Datenanalyse und Graphik mit R

Kursunterlagen zu den

Rechnergestützten Übungen zur Statistik für Studierende der Biologie

Bernhard Klar

5. Mai 2008

Bernhard Klar
Institut für Stochastik
Fakultät für Mathematik, Universität Karlsruhe
Englerstr. 2, 76128 Karlsruhe

email: Bernhard.Klar@kit.edu
Homepage: http://www.mathematik.uni-karlsruhe.de/stoch/∼klar/

Inhaltsverzeichnis

Einleitung

In diesem Kurs verwenden wir R, ein mächtiges System zur Datenanalyse. Die Stärken von R liegen insbesondere im flexiblen Erstellen von Graphiken und in der interaktiven Auswertung von Daten.

R ist kein typisches Statistikpaket wie STATISTICA oder JMP. R ist eine nichtkommerzielle Implementierung von S, einer Programmiersprache, die statistische Auswertungen sehr vereinfacht. Eine kommerzielle Implementierung von S ist unter dem Namen S-PLUS von der Firma Insightful erhältlich. [1]

R ist für viele Betriebssysteme (Windows, Linux, MacOS,...) verfügbar und unter

http://cran.r-project.org

frei erhältlich. Dort findet man auch eine große Zahl von Software-Bibliotheken sowie Literatur zu R (siehe auch Anhang 2).

Obwohl (oder gerade weil) R kein typisches Statistikpaket ist, gibt es viele Gründe für die Benutzung von R in einem Statistik-Praktikum:

- R beinhalt eine Vielzahl von Funktionen zur Datenmanipulation, Datenanalyse und Präsentation. Aus diesen Komponenten kann der Benutzer genau die gewünschte Analyse erstellen.
- In R kann man auf alle erzeugten Objekte zugreifen und sie in weiteren Schritten analysieren. Dies ist in Ausgabe-orientierten Statistikprogrammen nicht oder nur eingeschränkt möglich.
- Der Aufruf beispielsweise eines unverbundenen einseitigen 2-Stichproben-t-Tests mit Welch-Korrektur zum Testen der Hypothese $H_0 : \mu_x - \mu_y \leq 0$ gegen $H_1 : \mu_x - \mu_y > 0$ zusammen mit der Ausgabe eines 95%-Konfidenzintervalls für die Differenz der Mittelwerte durch

```
> t.test(x.sample, y.sample, alternative="greater", mu=0,
  paired=FALSE, var.equal=FALSE, conf.level = 0.95)
```

[1] Für Studenten gibt es eine kostenlose Version von S-Plus im Internet unter http://elms03.e-academy.com/splus/

mag auf den ersten Blick aufwändig erscheinen. Durch den expliziten Funktionsaufruf wird der Benutzer jedoch gezwungen, sich genau zu überlegen, welchen Test er verwenden will. [2]

- R (bzw. S) ist, wie Pascal oder C, eine funktionale Programmiersprache. Der Einsatz von R verlangt Grundkenntnisse im Programmieren. Eine Einführung in R ist also gleichzeitig eine informelle Einführung ins Programmieren.

[2] Der mit den Standardeinstellungen vertraute Benutzer erzielt das gleiche Ergebnis durch den Aufruf
`> t.test(x.sample, y.sample, alternative="greater")`

1 Der Start von R

Unter Windows gibt es zwei Versionen von R: eine Version, die nur aus einem Fenster zur Befehlseingabe besteht (Rterm.exe), und eine Version mit graphischer Benutzeroberfläche (Rgui.exe). Wir verwenden in diesem Kurs die zweite Version.

1.1 Grundlegende Befehle

Nach dem Start von R erscheint eine graphische Oberfläche mit einigen Menüs und der R-Console, in die die Befehle direkt eingegeben werden. Für weitere Arten der Befehlseingabe siehe Anhang B.2.

Gibt man 2+2 ein und drückt **Enter**, so erscheint

```
> 2+2
[1] 4
>
```

Die `[1]` besagt, dass das erste angeforderte Element folgt. Durch `>` wird angezeigt, dass die Berechnung zu Ende ist und ein weiteres Kommando eingegeben werden kann.

Hilfe bekommt man über das help-Menü; Hilfe zu einem bestimmten Thema, z.B. zur Sinus-Funktion, wird durch

```
> help(sin)   # alternativ: ?sin
```

angezeigt. Bei dieser Zeile wird die Hilfe nur einmal aufgerufen: alles, was nach dem Zeichen `#` folgt, wird von R als Kommentar gelesen, und nicht ausgewertet! Alternativ kann eine Html-Hilfe durch

```
> help.start()
```

gestartet werden.

Beendet wird R durch Anklicken des File-Menüs, dann auf Exit gehen, oder durch das quit-Kommando:

```
> q()
```

(beantwortet man dabei die Meldung “Save workspace image“mit “Yes“, so sichert dies alle Objekte, die während der Sitzung erzeugt worden sind).

1.2 Eine kurze R-Sitzung

Die meisten Datensätze haben die Form einer Matrix. Jede Zeile beinhaltet die Daten für eine Untersuchungseinheit, in den Spalten stehen die verschiedenen Merkmale. Die verschiedenen Spalten können aus unterschiedlichen Datentypen bestehen. Üblicherweise stehen die Namen der Merkmale in der ersten Zeile.

Der folgende Datensatz enthält Werte, die mit einem Gummiband ermittelt wurden: Dabei gibt das Merkmal Weite die Entfernung (in cm) an, um die sich das Gummiband nach Dehnung um eine bestimmte Strecke (in mm) bewegte, nachdem es losgelassen wurde.

Dehnung (mm)	Weite (cm)
46	148
54	182
48	173
50	166
44	109
42	141
52	166

Eine solche Datenstruktur wird in R durch einen sogenannten Data Frame repräsentiert. Man kann den Befehl `data.frame()` benutzen, um die Daten direkt als Data Frame einzugeben.

```
> gummi = data.frame(dehnung=c(46,54,48,50,44,42,52),
  weite=c(148,182,173,166,109,141,166))
```

Anschauen kann man einen Data Frame, in dem man seinen Namen eingibt:

```
> gummi
  dehnung weite
1      46   148
2      54   182
3      48   173
4      50   166
5      44   109
6      42   141
7      52   166
```

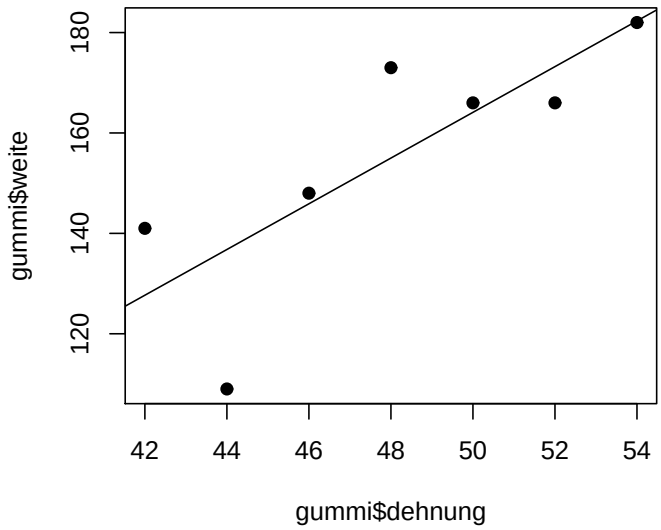


Abbildung 1.1: Scatterplot von Weite (in cm) gegen Dehnung (in mm)

Einen Scatterplot der Weite in Abhängigkeit von der Dehnung erhält man durch

```
> plot(gummi$dehnung, gummi$weite, pch=16)
```

Durch die Option `pch=16` werden ausgefüllte Kreise geplottet. Durch

```
> abline( lm(gummi$weite ~ gummi$dehnung) )
```

fügt man eine Regressionsgerade hinzu (jeweils Reihenfolge der beiden Merkmale beachten!). Abb. 1.1 zeigt das Ergebnis.

1.3 Weitere Bemerkungen

Eine Befehlseingabe kann über mehrere Zeilen gehen; wenn Sie z.B.

```
> help(     # Return druecken
  cos)
```

eingeben, erscheint auf dem Bildschirm

```
> help(     # Return druecken
+ cos)
```

Das +-Zeichen am Anfang der Fortsetzungszeile dürfen Sie nicht eingeben.

R unterscheidet zwischen Groß- und Kleinschreibung:

```
> Pi
Error: Object "Pi" not found
> pi
[1] 3.141593
```

Wichtig für alle Funktionen: Aufruf immer mit (evtl. leeren) Klammern, z.B. `q()` und nicht `q` (was passiert bei `q`?).

Zwei Hinweise:

- Namen von Objekten dürfen neben Buchstaben auch Zahlen, Punkt und Unterstrich (Underscore) enthalten. Anstatt `gummi` ist also auch `gummi1.df` ein zulässiger Name - die Endung deutet auf einen Data Frame hin.
- Mit `x = 3` wird `x` der Wert 3 zugewiesen. Für solche Zuweisungen verwenden wir immer das Zeichen `=` . Alternativ kann in R der Zuweisungspfeil `x <- 3` verwendet werden.

1.4 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 1 nach.
2. Die folgenden 18 Beobachtungen sind Schneebedeckungen von Eurasien im März von 1980 bis 1997 (Einheit: Millionen Quadratkilometer)[1]:

year	1980	1981	1982	1983	1984	1985	1986	1987	1988
snowcover	26.8	28.2	24.6	25.4	25.3	28.0	25.0	27.7	25.4
year	1989	1990	1991	1992	1993	1994	1995	1996	1997
snowcover	23.0	22.2	24.8	23.9	24.0	24.3	23.1	25.7	22.3

   a) Geben Sie die Daten in R als Data Frame mit Namen `snow` ein (anstatt `year=c(1980,1981,...,1997)` können Sie `year=1980:1997` schreiben!)

[1] Brown, R. 2002. Reconstructed North American, Eurasian, and Northern Hemisphere snow cover extent, 1915-1997. Boulder, CO: National Snow and Ice Data Center. Digital media.

b) Plotten Sie `snowcover` gegen `year`. Wiederholen Sie den Plot, wobei Sie die Option `type="l"` verwenden. Zeichnen Sie eine Regressionsgerade ein.

c) Fertigen Sie durch `stem(snow$snowcover)` eine Stammblattdarstellung an. Was passiert, wenn Sie die Option `scale=2` verwenden?

d) Zeichnen Sie ein Histogramm der `snowcover`-Werte mit Hilfe des `hist()`-Kommandos.

e) Wiederholen Sie b)-d), wobei Sie das neue Merkmal `log(snow$snowcover)`, also den Logarithmus von `snowcover` verwenden.

3. Laden Sie durch

```
> library(MASS); data(hills)
```

den in der Library MASS vorhandenen Datensatz `hills`. Dieser enthält die drei Merkmale `dist, climb` und `time`. Genauere Information erhalten Sie durch

```
> help(hills)
```

Plotten Sie `time` gegen `dist` und `time` gegen `climb` und zeichnen Sie jeweils eine Regressionsgerade ein.

2 Grundlegende R-Funktionen

2.1 Die Benutzung von R als Taschenrechner

In R sind eine Vielzahl von mathematischen Funktionen eingebaut. Auffällig ist, dass die meisten Funktionen Vektoren als Argumente zulassen.

```
> 2+2
[1] 4
> sqrt(10)   # Quadratwurzel
[1] 3.162278
> sqrt(c(10,100))
[1]  3.162278 10.000000
> 2*3*4*5
[1] 120
> 1000*(1+0.04)^5 - 1000  # Zinsen für 1000 Euro nach 5 Jahren bei 4% Zins
[1] 216.6529
> 2* pi*6378   # Erdumfang am Äquator, in km; Radius ist 6378 km
[1] 40074.16
> sin(c(30,60,90)*pi/180)  # Wandelt Grad- in Bogenmaß um
                           # und berechnet den Sinus
[1] 0.5000000 0.8660254 1.0000000
```

2.2 R-Objekte

Alle Elemente, die in R erzeugt wurde, auch Funktionen, existieren als Objekte im sogenannten Workspace. Durch `ls()` erhält man eine Liste aller Objekte im Workspace. Gibt man den Namen eines Objektes ein, so wird dessen Inhalt angezeigt. Durch `rm()` können Objekte gelöscht werden. Am Ende einer Sitzung kann der Benutzer alle innerhalb dieser Sitzung erzeugten Objekte löschen.

```
> a = 1.3
> ls()
[1] "a"   "b"   . . .
> rm(a)
> ls()
[1] "b"   . . .
```

2.3 Vektoren

Beispiele von Vektoren sind:

```
> c(2,3,5,2,7,1)
[1] 2 3 5 2 7 1
> 3:10      # die Zahlen 3,4,..,10
[1]  3  4  5  6  7  8  9 10
> c(T,F,F,F,T,T,F)     # ein logischer Vektor
[1] TRUE FALSE FALSE FALSE TRUE TRUE FALSE
> c("Berlin", "Frankfurt", "München", "Hamburg")
[1] "Berlin", "Frankfurt", "München", "Hamburg"
```

Wie im Beispiel gezeigt, wird durch `m:n` eine ganzzahlige Sequenz von m bis n erzeugt.

Vektoren können vom Datentyp numeric, logical oder character sein. Die ersten zwei Vektoren oben sind vom Typ numeric, der Dritte ist vom Typ logical (d.h. ein Vektor, bei dem alle Elemente vom Typ logical sind), und der vierte ist ein String-Vektor (d.h. ein Vektor, bei dem alle Elemente vom Typ character sind).

Das `c` in `c(2,3,5,7,1)` steht für "combine", also "zusammenfassen". Man kann auch zwei Vektoren zu einem neuen Vektor zusammenfassen:

```
> x  = c(2,3,5,2,7,1)
> x
[1] 2 3 5 2 7 1
> y = c(10,15,12)
> y
[1] 10 15 12
> z = c(x, y)
> z
[1] 2 3 5 2 7 1 10 15 12
```

Es gibt zwei gebräuchliche Arten, um auf Teilmengen von Vektoren zuzugreifen:

1. Man kann direkt die Stellen der Elemente angeben, auf die zugegriffen werden soll.

   ```
   > x = c(3,11,8,15,12)
   > x[4]   # auf viertes Element zugreifen
   [1] 15
   > x[c(2,4)]   # Auf Elemente 2 und 4 zugreifen
   [1] 11 15
   ```

2. Man kann auch einen Vektor von logischen Werten angeben. Dann werden die Elemente extrahiert, die den Wert `T` (`TRUE`) haben. Durch

   ```
   > y = x>10    # erzeugt Vektor vom Typ logical
   > y
   [1] FALSE  TRUE FALSE  TRUE  TRUE
   > x[y]        # oder direkt x[x>10]
   [1] 11 15 12
   ```

   erhält man diejenigen Werte von `x`, die größer als 10 sind.

   Dabei können die Relationen `<, <=, >, >=, ==` und `!=` verwendet werden. Die ersten vier vergleichen Größen, `==` testet auf Gleichheit, und `!=` testet auf Ungleichheit.

Operationen mit Vektoren sind immer elementweise zu verstehen:

```
> x = c(1,2,3); y = c(4,5,6)
> x+y
[1] 5 7 9
> x*y
[1]  4 10 18
```

Sind die beiden Vektoren nicht gleich lang, gibt es keine Fehlermeldung! Stattdessen wird der kürzere Vektor wiederholt verwendet:

```
> x-1            # entspricht x-c(1,1,1)
[1] 0 1 2
> c(x,y)-c(1,2)  # entspricht c(x,y)-c(1,2,1,2,1,2)
[1] 0 0 2 2 4 4
```

2.4 Data Frames

Data Frames sind neben Vektoren die grundlegende Datenstruktur. Ein Data Frame ist eine Verallgemeinerung einer Matrix, bei der die verschiedenen Spalten verschiedene Datentypen haben können. Alle Elemente einer Spalte müssen jedoch vom gleichen Typ sein, also alle vom Typ numeric, oder alle vom Typ factor, oder alle vom Typ character.

Ein Data Frame kann Zeilen- und Spaltennamen besitzen:

```
> library(MASS); data(hills)
> hills
                dist climb    time
Greenmantle      2.5   650  16.083
Carnethy         6.0  2500  48.350
Craig Dunain     6.0   900  33.650
Ben Rha          7.5   800  45.600
  ...            ...   ...     ...
```

Hier sind alle Spalten vom Typ numeric. Jeder der folgenden Befehle greift die dritte Spalte des hills-Datensatzes heraus und speichert sie in einem Vektor.

```
> Zeit = hills$time
> Zeit = hills[,3]
> Zeit = hills[,"time"]
```

Auf einzelne Elemente kann man durch Angabe von Zeilen- und Spaltenindex zugreifen:

```
> hills[2,3]        # Element in Zeile 2, Spalte 3
> hills[1:3,1:2]    # die Elemente in Zeile 1-3, Spalte 1-2
```

Übersichtlich anschauen und editieren kann man einen Data Frame (oder eine Matrix) in einem Spread-Sheet; dies erhält man unter R, indem man im Edit-Menü auf **Data Editor** klickt und den Namen des Data Frames eingibt.

Eine Liste aller Datensätze, die in den geladenen Libraries vorhanden sind, erhält man durch `data()`.

2.5 *Listen

Eine weitere, sehr flexible Datenstruktur sind Listen; diese können Objekte verschiedenster Art, z.B. Vektoren mit unterschiedlichen Datentypen, aber auch Data Frames, als Elemente enthalten. Listen werden durch `list()` erzeugt, der Zugriff auf die Elemente einer Liste erfolgt mittels des `[[]]`-Operators.

```
> liste = list(TRUE, "Liste", 5:8)
> liste[[1]]
[1] TRUE
> liste[[3]][1]
[1] 5
```

Die Elemente einer Liste können benannt werden; dann kann man auch mittels des Namens darauf zugreifen.

```
> liste = list(a=TRUE, b="Liste", c=5:8)
> liste$b
[1] "Liste"
> liste$c; liste$c[3]
[1] 5 6 7 8
[1] 7
```

2.6 Schleifen und Sequenzen

Im folgenden Beispiel einer for-Schleife wird die Temperatur in Grad Celsius nach Grad Fahrenheit umgewandelt.

```
> for (celsius in 25:30) {print(c(celsius, 9/5*celsius + 32))}
[1] 25 77
[1] 26.0 78.8
[1] 27.0 80.6
[1] 28.0 82.4
[1] 29.0 84.2
[1] 30 86
```

Da die allermeisten Funktionen in R Vektoren als Argumente zulassen, kann man Schleifen in R oft vermeiden.

```
> celsius = 25:30
> print(9/5*celsius + 32)
[1] 77.0 78.8 80.6 82.4 84.2 86.0
```

Weitere Befehle zur Konstruktion von Schleifen sind `while, repeat, break`.

Eine ganzzahlige Sequenz von m bis n wird durch `m:n` erzeugt. Um allgemeinere Sequenzen mit Start- bzw. Endwert `a` bzw. `b` und Schrittweite `h` zu erstellen, steht der Befehl `seq(a,b,h)` zur Verfügung.

Mit `rep(a,n)` kann das erste Element `n` mal wiederholt werden; `a` und `n` können auch Vektoren sein.

```
> seq(2,10,2)   # gerade Zahlen
[1]  2  4  6  8 10
> rep("m", 5)
[1] "m" "m" "m" "m" "m"
> rep( c("m","w"), c(3,2))
[1] "m" "m" "m" "w" "w"
```

2.7 Wichtige Funktionen

Tabelle 2.1 gibt einen Überblick über wichtige Funktionen.

Ein Objekt wird durch `print()` oder durch Eingabe des Objektnamens ausgegeben. Eine Klammer um eine Zuweisung führt ebenfalls zur Ausgabe des Objekts. Die folgenden drei Befehlszeilen führen also zum gleichen Ergebnis:

```
> x = 1:5; x   # Strichpunkt trennt Befehle in gleicher Zeile
> x = 1:5; print(x)
> (x = 1:5)
```

Innerhalb einer `for`-Schleife führt allerdings nur die Verwendung von `print(x)` zur Ausgabe von `x`!

Funktion	Beschreibung
`print()`	ein einzelnes R Objekt ausgeben
`summary()`	Zusammenfassung eines Objektes ausgeben
	(Details hängen vom Objekt ab)
`length()`	Anzahl der Elemente eines Vektors
	Anzahl der Merkmale eines Data Frames
`mean()`	Mittelwert
`var()`	Varianz
`sd()`	Standardabweichung
`median()`	Median
`quantile()`	Stichproben-Quantil
`IQR()`	Quartilsabstand
`min()`	Minimum
`max()`	Maximum
`sum()`	Summe
`prod()`	Produkt
`sort()`	ordnet die Elemente der Größe nach
`rank()`	bestimmt die Ränge der Elemente

Tabelle 2.1: Tabelle wichtiger Funktionen

Ist `x` ein Datenvektor, so gibt `quantile(x)` Minimum, unteres Quartil, Median, oberes Quartil und Maximum von `x` aus. Für die Berechnung eines beliebigen Quantils von x existiert der optionale Parameter `probs`: mit `quantile(x, probs = 0.9)` erhält man das empirische 90%-Quantil von `x`.

Beispiel: Berechnung der Summe der Zahlen von 1 bis 100 mit einer for-Schleife

```
> s=0; for (i in 1:100) {s=s+i}
> s
[1] 5050
```

oder mit der Funktion `sum()`:

```
> sum(1:100)
[1] 5050
```

Die Varianz der Stichprobe 3,1,6,4 kann man statt mit `var()` auch durch

```
> x = c(3,1,6,4)
> sum( (x-mean(x))^2 )/(length(x)-1)
[1] 4.333333
```

berechnen. Warum?

Die Funktion `sapply()` wendet eine Funktion auf alle Spalten eines Data Frames an:

```
> sapply(hills,mean)          # mean(hills) geht auch
       dist       climb        time
   7.528571 1815.314286   57.875714
> sapply(hills,IQR)           # IQR(hills) funktioniert nicht!
    dist    climb     time
   3.500 1475.000   40.625
```

2.8 *Definieren von Funktionen

Es ist einfach, Funktionen in R selbst zu definieren. Hier folgt ein Beispiel einer Funktion, die die Temperatur in Grad Celsius nach Grad Fahrenheit umgewandelt.

```
> c.nach.f = function(celsius) {9/5*celsius + 32}
> c.nach.f(25:30)     # Funktionsaufruf
[1] 77.0 78.8 80.6 82.4 84.2 86.0
```

Von einer Funktion wird immer der Wert des letzten (und in diesem Beispiel einzigen) Ausdrucks im Funktionskörper zurückgegeben. Beachten Sie, dass der Funktionskörper durch `{ }` eingeschlossen wird.

Alternativ ist es möglich, mit `return()` einen Wert explizit zurückzugeben, wie es das zweite Beispiel zeigt.

Argumente können mit Standardwerten belegt werden. Die folgende Funktion, die Mittelwert und empirische Standardabweichung einer Stichprobe ausgibt, wird bei Aufruf ohne Argument auf den Vektor `1:10` angewendet.

```
> mw.und.s = function(x=1:10)
  { mw = mean(x)
    s = sd(x)
    return(c(Mittelwert=mw, Standardabweichung=s))
  }
> mw.und.s()
        Mittelwert Standardabweichung
          5.500000           3.027650
```

Weitere Informationen zur Definition und zum Aufruf von R-Funktionen finden Sie in Anhang B.3.

2.9 Zufallsvariablen und Verteilungen

2.9.1 Erzeugung von Zufallszahlen

Durch `runif(N,a,b)` werden N im Intervall $[a, b]$ gleichverteilte Zufallszahlen erzeugt.

```
> runif(10,0,1)
  [1]  0.44187450  0.28597587  0.08808785  0.15907081  0.25074339
  [6]  0.41663454  0.38717038  0.19944928  0.59790573  0.89205275
```

Eine entsprechende Funktion gibt es für viele Verteilungen; zum Beispiel werden durch `rnorm(N,mu,sigma)` (stochastisch unabhängige) Realisierungen von normalverteilten Zufallsvariablen mit Erwartungswert μ und Standardabweichung σ (also der Varianz σ^2 !!) erzeugt. Der folgende Aufruf generiert somit fünf $N(2, 9)$-verteilte Zufallszahlen.

```
> rnorm(5,2,3)
[1] 4.813535 4.603353 6.396016 4.196891 5.995908
```

`rbinom(N,n,p)` erzeugt N binomialverteilte Zufallszahlen mit Parametern n und p :

```
> rbinom(5,3,0.5)
[1] 1 3 3 2 1
```

Bemerkung: Da sich die im Computer erzeugten Folgen von Zufallszahlen zwar in vieler Hinsicht wie zufällige Zahlenfolgen verhalten, sie aber rein deterministisch erzeugt werden, nennt man die Zahlen *Pseudozufallszahlen*.

2.9.2 Verteilungsfunktionen

Die Verteilungsfunktion $F(t) = P(X \leq t)$ einer Zufallsvariablen X an der Stelle t berechnet man, indem man bei den Funktionen für die Zufallszahlenerzeugung das `r` (für *random*) durch ein `p` (für *probability*) und die Anzahl N von Zufallszahlen durch t ersetzt. `pnorm(t,0,1)` berechnet also die Verteilungsfunktion einer Standardnormalverteilung an der Stelle t.

```
> pnorm(0,0,1)
[1] 0.5
```

Analog erhält man die Dichte bzw. das Quantil an der Stelle t, indem man das `r` durch ein `d` (für *density*) bzw. durch ein `q` (für *quantile*) ersetzt:

```
> dnorm(0,0,1)
[1] 0.3989423    # 1/sqrt(2*pi)
> qnorm(0.5,0,1)
[1] 0
```

Bei diskreten Zufallsvariablen X ergibt das Voranstellen von `d` die Wahrscheinlichkeit $P(X = k)$:

```
> dbinom(0:3,3,0.5)
[1] 0.125 0.375 0.375 0.125
```

Im folgenden Beispiel werden die $k\sigma$-Bereiche einer Normalverteilung mit $\mu = 3$ und $\sigma = 2$ für $k = 1, 2, 3$ berechnet. Ist X eine $N(3, 2)$-verteilte Zufallsvariable, so sind die $k\sigma$-Bereiche durch

$$P(\mu - k \cdot \sigma \leq X \leq \mu + k \cdot \sigma)$$

definiert.

```
> mu=3; sigma=2
> for (k in 1:3) {
    print( pnorm(mu+k*sigma,mu,sigma) - pnorm(mu-k*sigma,mu,sigma) )
  }
[1] 0.6826895
[1] 0.9544997
[1] 0.9973002
```

2.9.3 Übersicht über die Verteilungen in R

Verteilung	Zufallszahlen	Verteilungsfunktion
$Bin(n,p)$	`rbinom(N, n, p)`	`pbinom(t, n, p)`
$Hyp(n,r,s)$	`rhyper(N, r, s, n)`	`phyper(t, r, s, n)`
$Po(\lambda)$	`rpois(N, λ)`	`ppois(t, λ)`
$G(p)$	`rgeom(N, p)`	`pgeom(t, p)`
$U(a,b)$	`runif(N, a, b)`	`punif(t, a, b)`
$Exp(\lambda)$	`rexp(N, λ)`	`pexp(t, λ)`
$N(\mu,\sigma^2)$	`rnorm(N, μ, σ)`	`pnorm(t, μ, σ)`
$LN(\mu,\sigma^2)$	`rlnorm(N, μ, σ)`	`plnorm(t, μ, σ)`

In den Standard-Paketen von R sind außerdem vorhanden:
Beta-Verteilung (`beta`), Cauchy-Verteilung (`chauchy`), Chi-Quadrat-Verteilung (`chisq`), F-Verteilung (`f`), Gamma-Verteilung (`gamma`), logistische Verteilung (`logis`), negative Binomialverteilung (`nbinom`), Student'sche t-Verteilung (`t`), Tukey-Verteilung (`tukey`), Weibull-Verteilung (`weibull`), Wilcoxon-Verteilung (`wilcox`).

Viele weitere Verteilungen findet man in anderen Paketen, insbesondere im Paket `SuppDists`.

2.10 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 2 nach.
2. Definieren Sie `x=1:5` und `y=1:6` und wenden Sie alle Funktionen aus Abschnitt 2.7 auf `x` bzw. `y` an. Stimmen alle Ergebnisse mit den Definitionen der Vorlesung überein?
3. Multipizieren Sie die Zahlen von 1 bis 10 auf zwei verschiedene Arten: indem Sie eine `for`-Schleife und indem Sie die Funktion `prod()` verwenden.
4. Verwenden Sie `sapply()`, um Mittelwert und Standardabweichung der Merkmale `dehnung` und `weite` (im Datensatz `gummi` aus Kap. 1) zu berechnen. Wenden Sie auch die Funktion `summary()` auf diesen Datensatz an.
5. Nach Angaben der europäischen Zentralbank haben 1-Euro-Münzen ein Durchschnittsgewicht von 7.5 Gramm. Eine Messung bei $n = 10$ Münzen ergab die Gewichte

$$7.527, 7.486, 7.536, 7.442, 7.558, 7.531, 7.526, 7.560, 7.478, 7.545$$

a) Ein guter Schätzer für die Stichprobenvarianz bei bekanntem Mittelwert μ ist $\hat{\sigma}^2 = 1/n \sum_{i=1}^{n}(x_i - \mu)^2$. Berechnen Sie $\hat{\sigma}^2$ und $\hat{\sigma}$.

b) Bestimmen Sie mit Hilfe von R die Anzahl der Münzen, die schwerer als 7.5g sind.

6. Laden Sie den R-Datensatz `ChickWeight`. Dieser enthält die Merkmale `weight`, `Time`, `Chick` und `Diet`. Informieren Sie sich genauer über den Datensatz. Berechnen Sie die Durchschnittsgewichte der 10 Tage alten Küken bei den vier Futtersorten.

7. a) Sortieren Sie den Vektor (5,2,13,1,8,3,21) der Größe nach, und speichern Sie das zweit- und das drittgrößte Element ab.

   b) Erzeugen Sie 20 standardnormalverteilte Zufallszahlen, und speichern Sie sie im Vektor `x`. Berechnen Sie das 0.2-getrimmte Mittel von `x`. Die größte ganze Zahl kleiner oder gleich `a` kann dabei durch `floor(a)` berechnet werden.

8. a) Erzeugen Sie $n = 100$ normalverteilte Zufallszahlen mit Parametern $\mu = 3$ und $\sigma = 2$ und berechnen Sie den arithmetischen Mittelwert und die Standardabweichung.

   b) Wiederholen Sie Teil a) 1000 Mal, wobei Sie jeweils den arithmetischen Mittelwert abspeichern. Berechnen Sie die Standardabweichung der 1000 Mittelwerte.

9. a) Die Zufallsvariable X sei $U(1,2)$-verteilt. Berechnen Sie $P(X \leq 1.4)$, die Dichte an der Stelle 1.4, sowie das untere und obere Quartil (d.h. das 25%- und 75%-Quantil).

   b) Erzeugen Sie $n = 20$ auf dem Intervall $[1,2]$ gleichverteilte Zufallszahlen und berechnen Sie das (empirische) untere und obere Quartil. Wiederholen Sie dies für $n = 100$ und $n = 1000$.

10. Berechnen Sie die $k\sigma$-Bereiche einer Normalverteilung für $k = 1, \ldots, 5$ ohne Verwendung einer for-Schleife.

11.* a) Schreiben Sie eine Funktion *qa* zur Berechnung des Quartilsabstands.

   b) Schauen Sie sich den Quellcode und die Hilfe der Funktion IQR an.

12.* a) Schreiben und testen Sie eine Funktion, die die *Medianabweichung* (gleich dem Median von $|x_1 - \tilde{x}|, |x_2 - \tilde{x}|, \ldots, |x_n - \tilde{x}|$) einer Stichprobe berechnet.

   b) Erzeugen Sie $n = 10$ normalverteilte Pseudozufallszahlen und bestimmen Sie Medianabweichung und Quartilsabstand. Wiederholen Sie dies für $n = 100, 1000, 10000$. Welche Vermutung drängt sich auf? Können Sie diese Vermutung durch weitere Überlegungen stützen oder widerlegen?

13.* a) Schreiben Sie eine Funktion, die die Standardabweichung einer normalverteilten Stichprobe sowohl mit der empirischen Standardabweichung als auch mit Hilfe des Quartilsabstands schätzt.

Hinweis: $\Phi^{-1}_{\mu,\sigma^2}(p) = \mu + \sigma\,\Phi^{-1}_{0,1}(p)$

Testen Sie diese Funktion an mit `rnorm` erzeugten Pseudozufallszahlen.

b) Erzeugen Sie 100 normalverteilte Pseudozufallszahlen und schätzen Sie die Standardabweichung einmal mit Hilfe der empirischen Standardabweichung als auch mit Hilfe des Quartilsabstands. Sie erhalten Schätzwerte $\hat{\vartheta}_1(x_1, \ldots, x_{100})$ und $\hat{\vartheta}_2(x_1, \ldots, x_{100})$.

Führen sie dieses Experiment 10000 mal durch und berechnen Sie jeweils die Stichprobenvarianz dieser 10000 Schätzwerte. Interpretieren Sie das Ergebnis.

3 Graphik mit R

`plot()`, `curve()`, `points()`, `lines()`, `text()`, `mtext()`, `axis()`, `identify()` sind grundlegende Funktionen, um Punkte, Linien und Graphen zu zeichnen und mit Achsen und Beschriftungen zu versehen. Einen Eindruck über die Graphikfähigkeiten von R gewinnt man, indem man `demo(graphics)` eingibt und Enter drückt. Nachdem die erste Graphik gezeichnet wurde, klick man auf die Graphik-Seite, so dass das **History**-Menü erscheint. In diesem Menü wählt man **Recording**; jetzt kann man in den Graphik-Seiten vorwärts und rückwärts blättern. Danach kehrt man zum Kommando-Fenster zurück, und erhält durch erneutes Drücken der Enter-Taste weitere Graphiken.

3.1 plot() und verwandte Funktionen

Mit den folgenden Befehle erhält man einen Scatterplot von `y` gegen `x`:

```
> x = runif(20); y = runif(20)
> plot(x, y)
> plot(y ~ x)   # dasselbe in Formelschreibweise
```

Dabei müssen die Vektoren `x` und `y` natürlich gleich lang sein.

```
> library(MASS); data(hills)
> plot(time ~ climb)   # R kennt die Merkmale time und climb nicht
Error in eval(expr, envir, enclos) : Object "climb" not found
> plot(time ~ climb, data=hills)   # so sucht R im Data Frame hills
> plot(hills$time ~ hills$climb)   # funktioniert auch
```

Mit dem **attach**-Befehl findet R die Merkmale **time** und **climb** auch ohne explizite Angabe:

```
> attach(hills)
> plot(time ~ climb)
> detach(hills)
```

Eine mathematische Funktion kann man durch

```
> x = (0:40)*pi/10   # Sequenz von 0 bis 4*pi
> plot(x, sin(x), type="l")   # die Option type="l" verbindet die Punkte
> lines(x, cos(x))   # fuegt zweiten Plot hinzu
```

zeichnen. Die Funktionen `points()` bzw. `lines()` fügen Punkte bzw. Linien zu einem bereits vorhandenen Plot hinzu. Einfacher geht es durch

```
> curve(sin(x), from=0, to=4*pi)
> curve(cos(x), add=T)   # fuegt zweiten Plot hinzu
```

Hier muss als Variable `x` verwendet werden; der Aufruf `curve(sin(t),from=0,to=4*pi)` führt zu einer Fehlermeldung[1].

