Y durch eine lineare Transformation aus X hervorgeht, wobei sich (z.B. durch Messfehler, störende Umwelteinflüsse etc.) gewisse Abweichungen vom linearen Zusammenhang ergeben können, die hier durch die ε_i ausgedrückt werden. Die Parameter α und β , die den linearen Zusammenhang eindeutig bestimmen, sind uns dabei unbekannt und müssen folglich geschätzt werden. Dabei greift man auf die Methode der kleinsten Quadrate zurück und wählt als Schätzer $\hat{\alpha}, \hat{\beta}$ für α, β diejenigen Werte, die die Summe der quadratischen Abweichungen unseres linearen Ansatzes von den tatsächlichen Werten

$$SS_{res} = \sum_{i=1}^n (y_i - (\alpha + \beta x_i))^2$$

minimieren. Die gesuchten Schätzer ergeben sich hieraus zu

$$\begin{aligned}\hat{\alpha} &= \bar{y} - \hat{\beta} \bar{x}, \\ \hat{\beta} &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2},\end{aligned}$$

mit den arithmetischen Mitteln $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ und $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$. Wenn wir die Varianz der ε_i schätzen wollen, können wir dies mithilfe von $\frac{SS_{res}}{n-2}$ machen. Analog verfährt man bei mehreren Ausgangsvariablen bzw. Regressoren X_1, \dots, X_k , $k \in \mathbb{N}$.

Nun soll einerseits überprüft werden, ob das aufgestellte Modell die Zielvariable erklären kann, also ob die Regressoren (zusammen) einen signifikanten Einfluss auf Y haben.

Dazu wird wieder eine Varianzzerlegung vorgenommen:

$$\begin{aligned}
 SQT &= SQE + SQR \\
 \text{mit } SQT &= \sum_{i=1}^n (y_i - \bar{y})^2, \\
 SQE &= \sum_{i=1}^n (\hat{y}_i - \bar{y})^2, \\
 SQR &= \sum_{i=1}^n (y_i - \hat{y}_i)^2, \\
 \hat{y}_i &= \hat{\alpha} + \sum_{j=1}^k \beta_j x_{ji}, \\
 \bar{y} &= \frac{1}{n} \sum_{i=1}^n y_i.
 \end{aligned}$$

Hierbei steht SQE gerade für die durch den Regressionsansatz erklärte Streuung, und SQR ist die Summe der quadrierten Residuen, also die Reststreuung. x_{ji} sei der i -te Wert des Regressors X_j , β_j der zugehörige Koeffizient. Will man nun überprüfen, ob der Ansatz eines solchen Modells brauchbar ist (Hypothesen: $H_0: \beta_1 = \dots = \beta_k = 0$ gegen $H_1: \exists l: \beta_l \neq 0$), betrachten wir wieder den Quotienten der (durch die Anzahl ihrer Freiheitsgrade) gemittelten Streuungen SQE und SQR und erhalten somit wieder einen F-Test. Schließlich wird noch für jede einzelne Ausgangsvariable getestet, ob sie signifikanten Einfluss auf Y besitzt, also ob ihr Koeffizient im Modell 0 ist. Da wir von Normalverteilungen ausgehen, ergibt sich bei diesem Testproblem wieder ein t-Test.

Einen weiteren Anhaltspunkt für die Güte unseres Modells erhalten wir durch das **Bestimmtheitsmaß**

$$R^2 = \frac{SQE}{SQR},$$

das angibt, welcher Anteil der Gesamtstreuung durch die Regression erklärt wird. Je höher dieser Anteil ist, desto besser ist auch das Modell. Wenn man hier noch die Freiheitsgrade der Summen der Quadrate berücksichtigt, gelangt man zu einem angepassten Wert für R^2 (engl.: adjusted R^2).

Wie wir ein entsprechendes Modell aufstellen, haben wir im vorigen Kapitel bereits gesehen: Hier verwenden wir die Funktion `lm` und geben die Formel wie bereits gesehen mithilfe von `~`, `+`, `*` etc. ein. Mit den Funktionen `add1`, `drop1`, `update` und `step` können wir ein Ausgangsmodell schrittweise verändern und verbessern, wobei `step` uns als Ergebnis das Modell liefert, das den kleinsten AIC-Wert besitzt. Wenden wir auf ein mit `lm` erstelltes Modell die Funktion `summary` an, erhalten wir die Auswertung des Modells mit einigen Tests. Für jede Modellkomponente wird mit einem t-Test (Normalverteilung!) überprüft, ob die Komponente in das Modell eingeht oder nicht, also ob der zu

dieser Komponente gehörige Koeffizient im Modell Null ist oder nicht. An dieser Stelle müssten wir eigentlich überprüfen, ob die Werte der Zielvariablen wirklich aus Normalverteilungen hervorgegangen sind, allerdings ergibt sich dabei ein Problem: Für jedes Y_i ist der Mittelwert der Normalverteilung gegeben durch $\alpha + \beta X_i$, wodurch sich natürlich unterschiedliche Verteilungen ergeben. Wir können also nicht anhand der Zielvariable die Normalverteilung überprüfen, wir können lediglich nach der Modellierung untersuchen, ob - wie unterstellt - die Residuen normalverteilt sind.

Beispiel 7.1. Wir können den Datensatz `swiss` per Regressionsanalyse untersuchen. Die Zielvariable ist wie in den Übungsaufgaben `Fertility`. Die Einflussvariablen können direkt verwertet werden, ohne sie erst in Faktoren umwandeln zu müssen. Mit `step` lassen wir uns ein gut angepasstes Modell (nach Akaikes Informationskriterium) ausgeben. Wir geben zunächst nur die Haupteffekte der Ausgangsvariablen ein, die Wechselwirkungen können wir später noch hinzufügen. Hierbei ist anzumerken, dass zwischen den Variablen, wenn diese signifikant in das Modell eingehen, immer eine gewisse Wechselwirkung zu verzeichnen ist.

```
step(lm(Fertility~Agriculture+Examination+Education+Catholic
        +Infant.Mortality))
```

liefert uns als Modell mit dem kleinsten AIC-Wert folgendes:

```
fert.modell<-lm(formula=Fertility~Agriculture+Education+Catholic
                +Infant.Mortality)
```

Mit `summary` können wir nun dieses Modell auswerten:

```
summary(fert.modell)
```

