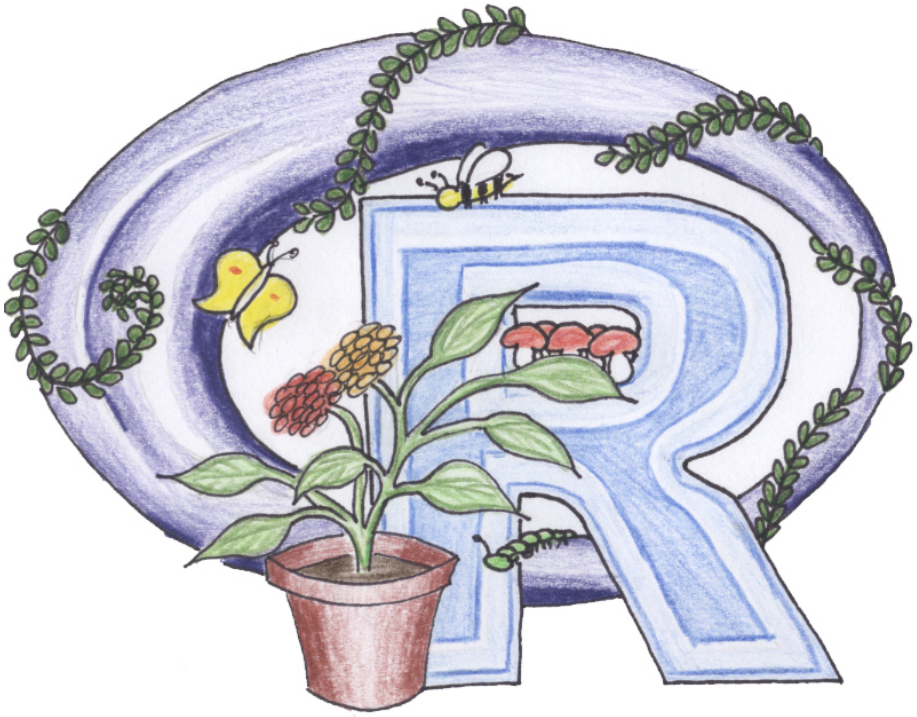
R-Handbuch für Biostatistik

Eine Einführung für Studierende der Gartenbauwissenschaften, Pflanzenbiotechnologie und Biologie

von Katharina Hoff



Eingereicht als Bachelorarbeit am Lehrgebiet für Bioinformatik,
Universität Hannover im September 2005.
Modifiziert an der Universität Greifswald im Januar 2019.

BETREUER

Prof. Dr. L. A. Hothorn, Universität Hannover
Universitetslektor J.-E. Englund, SLU Alnarp

Inhaltsverzeichnis

Kapitel 1

Einführung

1.1 Historisches

1976 begann John Chambers (Bell Laboratoires) damit, eine Programmiersprache namens S zu entwickeln. Die neue Sprache sollte die Programmierung mit Daten ermöglichen. Seitdem wurde S kontinuierlich weiterentwickelt.

Es gibt verschiedene Implementierungen der S-Sprache. Die kommerzielle Implementierung, S-Plus, wird seit Jahren von vielen Wissenschaftlern zur Datenanalyse eingesetzt.

Ross Ihaka und Robert Gentleman von der University of Auckland, Neuseeland, legten den Grundstein zu einer S-ähnlichen Open Source-Implementierung und gaben ihr – auf die Anfangsbuchstaben ihrer Vornamen referierend – den Namen R. R (R Development Core Team, 2004a) steht unter der GNU Public License (Stallman, 1991). Es ist frei verfügbar und darf unter bestimmten Bedingungen[1] verbreitet werden. Auf Grundlage dieser Lizenz wird R ständig von einer weltweiten Gemeinschaft weiterentwickelt und stellt zum heutigen Zeitpunkt ein leistungsfähiges System dar, das den Statistikansprüchen von Gartenbauwissenschaftlern, Biologen und Agraringenieuren weitgehend gerecht wird. Im Unterschied zu S-Plus entfallen die Lizenzgebühren zur Nutzung.

1.2 Aufgabenstellung der Bachelor-Arbeit

Das Thema der Arbeit lautet *Erstellung eines R-Handbuches für Biostatistik*. Es wurde aufgrund des Bedürfnisses nach einem R-Handbuch, das speziell auf die Ansprüche von Gartenbauwissenschaftlern zugeschnitten ist – im weiteren Sinne damit auch auf die der Biologen und Agraringenieure – ausgeschrieben. Die bislang existierenden R-Handbücher sind zwar sehr gute Anleitungen (z.B. Introductory Statistics with R (Dalgaard, 2002)), welche die wichtigsten Funktionen für grundlegende Statistik an allgemeinen Beispielen erläutern. Für einen Gartenbauwissenschaftler sind die Anwendungsbeispiele jedoch sehr abstrakt. Einige, für Feldexperiemente besonders interessante Funktionen neueren Datums, z. B. für multiple Vergleiche, fehlen in anderen Büchern noch.

Die Zielgruppe des Handbuches befindet sich im Grundstudium einer Biowissenschaft. Idealerweise soll das Buch die Grundvorlesung der Statistik begleiten. Mit zahlreichen gartenbau- und landwirtschaftlichen Beispielen wird der Bezug verschiedener R-Funktionen

[1] §1 You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

zur Praxis hergestellt.

1.3 Gründe zur Verwendung von R

R ist im Unterschied zu S-Plus kostenlos und ebenfalls leistungsfähig. Große Teile des in S-Plus geschriebenen Quellcodes laufen in R problemlos. Ein Gartenbauwissenschaftler im Grundstudium kennt S-Plus in den meisten Fällen nicht. Vielmehr ist der typische Gartenbaustudent an die Microsoft Office Suite gewöhnt und fragt sich, warum er zur Auswertung seiner Experimente nicht die mitgelieferte Tabellenkalkulation Excel verwenden soll. Es gibt einige Gründe für den Wechsel:

- R verfügt zwar nicht über eine intuitive Benutzeroberfläche und ist völlig kommandozeilenorientiert, dafür veschafft es dem Benutzer die vollständige Kontrolle. Alle Parameter lassen sich individuell setzen und mit dem eingebauten Hilfesystem ist es einfach, den Überblick über die vorhandenen Parameter zu behalten.
- Die Ausgabe eines statistischen Tests in R ist meist sehr viel umfassender, als die eines Office-Programmes. Konfidenzintervalle, Quantile ect. werden oft mitberechnet.
- R ist bei großen Datenmengen und komplizierten Befehlen (z.B. verschachtelte Funktionen) leistungsfähiger, als Office-Programme.
- Zur Verwendung von R zur Datenauswertung ist die Kenntnis mathematischer Formeln für statistische Prozeduren nicht zwingend notwendig.
- R ist eine objektorientierte Programmiersprache. Dies hat viele Vorteile. Zum Beispiel lassen sich mit einem einzigen kurzen Befehl (`plot(object.simint)`) die Konfidenzintervalle aus einem Objekt zeichnen, das den Testoutput von `simint()` beinhaltet.
- R ist plattformunabhängig. Es kann auf Unix, Linux, Windows und MacOS verwendet werden.
- Die Benutzung von R ist nicht schwieriger, als die von einem GUI-basierten Programm. Befehle werden zwar in die Kommandozeile getippt, doch die Befehlsstruktur ist logisch und damit leicht zu erlernen.
- Ein weiterer Vorteil ist die Integration in die Textsatzsprache LaTeX durch die Sweave Tools. LaTeX erfreut sich in wissenschaftlichen Kreisen durch die klare Strukturierung immer größerer Beliebtheit. Zusammen bieten LaTeX und R eine umfassende Arbeitsplattform, die alle Werkzeuge zum Auswerten und Publizieren wissenschaftlicher Experimente enthält (Gentleman, 2005).
- R kommuniziert zum Datenimport mit Microsoft Excel-Datenblättern (`RODBC` Paket). Das Paket `foreign` unterstützt darüber hinaus die Nutzung von Daten, die durch Statistikprogramme wie Minitab, S, SAS, SPSS, Stata ect. erzeugt wurden.

GUI bedeutet *Graphical User Interface.*

Diese Argumente sollten zur Benutzung von R überzeugen.

1.4 Download und Installation

Installationsfertige Pakete stehen für die Betriebssysteme Linux, Windows und Mac OS zur Verfügung. Hinweise zur Selbstkompilierung des Source Codes sowie der Installation auf Unix, Windows und Mac OS Plattformen gibt *R Installation and Administration* (R Development Core Team, 2004b).

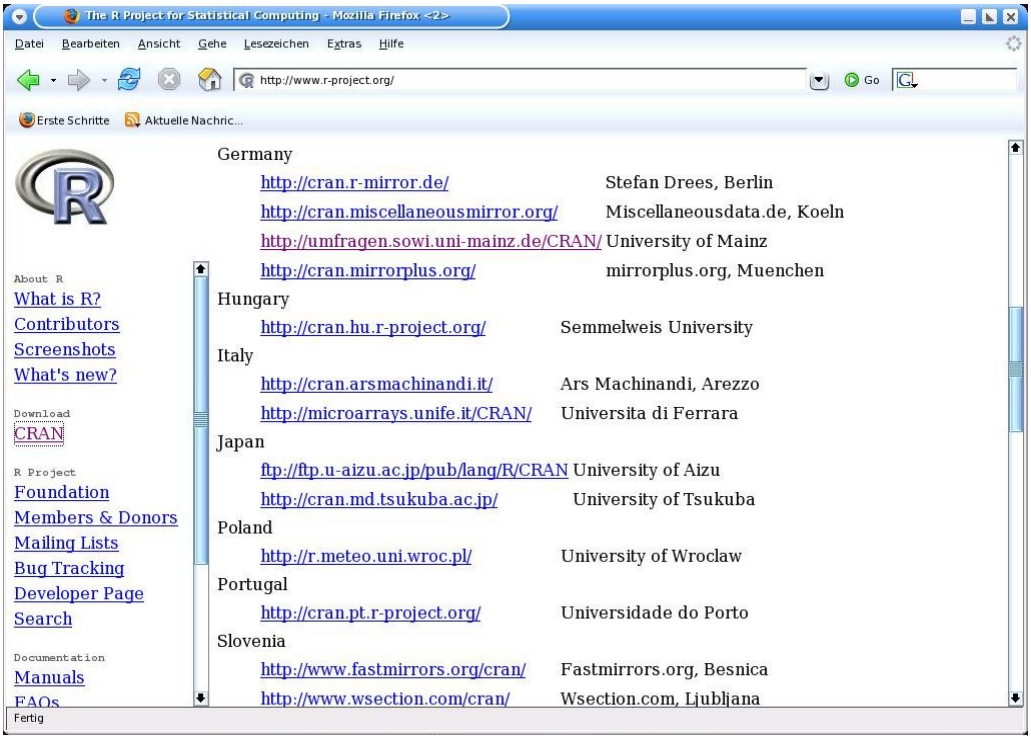


Abbildung 1.1: Auswahl eines nahe gelegenen Mirrors mit CRAN

1.4.1 Download

Wenn Sie mit der Installation von Programmen nicht sehr vertraut sind, merken Sie sich auf jeden Fall das Verzeichnis, in welchem Sie die *.exe-Datei bzw. das *.rpm-Paket gespeichert haben!

Auf der Website http://www.R-project.org unter dem Link CRAN (Comprehensive R Archive Network) steht R zum Download bereit. Um die Übertragungszeit zu minimieren, empfiehlt es sich, einen nahe gelegenen Mirror zu wählen (Abbildung 1.1). Passend zu Ihrem Betriebssystem laden Sie sich von dort die neueste Version des Basispaketes in Form einer *.exe-Datei für Windows oder eines *.rpm-Paketes für rpm-unterstützende Linux-Systeme in ein Verzeichnis ihres lokalen Computers.

1.4.2 Installation unter Windows

Die Installation lässt sich mit einem Doppelklick auf die heruntergeladene *.exe-Datei starten. Der Installationswizard fragt, in welches Verzeichnis R installiert werden soll. Im nächsten Schritt stehen verschiedene R-Komponenten zur Verfügung (Abbildung 1.2). Für die meisten Nutzer ist hier eine Übernahme der Default-Konfiguration ausreichend. Im Verlauf der weiteren Installation wird abgefragt, in welchem Ordner des Startmenüs das Verknüpfungs-Icon für R erstellt werden soll, ob ein Desktop-Icon erwünscht ist, und welche Registry-Einträge erstellt werden dürfen. Wählen Sie die Ihnen zusagende Konfiguration und klicken Sie auf Next >. R wird nun auf Ihrem Computer installiert. Anschließend kann das Programm mit einem Klick auf das Desktop-Icon, die Verknüpfung im Start-Menü oder mit einem Doppelklick auf die Datei `R/bin/Rgui.exe` geöffnet werden. Beenden lässt sich R entweder über **File** Unterpunkt **Exit** oder durch das Eintippen von `q()` in der R-Konsole.

Mit **Konsole** wird die Eingabezeile im Programm R bezeichnet.

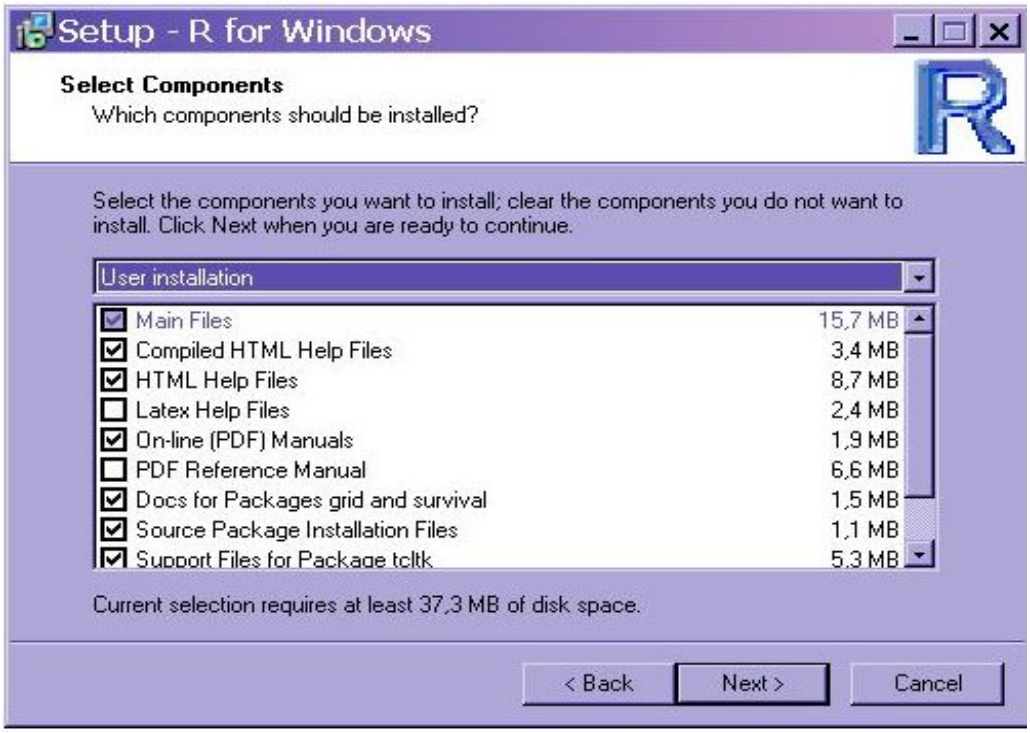


Abbildung 1.2: Komponentenauswahl bei der Installation unter Windows. Die Standardkonfiguration ist für die meisten Benutzer ausreichend.

1.4.2.1 Nachinstallation von Ergänzungspaketen

Das R-Basissystem enthält nicht alle Pakete. Speziell die Nachinstallation der Pakete `pastecs`, `exactRankTests`, `multcomp`, `mvtnorm`, `car`, `rodbc`, `Biobase` (erreichbar unter dem Link http://www.bioconductor.org/repository/release1.5/package/html/index.html) und `multtest` ist zum Lösen der Übungsaufgaben empfehlenswert.