Ohne die Option `add=T` wird ein neuer Plot erstellt (mit den im letzten Plot gewählten `from`- und `to`-Werten).

Die plot-Funktion ist eine generische Funktion, die spezielle Methoden für viele verschiedene Objekte hat. Wendet man die plot-Funktion zum Beispiel auf einen Data Frame an, so erhält man eine Scatterplotmatrix; dabei wird jedes numerische Merkmal gegen jedes andere geplottet:

```
> plot(hills)   # derselbe Effekt wie pairs(hills)
```

3.2 Verändern von Graphik-Parametern

Die Standardeinstellungen von Graphik-Parametern wie dem Linientyp können durch Aufruf von `par()` verändert werden. Kennengelernt haben wir schon den Parameter `pch`. Man wählt das Plot-Symbols durch

```
> par(pch=16)
> plot(climb ~ time, data=hills)
```

oder einfach durch

```
> plot(climb ~ time, data=hills, pch=16)
```

Farbe (`col` für color) und Linientyp (`lty` für line type) wählt man folgendermaßen:

[1] In diesem Beispiel ist dagegen der Aufruf `curve(sin,from=0,to=4*pi)` zulässig

```
> curve(sin, from=0, to=4*pi, col="blue")   # Sinus in blau
> curve(cos, add=T, col="red", lty=2)   # Cosinus rot gestrichelt
```

Das Ergebnis zeigt Abb. 3.1.

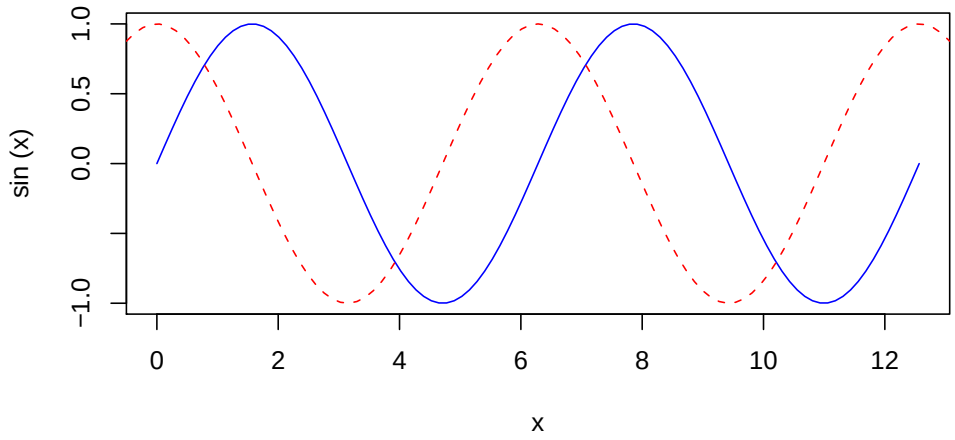


Abbildung 3.1: Sinus- und Cosinus im Intervall $[0, 4\pi]$

Der Parameter `mfrow` kann verwendet werden, um mehrere Abbildungen auf einer Seite zu plotten. Der Datensatz `Animals` in der library `MASS` enthält Körper- und Gehirngewichte von verschiedenen Tieren. Durch die folgenden Befehle werden diese Werte unterschiedlichen Transformationen unterzogen und geplottet.

```
> par(mfrow=c(2,2), pch=16)   # Anordnung der Plots in 2x2-Matrix
> library(MASS); data(Animals)
> attach(Animals)
> plot(body, brain)
> plot(sqrt(body), sqrt(brain))
> plot((body)^0.1, (brain)^0.1)
> plot(log(body),log(brain))
> detach(Animals)
> par(mfrow=c(1,1), pch=1)   # Wieder eine Abbildung pro Seite
```

Durch `help(par)` bekommt man eine Übersicht über alle verfügbaren Parameter.

3.3 Hinzufügen von Texten

Das folgende Beispiel zeigt, wie mit der Funktion `text()` Label an den Datenpunkten angebracht werde.

```
> Primaten = Animals[c("Potar monkey","Gorilla","Human",
  "Rhesus monkey","Chimpanzee"),]
> Primaten   # die Primaten im Data Frame Animals
                body brain
Potar monkey   10.00   115
Gorilla       207.00   406
Human          62.00  1320
Rhesus monkey   6.80   179
Chimpanzee     52.16   440
>
> attach(Primaten)
> plot(body, brain, xlim=c(5, 250))
> text(x=body, y=brain, labels=row.names(Primaten), adj=0)
  # adj=0: Text linksbuendig
> detach(Primaten)
```

Im `text()`-Befehl werden mit x und y die Koordinaten spezifiziert, bei denen der Text beginnt, falls `adj=0` gesetzt ist. Mit dem `row.names()`-Befehl werden die Zeilennamen eines Dataframes als Beschriftungen gewählt; `row.names(Primaten)` entspricht also dem character-Vektor `c("Potar monkey","Gorilla",...,"Chimpanzee")`. Abb. 3.2 zeigt das Resultat.

In Abb. 3.3 werden die Achsen mittels der Parameter `xlab` und `ylab` sinnvoller beschriftet. Die Label werden nach rechts geschoben; außerdem werden die Label `Potar monkey` und `Rhesus monkey` auseinandergezogen.

Hier folgt der R-Code für Abb. 3.3:

```
> attach(Primaten)
> plot(body, brain, pch=16, xlab="Körpergewicht (kg)",
  ylab="Gehirngewicht (g)", xlim=c(5,250), ylim=c(0,1500))
> text(x=body+10, y=brain+c(-30,0,0,30,0),
  labels=row.names(Primaten), adj=0)
> detach(Primaten)
```

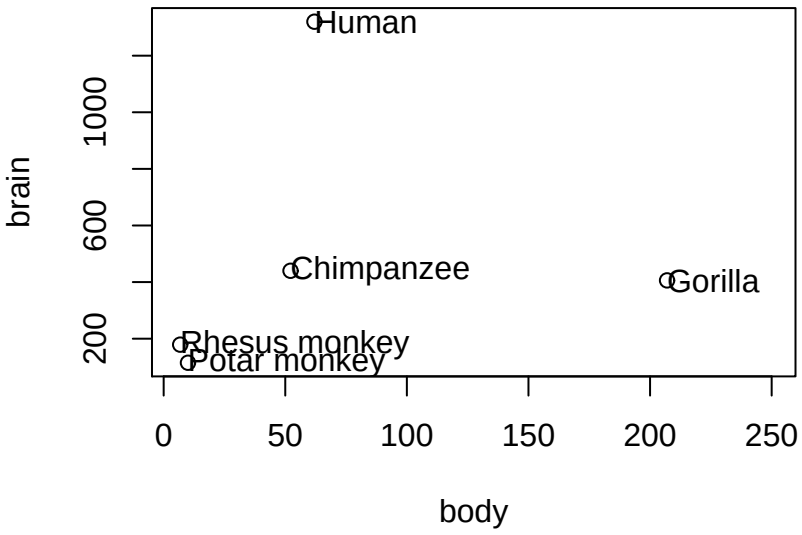


Abbildung 3.2: Plot des Datensatzes **Primaten**

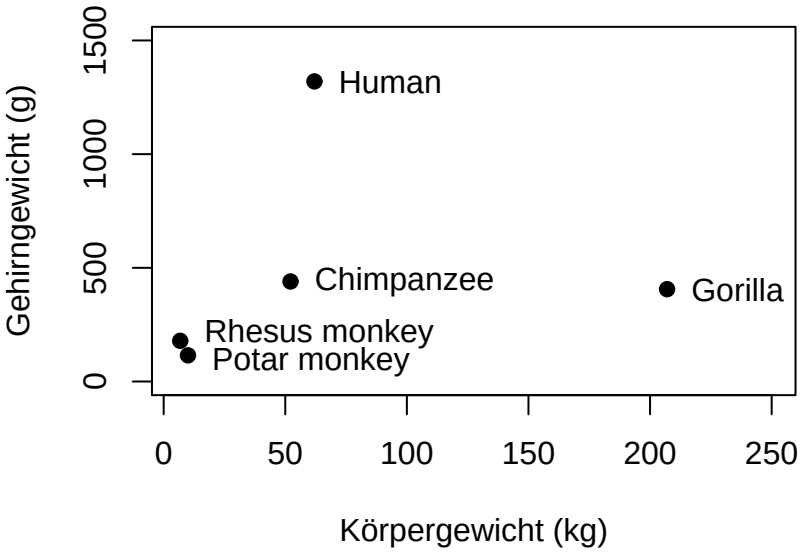


Abbildung 3.3: Verbesserte Version von Abb. 3.2

3.4 Visualisierung der Verteilung von Daten

3.4.1 Histogramme

Das Aussehen von Histogrammen hängt oft stark von der Klassenanzahl und bei wenigen Klassen manchmal auch von der Plazierung der Klassengrenzen ab. Abb. 3.4 zeigt die Wartezeiten des Old Faithful Geysirs zwischen zwei Eruptionen, links mit fünf Klassen, und rechts mit 12 Klassen.

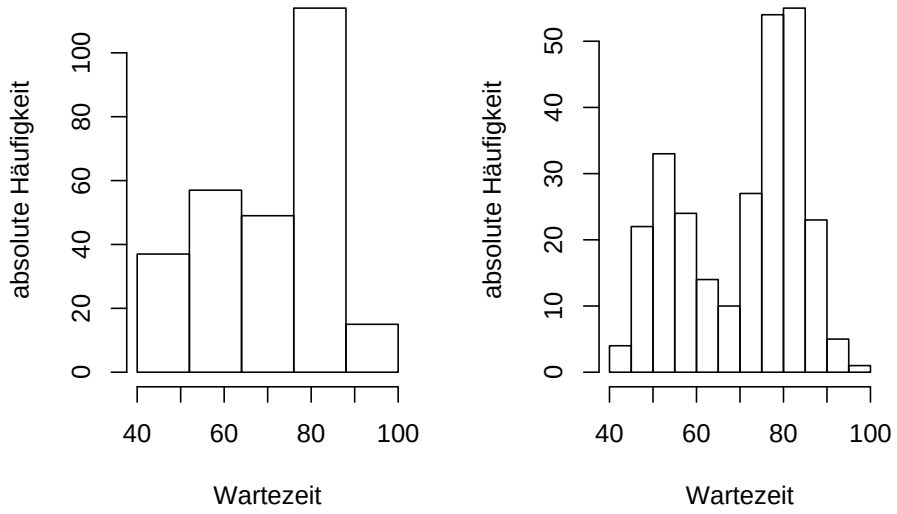


Abbildung 3.4: Wartezeiten des Old Faithful Geysirs

Die Histogramme wurden durch folgende Befehle erzeugt:

```
> data(faithful)
> summary(faithful)
   eruptions        waiting
 Min.   :1.600   Min.   :43.0
 1st Qu.:2.163   1st Qu.:58.0
 Median :4.000   Median :76.0
 Mean   :3.488   Mean   :70.9
 3rd Qu.:4.454   3rd Qu.:82.0
 Max.   :5.100   Max.   :96.0
>
> par(mfrow=c(1,2))
> hist(faithful$waiting, breaks=c(40,52,64,76,88,100), main="",
  xlab="Wartezeit", ylab="absolute Häufigkeit")
> hist(faithful$waiting, main="",
```

```
  xlab="Wartezeit", ylab="absolute Häufigkeit")
```

Das in R verwendete Verfahren zur Wahl der Klassenzahl führt meist zu guten Ergebnissen, wie im rechten Histogramm in Abb. 3.4.

3.4.2 Dichte-Plots

Im Gegensatz zu Histogrammen hängen Dichteplots nicht von der Wahl der Klassenzahl bzw. Klassengrenzen ab. Allerdings hängt das Aussehen der Dichteplots von den Parametern des zugrundeliegenden Kerndichteschätzers ab. Abb. 3.5 zeigt links einen Dichteplot der Dauer der Eruptionen des Old Faithful Geysirs; rechts sind Histogramm und Dichteplot überlagert.

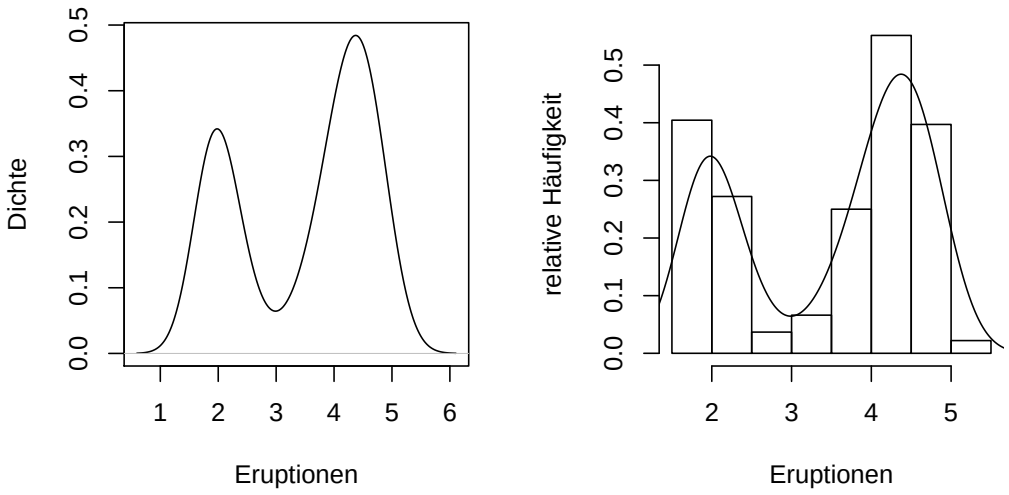


Abbildung 3.5: Dauer der Eruptionen des Old Faithful Geysirs

Der folgende R-Code erzeugt Abb. 3.5:

```
> par(mfrow=c(1,2))
> d = density(faithful$eruptions)   # Kerndichteschaetzung
> plot(d, main="", xlab="Eruptionen", ylab="Dichte")
> hist(faithful$eruptions, main="", xlab="Eruptionen",
  ylab="relative Häufigkeit", freq=F)
> lines(d)
```

Das Resultat der Dichteschätzung wird hier im Objekt `d` abgespeichert und mit `plot(d)` bzw. `lines(d)` in ein neues bzw. bereits vorhandenes Bild eingezeichnet. Die Option

`freq=F` beim Aufruf von `hist` sorgt dafür, dass im Histogramm anstelle der absoluten die relativen Häufigkeiten abgetragen werden.

Als weiteres Beispiel zeigt Abb. 3.6 ein Histogramm von 100 $N(1,4)$-verteilten Zufallszahlen, zusammen mit einem Dichteplot und der theoretischen Dichte der $N(1,4)$-Verteilung.

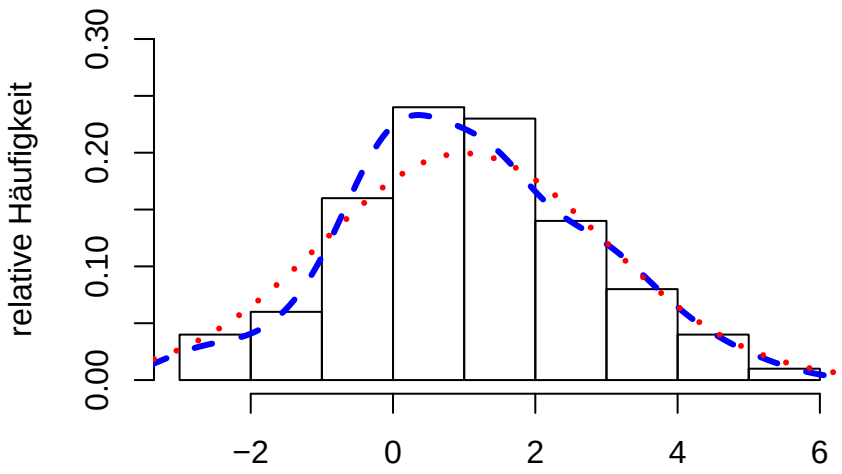


Abbildung 3.6: Histogramm von 100 $N(1,4)$-verteilten Zufallszahlen

Die Abbildung wurde durch die folgenden Befehle erzeugt.

```
> par(mfrow=c(1,1))
> sample = rnorm(100,1,2)
> hist(sample, freq=F, ylim=c(0,0.3), main="", xlab="",
  ylab="relative Häufigkeit")
> lines(density(sample), col="blue", lty=2, lwd=3)
> curve(dnorm(x,1,2), add=T, col="red", lty=3, lwd=3)
```

3.4.3 Boxplots

Abb. 3.7 zeigt, welche Informationen man einem Boxplot entnehmen kann. Dabei ist der Endpunkt des nach oben aufgesetzen Stabes die größte Beobachtung, die kleiner als das obere Quartil plus das 1.5-fache des Quartilsabstands ist. Analog ist der Endpunkt des nach unten angebrachten Stabes die kleinste Beobachtung, die größer als das untere Quartil minus das 1.5-fache des Quartilsabstands ist.

Wie der durch

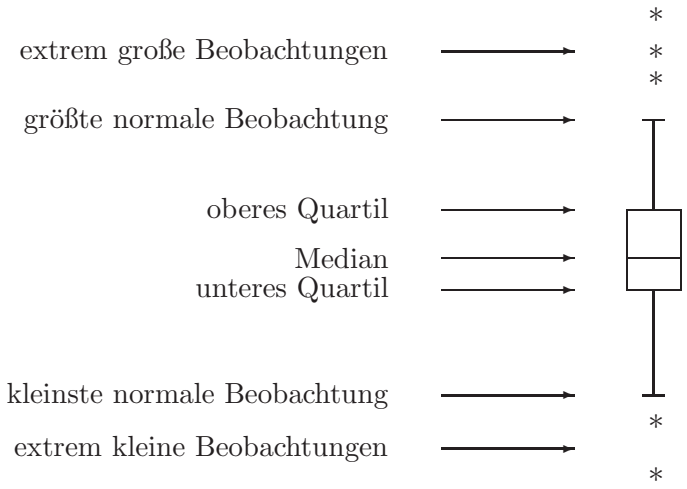


Abbildung 3.7: Boxplot

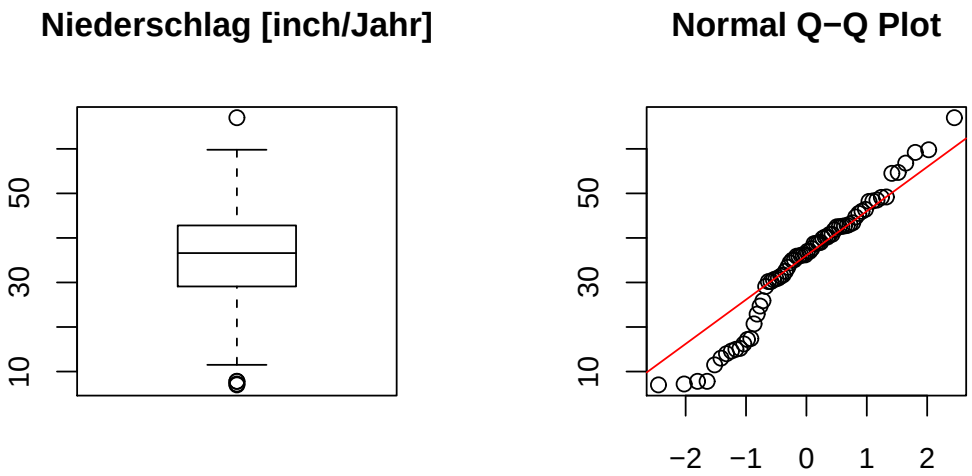


Abbildung 3.8: Boxplot und QQ-Plot der Niederschläge in 70 US-Städten

```
> data(precip)
> boxplot(precip, main="Niederschlag [inch/Jahr]")
```

erzeugte Boxplot in Abb. 3.8 zeigt, ist die Verteilung der Niederschläge in 70 US-Städten recht symmetrisch.

Im folgenden Beispiel werden 50 Realisierungen einer Normal-, einer Gleich- und einer Exponential-Verteilung mit gleichem Erwartungswert und gleicher Varianz erzeugt und jeweils ein Boxplot gezeichnet (siehe Abb. 3.9).

```
> x = rnorm(50,1,1)   # E(X)=V(X)=1
> y = runif(50,1-sqrt(3),1+sqrt(3))   # E(X)=V(X)=1
> z = rexp(50,1)   # E(X)=V(X)=1
> par(mfrow=c(1,3))
> boxplot(x, main="Normal")
> boxplot(y, main="Gleich")
> boxplot(z, main="Exponential")
```

An den Boxplots kann man gut erkennen, dass die Exponentialverteilung im Gegensatz zur Normal- und Gleichverteilung keine symmetrische Verteilung ist.

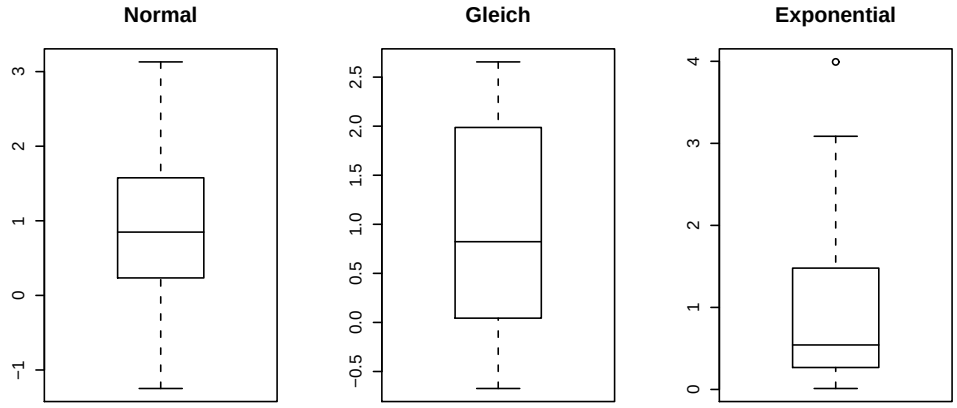


Abbildung 3.9: Boxplot von 50 Zufallszahlen aus einer Normal-, einer Gleich- und einer Exponential-Verteilung, jeweils mit $E(X) = V(X) = 1$

3.4.4 Quantil-Quantil-Plots

`qqnorm(x)` plottet einen Quantil-Quantil-Plot (QQ-Plot) auf Normalverteilung der Elemente von `x`. Die geplotteten Punkte liegen approximativ auf einer Geraden, wenn die Daten einer Normalverteilung entstammen.

Einen QQ-Plot der Niederschlagsdaten aus Abschnitt 3.4.3 zeigt Abb. 3.8. Dabei wurde mit dem Befehl `qqline()` eine Gerade hinzugefügt, die durch das untere und obere Quartil geht:

```
> data(precip)
```

```
> qqnorm(precip, xlab="", ylab="")
> qqline(precip, col="red")
```

Man erkennt gewisse Abweichungen von der Geraden, es ist aber schwierig zu entscheiden, ob sie gegen eine Normalverteilung sprechen.

Als formaler Anpassungstest auf Normalverteilung ist der Shapiro-Wilk-Test in R implementiert. Der Aufruf dieses Tests durch

```
> shapiro.test(precip)
Shapiro-Wilk normality test
data:  precip
W = 0.9646, p-value = 0.04493
```

liefert einen p-Wert von 0.045, die Niederschlagsdaten weichen also etwas von einer Normalverteilung ab.

Um an einem QQ-Plot Abweichungen von einer Normalverteilung zu erkennen, ist es hilfreich, QQ-Plots von simulierten Datensätzen vom gleichen Umfang anzuschauen.

Abb. 3.10 zeigt links oben nochmals den QQ-Plot der Niederschlagsdaten. Die restlichen Plots zeigen QQ-Plots von Stichproben vom Umfang 70 aus einer Normalverteilung, erzeugt mit dem Befehl `rnorm(70,0,1)`:

```
> data(precip)
> par(mfrow=c(2,4))
> qqnorm(precip, main="Niederschlag", xlab="", ylab="")
> qqline(precip)
> for(i in 1:7)
    {x = rnorm(70,0,1)
     qqnorm(x, main="Simuliert", xlab="", ylab="")
     qqline(x)
    }
```

Beim Betrachten von Abb. 3.10 gewinnt man den Eindruck, dass die Abweichung des QQ-Plots der Niederschlagsdaten von einer Geraden im unteren Bereich stärker ist, als dies gewöhnlich der Fall ist, wenn den Daten tatsächlich eine Normalverteilung zugrundeliegt.

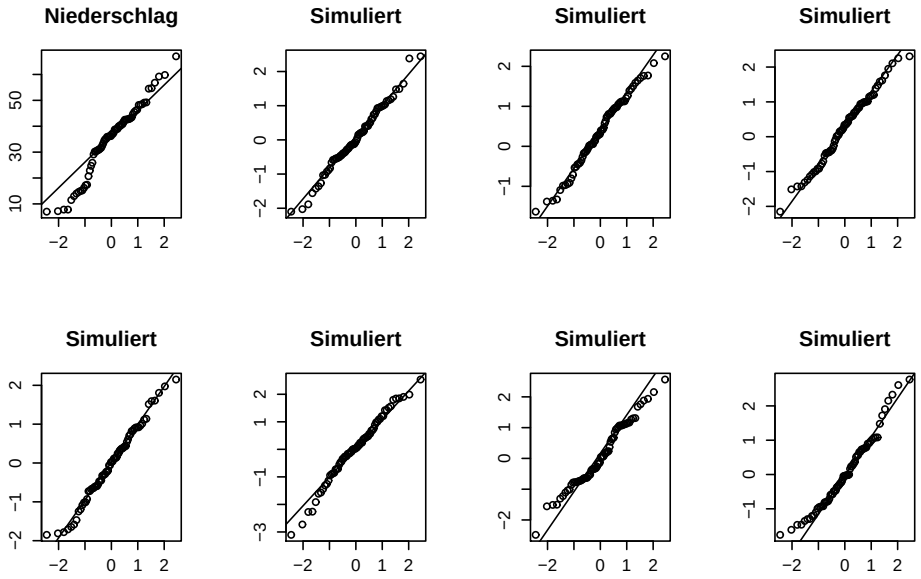


Abbildung 3.10: QQ-Plot der Niederschlagsdaten, zusammen mit QQ-Plots von Stichproben vom gleichen Umfang aus einer Normalverteilung

3.4.5 Rugplots

Boxplots eignen sich gut, um unsymmetrische Verteilungen oder Ausreißer zu erkennen. Manche Verteilungscharakteristika können sie dagegen naturgemäß nicht erfassen: Abb. 3.11 zeigt einen Boxplot der Dauer der Eruptionen des Old Faithful Geysirs. Am Boxplot kann man nicht erkennen, dass die zugrunde liegende Verteilung bimodal ist. Der Befehl `rug(x)` fügt an einer Seite des aktuellen Plots Striche hinzu, die die Verteilung der Werte von x zeigen. Abb. 3.11 wurde mit folgenden Befehlen erzeugt:

```
> data(faithful)
> par(mfrow=c(1,1))
> boxplot(faithful$eruptions, boxwex=0.3)
    # der Parameter boxwex kontrolliert die Breite der Box
> rug(faithful$eruptions, side = 2)   # Rugplot an linker Seite
```

3.5 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 3 nach.

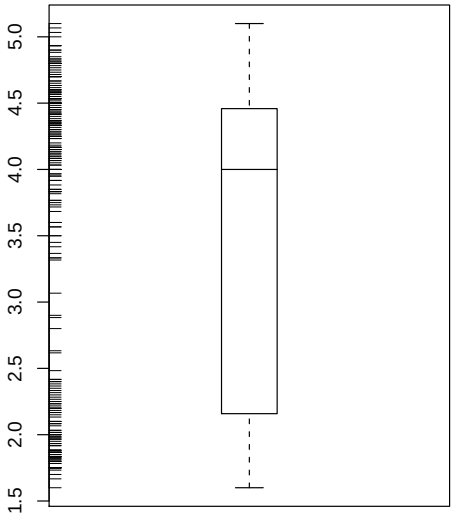


Abbildung 3.11: Verteilung der Eruptionsdauern des Old Faithful Geysirs. Der Rugplot an der linken Seite zeigt die tatsächlichen Daten.

2. Zeichnen Sie für $-1 \leq x \leq 1$ die Exponentialfunktion $f(x) = \exp(x)$ und die Parabel $g(x) = x^2/2 + x + 1$ mit dem `plot`-Befehl gemeinsam in einem Bild. Wiederholen Sie dies, wobei Sie den `curve`-Befehl verwenden.

3. Zeichnen Sie die Dichten von Lognormalverteilungen $LN(0.5, 1)$, $LN(2, 1)$ und $LN(0.5, 2)$ mit dem Befehl `curve` gemeinsam in einem Bild.

4. Machen Sie die Variablennamen des Data Frames **Animals** durch Eingabe von `attach(Animals)` direkt verfügbar.

   a) Plotten Sie Gehirngewicht gegen Körpergewicht für den **gesamten** Datensatz **Animals** in der MASS-Library. Beschriften Sie die Achsen geeignet. (Versehen Sie aber die Punkte **nicht** mit Beschriftungen.

   b) Wiederholen Sie den Plot aus Teil a), wobei Sie diesmal die Logarithmen (also `log(brain)` und `log(body)`) verwenden. Benutzen Sie die Zeilennamen, um die Punkte zu beschriften. Verwenden Sie dabei den Parameter `cex=0.5` in der Funktion `text()`, um die Schriftgröße zu verkleinern.

5. Die Jahresniederschlagsmengen in Karlsruhe in einem Zehnjahreszeitraum waren

   $$1014.5, 913.4, 711.9, 829.4, 695.8, 903.0, 810.2, 935.2, 632.3, 697.4$$

   Untersuchen Sie, ob die Annahme, dass die Daten aus einer normalverteilten Grundgesamtheit stammen, vernünftig ist.

6. a) Verwenden Sie `mfrow()`, um eine 3×4-Matrix von Plots zu erstellen. Erstellen Sie je vier QQ-Plots, die auf Zufalls-Stichproben vom Umfang 20 aus einer $N(0,1)$-Verteilung, einer $U(0,1)$-Verteilung und einer $Exp(1)$-Verteilung basieren.

   b) Wiederholen Sie den Plot aus Teil a), wobei Sie Stichproben vom Umfang 100 verwenden.

7. Untersuchen Sie die Verteilung der Daten in den ersten zwei Spalten des Data Frames **hills**, indem Sie

   a) Histogramme

   b) Dichteplots

   c) Boxplots

   d) QQ-Plots

   verwenden. Wiederholen Sie a) bis d) mit den logarithmierten Daten. Was sind die Vor- und Nachteile der einzelnen Darstellungen?

8.* Plotten Sie die Dichte einer Standardnormalverteilung, und zeichnen Sie einen „oberen Ablehnbereich eines Tests“ ein (indem Sie z.B. das obere 5%-Quantil markieren und die zugehörige Fläche kennzeichnen).

9.* Visualisieren Sie das empirische Gesetz von der Stabilisierung relativer Häufigkeiten. Simulieren Sie dazu eine große Anzahl von unabhängigen Wiederholungen eines Treffer/Niete-Experiments.

10.* Schreiben Sie eine Funktion, die die Verteilungsfunktion

$$G(z) \;=\; 1 - 2\sum_{j=1}^{\infty}(-1)^{j+1}e^{-2j^2z^2} \quad (z > 0)$$

der Kolmogorov-Smirnov-Verteilung berechnet und plotten Sie G.

11.* a) Beim Geburtstagsproblem ist nach der Wahrscheinlichkeit p_k gefragt, dass unter k rein zufällig ausgewählten Personen mindestens zwei an demselben Tag Geburtstag haben. Diese ist für $k \geq 2$ gegeben durch

$$p_k \;=\; 1 - \prod_{j=0}^{k-1}\left(\frac{365-j}{365}\right) \;=\; 1 - \prod_{j=1}^{k-1}\left(1 - \frac{j}{365}\right),$$

   falls die Annahme einer Gleichverteilung über alle 365 Tage gemacht wird. Bestimmen Sie p_k für $k = 1, \ldots, 50$ unter dieser Annahme und plotten Sie das Ergebnis.

b) Approximieren Sie die Wahrscheinlichkeit p_{23}, indem Sie das Experiment 10000 mal simulieren.
Hinweis: Ziehen mit Zurücklegen: Funktion `sample()`.
Bestimmung gleicher Elemente in einem Vektor: Funktion `duplicated()`.

c) Wiederholen Sie die Simulation, ohne die Annahme einer Gleichverteilung über alle 365 Tage zu machen. Verwenden Sie dazu die Option `prob` in der Funktion `sample()`.

12.* Visualisieren Sie den zentralen Grenzwertsatz am Beispiel von exponentialverteilten Zufallsvariablen.
Hinweis: Faltungsformel für die Gammaverteilung.

13.* Nehmen Sie an, es liegen ihnen ein großer Datensatz aus einer normalverteilten Grundgesamtheit vor. Welcher Anteil der Daten wird dann ungefähr bei einem Boxplot als Ausreißer deklariert?

4 Zwei-Stichproben-Tests

4.1 Der Zwei-Stichproben-t-Test

Modellannahme: $x_1, ..., x_n$ bzw. $y_1, ..., y_m$ sind Realisierungen von Zufallsvariablen $X_1, ..., X_n$ bzw. $Y_1, ..., Y_m$, wobei alle X_i,Y_j unabhängig sind mit Verteilung

$$\begin{aligned} X_i &\sim N(\mu, \sigma^2) \quad (i = 1, ..., n), \\ Y_j &\sim N(\nu, \sigma^2) \quad (j = 1, ..., m). \end{aligned}$$

μ, ν, σ^2 sind unbekannt.

Es liegt also ein Shift-Modell unter Normalverteilungsannahme vor (Abb. 4.1).

Beim **zweiseitigen** Zwei-Stichproben-t-Test

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

ist die **Prüfgröße**

$$T = \frac{\sqrt{\frac{m \cdot n}{m+n}} \cdot (\bar{x} - \bar{y})}{\sqrt{\frac{1}{m+n-2} \cdot \left((n-1) \cdot s_x^2 + (m-1) \cdot s_y^2\right)}} .$$

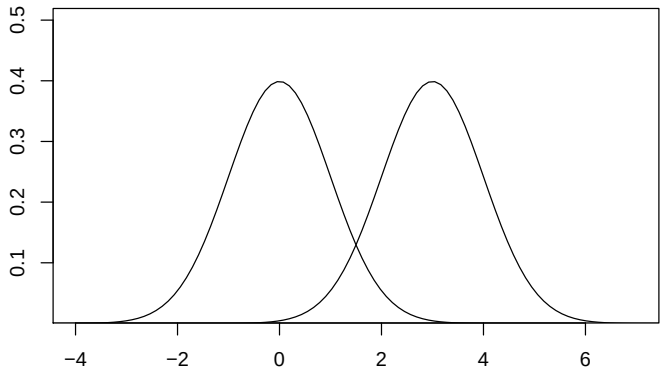


Abbildung 4.1: Modellannahme beim Zwei-Stichproben-t-Test

Testentscheid:

H_0 ablehnen, falls $|T| \geq t_{m+n-2,1-\alpha/2}$,

kein Widerspruch zu H_0, falls $|T| < t_{m+n-2,1-\alpha/2}$.

Bei Ablehnung von H_0 sind $\bar{x}$ und $\bar{y}$ „signifikant verschieden auf dem $\alpha \cdot 100$ %-Niveau".

Beim **einseitigen Test**

$$H_0 : \mu = \nu \text{ (bzw. } \mu \leq \nu) \quad \text{gegen} \quad H_1 : \mu > \nu$$

ist die Prüfgröße T dieselbe wie beim zweiseitigen Test.

Testentscheid: H_0 ablehnen, falls $T \geq t_{m+n-2,1-\alpha}$.