Call:

```
lm(formula=Fertility~Agriculture+Education+Catholic
    +Infant.Mortality)
```

Residuals:

Min	1Q	Median	3Q	Max
-14.6765	-6.0522	0.7514	3.1664	16.1422

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	62.10131	9.60489	6.466	8.49e - 08	***
Agriculture	-0.15462	0.06819	-2.267	0.02857	*
Education	-0.98026	0.14814	-6.617	5.14e - 08	***
Catholic	0.12467	0.02889	4.315	9.50e - 05	***
Infant.Mortality	1.07844	0.38187	2.824	0.00722	**

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

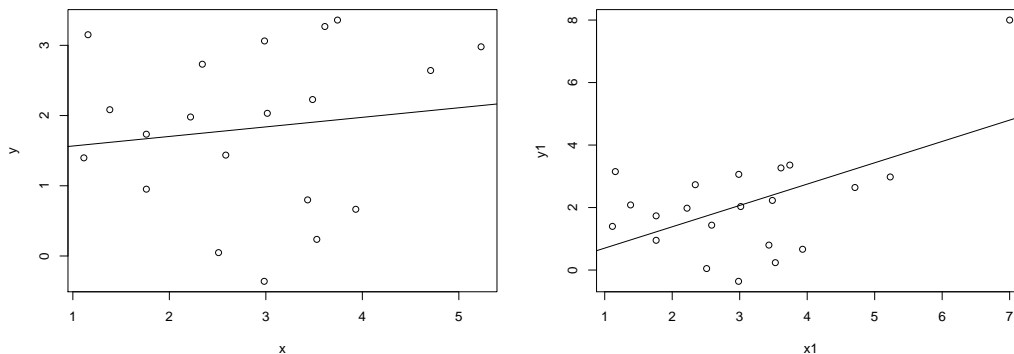
Residual standard error: 7.168 on 42 degrees of freedom
 Multiple R-squared: 0.6993, Adjusted R-squared: 0.6707
 F-statistic: 24.42 on 4 and 42 DF, p-value: 1.717e-10

Wir sehen nun, dass in dieses Modell alle berücksichtigten Variablen signifikant eingehen (das sind die Ergebnisse der t-Tests mit Nullhypothese, dass der jeweilige Koeffizient gleich Null ist). Schauen wir uns die Ausgabe genauer an: Sie beginnt mit den fünf Kennzahlen der Residuen, dem Minimum und Maximum, dem ersten und dritten Quartil und dem Median. Damit kann man schon einmal grob überprüfen, ob die Residuen normalverteilt sein könnten: Der Median sollte (ungefähr) bei Null liegen, die beiden Quartile im Vergleich zu den Extrema nah an Null und schließlich sollten die Quartile bzw. die Extrema annähernd symmetrisch zum Median liegen. Grob trifft das auch hier zu. Als nächstes erscheinen in der Ausgabe die Koeffizienten, die die einzelnen Variablen im Modell besitzen bzw. bei `Intercept` die Werte des Achsenabschnitts α . Als erstes wird jeweils ein geschätzter Wert für den entsprechenden Koeffizienten angegeben, gefolgt von (geschätzten) Standardabweichung der Schätzung des Koeffizienten und vom Wert der t-Statistik. Als letztes steht in jeder Zeile der p-Wert des durchgeführten Tests (der beim Achsenabschnitt nicht sonderlich sinnvoll ist, aber dennoch mit angegeben wird). In den letzten drei Zeilen der Auswertung werden die Ergebnisse der Gesamtanpassung und deren Auswertung mittels eines F-Tests aufgelistet. Es beginnt mit dem Standardfehler beim Schätzen der Residuen, gefolgt von den R^2 -Werten. Die letzte Zeile gibt den Wert der F-Statistik beim F-Test und den zugehörigen p-Wert an. Dieser ist hier sehr klein, die Linearkombination der Variablen (sprich: unser Modell) besitzt also höchst signifikanten Erklärungswert für die Zielvariable `Fertility`.

◇

Bis jetzt haben wir uns nur angesehen, wie die Modellierung und die Auswertung vonstatten gehen, allerdings wurde schon zuvor angedeutet, dass hier auch die Residuen unbedingt untersucht werden müssen. Wir können hier zum einen grafische Methoden zur Hilfe nehmen, zum anderen auch Tests auf Verteilungen. Bei hinreichend langen Stichproben kann man zum Beispiel den Kolmogorov-Smirnov-Test anwenden, ansonsten bietet sich auch der χ^2 -Anpassungstest an. Es sei `modell` das ausgewertete Modell. Die Residuen erhält man daraus mithilfe der Funktion `resid` (ebenso wie man die durch das Modell erklärten Werte mit der Funktion `fitted` erhält). Im wesentlichen wendet man nun die üblichen Methoden an: Man erstellt Streudiagramme und einen Normal-QQ-Plot, um sich einen Überblick über die Verteilung der Residuen zu verschaffen. Wir können natürlich diese Grafiken einzeln selber erstellen, leichter und schneller erhalten wir jedoch gleich mehrere solche Grafiken, wenn wir einfach das Modell in die Funktion `plot` einsetzen. Es werden dann vier Grafiken erzeugt, weshalb es sich hier empfiehlt, vorher mit `par(mfrow=c(2,2))` dafür zu sorgen, dass man alle vier zusammen betrachten und vergleichen kann. Die erste Grafik ist dabei ein Streudiagramm der Werte der

Residuen gegen die theoretischen Werte der Zielvariable, das zusätzlich eine Ausgleichskurve enthält. Das gibt uns die Möglichkeit, zu schauen, ob hinter den Abweichungen eine Systematik steckt. Wenn ja, deutet dies darauf hin, dass wir noch nicht das richtige Modell gefunden haben, da die Residuen wohl nicht normalverteilt sind (jedenfalls nicht mit derselben Varianz). Die zweite Grafik ist der Normal-QQ-Plot der standardisierten Residuen. Die dritte Grafik ist wieder ein Streudiagramm, diesmal der der Wurzel aus dem Betrag der standardisierten Residuen gegen die theoretischen Werte der Zielvariablen gemäß des Modells. Beim ersten Streudiagramm kann man gut überblicken, ob die Residuen symmetrisch zu Null sind, beim zweiten kann man die Größe der Residuen und deren Entwicklung gut nachvollziehen. Auch hier gilt: Lässt sich ein Muster erkennen (z.B. (strenge) Monotonie), spricht das eher gegen identische Verteilungen der Residuen gemäß einer Normalverteilung. In der letzten Grafik schließlich werden die standardisierten Residuen gegen ihren Hebel (leverage) abgetragen. Dahinter steckt folgende Idee: Bei einem Datensatz, bei dem fast alle Werte in einer Punktwolke liegen und nur wenige Außerhalb davon, haben diese wenigen Werte - dank einer gewissen Hebelwirkung - größeren Einfluss auf das Modell und die Steigung der Regressionsgeraden, wie die Grafiken zeigen. Dort wurde einem Datensatz ein einziger Punkt hinzugefügt, der außerhalb der allgemeinen Punktwolke der übrigen Werte liegt, wodurch die Steigung der Regressionsgeraden des entsprechenden linearen Modells deutlich erhöht wird. Es kann sogar geschehen, dass erst durch solche „Ausreißer“ ein linearer Zusammenhang oder eine von Null verschiedene Steigung entstehen. Daher ist es also interessant, sich die Residuen