Voraussetzung für die Nachinstallation von Paketen ist eine bestehende Internetverbindung. Die Installation wird durch einen Klick auf den Unterpunkt **Install package(s) from CRAN...** im Menü **Packages** gestartet (Abbildung 1.3). In einem Popup-Fenster wird das gewünschte Paket durch Anklicken ausgewählt und mit **OK** bestätigt. Das entsprechende Archiv wird automatisch heruntergeladen, entpackt und installiert. Anschließend stellt R die Frage: `Delete downloaded files (y/N)?`, welche man mit der Eingabe `y` (yes) beantworten kann. Es werden nicht die installierten Pakete, sondern nur die Quellen gelöscht.

Zur Nutzung der nachinstallierten Pakete werden diese mit `library(Paketname)` eingebunden.

Die **Kommandozeile** (das Terminalfenster) ist unter Linux die **Shell**, ein Terminalprogramm zum Ausführen von Befehlen. Auf der graphischen Oberfläche ist es meist unter einem Muschel-Icon zu finden.

1.4.3 Installation unter Linux

Zur Installation unter Linux ist es notwendig, als **root**[2] eingeloggt zu sein. Eine einfache GUI-basierte Installation ist z.B. bei Suse-Linux durch Anklicken des *.rpm-Pakets im Konquerer mit Yast möglich.

Sollte Ihre Linux-Distribution über keinen graphischen Installationsmanager verfügen,

Eine **Distribution** ist eine von einer Firma herausgegebene Linux-Version. Ein Distributor verkauft üblicherweise den Service, nicht das Programm selbst, welches der GNU Public License unterliegt.

[2] Bei der Installation über ein GUI wird die Eingabe des root-Passwortes automatisch abgefragt, in der Shell wird der Benutzer mit `su root` gewechselt.

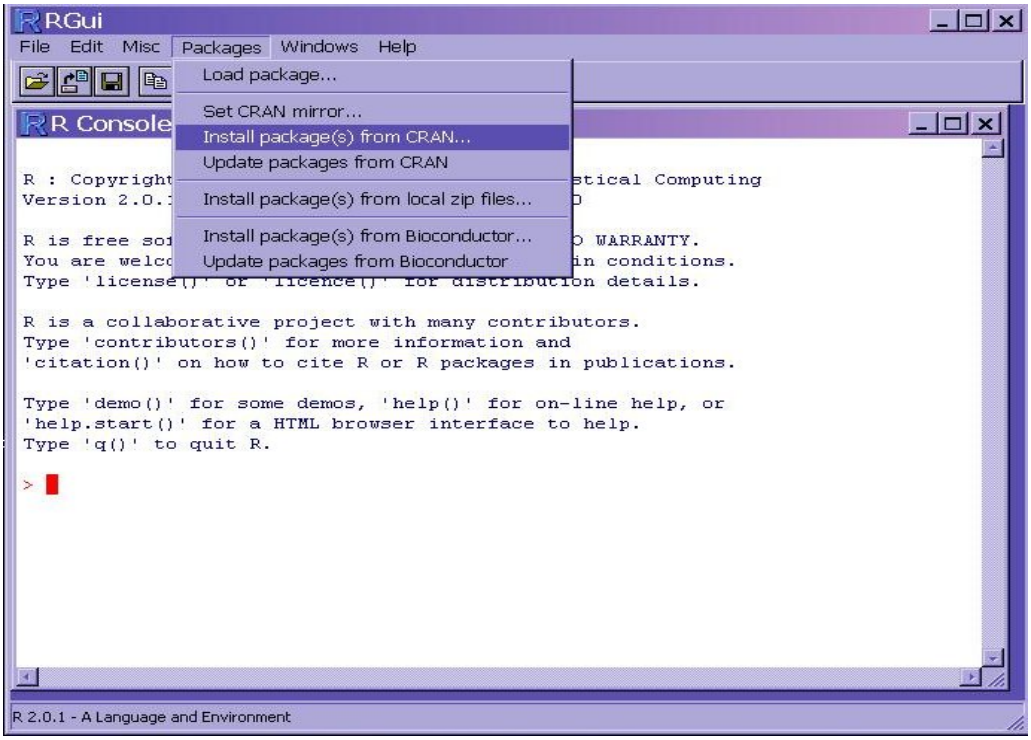


Abbildung 1.3: Nachinstallation von Paketen mit CRAN unter Windows.

kann das *.rpm-Paket auch mit folgendem Befehl in der Kommandozeile installiert werden:

```
rpm -ih /path/to/package/packagename
```

Mit dem Befehl `R` lässt sich R nach erfolgreich abgeschlossener Installation im Terminalfenster öffnen. Das Beenden erfolgt durch Eingabe von `q()` in der R-Konsole (= Terminalfenster, während R läuft). Zur Zeit ist in der Basisinstallation für Linux kein sauber funktionierendes GUI enthalten. Das Paket `gnomeGUI` verspricht eine neue R-Konsole für GNOME, wenn die entsprechenden GNOME-Entwicklungsbibliotheken installiert sind. Ich persönlich war jedoch nicht in der Lage, dieses Paket zu installieren.

1.4.3.1 Nachinstallation von Paketen

Wie schon in Abschnitt 1.4.2.1 erwähnt, sind in der Basisinstallation von R nicht alle Pakete enthalten. Ergänzungen lassen sich über die Kommandozeile nachinstallieren (ein Benutzerwechsel zu **root** ist notwendig).

Nach dem manuellen Download kann das Paket mit folgendem Befehl in der Shell-Kommandozeile (**nicht** im R-Terminal!) installiert werden (R Development Core Team, 2004b):

```
R CMD INSTALL -l /path/to/library /path/to/packagename.tar.gz
```

Der Pfad zur Bibliothek ist systemabhängig und lautet z.B. unter Suse-Linux:

```
/usr/lib/R/library
```

Die Angabe des Pfades zum heruntergeladenen Paket erübrigt sich, wenn Sie sich schon im entsprechenden Verzeichnis befinden[3]. Dann genügt die Angabe des Paketnamens.

Auch die Installation aus der R-Konsole ist bei bestehender Internet-Verbindung möglich. Zunächst wird die Option `CRAN` mit dem folgenden Befehl geändert bzw. gesetzt:

```
> options(CRAN = "http://cran.us.r-project.org")
```

Anschließend wird mit der Funktion

```
> install.packages(Paketname)
```

das entsprechende Paket installiert (R Development Core Team, 2004b).

Vor der Benutzung werden die Pakekte mit dem Befehl `library(Paketname)` eingebunden.

1.4.4 Dokumentationen und Hilfesystem

Durch die Eingabe von `help.start()` in der R-Konsole öffnet sich unter Linux ein Browserfenster, in dem verschiedene Manuals und Dokumentationen aufgelistet sind. Unter Windows öffnen sich die Hilfeseiten im GUI. Normalerweise sind die Handbücher in der R-Installation eingeschlossen. Sollten Sie bei einer benutzerdefinierten Installation fehlen, ist eine bestehende Internetverbindung Voraussetzung.

Die Befehle `?function()` oder `help(function)` rufen die Hilfe zu einzelnen Funktionen auf.

Unter **Linux** öffnet sich die Hilfeseite meist im Terminalfenster. Man navigiert dort mit den Pfeiltasten und kehrt durch Drücken von `q` zur Eingabezeile zurück.

Ist der Name der gesuchten Funktion unbekannt, kann man nach einem Begriff suchen:

```
help.search("search.item")
```

Beispiele zu einer Funktion lassen sich mit `example(function)` aufrufen. Mit `function` lässt sich nach Funktionen des entsprechenden Namens suchen.

1.4.5 Editoren

Ein **Texteditor** ist ein Computerprogramm zur Eingabe, Bearbeitung und Sicherung von einfachem Text. Es ist sinnvoll, bei der Arbeit mit R einen Editor zu benutzen, wenn man nach längerem Arbeiten oder bei der nächsten Sitzung auf bereits benutzte Funktionen unkompliziert und schnell zurückgreifen will.

Bei Nutzung des von Windows standardmäßig mitgelieferten oder eines anderen einfachen Editors werden sowohl die R-Konsole als auch das Schreibprogramm zu Beginn der Arbeit geöffnet. Eine parallele Anordnung auf dem Bildschirm, so dass beide Programme sichtbar sind, ist für die bevorstehende Copy & Paste-Arbeit sinnvoll. Befehle werden nun zunächst im Editor getippt und dann in die R-Konsole kopiert, welche die Ergebnisse errechnet. Das Editor-Dokument wird als .txt-Datei für späteren Wiederaufruf gespeichert (Verzeichnis und Dateiname merken!).

Es gibt viele erweiterte Editoren, deren Funktion über die einfache Texteingabe hinausgeht. Für die Arbeit mit R unter Windows hat sich der Editor WinEdt, erreichbar auf

[3] Verzeichniswechsel mit `cd /path/to/downloaded/package/`

der Website http://www.winedt.com bewährt. Er lässt sich für die Arbeit mit R so anpassen, dass man nur noch auf einen Button klickt, um Quelltext an die R-Maschine zu übergeben. Auch Emacs (erreichbar unter der Internetadresse emacs) bietet diese Möglichkeit und ist sogar plattformunabhängig. Beide Editoren verfügen außerdem über ein farbliches Hervorheben des Quelltextes.

1.5 Grundlagen

Dieser Abschnitt[4] befasst sich mit grundlegender Terminologie und einfachen Befehlen in R. Ein vollständiges Verständnis nach dem ersten Lesen ist nicht erforderlich, die folgenden Kapitel bauen jedoch inhaltlich darauf auf.

1.5.1 Umgang mit der Kommandozeile

Befehle werden in der R-Konsole hinter dem >-Zeichen eingegeben. Die Bestätigung des Befehls erfolgt durch das Drücken der **ENTER** oder **RETURN**-Taste. R verarbeitet die Eingabe und gibt gegebenenfalls das Ergebnis aus. Mit den Pfeiltasten ↑ und ↓ lassen sich vorherige Befehle aufrufen. **POS1** setzt den Cursor an den Anfang, **ENDE** setzt den Cursor an das Ende der Zeile.

Ein **Kommentar** wird verwendet, um den Quelltext für andere und sich selbst verständlich zu machen. Kommentare werden beim Kompilieren ignoriert.

Kommentare werden mit dem Hash-Zeichen (#) gekennzeichnet.

Leerzeichen werden normalerweise ignoriert. `4 + 7` bedeutet bei der Eingabe dasselbe wie `4+7`, allerdings dürfen Leerzeichen nicht innerhalb eines Befehls verwendet werden: `x <- 3` ⇒ drei wird x zugeordnet, aber mit einem Leerzeichen zwischen dem < und - gewinnt es die Bedeutung „x ist kleiner als -3?“.

Zeilenumbruch. Wenn Befehle über mehr als eine Zeile gehen, wird zu Beginn der neuen Zeile ein + dargestellt. Dieses Zeichen muss **nicht** eingetippt werden! Wenn die Eingabe eines Befehls unvollständig erfolgte und die Eingabe-Taste betätigt wurde, erscheint ebenfalls ein +. Der Befehl kann dann vervollständigt werden. Häufig fehlen Klammern.

1.5.2 Taschenrechner, Objekte und Funktionen

R lässt sich wie ein einfacher Taschenrechner zur Addition, Subtraktion, Multiplikation und Division verwenden. Auch Logarithmen ect. können berechnet werden:

```
> 4+7

[1] 11

> log(2)

[1] 0.6931472

> exp(0.6931472)

[1] 2
```

Achtung! `log()` berechnet den natürlichen, nicht den dekadischen Logarithmus!

```
> 30/6 # Bei "geteilt durch" muss man aufpassen. Ein Doppelpunkt
 führt zur Ausgabe der natürlichen Zahlen von 30 bis 6.

[1] 5

> log(-1)

[1] NaN
Warning message:
NaNs produced in: log(x)
```

`NaN` bedeutet „not a number“. Fehlende Werte werden durch `NA` (not available) gekennzeichnet.

In obigen Beispielen schreibt R das Ergebnis in einen **Vektor** (siehe Abschnitt 1.5.4.1), der nur ein einziges Element an der Position [1] enthält. Einen Vektor mit 19 Elementen erhält man beispielsweise beim Ausdruck der Zahlen von 30 bis 6:

```
> 30:6

 [1] 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10  9  8
 +   7  6
```

Um einen solchen Vektor in einem Objekt zu speichern, benutzt man das Symbol <-. Ein Objekt lässt sich durch Eingabe seines Namens aufrufen und in anderen Berechnungen und Funktionen weiterverwenden:

```
> a<-89
> b<-45
> result<-(a+b)^2 # Beispiel zur Verwendung von Objekten in Berechnungen
+ und Funktionen.
> result

[1] 17956
```

Objekte können überschrieben werden. Es erfolgt keine Warnmeldung. Eine eindeutige Bezeichnung von Objekten (z.B. im Extremfall *binom.formel.aus.a.b* anstatt *result*) erleichtert die Verwaltung. Auch Funktionen können ohne Vorwarnung überschrieben werden. Am sichersten ist es, den gewünschten Namen vor der Vergabe in die R-Konsole einzugeben. Falls eine gleichnamige Funktion existiert, wird sie ausgegeben. Weitere Hinweise zur Wahl von Objektnamen:

- Objektnamen dürfen nicht mit einer Zahl beginnen und es ist empfehlenswert, auch nicht mit einem Punkt zu beginnen,
- sie dürfen Punkt (.) und Unterstrich (_) enthalten, aber andere Sonderzeichen wie z.B. ~, @, !, #, %, ^, & sind nicht erlaubt,
- Groß- und Kleinschreibung müssen beachtet werden.

Objekte können mit Funktionen bearbeitet werden. Eine Funktion besteht aus dem Funktionsnamen und den darauf folgenden runden Klammern (), in denen verschiedene Argumente angegeben werden können. Die Funktion `objects()` listet alle vorhandenen Objekte auf. Durch das Argument `pattern` erfolgt die Ausgabe selektiv, d.h.

Eine **Funktion** ist die Implementation einer Methode; sie liefert einen Rückgabewert.

```
> objects(pattern="example")
```

druckt nur die Objekte aus, welche im Namen die Zeichenkette `example` enthalten. Weitere Informationen zur Funktion `objects()` erhält man durch Eingabe von `?objects()`.

Abschnitt 1.4.4 enthält allgemeine Hinweise zum Hilfesystem von R.

Mit der Funktion `rm()` lassen sich Objekte entfernen.

1.5.3 Datentypen

R-Objekte können verschiedene Datentypen enthalten. Wichtig sind hier:

Numeric: Zahlen. Nur mit numerischen Objekten kann gerechnet werden.

Character: Zeichenketten. Diese werden häufig für Gruppenbezeichnungen gewählt.

Logical: Hat die zwei Werte `TRUE` und `FALSE`. Abfragen haben häufig eine logische Ausgabe z.B.

```
> a <- 23
> b <- "Keine Zahl"
> is.numeric(a)

[1] TRUE

> is.numeric(b)

[1] FALSE
```

Factor: Kategoriale Daten, z.B. die Lichter einer Ampel: rot, gelb, grün. Die Ausprägung eines Faktors wird mit dem Begriff *level* bezeichnet. Faktoren können aus numerischen oder character-Objekten generiert werden. Im folgenden Beispiel wird zunächst aus einem Vektor ein Faktor erstellt. Bei Aufruf des Faktors werden der Vektordatensatz sowie die entsprechenden Levels ausgegeben. Man kann die Ausgabe der Levels auch durch die Funktion `levels()` erwirken.

```
> traffic.lights.vector <- c("green", "red", "green", "yellow", "yellow")
> traffic.lights.factor <- factor(x=traffic.lights.vector)
> traffic.lights.factor

[1] green  red    green  yellow yellow
Levels: green red yellow

> levels(traffic.lights.factor)
```

[4] Der Abschnitt *Grundlagen* wurde in Anlehnung an das Übungsskript zur Biometrie 1 (Froemke, 2004) erstellt.

```
[1] "green"  "red"    "yellow"
```

Die Levels sind alphabetisch geordnet. Für manche statistische Prozeduren müssen die Levels jedoch nach Gewicht bzw. Wichtigkeit geordnet werden. Eine Ordnung kann erzwungen werden durch:

```
> affection.factor <- factor(c("none","few","too many", "few",
+ "many","too many"))
> sorted.affection.factor <- ordered(x=affection.factor,
+ levels=c("none","few","many","too many"))
> sorted.affection.factor
```

```
[1] none     few      too many few      many     too many
Levels: none < few < many < too many
```

1.5.4 Datenein- und ausgabe

Daten können in R u.a. in folgenden Strukturen gespeichert werden: Vektor, Matrix, Liste und Data-Frame. Die Ausgabe in der R-Konsole erfolgt bei Aufruf eines Objektes oder als Ergebnis einer Funktion.