Kein Widerspruch zu H_0, falls $T < t_{m+n-2,1-\alpha}$.

Bei Ablehnung von H_0 ist „$\bar{x}$ signifikant größer als $\bar{y}$ auf dem $\alpha \cdot 100$ %-Niveau".

Will man dagegen $H_0 : \mu = \nu$ (bzw. $\mu \geq \nu$) gegen $H_1 : \mu < \nu$ testen, so lehnt man H_0 ab, falls $T \leq t_{m+n-2,\alpha}$ ist.

4.1.1 Beispiel

Untersucht wurde die Gewichtszunahme (in Gramm) bei Ratten, die mit unterschiedlichem Futter gefüttert wurden:

Gewichtszunahme bei Futter mit hohem Eiweißgehalt (n=12):
134 146 104 119 124 161 107 83 113 129 97 123

Gewichtszunahme bei Futter mit niedrigem Eiweißgehalt (m=7):
70 118 101 85 107 132 94

Frage: Liegt ein Unterschied in der Gewichtszunahme zwischen den beiden Gruppen vor?

Nach einem Boxplot der beiden Stichproben (Abb. 4.2) werden im folgenden QQ-Plots gezeichnet (Abb. 4.3); diese lassen keine Abweichungen von einer Normalverteilung erkennen. Danach wird ein zweiseitiger 2-Stichproben-t-Test durchgeführt.

```
> gain.high = c(134,146,104,119,124,161,107,83,113,129,97,123)
> gain.low = c(70,118,101,85,107,132,94)
> boxplot(gain.high,gain.low, boxwex=0.3)
> par(mfrow=c(1,2))
> qqnorm(gain.high, pch=16); qqline(gain.high)
> qqnorm(gain.low, pch=16); qqline(gain.low)
> t.test(gain.high, gain.low, alternative ="two.sided", var.equal = T)
```

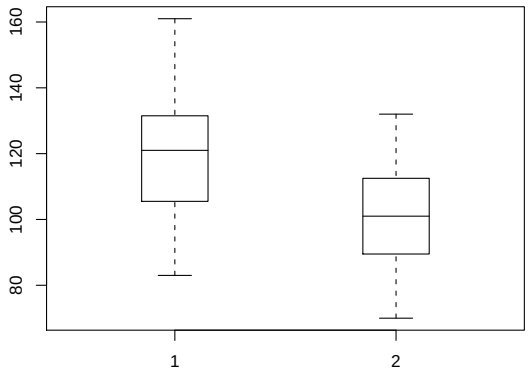


Abbildung 4.2: Boxplots der zwei Stichproben in Beispiel 4.1.1

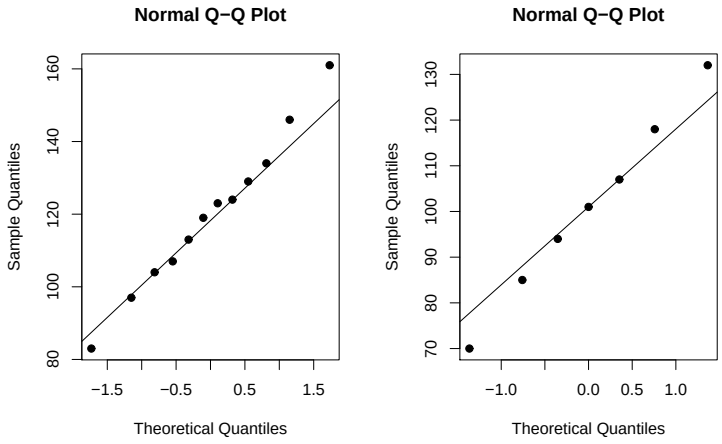


Abbildung 4.3: QQ-Plots der zwei Stichproben in Beispiel 4.1.1

```
      Two Sample t-test

data: gain.high and gain.low
t = 1.8914, df = 17, p-value = 0.07573
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-2.193679 40.193679
sample estimates:
mean of x mean of y
      120       101
```

Die Testgröße T hat den Wert `t=1.8914`. Die Stichprobenumfänge sind $n = 12$ und $m = 7$; somit ist $m+n-2 = 17$. Dies wird durch `df=17` angegeben, wobei `df` für degrees of freedom (Anzahl der Freiheitsgrade) steht. Die Hypothese H_0 wird abgelehnt, wenn der p-Wert kleiner oder gleich dem Niveau α ist. Im Beispiel gilt `p-value = 0.07573`, so dass die Hypothese gleicher Gewichtszunahme auf dem 5%-Niveau nicht verworfen werden kann. Weil die Stichprobenumfänge klein sind, ist das 95%-Konfidenzintervall $(-2.2, 40.2)$ für $\mu - \nu$ sehr groß.

Frage: „Macht Eiweiß dick"?

Ist eine Gewichtsabnahme durch besonders eiweißreiches Futter ausgeschlossen, so kann man einen einseitigen 2-Stichproben-t-Test

$$H_0 : \mu \leq \nu \quad \text{gegen} \quad H_1 : \mu > \nu$$

verwenden.

```
> t.test(gain.high, gain.low, alternative="greater", var.equal = T)

      Two Sample t-test

data: gain.high and gain.low
t = 1.8914, df = 17, p-value = 0.03787
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
1.525171 Inf
sample estimates:
mean of x mean of y
      120       101
```

Beim einseitigen Test wird die Hypothese gleicher Gewichtszunahme auf dem 5%-Niveau verworfen.

4.2 Der Welch-Test

Modellannahme wie in Abschnitt 4.1, jedoch gilt jetzt

$$X_i \sim N(\mu, \sigma^2), \qquad Y_j \sim N(\nu, \tau^2);$$

dabei sind $\mu, \nu, \sigma^2, \tau^2$ wieder unbekannt, es ist aber $\sigma^2 \neq \tau^2$ möglich (siehe Abb. 4.4).

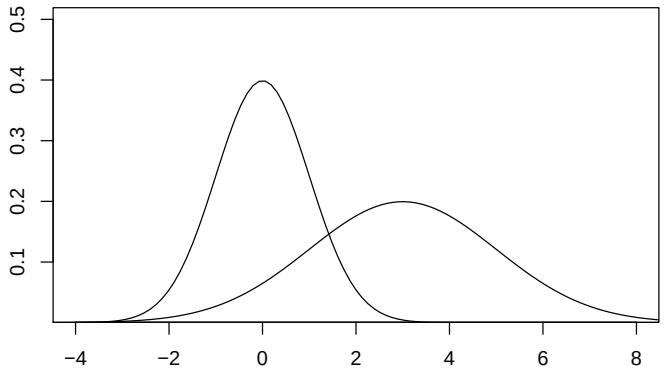


Abbildung 4.4: Modellannahme beim Welch-Test

Prüfgröße:

$$T \;=\; \frac{\bar{x}-\bar{y}}{\sqrt{\frac{1}{n}s_x^2+\frac{1}{m}s_y^2}}$$

Im Fall $\mu=\nu$ ist $T(X_1,...,X_n,Y_1,...,Y_m)$ approximativ t_k-verteilt, wobei

$$k \;=\; \left[1\Big/\left(\frac{c^2}{n-1}+\frac{(1-c)^2}{m-1}\right)\right], \qquad c \;=\; \frac{s_x^2}{n}\Big/\left(\frac{s_x^2}{n}+\frac{s_y^2}{m}\right).$$

Zweiseitiger Welch-Test:

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

H_0 ablehnen, falls $|T| \geq t_{k,1-\alpha/2}$,
kein Widerspruch zu H_0, falls $|T| < t_{k,1-\alpha/2}$.

Einseitiger Welch-Test:

$$H_0 : \mu = \nu \text{ (bzw.} \mu \leq \nu) \quad \text{gegen} \quad H_1 : \mu > \nu.$$

H_0 ablehnen, falls $T \geq t_{k,1-\alpha}$,
Kein Widerspruch zu H_0, falls $T < t_{k,1-\alpha}$.

4.2.1 Beispiel

Ein Welch-Test wird durch die Option `var.equal=F` in der Funktion `t.test` angefordert.

```
> t.test(gain.high, gain.low, alternative ="two.sided", var.equal=F)

      Welch Two Sample t-test

data: gain.high and gain.low
t = 1.9107, df = 13.082, p-value = 0.0782
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-2.469073 40.469073
sample estimates:
mean of x mean of y
      120       101
```

Der Test liefert dieselben Resultate wie der t-Test; das gilt auch für die einseitige Testdurchführung.

4.3 Der Mann-Whitney-U-Test oder Wilcoxon-Test

Modellannahme: $x_1, \ldots, x_n, y_1, \ldots, y_m$ Realisierungen unabhängiger Zufallsvariablen $X_1, \ldots, X_m$, $Y_1, \ldots, Y_n$, wobei

$X_i \sim F$ mit Dichte f,
$Y_j \sim G$ mit Dichte g.

Der zweiseitige Test

$$H_0 : F = G \quad \text{gegen} \quad H_1 : F \neq G.$$

Der U-Test erkennt selbst bei großem n nicht jeden Unterschied zwischen F und G, aber er erkennt Lage-Unterschiede sehr gut (siehe Abb. 4.5).

Annahme: alle $m + n$ Stichprobenwerte sind verschieden

Die Wilcoxon-Rangsummen-Statistik W ist die Summe der Ränge von $x_1, ..., x_n$ unter allen x- und y-Werten („gepoolte“ Stichprobe)

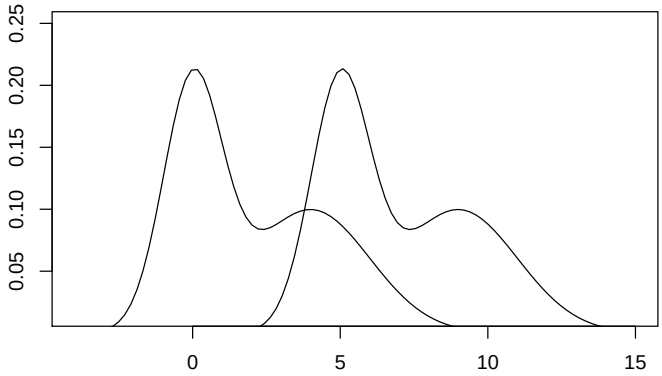
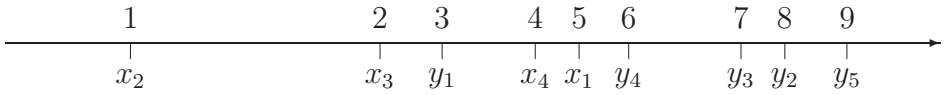


Abbildung 4.5: Dichten, die sich nur in der Lage unterscheiden



Bsp: $W = 1 + 2 + 4 + 5 = 12$.

Die **Prüfgröße des U-Tests** ist dann

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

Testentscheid: H_0 verwerfen, falls

$$U \geq U_{m,n,1-\alpha/2} \quad \text{oder} \quad U \leq m \cdot n - U_{m,n,1-\alpha/2}.$$

Der **einseitige Test** testet die Hypothese

$$H_0 : F = G \qquad (\text{bzw. } F \leq G, \text{ d.h. } F(x) \leq G(x),\ x \in \mathbb{R})$$

gegen die Alternative

$$H_1 : F \geq G \ \text{ und } \ F(x) > G(x) \ \text{ für mindestens ein } \ x.$$

Sei $X \sim F, \quad Y \sim G$.

Sprechweise für H_1: „X ist **stochastisch kleiner** als Y"

Bei einem Lage-Unterschied ist unter H_1 der X-Median kleiner als der Y-Median.

Der Test verwendet dieselbe Prüfgröße U wie der zweiseitige Test;

H_0 wird verworfen, falls $U \leq m \cdot n - U_{m,n;1-\alpha}$.

Das Problem von Bindungen: In der Praxis treten häufig aufgrund beschränkter Messgenauigkeit mehrfache Messwerte (Bindungen) auf. Dann müssen Bindungskorrekturen verwendet werden.

4.3.1 Beispiel

Der Aufruf des zweiseitigen U-Tests bzw. Wilcoxon-Tests erfolgt mit dem Befehl

```
> wilcox.test(gain.high, gain.low, alternative="two.sided",
  exact=F, correct=T)

        Wilcoxon rank sum test with continuity correction

data: gain.high and gain.low
W = 62.5, p-value = 0.09083
alternative hypothesis: true mu is not equal to 0
```

Der einseitige U-Test von $H_0 : X$ ist stochastisch kleiner als Y gegen die Alternative $H_1 : X$ ist stochastisch größer als Y wird wieder mit der Option `alternative = "greater"` angefordert.

```
> wilcox.test(gain.high, gain.low, alternative="greater")

        Wilcoxon rank sum test with continuity correction

data: gain.high and gain.low
W = 62.5, p-value = 0.04541
alternative hypothesis: true mu is greater than 0

Warning message:
Cannot compute exact p-value with ties in:
wilcox.test.default(gain.high, gain.low, alternative="greater")
```

Der Test wird im Beispiel mit den Standardeinstellungen `exact=T, correct=T` aufgerufen; die exakte Berechnung des p-Wertes ist aber nicht möglich, da in den Daten Bindungen vorhanden sind.

4.4 Varianz-Quotienten-Test

Modellannahme: Die Daten $x_1, \ldots, x_n, y_1, \ldots, y_m$ sind Realisierungen von Zufallsvariablen

$$X_i \sim N(\mu, \sigma^2), \quad i = 1, \ldots, n,$$
$$Y_j \sim N(\nu, \tau^2), \quad i = 1, \ldots, m,$$

$X_1, \ldots, X_n, Y_1, \ldots, Y_m$ sind unabhängig. Getestet wird

$$H_0 : \ \sigma^2 \ = \ \tau^2 \quad \text{gegen} \quad H_1 : \ \sigma^2 \ \neq \ \tau^2.$$

Prüfgröße:

$$T \ = \ \frac{s_y^2}{s_x^2} \ = \ \frac{\frac{1}{m-1}\sum_{j=1}^{m}(y_j - \bar{y})^2}{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

4.4.1 Beispiel

Mit dem Varianz-Quotienten-Test wird im folgenden geprüft, ob die beiden Stichproben in Beipiel 4.1.1 aus (Normal-)Verteilungen mit gleichen Varianzen stammen.

```
> var.test(gain.high, gain.low)

      F test to compare two variances

data: gain.high and gain.low
F = 1.0755, num df = 11, denom df = 6, p-value = 0.9788
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
0.198811 4.173718
sample estimates:
ratio of variances
      1.07552
```

Neben dem p-Wert von 0.98 wird auch ein 95%-Konfidenzintervall für τ^2/σ^2 berechnet; es ergibt sich das Intervall $[0.2, 4.2]$.

Der Varianz-Quotienten-Test wird oft als Vortest vor einem Zwei-Stichproben-t-Test verwendet. Gegen diese Praxis sprechen mehrer Gründe:

- Es liegt ein multiples Testproblem vor (siehe Abschnitt 5.3).
- Die Annahme gleicher Varianzen ist nicht gesichert, wenn die Hypothese nicht verworfen wird.
- Der Test ist nicht robust; auch eine (geringe) Abweichung von der Normalverteilung kann zu falschen Testentscheidungen führen.

Statt dessen schaut man sich Boxplots der Stichproben an; deuten diese auf stark unterschiedliche Varianzen hin, so verwendet man den Welch-Test, oder besser gleich den U-Test.

4.5 Der t-Test bei verbundenen Stichproben

Situation: Es liegen Daten $x_1, ..., x_n, y_1, ..., y_n$ vor, wobei x_i und y_i **zum gleichen** i-ten Versuchsobjekt gehören (verbundene Stichproben).

Modell: Sei $z_i = x_i - y_i$ $(i = 1, \ldots, n)$. Dann seien $z_1, ..., z_n$ Realisierungen von unabhängigen $N(\mu, \sigma^2)$-verteilten Zufallsvariablen; dabei sind μ und σ^2 unbekannt.

Beim **zweiseitigen Test**

$$H_0 : \ \mu \ = \ 0 \quad \text{gegen} \quad H_1 : \ \mu \ \neq \ 0.$$

ist die **Prüfgröße**

$$T \ = \ \frac{\sqrt{n} \cdot \bar{z}}{\sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (z_i - \bar{z})^2}} \ , \quad \text{wobei} \quad \bar{z} = \frac{1}{n} \sum_{i=1}^{n} z_i = \bar{x} - \bar{y}.$$

H_0 wird abgelehnt, falls $|T| \geq t_{n-1,1-\alpha/2}$.
Es besteht kein Widerspruch zu H_0, falls $|T| < t_{n-1,1-\alpha/2}$.

Der **einseitige Test**

$$H_0 : \mu = 0 \ (\text{ bzw. } \mu \leq 0) \quad \text{gegen} \quad H_1 : \mu > 0$$

verwendet dieselbe Testgröße.

H_0 wird abgelehnt, falls $T \geq t_{n-1,1-\alpha}$.
Es besteht kein Widerspruch zu H_0, falls $T < t_{n-1,1-\alpha}$.

Ist die Annahme einer Normalverteilung zweifelhaft, so können wieder nichtparametrische Tests wie zum Beispiel der Vorzeichen-Rang-Test von Wilcoxon verwendet werden.

Person	Material A	Material B	Material A - Material B
1	14.0(R)	13.2(L)	0.8
2	8.8(R)	8.2(L)	0.6
3	11.2(L)	10.9(R)	0.3
4	14.2(R)	14.3(L)	-0.1
5	11.8(L)	10.7(R)	1.1
6	6.4(R)	6.6(L)	-0.2
7	9.8(R)	9.5(L)	0.3
8	11.3(R)	10.8(L)	0.5
9	9.3(L)	8.8(R)	0.5
10	13.6(R)	13.3(L)	0.3

Tabelle 4.1: Materialverbrauch bei zwei verschiedenen Materialien

4.5.1 Beispiel

Bei 10 Personen wurde der Materialverbrauch an den Sohlen gemessen; dabei kamen zwei unterschiedliche Materialien zum Einsatz. Bei jeder Person wurde dabei ein Schuh mit Material A besohlt, der andere mit Material B. Die Wahl des Schuhs, der mit Material A besohlt wurde, geschah zufällig. Tabelle 4.1 enthält die Messwerte[1].

```
> Material.A = c(14.0,8.8,11.2,14.2,11.8,6.4,9.8,11.3,9.3,13.6)
> Material.B = c(13.2,8.2,10.9,14.3,10.7,6.6,9.5,10.8,8.8,13.3)
> t.test(Material.A, Material.B, var.equal=T)
   Two Sample t-test

data: Material.A and Material.B
t = 0.3689, df = 18, p-value = 0.7165
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-1.924924 2.744924
sample estimates:
mean of x mean of y
      11.04       10.63
> plot(Material.A, Material.B); abline(0,1)
```

Der Zwei-Stichproben-t-Test liefert keinen Hinweis auf Unterschiede im Materialverbrauch. Allerdings deutet der Scatterplot (Abb. 4.6) auf eine große Variabilität innerhalb der Stichprobe hin (within-sample variability), während der Unterschied bei einer

[1] vgl. Venables, Ripley: Modern Applied Statistics with S, 4. Auflage, Abschnitt 5.4

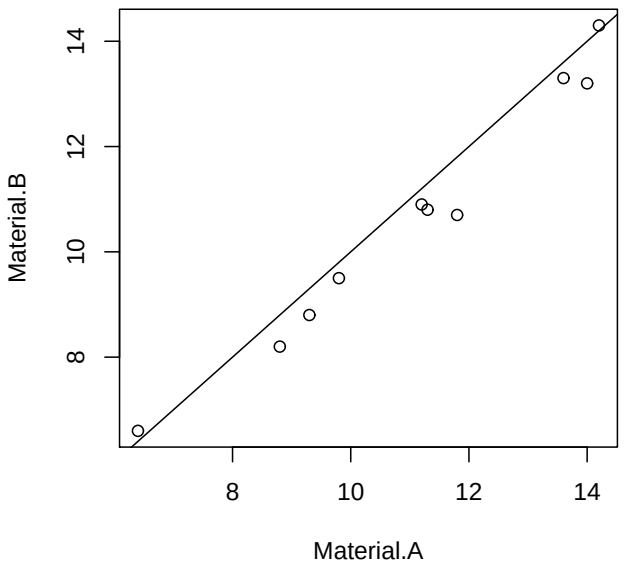


Abbildung 4.6: Scatterplot der Daten in Beispiel 4.5.1

Person (within-subject variability) sehr gering ist; die Datenpunkte liegen nämlich recht genau auf der Diagonalen.

Deshalb sollte hier ein verbundener Zwei-Stichproben-t-Test (Option `paired=T`) verwendet werden!

```
> t.test(Material.A, Material.B, paired = TRUE)

      Paired t-test

data: Material.A and Material.B
t = 3.3489, df = 9, p-value = 0.008539
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
0.1330461 0.6869539
sample estimates:
mean of the differences
      0.41
```

Die Hypothese gleicher Mittelwerte wird jetzt auf dem 1%-Niveau verworfen. Der Vorzeichen-Rang-Test von Wilcoxon liefert ein ähnliches Ergebnis:

```
> wilcox.test(Material.A, Material.B, exact=F, paired=TRUE)

      Wilcoxon signed rank test with continuity correction
```

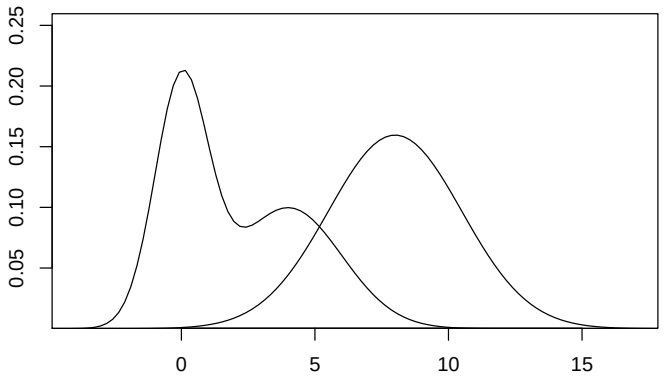


Abbildung 4.7: Situation beim Kolmogorov-Smirnov-Test: Zwei beliebige Dichten

```
data: Material.A and Material.B
V = 52, p-value = 0.01431
alternative hypothesis: true mu is not equal to 0
```

Durch Ausschalten der Variabilität zwischen den Personen ist es gelungen, einen Unterschied zwischen den Materialien statistisch zu sichern. Dies mag für einen Materialforscher von Interesse sein. Anders sieht dies für einen Schuhfabrikanten aus: der Unterschied zwischen den Materialien ist so gering, dass er praktisch keinen Einfluß auf die Lebensdauer der Sohlen hat; der Fabrikant wird die Materialwahl daher nach anderen Gesichtspunkten treffen. Hier zeigt sich:

Statistisch signifikant bedeutet nicht immer auch praktisch relevant!

4.6 Zwei-Stichproben-Kolmogorov-Smirnov-Test

Modellannahme: $x_1, ..., x_n,\ y_1, ..., y_m$ sind Realisierungen unabhängiger Zufallsvariablen $X_1, \ldots, X_n, Y_1, \ldots, Y_m$;

$X_i \sim F$ mit Dichte f, $Y_j \sim G$ mit Dichte g.

Der Kolmogorov-Smirnov-Test testet

$$H_0 : \ F \ = \ G \ \text{ gegen } \ H_1 : \ F \ \neq \ G;$$

er erkennt jede Alternative für „genügend großes" n.

Der Kolmogorov-Smirnov-Test vergleicht die empirischen Verteilungsfunktionen

$F_n(t)$ = relativer Anteil der $x_1, ..., x_n$, die $\leq t$ sind

und

$G_m(t)$ = relativer Anteil der $y_1, ..., y_m$, die $\leq t$ sind.

Die **Prüfgröße** ist

$$T = \max_t |F_n(t) - G_m(t)| .$$

H_0 zum Niveau α ablehnen, falls $T > \frac{K_{m,n,\alpha}}{m \cdot n}$.

Kein Widerspruch zu H_0, falls $T \leq \frac{K_{m,n,\alpha}}{m \cdot n}$.

Beachte: F_n und G_m sind zwischen Datenpunkten konstant.

4.6.1 Beispiel

Im folgenden werden die zu den Stichproben in Beispiel 4.1.1 gehörenden empirischen Verteilungsfunktionen gezeichnet (Abb. 4.8) und die Hypothese gleicher Verteilungen mit dem Kolmogorov-Smirnov-Test geprüft.

```
> plot.ecdf(gain.high, verticals=T, xlab="", main="",
    xlim=c(65,165), cex=0.5, pch=16)
> plot.ecdf(gain.low, add=T, verticals=T, lty=2, cex=0.5, pch=16,
    col.hor="red", col.vert="red", col.points="red")
> ks.test(gain.high, gain.low)

      Two-sample Kolmogorov-Smirnov test

data: gain.high and gain.low
D = 0.4405, p-value = 0.3576
alternative hypothesis: two.sided

Warning message:
cannot compute correct p-values with ties in: ks.test(gain.high, gain.low)
```

Unterscheiden sich wie beim U-Test in Abschnitt 4.3 die Dichten nur in der Lage, so kann man den Kolmogorov-Smirnov-Test auch einseitig durchführen. Allerdings wird hier der einseitige Test von $H_0 : X$ ist stochastisch kleiner als Y gegen die Alternative $H_1 : X$ ist stochastisch größer als Y mit der Option `alternative = "less"` angefordert. Dies

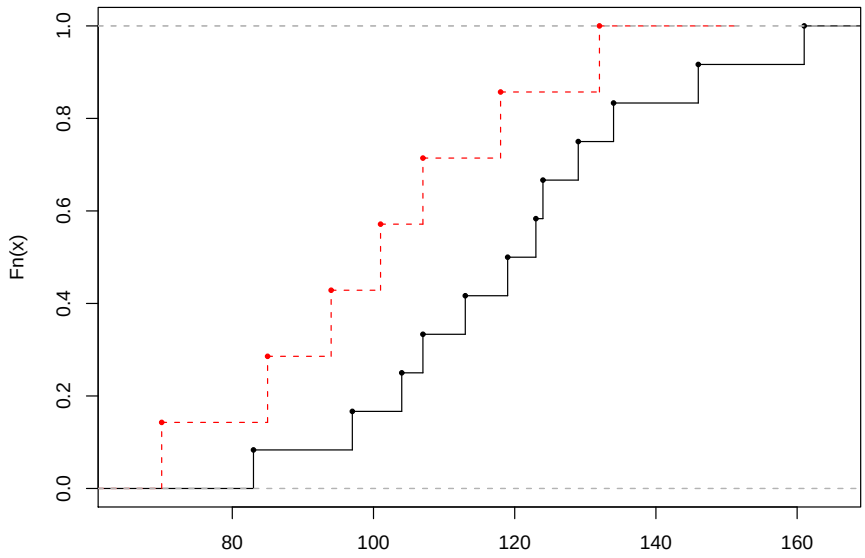


Abbildung 4.8: Plot der empirischen Verteilungfunktionen in Beispiel 4.6.1; gain.high: durchgezogen, gain.low: gestrichelt

bezieht sich auf die Verteilungsfunktionen: unter der Alternative ist $F(x) \leq G(x)$ für alle x.

```
> ks.test(gain.high, gain.low, alternative="less")

        Two-sample Kolmogorov-Smirnov test

data: gain.high and gain.low
D^- = 0.4405, p-value = 0.1799
alternative hypothesis: less
```

Als p-Wert erhält man beim einseitigen Test 0.18; beim einseitigen U-Test und t-Test lag der p-Wert unter 0.05. Dies ist eine häufig anzutreffende Situation: sind die Voraussetzungen des t-Tests bzw. des U-Tests approximativ erfüllt, so besitzen diese eine höhere Güte als der Kolmogorov-Smirnov-Test. Aber nur der Kolmogorov-Smirnov-Test ist in der Lage, zumindest für große Stichprobenumfänge jede Alternative zu erkennen.

4.7 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 4 nach.

2. Der Datensatz **birthwt** in der Library MASS enthält Daten von Geburten aus einem Krankenhaus in Springfield aus dem Jahre 1986. Erstellen Sie Vektoren mit Geburtsgewichten von Kindern mit weißen bzw. mit schwarzen Müttern durch

```
> library(MASS); data(birthwt); attach(birthwt)
> bwt1 = bwt[race==1]
> bwt2 = bwt[race==2]
```

   a) Berechnen Sie die üblichen Kennzahlen wie Stichprobenumfang, Mittelwert, Median, usw. der beiden Stichproben.

   b) Zeichen Sie Histogramme, Dichteplots, Boxplots und QQ-Plots der beiden Stichproben.

   c) Führen Sie einen Test auf Gleichheit der Varianzen durch.

   d) Verwenden Sie den t-Test, den Welch-Test und den U-Test, um zu testen, ob beide Stichproben aus der gleichen Grundgesamtheit stammen.

   e) Zeichnen Sie die empirischen Verteilungsfunktionen beider Stichproben und testen Sie mit dem Zwei-Stichproben-Kolmogorov-Smirnov-Test auf Gleichheit der zugrundeliegenden Verteilungen.

3. Bei einem Versuch mit 10 Hunden (gleiche Rasse, gleiches Geschlecht, gleiches Alter) wurde der Blutdruck gemessen. Nach Gabe eines Pharmakons über einen gewissen Zeitraum wurde der Blutdruck ein zweites Mal gemessen. Untersuchen Sie, ob das Pharmakon blutdrucksenkende Eigenschaften hat.

Nr.	Blutdruck (in mm Hg)	
	ohne Medikament	mit Medikament
1	80	75
2	85	80
3	110	115
4	120	100
5	70	60
6	90	85
7	110	110
8	110	100
9	95	85
10	120	110

4. a) Laden Sie den Datensatz `vitcap` in der Bibliothek `ISwR`. Er enthält Messungen der Lungenfunktion (Merkmal `vital.capacity`) von Arbeitern in der

Cadmium-Industrie; dabei bedeutet ein größerer Wert eine bessere Lungenfunktion. Ein Teil der Arbeiter war über 10 Jahre lang Cadmium ausgesetzt (`group==1`); der andere war nicht exponiert (`group==3`). Führen Sie einen t-Test durch, um die Lungenfunktion in den beiden Gruppen zu vergleichen. Bestimmen Sie ein 99%-Konfidenzintervall für die Differenz.

b) Warum können die Testergebnisse aus Teil a) irreführend sein?
Hinweis: Plotten Sie die Vitalkapazität in Abhängigkeit vom Alter und markieren Sie dabei die unterschiedlichen Gruppen durch

```
plot(vital.capacity ~ age, pch = group, data = vitcap)
```

5.* Bei der Behandlung einer Erkrankung sei bekannt, dass in 20% der Fälle postoperative Komplikationen auftreten. Ein Chirurg testet ein neues Verfahren an 10 Patienten; dabei treten keine Komplikationen auf. Ist die neue Methode besser?

(Formulieren Sie eine geeignete Hypothese und Alternative. Geben Sie eine geeignete Testgröße an. Bestimmen Sie den p-Wert).

6.* Erzeugen Sie 20 standardnormalverteilte Zufallsvariablen und führen Sie mit Hilfe der Funktion `t.test` einen Ein-Stichproben-t-test ($H_0 : \mu = 0$ gegen $H_1 : \mu \neq 0$) durch. Speichern Sie den p-Wert ab (durch `t.test(x)$p.value` kann man auf den p-Wert zugreifen).

Wiederholen Sie dies 1000 mal und untersuchen Sie die empirische Verteilung des p-Werts (Histogramm, Dichteplot,...).

7.* Erzeugen Sie 20 standard-normalverteilte Pseudozufallszahlen. Testen Sie mit dem Ein-Stichproben-t-Test die Hypothese $\mu = 0$ gegen die Alternative $\mu \neq 0$ (Niveau $\alpha = 0.05$). Automatisieren Sie dies so, dass Sie 5000 Wiederholungen durchführen können, und bestimmen Sie das empirische Niveau des Tests (d.h. den relativen Anteil an Ablehnungen).

Wiederholen Sie das Experiment, wobei Sie folgende Verteilungen verwenden: $Y = X - 1$ mit $X \sim Exp(1)$, $Y \sim t_2$ und $Y = 2X - 1$ mit $X \sim Bin(1, 1/2)$ (es gilt immer $EY = 0$).

5 Varianzanalyse und Multiple Vergleiche

5.1 Einfache Varianzanalyse

Die m-fache Varianzanalyse (ANOVA) dient der Untersuchung der Wirkung von m Einflussgrößen (Faktoren) auf eine Zielgröße.

Im Fall $m = 1$ (einfache ANOVA) hat man die folgende Situation: Gegeben sind Daten (stetiges Merkmal), die in k (≥ 3) Gruppen (Klassen) unterteilt sind.

$$\begin{array}{cccc} x_{1,1} & x_{1,2} & \cdots & x_{1,n_1} \quad (1.\text{ Gruppe, Umfang} = n_1) \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n_2} \quad (2.\text{ Gruppe, Umfang} = n_2) \\ \vdots & \vdots & \vdots & \vdots \qquad\qquad \vdots \\ x_{k,1} & x_{k,2} & \cdots & x_{k,n_k} \quad (k\text{-te Gruppe, Umfang} = n_k) \end{array}$$

Der Gesamt-Stichprobenumfang ist $n = n_1 + n_2 + ... + n_k$.

Beispiel:
Einflussgröße: Düngersorte (k Möglichkeiten),
Zielgröße: Ertrag unter sonst gleichen Versuchsbedingungen (Boden, Klima,...)

$x_{i,j}$ ist der Ertrag mit Düngersorte i auf Parzelle j.

Modellannahme: $x_{i,j}$ ist eine Realisierung der Zufallsvariablen $X_{i,j}$, wobei alle $X_{i,j}$ unabhängig sind, und

$$X_{i,j} \quad \sim \quad N(\mu_i, \sigma^2).$$

$\mu_1, ..., \mu_k, \sigma^2$ sind unbekannt. μ_i ist das theoretisches Verteilungsmittel der i-ten Gruppe, die Varianzen sind in allen Gruppen gleich! Es liegt also ein Shift-Modell unter Normalverteilungsannahme vor (Abb. 5.1).

Getestet wird die Hypothese

$$H_0: \qquad \mu_1 = \mu_2 = \mu_3 = \ldots = \mu_k \quad \text{(gleiche Grundgesamtheiten)}$$

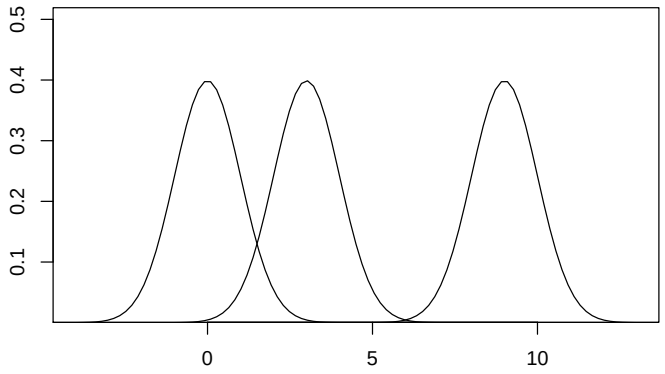


Abbildung 5.1: Modellannahme bei der einfachen Varianzanalyse

gegen die Alternative

$$H_1: \quad \text{mindestens zwei der } \mu_i \text{ sind verschieden.}$$

Mit den **Bezeichnungen**

$$\bar{x}_{i+} = \frac{1}{n_i}\sum_{j=1}^{n_i} x_{i,j} \quad (i\text{-tes Gruppen-Mittel}),$$

$$\bar{x}_{++} = \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{n_i} x_{i,j} \quad (\text{Gesamt-Mittel})$$

gilt

$$\sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{i,j}-\bar{x}_{++})^2 = \sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{i,j}-\bar{x}_{i+})^2 + \sum_{i=1}^{k} n_i(\bar{x}_{i+}-\bar{x}_{++})^2$$

$$= SQI + SQZ,$$

wobei

SQI = Summe der Quadrate innerhalb der Gruppen,
SQZ = Summe der Quadrate zwischen den Gruppen.