in Relation zum Hebel der zugehörigen Werte anzusehen. Hier sollten die Werte wieder nah bei Null liegen, insbesondere die bei größerem Hebel, und es sollte keine Systematik hinter den Residuen erkennbar sein. In diese Grafik sind mehrere Linien eingefügt: Eine Ausgleichskurve der Residuen und vier gestrichelte Linien, durch die man die Größe der sogenannten Cook-Distanz ablesen bzw. einschätzen kann. Mit dieser Distanz wird - grob gesagt - der Einfluss jedes einzelnen Punktes auf das Modell gemessen, sprich, wie sehr sich das Modell änderte, wenn man den betrachteten Punkt entfernte. Das ist insbesondere interessant für Punkte mit einem betragsmäßig hohen Residuum oder einem großen Hebel. Wenn der Wert dieser Distanz für einen Punkt größer als 1 ist, sollte man ihn

sich genauer anschauen und die Veränderung des Modells bei Entfernung dieses Punktes untersuchen. Ab Distanzwerten von 0.5 werden Betrachtungen dieser Art sinnvoll.

Beispiel 7.2. Für den Datensatz `swiss` haben wir im vorhergehenden Beispiel bereits ein Modell erstellt und ausgewertet, müssen jedoch die Residuen noch gründlicher untersuchen. Da die Stichprobenlänge jeweils 47 beträgt und somit lang genug ist, können wir einen Kolmogorov-Smirnov-Test durchführen. Dabei wollen wir überprüfen, ob die Residuen normalverteilt sind mit Erwartungswert 0 und der Standardabweichung σ , die wir einfach durch die Wurzel der Stichprobenvarianz der Residuen schätzen. Dieser Wert ist in der durch `summary` herausgegebenen Auswertung in der drittletzten Zeile enthalten und beträgt in diesem Fall 7.168. Wir geben also ein:

```
ks.test(resid(fert.modell),pnorm,sd=7.168)
```

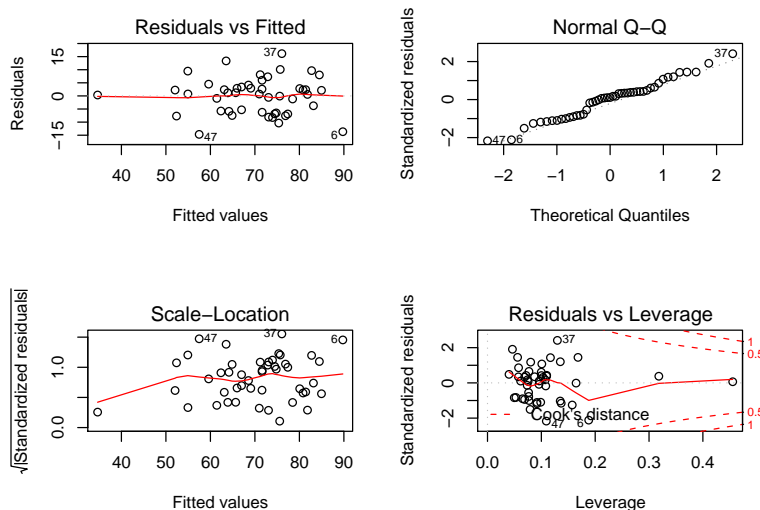
One-sample Kolmogorov-Smirnov test

```
data: resid(fert.modell)
D = 0.0945, p-value = 0.7595
alternative hypothesis: two-sided
```

Hier würden wir die Nullhypothese, dass die Residuen $\mathcal{N}(0, 7.168)$ -verteilt sind, nicht ablehnen, was für unser Modell schon einmal gute Nachrichten sind. Nun zur grafischen Analyse:

```
plot(fert.modell)
```

liefert die oben beschriebenen vier Grafiken. Bei der ersten Grafik sehen wir, dass die



Residuen relativ symmetrisch und ohne erkennbares Muster um die Null streuen. Dennoch gibt es paar besonders hohe bzw. niedrige Werte unter den Residuen. Diese sind durch ihre Nummer im Datensatz gekennzeichnet: Die Werte der Zeilen 6, 37 und 47 weichen also stark von den durch das Modell erklärten Werten ab. Im Normal-QQ-Plot liegen die Punkte ungefähr auf einer Geraden, zugegeben nicht unbedingt auf der Geraden, die gestrichelt eingezeichnet ist. Da diese Gerade durch die Punkte der ersten und dritten Quantile festgelegt wird, kann sie ohnehin nur als vage Richtschnur betrachtet werden, da ja gerade diese beiden Punkte nicht unbedingt die besten Repräsentanten für die allgemeine Tendenz sein müssen. Die Punkte zeigen keine perfekte Gerade, unser Modell ist also eventuell noch nicht ganz den Voraussetzungen genügend. Die letzten beiden Grafiken vermitteln im Grunde denselben Eindruck wie die beiden ersten: Die Residuen könnten normalverteilt sein. Hinter den Abweichungen ist kein klares Muster zu erkennen, die Werte sind aber auch nicht perfekt angepasst. In allen vier Grafiken sehen wir, dass es im Ausgangsdatsatz einen Wert der Zielvariablen gibt, der deutlich kleiner ist als alle anderen, könnte also einen größeren Einfluss auf das Modell besitzen, doch in der letzten Grafik sehen wir, dass die Cook-Distanz dieses Wertes nahezu 0 ist, d.h. durch ihn wird das Modell nicht übermäßig beeinflusst.