1.5.4.1 Vektor

Der Vektor ist eine eindimensionale Datenstruktur, die nur einen Datentyp, z.B. numeric oder character, enthalten kann. Vektoren mit nur einem Element werden durch einfache Zuweisung erstellt (siehe auch Abschnitt 1.5.2):

```
> vec.1 <- "cucumber"
> vec.1
```

```
[1] "cucumber"
```

Zur Erstellung von Vektoren mit mehr als einem Element muss die Funktion `c()` (c = concatenate = engl. verbinden, verknüpfen) verwendet werden. (Sie kann natürlich auch nur ein einziges Element verknüpfen.)

```
> vec.2 <- c(2,3,4,5,6,3.4)
> vec.2
```

```
[1] 2.0 3.0 4.0 5.0 6.0 3.4
```

```
> vec.3 <- c("cauliflower", "cucumber", "tomato")
> vec.3
```

```
[1] "cauliflower" "cucumber"    "tomato"
```

Bei Eingabe mehrerer Datentypen in einen Vektor schreibt R alle in einen gemeinsamen Datentyp um. Im Beispiel verwandelt R die numerischen Einträge bei Auftauchen eines Characters ebenfalls in Daten des Typs character:

```
> vec.4 <- c(1:4,10.5,"flower")
> vec.4
```

```
[1] "1"      "2"      "3"      "4"      "10.5"   "flower"
```

Mit der Funktion `seq()` können regelmäßige Sequenzen generiert werden:

```
> vec.5 <- seq(from =1, to =5, by = 0.5)
> vec.5

[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
```

Die Funktion `rep()` wiederholt Elemente in Vektoren und Listen:

```
> vec.6 <- rep(x=c("A","B","C"), times = 3)
> vec.6

[1] "A" "B" "C" "A" "B" "C" "A" "B" "C"

> vec.7 <- rep(x=c("A","B","C"), each = 3)
> vec.7

[1] "A" "A" "A" "B" "B" "B" "C" "C" "C"
```

An die Positionsnummern von Vektorelementen können Namen vergeben werden. Dabei ist darauf zu achten, dass die Anzahl der Namen mit der Anzahl der Vektorelemente übereinstimmt:

```
> vec.8 <- seq(from=1,to=9,by=2)
> vec.8

[1] 1 3 5 7 9

> names(x=vec.8)<-c("a","b","c","d","e")
> vec.8

a b c d e
1 3 5 7 9
```

Mit `length()` und `mode()` lassen sich die Länge und der Zustand von Vektoren, Matrizen, Listen und Data-Frames ausgeben. Mit der Funktion `sort()` kann ein Vektor der Größe nach oder alphabetisch sortiert werden. Ansteigend ist dabei der Defaultwert, er lässt sich durch Setzen des Argumentes `decreasing = TRUE` ändern.

1.5.4.2 Matrix

Die Matrix hat im Gegensatz zum Vektor zwei Dimensionen, aber auch hier ist nur jeweils ein Datentyp pro Matrix möglich. Eine Matrix kann mit der Funktion `cbind()` (column bind = Spaltenbindung), `rbind()` (row bind = Zeilenbindung) oder `matrix()` erstellt werden. Bei letzterer Funktion geben die Argumente `ncol` bzw. `nrow` die Spalten-/Zeilenanzahl an:

```
> mat.1 <- cbind(1:3, c(4,3,6))
> mat.1
```

```
     [,1] [,2]
[1,]    1    4
[2,]    2    3
[3,]    3    6

> mat.2 <- rbind(1:3, c(4,3,6))
> mat.2

     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    3    6

> mat.3 <- matrix(data=c("A","B","C","D","E","F"), nrow=3)
> mat.3

     [,1] [,2]
[1,] "A"  "D"
[2,] "B"  "E"
[3,] "C"  "F"

> mat.4 <- matrix(data=c("A","B","C","D","E","F"), ncol=3)
> mat.4

     [,1] [,2] [,3]
[1,] "A"  "C"  "E"
[2,] "B"  "D"  "F"
```

Die Namen der Spalten und Zeilen lassen sich mit der Funktion `colnames()` bzw. `rownames()` (letztere ist auch bei der Bearbeitung von Data-Frames hilfreich) setzen:

```
> colnames(mat.2) <- c("one","two","three")
> rownames(mat.2) <- c("A","B")
> mat.2

  one two three
A   1   2     3
B   4   3     6
```

Eine Matrix lässt sich mit der Funktion `t()` (für transponieren) drehen. Mit `dim()` lassen sich die Dimensionen, also die Anzahl der Zeilen und Spalten der Matrix anzeigen.

1.5.4.3 Liste

Eine Liste ist eine Zusammenstellung von beliebigen Objekten (z.B. der Ausdruck von Testergebnissen). Da die Objekte beliebig sind, können problemlos verschiedene Datentypen gleichzeitig in einer Liste kombiniert werden:

```
> vec.numeric <- c(1:6)
> mat.character <- rbind(c("tomato","cucumber", "iceberg","pepper",
+ "egg fruit","cauliflower"), c(1,4,6,2,7,9), c("D5","A1","E9","G3",
+ "B5","P1"))
> list.1 <- list(example.vec=vec.numeric, example.mat=mat.character)
> list.1
```

```
$example.vec
[1] 1 2 3 4 5 6

$example.mat
     [,1]     [,2]       [,3]      [,4]     [,5]        [,6]
[1,] "tomato" "cucumber" "iceberg" "pepper" "egg fruit" "cauliflower"
[2,] "1"      "4"        "6"       "2"      "7"         "9"
[3,] "D5"     "A1"       "E9"      "G3"     "B5"        "P1"
```

Benennen und Hinzufügen von Listenelementen:

```
> names(list.1)[2] <- "new name"
> list.1$new.element <- c(9,8,7,6,5)
> list.1

$example.vec
[1] 1 2 3 4 5 6

$`new name`
     [,1]     [,2]       [,3]      [,4]     [,5]        [,6]
[1,] "tomato" "cucumber" "iceberg" "pepper" "egg fruit" "cauliflower"
[2,] "1"      "4"        "6"       "2"      "7"         "9"
[3,] "D5"     "A1"       "E9"      "G3"     "B5"        "P1"

$new.element
[1] 9 8 7 6 5
```

Mit `names()` lassen sich die Namen von Listen- und Data-Frame-Elementen ausgeben.

1.5.4.4 Data-Frame

Der Data-Frame wird in der Biometrie am häufigsten verwendet. Es handelt sich um eine zweidimensionale Datenstruktur, die spaltenweise verschiedene Datentypen enthalten kann. Alle Spalten müssen gleich lang sein:

```
> x <- c(1:6)
> x[2] <- 12
> treatment <- rep(x=c("A","B"), each = 3)
> my.frame <- data.frame(group=treatment, value=x)
> my.frame

  group value
1     A     1
2     A    12
3     A     3
4     B     4
5     B     5
6     B     6
```

Ein Data-Frame lässt sich durch die Funktion `transform()` bearbeiten:

```
> new.frame<-transform(my.frame,evaluation=c("wenig",NA,"mittel","mittel",
+ "viel","viel"))
> new.frame
```

```
  group value evaluation
1     A     1      wenig
2     A    12       <NA>
3     A     3     mittel
4     B     4     mittel
5     B     5       viel
6     B     6       viel
```

1.5.4.5 Teilmengen

Der Befehl `Vektorname[Positionsnummer(n)]` ermöglicht den Zugriff auf Einzelwerte in Vektoren:

```
> vec.8[2]

b
3

> vec.8[2:4]

b c d
3 5 7

> vec.8[c(1,3,4)]

a c d
1 5 7
```

Die Anwendung auf eine Matrix funktioniert ähnlich. Hier müssen aber Zeile und Spalte angegeben werden (`Matrixname[Zeilennummer(n),Spaltennummer(n)]`) . Durch das Auslesen ändert sich der Datentyp. Die Matrixdaten verwandeln sich in einen Vektor:

```
> mat.3[1,2]

[1] "D"

> mat.3[c(2,3),2]

[1] "E" "F"
```

Bei Listen wird durch Anwendung des Befehls `Listenname[Listenelementnummer]` eine neue Liste zurückgegeben, die das Element der entsprechenden Nummer enthält. Durch den Befehl `Listenname[[Listenelementnummer]]` erfolgt eine Rückgabe in Form der ursprünglichen Datenform dieses Elements, z.B. als Vektor:

```
> list.1[1]

$example.vec
[1] 1 2 3 4 5 6

> list.1[[1]]
```

```
[1] 1 2 3 4 5 6
```

Im Data-Frame lassen sich Spalten, Zeilen und Einzelwerte wie bei Matrix beschrieben aufrufen. `Objektname$Elementname/Spaltenname` ermöglicht in Listen und Data-Frames eine andere Art des Objektzugriffs:

```
> list.1$example.vec
```

```
[1] 1 2 3 4 5 6
```

```
> my.frame$group
```

```
[1] A A A B B B
Levels: A B
```

Bei häufigem Zugriff auf Elemente von Listen und Data-Frames ist es sinnvoll, die Funktion `attach()` zu verwenden. Zur Ausgabe eines Data-Frame-Elements reicht dann der Name bzw. die Spaltenüberschrift. Nach Benutzung sollten Objekte mit der Funktion `detach()` ausgekoppelt werden, da es sonst zu Konflikten mit anderen Objekten kommen kann (z.B. bei identischen Spaltennamen in verschiedenen eingebundenen Datensätzen):

```
> attach(list.1)
> example.vec
```

```
[1] 1 2 3 4 5 6
```

```
> detach(list.1)
```

Mit der Funktion `subset()` lassen sich Teilmengen ausgeben, die bestimmte Kriterien erfüllen, z.B. alle Elemente aus `my.frame`, die größer als 3 sind:

```
> subset(x = my.frame, subset = value > 3)
```

```
  group value
2     A    12
4     B     4
5     B     5
6     B     6
```

1.5.5 Import und Export von Datensätzen

Unter Windows ist der Import von Exceldatenblättern mit Hilfe des Paketes ODBC möglich. Es ist darauf zu achten, dass die Daten in der Exceldatei im flat file Format vorliegen:

Im **flat file Format** enthält jede Zeile die vollständige Information zu einem Eintrag, z.B. Block: A, Wiederholung: 3, Pflanzenhöhe: 5.

```
> library(rodbc)
> full.data <- odbcConnectExcel("filename.xls")
> sqlTables(full.data)
> data <- sqlQuery(full.data, 'select * from "Sheet1$"')
> odbcCloseAll()
```

In deutschen Excel-Versionen wird das Arbeitsblatt mit **Tabelle** anstelle von **Sheet** angegeben.

Unter Windows kann die Angabe des Pfades zur Textdatei entfallen, wenn zuvor im Menü **File** der Punkt **Change Directory** angeklickt und ins entsprechende Verzeichnis gewechselt wurde. Unter Linux wird das Arbeitsverzeichnis mit der Funktion `setwd()` geändert.

Unter Windows ist außerdem der Datenimport durch Copy & Paste möglich. Dazu wird der Datensatz mit **STRG C** kopiert und durch folgenden Befehl in der R-Konsole aufgerufen:

```
> data <- read.table(file("clipboard"), header = TRUE)
```

`header` steht für die Überschrift der Spalten. Enthalten die Spalten in der Ursprungstabelle keine Überschrift, so kann das Argument weggelassen werden, da FALSE der Default-Wert ist.

Unter Linux ist weder der Import von Excel-files noch Copy & Paste möglich. Um dennoch an Daten aus Excel-files zu gelangen, kann man diese als *.txt- oder *.csv-Datei speichern, bzw. den Datensatz samt Überschrift kopieren, in einen Editor einfügen und dort speichern. Der Import von Textdateien mit `read.table()` funktioniert auf allen Plattformen:

```
> data <- read.table(file = "/path/to/file/filename.txt", header = TRUE,
+ sep = "\t", dec = ",")
```

Das Argument `dec` steht für Dezimaltrennzeichen. Im deutschsprachigen Raum sind Kommata üblich. Dient der Export jedoch nur der Zwischenspeicherung und ein Datensatz soll später wieder in R importiert werden, dann bietet es sich an, den englischen Punkt zu übernehmen, weil so beim nächsten Import die Spezifizierung des Trennzeichens entfällt (Default in R ist der Punkt).

`sep` bedeutet Separator, `\t` steht für Tabulator.

Der Export von Daten in *.txt-files ist mit der Funktion `write.table()` möglich:

```
> write.table(x = my.frame, file = "/path/to/file/filename.txt",
+ sep = "\t", dec=".", col.names = TRUE)
```

`col.names` spezifiziert, ob die Spaltenüberschriften im exportierten Dokument als Überschrift enthalten sein sollen oder nicht. TRUE ist der Defaultwert.

1.5.6 Verwaltung des Arbeitsplatzes

Die Navigation durch bereits verwendete Befehlszeilen mit den Pfeiltasten (Abschnitt 1.5.1) ist praktisch, geht aber beim Restart von R verloren. Folgende Funktionen können verwendet werden, um Befehlszeilen zu sichern bzw. wiederaufzurufen:

```
> savehistory(file = "filename.Rhistory")
> loadhistory(file = "filename.Rhistory")
```

Unter Windows dient der Unterpunkt **Save workspace...** im Menü **File** zum Sichern aller gerade verwendeten Objekte. Gespeicherte Objekte können später durch **Load workspace...** wieder aufgerufen werden. Auf allen Plattformen können Objekte durch die Funktion `save()` gespeichert bzw. durch `load()` aufgerufen werden:

```
> save(list = ls(), file = "filename.RData")
> load(file = "filename.RData")
```

Es ist unter Windows auch möglich, den produzierten Quelltext über das GUI durch Wahl von **Save to file...** im Menü **File** abzuspeichern.

Im Zusammenhang mit Sichern und Laden von Datein (auch Import bzw. Export von Datensätzen) ist die Funktion `setwd()` hilfreich. Sie setzt ein Arbeitsverzeichnis:

```
setwd("/directory")
```

Unter Windows kann das Arbeitsverzeichnis im Menü **File** – **Change Directory** gewählt werden. Die Funktion `getwd()` ruft das momentane Arbeitsverzeichnis auf.

Zur langfristigen, übersichtlichen Sicherung von Projekten ist die Verwendung eines Editors (siehe Abschnitt 1.4.5) sinnvoll.

✎ Übungsaufgabe 1

1. Berechnen Sie in R die zweite binomische Formel

   $$(a - b)^2$$

   mit a = 12 und b = 7. Erstellen Sie dazu zunächst die Objekte a und b! Speichern Sie das Ergebnis in einem Objekt. Wählen Sie einen eindeutigen Objektnamen!

2. Erzeugen Sie ein Objekt, welches rücklaufend die Zahlen 28 bis -34 enthält!

3. Rufen Sie die Hilfe zur Funktion `objects()` auf und beenden Sie die Hilfe anschließend korrekt! Verwenden Sie die Funktion `objects()`, um sich alle nun vorhandenen Objekte anzeigen zu lassen! Entfernen Sie das Objekt a!

4. Erstellen Sie zu Tabelle 1.1 einen Data-Frame im flat file-Format!

Batch	Culture Solution	Plant 1	Plant 2	Plant 3
A	Complete	1172	750	784
B	Lacking magnesium	67	95	59
C	Lacking nitrogen	148	234	92
D	Lacking micro-nutrients	297	243	263

Tabelle 1.1: Daten eines Pflanzenernährungsexperimentes mit Sonnenblumen in Flüssigkultur. Endpunkt Trockengewicht (mg) (Bishop, 1980, S. 1).