Als **Prüfgröße** verwendet man

$$T = \frac{MQZ}{MQI},$$

wobei $MQI = SQI/(n-k)$ die mittlere Quadratsumme innerhalb der Gruppen und $MQZ = SQZ/(k-1)$ die mittlere Quadratsumme zwischen den Gruppen ist.

H_0 wird für "große" Werte von T abgelehnt.

5.1.1 Beispiel

Die Erträge einer Getreideart in kg werden bei 3 Düngersorten auf jeweils 5 Parzellen gemessen.

	Versuchsnummer				
Dünger Nr.	1	2	3	4	5
1	8.3	9.3	10.2	8.8	10.7
2	8.7	8.9	9.4	8.6	9.2
3	10.7	9.9	9.6	10.9	10.2

Im folgenden R-Code werden zuerst die Daten als Dataframe eingegeben. Dabei wird die Düngersorte als Faktor mit drei Ausprägungen 1,2,3 definiert. Ein Faktor beschreibt qualitative, also nominale oder ordinale, Merkmale. In R ist ein Faktor eine spezielle Art von Vektor, der intern als numerischer Vektor mit Werten $1, 2, \ldots, k$ gespeichert wird. k gibt die Anzahl der Merkmalsausprägungen an. In den Analysen wird ein Faktor jedoch anders behandelt wie eine numerische Variable, wie man schon am Ergebnis des Befehls `summary(duenger)` sehen kann.

Für die ANOVA muss die unabhängige Variable ein Faktor oder ein Character-Vektor sein; sonst wird eine falsche Analyse durchgeführt.

Vor der Varianzanalyse werden Boxplots des Ertrags in Abhängigkeit von der Düngersorte gezeichnet (Abb. 5.2).

```
> duenger = data.frame(sorte=factor(c(rep(1,5),rep(2,5),rep(3,5))),
  ertrag=c(8.3,9.3,10.2,8.8,10.7,8.7,8.9,9.4,8.6,9.2,10.7,9.9,9.6,
  10.9,10.2))
> summary(duenger)
 sorte  ertrag
 1:5    Min. : 8.30
 2:5    1st Qu.: 8.85
 3:5    Median : 9.40
        Mean : 9.56
        3rd Qu.:10.20
        Max. :10.90
```

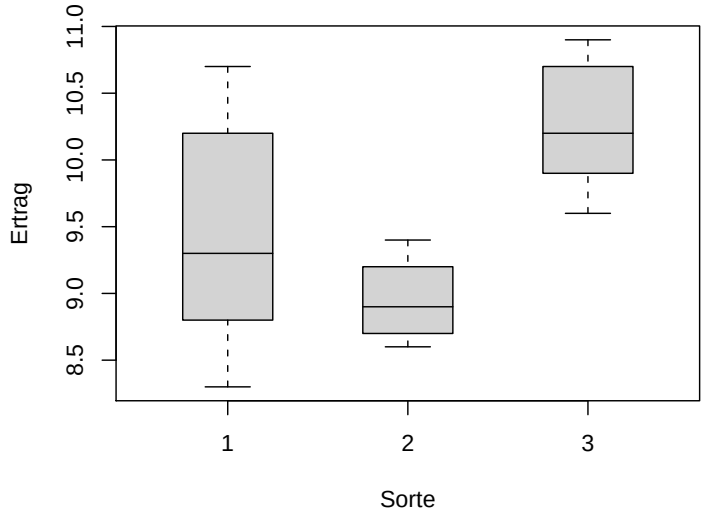


Abbildung 5.2: Boxplot der Stichproben in Beispiel 5.1.1

```
> boxplot(ertrag ~ sorte, data=duenger, xlab="Sorte", ylab="Ertrag",
  main="", col="lightgray", boxwex=0.5)
> duenger.aov = aov(ertrag ~ sorte, data=duenger)
> summary(duenger.aov)

            Df  Sum Sq  Mean Sq  F value   Pr(>F)
 sorte       2  4.3000   2.1500   4.6773  0.03149  *
 Residuals  12  5.5160   0.4597
-- Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Ausgegeben wird die sogenannte ANOVA-Tabelle, die immer die folgende Form hat:

	Freiheits-grade	Summe der Abweichungs-quadrate	Mittlere Summe	F-Statistik	p-Wert
zwischen den Gruppen	$k-1$	SQZ	MQZ	T	p
Residual-streuung	$n-k$	SQI	MQI		

Der p-Wert ist hier 0.03, auf dem 5 %-Niveau sind also mindestens zwei Mittelwerte signifikant verschieden.

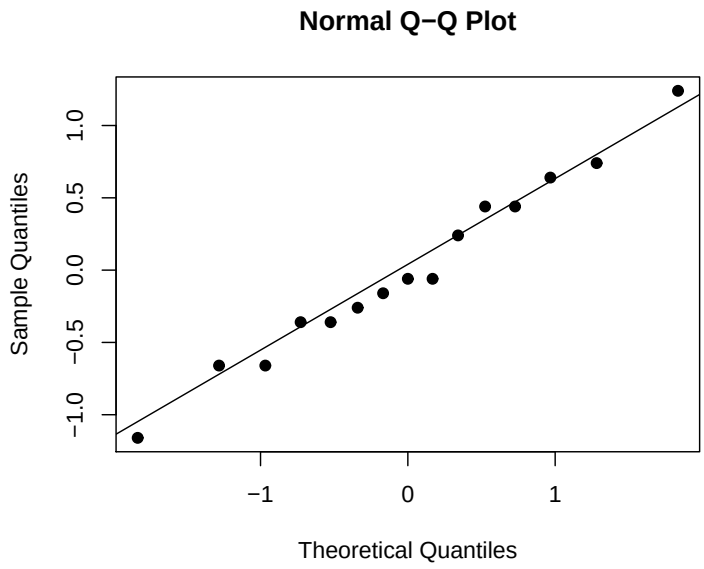


Abbildung 5.3: QQ-Plot der Residuen

Die **Frage**, welche Mittelwerte verschieden sind, wird durch dieses Ergebnis allerdings nicht beantwortet!

Um zu prüfen, ob die Annahmen der ANOVA zumindest approximativ erfüllt sind, sollte man neben den Boxplot einen QQ-Plot der Residuen $x_{i,j} - \bar{x}_{i+}$ und einen Plot der Residuen gegen die Mittelwerte $\bar{x}_{i+}$ zeichnen. Der QQ-Plot in Abb. 5.3 zeigt keine gravierenden Abweichungen von einer Normalverteilung. Dagegen lassen sowohl der Boxplot in Abb. 5.2 als auch der Residuen-Plot in Abb. 5.4 Zweifel an der Annahme gleicher Varianzen aufkommen.

```
> resd = residuals(duenger.aov)
> qqnorm(resd, pch=16)
> qqline(resd)
> plot(resd ~ fitted.values(duenger.aov), pch=16)
> abline(h=0)
```

Als formale Tests auf Gleichheit der Varianzen in allen k Gruppen steht in R der parametrische Bartlett-Test (Aufruf durch `bartlett.test(duenger$ertrag,duenger$sorte)`) und der Fligner-Killeen-Test (Funktion `fligner.test()`) zur Verfügung; der Fligner-Killeen-Test ist robust gegen Abweichungen von der Normalverteilungsannahme.

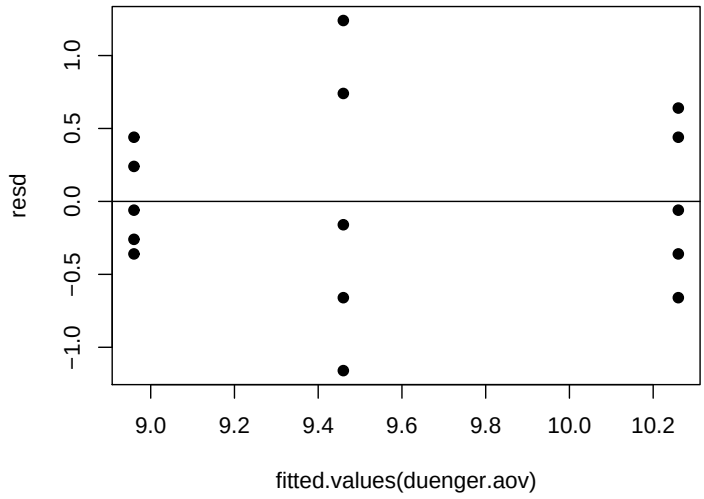


Abbildung 5.4: Plot der Residuen gegen die Mittelwerte

5.1.2 Bemerkung

Im obigen Beispiel werden aus dem ANOVA-Objekt `duenger.aov` mit dem Befehl `residuals(duenger.aov)` die Residuen extrahiert, die neben weiteren Informationen in `duenger.aov` gespeichert sind. Alternativ kann man auf die Residuen wie auf die Spalten eines Dataframes zugreifen:

```
> duenger.aov$residuals

   1      2      3      4      5      6      7      8      9
-1.16  -0.16   0.74  -0.66   1.24  -0.26  -0.06   0.44  -0.36
  10     11     12     13     14     15
0.24   0.44  -0.36  -0.66   0.64  -0.06
```

Analog erhält man durch die Funktion `fitted.values()` die angepassten Werte, dass sind bei einer einfachen Varianzanalyse die Gruppenmittelwerte. Die Funktion `coef()` liefert die Modellkoeffizienten, was insbesondere bei Regressionsmodellen wichtig ist. Bei einer ANOVA ist der erste Koeffizient der Mittelwert der ersten Gruppe $\bar{x}_{1+}$; die weiteren Koeffizienten sind die Differenzen $\bar{x}_{i+} - \bar{x}_{1+}$.

```
> fitted.values(duenger.aov)
```

```
  1     2     3     4     5     6     7     8     9
9.46  9.46  9.46  9.46  9.46  8.96  8.96  8.96  8.96

 10    11    12    13    14    15
8.96 10.26 10.26 10.26 10.26 10.26
```

```
> coef(duenger.aov)
 (Intercept)  sorte2  sorte3
    9.46      -0.50    0.80
```

5.2 Verteilungsfreie Lage-Vergleiche (Kruskal-Wallis-Test)

Es liegt die Situation wie in Abschnitt 5.1 vor; es wird aber keine Normalverteilungsannahme gemacht, sondern man nimmt nur an, dass ein stetiges Merkmal vorliegt.

Die Dichten der Verteilungen der k Gruppen seien $f_1, f_2, \ldots, f_k$.

Um zu testen, ob alle Dichten gleich sind, kann man in dieser Situation den Kruskal-Wallis-Test verwenden. Er ist eine Verallgemeinerung des U-Tests für zwei Gruppen, und verwendet ebenfalls die Ränge der Beobachtungen. Wie der U-Test erkennt auch der Kruskal-Wallis-Test Lage-Unterschiede besonders gut.

5.2.1 Beispiel

```
> kruskal.test(ertrag ~ sorte, data=duenger)

      Kruskal-Wallis rank sum test

data: ertrag by sorte
Kruskal-Wallis chi-squared = 6.2824, df = 2, p-value = 0.04323
```

Der p-Wert ist 0.04, also ähnlich wie bei der Varianzanalyse.

Auch hier stellt sich wieder die Frage, welche Düngersorten sich denn im Ertrag unterscheiden.

5.3 Paarweise Vergleiche

Um die oben gestellte Frage zu beantworten, liegt es nahe, jeweils zwei Düngersorten mit dem Zwei-Stichproben-t-Tests oder dem U-Test miteinander zu vergleichen. Dies kann mit den Funktionen `pairwise.t.test` und `pairwise.wilcox.test` geschehen:

```
> pairwise.t.test(duenger$ertrag, duenger$sorte, p.adjust.method="none")
$method
[1] "t tests with pooled SD"
$data.name
[1] "duenger$ertrag and duenger$sorte"
$p.value
           1          2
2 0.26624561         NA
3 0.08671016 0.01043281
$p.adjust.method
[1] "none"
attr(,"class")
[1] "pairwise.htest"

> pairwise.wilcox.test(duenger$ertrag, duenger$sorte,
  p.adjust.method="none")
$method
[1] "Wilcoxon rank sum test"
$data.name
[1] "duenger$ertrag and duenger$sorte"
$p.value
          1           2
2 0.5476190          NA
3 0.2072998 0.007936508
$p.adjust.method
[1] "none"
    ...
```

Bei beiden Verfahren ergibt sich ein signifikanter Unterschied zwischen Dünger 2 und 3 auf dem 1%-Niveau.

Bei diesem Vorgehen hat man allerdings nicht einen, sondern drei Tests durchgeführt. Dabei tritt das Problem des multiplen Testens auf: unter vielen Tests sind einige falsch-signifikante Resultate aufgrund des Fehlers 1. Art zu erwarten.

Führt man insgesamt m Tests jeweils zum Niveau α durch, so lässt sich zeigen, dass die Wahrscheinlichkeit für einen Fehler 1. Art in mindestens einem der Tests zwischen α und $m\alpha$ liegt.

Aus dieser Abschätzung ergibt sich eine einfache Methode, die sogenannte **Bonferroni-Korrektur**, die garantiert, dass die Wahrscheinlichkeit für einen Fehler 1. Art insgesamt höchsten α ist: alle m Tests werden auf dem Niveau α/m durchgeführt. Für die in R ausgegebenen Testergebnisse bedeutet dies, dass jeder p-Wert einfach mit m multipliziert wird.

```
> pairwise.t.test(duenger$ertrag, duenger$sorte,
  p.adjust.method="bonferroni")
    ...
$p.value
          1          2
2 0.7987368         NA
3 0.2601305 0.03129844
$p.adjust.method
[1] "bonferroni"
    ...

> pairwise.wilcox.test(duenger$ertrag, duenger$sorte,
  p.adjust.method="bonferroni")
    ...
$p.value
          1          2
2 1.0000000         NA
3 0.6218995 0.02380952
$p.adjust.method
[1] "bonferroni"
    ...
```

Die p-Werte sind jetzt dreimal so groß wie ohne Bonferroni-Korrektur (allerdings sinnvollerweise nie größer als 1). Beim parametrischen wie beim nichtparametrischen Verfahren ist der Unterschied zwischen Dünger 2 und 3 nur noch auf dem 5%-Niveau signifikant. Dafür ist gesichert, dass die Wahrscheinlichkeit für einen Fehler 1. Art insgesamt höchstens α ist.

Für spezielle Situationen gibt es oft bessere Verfahren als die Bonferroni-Korrektur. Liegt die Situation der einfachen Varianzanalyse aus Abschnitt 5.1 vor, so sollte man die Tukey-Methode verwenden. Diese besitzt eine höhere Güte; allerdings liefert sie anstelle

von p-Werten simultane Konfidenzintervalle für die Mittelwertsdifferenzen. Wird dabei das Konfidenzniveau $1 - \alpha$ gewählt, so kann die Hypothese $\mu_j = \mu_k$ auf dem Niveau α verworfen werden, falls das Konfidenzintervall für die Differenz $\mu_j - \mu_k$ die Null **nicht** enthält.

```
> duenger.aov = aov(ertrag ~ sorte, data=duenger)
> TukeyHSD(duenger.aov, conf.level=0.95)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = ertrag ~ sorte, data = duenger)

$sorte
    diff        lwr       upr
2-1 -0.5 -1.6439713 0.6439713
3-1  0.8 -0.3439713 1.9439713
3-2  1.3  0.1560287 2.4439713

> plot(TukeyHSD(duenger.aov))
```

Hier ist $\bar{x}_{2+} - \bar{x}_{1+} = -0.5$, $\bar{x}_{3+} - \bar{x}_{1+} = 0.8$ und $\bar{x}_{3+} - \bar{x}_{2+} = 1.3$. Simultane Konfidenzintervalle für $\mu_2 - \mu_1, \mu_3 - \mu_1$ und $\mu_3 - \mu_2$ sind $(-1.6, 0.6), (-0.3, 1.9)$ und $(0.2, 2.4)$. Da nur das dritte Intervall die Null nicht einschließt, besteht nur zwischen Dünger 2 und 3 ein signifikanter Unterschied auf dem 5%-Niveau.

Die mit dem letzten Befehl erzeugte graphische Darstellung der Konfidenzintervalle zeigt Abb. 5.5.

5.4 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 5 nach.
2. Der R-Datensatz **Insectsprays** enthält die Anzahlen von getöteten Insekten (unter einer gewissen Gesamtzahl) bei Einsatz eines Insektizids. Dabei wurden sechs verschiedene Insektizide jeweils zwölfmal verwendet.

Abbildung 5.5: Simultane Konfidenzintervalle nach Tukey zu Beispiel 5.1.1

Insektizid	Anzahl											
A	10	7	20	14	14	12	10	23	17	20	14	13
B	11	17	21	11	16	14	17	17	19	21	7	13
C	0	1	7	2	3	1	2	1	3	0	1	4
D	3	5	12	6	4	3	5	5	5	5	2	4
E	3	5	3	5	3	6	1	1	3	2	6	4
F	11	9	15	22	15	16	13	10	26	26	24	13

Nehmen Sie den Datensatz durch `data(InsectSprays); attach(InsectSprays)` in den Suchpfad des Systems auf.

a) Zeichnen Sie Boxplots der Variablen **count** in Abhängigkeit von der Variablen **spray**.

b) Zeichnen Sie QQ-Plots der Variablen **count** getrennt für die verschiedenen Insektizide. Warum ist es nicht sinnvoll, einen QQ-Plot für alle Werte gemeinsam anzufertigen?

c) Führen Sie eine Varianzanalyse durch; zeichnen Sie einen QQ-Plot der Residuen und einen Plot der Residuen gegen die Mittelwerte.

d) Testen Sie mit dem Kruskal-Wallis-Test auf gleiche Wirkung der Insektizide.

e) Führen Sie paarweise t-Tests (ohne Korrektur) durch. Verwenden Sie dabei einmal die Option `pool.sd=T` (das ist der default-Wert, dabei wird die Stichprobenstandardabweichung aus allen Gruppen zusammen geschätzt und

dieser Wert für alle Tests verwendet). Danach verwenden Sie die Optionen `pool.sd=F, var.equal=F`. Jetzt wird zwischen zwei Gruppen jeweils ein (zweiseitiger) Welch-Test wie in Abschnitt 4.2 durchgeführt. Vergleichen Sie die Ergebnisse.

Hinweis: Die Varianzanalyse setzt gleiche Varianzen in allen Gruppen voraus; unter dieser Annahme sollten sich die Resultate nicht allzu sehr unterscheiden.

f) Wiederholen Sie a) bis e), wobei Sie eine Wurzeltransformation durchführen: statt **count** verwenden Sie die Variable **sqrt(count)**.

g) Führen Sie paarweise t-Tests für die Variable **sqrt(count)** mit Bonferroni-Korrektur durch. Bestimmen Sie simultane Konfidenzintervalle mit der Tukey-Methode; welche Mittelwerte unterscheiden sich auf dem 5%-Niveau signifikant? Plotten Sie die simultanen Konfidenzintervalle.

3. Lesen Sie den in der Datei `c:\daten\noten.csv` gespeicherten Datensatz durch

```
noten = read.csv2("c:/daten/noten.csv")
```

ein. Er enthält die Abiturnote (Merkmal `Abinote`), die Vordiplomnote (Merkmal `Vordiplom`) und das Studienfach (Merkmal `Fach`) von Studierenden verschiedener Mathematikstudiengänge.

a) Zeichnen Sie Boxplots der Noten getrennt nach Fach.

b) Testen Sie, ob sich die Vordiplomnoten in den verschiedenen Studienrichtungen signifikant unterscheiden. Welcher Test ist hier geeignet?

c) Plotten Sie die Vordiplomnote in Abhängigkeit von der Abiturnote (beachten Sie dabei den Stichprobenumfang!). Verwenden Sie danach anstelle der Funktion `plot()` die Funktion `sunflowerplot()`.

d) Definieren Sie neue Merkmale `Abinote2 = jitter(noten$Abinote)` und `Vordiplom2 = jitter(noten$Vordiplom)`, und zeichnen Sie ein Streudiagramm dieser beiden Merkmale. Vergleichen Sie die Plots in c) und d); was wird jeweils dargestellt?

6 Unabhängigkeits- und Korrelations-Tests

Es liegen Datenpaare $(x_1, y_1), ..., (x_n, y_n)$ vor, dabei ist

x_i die Beobachtung von Merkmal 1 an Objekt Nr. i,
y_i die Beobachtung von Merkmal 2 an Objekt Nr. i

Frage: Besteht ein statistischer Zusammenhang zwischen Merkmal 1 und Merkmal 2?

Mathematisches Modell:

(x_i, y_i) sind unabhängige Realisierungen eines zweidimensionalen Zufallsvektors (X, Y) mit unbekannter Verteilung.

6.1 Unabhängigkeitstest bei zweidimensionaler Normalverteilung

Annahme: $(X, Y) \sim N(\mu, \nu; \sigma^2, \tau^2, \rho)$

Vorsicht: Diese Annahme ist häufig zweifelhaft! Das Testverfahren ist nicht sehr robust gegen Abweichungen von dieser Annahme!

Sei $\rho = \rho(X, Y)$ der Korrelationskoeffizient von X und Y (unter NV gilt: X und Y unabhängig $\Leftrightarrow$ $\rho = 0$!); weiter sei

$$r_{xy} = \frac{\frac{1}{n-1}\sum_{j=1}^{n}(x_j - \bar{x})(y_j - \bar{y})}{s_x \cdot s_y}$$

der empirische Korrelationskoeffizient von $(x_j, y_j)_{1 \le j \le n}$. r_{xy} ist ein Schätzwert für den Korrelationskoeffizient der zugrundeliegenden Normalverteilung.

Beim **zweiseitigen Test**

$$H_0 : \rho = 0 \quad (\Leftrightarrow (X, Y) \text{ unabhängig})$$

gegen $H_1 : \rho \neq 0$ $(\Leftrightarrow (X, Y)$ abhängig)

verwendet man die **Prüfgröße**

$$T \;=\; \sqrt{n-2}\cdot\frac{r_{xy}}{\sqrt{1-r_{xy}^2}}\,.$$

Testentscheid:

H_0 ablehnen, falls $|T| \geq t_{n-2,1-\alpha/2}$.

Kein Widerspruch zu H_0, falls $|T| < t_{n-2,1-\alpha/2}$.

Der **einseitige Test**

$$\begin{aligned} H_0: &\quad \rho = 0 \quad (\text{bzw. } \rho \leq 0),\\ H_1: &\quad \rho > 0 \quad (\text{d.h. } X \text{ und } Y \text{ positiv korreliert}) \end{aligned}$$

verwendet die gleiche Prüfgröße wie der zweiseitige Test.

Testentscheid:

H_0 ablehnen, falls $T \geq t_{n-2,1-\alpha}$.

Kein Widerspruch zu H_0, falls $T < t_{n-2,1-\alpha}$.

6.1.1 Beispiel

Der Datensatz `wuermer.csv` im Verzeichnis `c:\daten` enthält Daten über Schwermetallbelastungen von Würmern, die als Parasiten in Fischen vorkommen. Tabelle 6.1 zeigt einen Teil des Datensatzes so, wie er in einem Texteditor angezeigt wird.

Die erste Zeile enthält die Namen der einzelnen Merkmale; danach folgen die eigentlichen Daten, jeweils durch Strichpunkt getrennt. Zur besseren Übersicht kann man den Datensatz auch direkt in einer Tabellenkalkulation wie Microsoft Excel öffnen.

Mit dem Befehl `read.csv2` kann man csv-Dateien als Data Frame in R einlesen, wobei die Character-Variable `Gruppe` automatisch als Faktor gespeichert wird:

```
> wuermer = read.csv2("c:/daten/wuermer.csv")
```

Achtung: In der Pfadangabe ist statt `\` das Zeichen `/` zu verwenden, da `\` ein Sonderzeichen in R ist! Alternativ kann auch `\\` verwendet werden, also

```
> wuermer = read.csv("c:\\daten\\wuermer.csv")
```

Befinden sich die Daten in einem anderen Verzeichnis, etwa in

```
"Gruppe";"Anzahl";"Masse";"Ind.Masse";"Pb";"Cd"
"A";11;6,1;0,6;10;7
"A";24;13,3;0,6;2;6
"A";14;4,2;0,3;6;5
"A";14;5,8;0,4;7;4
"A";9;4,8;0,5;9;9
"A";9;6,7;0,7;7;8
"A";19;13,3;0,7;20;12
"A";11;6,5;0,6;3;11
"A";5;3,2;0,6;18;13
"A";5;6,1;1,2;3;4
"B";6;7,8;1,3;20;40
"B";11;4,4;0,4;4;2
"B";4;3,6;0,9;9;13
"B";3;8,3;2,8;28;59
```

Tabelle 6.1: Datensatz `wuermer.csv`

`c:\programme\r\daten\wuermer.csv`

so muss der Befehl zum Einlesen der Daten entsprechend in

```
> wuermer = read.csv("c:/programme/r/daten/wuermer.csv")
```

geändert werden.

Unter anderem sollte untersucht werden, ob ein Zusammenhang zwischen dem Gewicht der Würmer (Variable `Ind.Masse`) und der Bleianreicherung der Würmer (Variable `Pb`) besteht. Die Pearson-Korrelation kann mit der Funktion `cor` berechnet werden:

```
> summary(wuermer); attach(wuermer)
    ...
> cor(Ind.Masse,Pb)
[1] 0.5974002
```

Das Ergebnis des Pearson-Unabhängigkeitstests, zusammen mit einem Konfidenzintervall für den Korrelationskoeffizienten, liefert die Funktion `cor.test`. Dabei könnte man hier den Test auch einseitig durchführen (Option `alternative="greater"`), da eine negative Korrelation auszuschließen ist.

```
> cor.test(Ind.Masse, Pb, method="pearson")

        Pearson's product-moment correlation

data:  Ind.Masse and Pb
t = 4.3437, df = 34, p-value = 0.0001197
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3345198 0.7740210
sample estimates:
      cor
0.5974002
```

6.2 Unabhängigkeitstest bei stetigen und ordinalen Merkmalen

Hier wird die Hypothese

$$H_0: \quad X \text{ und } Y \text{ sind unabhängig}$$

gegen die Alternative

$$H_1: \quad X \text{ und } Y \text{ sind „rang-korreliert“}$$

getestet. Dabei wird keine Normalverteilungsannahme gemacht.

Prüfgröße: Hotelling-Pabst-Statistik

$$T = \sum_{j=1}^{n} (U_j - V_j)^2.$$

Dabei ist

$$\begin{aligned} U_j: &\quad \text{(Durchschnitts-) Rang von } x_j \text{ unter } x_1, \ldots, x_n \\ V_j: &\quad \text{(Durchschnitts-) Rang von } y_j \text{ unter } y_1, \ldots, y_n. \end{aligned}$$

Bemerkung: Treten keine Bindungen auf, so gilt $T = \sum_{j=1}^{n} (j - R_j)^2$, wobei R_j der Rang des zum j-kleinsten x gehörenden y-Wertes ist. Die Testgröße entspricht bis auf eine lineare Transformation dem Rang-Korrelationskoeffizient nach Spearman

$$\rho_{xy} \;=\; 1 - \frac{6}{n(n^2-1)} \sum_{j=1}^{n} (j - R_j)^2 \; .$$

6.2.1 Beispiel

Den Rang-Korrelationskoeffizient liefert wieder die Funktion `cor`, wenn sie mit der Option `method="spearman"` aufgerufen wird. Der Spearman-Unabhängigkeitstest kann wieder mit der Funktion cor.test durchgeführt werden:

```
> cor(Ind.Masse, Pb, method="spearman")
[1] 0.7011816
> cor.test(Ind.Masse, Pb, method="spearman")

        Spearman's rank correlation rho

data:  Ind.Masse and Pb
S = 2322, p-value = 3.987e-06
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.7011816
    ...
```

Fazit: Beide Tests ergeben eine hochsignifikante positive Korrelation zwischen Gewicht und Bleianreicherung der Würmer. Dies zeigt auch der Scatterplot in Abb. 6.1, erzeugt durch

```
> plot(Pb ~ Ind.Masse, pch=16)
> text(.75, 120, "rho = 0.70")
```

Unterteilt man die Stichprobe nach dem weiteren Merkmal Gruppe (Fanggebiet der Fische), so erhält man allerdings geringere Korrelationen. Durch den Aufruf

```
> cor.test(Ind.Masse[Gruppe=="A"], Pb[Gruppe=="A"], method="spearman")
```

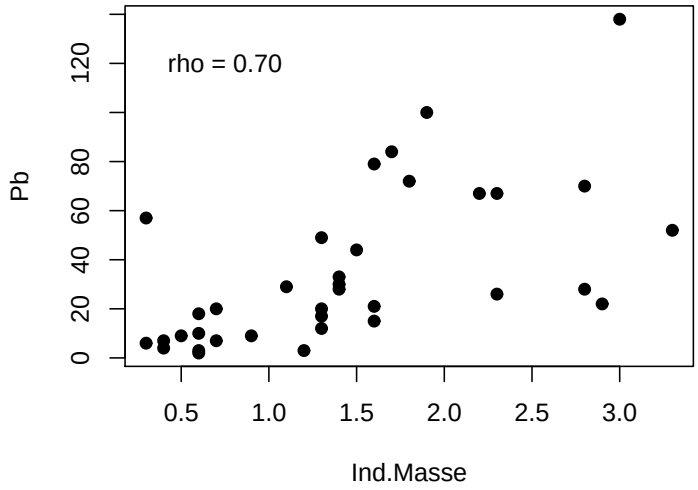


Abbildung 6.1: Bleianreicherung gegen Gewicht der Würmer

führt man den Test innerhalb der ersten Gruppe durch; der Rangkorrelationskoeffizient ρ ist 0.04! Ähnliche Ergebnisse ergeben sich in den anderen Gruppen. Graphisch kann man dies durch die Funktion `coplot` gut veranschaulichen.