◇

Beispiel 7.3. Wir können auch den Datensatz `mtcars` bzw. die darin enthaltene Variable `qsec` per Regression untersuchen. Die dahinterstehende Frage ist hier also, ob und wie sich die benötigte Zeit für das Fahren einer Viertelmeile (aus dem Stand heraus) durch die anderen Variablen ausgedrückt werden kann. Wir suchen zunächst das Modell mit dem kleinsten AIC-Wert und wenden dieses dann aus:

```
step(lm(qsec mpg+cyl+disp+hp+drat+wt+vs+am+gear+carb))
autos<-lm(formula = qsec ~ cyl + disp + wt + vs + am + carb)
summary(autos)
```

Call:

```
lm(formula = qsec ~ cyl + disp + wt + vs + am + carb)
```

Residuals:

Min	1Q	Median	3Q	Max
-1.052236	-0.470154	-0.006656	0.293782	2.276464

Coefficients:

Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	18.611144	1.694356	10.984	4.67e - 11 ***
cyl	-0.369984	0.239525	-1.545	0.13500
disp	-0.008899	0.003737	-2.381	0.02520 *
wt	1.475086	0.391206	3.771	0.00089 ***
vs	0.968162	0.524030	1.848	0.07654 .
am	-0.902579	0.497398	-1.815	0.08160 .
carb	-0.434722	0.129112	-3.367	0.00246 **

7 Regressionsanalyse

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.733 on 25 degrees of freedom

Multiple R-squared: 0.8643, Adjusted R-squared: 0.8317

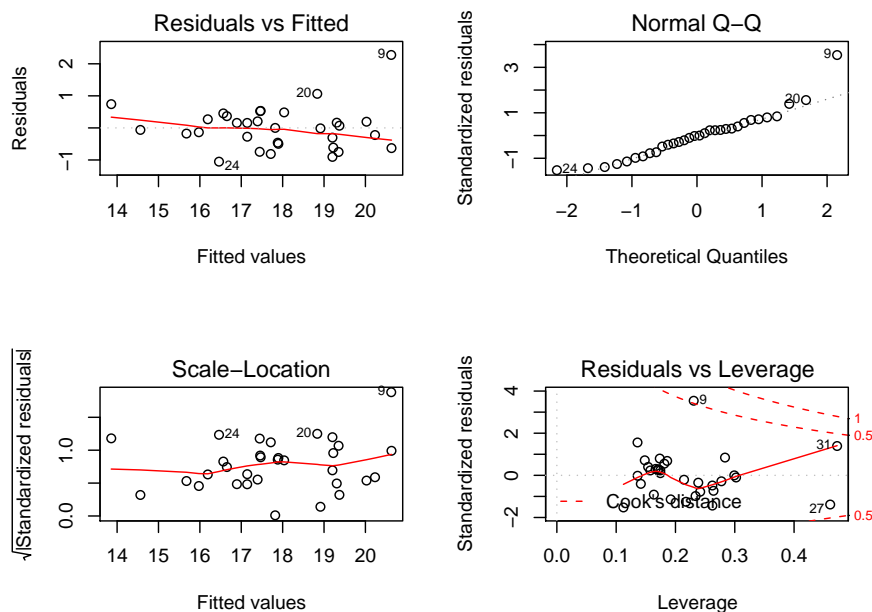
F-statistic: 26.54 on 6 and 25 DF, p-value: 1.076e-09

Wir sehen also, dass unser Modell einen höchst signifikanten Einfluss auf die Zielvariable besitzt, zur Beschreibung also sehr gut geeignet zu sein scheint. Bleibt zu prüfen, ob denn auch die entsprechenden Verteilungen vorliegen, so dass die Auswertungen mit t- bzw. F-Test gerechtfertigt sind.

```
ks.test(resid(autos),pnorm,sd=0.733)
```

p-value = 0.5077

Der Test liefert einen angenehm hohen p-Wert, allerdings ist unser Residuenvektor mit einer Länge von 32 eventuell ein wenig zu kurz, so dass der Testwert leicht verfälscht sein könnte. Weiter ergeben sich folgende vier Grafiken, die die Normalverteilungsan-



nahme für die Residuen rechtfertigen. Wir sehen hier aber auch, dass der durch Zeile 9 im Datensatz gegebene Punkt einen hohen Wert bei der Cook-Distanz besitzt (knapp über 0.5). Hier könnte es sich also lohnen, diesen Punkt aus dem Datensatz zu entfernen und für den reduzierten Datensatz noch einmal ein Modell zu erstellen.

7 Regressionsanalyse

```
auto<-mtcars[-9,]
autos1<-lm(formula = qsec ~ cyl + disp + wt + vs + am + carb)
summary(autos1)
```

Call:

```
lm(formula = qsec ~ cyl + disp + wt + vs + am + carb)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.95389	-0.37985	0.05677	0.28978	1.04849

Coefficients:

Estimate	Std. Error	t value	Pr(> t)		
(Intercept)	16.733688	1.279372	13.080	2.06e-12	***
cyl	-0.078501	0.182516	-0.430	0.67095	
disp	-0.009732	0.002699	-3.606	0.00142	**
wt	1.497426	0.281946	5.311	1.89e-05	***
vs	1.217297	0.381013	3.195	0.00389	**
am	-0.319873	0.377542	-0.847	0.40522	
carb	-0.521442	0.094700	-5.506	1.16e-05	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5282 on 24 degrees of freedom

Multiple R-squared: 0.9078, Adjusted R-squared: 0.8848

F-statistic: 39.39 on 6 and 24 DF, p-value: 2.867e-11

Hier zeigt sich nun, dass dieses Modell für den reduzierten Datensatz die Zielvariable Y besser erklärt, da sowohl beide R^2 -Werte größer als auch der p-Wert beim F-Test kleiner geworden sind. Weiter sehen wir auch, dass sich für ein paar der Regressoren größere Signifikanzen ergeben, einige aber auch sehr viel höhere p-Werte beim t-Test bekommen. Dieses Modell ist also möglicherweise in der Situation des verringerten Datensatzes nicht mehr das beste. Für die Residuen zeigt sich folgendes:

```
ks.test(resid(autos1),pnorm,sd=0.5282)
```

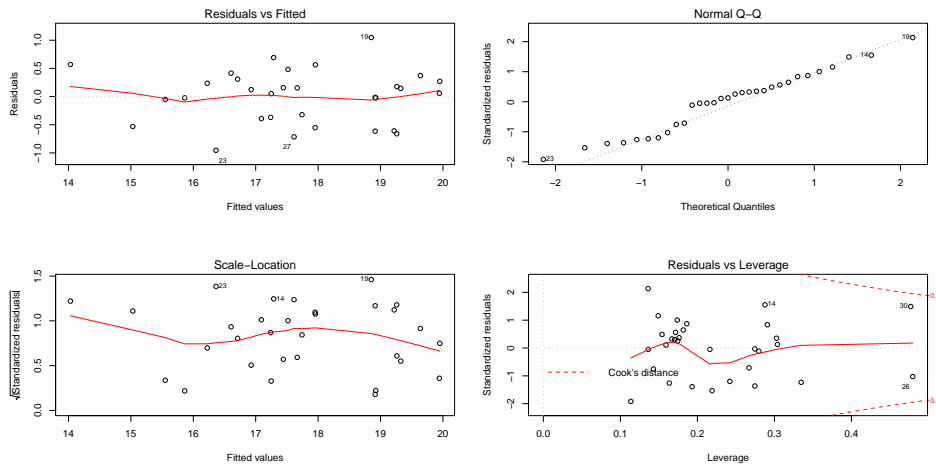
p-value = 0.5409

Eine Normalverteilungsannahme ist also nicht abzulehnen. Bei der grafischen Analyse ergibt sich dasselbe.

`plot(autos1)` Nun stellen wir ein neues Modell für den reduzierten Datensatz auf und verwenden dazu wieder die Funktion `step`:

```
step(lm(qsec~mpg+cyl+disp+hp+drat+wt+vs+am+gear+carb))
```

7 Regressionsanalyse



liefert

```
autos2<-lm(formula = qsec ~ mpg + disp + hp + wt + vs + gear + carb)
```

als das Modell mit dem kleinsten AIC-Wert. Die Analyse ergibt nun:

```
summary(autos2)
```

Call:

```
lm(formula = qsec ~ mpg + disp + hp + wt + vs + gear + carb)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.89952	-0.20207	0.03037	0.14508	0.66467

Coefficients:

Estimate	Std. Error	t value	Pr(> t)		
(Intercept)	16.563334	1.377279	12.026	2.12e-11	***
mpg	0.059108	0.033962	1.740	0.0952	.
disp	-0.006351	0.002703	-2.350	0.0277	*
hp	-0.005916	0.003632	-1.629	0.1169	
wt	1.317167	0.277272	4.750	8.67e-05	***
vs	1.430011	0.278523	5.134	3.35e-05	***
gear	-0.527822	0.207403	-2.545	0.0181	*
carb	-0.210529	0.128101	-1.643	0.1139	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4439 on 23 degrees of freedom

7 Regressionsanalyse

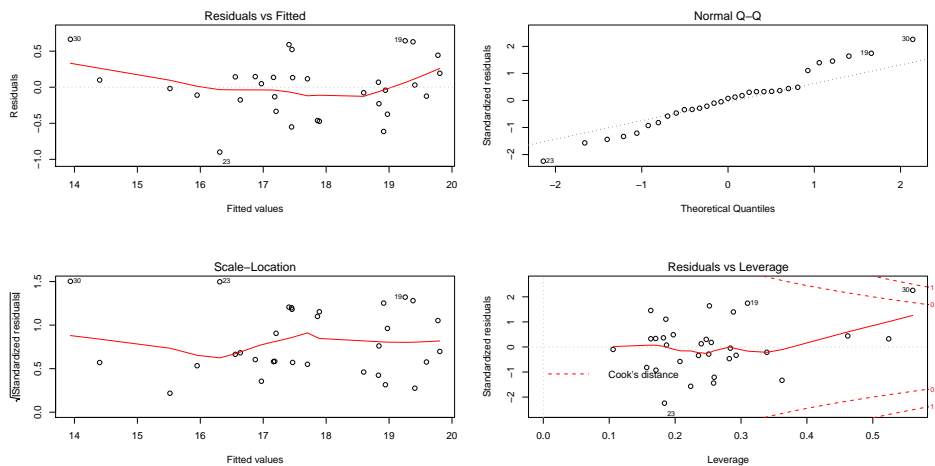
Multiple R-squared: 0.9376, Adjusted R-squared: 0.9186
F-statistic: 49.39 on 7 and 23 DF, p-value: 2.299e-12

Dieses Modell beschreibt die Zielvariable `qsec` besser als das vorige, wie man an den R^2 -Werten und dem F-Test erkennen kann. Für die Residuen ergibt sich hier:

```
ks.test(resid(autos2), pnorm, sd=0.4439)
```

```
p-value = 0.4885
```

```
plot(autos2)
```

 Die Grafiken zeigen hier, dass die Normalverteilungsannahme wohl ge-

rechtfertigt ist, allerdings erhalten wir hier für einen weiteren Wert mit sehr hoher Cook-Distanz (fast 1!), müssten den Punkt also wieder gesondert betrachten... Insgesamt zeigt sich also, dass der durch Zeile 9 des Datensatzes gegebene Punkt das Modell deutlich beeinflusst.

◇

8 Schätzung und Simulation

In diesem Kapitel schauen wir uns ein paar Sachen an, die bisher etwas zu kurz gekommen sind: Schätzungen und Simulationen bzw. Kombinationen davon. Die Zeit reicht allerdings nicht für mehr als einen knappen Einblick.

Wollen wir bei einer bekannten Verteilung eine gewisse Größe schätzen, können wir dies tun, indem wir eine Stichprobe erzeugen und daraus einen Schätzer für die gesuchte Größe bilden. Dabei begegnen wir allerdings einem Problem: Durch die Beschränkung auf eine Stichprobe und das Schätzen der Größe aus dieser Stichprobe gehen Unsicherheiten in den Schätzwert ein. Ist die Stichprobenlänge frei wählbar, können wir durch Erzeugen einer großen Stichprobe einen Teil der Unsicherheit entfernen. Wir können auch mehrere (am besten sehr viele) Stichproben derselben Länge gemäß derselben Verteilung erzeugen, für jede den Schätzwert bilden und schließlich davon das arithmetische Mittel betrachten.