Kapitel 2

Deskriptive Statistik

2.1 Grundfunktionen

Um die Anwendung der Funktionen zur deskriptiven Statistik zu veranschaulichen, wird ein Beispielvektor erstellt:

```
> data <- c(34,5,23,17,23,19,21,12,25,22,19,19,12,22,17)
```

`mean()` berechnet den Mittelwert von `data`:

```
> mean(data)

[1] 19.33333
```

Auf dieselbe Art und Weise lassen sich Standardabweichung (`sd()`), Median (`median()`), Varianz (`var()`), Interquartilabstand (`IQR()`), Minimum (`min()`), Maximum (`max()`), Minimum und Maxium (`range()`), Bereich (`diff()`) und Summe (`sum()`) berechnen.

Den Variationskoeffizienten erhält man durch folgende Eingabe:

```
> var.coeff <- sd(data)/mean(data)
> var.coeff

[1] 0.3429134
```

Die Funktion `quantile()` errechnet per Default das 0%, 25%, 50%, 75% und 100% Quantil. Man kann die gewünschten Quantile jedoch auch mit `probs` angeben:

```
> quantile(data, probs=c(0.25,0.75)) # berechnet das 25 und
+ 75 Prozent Quartil

 25%  75%
17.0 22.5
```

Die Funktion `summary()` fasst die gibt wichtigsten Kenngrößen der deskriptiven Statistik zusammen:

```
> summary(data)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   5.00   17.00   19.00   19.33   22.50   34.00
```

2.2 Die Funktion tapply()

Die Schleifenfunktion `tapply()` ermöglicht die schnelle und einfache Auswertung von Datensätzen, die in verschiedene Kategorien (z.B. Behandlungen) eingeteilt sind, mit den Grundfunktionen der deskriptiven Statistik.

```
tapply(X, INDEX, FUN = NULL, ...)
```

`X` gibt die Messwerte z.B. als Spalte in einem Data-Frame an. `INDEX` gibt die dazugehörige Spalte mit den verschiedenen Kategorien an.

`FUN` spezifiziert die verwendete Funktion der deskriptiven Statistik, z.B. `sum`, `mean`, `var` oder `IQR`.

`tapply()` gibt einen Array mit den errechneten Daten zurück.

2.2.1 Beispiel Bodenatmung (1)

2.2.1.1 Versuchsbeschreibung

Das Pflanzenwachstum wird durch die mikrobielle Aktivität im Boden beeinflusst. Ein Maß für diese ist die Bodenatmung. In einer Studie wurden Bodenproben von zwei charakteristischen Gegenden im Wald entnommen: Auf Lichtungen zwischen den Baumkronen („gap“) und in nahe gelegenen Gegenden mit dichtem Baumbestand („growth“). Die Menge des von der Bodenprobe abgegebenen Kohlendioxids wurde in mol CO_2 g^{-1} Boden hr^{-1} gemessen (siehe Data 2.1) (Fierer, 1994) zitiert nach (Samuels and Witmer, 2003, S. 289).

Growth	Gap
17	22
20	29
170	13
315	16
22	15
190	18
64	14
	6

Data 2.1: Daten zur Bodenatmung (mol CO_2/g Boden/hr).

2.2.1.2 Auswertung der Daten

Berechnung des Mittwelwertes, der Standardabweichung, des Medians, der Varianz und der Quartile mit `tapply()`:

```
> soil <- data.frame(treatment = c(rep(c("growth"), times = 7),
+ rep(c("gap"), times = 8)),response = c(17,20,170,315,22,190,
+ 64,22,29,13,16,15,18,14,6))
> tapply(X = soil$response, INDEX = soil$treatment, FUN = mean)

    gap  growth
 16.625 114.000

> tapply(X = soil$response, INDEX = soil$treatment, FUN = sd)

       gap     growth
  6.759913 114.398427

> tapply(X = soil$response, INDEX = soil$treatment, FUN = median)

   gap growth
  15.5   64.0

> tapply(X = soil$response, INDEX = soil$treatment, FUN = var)
```

```
        gap      growth
   45.69643 13087.00000
```

```
> tapply(X = soil$response, INDEX = soil$treatment, FUN = quantile)
```

```
$gap
   0%   25%   50%   75%  100%
 6.00 13.75 15.50 19.00 29.00

$growth
  0%  25%  50%  75% 100%
  17   21   64  180  315
```

2.3 Die Funktion `stat.desc()`

Das Paket `pastecs` liefert die Funktion `stat.desc()`, mit welcher sich eine Tabelle der deskriptiven statistischen Kenndaten zu mehreren Variablen angeben lässt:

```
stat.desc(x, basic=TRUE, desc=TRUE, p=0.95, ...)
```

`x` ist ein Data-Frame mit unterschiedlichen Variablen.

Per Default ist `basic` auf `TRUE` gesetzt. Es bedeutet, dass die Werte für *Anzahl der Beobachtungen*, *Anzahl der Nullwerte*, *Anzahl der fehlenden Werte*, *Minimum*, *Maximum*, *Bereich* und *Summe aller nicht fehlenden Werte* in der Ausgabetabelle ausgegeben werden. Wird das Argument auf `FALSE` gesetzt, so fehlen diese Ergebnisse im Output.

Das Argument `desc` steuert die Ausgabe der Kenngrößen der deskriptiven Statistik: Median, Mittelwert, Standardfehler des Mittelwertes, das Konfidenzintervall des Mittelwertes zum festgelegten p-Wert, Varianz, Standardabweichung und Variationskoeffizient (definiert als Standardabweichung geteilt durch den Mittelwert). In der Defaultkonfiguration steht dieses Argument auf `TRUE`.

`p` definiert das Konfidenzniveau.

2.3.1 Beispiel Bodenatmung (2)

Das Paket `pastecs` wird mit der Funktion `library()` eingebunden:

```
> library(pastecs)
```

Mit der Funktion `stat.desc()` wird ein umfassender Output erzeugt:

```
> stat.desc(x = soil)
```

```
         treatment   response
nbr.val         NA   15.00000
nbr.null        NA    0.00000
nbr.na          NA    0.00000
min             NA    6.00000
max             NA  315.00000
range           NA  309.00000
sum             NA  931.00000
```

```
median          NA   20.00000
mean            NA   62.06667
SE.mean         NA   23.32390
CI.mean         NA   50.02480
var             NA 8160.06667
std.dev         NA   90.33309
coef.var        NA    1.45542
```

✎ Übungsaufgabe 2

Für 16 Tage wurden die beiden Salatvarietäten *Salad Bowl* und *Bibb* unter gleichen Bedingungn im Gewächshaus herangzogen. Data 2.2 zeigt das Trockengewicht der Blätter von neun *Salad Bowl*- und sechs *Bibb*-Pflanzen (Samuels and Witmer, 2003, S. 226).

Erstellen Sie einen Data-Frame im Flat File Format zu den Daten!

Berechnen mit Hilfe der Funktion `tapply()` für die beiden Varietäten jeweils Mittelwert, Standardabweichung, Median, Varianz, Minimum, Maximum, die Quartile, Summe und IQR!

Salad Bowl	**Bibb**
3.06	1.31
2.78	1.17
2.87	1.72
3.52	1.20
3.81	1.55
3.60	1.53
3.30	
2.77	
3.62	

Data 2.2: Trockengewicht zweier Salatvarietäten (g)

Kapitel 3

Graphiken in R

R verfügt über viele Funktionen zur Graphikerstellung. Die meisten Parameter von Plotfunktionen sind universell anwendbar. Am Beispiel des Boxplots wird der Unterschied zwischen Default-Eingabe und der Spezifizierung weiterer Argumente exemplarisch veranschaulicht.

3.1 Boxplot

Ein Boxplot zeigt die Verteilung einer Stichprobe an. Er wird häufig zur Überprüfung der Gaußverteilung benutzt. Mehrere Boxplots können zur Beurteilung der Varianzhomogenität von mehreren Stichproben herangezogen werden (vgl. Abschnitt 5.1).

Einige Parameter der Funktion `boxplot()`:

```
boxplot(x, col = NULL, xlab = "...", ylab = "...", main = "...")
```

`x` ist entweder ein Vektor oder eine Liste, die mehrere Vektoren enthält. Alternativ können die Daten mit einem `formula`-Konstrukt angegeben werden:

```
formula = observations ~ grouping factor with two levels,
data = ..., subset = ..., na.action
```

Bei Benutzung des `formula`-Konstruktes werden die einzelnen Gruppennamen von Funktionen alphabetisch behandelt, das heißt, die im Alphabet als an erster Stelle auftauchende Gruppe, wird beim Boxplot stets die erste Box darstellen.

`col` spezifiziert die Füllfarbe der Graphik. Mit der Funktion `color()` lassen sich die vordefinierten Farben auflisten.

Mit `xlab` und `ylab` werden die Achsennamen spezifiziert. Per Default ist als X-Achsenbeschriftung der Vektorname gesetzt.

Mit `main` lässt sich ein Diagrammtitel hinzufügen. Alternativ kann dies auch nachträglich mit der Funktion `title()` geschehen.

Abbildung 3.1 zeigt den Unterschied zwischen der Angabe des einzigen obligatorischen Argumentes und der Spezifikation weiterer Argumente.

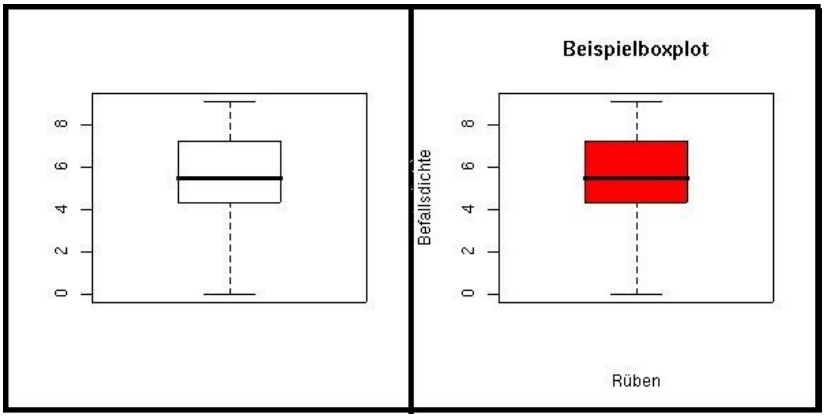


Abbildung 3.1: Beispiel zur Verdeutlichung des Unterschieds zwischen Benutzung der Defaulkonfiguration (**links:** boxplot(x = data)) und Spezifikation der einzelnen Argumente (**rechts:** boxplot(x = data, col = "red1", xlab = "Rüben", ylab = "Befallsdichte", main = "Beispielboxplot").

3.1.1 Beispiel Bodenatmung (3)

Auf die Daten aus Abschnitt 2.2.1 zurückgreifend, werden die Boxplots zur Bodenatmung auf Lichtungen und im dichten Waldbestand erstellt (Abbildung 3.2):

```
> boxplot(formula = response~treatment, data = soil, col = "red1",
+ ylab = "soil respiration (mol CO2/g soil/hr")
> title("Soil Respiration in the Forest")
```

3.2 Histogramm

Mit einem Histogramm werden Häufigkeiten dargestellt.

3.2.1 Beispiel Sojabohnen

3.2.1.1 Versuchsbeschreibung

Eine Pflanzenphysiologin hat dreizehn in einzelne Töpfe gepflanzte Sojabohnensämlinge des Typs Wells II. auf ihr Wachstum hin untersucht. Die Pflanzen wurden im Gewächshaus unter gleichen Umweltbedingungen (Licht, Temperatur, Boden usw.) aufgezogen. Die Stängellänge jeder Pflanze wurde nach 16 Tagen gemessen (siehe Daten 3.1) (Pappas and Mitchell, 1984, das tatsächliche Experiment enthielt mehrere Versuchsgruppen, die mit unterschiedlichen Umweltbedingungen behandelt wurden.), Rohdaten publiziert in (Samuels and Witmer, 2003, S. 179).

20.2	22.9
23.2	20.0
19.4	22.0
22.1	22.0
21.9	21.5
19.7	21.5
20.9	

Data 3.1: Stängellänge von Sojabohnensämlingen.

3.2.1.2 Graphische Darstellung der Daten

Erstellung eines Histogramms mit `hist()` zu den Sojabohnensämlingsdaten (Abbildung 3.3):

```
> beans <- c(20.2,22.9,23.3,20,19.4,22,22.1,22,21.9,21.5,19.7,21.5,20.9)
> hist(beans, col = "red1", main = "Histogram of Soybean Seedlings",
+ breaks = 5)
```

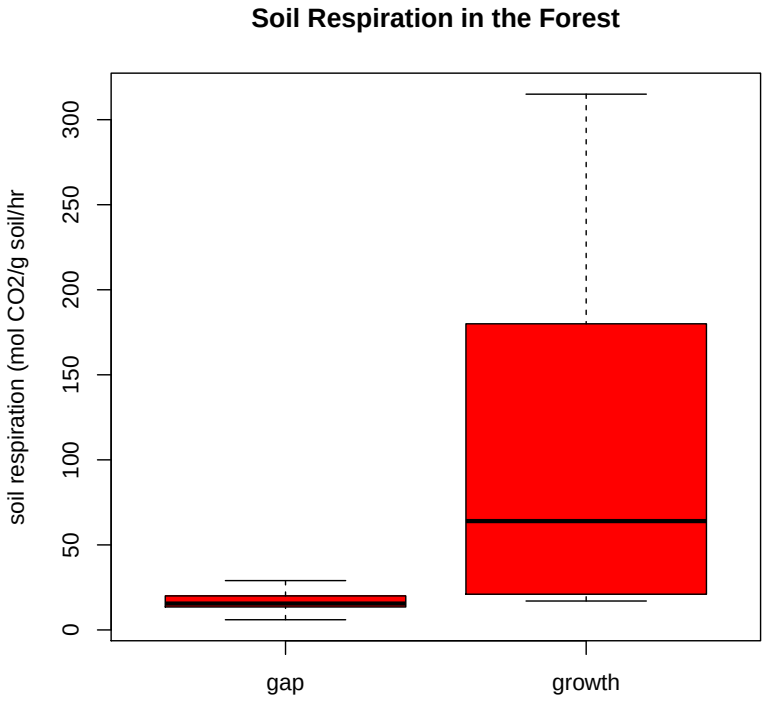


Abbildung 3.2: Boxplots der Bodenatmung im Wald.

Das Argument `breaks` definiert hier die Anzahl der Säulen.

3.3 Scatterplot

Die Funktion `plot()` gibt in ihrer Grundfunktion eine empirische kumulative Verteilungsgraphik zurück.

Zur Veranschaulichung werden die Daten aus dem Beispiel Zuckerrüben in Abschnitt 9.2.5 verwendet (Abbildung 3.4).

```
> beets <- read.table(file = "../text/beets.txt", sep = "\t",
+ header = TRUE)
> plot(yield~water, data = beets, col = "red1", xlab = "irrigation (mm)",
+ ylab = "yield (t/ha)")
> title("Sugarbeet Irrigation")
```

Vorsicht! Die Funktion `plot` akzeptiert zwar eine Datenangabe mit dem `formula`-Konstrukt, jedoch nur, wenn `formula =` **nicht** getippt wird!

Mit der Funktion `abline()` lässt sich z.B. eine horizontale Gerade durch den Graph legen:

```
> abline(h = 14, col = "red1")
```

Das Argument `type` kann beispielsweise `p` für Punkte, `l` für Linie oder `b` für sowohl Linie als auch Punkte lauten.