```
coplot(Pb ~ Ind.Masse | Gruppe, pch=16)
text( grconvertX(0.1,"npc"), grconvertY(0.2, "npc"), "rho = 0.04", cex=0.8)
text( grconvertX(0.6,"npc"), grconvertY(0.2, "npc"), "rho = 0.55", cex=0.8)
text( grconvertX(0.1,"npc"), grconvertY(0.6, "npc"), "rho = 0.10", cex=0.8)
text( grconvertX(0.6,"npc"), grconvertY(0.6, "npc"), "rho = 0.60", cex=0.8)
```

Abb. 6.2 zeigt, dass die beiden untersuchten Größen innerhalb eines Fanggebietes eine geringere Korrelation aufweisen; außerdem sehen die Scatterplots sehr unterschiedlich aus. Für weitergehende Aussagen müsste also der Einfluss des Fanggebietes genauer untersucht werden.

Bemerkung: In Abb. 6.1 wird ein Koordinatensystem benutzt, das in x-Richtung von 0 bis etwa 3.5 und in y-Richtung von 0 bis 140 reicht. Mit `text(.75, 120, "rho = 0.70")` wird folglich Text in der oberen linken Ecke plaziert.

Bei der Funktion `coplot` ist unklar, welches Koordinatensystem benutzt wird; Text könnte durch Probieren plaziert werden. In Abb. 6.2 werden stattdessen die Funktionen `grconvertX` und `grconvertY` verwendet, die dafür sorgen, dass man die Position als

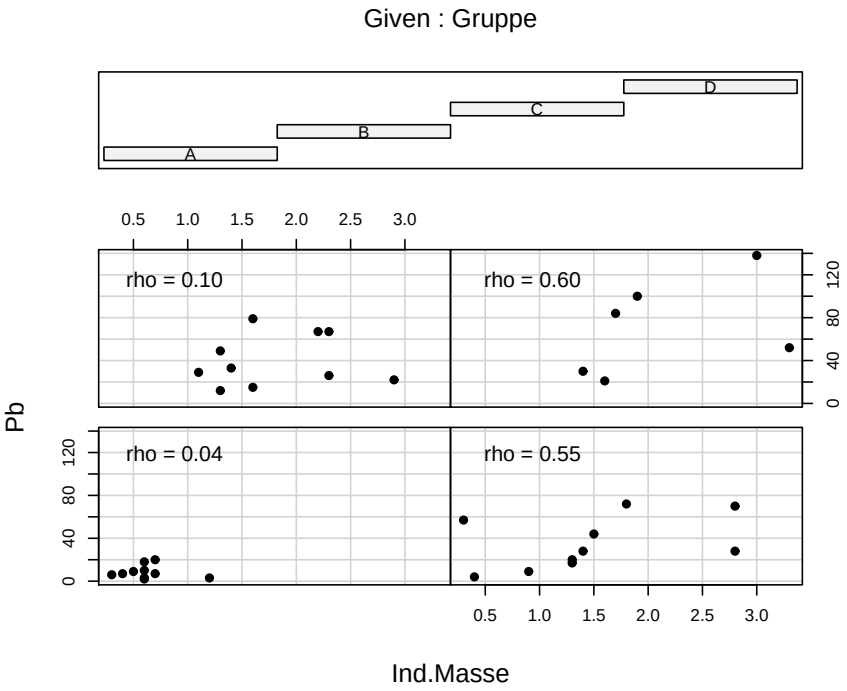


Abbildung 6.2: Bleianreicherung gegen Gewicht der Würmer innerhalb der Fanggebiete

Bruchteil zwischen 0 und 1 angeben kann. Der Text `rho = 0.04` wird also so plaziert, dass darunter 20% und links davon 10% des gesamten Plots liegen.

Bei der Interpretation von Korrelationen treten häufig Probleme auf. Dabei lautet die wichtigste Regel:

Korrelation bedeutet **nicht** Kausalzusammenhang!

Verschiedene Umstände können zu hohen Korrelationen führen, ohne dass ein Kausalzusammenhang besteht. Häufig liegt eine der folgenden Ursachen vor:

1. Beide Messgrößen können von einer dritten Variablen abhängen, z.B.

   Lesefähigkeit $\leftrightarrow$ Schuhgröße von Kindern (Alter)

   Zahl der Ärzte $\leftrightarrow$ Zahl der Unfälle (wachsende Population)

2. Vermischung zweier oder mehrerer Populationen: Diese Situation zeigt Abb. 6.3; sie liegt auch in Beispiel 6.2.1 vor.

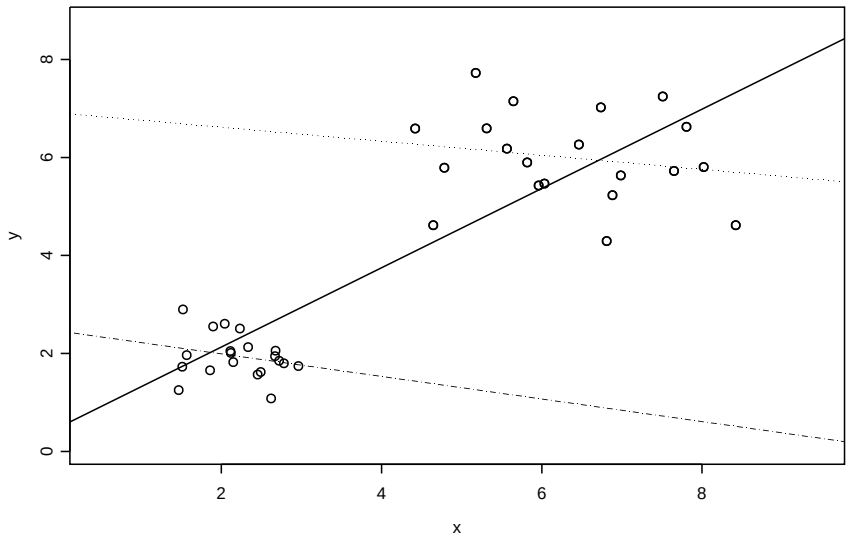


Abbildung 6.3: Scheinkorrelation durch Vermischung zweier Populationen

3. Verwendung von Quotienten: Sind $x_1, \ldots, x_n$, $y_1, \ldots, y_n$, $z_1, \ldots, z_n$ Realisierungen von unabhängigen Zufallsvariablen $X_1, \ldots, X_n$, $Y_1, \ldots, Y_n$,$Z_1, \ldots, Z_n$, so sind $x_1/z_1, \ldots, x_n/z_n$ und $y_1/z_1, \ldots, y_n/z_n$ positiv korreliert, wie man in Abb. 6.4 sehen kann.

6.3 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 6 nach.
2. Untersuchen Sie, ob ein Zusammenhang zwischen dem Gewicht der Würmer und der Cadmiumanreicherung der Würmer (Variable `Cd`) besteht.
3. Erstellen Sie mit einem beliebigen Editor eine Text-Datei, die die Gummiband-Daten aus Abschnitt 1.2 enthält. Lesen Sie diese Datei mit dem Befehl `read.table` in R ein. Untersuchen Sie, ob die Merkmale `Dehnung` und `Weite` korreliert sind und zeichnen Sie einen Scatterplot.
4. Die Funktion `mvrnorm` in der MASS-library erzeugt Zufallszahlen aus einer mehrdimensionalen Normalverteilung. 500 Zufallszahlen aus $N(\mu, \nu; \sigma^2, \tau^2, \rho)$ mit $\mu = \nu = 0, \sigma = \tau = 1$ und $\rho = 0.5$ erhält man durch

```
> library(MASS)
> mu=0; nu=0; sigma=1; tau=1; rho=0.5
```

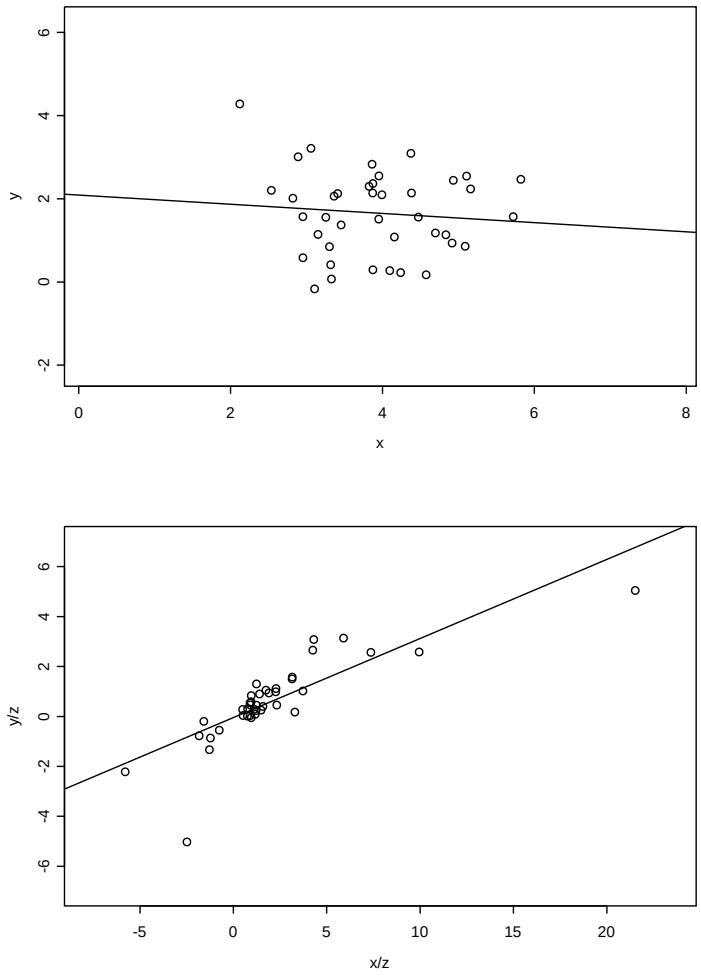


Abbildung 6.4: Scheinkorrelation durch Quotientenbildung

```
> S = matrix(c(sigma^2,sigma*tau*rho,sigma*tau*rho,tau^2),2,2)
> S
      [,1] [,2]
 [1,]  1.0  0.5
 [2,]  0.5  1.0
> z = mvrnorm(n=500, c(mu,nu), S)
```

Plotten kann diese zweidimensionale Punktwolke etwa durch

```
> plot(z, xlim=c(-4,4), ylim=c(-4,4))
> abline(h=0); abline(v=0)
```

Der Pearson-Korrelationskoeffizienten zwischen den x- und den y-Werten läßt sich durch den Befehl `cor(z[,1],z[,2])` berechnen.

	Land	Flaeche (km^2)	Stoerche (Paare)	Menschen (10^6)	Geburtenrate (10^3/Jahr)
1	Albanien	28750	100	3.2	83
2	Belgien	30520	1	9.9	87
3	Bulgarien	111000	5000	9.0	117
4	Dänemark	43100	9	5.1	59
5	Deutschland	357000	3300	78	901
6	Frankreich	544000	140	56	774
7	Griechenland	132000	2500	10	106
8	Holland	41900	4	15	188
9	Italien	301280	5	57	551
10	Österreich	83860	300	7.6	87
11	Polen	312680	30000	38	610
12	Portugal	92390	1500	10	120
13	Rumänien	237500	5000	23	23
14	Spanien	504750	8000	39	439
15	Schweiz	41290	150	6.7	82
16	Türkei	779450	25000	56	1576
17	Ungarn	93000	5000	11	124

Tabelle 6.2: Daten von 1990 für 17 europäische Länder

Plotten Sie entsprechende Punktwolken mit $\rho = 0, \rho = 0.3, \rho = 0.9$ und $\rho = -0.7$ zusammen auf eine Graphikseite. Berechnen Sie jeweils den Pearson-Korrelationskoeffizienten. Wiederholen Sie dies für $n = 20$ und $n = 50$.

5. Tabelle 6.2 ist einem Artikel aus der Zeitschrift *Stochastik in der Schule* entnommen[1]. In dem Artikel wird der Zusammenhang zwischen der absoluten Anzahl in einem Land pro Jahr geborener Menschen (Merkmal `Geburtenrate`) und der absoluten Anzahl der in einem Land lebenden Storchenpaare (Merkmal `Stoerche`) mit Hilfe des Pearson-Korrelationskoeffizienten untersucht. (Die Daten finden Sie auch als Dataframe **storch.csv** im gleichen Verzeichnis wie den Dataframe wuermer.csv.)

   a) Zeichnen Sie einen Scatterplot der Geburtenrate gegen die Anzahl der Störche. Berechnen Sie die Pearson-Korrelation zwischen diesen beiden Merkmalen.

[1] Stochastik in der Schule 21 (2001), S.21-23; siehe auch Stochastik in der Schule 22 (2002), S.29-34

Führen Sie einen Test auf Unabhängigkeit der beiden Merkmale durch. Wie lassen sich die Ergebnisse erklären?

b) Berechnen Sie die Rang-Korrelation zwischen Geburtenrate und der Zahl der Störche, und führen Sie den entsprechenden Unabhängigkeitstest durch.

c) Wiederholen Sie a) und b) für die Variablen `Geburtenrate/Menschen` und `Stoerche/Flaeche`.

d) Was fällt ihnen am Verhältnis `Geburtenrate/Menschen` auf?

6. Lesen Sie den in der Datei `c:\daten\worldfactbook.csv` gespeicherten Datensatz durch

```
states1 = read.csv2("c:/daten/worldfactbook.csv")
```

ein. Er enthält Daten von 225 Staaten[2], nämlich die Merkmale `Country`, `Area` (in km^2), `Birthrate` (Geburten pro tausend Einwohner), `Lifeexpectancy`, `Population` und `Density` (Bevölkerungsdichte).

a) Plotten Sie die Bevölkerungsdichte und identifizieren Sie Ausreißer durch

```
plot(states1$Density)
identify(states1$Density, labels = states1$Country)
```

Nach Ausführung des zweiten Befehls klicken Sie dazu im Plot in die Nähe eines Datenpunktes. Erstellen Sie durch

```
states = states1[states1$Density < 2000, ]
```

einen neuen Datensatz, der Staaten mit extremer Bevölkerungsdichte ausschließt.

b) Untersuchen Sie, ob die Merkmale Bevölkerung und Bevölkerungsdichte korreliert sind.

c) Untersuchen Sie, welche Abhängigkeiten zwischen den Merkmalen `Birthrate`, `Lifeexpectancy` und `Density` bestehen.

[2]Quelle: CIA World Factbook

7 Regressionsmodelle für stetige Zielgrößen

Zweck: Untersuchung der Abhängigkeit einer Zielgröße Y von Einflussgrößen (Regressoren).

Allgemeines parametrisches Regressionsmodell:

$$Y \;=\; g(x, z, u, \ldots; a, b, c, \ldots) \;+\; \varepsilon, \tag{7.1}$$

wobei g : eine gegebene Funktion von bekannten Einflussgrößen x, z, $u, \ldots$ und unbekannten Parametern a, b, $c, \ldots$ ist; die Parameter $a, b, c, \ldots$ müssen geschätzt werden.

Der Zufallsfehler ε ist $N(0, \sigma^2)$-verteilt. Die Varianz σ^2 ist unbekannt. Somit gilt

$$Y \;\sim\; N(g(x, z, u, \ldots; a, b, c, \ldots),\, \sigma^2).$$

n unabhängige Versuche werden durchgeführt; die Versuchsbedingungen beim j-ten Versuch sind $x_j, z_j, u_j, \ldots$; ϵ_j ist die Realisierung von ε beim j-ten Versuch.

Man beobachtet also Realisierungen

$$y_j \;=\; g(x_j, z_j, u_j, \ldots; a, b, c, \ldots) \;+\; \epsilon_j$$

von Y.

Spezialfälle von (7.1) sind:

Einfache lineare Regression

$$Y \;=\; a + b \cdot x \;+\; \varepsilon$$

Linearer Zusammenhang zwischen x und Y bis auf Zufallsfehler!

Quadratische Regression

$$Y \;=\; a + b \cdot x \;+\; c \cdot x^2 \;+\; \varepsilon$$

Quadratischer Zusammenhang zwischen x und Y bis auf Zufallsfehler!

Polynomiale Regression

$$Y \;=\; a_0 \,+\, a_1 \cdot x \,+\, a_2 \cdot x^2 \,+\, \ldots \,+\, a_k \cdot x^k \,+\, \varepsilon$$

Polynomiale Regression k-ten Grades: g ist Polynom k-ten Grades der Einflussgröße x.

Multiple lineare Regression (mehrere Einflussgrößen):

$$Y \;=\; a \,+\, b \cdot x^{(1)} \,+\, c \cdot x^{(2)} \,+\, \ldots \,+\, \varepsilon$$

a, b, c sind unbekannte, zu schätzende Parameter.

7.1 Einfache lineare Regression

Das Modell ist

$$Y_j \;=\; a \,+\, b \cdot x_j \,+\, \varepsilon_j \qquad (j = 1, ..., n),$$

die Fehler $\varepsilon_j \quad (j = 1, \ldots, n)$ sind stochastisch unabhängig mit Verteilung $N(0, \sigma^2)$.

$x_1, ..., x_n$ sind bekannt; sie müssen nicht alle verschieden sein! Dagegen sind a, b und σ^2 unbekannt. Somit gilt

$$Y_j \;\sim\; N(a + bx_j\,, \sigma^2).$$

Im j-ten Experiment wird die Realisierung y_j von Y_j beobachtet; die Gerade $y = a + bx$ heißt Regressionsgerade.

Die **Schätzung von a und b** erfolgt nach der Methode der kleinsten Quadrate, also durch Minimieren von

$$\sum_{j=1}^{n} (y_j - a - b \cdot x_j)^2$$

bezüglich a und b. Als Lösung ergibt sich

$$\hat{b} \;=\; \frac{\sum_{j=1}^{n}(x_j - \bar{x})(y_j - \bar{y})}{\sum_{j=1}^{n}(x_j - \bar{x})^2} \;=\; r_{xy} \cdot \frac{s_y}{s_x}\,,$$

$$\hat{a} \;=\; \bar{y} - \hat{b} \cdot \bar{x}.$$

Ein Schätzwert für die Restvarianz σ^2 ist

$$\hat{\sigma}^2 \;=\; \frac{1}{n-2}\cdot\sum_{j=1}^{n}\Big(y_j-\hat{a}-\hat{b}\cdot x_j\Big)^2 \;=\; \frac{n-1}{n-2}\cdot s_y^2\,\left(1-r_{xy}^2\right).$$

7.1.1 Beispiel

Untersuchung der Zielgröße Roggenertrag in Abhängigkeit von der Düngermenge:

x_j : (Düngermenge in kg/ha):	80	90	110	90	110	130	90	110	130
y_j : (Roggenertrag in dt/ha):	38	45	50	45	51	53	50	55	61

```
> duenger = c(80,90,110,90,110,130,90,110,130)
> ertrag = c(38,45,50,45,51,53,50,55,61)
> ertrag.lm = lm(ertrag ~ duenger)
> summary(ertrag.lm)

Call:
lm(formula = ertrag ~ duenger)

Residuals:
   Min       1Q     Median     3Q      Max
-4.8559  -1.5339  -0.2119  3.1441  4.7881

Coefficients:
             Estimate  Std. Error  t value  Pr(>|t|)
(Intercept)  16.76271    7.30860    2.294   0.05552   .
  duenger     0.31610    0.06906    4.577   0.00255   **
---
Signif. codes:  0  ***  0.001  **  0.01  *  0.05  .  0.1    1

Residual standard error: 3.536 on 7 degrees of freedom
Multiple R-Squared: 0.7496, Adjusted R-squared: 0.7138
F-statistic: 20.95 on 1 and 7 DF, p-value: 0.002552
```

Unter der Annahme eines linearen Regressionsmodells folgt also aus der Koeffizienten-Tabelle

$$\hat{b}=0.32,\quad \hat{a}=16.76.$$

Somit ist $y=16.76+0.32\cdot x$ die geschätzte Regressionsgerade. Die geschätzte Standardabweichung ist $\hat{\sigma}=3.54$.

Eine wichtige Größe ist die Quadratsumme der Fehler (Sum of Squares Error)

$$SS_E \;=\; \sum_{j=1}^{n} R_j^2 \;=\; \sum_{j=1}^{n} \Big(y_j - \hat{a} - \hat{b}\cdot x_j\Big)^2,$$

wobei $R_j = y_j - (\hat{a} + \hat{b}\cdot x_j)$ die sogenannten Residuen sind. Mit der Quadratsumme von Y

$$SS_Y \;=\; \sum_{i=1}^{n} (y_i - \bar{y})^2$$

wird dann das Bestimmtheitsmaß

$$R^2 \;=\; \frac{SS_Y - SS_E}{SS_Y}$$

definiert. Falls dieser Wert nahe bei eins liegt, so ist die Vorhersage $\hat{y} = \hat{a} + \hat{b}\cdot x$ für y gut; außerdem gilt der Zusammenhang

$$r_{xy}^2 \;=\; R^2$$

zwischen dem empirischen Korrelationskoeffizienten von $(x_i, y_i)_i$ und R^2. Im Beispiel ist also $r_{xy}^2 = R^2 = 0.75$.

Das adjustierte Bestimmtheitsmaß R_a^2 ist definiert als $R_a^2 = 1 - \frac{n-1}{n-p}(1 - R^2)$, wobei p die Zahl der Parameter ist, hier also $p = 2$.

Die letzte Spalte der Tabelle enthält p-Werte für t-Tests, die prüfen, ob der entsprechende Koeffizient signifikant von Null verschieden ist. Anstelle von Konfidenzintervallen für die Koeffizienten gibt die Spalte `Std. Error` Standardabweichungen der Schätzer der Koeffizienten an.

Aus der 2-σ-Regel folgt, dass

$$Estimate \pm 2 \cdot Std.Error$$

approximativ ein 95%-Konfidenzintervall ist. Also ist etwa $\hat{b} \pm 2 \cdot 0.069 = [0.18, 0.45]$ approximativ ein 95%-Konfidenzintervall für b.

Einen Scatterplot mit Regressionsgeraden (Abb. 7.1) erhält man wieder durch

```
> plot(ertrag ~ duenger, pch=16, xlab="Düngermenge in kg/ha",
  ylab="Roggenertrag in dt/ha")
> abline(ertrag.lm)
```

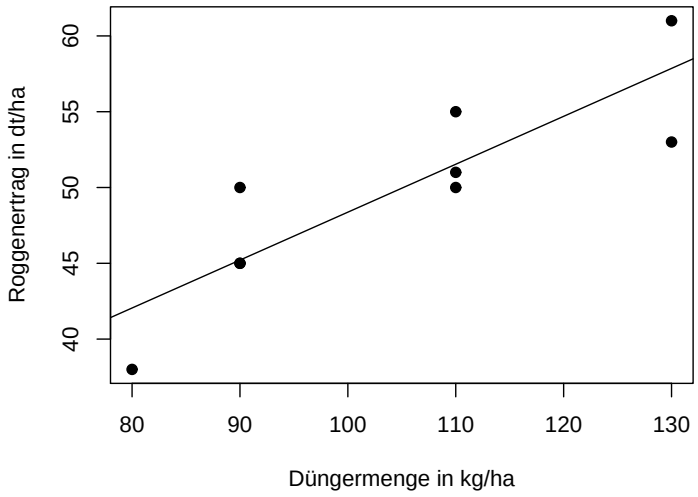


Abbildung 7.1: Scatterplot zu Beispiel 7.1.1

Die Auswertung des Regressionsmodells an neuen Stellen (zusammen mit punktweisen Konfidenzintervallen) erfolgt durch

```
> new = data.frame(duenger = seq(80,140,20))
> predict(ertrag.lm, new, interval = "confidence", level=0.95)
```

	fit	lwr	upr
1	42.05085	37.18217	46.91953
2	48.37288	45.49253	51.25323
3	54.69492	50.92365	58.46618
4	61.01695	54.57629	67.45761

7.2 Residuen-Analyse

Die oben durchgeführten Tests sind nur gültig unter der Modellannahme, dass die Fehler ε_j $(j = 1, \ldots, n)$ stochastisch unabhängig sind mit Verteilung $N(0, \sigma^2)$, so dass im Modell $Y_j \sim N(a + bx_j, \sigma^2)$ gilt. Diese Annahme kann man aufspalten in

(i) Die ε_j sind unabhängig.

(ii) $E(\varepsilon_j) = 0$.

(iii) Sie haben alle die gleiche Varianz $V(\varepsilon_j) = \sigma^2$.

(iv) Sie sind normalverteilt.

Das Überprüfen dieser Voraussetzungen ist wesentlich für eine Regressionsanalyse!

Problem: Der Fehler ε_j ist unbekannt; man ersetzt ihn durch die Residuen $R_j = Y_j - (\hat{a} + \hat{b} \cdot x_j)$!

Zuerst sollte man sich die Kennzahlen der Residuen anschauen (in Beispiel 7.1.1 in der Tabelle "Residuals").

Annahme (iv) kann man mit einem QQ-Plot der Residuen überprüfen: Unter (i) bis (iv) sind die R_j wieder normalverteilt. Dabei treten allerdings Probleme auf:

- Die Varianzen sind nicht mehr alle gleich (aber oft annähernd gleich); evtl. standardisieren
- Residuen sind nicht unabhängig ($\Rightarrow$ Vorsicht bei Interpretation)

Achtung: Ein QQ-Plot für die Y_j ist sinnlos, da sie verschiedene Erwartungswerte haben!

Annahme (iii) kann mit einem „Tukey-Anscombe plot" überprüft werden: dies ist ein Streudiagramm der Residuen R_j gegen die angepaßten Werte $\hat{y}_j = \hat{a} + \hat{b} \cdot x_j$.

Der Tukey-Anscombe-Plot ist gut zum Erkennen der häufigsten Abweichung von (iii): eine zunehmende Streuung mit zunehmendem Wert der Zielgröße.

Achtung: Kein Diagramm Residuen gegen Y_j verwenden!

Was tun, falls obige Annahmen zweifelhaft? Oft hilft eine Datentransformation, oder man muss andere Regressionsmodelle verwenden.

7.2.1 Beispiel
Ozonkonzentration in New York

Der R-Datensatz **airquality** enthält Messungen der Luftqualität in New York, vom Mai bis September 1973. Im folgenden soll die Ozonkonzentration in Abhängigkeit von der Temperatur durch eine einfache lineare Regression modelliert werden.

```
> data(airquality); attach(airquality)
> help(airquality)
> summary(airquality)
  . . .
> length(Temp)   # nach R-Hilfe: 154
```

```
[1] 153
> length(Ozone[!is.na(Ozone)])
[1] 116   # es sind nur 116 Ozon-Messungen vorhanden
> plot(Temp, Ozone, pch=16)
> air.lm = lm(Ozone ~ Temp)
> abline(air.lm)
> summary(air.lm)

Call: lm(formula = Ozone ~ Temp)

Residuals:
    Min        1Q      Median      3Q       Max
 -40.7295  -17.4086  -0.5869  11.3062  118.2705

                    |   Value     Std. Error  t value  Pr(>|t|)
Coefficients:  (Intercept) | -146.9955    18.2872    -8.038   9.37e-13  ***
                    Temp   |   2.4287     0.2331     10.418   < 2e-16   ***

Residual standard error: 23.71 on 114 degrees of freedom
Multiple R-Squared: 0.4877, Adjusted R-squared: 0.4832
F-statistic: 108.5 on 1 and 114 DF, p-value: 0
```

Die geschätzte Regressionsgerade ist also $y = -147 + 2.43 \cdot x$, wobei die Parameter a und b signifikant von Null verschieden sind.

Das Bestimmtheitsmaß R^2 hat den Wert 0.49. Die Daten mit der Regressionsgeraden zeigt Abb. 7.2.

Den QQ-Plot und den Tukey-Anscombe-Plot erhält man durch

```
> qqnorm(residuals(air.lm))
> qqline(residuals(air.lm))
> plot(residuals(air.lm) ~ fitted.values(air.lm))
> abline(h=0, lty=2)
```

Diese beiden Plots erhält man auch durch Eingabe der Befehle `plot(air.lm, which=1)` bzw. `plot(air.lm, which=2)`.

Der QQ-Plot in Abb. 7.3 zeigt starke Abweichungen von einer Normalverteilung; der Tukey-Anscombe-Plot in Abb. 7.4 läßt ein Anwachsen der Streuung der Residuen mit der Temperatur erkennen. Insgesamt scheint das lineare Regressionsmodell nicht gut zu passen.

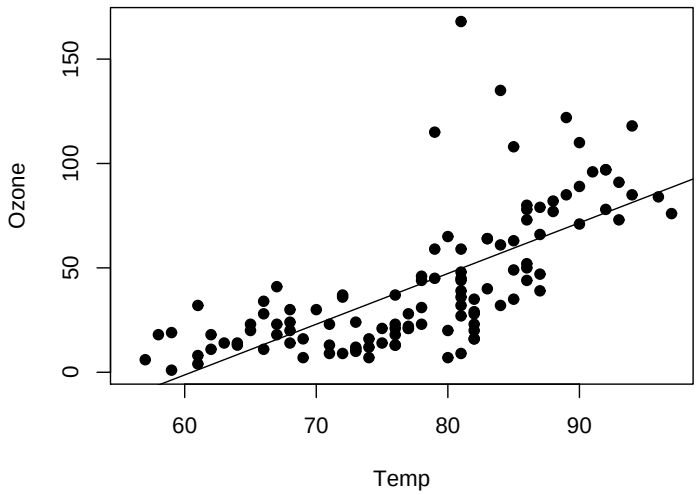


Abbildung 7.2: Ozonkonzentration in Abhängigkeit von der Temperatur mit Regressionsgerade

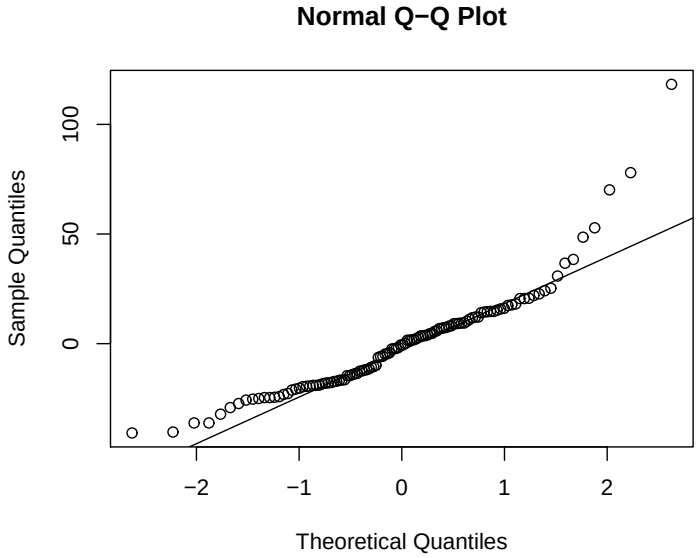


Abbildung 7.3: QQ-Plot zu Beispiel 7.2.1

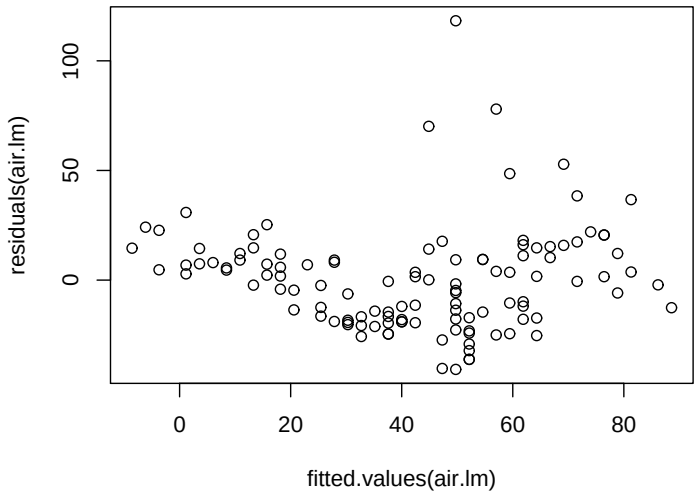


Abbildung 7.4: Tukey-Anscombe-Plot zu Beispiel 7.2.1

7.3 Einfache quadratische Regression

Der Scatterplot in Abb. 7.2 zeigt ein Anwachsen der Ozonkonzentration bei hohen Temperaturen, das stärker als linear ist. Deshalb kann man in Beispiel 7.2.1 als weiteres Modell

$$Y \;=\; a + b \cdot x \;+\; c \cdot x^2 \;+\; \varepsilon,$$

also ein quadratisches Regressionsmodell, ansetzen.

7.3.1 Beispiel

```
> air2.lm = lm(Ozone ~ Temp + I(Temp^2))
> summary(air2.lm)
Call: lm(formula = Ozone ~ Temp + I(Temp^2))

Residuals:
   Min      1Q  Median    3Q     Max
-37.619 -12.513  -2.736  9.676 123.909

Coefficients:
             Estimate  Std. Error  t value  Pr(>|t|)
(Intercept) 305.48577  122.12182    2.501  0.013800   *
Temp         -9.55060    3.20805   -2.977  0.003561  **
I(Temp^2)     0.07807    0.02086    3.743  0.000288 ***
```

```
Residual standard error: 22.47 on 113 degrees of freedom
Multiple R-Squared: 0.5442, Adjusted R-squared: 0.5362
F-statistic: 67.46 on 2 and 113 DF, p-value: 0
```

```
> a = coef(air2.lm)[1]
> b = coef(air2.lm)[2]
> c = coef(air2.lm)[3]
> print(c(a,b,c))
  (Intercept)         Temp    I(Temp^2)
 305.48576778  -9.55060391  0.07806798
```

Beim Aufruf muss man `I(Temp^2)` statt `Temp^2` verwenden (sonst wird hier wieder ein lineares Modell gefittet). Man erhält das Modell

$$Y \;=\; 305.5 - 9.55 \cdot x + 0.078 \cdot x^2 \;+\; \varepsilon.$$

Im Modell sind alle Koeffizienten signifikant von 0 verschieden. Die angepasste Regressionsfunktion kann man folgendermaßen in die Datenpunkte einzeichnen.

```
> plot(Temp,Ozone)
> curve(a+b*x+c*x^2, add=T, col="red")
> qqnorm(residuals(air2.lm)); qqline(residuals(air.lm))
> plot(fitted.values(air2.lm), residuals(air2.lm)); abline(h=0, lty=2)
```

Das quadratische Modell scheint besser zu passen (Abb. 7.5); auch der QQ- und der Tukey-Anscombe-Plot (Abb. 7.6) sehen bis auf einige Ausreißer besser aus. Man beachte aber, dass die Regressionsfunktion zwischen 60 und 65 Grad Fahrenheit ihr Minimum hat; das Modell ist folglich für niedrigere Temperaturen unbrauchbar.

7.4 Multiple lineare Regression

Anhand der **airquality**-Daten soll noch ein Beispiel für ein multiples lineares Regressionsmodell der Form

$$Y \;=\; a \;+\; b \cdot x^{(1)} \;+\; c \cdot x^{(2)} \;+\; d \cdot x^{(3)} \;+\; \varepsilon$$

kurz angesprochen werden. Weitere Einflußgrößen neben der Temperatur sind Sonneneinstrahlung und Windgeschwindigkeit.
Die Residuen sind hier $R_j = y_j - (\hat{a} + \hat{b} \cdot x_j^{(1)} + \hat{c} \cdot x_j^{(2)} + \hat{d} \cdot x_j^{(3)})$, wobei $\hat{a}, \hat{b}, \hat{c}$ und $\hat{d}$ wieder mit der Methode der kleinsten Quadrate geschätzte Parameter sind.

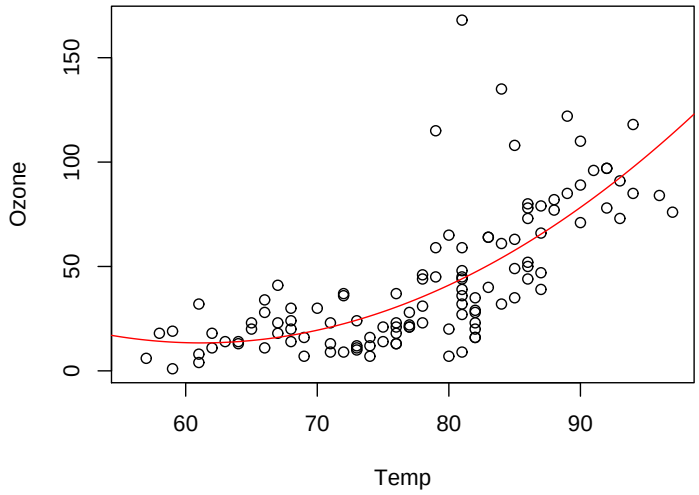


Abbildung 7.5: Ozonkonzentration in Abhängigkeit von der Temperatur mit quadratischer Regressionsfunktion

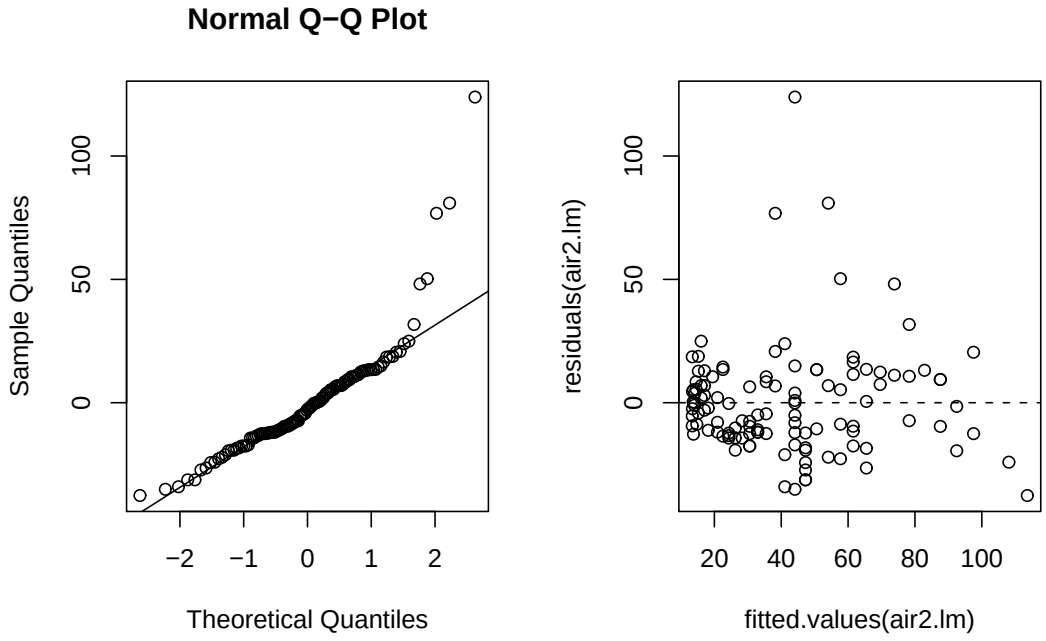


Abbildung 7.6: QQ-Plot und Tukey-Anscombe-Plot zu Beispiel 7.3.1

7.4.1 Beispiel

```
> air3.lm = lm(formula = Ozone ~ Temp + Wind + Solar.R)
> summary(air3.lm)

Call: lm(formula = Ozone ~ Temp + Wind + Solar.R)

Residuals:
   Min       1Q     Median    3Q      Max
-40.485  -14.219  -3.551  10.097  95.619

Coefficients:
             Estimate   Std. Error  t value  Pr(>|t|)
(Intercept) -64.34208   23.05472    -2.791   0.00623   **
    Temp      1.65209    0.25353     6.516   2.42e-09  ***
    Wind     -3.33359    0.65441    -5.094   1.52e-06  ***
  Solar.R     0.05982    0.02319     2.580   0.01124   *

Residual standard error: 21.18 on 107 degrees of freedom
Multiple R-Squared: 0.6059, Adjusted R-squared: 0.5948
F-statistic: 54.83 on 3 and 107 DF, p-value: 0
```

Es ergibt sich das lineare Regressionsmodell

$$Y = -64.3 \; + \; 1.7 \cdot Temp \; - \; 3.3 \cdot Wind \; + \; 0.06 \cdot Solar.R \; + \; \varepsilon \; .$$

Der F-Test überprüft, ob die Gesamtheit aller erklärenden Variablen die Zielgröße überhaupt beeinflußt (letzte Zeile des Outputs). Mit den t-Tests in der Koeffizienten-Tabelle wird wieder die Hypothese $a = 0, b = 0$ usw. getestet.

Das multiple Regressionsmodell kann nicht mehr einfach graphisch dargestellt werden. Einen QQ- bzw. Tukey-Anscombe-Plot erhält man aber wie in den bisherigen Beispielen (Abb. 7.7).

Fehlende Werte

Der Datensatz **airquality** enthält fehlende Werte (missing values), die in R durch `NA` (Not Available) gekennzeichnet sind. Viele Funktionen (z.B. `lm` oder `summary`) lassen die entsprechenden Daten weg, wenn man nichts anderes angibt. Meist kann man durch die Option `na.action=na.fail` den Programmabbruch bei Auftreten von fehlenden Werten erzwingen. Dies ist oft sinnvoll, da man dadurch auf das Fehlen von Werten aufmerksam gemacht wird. Dagegen brechen manche Funktionen bei fehlenden Werten ab:

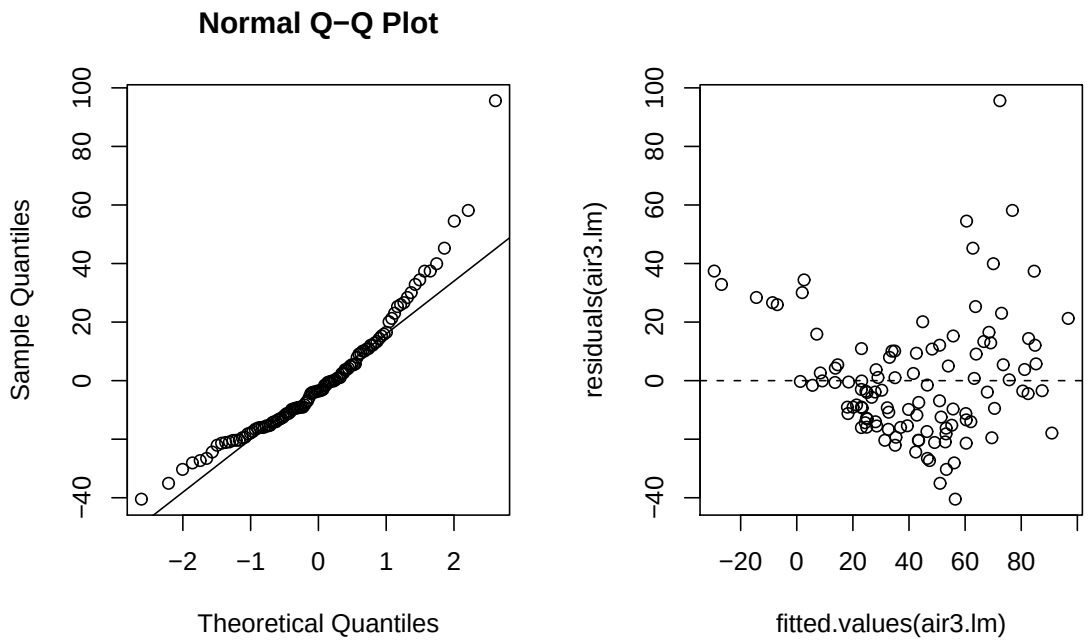


Abbildung 7.7: QQ-Plot und Tukey-Anscombe-Plot zu Beispiel 7.4

```
> mean(Ozone)
[1] NA
```

Hier kann man umgekehrt durch

```
> mean(Ozone, na.rm=T)
[1] 42.12931
```

erzwingen, dass der Mittelwert der vorhandenen Werte berechnet wird.

Um aus dem Datensatz **airquality** von vornherein nur diejenigen Zeilen auszuwählen, in denen alle Werte vorhanden sind, kann man mit

```
> airquality2 = na.omit(airquality)
```

einen neuen Datensatz erstellen, der nur die vollständigen Zeilen enthält. Denselben Zweck erfüllen

```
> airquality2 = subset(airquality,complete.cases(airquality))
```

bzw.