Beispiel 8.1. Gesucht sei die Varianz des Medians bei Stichprobenlänge 10 und Standardnormalverteilung. Wir können nun 100.000 Stichproben der Länge zehn erzeugen und für jede den Median berechnen, um daraus die Varianz zu schätzen. Dabei speichern wir die Stichproben in Form einer Matrix, da wir dann mithilfe der Funktion `apply` leichter die Medianbildung vornehmen können.

```
stichproben<-matrix(rnorm(1000000),ncol=10)
mediane<-apply(stichproben,1,median)
var(mediane)
[1] 0.1389901
```

Hier haben wir natürlich nur die Stichprobenvarianz der Mediane der Stichproben bestimmt, die allerdings ein erwartungstreuer Schätzer für die Varianz ist.

◇

Beispiel 8.2. Es soll der Wert von π geschätzt werden. Dabei kann man ausnutzen, dass der Flächeninhalt eines Kreises mit Radius 1 π ist. Um eine einfache Verteilung zu haben, gemäß derer man nun Zufallszahlen innerhalb des Kreises simulieren könnte, ziehen wir einfach ein Quadrat um den Kreis mit demselben Mittelpunkt (z.B. der Ursprung) und erzeugen darin Zufallspunkte gemäß der Gleichverteilung. Wenn man nun das Verhältnis der Punkte im Kreis zur Gesamtanzahl bildet, konvergiert dieses (nach dem Gesetz der großen Zahlen) gegen das Verhältnis der beiden Flächeninhalte $\frac{\pi}{4}$.

```
x<-runif(1000000,min=-1,max=1)
y<-runif(1000000,min=-1,max=1)
im.kreis<-sum(x^2+y^2<=1)
4*im.kreis/length(x)
[1] 3.141832
pi-4*im.kreis/length(x)
[1] -0.0002393464
```

Die aus den Zufallszahlen gewonnene Schätzung für π unterscheidet sich vom tatsächlichen Wert nur um -0.0002393464, kann also als gute Näherung genommen werden.

◇

Beispiel 8.3. Aufgabe 42 auf Blatt 13.

◇

Hat man eine Stichprobe gegeben, deren zugrundeliegende Verteilung unbekannt ist, und möchte dennoch eine Größe schätzen, die von der Verteilung abhängt, gestaltet sich das Vorgehen etwas schwieriger. Man muss dann auf Näherungen der Verteilung zurückgreifen, z.B. indem man die Verteilungsfunktion durch die empirische Verteilungsfunktion der Stichprobe annähert (wofür die Stichprobe lang sein sollte). Um auch hier mehrere Stichproben zu haben, für die man die Größe berechnet und schließlich Streuung der Schätzung angeben zu können, erzeugt man weitere Stichproben durch zufälliges Ziehen aus der bereits vorhandenen Stichprobe. Haben wir also eine Stichprobe der Länge n gegeben, erzeugen wir aus ihr durch Resampling R weitere Stichproben der Länge n . Die empirischen Verteilungsfunktionen der neuen Stichproben entsprechen dabei approximativ der uns unbekannt wirklichen Verteilungsfunktion. Dieses Verfahren heißt **Bootstrapping**, und kann in R mit der Funktion `boot` aus dem Paket `boot` durchgeführt werden (mit der ich leider noch nicht richtig umgehen kann). Wir können das Bootstrapping schrittweise selber durchführen, bekommen dann aber nicht automatisch die Schätzungen für die Standardfehler u.ä. Durch das Resampling können deutlich mehr der in der Ursprungsstichprobe enthaltenen Informationen ausgewertet werden und der Schätzwert für die gesuchte Größe wird besser.

boot

Beispiel 8.4. Im Datensatz `faithful` sind die Eruptionsdauern des Old Faithful Geysers (und die vorhergehenden Wartezeiten) enthalten. Wollen wir den Median der Verteilung der Eruptionen bestimmen, können wir ihn durch den Stichprobenmedian schätzen:

```
median(eruptions)
[1] 4
```

Diese Schätzung wird höchstwahrscheinlich nicht den richtigen Wert angeben. Wir nutzen nun die in der Stichprobe enthaltenen Informationen stärker aus, indem wir durch Ziehen mit Zurücklegen neue Stichproben bilden, die dieselbe Länge besitzen. Die Anzahl R der neuen Stichproben sollte hoch sein, mindestens im dreistelligen Bereich; wir

erstellen hier 1000 neue Stichproben.

```
length(eruptions)
[1] 272
stichproben<-matrix(sample(eruptions,272000,replace=T),nrow=1000)
```

Den Median können wir nun schätzen, indem wir für jede Stichprobe den Median berechnen und davon das arithmetische Mittel bilden.

```
mean(apply(stichproben,1,median))
[1] 3.984246
var(apply(stichproben,1,median))
[1] 0.005993384
```

Die Streuung unserer Schätzung können wir mithilfe der Stichprobenvarianz des Vektors der Mediane annähern.

◇

Eine weitere Art der Schätzung, die wir uns noch anschauen wollen, ist die Kern-dichteschätzung, bei der man versucht aus einer gegebenen Stichprobe die Dichte der unbekanntem Verteilung von $X = (X_1, \dots, X_n)$ zu schätzen. Dabei nutzt man aus, dass die Dichte $f_X(x)$ der Ableitung der Verteilungsfunktion $F_X(x)$ entspricht:

$$f_X(x) = \frac{d}{dx}F_X(x) = \lim_{h \rightarrow 0} \frac{1}{2h}P(x-h < X < x+h).$$

Die hier auftauchende Wahrscheinlichkeit ist natürlich auch unbekannt, sie kann jedoch durch den Anteil der Beobachtungen geschätzt werden, die im Intervall $(x-h, x+h)$ liegen, woraus sich als Schätzer $\hat{f}_X(x)$ für die Dichte ergibt:

$$\begin{aligned} \hat{f}_X(x) &= \frac{1}{2hn} |\{x_i : x_i \in (x-h, x+h), 1 \leq i \leq n\}| \\ &= \frac{1}{2hn} |\{x_i : x_i - h < x < x_i + h, 1 \leq i \leq n\}| \\ &= \frac{1}{2hn} |\{x_i : -1 < \frac{x-x_i}{h} < 1, 1 \leq i \leq n\}|. \end{aligned}$$