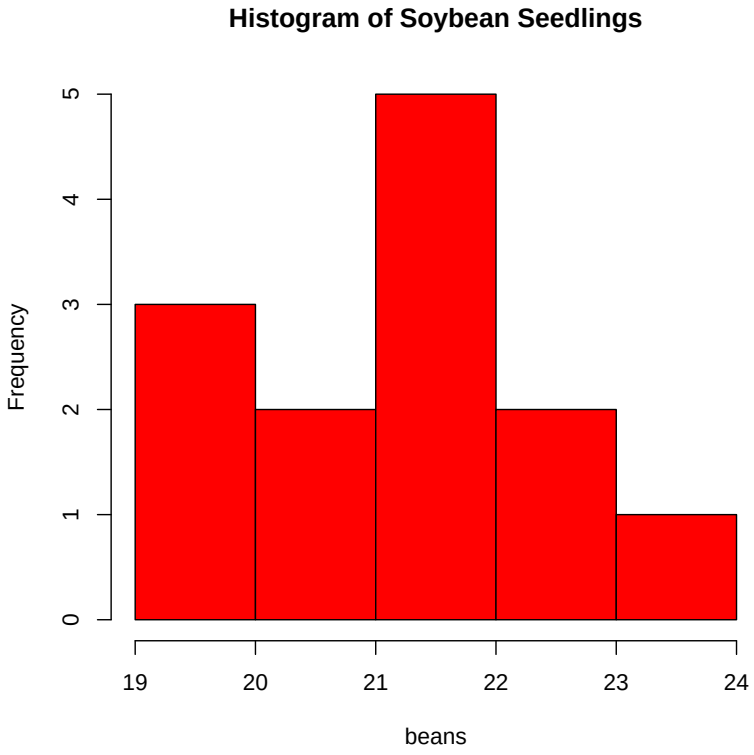


Abbildung 3.3: Histogramm zu Sojabohnensämlingen.

3.4 QQ-Plot

Der QQ-Plot für Normalverteilung kann mit der Funktion `qqnorm()` erzeugt werden (siehe Abbildung 3.5). Mit `qqline()` lässt sich eine Gerade hindurchlegen:

```
> qqnorm(beans, col = "red1", main = "QQ-Plot of Soybeans")
> qqline(beans, col = "red1")
```

Das Paket `car` stellt die Funktion `qq.plot()` bereit, die auch für andere Verteilungen verwendet werden kann.

Weitere Beispiele für die Verwendung der Funktionen `plot()` und `qqnorm()` sind ab Abschnitt 9.2.3 zu finden.

3.5 Weitere Graphikfunktionen

Viele Objekte lassen sich mit der Funktion `plot()` zeichnen, beispielsweise Konfidenzintervalle der Funktion `simint()` und die Regressionsdiagnostik eines linearen Modells (`lm(model)`) (siehe Abschnitt 9.2.3 und 9.2.5.3).

Stamm-Blatt-Diagramm (`stem()`), Balkendiagramm (`barplot()`) und Tortendiagramm (`pie()`).

Abbildung 3.4: Die Zuckerrübendaten als Beispiel für einen Scatterplot.

✎ Übungsaufgabe 3

Benutzen Sie die Daten aus Übungsaufgabe 2 (Data 2.2), um sich die Boxplots pro Sorte zeichnen zu lassen! Vergeben Sie einen Graphiktitel, Achsenbezeichnungen und eine Füllfarbe für die Boxen!

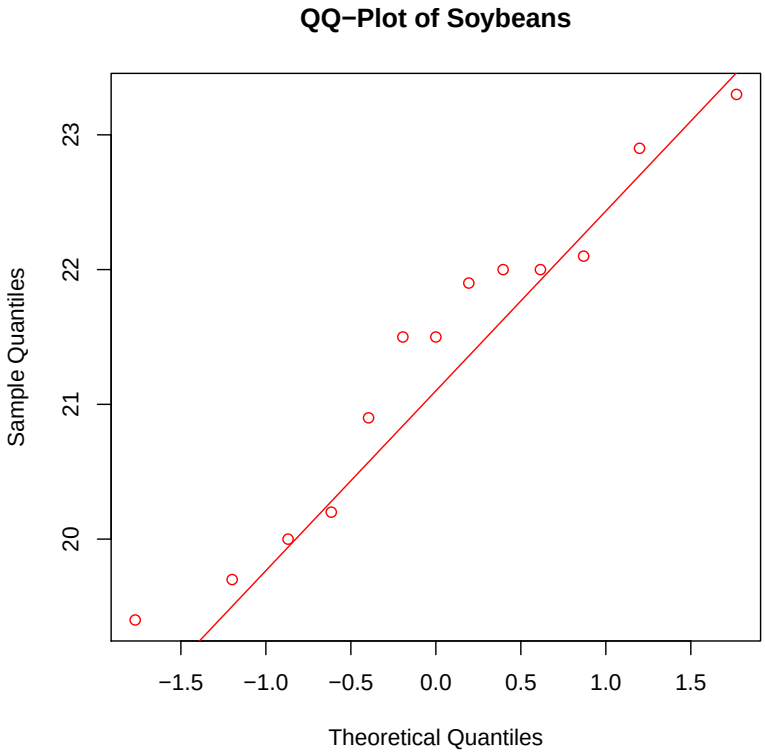


Abbildung 3.5: QQ-Plot der Sojabohnendaten.

Kapitel 4

F-Test

4.1 Voraussetzungen

Der F-Test[1] ist ein Vortest für parametrische Zweistichprobentests. Er untersucht, ob die Varianzen zweier Stichproben inhomogen sind und ergänzt die Betrachtung der Boxplots wie in Abschnitt 5.1 beschrieben.

Das Hypothesenpaar dieses Tests lautet:

$$H_0 : \frac{\sigma_A}{\sigma_B} = 1$$
$$H_1 : \frac{\sigma_A}{\sigma_B} \neq 1$$

Eine wichtige Voraussetzung für den F-Test ist Normalverteilung der beiden Stichproben (siehe Abschnitt 5.1).

Vorsicht! Liegt eine Signifikanz beim F-Test vor, so kann von Varianzheterogenität ausgegangen werden. Dies gilt aber nicht umgekehrt. Liegt keine Signifikanz vor, so darf nicht einfach von einer Varianzhomogenität ausgegangen werden. Liegt der p-Wert relativ nahe 1.0, wird hier zusammen mit Betrachtung der Boxplots davon ausgegangen, dass keine Heterogenität vorliegt.

4.2 Anwendung

4.2.1 Die Funktion `var.test()`

```
var.test(x, y, ratio = 1, alternative = c("two.sided", "less", "greater"),
              conf.level = 0.95, ...)
```

`x` und `y` sind zwei numerische Datenvektoren. Alternativ kann ein `formula`-Konstrukt (siehe Abschnitt 3.1) verwendet werden.

`ratio` bezieht sich auf das Verhältnis der Varianzen in den Arbeitshypothesen. Der Default-Wert ist `1`.

`alternative` spezifiert, ob ein- oder zweiseitig getestet wird. Der Default-Wert ist `two.sided`.

`conf.level` gibt das Konfidenzniveau an, `0.95` ist der Defaultwert.

Als Vortest für einen t-Test oder Wilcoxon-Rangsummen-Test genügt die Angabe der Vektoren bzw. des `formula`-Konstruktes. Es wird automatisch ein zweiseitiger Test mit einem Verhältnis von 1 und einem Konfidenzniveau von 0.95 berechnet.

[1] Es gibt noch einen anderen F-Test, die ANOVA (Abschnitt 10), welche die gleiche Verteilung zu einem anderen Zweck verwendet. Durch Anwendung der ANOVA wird mit Hilfe der Varianzanalyse nach Unterschieden in mehr als zwei Stichproben gesucht.

4.2.2 Beispiel „Wisconsin Fast Plant“(1)

Control	Ancy
10.0	13.2
13.2	19.5
19.8	11.0
19.3	5.8
21.2	12.8
13.9	7.1
20.3	7.7
9.6	

Data 4.1: Höhe der Brassicapflanzen nach 14 Tagen (cm).

4.2.2.1 Versuchsbeschreibung

Brassica campestris, auch bekannt als „Wisconsin Fast Plant“, hat einen schnellen Wachstumszyklus. Faktoren, die das Pflanzenwachstum beeinflussen, lassen sich deshalb besonders gut an diesem Modell untersuchen. In einer Studie wurden sieben Pflanzen mit der Substanz Ancymidol (ancy) behandelt und mit acht Kontrollpflanzen, die nur Wasser erhielten, verglichen. Ancymidol ist ein Wachstumshemmer und wird u.a. als Herbizid eingesetzt. Die Höhe von allen Pflanzen wurde nach 14 Tagen in cm gemessen (Data 4.1 nebenstehend) (Ahern, 1998) zitiert nach (Samuels and Witmer, 2003, S. 228, Autor gibt an, dass es sich nur um einen Teildatensatz des Experimentes handelt).

4.2.2.2 Auswertung der Daten

```
> brassica <- read.table("../text/brassica.txt", sep = "\t", header = TRUE)
> var.test(formula = height~group, data = brassica)

        F test to compare two variances

data:  height by group
F = 0.97316, num df = 6, denom df = 7, p-value = 0.9898
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.1901215 5.5425762
sample estimates:
ratio of variances
         0.9731551
```

4.2.2.3 Interpretation des Outputs

```
F = 1.0276, num df = 7, denom df = 6, p-value = 0.9898
```

Für die Interpretation des Outputs gelten praktisch dieselben Bedingungen wie für den t-Test (Abschnitt 5.2.3.2). Der p-Wert wird mit einem a priori festgelegten α verglichen. Ist er kleiner als α, so wird die Alternativhypothese angenommen.

Der F-Test prüft auf Varianzheterogenität. Für die Testauswahl interessiert aber die Varianzhomogenität. Es gibt keine mir bekannte Lösung, um mit R auf Varianzhomogenität zu testen. Auch allgemeine Faustregeln existieren nicht. Wenn der p-Wert des F-Tests groß ist, gehe ich gemeinsam mit der Boxplotbetrachtung in allen Beispielen von Varianzhomogenität aus. Die Funktion `var.test()` wird in Kapitel 5 beschrieben.

Der p-Wert von 0.9898 indiziert, dass keine signifikante Varianzheterogenität zu einem α von 5% vorliegt $\Longrightarrow$ Varianzhomogenität.

Kapitel 5

t-Test

5.1 Voraussetzungen

Der t-Test ist ein parametrischer Test und dient dem Mittelwertvergleich zweier Stichproben.

Der klassische **t-Test** sollte verwendet werden, wenn folgende Voraussetzungen erfüllt sind:

- **Annähernde Gaußverteilung der Daten** lässt sich aus den Boxplots der Daten ablesen: Der Median sollte mittig in der Box liegen und die beiden Whisker etwa gleich lang sein. (Siehe Abb. 5.1. Die Boxplots einzeln betrachten!) Aus der Gaußverteilung ergibt sich die Stetigkeit der Daten, z.B. Temperaturen in Kelvin oder Längen in Metern.
- **Varianzhomogenität** ist graphisch aus den Boxplots ersichtlich: Die verschiedenen Boxen nebst Whiskern sollten gleich lang sein (siehe Abb. 5.1). Ein statistischer Test kann diese Evaluation ergänzen. In Abschnitt 4 wird der F-Test zum Vergleich zweier Varianzen beschrieben (`var.test()`).
- **Unabhängigkeit der Daten** ist nicht gegeben, wenn z.B. der Schädlingsbefall an denselben Obstbäumen in zwei aufeinanderfolgenden Jahren gemessen wurde, oder wenn Explantate für 100 Petrischalen, die miteinander verglichen werden sollen, von einer Ausgangspflanze stammen. Mehrere Explantate in einer Petrischale dürfen auch nicht als unabhängig betrachtet werden.

Der **Welch t-Test** ist dem normalen t-Test sehr ähnlich. Grundvoraussetzung sind ebenfalls gaußverteilte, unabhängige Daten, aber der Welch t-Test ist toleranter gegenüber Varianzheterogenität.

Für die Verwendung des **gepaarten t-Tests** müssen folgende Voraussetzungen erfüllt sein:

- **Gepaarte Daten**: Eine gepaarte Stichprobe erhält man beispielsweise, wenn an einem Apfelbaum die Wirkung zweier unterschiedlicher Insektizide an verschiedenen Ästen untersucht wird.
- **Gaußverteilung der Paardifferenzen** (Boxplot).

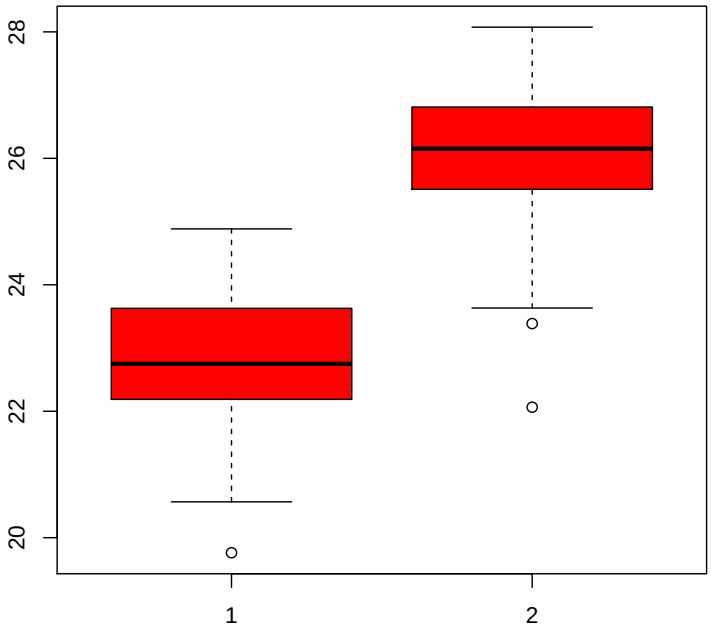


Abbildung 5.1: Boxplotbeispiel für t-Test.

5.2 Anwendung

5.2.1 Die Funktion `t.test()`

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

oder

```
t.test(formula, data, subset, na.action, ...)
```

In der Funktion `t.test()` stehen `x` und `y` für zwei Vektoren, die gegeneinander getestet werden. `y` muss nicht angegeben werden, denn der t-Test kann auch als Einstichprobentest verwendet werden. Alternativ kann man die Daten auch mit dem `formula`-Konstrukt spezifizieren (3.1).

`data` gibt den Datensatz an, auf den das `formula`-Konstrukt angewendet wird.

`subset` spezifiziert, ob Daten mit bestimmten Merkmalen ausgeschlossen bzw. zugelassen werden (siehe Abschnitt 1.5.4.5).

Mit `na.action` wird festgelegt, was bei nicht verfügbaren Werten geschehen soll. Der Aufruf der vorhandenen Optionen geschieht durch:

```
getOption("na.action").
```

`alternative` gibt an, ob zweiseitig (H_1: $\mu_1 \neq \mu_2$), einseitig auf Anstieg (H_1: $\mu_1 > \mu_2$) oder einseitig auf Abfall (H_1: $\mu_1 < \mu_2$) getestet wird.

`var.equal` indiziert, ob die Varianzen heterogen (`FALSE`) oder homogen (`TRUE`) sind. Für einen klassischen t-Test muss `var.equal = TRUE` gesetzt werden. Per Default wird ein Welch t-Test berechnet.

`conf.level` gibt das Konfidenzlevel an. Der Fehler 1. Art ergibt sich aus 1 - `conf.level`. Der Default ist 0.95, also 95% $\Rightarrow \alpha = 5\%$.

`paired` ist als Default auf `FALSE` gesetzt. Im Falle abhängiger Daten wird `paired` als `TRUE` angegeben, um einen gepaarten t-Test durchzuführen.

Achtung! R sortiert die Variablen bei Angabe des `formula`-Konstruktes alphabetisch. Um B > A anzugeben, muss `alternative` als `less` angegeben werden.

5.2.2 Die Funktion `qt()`

Der kritische Wert (Quantil) zu einem gegebenen p-Wert und Freiheitsgrad kann separat mit der Funktion `qt()` berechnet werden.

```
qt(p, df, lower.tail = TRUE)
```

`p` ist der bekannte p-Wert (der mit dem einem vorher festgelegten Alpha-Fehler, häufig 0.05, verglichen wird), `df` (= degrees of freedom) steht für die Freiheitsgrade.

Der Defaultwert `lower.tail = TRUE` wird bei zweiseitigen oder einseitig abfallenden Tests ($X \leq x$) verwendet. Für einen einseitig ansteigenden Test muss das Argument auf `FALSE` gesetzt werden.