```
> airquality2 = airquality[complete.cases(airquality),]
```

7.5 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 7 nach.
2. a) Passen Sie ein einfaches lineares Regressionsmodell an y_1 in Abhängigkeit von x an, wobei

$$x = (0.1,\ 0.2,\ 0.3,\ 0.4,\ 0.5,\ 0.6,\ 0.7,\ 0.8,\ 0.9,\ 1.0),$$
$$y_1 = (1.40,\ 1.49,\ 1.43,\ 1.59,\ 1.94,\ 2.00,\ 1.97,\ 2.71,\ 2.55,\ 3.06).$$

   Bestimmen Sie die geschätzte Restvarianz $\hat{\sigma}^2$ und das Bestimmtheitsmaß R^2. Zeichen Sie die zugehörige Punktwolke mit Regressionsgeraden und einen Tukey-Anscombe-Plot der Residuen.

   b) Wiederholen Sie Teil a) mit

$$y_2 = c(1.66,\ 1.66,\ 1.50,\ 1.56,\ 1.81,\ 1.77,\ 1.64,\ 2.28,\ 2.02,\ 4.26)$$

   und mit

$$y_3 = (1.95,\ 1.67,\ 1.34,\ 1.32,\ 1.58,\ 1.64,\ 1.70,\ 2.62,\ 2.73,\ 3.61).$$

3. a) Passen Sie ein einfaches lineares Regressionsmodell an die Ozondaten in Bsp. 7.2.1 an, wobei Sie anstatt **Ozone** die Quadratwurzel davon als abhängige Größe verwenden. Definieren Sie dazu zuerst eine neue Variable **Ozone2** durch `Ozone2 = sqrt(Ozone)`. Zeichen Sie die zugehörige Punktwolke mit Regressionsgeraden und einen QQ- und einen Tukey-Anscombe-Plot der Residuen.

   b) Vergleichen Sie die Resultate mit den Modellen aus Bsp. 7.2.1 und Bsp. 7.3.1. Zeichnen Sie einen Scatterplot von **Ozone** gegen **Temp**, und fügen Sie alle drei geschätzten Regressionsfunktionen der Abbildung hinzu.

   Hinweis: $\sqrt{y} = a + b \cdot x$ ist äquivalent zu $y = (a + b \cdot x)^2$!

   c) Wiederholen Sie a) und b), wobei Sie eine Logarithmustransformation der Ozonwerte durchführen.

   d) Welches der vier Modelle beschreibt ihrer Meinung nach den Zusammenhang zwischen Temperatur und Ozonkonzentration am besten?

4. Tabelle 7.1 enthält Messungen der Länge (in cm) und des Gewichts (in kg) von 25 Alligatoren aus Florida. Während die Länge von wild lebenden Tieren per Luftaufnahme leicht erfasst werden kann, ist das Gewicht sehr schwer zu bestimmen. Aus diesem Grund ist man an einem Modell interessiert, mit dessen Hilfe das Gewicht eines Alligators aufgrund seiner Länge geschätzt werden kann.

Länge	239	147	160	183	208	183	173	193	198	188	218	218	325
Gewicht	58.9	12.7	15.0	38.5	36.2	27.6	17.7	19.0	25.8	23.1	36.2	40.8	165.8
Länge	218	188	226	290	373	238	175	216	224	155	229	229	
Gewicht	37.6	24.5	38.0	89.2	289.9	49.8	16.3	38.0	31.7	19.9	48.0	46.2	

Tabelle 7.1: Länge und Gewicht von 25 Alligatoren

a) Erstellen Sie einen Dataframe mit den Daten aus Tabelle 7.1.

b) Plotten Sie das Gewicht in Abhängigkeit der Länge und zeichnen Sie die zugehörige Regressionsgeraden ein. Die Anpassung ist offensichtlich sehr schlecht.

c) Da das Gewicht vermutlich proportional zum (dreidimensionalen) Volumen ist, liegt es nahe, eine polynomiale Regression 3. Grades durchzuführen. Zeichnen Sie die zugehörige Regressionsfunktion in die Abbildung aus a) ein.

d) Denkbar ist auch ein Modell der Form $Gewicht = a \cdot Laenge^3$, das von einer strengen Proportionalität ausgeht (wenn doppelt so lang, dann auch doppelte Dicke und doppelte Breite). Dieses Modell kann man durch

```
> lm(gewicht ~ 0+I(laenge^3))
```

anpassen. Zeichnen Sie auch hier die zugehörige Regressionsfunktion in die Abbildung aus a) ein.

e) Wiederholen Sie die Plots, wobei Sie die x-Achse von 0 bis 380 und die y-Achse von -100 bis 300 zeichnen.

f) Zeichen Sie Tukey-Anscombe-Plots der Residuen für die drei Modelle und vergleichen Sie sie.

g) Werten Sie die drei Regressionsmodelle an den Stellen 100, 200, 300, 400 (cm) aus.

5. Laden Sie den Datensatz **trees**, der Durchmesser (Variable **Girth**), Höhe (Variable **Height**) und Volumen (Variable **Volume**) von Bäumen enthält.

   a) Bestimmen Sie Pearson- und Rang-Korrelationen zwischen den Variablen.

   b) Zeichnen Sie Scatterplots von **Volume** gegen **Height** und **Volume** gegen **Girth**. Zeichnen Sie Regressionsgeraden ein.

   c) Ist **Girth=0**, so sollte auch **Volume=0** sein. Schätzen Sie deshalb ein lineares Regressionsmodell ohne konstanten Term (Intercept) durch

```
> lm(Volume ~ 0 + Girth)
```

und plotten Sie es. Interpretieren Sie das Resultat!

d) Ein Baumstamm kann approximativ als Zylinder angesehen werden. Wovon hängt das Volumen eines Zylinders ab? Können Sie ein besseres Modell für das Baumvolumen entwickeln?

6.* Laden Sie den Datensatz `vitcap2` in der Bibliothek `ISwR`. Er enthält Messungen der Lungenfunktion (Merkmal `vital.capacity`) von Arbeitern in der Cadmium-Industrie. Ein Teil der Arbeiter war über 10 Jahre lang Cadmium ausgesetzt; ein weiterer Teil war weniger als 10 Jahre lang Cadmium ausgesetzt; der Rest war nicht exponiert. Untersuchen Sie, welche Merkmale einen Einfluß auf die Lungenfunktion haben.

7.* Viele Untersuchungen zeigen, dass Rauchen die Lungenfunktion beeinflusst. Meistens werden diese Untersuchungen an Erwachsenen durchgeführt.

Um zu untersuchen, ob solche Effekte auch schon bei rauchenden Kindern und Jugendlichen auftreten, wurde die Lungenfunktion bei 654 Kindern und Jugendlichen im Rahmen von Routineuntersuchungen in einer Kinderklinik gemessen. Außerdem wurden die Teilnehmer der Studie gefragt, ob sie Raucher sind.

Ein Standardmaß für die Lungenfunktion ist FEV (forced expiratory volume), dass angibt, wieviel Luft eine Person in einer kurzen Zeitspanne ausatmen kann. Ein hoher FEV-Wert ist also ein Indikator für eine gute Lungenfunktion. Es ist bekannt, dass Rauchen über einen längeren Zeitraum bei Erwachsenen die Lungenfunktion im Mittel verschlechtert.

Der Datensatz `FEV.csv`[1] enthält die Daten der 654 Kinder und Jugendlichen. Gemessen wurden Alter (Spalte 3, in Jahren), FEV (Spalte 4, Einheit Liter/Sekunde), Größe (Spalte 5, Einheit inch), Geschlecht (Spalte 6, 1: männlich, 2: weiblich), und das Rauchverhalten (Spalte 7, 1: Raucher, 2: Nichtraucher).

Lesen sie diesen Datensatz in R ein und versehen Sie die Spalten mit Namen. Analysieren Sie diesen Datensatz. Untersuchen Sie insbesondere die Frage, ob ein Einfluß des Rauchens auf die Lungenfunktion erkennbar ist.

8.* Analysieren Sie (mit Methoden aus Kapitel 3-7) den Datensatz **iris**, der Messungen der Länge und Breite der Kelch- und Blütenblätter (Variablen **Sepal.Length** und **Sepal.Width** bzw. **Petal.Length** und **Petal.Width**) von drei verschiedenen Irisarten enthält.

[1] Sie finden den Datensatz unter http://courses.washington.edu/b518/datasets/FEVdata.csv und zugehörige Dokumentation unter http://courses.washington.edu/b518/datasets/FEVinfo.txt

9.* Der Datensatz `corbicula.csv` enthält Messungen von Länge, Breite, Höhe, Gesamtmasse und Weichgewebemasse von 199 Körbchenmuscheln.

a) Passen Sie ein multiples lineares Regressionsmodell für die Gesamtmasse in Abhängigkeit von Länge, Breite und Höhe an. Zeichnen Sie einen QQ-Plot und einen Tukey-Anscombe-Plot der Residuen.

Wiederholen Sie dies für die Weichgewebemasse.

b) Wie hängen Gesamtmasse und Weichgewebemasse voneinander ab?

8 Kategorielle Daten

8.1 2×2-Kontingenztafeln

Modellannahme: $x_1, ..., x_{n_1}$, $y_1, ..., y_{n_2}$ sind Realisierungen unabhängiger Zufallsvariablen $X_1, ..., X_{n_1}$, $Y_1, ..., Y_{n_2}$ mit

$$\begin{aligned} P(X_i = 1) &= p_1, \qquad P(X_i = 0) = 1 - p_1, \\ P(Y_j = 1) &= p_2, \qquad P(Y_j = 0) = 1 - p_2. \end{aligned}$$

Die Trefferwahrscheinlichkeiten p_1 und p_2 sind unbekannt.

a_j sei die Anzahl der Treffer in der j-ten Stichprobe.

Die Daten lassen sich in einer 2×2-Kontingenztafel darstellen:

	Treffer	Niete	$\sum$
1. Stichpr.	a_1	$n_1 - a_1$	n_1
2. Stichpr.	a_2	$n_2 - a_2$	n_2
$\sum$	$a_1 + a_2$	$n_1 + n_2 - a_1 - a_2$	$n_1 + n_2$

8.1.1 Beispiel

An 111 Patienten mit Kardiogenem Schock wurde im Zeitraum 1993-1999 eine Herzkranzgefäßerweiterung (PTCA) vorgenommen. Bei der Auswertung der Daten sollte unter anderem untersucht werden, ob ein Myokardinfarkt in der Vorgeschichte eines Patienten einen Einfluß auf die Erfolgsaussichten dieser Operation hat. Die zugehörige 2×2-Kontingenztafel

	PTCA erfolgreich	PTCA nicht erfolgreich
Vor-Myokardinfarkt	35	13
kein Vor-Myokardinfarkt	76	7

und eine graphische Darstellung der Anteile in einem Mosaik-Plot (siehe Abb. 8.1) erhält man in R durch die folgenden Befehle.

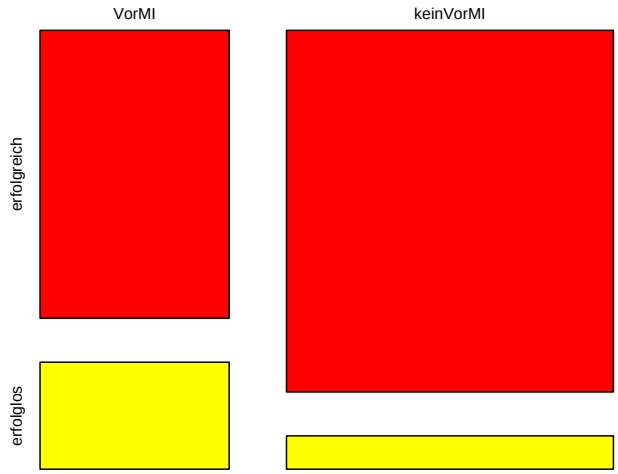


Abbildung 8.1: Mosaik-Plot der 2×2-Kontingenztafel aus Beispiel 8.1.1

```
> erfolgreich = c(35,76)
> erfolglos = c(13,7)
> PTCA = cbind(erfolgreich,erfolglos)
> rownames(PTCA) = c("VorMI","keinVorMI")
> PTCA
           erfolgreich  erfolglos
VorMI               35         13
keinVorMI           76          7
> mosaicplot(PTCA, color=T)
```

Man erkennt, dass der Anteil der erfolgreich operierten in der Gruppe ohne Vor-Myokardinfarkt deutlich größer ist.

8.2 Vergleich von zwei Wahrscheinlichkeiten

Bei 2×2-Kontingenztafeln interessiert man sich meist dafür, ob sich die Trefferwahrscheinlichkeiten p_1 und p_2 unterscheiden.

Es sei $\hat{p}_j = a_j/n_j$ die relative Trefferzahl in der j-ten Stichprobe,

$\hat{p} = \frac{a_1+a_2}{n_1+n_2}$ sei die relative Gesamt-Trefferzahl aus beiden Stichproben.

Beim **zweiseitigen Test**

$$H_0 : \; p_1 \; = \; p_2 \quad \text{gegen} \quad H_1 : \; p_1 \; \neq \; p_2$$

verwendet man die **Prüfgröße**

$$T \;\; = \;\; \frac{n_1 \cdot n_2}{n_1 + n_2} \cdot \frac{(\hat{p}_1 - \hat{p}_2)^2}{\hat{p} \cdot (1 - \hat{p})}, \quad \text{falls } n_1 + n_2 \geq 60 \text{ gilt.}$$

Testentscheid:

H_0 verwerfen, falls $T \geq \chi^2_{1,1-\alpha.}$
Keinen Widerspruch zu H_0, falls $T < \chi^2_{1,1-\alpha.}$

Beim **einseitigen Test**

$$H_0 : p_1 = p_2 \;\; (\text{bzw. } p_1 \leq p_2) \quad \text{gegen} \quad H_1 : p_1 > p_2$$

ist die **Prüfgröße**

$$T \;\; = \;\; \sqrt{\frac{n_1 \cdot n_2}{n_1 + n_2}} \cdot \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p} \cdot (1 - \hat{p})}} \, .$$

Testentscheid:

H_0 verwerfen, falls $T \geq c_{1-\alpha}$,
kein Widerspruch zu H_0, falls $T < c_{1-\alpha}$,

wobei $c_{1-\alpha}$ das $(1-\alpha)$-Quantil von N(0,1) ist.

Bemerkung: Für mittlere Stichprobenumfänge (etwa $20 < n_1 + n_2 < 60$) sollte man die Yates-Korrektur verwenden; für $n_1 + n_2 < 20$ kann man den exakten Test von Fisher verwenden.

8.2.1 Beispiel

Wir wenden den zweiseitigen Test ohne Yates-Korrektur auf die Daten aus Beispiel 8.1.1 an:

```
> prop.test(PTCA, alternative="two.sided", correct = F)
     2-sample test for equality of proportions
     without continuity correction
data:  PTCA
```

```
X-squared = 8.1767, df = 1, p-value = 0.004243
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.32570363 -0.04728834
sample estimates:
   prop 1    prop 2
0.7291667 0.9156627
```

Man erhält einen p-Wert von 0.004, die Hypothese kann also auf dem 1%-Niveau verworfen werden.

Die relativen Häufigkeiten sind $\hat{p}_1 = 0.73, \hat{p}_2 = 0.92$; somit ergibt sich eine Differenz $\hat{p}_1 - \hat{p}_2 = -0.19$. Die Wahrscheinlichkeit einer erfolgreichen PTCA ist also geringer bei Patienten, die schon einmal einen Herzinfarkt erlitten haben.

Ein 95%-Konfidenzintervall für die Differenz $p_1 - p_2$ ist gegeben durch $(-0.33, -0.05)$.

Da der Stichprobenumfang recht groß ist, ergeben sich ähnliche Ergebnisse, wenn man die Yates-Korrektur (Option `correct = T`) verwendet. Man erhält einen p-Wert von 0.009. Auch der exakte Test von Fisher führt hier zum gleichen Ergebnis; der p-Wert ist 0.006:

```
> fisher.test(PTCA)
     Fisher's Exact Test for Count Data
data:  PTCA
p-value = 0.005824
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.0775178 0.7459275
sample estimates:
odds ratio
 0.2509021
```

Dabei ist das besonders in der Medizin beliebte *odds ratio* (Quotenverhältnis) definiert als

$$or = \frac{p_1/(1-p_1)}{p_2/(1-p_2)}.$$

$or = 1$ gilt genau dann, wenn $p_1 = p_2$ ist; die zweiseitige Alternative $p_1 \neq p_2$ kann also auch als $or \neq 1$ formuliert werden. Die entsprechende empirische Größe ist

$$\hat{or} \;=\; \frac{\hat{p}_1/(1-\hat{p}_1)}{\hat{p}_2/(1-\hat{p}_2)} \;=\; \frac{a_1(n_2-a_2)}{a_2(n_1-a_1)} \;=\; \frac{35 \cdot 7}{76 \cdot 13} \;=\; 0.25.$$

Das Konfidenzintervall $(0.08, 0.75)$ für or mit Werten kleiner als 1 korrespondiert mit dem Konfidenzintervall für $p_1 - p_2$, das nur Werte kleiner als Null enthält.

8.3 $2 \times k$-Kontingenztafeln und Multiple Tests

Liegt wieder eine binäre Zielgröße vor, die aber in $k > 2$ Gruppen beobachtet wird, so können die Daten in einer $2 \times k$-Kontingenztafel dargestellt und analysiert werden.

8.3.1 Beispiel

Insgesamt überlebten von 139 Patienten mit Kardiogenem Schock nur 59 einen bestimmten Zeitraum. An der folgenden Tabelle ist zu erkennen, dass sich die relative Überlebenshäufigkeit $\hat{p}_j$ in verschiedenen Altersgruppen stark unterscheidet:

Gruppe	Alter	überlebt	tot	$\hat{p}_j$
1	jünger als 55	16	10	0.62
2	zwischen 55 und 65	22	15	0.59
3	zwischen 65 und 75	16	33	0.33
4	älter als 75	5	22	0.19

Auch für diese 2×4-Tafel kann man mit der Funktion `prop.test` die Hypothese testen, ob alle Überlebenswahrscheinlichkeiten gleich sind, ob also die Hypothese $H_0 : p_1 = p_2 = p_3 = p_4$ gilt.

```
> lebt = c(16,22,16,5)
> tot = c(10,15,33,22)
> TodKH = cbind(lebt,tot)
> rownames(TodKH) = c("Alter<55", "Alter<65", "Alter<75", "Alter>75")
> TodKH
         lebt tot
Alter<55   16  10
Alter<65   22  15
```

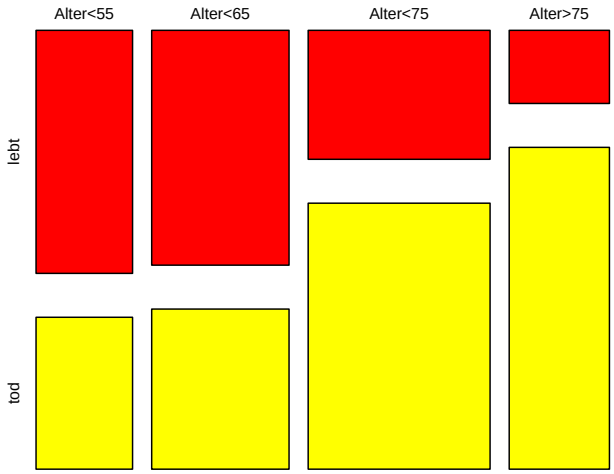


Abbildung 8.2: Mosaik-Plot der $2 \times k$-Kontingenztafel aus Beispiel 8.3.1

```
Alter<75   16  33
Alter>75    5  22
> mosaicplot(TodKH, color=T)
> prop.test(TodKH)
     4-sample test for equality of proportions
     without continuity correction
data:  TodKH
X-squared = 16.5149, df = 3, p-value = 0.0008891
alternative hypothesis: two.sided
sample estimates:
   prop 1    prop 2    prop 3    prop 4
0.6153846 0.5945946 0.3265306 0.1851852
```

Der p-Wert von 0.001 ist hochsignifikant. Dennoch hilft einem dieses Resultat kaum weiter: man möchte wissen, zwischen welchen Altersgruppen sich die Überlebenswahrscheinlichkeiten unterscheiden. Einen guten Eindruck vermittelt der Mosaikplot in Abb. 8.2.

Es liegt nun nahe, jeweils zwei Gruppen wie in Abschnitt 8.2 miteinander zu vergleichen. Dies kann mit der Funktion `pairwise.prop.test` geschehen:

```
> pairwise.prop.test(TodKH, p.adjust.method="none")
     Pairwise comparisons using Pairwise comparison of proportions
```

```
data:  TodKH
          Alter<55  Alter<65  Alter<75
Alter<65  1.0000    -         -
Alter<75  0.0306    0.0239    -
Alter>75  0.0035    0.0025    0.2933
P value adjustment method: none
```

Man erhält signifikante Unterschiede (auf dem 5%-Niveau) zwischen Gruppe 1 und den Gruppen 3 und 4, sowie zwischen Gruppe 2 und den Gruppen 3 und 4. Allerdings hat man jetzt sechs Tests durchgeführt, so dass die Gesamtwahrscheinlichkeit für einen Fehler 1. Art bis zu 30% betragen kann (vgl. Abschitt 5.3). Um dies zu vermeiden, kann man wieder die Bonferroni- oder die Holmskorrektur verwenden. Damit wird gesichert, dass die Wahrscheinlichkeit für einen Fehler 1. Art insgesamt ein bestimmtes Niveau einhält.

```
> pairwise.prop.test(TodKH, p.adjust.method="holm")
     Pairwise comparisons using Pairwise comparison of proportions
data:  TodKH
          Alter<55  Alter<65  Alter<75
Alter<65  1.000     -         -
Alter<75  0.096     0.096     -
Alter>75  0.017     0.015     0.587
P value adjustment method: holm
```

Allerdings sind jetzt die Unterschiede zwischen Gruppe 1 und Gruppe 3, sowie zwischen Gruppe 2 und Gruppe 3, nicht mehr signifikant. Zwar ist jetzt die Niveaueinhaltung gesichert, aber auf Kosten einer geringeren Güte! Aus diesem Grunde ist es wichtig, sich vor Testdurchführung zu überlegen, welche Gruppeneinteilung sinnvoll ist, und welche Gruppen man überhaupt vergleichen will: oft ist man zum Beispiel nur an Vergleichen zwischen den Behandlungsgruppen und einer Kontrollgruppe interessiert, was die Zahl der paarweisen Vergleiche stark reduziert. Im vorliegenden Beispiel könnte man (gestützt auf andere Untersuchungen) von vornherein alle Patienten bis 65 Jahre in einer Gruppe zusammenfassen. Das Resultat sieht dann folgendermaßen aus.

```
> lebt2 = c(38,16,5)
> tot2 = c(25,33,22)
> TodKH2 = cbind(lebt2,tot2)
> rownames(TodKH2) = c("Alter<65", "Alter<75", "Alter>75")
```

```
> TodKH2
          lebt2  tot2
Alter<65     38    25
Alter<75     16    33
Alter>75      5    22

> pairwise.prop.test(TodKH2, p.adjust.method="holm")
     Pairwise comparisons using Pairwise comparison of proportions
data:  TodKH2
          Alter<65  Alter<75
Alter<75  0.0132    -
Alter>75  0.0020    0.2933
P value adjustment method: holm
```

Bei dieser Gruppenwahl sind Unterschiede zwischen der Gruppe "jünger als 65" und den beiden anderen Altersgruppen signifikant.

8.4 Der χ^2-Unabhängigkeits-Tests in Kontingenztafeln

Merkmal 1 nimmt die Werte $1, 2, \ldots, r$ an,
Merkmal 2 nimmt die Werte $1, 2, \ldots, s$ an.

$n_{k,l}$: die Anzahl der Paare (x_i, y_i) $(i = 1, \ldots, n)$ mit $x_i = k$, $y_j = l$.

Diese Daten lassen sich in einer $r \times s$-Kontingenztafel anordnen:

	l					
k	1	2	3	$\cdots$	s	$\sum$
1	$n_{1,1}$	$n_{1,2}$	$n_{1,3}$	$\cdots$	$n_{1,s}$	n_{1+}
2	$n_{2,1}$	$n_{2,2}$	$n_{2,3}$	$\cdots$	$n_{2,s}$	n_{2+}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
r	$n_{r,1}$	$n_{r,2}$	$n_{r,3}$	$\cdots$	$n_{r,s}$	n_{r+}
$\sum$	n_{+1}	n_{+2}	n_{+3}	$\cdots$	n_{+s}	n

Modell: X, Y sind diskrete Zufallsvariablen, $P(X = k, Y = l) = p_{k,l}$ sind unbekannt. Man will die Hypothese

$$H_0: \quad X,\ Y \text{ unabhängig} \quad (\Leftrightarrow\ p_{k,l} = P(X = k) \cdot P(Y = l)\ \ \forall\, k, l)$$

gegen die allgemeine Alternative

$$H_1: \quad X,\ Y \text{ nicht unabhängig}$$

testen.

Beachte: Unter H_0 ist $\frac{n_{k,l}}{n} \approx \frac{n_{k+}}{n} \cdot \frac{n_{+l}}{n}$ zu erwarten.

Prüfgröße:

$$T = \sum_{k=1}^{r} \sum_{l=1}^{s} \frac{\left(n_{k,l} - \frac{1}{n} \cdot n_{k+} \cdot n_{+l}\right)^2}{\frac{1}{n} \cdot n_{k+} \cdot n_{+l}}$$

Testentscheid:

H_0 ablehnen, falls $T \geq \chi^2_{(r-1)\cdot(s-1),1-\alpha}$

Kein Widerspruch, falls $T < \chi^2_{(r-1)\cdot(s-1),1-\alpha}$.

8.4.1 Beispiel

Der Datensatz `caith` aus der MASS-library enthält Augen- und Haarfarbe von Personen, die aus der schottischen Region Caithness stammen.

```
> library(MASS)
> data(caith)
> help(caith)
> caith
       fair  red  medium  dark  black
blue    326   38     241   110      3
light   688  116     584   188      4
medium  343   84     909   412     26
dark     98   48     403   681     85

> chisq.test(caith, correct = F)
     Pearson's Chi-squared test
data:  caith
X-squared = 1240.039, df = 12, p-value = < 2.2e-16
```

Die Hypothese der Unabhängigkeit von Augen-und Haarfarbe wird offensichtlich verworfen.

8.5 Der χ^2-Anpassungstest

Situation: Es werden n unabhängige Zufallsexperimente mit s möglichen Ausgängen 1, 2,. . .,s durchgeführt.

Die Wahrscheinlichkeiten p_j für Ausgang j $(j = 1, \ldots, s)$ sind unbekannt.

$q_1, ..., q_s$ sind gegebene hypothetische Wahrscheinlichkeiten. Getestet werden soll die

$$\text{Hypothese } H_0: \quad p_1 = q_1,\ p_2 = q_2,\ \ldots, p_s = q_s,$$

gegen die

$$\text{Alternative } H_1: \quad p_j \neq q_j \text{ für mindestens ein } j.$$

Daten: Ist n_j die Anzahl der Zufallsexperimente mit Ergebnis j, so ist $n_j \approx n \cdot q_j$ unter H_0 zu erwarten. Weiter ist $n_1 + n_2 + ... + n_s = n$.

Prüfgröße:

$$T = \sum_{j=1}^{s} \frac{(n_j - nq_j)^2}{nq_j}$$

Testentscheid: (Niveau α-Test)

H_0 ablehnen, falls $T \geq \chi^2_{s-1,1-\alpha}$.

Kein Widerspruch zu H_0, falls $T < \chi^2_{s-1,1-\alpha}$.

Hierbei sollte $n \cdot q_j \geq 5$ für alle j sein.

8.5.1 Beispiel

Gregor Mendel beobachtete 1865 in einem Experiment simultan die Form (rund, kantig) und Farbe (gelb, grün) von Erbsen, die er gezüchtet hatte. Nach seiner Theorie der Vererbungslehre verhalten sich die Wahrscheinlichkeiten für die Merkmalsausprägungen (r, ge), (r, gr), (k, ge) und (k, gr) wie 9:3:3:1. Er zählte unter $n = 556$ Erbsen 315 mal (r, ge), 108 mal (r, gr), 101 mal (k, ge) und 32 mal (k, gr).

Die Vererbungstheorie liefert also die hypothetischen Wahrscheinlichkeiten
$q_1 = 9/16$, $q_2 = 3/16 = q_3$, $q_4 = 1/16$

$\Rightarrow\ n \cdot q_1 = 312.75,\ n \cdot q_2 = 104.25, n \cdot q_3 = 104.25,\ n \cdot q_4 = 34.75.$

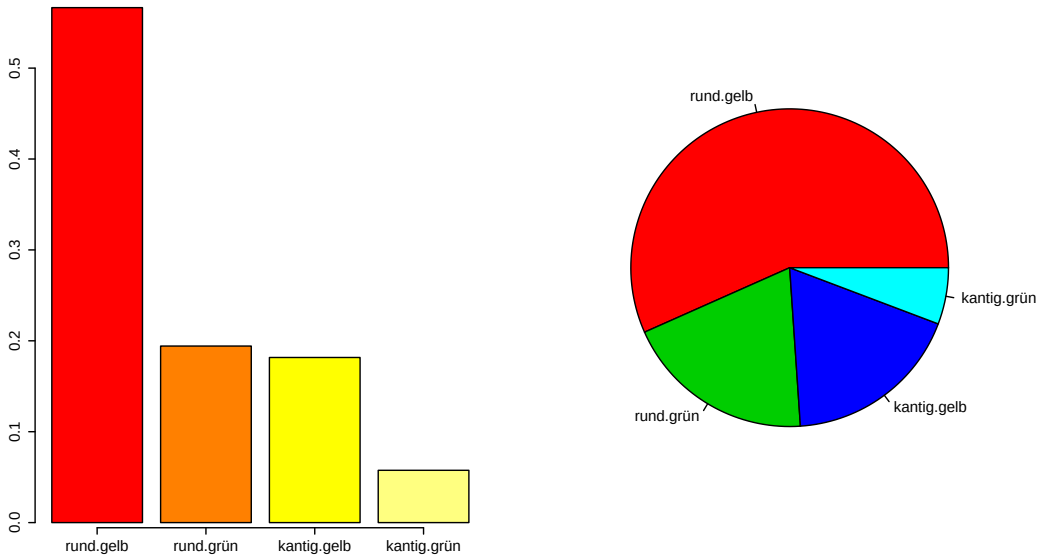


Abbildung 8.3: Balken- und Kreisdiagramm zu Beispiel 8.5.1

Der folgende R-Code liefert zuerst ein Balken- und ein Kreisdiagramm der beobachteten Häufigkeiten (siehe Abb. 8.3). Dabei sind im allgemeinen Balkendiagramme vorzuziehen, da das Auge Flächenverhältnisse nur schlecht beurteilen kann. Es folgt ein Balkendiagramm (Abb. 8.4), bei dem die hypothetischen Wahrscheinlichkeiten und die relativen Häufigkeiten zum Vergleich nebeneinander gestellt werden. Danach wird mit dem χ^2-Test geprüft, ob Mendel's Beobachtungen seiner Theorie entsprechen.