Die zweite Gleichung geht dabei auf die Überlegung zurück, dass wir statt um x auch um jedes x_i eine h -Umgebung legen und dann zählen können, in wie vielen der so entstandenen Intervalle x liegt. Die dritte Gleichung geht aus einer einfachen Umformung der beiden vorherigen Bedingungen hervor. Hier gilt nun, dass alle Punkte in der h -Umgebung von x denselben Beitrag zum Schätzer der Dichte liefern, wodurch dieser Schätzer zu einer Treppenfunktion wird. Da man aber einerseits häufig eine glattere Dichtefunktion haben möchte und andererseits Punkten, die (innerhalb des Intervalls) weiter von x entfernt liegen, weniger Einfluss verleihen möchte, kann man zusätzlich mit

einer Gewichtsfunktion w arbeiten.

$$\hat{f}_X(x) = \frac{1}{n} \sum_{i=1}^n \frac{1}{h} w\left(\frac{x - x_i}{h}\right)$$

Wenn wir $w(t) = 0.5 \cdot \mathbf{1}_{(-1,1)}(t)$ wählen, gelangen wir gerade zum vorherigen Ansatz für den Dichteschätzer zurück. Wählt man jedoch eine Dichte als Gewichtsfunktion, wird auch der Dichteschätzer eine glatte Funktion, wobei häufig die Dichte der Standardnormalverteilung verwendet wird. Die Gewichtsfunktion nennt man auch **Kernfunktion** und entsprechend den zugehörigen Dichteschätzer **Kerndichteschätzer**. Hierbei hängt der Kerndichteschätzer natürlich davon ab, wie groß man h wählt und welche Kernfunktion. Zwei bekannte Ansätze für h sind die von Scott und Siverman, die auf der Stichprobenvarianz und dem Interquartilsabstand basieren. Diese beiden Ansätze sind auch in der Funktion `density`, mit der man in R eine Dichteschätzung durchführen kann, eingebaut.

`density`

```
density(x,bw,kernel,...)
```

Dabei ist `x` der Datensatz, `bw` (Abkürzung für engl.: bandwidth) ist h und `kernel` ist die Kernfunktion. Standardmäßig wird als Kernfunktion die Dichte der Standardnormalverteilung gewählt, andere Möglichkeiten werden im Hilfetext beschrieben. Für die Bandbreite h wird automatisch `nrd0` (Silvermans Ansatz) verwendet; Scotts Ansatz kann als `nrd` für `bw` eingesetzt werden. Prinzipiell kann man jeden beliebigen Wert dafür verwenden (die meisten davon werden sich aber als wenig sinnvoll herausstellen). Bei größerem h wird natürlich die Dichteschätzung immer glatter, man büßt dabei aber auch viel mehr lokale Informationen ein, sprich: Die Schätzung wird deutlich ungenauer. Wenn wir `density` anwenden, erhalten wir als Ausgabe die sechs Kennzahlen der eingesetzten Stichprobe (Extrema, erstes und drittes Quartil, Median und arithmetisches Mittel) unter dem Namen `x` und die Werte des Kerndichteschätzers an diesen Stellen unter dem Namen `y`. Da wir damit häufig nicht viel anfangen können, sollten wir eine Grafik der Kerndichteschätzung anfertigen, indem wir sie in `plot` einsetzen.

Beispiel 8.5. Wir haben uns schon in einigen Beispielen und Aufgaben mit Andersons `iris`-Datensatz befasst und unter anderem untersucht, welche Verteilungen den Merkmalen zugrundeliegen könnten. Nun können wir dieser Fragestellung noch einmal mithilfe der Kerndichteschätzung nachgehen.

```
attach(iris)
density(Sepal.Length)
```

```
Call:
  density.default(x = Sepal.Length)
```

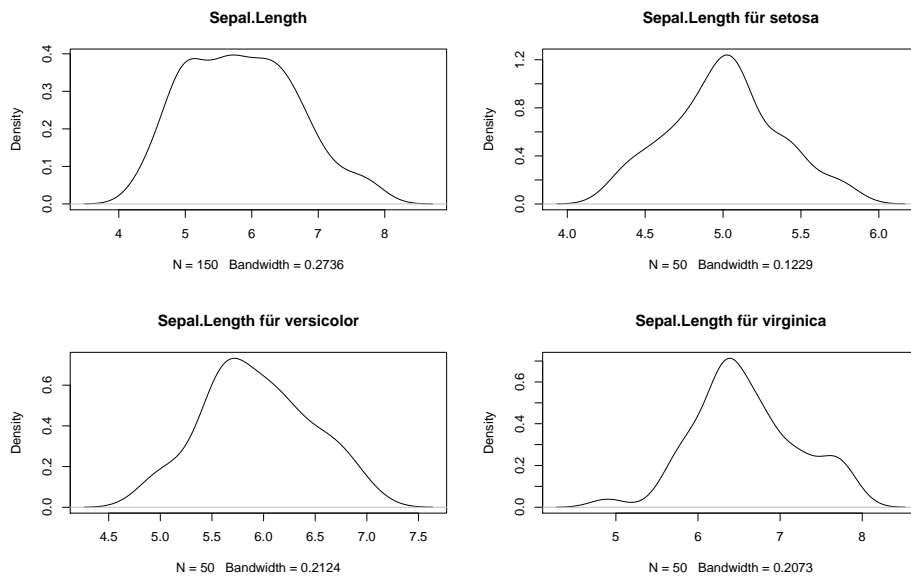
```
Data: Sepal.Length (150 obs.); Bandwidth 'bw' = 0.2736
```

8 Schätzung und Simulation

	x	y
Min.	: 3.479	Min. : 0.0001495
1st Qu.	: 4.790	1st Qu. : 0.0341599
Median	: 6.100	Median : 0.1534105
Mean	: 6.100	Mean : 0.1905934
3rd Qu.	: 7.410	3rd Qu. : 0.3792237
Max.	: 8.721	Max. : 0.3968365

```
par(mfrow=c(2,2))
plot(density(Sepal.Length),main="Sepal.Length")
plot(density(Sepal.Length[Species=="setosa"]),
     main="Sepal.Length für setosa")
plot(density(Sepal.Length[Species=="versicolor"]),
     main="Sepal.Length für versicolor")
plot(density(Sepal.Length[Species=="virginica"]),
     main="Sepal.Length für virginica")
```

Hier wurde das Merkmal `Sepal.Length` einmal für die Gesamtstichprobe und jeweils unterteilt nach den verschiedenen Spezies untersucht. Bei allen vier Grafiken zeigt sich, dass es sich um Normalverteilungen handeln könnte; allerdings sieht es etwas verzerrt aus. Man müsste also noch einmal nachsehen, wie sich die Bilder bei anderen Bandbreiten darstellen.