5.2.3 Beispiel „Wisconsin Fast Plant“(2)

Auf die Daten aus Abschnitt 4.2.2 zurückgreifend lautet die Fragestellung, ob die beiden Gruppen sich signifikant voneinander unterschieden.

5.2.3.1 Auswertung der Daten

```
> brassica <- read.table("../text/brassica.txt", sep = "\t", header = TRUE)
> boxplot(formula = height~group, data = brassica, ylab = "height in cm",
+ main="Height of Brassica Plants", col = "red")
```

- ✔ annähernde Gaußverteilung, denn der Median liegt etwa in der Mitte der Boxen (siehe Abb. 5.2)
- ✔ annähernde Varianzhomogenität, siehe F-Test-Ergebnis in Abschnitt 4.2.2
- ✔ stetige Daten, denn die Höhe ist in cm angegeben
- ✔ unabhängige Daten, denn es handelt sich um unabhängig voneinander behandelte Pflanzen

$\Longrightarrow$ Die Daten sind für eine Auswertung mit dem t-Test geeignet. Verwendung eines einseitigen Tests, da die Erwartung vorliegt, dass mit Ancymidol behandelte Pflanzen kleiner sind, als die unbehandelte Kontrollgruppe. Hypothesenpaar:

$$H_0 : \mu_{control} \leq \mu_{ancy}$$
$$H_1 : \mu_{control} > \mu_{ancy}$$

Abbildung 5.2: Boxplots der Brassicapflanzen nach 14 Tagen.

```
> t.test(formula = height~group, data = brassica, var.equal = TRUE,
+ alternative = "less", conf.level = 0.95)

        Two Sample t-test

data:  height by group
t = -1.9919, df = 13, p-value = 0.03391
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
      -Inf -0.543402
sample estimates:
   mean in group ancy mean in group control
             11.01429              15.91250
```

5.2.3.2 Interpretation des Outputs

```
        Two Sample t-test
```

Die Überschrift des Ouputs gibt noch einmal den verwendeten Test an. Würde der Parameter `var.equal = TRUE` nicht gesetzt, stände hier `Welch Two Sample t-test`.

```
data:  height by group
```

Das Formula-Konstrukt zu den beiden Stichproben wird ausgegeben.

```
t = -1.9919, df = 13, p-value = 0.03391
```

Die Teststatistik `t` beträgt 1.9919. Dies ist die Testgröße, die normalerweise mit dem Tabellenwert verglichen wird. Ist sie extremer, als der Tabellenwert des Quantils zum entsprechenden Freiheitsgrad, gilt der Mittelwertvergleich als signifikant. Die Freiheitsgrade sind als `df` angegeben, hier 13. Der `p-value` wird mit dem gewünschten α-Fehler verglichen. Ist der p-Wert kleiner als α, so ist das Testergebnis signifikant. Der Wert von α ist a priori vor dem Test festzulegen. Der Default in R ist ein α-Fehler von 5% (`conf.level = 0.95`). Die mit Ancymidol behandelten Brassicapflanzen sind signifikant kürzer, als die unbehandelte Kontrollgruppe, da $0.03391 < 0.5$ ist. Die Alternativhypothese wird angenommen.

Die Teststatistik lässt sich auch separat mit der Funktion `qt()` errechnen:

```
> qt(0.03391, 13, lower.tail = FALSE)

[1] 1.99187
```

```
alternative hypothesis: true difference in means is greater than 0
```

Die Alternativhypothese wird ausgegeben.

```
95 percent confidence interval:
      -Inf -0.543402
```

Diese Ausgabe gibt das 95%-Konfidenzintervall für die Differenz der wahren Parameter $\mu_{control}$ - μ_{ancy} zurück. Angenommen, man würde den Versuch unendlich oft indentisch wiederholen, dann läge die wahre Differenz in 95% der Fälle im jeweiligen Konfidenzintervall. Für den aktuellen Versuch kann allerdings nichts ausgesagt werden.

Praktische Anwendung: Wenn in einem Konfidenzintervall die Null eingeschlossen ist, gilt das Testergebnis als nicht signifikant. Im Falle einer Signifkanz stellt der Abstand von Null ein Maß für den Grad der Ablehnung der H_0-Hypothese dar. Die Breite des Konfidenzintervalls ist ein Maß für die Streuung und Fallzahl. Konfidenzintervalle stehen grundsätzlich in der Dimension der Originaldaten, d.h. in diesem Fall in cm. Aus dem gegebenen Konfidenzintervall `0.543402 Inf` ist ersichtlich, dass eine Signifikanz zum Konfidenzniveau von 0.95 vorliegt, denn die Null ist nicht im Konfidenzintervall enthalten, d.h. mit einer Irrtumswahrscheinlichkeit von 5% kann $\mu_{control}$ - $\mu_{ancy} = 0$ ausgeschlossen werden. Insbesondere ist daraus zu schließen, dass die Pflanzen der Kontrollgruppe in 95% der Fälle mindestens 0.542402 cm länger sind, als die mit Ancymidol behandelten.

```
sample estimates:
mean in group ancy mean in group control
          11.01429              15.91250
```

Die beiden Mittelwerte werden ausgegeben. Die mit Wasser behandelte Gruppe hat eine durchschnittliche Höhe von 15.9 cm, die mit Ancymidol behandelte Gruppe hat eine durchschnittliche Höhe von 11.0 cm.

Abschließend lässt sich zu diesem Beispiel sagen, dass die Altnernativhypothese bei einem Konfidenzlevel von 0.95 angenommen wird.

✎ Übungsaufgabe 4

Kleine weiße Würmer führen bei Erdbeeren zu einer Erntedepression. Der Parasit kann mit einem Desinfektionsmittel behandelt werden. Ein neu entwickeltes Additiv soll zu einer längeren Wirksamkeit der Desinfektion führen, jedoch mit unbekannten Nebenwirkungen auf die Erdbeerpflanzen selbst. Um den Gesamteffekt zu untersuchen wurden fünf Plots auf einer Versuchsanlage zufällig ausgewählt, jeweils zufällig in zwei Hälften geteilt und die eine Hälfte wurde mit dem Desinfektionsmittel alleine, die andere mit dem Zusatz des neuen Additivs behandelt. Die Erdbeerernte sah aus, wie in Data 5.2 angegeben (Wonnacott and Wonnacott, 1990, S. 273).

Standard	Additiv
109	107
68	72
82	88
104	101
93	97

Data 5.2: Wirkung eines neuen Additivs zur Desinfektion von Erdbeeren gegen kleine weiße Würmer.

Formulieren Sie geeignete Hypothesen. Liegt Normalverteilung vor? Liegt Varianzhomogenität vor? Für welchen Test entscheiden Sie sich deshalb? Interpretieren Sie den R-Output!

5.2.4 Beispiel Wurzelwachstum von Senfsämlingen

5.2.4.1 Versuchsbeschreibung

In einem Versuch wurde der Einfluss von Licht und Dunkelheit auf das Wachstum der Wurzeln von Senfsämlingen beobachtet (Hand et al., 1994, S. 75, es handelt sich hier um einen Teil des vollständigen Datensatzes). Man möchte wissen, ob sich die Länge der Wurzeln voneinander unterscheidet (Data 5.3).

light	dark
21	22
39	16
31	20
13	14
52	32
39	28
55	36
50	41
29	17
17	22

Data 5.3: Daten zum Wurzelwachstum von Senfsämlingen.

5.2.4.2 Auswertung der Daten

```
> mustard <- read.table(file = "../text/mustard.txt", sep = "\t",
+ header = TRUE)
> boxplot(formula = response~treatment, data = mustard, col = "red",
+ ylab = "rootlength (cm)")
> title("Root Growth of Mustard Seedlings")
```

- ✔ Annähernde Gaußverteilung (Abbildung 5.3) und Stetigkeit der Daten (Wurzeln wurden in cm gemessen)
- ✔ Varianzheterogentiät (Abbildung 5.3, die Boxen sind unterschiedlich lang.)
- ✔ Die Versuchsreihen sind nicht abhängig voneinander.

Da man wissen möchte, ob sich das Wurzelwachstum überhaupt voneinander unterscheidet und a priori nicht ersichtlich ist, wie die Belichtung das Wurzelwachstum beeinflussen könnte, testet man zweiseitig ($\alpha = 5\%$). Das Hypothesenpaar lautet:

$$H_0 : \mu_{light} = \mu_{dark}$$
$$H_1 : \mu_{light} \neq \mu_{dark}$$

```
> t.test(formula = response~treatment, data = mustard,
+ alternative = "two.sided", conf.level = 0.95)

        Welch Two Sample t-test

data:  response by treatment
```

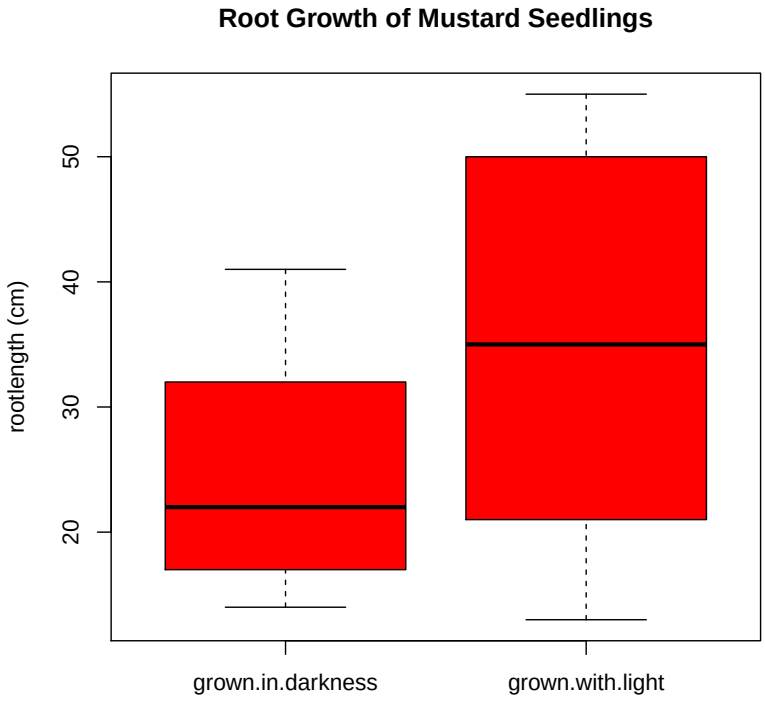


Abbildung 5.3: Boxplots zum Wurzelwachstum von Senfsämlingen.

```
t = -1.7748, df = 14.879, p-value = 0.09638
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -21.57753   1.97753
sample estimates:
mean in group grown.in.darkness  mean in group grown.with.light
                           24.8                            34.6
```

5.2.4.3 Interpretation

Die Interpretation der Ausgabetabelle erfolgt wie in Abschnitt 5.2.3.2.

```
t = -1.7748, df = 14.879, p-value = 0.09638
```

Der `p-value` ist größer als 0.05. Die Wurzeln von in Dunkelheit und in Licht gezogenen Senfsämlingen unterscheiden sich nicht signifikant zu einem α von 5%. Der p-Wert hätte auch mit einem anderen α verglichen werden können, z.B. 10%, also 0.1, dazu wären die Wurzellängen signifikant unterschiedlich gewesen. Diese Entscheidung muss aber a priori, also vor dem Test, gefällt werden.

Die Anzahl der Freiheitsgrade ist im Vergleich zum t-Test reduziert (Prinzip des Welch t-Tests).

```
95 percent confidence interval:
 -21.577530   1.977530
```

Die Null ist im Konfidenzintervall eingeschlossen, d.h. das Testergebnis ist zu einem α von 5% nicht signifikant.

```
sample estimates:
mean in group grown.in.darkness  mean in group grown.with.light
                           24.8                           34.6
```

Die Pflanzengruppe,

✎ Übungsaufgabe 5

die bei Dunkelheit angezogen wurde, hat eine durchschnittliche Wurzellänge von 24.8 cm. Die Gruppe, die belichtet wurde, hat eine durchschnittliche Wurzellänge von 34.6 cm.

Abschließend lässt sich zu diesem Beispiel sagen, dass die Nullhypothese bei einem Konfidenzlevel von 0.95 nicht abgelehnt wird. Das sagt natürlich nichts darüber aus, ob die beiden Stichproben gleich sind, denn der t-Test ist kein Homogenitätstest.

Zwei Salatvarietäten, *Salad Bowl* und *Bibb.*, wurden für 16 Tage in einem kontrollierten Umfeld herangezogen. Data 5.4 zeigt das Trockengewicht der Blätter (Knight and Mitchell, 2000, Die tatsächliche Größe der Stichproben war gleich.) zitiert nach (Samuels and Witmer, 2003, S. 226).

Salad Bowl	Bibb
3.06	1.31
2.78	1.17
2.87	1.72
3.52	1.20
3.81	1.55
3.60	1.53
3.30	
2.77	
3.62	

Data 5.4: Trockengewicht der Blätter zweier Salatvarietäten.

Formulieren Sie geeignete Hypothesen. Liegt Normalverteilung vor? Liegt Varianzhomogenität vor? Für welchen Test entscheiden Sie sich deshalb? Interpretieren Sie den R-Output!

5.2.5 Beispiel Wachstumsinduktion

In einem Experiment sollte die Wachstumsinduktion, ausgelöst durch eine bestimmte Behandlung, untersucht werden. 20 Pflanzen wurden in zwei Gruppen eingeteilt; dabei wurden Paare von Pflanzen gebildet, die sich so ähnlich wie möglich waren. Eine Gruppe wurde behandelt, die andere diente als Kontrollgruppe (Data 5.5) (Mead et al., 2003, S. 72, Daten leicht verändert).

Treated Plant	Control Plant
7	4
10	6
9	10
8	8
7	5
6	3
8	10
9	8
12	8
13	10

Data 5.5: Wachstumsinduktion.

```
> growth <- read.table("../text/growth.txt")
> differences <- growth$height[1:10] - growth$height[11:20]
> boxplot(x = differences, col = "red", ylab = "growth",
+ main = "Pair Differences")
```

- ✔ Annähernde **Gaußverteilung** der Paardifferenzen (Abbildung 5.4, es wird Robustheit des Tests gegenüber dem schief in der Box liegenden Median angenommen).
- ✔ **Gepaarte Daten**, da Pflanzenpaare gebildet wurden, die sich möglichst ähnlich sein sollten.

$\Longrightarrow$ Gepaarter einseitiger t-Test (die eine Pflanzengruppe wurde mit einem Wachstumsinduktor behandelt, was grössere Pflanzen erwarten lässt).

```
> t.test(formula = height~treatment, data = growth , paired = TRUE,
+ alternative = "less")

        Paired t-test
```

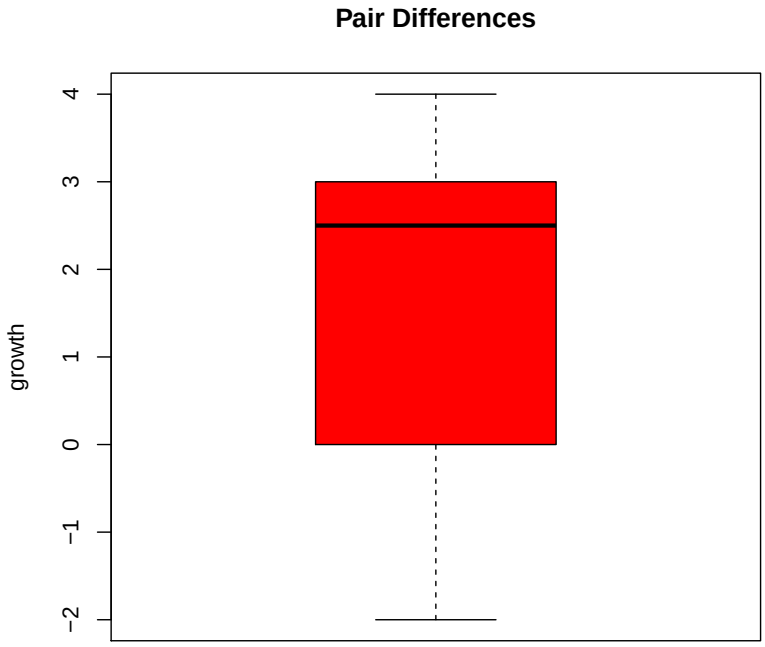


Abbildung 5.4: Boxplots zur Wachstumsinduktion.

```
data:  height by treatment
t = -2.5468, df = 9, p-value = 0.01568
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
       -Inf -0.4763981
sample estimates:
mean of the differences
                  -1.7
```

Der p-Wert ist kleiner als 0.05, deshalb ist das Ergebnis signifkant. Die Behandlung zur Wachstumsinduktion bewirkt ein verstärktes Wachstum.