```
> n = c(315,108,101,32)
> names(n) = c("rund.gelb","rund.grün","kantig.gelb","kantig.grün")
> par(mfrow=c(1,2), cex=0.7)
> barplot(n/sum(n))
> pie(n, col=2:5)
> prob = c(9/16,3/16,3/16,1/16)
> w = rbind(n/sum(n),prob)
> par(mfrow=c(1,1), cex=1)
> barplot(w, beside=T, ylim=c(0,0.6), space=c(0.2,1))
> chisq.test(n, p=prob)
     Chi-squared test for given probabilities
data:  n
X-squared = 0.47, df = 3, p-value = 0.9254
```

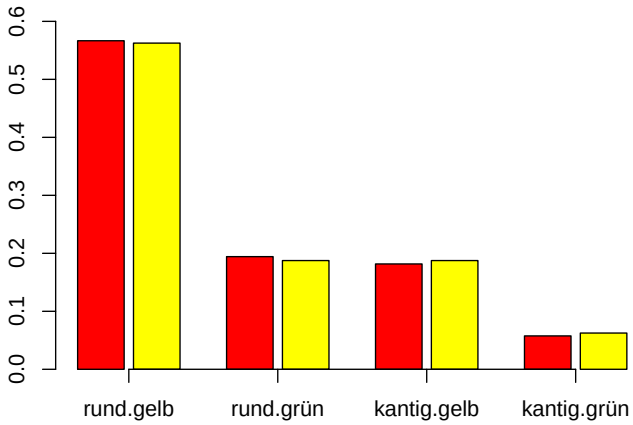


Abbildung 8.4: Vergleich der relativen Häufigkeiten (links) mit den hypothetischen Wahrscheinlichkeiten (rechts)

8.5.2 Bemerkung

1. Das Ziel besteht eigentlich darin, eine Verteilungshypothese zu bestätigen. Bei Nichtablehnung von H_0 ist die Hypothese aber statistisch nicht gesichert!

   Lösung: Hypothese und Alternative vertauschen?

   $\hookrightarrow$ funktioniert bei Anpassungstests nicht!

2. Der χ^2-Anpassungstest kann auch verwendet werden, falls unbekannte Parameter vorliegen.

   Beispiel: H_0 : Es liegt $Bin(5, p)$-Verteilung vor, $p \in (0, 1)$ unbekannt.

   Allgemein: Sei $\vartheta = (\vartheta_1, \ldots, \vartheta_l)$,

   $$H_0 : \; p_j = q_j(\vartheta) \; (j = 1, \ldots, s) \text{ für ein } \vartheta \in \Theta.$$

   Falls $\hat{\vartheta}$ Maximum-Likelihood-Schätzer, so ist

   $$T \;=\; \sum_{j=1}^{s} \frac{\left(n_j - n \cdot q_j(\hat{\vartheta})\right)^2}{n \cdot q_j(\hat{\vartheta})}$$

   für großes n unter H_0 approximativ χ^2_{s-1-l}-verteilt

   $\Rightarrow$ Ablehnung von H_0 falls $T > \chi^2_{s-1-l,1-\alpha}$.

   Die Funktion `goodfit` im Package `vcd` führt diesen Test für die Binomialverteilung durch.

3. Liegen Zähldaten auf $\mathbb{N}_0$ vor (z.B. Poissonverteilung), so ist eine Gruppierung notwendig. Für die Poissonverteilung und die negative Binomialverteilung ist der Test ebenfalls mittels der Funktion `goodfit` durchführbar.
4. Für stetige Verteilungen sollte der χ^2-Test nicht verwendet werden.

8.6 Übungen

1. Vollziehen Sie alle Beispiele in Kap. 8 nach.
2. Untersucht wurde die Wirkung einer Polio-Reihenimpfung. Dabei wurden $n_1 = 200745$ Personen geimpft; davon erkrankten $a_1 = 57$ (in einem bestimmtem Zeitraum) an Polio. $n_2 = 201229$ Personen erhielten ein Placebo; davon erkrankten $a_2 = 142$ Personen im gleichen Zeitraum. Stellen Sie eine Kontingenztafel auf und untersuchen Sie mit einem geeigneten zweiseitigen Test, ob die Impfung einen Einfluß auf die Erkrankungswahrscheinlichkeit hat.

   Wie müßte ein einseitiger Test formuliert werden?
3. Wandeln Sie den Data Frame `caith` (vgl. Beispiel 8.4.1) durch

   ```
   > caith2 = as.matrix(caith)
   ```

   in eine Matrix bzw. Kontingenztafel um. Fertigen Sie zuerst einen Mosaik-Plot des Datensatzes `caith2` an, und danach einen Assoziationsplot mittels der Funktion `assocplot()`. Vergleichen Sie beide Plots. Was stellt der zweite Plot dar? (Hierzu sollten Sie auch die Hilfe zu `assocplot` zu Rate ziehen.) Welche Schlüsse könne Sie aus den beiden Plots ziehen?
4. An 48 Blüten einer bestimmten Pflanze wurden die Merkmale *Farbe* und *Blatt-Beschaffenheit* bestimmt. Man erhielt die folgende 2×2-Kontingenztafel.

MM 1 / MM 2	weiß	rosa	$\sum$
glatt	12	4	16
rauh	9	23	32
$\sum$	21	27	48

   Zeichnen Sie Mosaik- und Assoziationsplots, und testen Sie die Hypothese, dass die beiden Merkmale unabhängig sind.
5. In dieser Aufgabe sollen die Daten der 83 Patienten aus Beispiel 8.1.1 verwendet werden, die keinen Myokardinfarkt in der Vorgeschichte hatten.

Bestimmen Sie mit Hilfe der Funktion `binom.test` das Pearson-Clopper-Konfidenzintervall zum Konfidenzniveau $1 - \alpha = 0.95$ für die Wahrscheinlichkeit p einer erfolgreichen PTCA bei Patienten ohne Vor-Myokardinfarkt.

Bestimmen Sie zum Vergleich das Wald-Konfidenzintervall mit Intervallgrenzen

$$\hat{p} \pm c_{1-\alpha/2} \cdot \sqrt{\hat{p} \cdot (1-\hat{p})/n}, \tag{8.1}$$

wobei $c_{1-\alpha/2}$ das $(1-\alpha/2)$-Quantil der Standard-Normalverteilung bezeichnet.

Bestimmen Sie auch das adjustierte Wald-Konfidenzintervall, indem Sie zwei Erfolge und zwei Misserfolge zu den Daten dazu nehmen und wieder die Grenzen in (8.1) verwenden.

6. a) Erzeugen Sie $n = 10$ Zufallszahlen aus einer $Bin(5, 0.5)$-Verteilung durch

```
> x = rbinom(10,5,0.5)
```

Erstellen Sie daraus mit

```
> tabelle = table(factor(x,levels=0:5))
```

eine Häufigkeitstabelle [1]. Zeichnen Sie ein Balkendiagramm. Wiederholen Sie dies für die Stichprobenumfänge $n = 100$ und $n = 1000$.

b) Berechnen Sie mit der Funktion `dbinom` die theoretischen Binomialwahrscheinlichkeiten. Zeichnen Sie ein Balkendiagramm, indem die theoretischen Wahrscheinlichkeiten (Vektor `p`) und die empirischen Wahrscheinlichkeiten (Vektor `pn`) nebeneinander abgetragen sind.

c) Alternativ zu b) können Sie auch durch

```
> library(vcd)
> rootogram(pn, p, scale = "raw", type = "standing")
```

ein sogenanntes Rootogram zeichnen. Verwenden Sie statt `scale = "raw"` auch die Option `scale = "sqrt"`.

d) Wiederholen Sie a) bis c) für die Trefferwahrscheinlichkeit $p = 0.1$.

7. Lesen Sie den in der Datei `parasit.csv` abgespeicherten Datensatz ein, etwa als Data Frame `parasit.df` . Der Datensatz enthält Anzahlen einer bestimmten Parasitenart, die in Aalen gefunden wurden, sowie das jeweilige Fangjahr. Passen Sie jeweils eine Poissonverteilung an die Daten eines Fangjahres an, und testen Sie, ob die Daten mit einer Poissonverteilung verträglich sind. Verwenden Sie dazu die Funktion `goodfit` (vgl. Bem. 8.5.2):

[1] Der Befehl `tabelle = table(x)` führt nicht zum gewünschten Resultat: enthält `x` z.B. die Zahlen 2,4,4,3,1,1,2,3,1,1, so ergibt der Aufruf `table(x)` die Tabelle

1	2	3	4
4	2	2	2

```
> library(vcd)
> count = parasit.df$Anzahl[parasit.df$Fangjahr == 2002]
> fit = goodfit(count, type = "poisson", method = "ML")
> summary(fit)
```

Zeichnen Sie jeweils auch ein Rootogramm durch `rootogram(fit, ...)`.

Wiederholen Sie dies, wobei Sie die Negative Binomialverteilung anstelle der Poissonverteilung verwenden.

8. Ein Treffer/Niete-Versuch wird in zwei verschiedenen Gruppen jeweils 20 mal durchgeführt. Man beobachtet in Gruppe 1 die Stichprobe

   1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 0 1 1 0

   und in Gruppe 2

   0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0

   Um zu testen, ob beide Stichproben aus der gleichen Grundgesamtheit stammen, wird ein Zwei-Stichproben-t-Test vorgeschlagen. Was halten Sie davon? Führen Sie den Test durch. Verwenden Sie dann den Test aus Abschnitt 8.2.

9.* Visualisieren Sie die Poisson-Approximation der Binomialverteilung (auch Gesetz seltener Ereignisse genannt).

9 Binäre Regression

Bei der binären Regression besitzt die Zielgröße Y nur die zwei Ausprägungen 1 und 0.

Beispiel (Toxizitätsversuch): Insekten werden verschiedenen Konzentrationen eines Gifts ausgesetzt; danach wird gezählt, wie viele Insekten tot sind (Ausprägung 1) bzw. wie viele überlebt haben (Ausprägung 0).

Frage: Wie hängt die Wahrscheinlichkeit für Ausprägung $Y = 1$ von der Dosis x ab?

Gesucht ist also $P(Y = 1|x) = ?$

Einfachster Versuch:

$$P(Y = 1|x) \quad = \quad a + b \cdot x.$$

Nachteil: dies kann auf negative Wahrscheinlichkeiten (bzw. auf Wahrscheinlichkeiten, die größer als 1 sind) führen.

Deshalb wählt man den Ansatz

$$P(Y = 1|x) \quad = \quad G(a + b \cdot x),$$

wobei $G : \mathbb{R} \to (0, 1)$ eine Verteilungsfunktion ist, und a, b unbekannte Parameter sind. G heißt auch Link-Funktion.

Wie bei der linearen Regression sind Verallgemeinerungen auf mehr als eine Einflußgröße möglich; z.B. ergibt sich für 2 Einflussgrößen das Modell

$$P(Y = 1|x, z) \quad = \quad G(a + b \cdot x + c \cdot z)$$

mit unbekannten Parametern a, b, c.

9.1 Logistisches Regressionsmodell

Als Linkfunktion wird die logistische Funktion

$$G(t) = \frac{1}{1+\exp(-t)},$$

verwendet. Es ist also

$$P(Y=1|x) \;=\; \frac{1}{1+\exp(-(a+b\cdot x))}$$

oder äquivalent dazu

$$\log\,\frac{P(Y=1|x)}{P(Y=0|x)} \;=\; a+b\cdot x\,.$$

Die Verhältnisse $\dfrac{P(Y=1|x)}{P(Y=0|x)} = \dfrac{P(Y=1|x)}{1-P(Y=1|x)}$ werden wieder als „odds“ bezeichnet; $\log\dfrac{P(Y=1|x)}{P(Y=0|x)}$ sind dann die „log odds“.

Parameter-Schätzung beim logistischen Modell:

n unabhängige Experimente,
x_j: Wert der Einflussgröße beim j-ten Experiment,
y_j: realisierter Wert (0 oder 1) im j-ten Experiment.

Setzt man $\pi_j = P(Y_j=1|x_j) = 1/(1+\exp(-(a+b\cdot x_j)))$, so erhält man die Likelihood-Funktion

$$L(y_1,\ldots,y_n) \;=\; \prod_{j=1}^{n} P(Y_j=y_j|x_j) \;=\; \prod_{j=1}^{n}\left(\pi_j^{y_j}\cdot(1-\pi_j)^{1-y_j}\right)\,.$$

Numerische Maximierung von $L(y_1,\ldots,y_n)$ bezüglich a und b ergibt Maximum-Likelihood-Schätzwerte $\hat{a}$ und $\hat{b}$.

9.1.1 Beispiel

Jeweils 40 Insekten wurden einer bestimmten Dosis x (in μg) eines Insektizids ausgesetzt[1]:

[1] Beispiel aus Venables und Ripley (2002, S. 190-194)

Dosis	1	2	4	8	16	32
$x = log_2$(Dosis)	0	1	2	3	4	5
tot (von 40)	1	6	15	23	30	36

Insgesamt wurden also 240 unabhängige Experimente durchgeführt: 40 Experimente bei Dosis 1, 40 Experimente bei Dosis 2, usw.

Die Dateneingabe ist in R auf mehrere Arten möglich.

- Man kann die Dosis als numerischen und die Response als logischen Vektor (hier jeweils der Länge 240) eingeben: `T` steht dabei für Erfolg; im Beispiel wird man den Tod eines Insekts als Erfolg ansehen.
- Werden viele Versuche bei gleichem Wert der Einflußgröße durchgeführt, so kann man die Anzahl der Erfolge bzw. Misserfolge in eine Matrix mit zwei Spalten schreiben: Spalte 1 enthält jeweils die Anzahl der Erfolge, Spalte 2 die Anzahl der Misserfolge.

Logistisches Modell:

Die folgenden Befehle verwenden die erste Möglichkeit der Dateneingabe. Danach wird ein logistisches Regressionsmodell mit Hilfe der Funktion `glm` angepasst.

```
> dosis = c(rep(1,40),rep(2,40),rep(4,40),rep(8,40),
  rep(16,40),rep(32,40))
> response = c(rep(T,1),rep(F,39),rep(T,6),rep(F,34),rep(T,15),
  rep(F,25),rep(T,23),rep(F,17),rep(T,30),rep(F,10),rep(T,36),rep(F,4))
     # Bei Dosis 1: 1 Erfolg, 39 Misserfolge, usw.
> wirkung.logit = glm(response ~ dosis, family=binomial)
> wirkung.logit$coefficients
(Intercept)       dosis
 -1.5739231   0.1530266
```

Als Schätzwerte für a und b ergeben sich also $\hat{a} = -1.57$ und $\hat{b} = 0.15$. Daraus folgt das Modell

$$p(x) \;=\; P(Y = 1|x) \;=\; \frac{1}{1 + \exp(1.57 - 0.15x)}\,.$$

Um die Anpassung zu beurteilen, kann man das angepasste logistische Modell zusammen mit den beobachteten relativen Erfolgshäufigkeiten zeichnen. Das Ergebnis ist in Abb. 9.1 zu sehen. Der Vergleich der logistischen Funktion gegen die relativen Häufigkeiten zeigt, dass das Modell nicht gut passt.

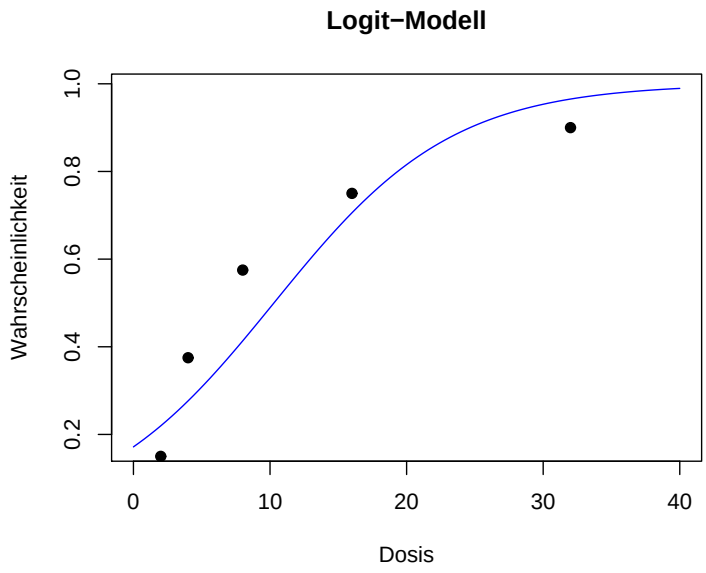


Abbildung 9.1: Logistisches Regressionsmodell in Abhängigkeit von der Dosis

```
> a = wirkung.logit$coefficients[1]
> b = wirkung.logit$coefficients[2]
> curve(1/(1+exp(-1*(a+b*x))), from=0, to=40, col="blue",
  xlab="Dosis", ylab="Wahrscheinlichkeit", main="Logit-Modell")
> points(c(1,2,4,8,16,32),c(1,6,15,23,30,36)/40, pch=16)
```

Als weiterer Versuch wird deshalb ein logistisches Modell in Abhängigkeit von $x = \log_2(\text{Dosis})$ angepasst. Dabei wird die zweite Art der Dateneingabe verwendet.

```
> logdosis = 0:5
> anzahl.erfolg = c(1,6,15,23,30,36)
> anzahl = cbind(anzahl.erfolg, anzahl.niete=40-anzahl.erfolg)
> wirkung2.logit = glm(anzahl ~ logdosis, family=binomial)
> wirkung2.logit$coefficients
(Intercept)    logdosis
  -2.766087    1.006807
> a = wirkung2.logit$coefficients[1]
> b = wirkung2.logit$coefficients[2]
> x = c(1,3,5)
> 1/(1+exp(-1*(a+b*x)))
```

```
[1] 0.1468805 0.5632428 0.9061874
> curve(1/(1+exp(-1*(a+b*x))), from=-1, to=6, col="blue",
  xlab="log(Dosis)", ylab="Wahrscheinlichkeit", main="Logit-Modell")
> points(logdosis,anzahl.erfolg/40, pch=16)
> library(MASS)
> dose.p(wirkung2.logit, p=1:3/4)
            Dose      SE
p = 0.25: 1.656201 0.2073056
p = 0.50: 2.747386 0.1656134
p = 0.75: 3.838571 0.2185321
```

Als Schätzwerte für a und b ergeben sich also $\hat{a} = -2.77$ und $\hat{b} = 1.01$. Daraus folgt das Modell

$$p(x) \;=\; P(Y = 1|x) \;=\; \frac{1}{1 + \exp(2.77 - 1.01x)}\,.$$

Danach wird die Wahrscheinlichkeit (unter diesem Modell) berechnet, mit der Insekten getötet werden, wenn die logarithmierte Dosis 1,3 oder 5 beträgt. Es folgt wieder der entsprechende Plot; man sieht, dass das Modell mit den logarithmierten Daten sehr viel besser passt (durchgezogene Linie in Abbildung 9.2).

Zuletzt wird mit Hilfe der Funktion `dose.p` aus der MASS-Library die sogenannten LD25, LD50 und LD75 Werte berechnet: das ist die (logarithmierte) Dosis, die notwendig ist, um 25% (bzw. 50%, 75%) der Schädlinge zu vernichten. Es ist also die Dosis $2^{3.84} = 14.3\mu g$ notwendig, um 75% der Schädlinge zu vernichten.

9.2 Probit-Modell

Als Linkfunktion wird $G(t) = \Phi(t)$ verwendet, wobei Φ die Verteilungsfunktion der Standardnormalverteilung ist. Hier gilt also

$$P(Y = 1|x) \;=\; \Phi(a + b \cdot x).$$

9.2.1 Beispiel

Ein Probit-Modell wird in R durch die Option `family=binomial(link=probit)` angefordert.

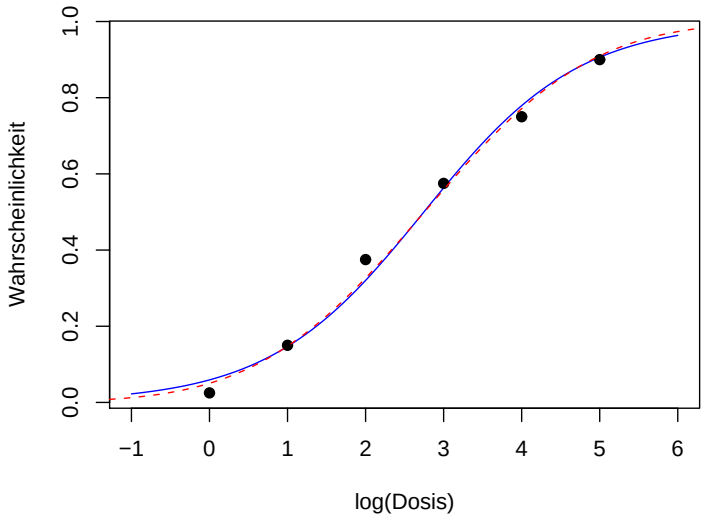


Abbildung 9.2: Logistisches Regressionsmodell (durchgezogene Linie) und Probit-Modell (gestrichelte Linie) in Abhängigkeit von log(Dosis)

```
> wirkung.probit = glm(anzahl ~ logdosis, family=binomial(link=probit))
> wirkung.probit$coefficients
(Intercept)    logdosis
 -1.6430946   0.5965797
> a = wirkung.probit$coefficients[1]
> b = wirkung.probit$coefficients[2]
> curve(pnorm(a+b*x), add=T, col="red", lty=2)
> dose.p(wirkung.probit, p=3/4)
            Dose      SE
p = 0.75: 3.884786 0.2107033
```

Man erhält das Modell

$$p(x) \;=\; P(Y=1|x) \;=\; \Phi(-1.64+0.60x).$$

Hier sind $2^{3.885} = 14.8\mu g$ erforderlich, um 75% der Insekten zu töten.

Abbildung 9.2 zeigt das angepasste logistische Regressionsmodell (durchgezogene Linie), das Probit-Modell (gestrichelt) sowie die empirischen Häufigkeiten. Man erkennt, dass sich beide Modelle kaum unterscheiden, zumindest solange $p(x)$ im Bereich von 0.1 bis 0.9 liegt.

9.3 Übungen

1. Vollziehen Sie die Beispiele in Kap. 9 nach.
2. Die Challenger–Katastrophe am 26.1.1986 war Folge einer unzureichenden Risikoanalyse. Da für den Tag des Starts der Challenger mit extremer Kälte (Temperaturen um 0° Celsius) gerechnet wurde, fragte man sich am Abend vorher, ob die Außentemperatur beim Start einen Einfluss auf die Zuverlässigkeit der Dichtungsringe hat.

   a) Dabei untersuchte man nur diejenigen früheren Shuttle–Flüge, bei denen Probleme an Dichtungsringen aufgetreten waren. Die bei dem jeweiligen Start herrschenden Außentemperaturen waren (der Größe nach geordnet)

   $$12, 14, 14.5, 17, 21, 21, 24.$$

   Stellen Sie die Werte graphisch dar. Können Sie an diesen Daten einen Einfluss der Außentemperatur erkennen?

   b) Nach der Katastrophe nahm eine von der U.S. Regierung eingesetzte Expertenkommission die Starttemperaturen der problemlos verlaufenen Flüge hinzu und stellte einen deutlich sichtbaren Einfluss der Außentemperatur fest: Die Probleme waren bei tendenziell niedrigeren Temperaturen aufgetreten, während sämtliche Flüge ohne Probleme im Temperaturbereich zwischen 19° und 27° Celsius stattfanden. Die geordnete Stichprobe der Temperaturen der problemlos verlaufenen Flüge ist

   $$19, 19.5, 19.5, 19.5, 20, 20.5, 21, 22, 22, 23, 23, 24.5, 24.5, 25.5, 26, 27.$$

   Stellen Sie diese Werte gemeinsam mit den Daten aus a) graphisch dar.

   c) Beschreiben Sie die Wahrscheinlichkeit $p(t)$ für den Ausfall eines Dichtungsrings in Abhängigkeit von der Außentemperatur t zum Startzeitpunkt durch ein logistisches Modell. Welche Schätzwerte ergeben sich für die Modellparameter? Plotten Sie die Ausfallwahrscheinlichkeit eines Dichtungsrings in Abhängigkeit von der Außentemperatur. Mit welcher Wahrscheinichkeit war (unter diesem Modell) bei einer prognostizierten Temperatur von 0° Celsius mit einem Ausfall zu rechnen?

   d) Wiederholen Sie Teil c) mit einem Probit-Modell.

3. a) Unter 747 Fällen des "Rocky Mountain Fleckfieber" im Westteil der USA starben 210 Patienten; von 661 Fällen im Ostteil der USA starben 122[2]. Besteht ein statistisch signfikanter Unterschied in der Letalität?

   b) Trennt man die Patienten in drei Altersgruppen, so erhält man die folgende Tabelle.

	Westteil		Ostteil	
Altersgruppe	Gesamt	Gestorben	Gesamt	Gestorben
Unter 15	108	13	310	40
15–39	264	40	189	21
Über 39	375	157	162	61
	747	210	661	122

   Erstellen Sie einen Dataframe mit diesen Daten. Erzeugen Sie daraus mit Hilfe des `table()`-Befehls eine dreidimensionale Kontingenztafel und zeichnen Sie einen Mosaikplot des Datensatzes. Analysieren Sie den Datensatz, indem Sie ein logistisches Modell anpassen. Versuchen Sie, die Güte der Anpassung zu beurteilen.

[2]Daten aus Peter Dalgaard, Introductory Statistics with R, S. 138 und S. 209

A Ergänzungen zur Statistik

A.1 Der p-Wert

In den meisten Statistik-Funktionen von R werden p-Werte ausgegeben. Deshalb sei an dieser Stelle nochmals an diesen Begriff erinnert.

Die Wirkungstabelle eines statistischen Tests sieht folgendermaßen aus:

	Test-Entscheidung	
	H_0 nicht verwerfen	H_0 wird verworfen, also gilt H_1
in Wirklichkeit gilt H_0	richtige Entscheidung	Fehler 1. Art
in Wirklichkeit gilt H_1	Fehler 2. Art	richtige Entscheidung

Vor Durchführung eines Tests wird ein Höchstwert α für die Wahrscheinlichkeit, einen Fehler 1. Art zu begehen, festgelegt. Ein Test mit dieser Eigenschaft heißt *Test zum Signifikanz-Niveau* α. Es gilt also

$$P_{H_0}(\text{Test entscheidet für } H_1) = \alpha \tag{A.1}$$

Zur Durchführung des Tests wird aus den Daten $x_1, ..., x_n$ der Wert $T(x_1, ..., x_n)$ einer geeigneten Testgröße $T = T(X_1, ..., X_n)$ berechnet. Zu beachten ist dabei, dass die Testgröße T (vor Versuchsdurchführung) eine Zufallsvariable mit einer bekannten Wahrscheinlichkeitsverteilung (z.B. einer χ^2- oder t-Verteilung) ist; berechnet man aber $T(x_1, ..., x_n)$ aus den Versuchsdaten $x_1, ..., x_n$, so erhält man eine bestimmte Zahl.

Die Hypothese H_0 wird abgelehnt, falls die Testgröße einen kritischen Wert c erreicht oder überschreitet. In diesem Fall sagt man, das Resultat des Tests sei *statistisch signifikant* (zum Niveau α). Es gilt also

H_0 wird verworfen,	falls $T(x_1, ..., x_n) \geq c$.
Es besteht kein Einwand gegen H_0,	falls $T(x_1, ..., x_n) < c$.

Gleichung (A.1) kann man also schreiben als

$$P_{H_0}(T \geq c) = \alpha \tag{A.2}$$

Bei Statistik-Programmen wird aus der Stichprobe $x = (x_1, \ldots, x_n)$ der sogenannte p–Wert $p^* = p^*(x)$ berechnet. Der p-Wert ist die Wahrscheinlichkeit, dass die Testgröße bei Gültigkeit der Nullhypothese mindestens den aus der Stichprobe berechneten Wert $T(x)$ annimmt:

$$p^* = P_{H_0}\left(\, T \geq T(x_1, \ldots, x_n) \,\right) \tag{A.3}$$

Gilt nun $p^* = P_{H_0}(T \geq T(x_1, \ldots, x_n)) < \alpha$, so folgt aus einem Vergleich von (A.2) mit (A.3)[1], dass $T(x_1, \ldots, x_n) > c$ gelten muss; also wird H_0 verworfen.

Ist dagegen $P_{H_0}(T > T(x_1, \ldots, x_n)) > \alpha$, so muss notwendigerweise $T(x_1, \ldots, x_n) < c$ gelten.

Bei Verwendung des p-Wertes wird also wie folgt entschieden:

$$p^* \leq \alpha \qquad \text{Nullhypothese verwerfen} \tag{A.4}$$

$$p^* > \alpha \qquad \text{Nullhypothese nicht verwerfen} \tag{A.5}$$

Wie man an (A.4) und (A.5) erkennen kann, besitzt der p-Wert auch die folgende Interpretation: p^* ist die kleinste Zahl, die man als Signifikanzniveau wählen kann, so dass der Test (bei den vorliegenden Daten) gerade noch zur Ablehnung von H_0 führt. Gilt also $p^* = 0.03$, so würde die Hypothese H_0 auf dem 10%-Niveau (d.h. für $\alpha = 0.1$), auf dem 5%-Niveau und sogar auf dem 3%-Niveau verworfen, nicht jedoch auf dem 1%-Niveau.

Der Vorteil des p-Wertes besteht darin, dass man daran ablesen kann, wie weit das (vorher) gewählte Signifikanzniveau unter- oder überschritten wird, ob die Nullhypothese also nur sehr knapp oder sehr deutlich widerlegt wird. Dasselbe kann man prinzipiell auch durch explizite Angabe des Wertes $T(x_1, \ldots, x_n)$ erreichen, allerdings benötigt der Leser dann eine Tabelle der Testverteilung, um diesen Wert beurteilen zu können.

[1] Ist Z eine stetige Zufallsvariable (mit streng monoton wachsender Verteilungsfunktion), so gilt

$$a < b \quad \Leftrightarrow \quad P(Z \geq a) > P(Z \geq b)$$

Hochsignifikante Resultate

Intuitiv wird man einen p-Wert $p^* = 0.002$ für „signifikanter“ halten als den Wert $p^* = 0.04$. Nach der eigentlichen Definition des Begriffes ist dies nicht korrekt: nach Wahl des Signifikanzniveaus ist ein Testergebnis entweder signifikant oder nicht.

Dennoch spricht man oft von einem hochsignifikanten Resultat, wenn der p-Wert kleiner als 0.01 ist. Viele Computerprogramme (auch viele R-Funktionen) vergeben Signifikanz-Sterne: Gilt $p^* < 0.001$, so wird der p-Wert mit drei Sternen versehen (extrem signifikant), ein p-Wert zwischen 0.001 und 0.01 erhält zwei Sterne (hoch signifikant), ein p-Wert zwischen 0.01 und 0.05 erhält noch einen Stern (signifikant).

Selbst wenn der p-Wert zwischen 0.05 und 0.1 liegt, wird er in R noch mit einem Punkt gekennzeichnet.

A.2 Multiple Tests

In Abschnitt 5.3 wurde ein Problem angesprochen, das auftritt, wenn man (nach einer Varianzanalyse) paarweise Tests zwischen den einzelnen Gruppen durchführt: die Wahrscheinlichkeit für einen Fehler 1. Art in mindestens einem der Tests ist größer als das Testniveau α. Dieses Problem nennt man das Problem des multiplen Testens.

Allgemein seien im folgenden m Hypothesen zu testen, man spricht dann von einer Familie von m Tests. Gibt es z.B. in der Situation der Varianzanalyse k Gruppen, und führt man Zweistichproben-Tests zwischen allen möglichen Paaren durch, so ergibt sich eine Familie von $\binom{k}{2} = (k-1)k/2$ Tests; vergleicht man eine Kontrollgruppe mit allen anderen Gruppen, so liegen $k-1$ Tests vor. Weiter sei

$$\begin{aligned} S &= \{\text{mindestens einer der } m \text{ Tests ist signifikant}\}, \\ S_l &= \{\text{der } l\text{-te Test ist signifikant}\} \qquad (l = 1 \ldots, m). \end{aligned}$$

Jeder Test mit Nullhypothese $H_{0,l}$ (auch Elementarhypothese genannt) wird zum Niveau α durchgeführt:

$$P_{H_{0,l}}(S_l) = \alpha \;\text{ für }\; l = 1, \ldots, m$$

(Sprechweise: das lokale Signifikanzniveau ist α).

Die globale Nullhypothese H_0 ist dann: alle m Nullhypothesen $H_{0,l}$ gelten.

Frage: Was ist der Fehler 1. Art für H_0 (sogenanntes globales Signifikanzniveau FWE_w)?

Annahme: $S_1, \ldots, S_k$ seien (stochastisch) unabhängig. Dann folgt

$$\begin{aligned} \mathrm{FWE}_w &= P_{H_0}(S) = P_{H_0}\left(\bigcup_{l=1}^{m} S_l\right) = 1 - P_{H_0}\left(\bigcap_{l=1}^{m} S_l^C\right) \\ &= 1 - \prod_{l=1}^{m} P_{H_{0,l}}\left(S_l^C\right) = 1 - (1-\alpha)^m \approx \alpha m. \end{aligned}$$

Meistens ist die Unabhängigkeitsannahme verletzt (z.B. bei den obigen Beispielen); in jedem Fall gilt die Ungleichung

$$\alpha \le \mathrm{FWE}_w = P_{H_0}\left(\bigcup_{l=1}^{m} S_l\right) \le \sum_{l=1}^{m} P_{H_{0,l}}(S_l) = \alpha m.$$

Daraus folgt eine Methode, um $\mathrm{FWE}_w \leq \alpha$ (ohne irgendeine Annahme) zu garantieren, die **Bonferroni-Korrektur**: alle m Tests werden auf dem Niveau α/m durchgeführt. Dann gilt:

$$\mathrm{FWE}_w \;\leq\; \sum_{l=1}^{m} P_{H_{0,l}}(S_l) \;=\; \sum_{l=1}^{m} \alpha/m = \alpha.$$

Wichtiger als das globale Signifikanzniveau FWE_w ist meist das *multiple* Signifikanzniveau FWE_s.

$\mathrm{FWE}_w \leq \alpha$ bedeutet: die Wahrscheinlichkeit, mindestens eine Elementarhypothese irrtümlich abzulehnen, ist höchstens α, vorausgesetzt, alle Elementarhypothesen sind wahr.

$\mathrm{FWE}_s \leq \alpha$ bedeutet: die Wahrscheinlichkeit, mindestens eine Elementarhypothese irrtümlich abzulehnen, ist höchstens α, unabhängig von der Anzahl wahrer Elementarhypothesen.

Für das multiple Testproblem mit Bonferroni-Korrektur gilt sogar

$$\mathrm{FWE}_s \leq \alpha.$$

Problem: Die Abschätzung ist manchmal recht grob; der Test hat dann eine geringe Güte.

Verfeinerung von Holm: Man führt den ersten Test (dabei darf man den "signifikantesten" auswählen) auf dem Niveau α/m durch; falls dieser signifikant ist, führt man den zweiten Test mit $\alpha/(m-1)$ durch, usw.

Das Verfahren stoppt beim ersten nicht-signifikanten Resultat, alle weiteren Hypothesen können nicht verworfen werden.

Durchführung mit *p*-Werten: Man vergleicht den kleinsten p-Wert mit α/m, ist er kleiner, so vergleicht man den zweitkleinsten mit $\alpha/(m-1)$, usw.

Das Verfahren von Holm besitzt ebenfalls das multiple Niveau α. Ein Nachteil dieses Verfahrens ist, das das Ergebnis eines Tests zwischen zwei Gruppen von anderen Gruppen beeinflusst werden kann.

Die Durchführung der paarweisen t-Tests oder Wilcoxon-Tests mit Bonferroni oder Holm-Korrektur erfolgt in R mit den Funktionen `pairwise.t.test` und `pairwise.wilcox.test`, wobei man jetzt `p.adjust.method="bonferroni"` bzw. `p.adjust.method="holm"` wählt. Dabei werden korrigierte p-Werte ausgegeben: bei der Bonferroni-Korrektur mp, wie schon in Abschnitt 5.3 gezeigt wurde, bei der Korrektur nach Holm entsprechend der Größe der p-Werte $mp, (m-1)p, \ldots$.

```
> pairwise.t.test(duenger$ertrag, duenger$sorte, p.adjust.method="holm")
$method
[1] "t tests with pooled SD"
$data.name
[1] "duenger$ertrag and duenger$sorte"
$p.value
          1          2
2 0.2662456         NA
3 0.1734203 0.03129844
$p.adjust.method
[1] "holm"
    ...