◇

Zum Abschluss schlagen wir den Bogen zum Anfang zurück und schauen uns die Auswertung geschichteter Stichproben an. Bei der Schichtung wird die Grundgesamt-

heit in k disjunkte Schichten G_1, \dots, G_k mit den Umfängen N_1, \dots, N_k zerlegt, wobei $\sum_{i=1}^k N_i = N$ gelte. Untersucht werde das Merkmal X , dessen Erwartungswert μ wir schätzen wollen. Dazu wird aus jeder Schicht eine Zufallsstichprobe $x_i = (x_{i1}, \dots, x_{in_i})$ gezogen mit nicht unbedingt derselben Länge. Ein erwartungstreuer Schätzer für den Erwartungswert μ erhalten wir offenbar mit

$$\hat{\mu} = \sum_{i=1}^k \frac{N_i}{N} \bar{x}_i,$$

wobei \bar{x}_i das arithmetische Mittel der Stichprobe aus der i -ten Schicht sei. Dieser Schätzer besitzt die Varianz

$$\text{Var}(\hat{\mu}) = \sum_{i=1}^k \frac{N_i^2}{N^2} \frac{\sigma_i^2}{n_i}$$

mit der Varianz σ_i^2 des Merkmals X in der i -ten Schicht. In Abhängigkeit von der Stichprobenlänge n_i ändert sich also auch die Varianz des Schätzers. Wählt man n_i proportional zum Anteil der Schicht an der Grundgesamtheit, ergibt sich ein Schätzer mit nicht optimaler Varianz. Einen solchen erhält man bei der Wahl der n_i durch

$$n_i = n \cdot \frac{N_i \sigma_i}{\sum_{i=1}^k N_i \sigma_i},$$

was man auch als optimale Aufteilung bezeichnet. Dabei ist n die Gesamtlänge der geschichteten Stichproben, also $n = \sum_{i=1}^k n_i$. Diese Aufteilung zeigt deutlich, dass es günstig ist, bei Schichten mit großer Streuung viele Beobachtungen und bei Schichten mit geringer Streuung nur wenige Beobachtungen zu machen.

Der Vorteil der Schichtung gegenüber einer einfachen Zufallsstichprobe besteht darin, dass wir mehr Informationen aus der Grundgesamtheit ausnutzen (wobei wir im Vorfeld auch mehr Informationen über die Gesamtheit benötigen). Die Schätzung des Erwartungswertes wird durch Schichtung bei optimaler Aufteilung daher besser.

Beispiel 8.6. Wir untersuchen wieder den `iris`-Datensatz bzw. das Merkmal `Sepal.Length`. Es soll insgesamt eine Stichprobe der Länge 25 gezogen werden, doch wir unterteilen den Vektor zunächst in Schichten. Der Datensatz ist natürlich nicht die eigentliche Grundgesamtheit, aber wir tun hier einfach mal so, als ob das so wäre. Die Einteilung der Schichten können wir zum Beispiel so vornehmen:

```
sp<-cut(Sepal.Length,4:8,include.lowest=T)
table(sp)->N
```

Der erste der beiden Vektoren ist ein Faktor, der jedem Eintrag in `Sepal.Length` die entsprechende Schicht zuordnet, der zweite enthält die Umfänge N_i der Schichten. Nun muss noch berechnet werden, wie groß die Zufallsstichproben aus den einzelnen Schichten sein sollten, sprich, wie die n_i zu wählen sind. Dazu benötigen wir zusätzlich die Streuungen σ_i^2 der Werte in den Schichten.

```
sapply(split(Sepal.Length,sp),var)->varianz
sqrt(varianz)->sigma
n<-25
n.i<-n*N*sigma/sum(N*sigma)
n.i
4.394017 10.513410 8.038250 2.054322
```

Wir sollten also aus der ersten Schicht 4 Werte ziehen, aus der zweiten 11, aus der dritten 8 und aus der vierten 2.

```
s1<-sample(Sepal.Length[sp=="(4,5)"],4)
s2<-sample(Sepal.Length[sp=="(5,6)"],11)
s3<-sample(Sepal.Length[sp=="(6,7)"],8)
s4<-sample(Sepal.Length[sp=="(7,8)"],2)

mittel1<-N[1]/sum(N)*mean(s1)
mittel2<-N[2]/sum(N)*mean(s2)
mittel3<-N[3]/sum(N)*mean(s3)
mittel4<-N[4]/sum(N)*mean(s4)
mittel1+mittel2+mittel3+mittel4
5.860955
```

Der geschätzte Erwartungswert ist demnach 5.860955. Wir können zum Vergleich eine Zufallsstichprobe der Länge 25 aus dem Gesamtvektor `Sepal.Length` ziehen und davon den Mittelwert bilden:

```
mean(sample(Sepal.Length,25))
[1] 5.728

mean(Sepal.Length)
[1] 5.843333
```

Der Vergleich mit dem Mittelwert von `Sepal.Length` zeigt, dass der Schätzer aus der geschichteten Stichprobe den Wert besser annähert als der einfache Mittelwert aus einer einfachen Zufallsstichprobe.

◇

Literaturverzeichnis

- [1] BRAUN, W. JOHN und MURDOCH, DUNCAN J.(2007). *A First Course in Statistical Programming with R*, Cambridge University Press.
- [2] CHAMBERS, JOHN M. und HASTIE, TREVOR J.(1992). *Statistical Models in S*, Wadsworth&Brooks/Cole.
- [3] DALGAARD, PETER(2002). *Introductory Statistics with R*, Springer, New York.
- [4] HANDL, ANDREAS. *Einführung in die Statistik mit R*, <http://www2.wiwi.uni-bielefeld.de/~frohn/Lehre/Statistik1/Skript/stat12b.pdf>.
- [5] KONOPKA, JÖRG. *Dokumentation des Statistik-Praktikums WS2000/01 und SS2001*, persönliche Unterlagen.
- [6] KRAUSE, ANDREAS und OLSON, MELVIN(2002). *The Basics of S-PLUS*, Springer, New York.
- [7] VENABLES, WILLIAM N. und RIPLEY, BRIAN D.(1997). *Modern Applied Statistics with S*, Springer, New York.