Zu diesem Ergebnis kann man auch durch die Analyse des Konfidenzintervalls gelangen: Die Null ist nicht im Intervall eingeschlossen, also ist das Ergebnis signifikant. Die mit dem Wachstumsinduktor behandelten Pflanzen sind um mindestens 0.47 cm größer, als die Kontrollgruppe.

Kapitel 6

Wilcoxon-Rangsummentest

6.1 Voraussetzungen

Der t-Test ist nicht robust gegenüber Abweichungen von der Gaußverteilung. Bei Annahme einer unbekannten Verteilung wird deshalb der Wilcoxon-Rangsummentest verwendet. Die Voraussetzungen sind:

- **Varianzhomogenität**.
- Mindestens **ordinale Skalierung**.
- **Unabhängige Daten**.

6.2 Anwendung

6.2.1 Die Funktion `wilcox.test()`

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

oder mit dem `formula`-Konstrukt:

```
wilcox.test(formula, data, subset, na.action, ...)
```

`x` ist ein numerischer Vektor. `y` ist ein optionaler zweiter numerischer Vektor für einen Zweistichprobentest.

Alternativ können die numerischen Vektoren auch mit dem `formula`-Konstrukt angegeben werden (Erläuterung in Abschnitt 3.1).

`alternative` gibt an, ob zweiseitig, einseitig auf Anstieg oder auf Abfall getestet wird.

Mit `paired` lässt sich spezifizieren, ob es sich um einen Datensatz mit Abhängigkeit voneinander handelt (siehe auch Abschnitt 5.1). Der Default-Wert ist `FALSE`.

`correct` indiziert, ob eine Kontinuitätskorrektur angewandt wird. Der Default-Wert ist `TRUE`.

`exact` spezifiziert, ob ein exakter p-Wert berechnet werden soll. Per Default wird ein asymptotischer p-Wert errechnet. Bei einer Fallzahl pro Gruppe kleiner als 50 und ohne

Control	Stress
25.2	24.7
29.5	25.7
30.1	26.5
30.1	27.0
30.2	27.1
30.2	27.2
30.3	27.3
30.6	27.7
31.1	28.7
31.2	28.9
31.4	29.7
33.5	30.0
34.3	30.6

Data 6.1: Mechanischer Stress als Wachstumshemmer. Stängellänge von Sojabohnenpflanzen nach 16 Tagen in cm.

Bindungen sollte ein exakter Test durchgeführt werden. Die Funktion `wilcox.test()` ist beim Vorliegen von Bindungen nicht in der Lage, einen exakten p-Wert zu berechnen. Deshalb wird beim Vorliegen von Bindungen auch kleiner Fallzahl der asymptotische p-Wert berechnet. Das Paket `exactRankTests` löst dieses Problem (siehe Abschnitt 6.2.2).

`conf.int` kann auf `TRUE` gesetzt werden, dann berechnet `wilcox.test()` das Hodges-Lehmann-Konfidenzintervall.

`conf.level` gibt das Konfidenzniveau an. Der Defaultwert ist 0.95.

6.2.2 Die Funktion `wilcox.exact()`

Zur Benutzung der Funktion `wilcox.exact()` muss zunächst die entsprechende Bibliothek mit dem Befehl `library(exactRankTests)` nachgeladen werden. (Hinweise zur Nachinstallation von Paketen in den Abschnitten 1.4.2.1 und 1.4.3.1.)

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

oder auch

```
wilcox.exact(formula, data, subset, na.action, ...)
```

Für Funktion `wilcox.exact()` gilt im Grunde dasselbe, wie für die oben beschriebene Funktion `wilcox.test()`, nur dass sie bei Bindungen in der Lage ist, den exakten p-Wert zu berechnen. Es bietet sich also an, grundsätzlich diese Funktion zu benutzen.

6.2.3 Beispiel mechanischer Stress

6.2.3.1 Versuchsbeschreibung

Ein Pflanzenphysiologe versucht in einem Experiment herauszufinden, ob mechanischer Stress das Wachstum von Sojabohnenpflanzen hemmt. Junge Pflanzen wurden per Zufallsprinzip in zwei Gruppen von je 13 Stück aufgeteilt. Die Pflanzen in der einen Gruppe wurden durch zweimal täglich 20 Minuten Schütteln mechanisch behandelt. Die Pflanzen der Kontrollgruppe wurden nicht geschüttelt. Nach 16 Tagen wurde die Stängellänge jeder Pflanze in cm gemessen, Ergebnis nebenstehend in Data 6.1 (Pappas and Mitchell, 1984), Rohdaten publiziert in (Samuels and Witmer, 2003, S. 302, das tatsächliche Experiment beinhaltete mehrere Pflanzengruppen, die unter verschiedenen Umweltbedingungen herangezogen wurden).

6.2.3.2 Auswertung der Daten

Aus der vorangegangenen Forschung ist ersichtlich, dass die Tendenz kürzerer Pflanzen durch mechanische Stresseinwirkung zu erwarten ist $\Longrightarrow$ einseitiger Test mit folgendem Hypothesenpaar:

$$H_0 : F_{control}(y) \leq F_{stress}(y)$$
$$H_1 : F_{control}(y) > F_{stress}(y)$$

```
> growth.retardant <- read.table(file = "../text/retardant.txt", header = TRUE,
+ sep = "\t")
```

```
> boxplot(formula = responseˆtreatment, data = growth.retardant, col="yellow",
+ ylab ="stem length (cm)", names = c("control","stress"),
+ main = "Stem Length of Soybean Plants (Treated)")
```

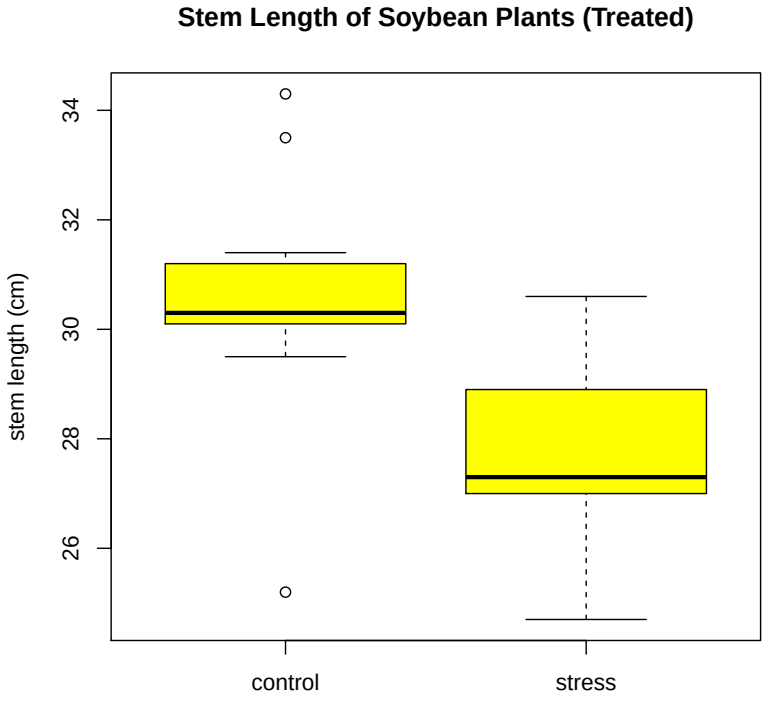


Abbildung 6.1: Boxplots zur Stängellänge von mit seismischem Stress behandelten Sojabohnenpflanzen. Die Daten sind schief verteilt.

- ✔ Stetige Daten (die Länge wurde in cm gemessen).
- ✔ Varianzhomogenität kritisch, Robustheit wird angenommen.
- ✔ Unabhängige Daten (es wurden keine Paare gebildet, die Messungen wurden an unabhängigen Pflanzen vorgenommen).

6.2.3.3 Berechnung des asymptotischen p-Wertes mit der Funktion `wilcox.test`

```
> wilcox.test(formula = responseˆtreatment, data = growth.retardant,
+ correct = FALSE, exact = FALSE, alternative = "greater", conf.int = TRUE)

        Wilcoxon rank sum test

data:  response by treatment
W = 148.5, p-value = 0.0005122
alternative hypothesis: true location shift is greater than 0
95 percent confidence interval:
 1.50005     Inf
```

```
sample estimates:
difference in location
              3.000042
```

Wie beim t-Test wird die Überschrift des Tests, die getesteten Stichproben sowie eine Wiederholung der Alternativhypothese ausgegeben.

```
W = 148.5, p-value = 0.0005122
```

`W` ist die Wilcoxon-Teststatistik. Der `p-value` von 0.0005122 sagt in diesem Fall aus, dass die Pflanzen, die mechanischem Stress ausgesetzt wurden, zu einem Konfidenzniveau von 5% hoch signifikant kürzer sind, als die Pflanzen der Kontrollgruppe.[1]

```
95 percent confidence interval:
 1.500050      Inf
```

Im Konfidenzintervall des Wilcoxon-Rangsummentest ist die Null ist nicht eingeschlossen, d.h. das Testergebnis ist signifikant (zu einem α von 5%).

```
sample estimates:
difference in location
              3.000042
```

gibt den Schätzer für den Lokalisationsunterschied der beiden Verteilungen an.

6.2.3.4 Berechnung des exakten p-Wertes mit der Funktion `exact.wilcox()`

Die Fallzahl je Gruppe ist kleiner als 50; ein exakter Test ist angebracht. Im Datensatz liegen Bindungen vor, daher muss der exakte p-Wert mit dem Paket exactRankTests berechnet werden:

```
> library(exactRankTests)
> wilcox.exact(formula = response~treatment, data = growth.retardant,
+ exact = TRUE, alternative = "greater", conf.int = TRUE)

        Exact Wilcoxon rank sum test

data:  response by treatment
W = 148.5, p-value = 0.0002604
alternative hypothesis: true mu is greater than 0
95 percent confidence interval:
 1.5 Inf
sample estimates:
difference in location
                     3
```

Die Teststatistik `W`, der exakte p-Wert sowie das Konfidenzintervall werden ausgegeben.

6.2.3.5 Schlussfolgerung

Pflanzen, die mit seismischen Stress behandelt wurden, sind mit einer Irrtumswahrscheinlichkeit von 5% signifikant kleiner, als die Pflanzen der unbehandelten Kontrollgruppe.

[1] Aufgrund der kleinen Fallzahl ist hier eigentlich die Berechnung des exakten p-Wertes angebracht.

✎ Übungsaufgabe 6

Ein Forscher untersuchte den Effekt von grünem und rotem Licht auf die Wachstumsrate von Sojabohnenpflanzen. Endpunkt war die Höhe der Pflanzen in inches zwei Wochen nach der Keimung. Die Lichtfarben wurden durch dünnes gefärbtes Plastik, dass u.a. für Theaterscheinwerfer verwendet wird, erzeugt (siehe Data 6.2) ((Gent, 1999), veröffentlicht in (Samuels and Witmer, 2003, S. 243)).

- Mit welchem Test lassen sich diese Daten auswerten?
- Wird einseitig oder zweiseitig getestet?
- Stellen Sie das Hypothesenpaar auf!
- Implementieren Sie den entsprechenden exakten Test, interpretieren Sie den Output!

Red	Green
8.4	8.6
8.4	5.9
10.0	4.6
8.8	9.1
7.1	9.8
9.4	10.1
8.8	6.0
4.3	10.4
9.0	10.8
8.4	9.6
7.1	10.5
9.6	9.0
9.3	8.6
8.6	10.5
6.1	9.9
8.4	11.1
10.4	5.5
	8.2
	8.3
	10.0
	8.7
	9.8
	9.5
	11.0
	8.0

Data 6.2: Höhe von Sojabohnenpflanzen behandelt mit rotem und grünem Licht zwei Wochen nach der Keimung (Maß in Inches).

Kapitel 7

χ^2-Test

7.1 Voraussetzungen

Der χ^2-Test ist ein nicht parametrischer Test, der unter anderem für dichotome Daten geeignet ist. Dichotome Daten sind eine spezielle Art diskreter Daten. Beispielsweise sind Mendels gelbe oder grüne Erbsenfarbe, viel oder wenig Schädlingsbefall, gezackte oder runde Blätter dichotome Endpunkte.

7.1.1 χ^2-Anpassungstest

Der χ^2-Anpassungstest vergleicht eine gemessene Verteilung mit einer bekannten, theoretischen Verteilung. Klassisches Beispiel ist der Vergleich eines empirischen Spaltungsverhältnisses mit einem angenommen Spaltungsverhältnis in der Genetik. Zweiseitiges Hypothesenpaar:

$$H_0 : F_0(x) = F_1(x)$$
$$H_1 : F_0(x) \neq F_1(x)$$

7.1.2 χ^2-Homogenitätstest

Der χ^2-Homogenitätstest testet, ob die prozentualen Verhältnisse zweier Stichproben sich unterscheiden, beispielsweise `kein Befall` und `Befall` bei Behandlung mit und ohne Insektizid.

$$H_0 : \pi_0(x) = \pi_1(x)$$
$$H_1 : \pi_0(x) \neq \pi_1(x)$$

Beide Tests können natürlich auch einseitig berechnet werden.

7.2 Anwendung

7.2.1 χ^2-Anpassungstest - `chisq.test()`

Die Funktion `chisq.test()` wird wie folgt verwendet:

```
chisq.test(x, p = ...)
```

`x` ist ein Vektor, welcher die beobachtete Verteilung enthält.

`p` für *probability* ist ein Vektor derselben Länge wie `x`, welcher die erwartete Verteilung enthält.

7.2.2 χ^2-Homogenitätstest für 2x2-Tafeln - `chisq.test()`

```
chisq.test(x, correct = TRUE)
```

`x` ist eine Matrix in Form der 2x2-Tafel.

`correct` bezieht sich auf die Yates-Korrektur. Sie sollte auf `TRUE` gesetzt werden, wenn die Fallzahl kleiner als 20 ist. Per Default (`FALSE`) wird der Original χ^2-Test nach Pearson gerechnet.

7.2.3 Weitere hilfreiche Funktionen zum χ^2-Test

Die Ausgabe eines p-Wertes zu einem bekannten kritischen Wert und einem bestimmten Freiheitsgrad kann mit der Funktion `pchisq()` erzeugt werden:

```
pchisq(q, df, lower.tail = TRUE)
```

`q` ist der bekannte χ^2-Wert, die Teststatistik.

`df` gibt die Freiheitsgrade an.

Mit `lower.tail` lässt sich die Art der angegebenen Wahrscheinlichkeit angeben. Entweder `TRUE` (entspricht 1 - α) oder `FALSE` (entspricht α). `TRUE` ist Defaultwert. Für einen α-Fehler von 5% wird also 0.95 angegeben.

Die Ausgabe des kritischen Wertes zu einer bekannten Wahrscheinlichkeit und einem bestimmten Freiheitsgrad lässt sich mit der Funktion `qchisq()` erzeugen:

```
qchisq(p, df, lower.tail = TRUE)
```

`p` ist die bekannte Wahrscheinlichkeit.

7.2.4 Beispiel Löwenmäulchen

7.2.4.1 Versuchsbeschreibung

Ein Genetiker untersuchte, ob die Mendelschen Vererbungsregeln auf das Merkmal Blütenfarbe bei der Gartenpflanze Löwenmäulchen zutreffen. Er vermehrte 234 selbstbefruchtende rosa blühende Pflanzen und erhielt die in Tabelle 7.1 gezeigte Spaltung (Baur et al., 1931) zitiert nach (Samuels and Witmer, 2003, S. 392f).