> pairwise.wilcox.test(duenger$ertrag, duenger$sorte,
  p.adjust.method="holm")
    ...
$p.value
          1          2
2 0.5476190         NA
3 0.4145997 0.02380952
$p.adjust.method
[1] "holm"
    ...
```

A.2.1 Multiple Tests und simultane Konfidenzintervalle bei der Varianzanalyse

Die Bonferroni- oder Holm-Korrektur sind bei jedem multiplen Testproblem anwendbar. Wegen dieser Allgemeinheit sind sie allerdings unter speziellen Voraussetzungen oft nicht die besten Verfahren. In der Situation der einfachen Varianzanalyse aus Abschnitt 5.1 gibt es neben der schon in 5.3 angesprochenen Tukey-Methode noch weitere gebräuchliche Verfahren, die simultane Konfidenzintervalle für Mittelwertsdifferenzen liefern.

Simultane Konfidenzintervalle bei der Varianzanalyse mit 3 Gruppen: Gesucht sind Konfidenzintervalle I_{12}, I_{13}, I_{23} für $\mu_1 - \mu_2, \mu_1 - \mu_3, \mu_2 - \mu_3$, so dass alle Mittelwertsdifferenzen mit Wahrscheinlichkeit $1 - \alpha$ vom zugehörigen Intervall überdeckt werden.

Zusammenhang zwischen Tests und Konfidenzintervallen: Die Hypothese $\mu_1 = \mu_2$ wird abgelehnt, falls I_{12} den Wert Null nicht überdeckt; analoges gilt für die übrigen Paare.

Gewöhnliche Konfidenzintervalle zur Konfidenzwahrscheinlichkeit $1-\alpha$ ergeben ein multiples Testverfahren, bei dem das *lokale Signifikanzniveau* α ist.

Simultane Konfidenzintervalle zur Konfidenzwahrscheinlichkeit $1-\alpha$ ergeben ein multiples Testverfahren, bei dem das *multiple Signifikanzniveau* α ist.

Im folgenden seien alle Voraussetzungen der einfaktoriellen Varianzanalyse erfüllt; alle möglichen Paare sollen verglichen werden.

Fisher's LSD-Methode (least significant difference):

Zuerst wird eine ANOVA zum Niveau α durchgeführt. Liefert diese ein signifikantes Resultat, so werden paarweise t-Tests (mit gepoolter Varianz MQI) durchgeführt. Die entsprechenden Konfidenzintervalle sind:

$$\mu_i - \mu_j \in \left[\bar{Y}_{i\cdot} - \bar{Y}_{j\cdot} \pm t_{n-k,1-\frac{\alpha}{2}} \left(MQI \left(\frac{1}{n_i} + \frac{1}{n_j} \right) \right)^{1/2} \right]$$

Wegen der zuerst durchgeführten Varianzanalyse gilt $\mathrm{FWE}_w = \alpha$. Man erhält aber keine simultanen Konfidenzintervalle zum Niveau $1-\alpha$; somit gilt $\mathrm{FWE}_s \leq \alpha$ nicht!

Scheffé-Methode:

$$\mu_i - \mu_j \in \left[\bar{Y}_{i\cdot} - \bar{Y}_{j\cdot} \pm \left((k-1) F^{1-\alpha}_{k-1,n-k} \cdot MQI \cdot \left(\frac{1}{n_i} + \frac{1}{n_j} \right) \right)^{1/2} \right]$$

Hier gilt $\mathrm{FWE}_s \leq \alpha$. Der Vorteil der Scheffé-Methode ist, dass sie auch für beliebige lineare Kontraste anwendbar ist: sei $\sum_{i=1}^k c_i = 0$,

$$\sum_{i=1}^k c_i \mu_i \in \left[\sum_{i=1}^k c_i \bar{Y}_{i\cdot} \pm \left((k-1) F^{1-\alpha}_{k-1,n-k} \, MQI \, \sum_{i=1}^k \frac{c_i^2}{n_i} \right)^{1/2} \right].$$

Mit Wahrscheinlichkeit größer oder gleich $1-\alpha$ sind dann diese Intervalle simultan korrekt für **alle** möglichen $c = (c_1, \ldots, c_k)$ mit $\sum_{i=1}^k c_i = 0$.

Die Mittelwertsdifferenzen $\mu_i - \mu_j$ sind ein Spezialfall mit $c_i = 1, c_j = -1, c_l = 0$, sonst.

Tukey-Methode (im balancierten Fall $n_1 = n_2 = \ldots = n_k$):

$$\mu_i - \mu_j \in \left[\bar{Y}_{i\cdot} - \bar{Y}_{j\cdot} \pm q^{1-\alpha}_{k,n-k} \, (MQI/n)^{1/2} \right]$$

$q^{\gamma}_{n,\nu}$ ist das γ-Quantil der studentisierten Spannweite mit Parametern n und ν. Hier ist $\text{FWE}_s = \alpha$; die Tukey-Methode ist in diesem Fall besser als die Scheffé-Methode, d.h. sie liefert kürzere Konfidenzintervalle.

Im nichtbalancierten Fall: **Tukey-Kramer-Intervalle:**

$$\mu_i - \mu_j \in \left[\bar{Y}_{i\cdot} - \bar{Y}_{j\cdot} \pm q^{1-\alpha}_{k,n-k} \left(MQI \, \frac{1}{2} \left(\frac{1}{n_i} + \frac{1}{n_j} \right) \right)^{1/2} \right]$$

Hier gilt $\text{FWE}_s \leq \alpha$, die Methode ist besser als die Scheffé-Methode.

Bemerkungen:

- Vergleich einer Kontrollgruppe gegen andere Gruppen: Methode von Dunett.
- Es gibt Erweiterungen der Verfahren auf den Fall ungleicher Varianzen, sie werden allerdings selten verwendet.
- Außer paarweisen Wilcoxon-Tests mit Bonferroni- oder Holm-Korrektur sind keine weiteren nichtparametrischen Verfahren verbreitet.

In den Standard-Paket von R steht die Methode von Tukey zur Verfügung (siehe Abschnitt 5.3). Weitere Verfahren auch für viel allgemeinere Situationen sind im multcomp-Paket in den Funktionen `simint` und `simtest` verfügbar. In S-Plus sind viele Verfahren in der Prozedur `multicomp` implementiert.

B Ergänzungen zu R

B.1 Der Start im Poolraum RZ 114

Bevor R an den Rechnern in den Poolräumen des Rechenzentrums benutzt werden kann, sollten die folgenden zwei Schritte durchgeführt werden:

1. Prüfen Sie, ob das Programm R im Windows-Startmenü vorhanden ist. Falls nicht, führen Sie einen Neustart des Rechners durch.
2. Erstellen Sie auf dem Desktop eine Verknüpfung mit R. Zu diesem Zweck können Sie folgendermaßen vorgehen:

   Klicken Sie mit der rechten Maus auf den Desktop. Wählen Sie Neu|Verknüpfung. Es öffnet sich ein Fenster. Geben Sie dort die folgende Zeile ein[1]:

   ```
   C:\Programme\R\R-2.7.0\bin\Rgui.exe
   ```

   Klicken Sie auf **weiter**. Im nächsten Fenster geben Sie `Rgui` ein und klicken auf **Fertigstellen**. Nun finden Sie ein Icon mit dem Namen Rgui auf ihrem Bildschirm.

   Klicken Sie mit der rechten Maus auf das Icon Rgui und wählen Sie **Eigenschaften**. Geben Sie dort unter **ausführen in** das Verzeichnis `Z:` ein; dann klicken Sie auf OK.

Jetzt können Sie R durch Doppelklick auf das Icon starten.

Bemerkungen:

1. Durch den zweiten Schritt wird dafür gesorgt, dass R den Workspace (siehe Abschnitt 2.2) im Verzeichnis `Z:` abspeichert; dies ist nötig, da Benutzer keinen Zugriff auf das Laufwerk `C:` haben. Statt `Z:` kann auch ein (bereits existierendes) Unterverzeichnis, etwa `Z:\statistik`, angegeben werden.

   Dieses Vorgehen ist auch allgemein sehr nützlich, um Daten und Objekte, die zu verschiedenen Projekten gehören, zu trennen. Man erstellt einfach für jedes Projekt

[1] Hierbei muss R-2.7.0 durch die aktuell installierte Version ersetzt werden

ein eigenes Icon wie oben beschrieben und gibt unter **ausführen in** jeweils ein zum Projekt gehörendes Verzeichnis an.

2. Für R existieren verschiedene länderspezifische Lokalisierungen; das bedeutet, das Menüeinträge und Fehlermeldungen in Landessprache ausgegeben und landesspezifische Datums- und Währungsangaben verwendet werden. Die Wahl der Lokalisierung erfolgt automatisch; wird R etwa auf einem Computer mit deutschem Windows-Betriebssystem installiert, so wird die deutsche Lokalisierung verwendet.

   Läuft der Computer mit einer englischen (bzw. US-amerikanischen) Version des Betriebssystems, so wird für R entsprechend die englische Lokalisierung verwendet. Möchte man dennoch die deutsche Lokalisierung verwenden, so geht man am einfachsten folgendermaßen vor:

   Bei der Installation von R wird ein Unterverzeichnis `etc` angelegt. In diesem Verzeichnis erstellt man eine Datei `Renviron.site`, in die man den Befehl `LC_ALL=de` einträgt. Beim nächsten Start von R wird dann die deutsche Lokalisierung verwendet.

B.2 Verschiedene Arten der Befehlseingabe

B.2.1 Direkte Befehlseingabe an der R-Console

Gibt man Befehle direkt an der Console ein, so kann man sich viel Tipp-Arbeit durch die Vervollständigungsfunktion und die **R history** ersparen. In letzterer sind alle in einer Sitzung ausgeführten Befehle gespeichert. Wie man auf die letzten Befehle zugreifen und diese dann editieren kann, zeigt Tabelle B.1.

Taste	Beschreibung
↑	der letzte Befehl wird wiederholt; bei mehrfachem Drücken bewegt man sich in der Liste ausgeführter Befehle rückwärts
↓	man bewegt sich in der Liste ausgeführter Befehle vorwärts
←	bewegt Cursor nach links
→	bewegt Cursor nach rechts
Pos1	Sprung zum Anfang der Befehlszeile
Ende	Sprung zum Ende der Befehlszeile

Tabelle B.1: Tastaturbefehle zum Editieren der Kommandozeile

Die Vervollständigungsfunktion listet bei Drücken der Tabulatortaste alle Objekte (z.B. Funktionen) mit gleichem Anfang auf und vervollständig eindeutige Objektnamen. Bei Funktionen listet sie alle Parameter auf.

```
> prop # eingeben, und 2 mal Tabulatortaste drücken
prop            prop.table      prop.test       prop.trend.test
> prop
> prop.te # Befehl ergänzen, bis eindeutig, dann Tabulatortaste drücken
> prop.test # Klammer auf ergänzen, dann 2 mal Tabulatortaste drücken
prop.test(
x=     n=     p=     alternative=  conf.level=   correct=
> prop.test(
```

B.2.2 Befehlseingabe mittels Skriptfiles

Die Befehlseingabe direkt in der R-Console ist meist nur für wenige kurze Befehle sinnvoll. Für größere statistische Analysen ist es besser, die Befehle in eine Textdatei zu

schreiben und sie von dort an die R-Console zu übertragen. Die Datei kann editiert und gespeichert werden, so dass am Ende die vollständige Datenauswertung vorliegt.

Im Datei-Menü klickt man auf **Neues Skript**; es öffnet sich ein Fenster, in dem Befehle eingegeben, editiert und gespeichert werden können. Ein bereits vorhandenes Skriptfile kann ebenfalls über das Datei-Menü geöffnet werden.

Mit der Taste F5 kann man die aktuelle Zeile bzw. den markierten Bereich ausführen lassen. Durch den Eintrag **Alles ausführen** im Bearbeiten-Menü werden alle Befehle im Skriptfile der Reihe nach ausgeführt.

B.3 Funktionen in R

Die Deklaration einer Funktion erfolgt durch

```
fname = function(<Formale Parameterliste>) <Funktionskoerper>
```

Dabei ist `<Funktionskoerper>` ein beliebiger Ausdruck; mehrere Ausdrücke müssen in geschweifte Klammern eingeschlossen werden.

`<Formale Parameterliste>` ist eine Liste von formalen Parametern, die durch Komma getrennt werden. Die (benannten) Parameter können mit Default-Werten vorbelegt werden. Die Parameterliste der Funktion `t.test` ist z.B. folgendermaßen vereinbart:

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

Während `x` keinen Default-Wert besitzt, ist `y` mit dem leeren Objekt `NULL` vorbelegt.

Das spezielle formale Argument `...` kennzeichnet eine variable Parameterliste. Dadurch kann eine beliebige Anzahl von Argumenten übergeben werden. So berechnet etwa die Funktion `max(..., na.rm=FALSE)` das Maximum aller übergebenen Werte:

```
> max( 1:5, seq(2.3,6,3))
[1] 5.3
```

Der Wert, der von einer Funktion zurückgegeben wird, ist normalerweise der Wert des letzten Ausdrucks im Funktionskörper. Alternativ kann mit `return()` die Funktionsauswertung an einer beliebigen Stelle abgebrochen und ein Wert zurückzugeben werden.

Der Funktionsaufruf erfolgt in der Form

```
fname(<Aktuelle Parameterliste>)
```

wobei zwei Konventionen für die Zuordnung von aktuellen und formalen Parametern existieren:

1. Werte werden der Position nach zugeordnet.
2. Namen können benutzt werden, um Werte gezielt zuzuordnen.

Diese beiden Konventionen können auch gemischt werden. Beispielsweise sind die folgenden vier Aufrufe der Funktion `t.test` äquivalent.

```
> t.test(x = x1, y = y1, alternative = "greater", var.equal = TRUE)
> t.test(x1, y1, var.equal = TRUE, alternative = "greater")
> t.test(x1, y1, "greater", , , TRUE)
> t.test(x1, y1, alt = "g", veq = T)
```

Die dritte Variante ist fehlerträchtig und sollte nicht benutzt werden. Der letzte Aufruf zeigt, dass Parameternamen auch abgekürzt werden dürfen (sog. *partial matching*). Speziell in der Funktion `t.test` können auch `"two.sided"`, `"greater"`, `"less"` abgekürzt werden.

Im Allgemeinen ist es notwendig, die Hilfe aufzurufen, um die Parameternamen und die Default-Werte zu finden. Manchmal reicht auch ein Aufruf der `args()`-Funktion:

```
> args(cor)
function (x, y = NULL, use = "all.obs", method = c("pearson",
    "kendall", "spearman"))
NULL
> args(hist)
function (x, ...)
NULL
```

Beim Funktionsaufruf wird nur der Wert des aktuellen Parameters übergeben. Diese Übergabeart für Parameter einer Funktion heißt *Call by value*. Auch Zuweisungen innerhalb einer Funktion sind nur lokal gültig. Wird eine neue Variable oder eine neue Funktion innerhalb einer Funktion erzeugt, so sind diese nur innerhalb der Funktion sichtbar.

```
> x = 1; y = 2
> test.function = function(x) {y = 4; u = 5; x}
> z = test.function(3)
> c(x,y,z); u
[1] 1 2 3
Error: Object "u" not found
```

Der rekursive Aufruf einer Funktion ist erlaubt:

```
> fakul = function(n) {if (identical(1,n)) 1 else  n*fakul(n-1)}
> fakul(5)
[1] 120
```

Mehr Informationen über Funktionen in R enthält das Buch Venables, W.N., Ripley, B.D. S Programming. Springer, 2000.

B.4 Arbeiten mit Dataframes

In diesem Abschnitt werden am Beispiel des Datensatzes `vitcap2` aus dem Packet `ISwR` einige Befehle eingeführt, die man im Umgang mit Daten häufig benötigt. Der Datensatz `vitcap2` enthält Messungen der Vitalkapazität von Arbeitern in der Cadmium-Industrie, die unterschiedlich lang dem Schwermetall Cadmium ausgesetzt waren (vgl. Aufg. 6 in Kapitel 7).

```
> library(ISwR); data(vitcap2)
> summary(vitcap2)
     group            age        vital.capacity
 Min.   :1.000   Min.   :18.00   Min.   :2.700
 1st Qu.:2.000   1st Qu.:32.00   1st Qu.:3.935
 Median :3.000   Median :41.00   Median :4.530
 Mean   :2.381   Mean   :40.55   Mean   :4.392
 3rd Qu.:3.000   3rd Qu.:48.00   3rd Qu.:4.947
 Max.   :3.000   Max.   :65.00   Max.   :5.860
```

Einen R-Datensatz, also einen Dataframe, kann man mit Hilfe des `write.table`-Befehls in einer Textdatei abspeichern. Standardmäßig werden Leerzeichen als Trennzeichen verwendet. Ist das Verzeichnis noch nicht vorhanden, so muss es zuerst angelegt werden. Die Option `row.names = FALSE` unterdrückt das Abspeichern von Zeilennahmen.

```
> dir.create("c:\\daten")
> write.table(vitcap2, "c:\\daten\\vitcap2.txt", row.names = FALSE)
```

Hinweis: Durch die Option `sep = ","` erstellt man eine durch Kommas getrennte ASCII-Datei (csv-Datei). In deutschen Excel-Versionen wird als Trennzeichen bei einer csv-Datei allerdings das Semikolon verwendet. Der entsprechende Befehl lautet also

```
> write.table(vitcap2, "c:\\daten\\vitcap2.csv", row.names = FALSE,
    sep = ";")
```

Im Folgenden wird der Datensatz mit den Spaltennamen unter dem Namen `vc2` wieder eingelesen. Das numerische Merkmal `group` wird in eine Faktor-Größe mit den drei Ausprägungen (Levels) 1,2 und 3 umgewandelt (dazu wird eine Spalte einfach durch eine andere ersetzt). Danach werden dem Dataframe `vc2` drei neue Spalten hinzugefügt.

Die erste enthält das Alter in Monaten. Die zweite enthält nochmals die Gruppenzugehörigkeit; allerdings werden die Levels mit dem `level`-Befehl in `"E>10"` (länger als 10 Jahre exponiert), `"E<10"` (weniger als 10 Jahre exponiert) und `"Not"` (nicht exponiert) geändert. Die dritte Spalte enthält die Zugehörigkeit zu einer Altersgruppe (Gruppe 1: zwischen 18 und 35 Jahren, Gruppe2: zwischen 36 und 50 Jahren, Gruppe 3: zwischen 51 und 65 Jahren). Eine derartige Klassenbildung kann mit der Funktion `cut` vorgenommen werden.

```
> vc2 = read.table("c:\\daten\\vitcap2.txt", header = TRUE)
> vc2$group = as.factor(vc2$group)
> # 3 Spalten an Datensatz anfuegen:
> vc2$age2 = vc2$age * 12
> vc2$group2 = vc2$group
> levels(vc2$group2) = c("E>10", "E<10", "Not")
> vc2$age.grouped = cut(vc2$age, breaks = c(17,35,50,65),
    labels = c("age1", "age2", "age3"))
> summary(vc2[, c(1,4:6)])
 group       age2         group2   age.grouped
 1:12   Min.   :216.0   E>10:12   age1:28
 2:28   1st Qu.:384.0   E<10:28   age2:41
 3:44   Median :492.0   Not :44   age3:15
        Mean   :486.6
        3rd Qu.:576.0
        Max.   :780.0
```

Spalten eines Dataframes kann man streichen, indem man den Spaltenindex mit einem Minuszeichen versieht. Will man eine Spalte an einer bestimmtem Position einfügen, so müssen die verschiedenen Spaltenblöcke mit dem `cbind`-Befehl neu positioniert und dabei die Spaltennamen teilweise neu definiert werden. Dies ist etwas umständlich, meistens aber auch unnötig.

```
> vc2 = vc2[ ,-4]
> vc2 = cbind(vc2[,1:2], age.grouped = vc2[,5], vc2[,3:4])
> names(vc2)
[1] "group"  "age"  "age.grouped"  "vital.capacity"  "group2"
```

Einen Teildatensatz erhält man, indem man bestimmte Zeilen auswählt. Meist verwendet man dazu logische Abfragen. Im folgenden Beispiel werden diejenigen Personen aus-

gewählt, die zu den Gruppen "E>10" oder "E<10" gehören, und im Dataframe vc abgespeichert. Allerdings hat das Merkmal group2 auch in vc immer noch drei Level; es gibt aber keine Person mit dem dritten Level "Not". Dies ist oft unerwünscht; Faktor-Level, die in einem Datensatz nicht vorkommen, kann man mit dem level-Befehl und der Option exclude=NULL entfernen.

```
> vc = vc2[vc2$group2 == "E>10" | vc2$group2 == "E<10", ]
> levels(vc$group2)
[1] "E>10" "E<10" "Not"
> summary(vc$group2)
E>10 E<10  Not
  12   28    0
> vc$group = factor(vc$group, exclude=NULL)
> vc$group2 = factor(vc$group2, exclude=NULL)
> levels(vc$group2)
[1] "E>10" "E<10"
```

Teilspalten wählt man ebenfalls durch logische Abfragen aus. Alternativ kann man oft auch den Befehl split(x, g) verwenden, die den Vektor x in Gruppen aufteilt, die durch den Faktor g definiert werden. Die Teilvektoren werden als Liste abgespeichert, auf deren Listenelemente man einzeln zugreifen kann. Die split-Funktion kann auch auf Dataframes angewendet werden.

```
> age1 = vc$age[vc2$group2 == "E>10"]
> age2 = vc$age[vc2$group2 == "E<10"]
> t.test(age1, age2)$p.value
[1] 0.001050193
> # alternativ: mit split-Funktion
> age.split = split(vc$age, vc$group2)
> age.split$"E>10"
 [1] 39 40 41 41 45 49 52 47 61 65 58 59
> t.test(age.split$"E>10", age.split$"E<10")$p.value
[1] 0.001050193
```

Manchmal ist es notwendig, eine Faktorgröße wieder in eine numerische Größe umzuwandeln. Abbildung B.1 zeigt das Ergebnis des folgenden plot-Befehls, der Farbe und Symbol abhängig von der Gruppenzugehörigkeit wählt. Dabei müssen den Parametern pch und col allerdings ganzzahlige Werte übergeben werden.

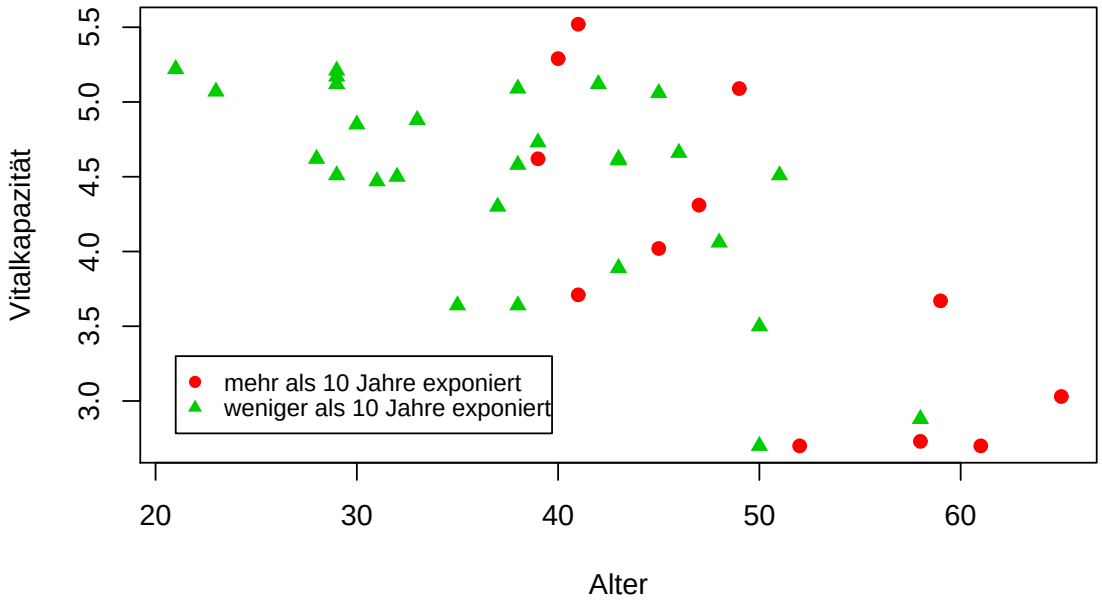


Abbildung B.1: Plot zum Datensatz `vc`

```
> plot(vital.capacity ~ age, pch = as.integer(group2)+15,
    col = as.integer(group2)+1, xlab = "Alter",
    ylab = "Vitalkapazität", data=vc)
> legend(21, 3.3, legend = c("mehr als 10 Jahre exponiert",
    "weniger als 10 Jahre exponiert"), col=2:3, pch=16:17, cex=0.8)
```

B.5 Der Datenaustausch mit Excel

In diesem Abschnitt soll ausführlich auf den Datenaustausch mit einer deutschen Version[2] von Microsoft Excel eingegangen werden; in dieser Version wird ein Komma als Dezimaltrennzeichen verwendet. Dazu werden drei verschiedene Wege beschrieben: der Datenaustausch über die Zwischenablage, der Datenaustausch über eine CSV-Datei, und der Datenaustausch von Dateien im Excel-Format (Endung `xls`) über eine Datenbankverbindung.

B.5.1 Der Datenaustausch über die Zwischenablage

Dies ist der schnellste Weg, aber nur für kleine Datensätze empfehlenswert. Nehmen wir an, der folgende Datensatz mit zwei Merkmalen ist in Excel eingegeben worden:

Groesse	Gewicht	Geschlecht
1,72	68,3	w
1,84	81,8	m

In der ersten Zeile stehen also die Namen der einzelnen Merkmale, wobei darauf geachtet werden muss, dass keine Umlaute und kein Eszett in den Namen vorkommen.

Nun markiert man diese 9 Zellen in Excel mit der Maus, kopiert sie (mit **Strg-C**) in die Zwischenablage, und liest sie in R ein mittels des `read.delim2`-Befehls:

```
> daten1 = read.delim2("clipboard")
> daten1
  Groesse Gewicht Geschlecht
1    1.72    68.3          w
2    1.84    81.8          m
```

Das Merkmal `Geschlecht` wird von R als Faktor interpretiert, da die Werte vom Typ `character` sind.

Dasselbe Ergebnis erreicht man auch mit dem `read.table`-Befehl, muss dabei allerdings einige Parameter richtig setzen:

```
> daten1 = read.table("clipboard", header=TRUE, dec=",", sep="\t")
```

[2] oder allgemeiner einer Version, die den in der ISO-Norm 31 festgelegten Standard einhält

Der optionale Parameter `header=TRUE` legt fest, dass die erste Zeile die Spaltennamen enthält; mit `dec=","` wird das Komma als Dezimalpunkt vereinbart; und schließlich wird durch `sep="\t"` das für Excel zutreffende Trennzeichen festgelegt. Beim `read.delim2`-Befehl sind diese Werte dagegen schon richtig gesetzt.

Um einen in R vorhandenen Datensatz in die Zwischenablage zu kopieren, muss der `write.table`-Befehl verwendet werden, da kein Gegenstück zum `read.delim2`-Befehl existiert. Man führt in R den Befehl

```
write.table(daten1, "clipboard", dec=",", sep="\t", row.names=FALSE)
```

aus, öffnet ein Datenblatt in Excel und fügt (mit **Strg-V**) den Datensatz `daten1` aus der Zwischenablage in Excel ein. Der Parameter `row.names=FALSE` ist nötig, da der Datensatz keine Zeilennamen enthält.

B.5.2 Der Datenaustausch über eine CSV-Datei

Dazu speichert man die obige Datei in Excel als CSV-Datei[3] („Trennzeichen-getrennt“, mit der Endung `csv`), zum Beispiel unter dem Namen `daten1.csv` im Verzeichnis `c:\daten`. Diese Datei kann dann in R durch

```
> daten1 = read.csv2("c:/daten/daten1.csv")
```

oder auch durch

```
> daten1 = read.table("c:/daten/daten1.csv", header=TRUE,
+  dec=",", sep=";")
```

eingelesen werden.

Umgekehrt kann der R-Datensatz `daten1` durch

```
> write.csv2(daten1, "c:/daten/daten2.csv")
```

in eine Textdatei exportiert und in Excel direkt wieder eingelesen werden.

[3] CSV steht für „comma separated values“, als Trennzeichen wird allerdings ein Strichpunkt verwendet, da das Komma schon als Dezimaltrennzeichen benutzt wird

B.5.3 Der Datenaustausch über eine Datenbankverbindung

Will man direkt eine Exceldatei einlesen, so muss man eine Datenbankverbindung mithilfe des `RODBC`-Pakets öffnen; dies geschieht mit dem Befehl `channel()`. Durch den `sqlFetch()`-Befehl kann dann der Excel-Datensatz eingelesen werden; danach sollte man die Verbindung wieder schließen. Im Folgenden nehmen wir an, dass der Datensatz aus B.5.1 als Excel-Datei unter `c:/daten/daten1.xls` gepeichert worden ist.

```
> library(RODBC) # Paket RODBC laden
> channel = odbcConnectExcel("c:/daten/daten1.xls") # Verbindung öffnen
> daten3 = sqlFetch(channel, "Tabelle1")
> daten3
  Groesse Gewicht Geschlecht
1    1.72    68.3          w
2    1.84    81.8          m
> close(channel)
```

Auf ähnliche Weise kann man auch auf Microsoft Access und weitere Datenbanken zugreifen; außerdem können komplexe Datenbankabfragen durchgeführt werden. Für Einzelheiten sei auf die Hilfe von R verwiesen, wo man unter dem Eintrag **R Data Import/Export** nähere Informationen findet.

Das Speichern von R-Datensätzen direkt als Excel-Files ist leider nicht möglich.

B.5.4 Informationen über Objekte erhalten

Um Informationen über R-Objekte zu erhalten, haben wir meist die `summary`-Funktion verwendet. Ebenfalls sehr nützlich für diesen Zweck sind die Funktionen `names()`, die alle Namen eines Objekts auflistet, und `str()`. Die `str`-Funktion versucht, zu jedem R-Objekt Informationen über die interne Struktur zu liefern, und diese möglichst kompakt darzustellen.

```
> liste = list(a=TRUE, b="Liste", c=5:8)
> summary(liste)
  Length Class  Mode
a 1      -none- logical
b 1      -none- character
c 4      -none- numeric
> names(liste)
[1] "a" "b" "c"
> str(liste)
List of 3
 $ a: logi TRUE
 $ b: chr "Liste"
 $ c: int [1:4] 5 6 7 8
```

Für obige Liste sind die Informationen, die man durch `summary` bzw. `str()` erhält, sehr ähnlich. Im folgenden Beispiel verwenden wir aus Beispiel 5.1.1 den Datensatz `duenger` und das Objekt `duenger.aov`, das die Ergebnisse der Varianzanalyse enthält. Die `summary`-Funktion wurde in Beispiel 5.1.1 auf beide Objekte angewandt. Die anderen Funktionen ergeben hier die folgende Ausgabe.

```
> names(duenger)
[1] "sorte"  "ertrag"
> str(duenger)
'data.frame':   15 obs. of  2 variables:
 $ sorte : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 2 2 2 2 2 ...
 $ ertrag: num  8.3 9.3 10.2 8.8 10.7 8.7 8.9 9.4 8.6 9.2 ...
> names(duenger.aov)
 [1] "coefficients" "residuals" "effects"     "rank"      "fitted.values"
 [6] "assign"       "qr"        "df.residual" "contrasts" "xlevels"
```

```
[11] "call"         "terms"     "model"
> str(duenger.aov)
List of 13
 $ coefficients : Named num [1:3]  9.46 -0.50  0.80
  ..- attr(*, "names")= chr [1:3] "(Intercept)" "sorte2" "sorte3"
 $ residuals    : Named num [1:15] -1.16 -0.16  0.74 -0.66  1.24 ...
  ..- attr(*, "names")= chr [1:15] "1" "2" "3" "4" ...
    (weitere 11 Elemente weggelassen)
```

Man sieht insbesondere, dass das Objekt `duenger.aov` ganz andere Elemente enthält, als durch die `summary`-Funktion ausgegeben werden.

An solchen komplizierteren Objekten zeigt sich die Nützlichkeit der `str`-Funktion. Offensichtlich werden durch `summary(duenger.aov)` aus den in `duenger.aov` enthaltenen 13 Elementen neue Größen berechnet und ausgegeben, unter anderem der Wert der F-Statistik. Aber wie kann man darauf weiter zugreifen? Die Funktionen `summary` und `names` helfen hier nicht weiter, im Gegensatz zu `str`!

```
> duenger.summary = summary(duenger.aov)
> summary( duenger.summary )
     Length Class Mode
[1,] 5      anova list
> names(duenger.summary)
NULL
> str(duenger.summary)
List of 1
 $ :Classes 'anova' and 'data.frame':   2 obs. of  5 variables:
  ..$ Df     : num [1:2] 2 12
  ..$ Sum Sq : num [1:2] 4.30 5.52
  ..$ Mean Sq: num [1:2] 2.15 0.46
  ..$ F value: num [1:2] 4.68   NA
  ..$ Pr(>F) : num [1:2] 0.0315     NA
 - attr(*, "class")= chr [1:2] "summary.aov" "listof"
```

Das Ergebnis des letzten Befehls sagt aus, dass `duenger.summary` eine Liste mit einem Element, nämlich einem Data Frame ist; der Datensatz enthält die fünf Merkmale `Df`, `Sum Sq`, `Mean Sq`, `F value` und `Pr(>F)`. Auf diese Größen kann man nach den allgemeinen Regeln für Listen bzw. Data frames zugreifen.

```
> duenger.summary[[1]]$Df   # oder: duenger.summary[[1]]$"Df"
[1]  2 12
> duenger.summary[[1]]$Df[2]
[1] 12
> duenger.summary[[1]][2,1]
[1] 12
>
> duenger.summary[[1]]$"F value"
[1] 4.677302       NA
> duenger.summary[[1]]$"F value"[1]
[1] 4.677302
> duenger.summary[[1]][1,4]
[1] 4.677302
```

B.6 Literatur zu R und S-PLUS

Lehrbücher:

- Dalgaard, P. Introductory Statistics with R. Springer, 2002.
- Dolić, D. Statistik mit R. Oldenbourg, 2003.
- Everitt, B., Rabe-Hesketh, S. Analyzing Medical Data Using S-PLUS. Springer, 2001.
- Krause, A. Einführung in S und S-PLUS. Springer, 1997.
- Krause, A., Olson, M. The Basics of S and S-PLUS. Second edition, Springer, 2000.
- Ligges, U. Programmieren mit R. Springer, 2004.
- Maindonald, J., Braun, J. Data Analysis and Graphics Using R – an Example-based Approach. Cambridge University Press, 2003.
- Selvin, S. Modern Applied Biostatistical Methods using S-PLUS. Oxford University Press, 1998.
- Venables, W.N., Ripley, B.D. Modern Applied Statistics with S. Fourth Edition, Springer, 2002.
- Verzani, J. Using R for Introductory Statistics. Chapman & Hall/CRC, 2004.

Sonstiges:

- Venables, W.N., Smith, D.M. and the R Development Core Team. An Introduction to R – Notes on R: A Programming Environment for Data Analysis and Graphics (Version 1.9.0). 2004, PDF-File bei jeder R-Installation oder unter http://cran.r-project.org/ (Eintrag Documentation - Manuals)
- Maindonald, J. Using R for Data Analysis and Graphics – An Introduction. 2001, PDF-File unter http://cran.r-project.org/ (Eintrag Documentation - Contributed)
- Paradis, E. R for Beginners. 2002, PDF-File unter http://cran.r-project.org/ (Eintrag Documentation - Contributed)