Red	Pink	White
54	122	58

Tabelle 7.1: Das Spaltungsverhältnis der Snapdragon-Blütenfarbe in der F1-Generation.

Unterscheidet sich das beobachtete Ergebnis vom erwarteten Spaltungsverhältnis 1:2:1 der F1 für einen intermediären Erbgang (α-Fehler 5%)?

7.2.4.2 Auswertung der Daten

Die Yates-Korrektur wird nicht benutzt, da mehr als 20 Beobachtungen vorhanden sind.

```
> snapdragon <- c(54,122,58)
> mendel.probs <- c(1,2,1)/4
> chisq.test(x = snapdragon, p = mendel.probs)

        Chi-squared test for given probabilities

data:  snapdragon
X-squared = 0.5641, df = 2, p-value = 0.7542
```

`X-squared` ist der kritische Wert.

`df` gibt die Zahl der Freiheitsgrade aus.

`p-value` gibt den zweiseitigen p-Wert aus. (`chisq.test()` testet grundsätzlich immer zweiseitig.)

7.2.4.3 Interpretation

Das beobachtete Spaltungsverhältnis unterscheidet sich nicht signifikant von Mendels beobachtetem Spaltungsverhältnis für ein Merkmal in der F1-Generation im intermediären Erbgang, die H_0-Hypothese wird beibehalten.

Farbe	Level	Zahl
braun	niedrig	15
braun	mittel	26
braun	hoch	15
gefleckt	niedrig	0
gefleckt	mittel	8
gefleckt	hoch	8

Tabelle 7.2: Das Spaltungsverhältnis der Flachssamen in der F1-Generation.

✎ Übungsaufgabe 7

Forscher untersuchten einen Flachssamenmutantentyp, von dem sie sich die Produktion von Öl für Margarine und Backfett erhofften. Der Gehalt an Palmitin war ein wichtiges Merkmal in dieser Hinsicht. Die Samenfärbung (braun oder gefleckt) wurde damit assoziiert. Die Samen wurden in 6 Kombinationsklassen von Samenfarbe und Palmitinsäuregehalt eingeteilt (siehe Tabelle 7.2) (Saedi and Rowland, 1997) zitiert nach (Samuels and Witmer, 2003, S. 395).

Unterscheidet sich die beobachtete Verteilung vom hypothetischen genetischen Spaltungsmodell 3:6:3:1:2:1?

7.2.5 Beispiel Gerste

7.2.5.1 Versuchsbeschreibung

An Gerstensamen wurde der Effekt einer bestimmten Hitzebehandlung auf die Überlebensrate der Samen untersucht. Probe A diente als unbehandelte Kontrollgruppe, während Probe B mit Hitze behandelt wurde. Alle Samen wurden mit einem Skalpell längs geteilt und für eine halbe Stunde in einer 0.1% 2,3,5-Triphenyltetrazoliumchlorid-Lösung bei Dunkelheit inkubiert. In lebenden Samen reduziert der atmende Embryo das Tetrazoliumchlorid zum intensiv rot gefärbten, unlöslichen Triphenylformazan. Die überlebenden Samen wurden anschließend anhand der Färbung ausgezählt (siehe Tabelle 7.3) (Bishop, 1980, S. 76).

7.2.5.2 Auswertung der Daten

Verringert die Hitzebehandlung die Überlebensrate der Samen? $\alpha = 1\%$.

$$H_0 : \pi_{noheat}(x) \leq \pi_{heat}(x)$$

$$H_1 : \pi_{noheat}(x) > \pi_{heat}(x)$$

Keine Yates-Korrektur, da die Fallzahl ausreichend hoch ist.

	überlebend	**tot**
A	64	16
B	34	46

Tabelle 7.3: Daten zur Überlebensrate von Gerstensamen mit und ohne Hitzebehandlung.

```
> barley <- matrix(c(64,34,16,46), ncol = 2)
> line.names <- c("treatment.A","treatment.B")
> col.names <- c("viable", "not.viable")
> dimnames(barley) <- list(line.names,col.names)
> barley.chi <- chisq.test(barley,correct = FALSE)
> barley.chi

        Pearson's Chi-squared test

data:  barley
X-squared = 23.7, df = 1, p-value = 1.126e-06
```

Die Funktion `chisq.test()` berechnet grundsätzlich den zweiseitigen p-Wert. Man muss daher entweder den p-Wert durch zwei teilen oder ihn mit einem verdoppelten α vergleichen.

```
> barley.p <- barley.chi$p.value/2
> barley.p

[1] 5.629705e-07
```

Ja, die Hitzebehandlung führt mit einem α-Fehler von 1% zu einer signifikanten Verringerung der Überlebensrate der Samen.

✎ Übungsaufgabe 8

	Presence A	**Absence A**
Presence B	25	75
Absence B	25	75

Tabelle 7.4: Daten zur fraglichen Interaktion zweier Spezies in einem Ökosystem.

Einige Spezies zeigen sich in ihrem Auftreten in bestimmten Habitaten assoziiert miteinander. Das mag daran liegen, dass beide ähnlich durch Mikroklimate beeinflusst werden (z.B. Schattenpflanzen treten meist mit anderen Schattenpflanzen zusammen auf), oder an Bodenkonditionen (kalkliebende Pflanzen werden häufig gemeinsam auftreten), oder weil die eine Spezies gute Konditionen für die andere schafft (z.B. Wirts-Parasit-Beziehung), oder aus diversen anderen möglichen Gründen. (...) Die übliche Methode zur Analyse solcher Beziehungen ist das Festlegen von Quadraten innerhalb derer die einzelnen Spezies ausgezählt werden. Ein Beispieldatensatz findet sich in Tabelle 7.4 (Bishop, 1980, S. 111).

Sind die beiden Spezies miteinander assoziiert? $\alpha = 10\%$.

Kapitel 8

Korrelationsanalyse

8.1 Voraussetzungen

Mit Hilfe der Korrelationsanalyse lässt sich quantitativ prüfen, ob ein linearer Zusammenhang zwischen zwei oder mehreren zufälligen Variablen einer Stichprobe vorliegt. Die Funktion des Zusammenhanges lässt sich durch Korrelation allein nicht überprüfen. Der Korrelationskoeffizient r liegt zwischen -1 und +1. Je näher sein Betrag an 1 liegt, desto besser ist die Korrelation. Der Koeffizient ist negativ, wenn die Werte der einen Variablen groß, die der anderen hingegen klein sind. Ein positiver Koeffizient wird ausgegeben, wenn beide Variablen groß oder klein sind.

Der Korrelationskoeffizient macht keine Aussage über die Signifikanz einer Korrelation, deshalb wird ein dem t-Test ähnlicher Test zur Überprüfung verwendet.

8.1.1 Pearson

Voraussetzung für eine Korrelation nach Pearson sind:

- Normal verteilte Daten
- Unabhängigkeit der Observationen voneinander

Der Korrelationskoeffizient nach Pearson wird mit ρ bezeichnet.

8.1.2 Spearman

Die Korrelation nach Spearman ist nicht parametrisch und unabhängig von monotonen Transformationen der Koordinaten. Voraussetzungen:

- keine Gaußverteilung der Daten notwendig
- Unabhängigkeit der Observationen voneinander

8.2 Anwendung

8.2.1 Die Funktion `cor()`

Die Funktion `cor()` wird verwendet wie folgt:

```
cor(x, y = NULL, use = "all.obs",
          method = c("pearson", "spearman"))
```

`x` gibt einen Vektor oder einen Data-Frame an. `y` ist die zweite Variable im Falle der Vektorangabe.

Der Default-Wert für `use` ist `all.obs` (= all observations, engl. für alle Beobachtungen). Bei fehlenden Werten wird eine Fehlermeldung produziert. `pairwise.complete.obs` verwendet alle *vollständigen Paare* der Variablenbeobachtungen.

Mit `method` lässt sich spezifizieren, ob eine Korrelation nach Pearson oder Spearman durchgeführt werden soll.

Die Ausgabe der Funktion ist eine Tabelle mit den Korrelationskoeffizienten aller möglichen Zusammenhänge.

weight (g)	length (cm)
0.7	1.7
1.2	2.2
0.9	2.0
1.4	2.3
1.2	2.4
1.1	2.2
1.0	2.0
0.9	1.9
1.0	2.1
0.8	1.6

Data 8.1: Gewicht und Länge von Puffbohnen.

8.2.2 Die Funktion `cor.test()`

Mit `cor.test()` lässt sich die Signifikanz einer Korrelation testen. Das Hypothesenpaar für den zweiseitigen Test:

$$H_0 : \rho = 0$$

$$H_1 : \rho \neq 0$$

```
cor.test(x, y,
              alternative = c("two.sided", "less", "greater"),
              method = c("pearson", "spearman"),
              conf.level = 0.95, ...)
```

`x, y` gibt zwei Vektoren an. Es kann auch ein `formula`-Konstrukt in folgender Form verwendet werden:

```
formula = ~var1+var2, data = frame.name
```

`method` spezifiziert, ob eine Korrelation nach Pearson oder Spearman's Rangkorrelation verwendet wird.

Mit `conf.level` wird das Konfidenzniveau des Tests angegeben, der Default ist ein α-Fehler von 5%.

8.2.3 Beispiel Puffbohnen

8.2.3.1 Versuchsbeschreibung

Eine Probe Puffbohnen der Varietät *Roger's Emperor* wurde auf Länge und Gewicht hin untersucht (Data 8.1) (Bishop, 1980, S. 64).

8.2.3.2 Auswertung der Daten

```
> broad <- read.table(file = "../text/broad.txt", sep = "\t",
+ header = TRUE)
> plot(length~weight, data = broad, col = "green3", xlab = "length (cm)",
```

```
+ ylab = "weight (g)", main = "Broad Bean Data")
> boxplot(x = broad$weight, broad$length, col = "green3")
> title("Boxplots of the Broad Bean Data")
```

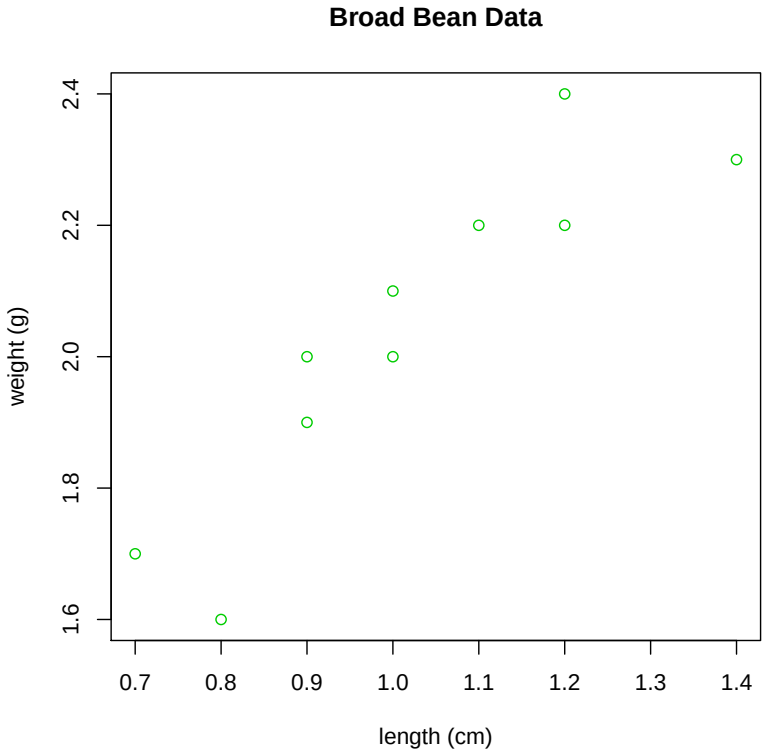


Abbildung 8.1: Scatterplot der Puffbohnendaten.

Abbildung 8.1 zeigt, dass vermutlich ein linearer Zusammenhang mit positivem Korrelationskoeffizient vorliegt (⇒ später einseitiger Test).

- ✔ Gaußverteilung der Variablen (siehe Abbildung 8.2)
- ✔ Unabhängigkeit der Observationen voneinander wird angenommen

$\Longrightarrow$ Korrelation nach Pearson.

Die Ausgabe der möglichen Korrelationskoeffizienten wird erzeugt durch:

```
> cor(broad, method = "pearson")

          weight    length
weight 1.0000000 0.8983172
length 0.8983172 1.0000000
```

Untersuchung, ob die Korrelation von Länge und Gewicht bei broad beans signifikant ist:

```
> cor.test(formula = ~length+weight, data = broad, method = "pearson",
+ alternative = "greater")
```

leaf area	dry weight
411	2.00
550	2.47
471	2.11
393	1.89
427	2.05
431	2.30
492	2.46
371	2.06
470	2.25
419	2.07
407	2.17
489	2.32
439	2.12

Data 8.2: Blattfläche (cm^2) und Trockengewicht (g) von Sojabohnensämlingen.

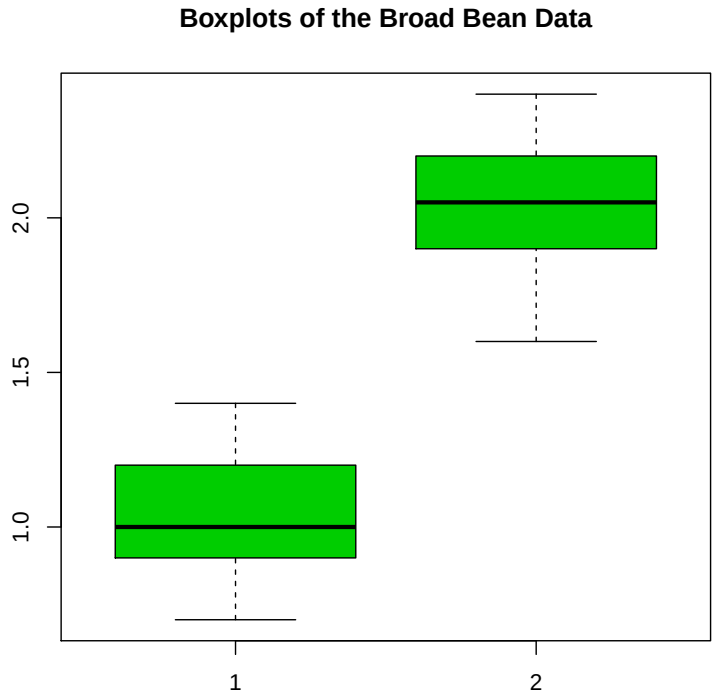


Abbildung 8.2: Boxplot der Puffbohnendaten zur Feststellung der Gaußverteilung.

```
        Pearson's product-moment correlation

data:  length and weight
t = 5.7832, df = 8, p-value = 0.0002065
alternative hypothesis: true correlation is greater than 0
95 percent confidence interval:
 0.6867277 1.0000000
sample estimates:
      cor
0.8983172
```

Die Korrelation der beiden Variablen nach Pearson ist mit dem Koeffizienten (ausgegeben unter `cor`) r = 0.0002065 zu einem Konfidenzniveau von 95% signifikant (der p-Wert ist viel kleiner als 0.5). In Abschnitt 5.2.3.2 wird die Interpretation von Konfidenzintervallen erläutert.

8.2.4 Beispiel Sojabohne

Ein Pflanzenphysiologe zog 13 einzeln getopfte Sojabohnensämlinge in einem Gewächshaus auf. Nach 16 Tagen Wachstum wurde die Blattfläche (cm^2) und das Trockengewicht (g) jeder Pflanze gemessen, Data 8.2 (Pappas and Mitchell, 1984), Rohdaten veröffentlicht in (Samuels and Witmer, 2003, S. 563f, hier leicht verändert, der zweite Trockengewichtswert ist im Original 2.46, nicht 2.47).

Ascorbic acid concentration ($\frac{\mu g}{cm^3}$)	**Messwert**
150	5.9
300	4.8
450	3.7
600	2.4
750	0.9
900	0.0

Data 8.3: Messdaten des Photometers zur Bestimmung des Ascorbinsäuregehaltes anhand des blauen Iod-Stärkekomplexes.

```
> bean <- read.table(file = "../text/bean.txt", sep = "\t", header = TRUE)
> plot(area~weight, data = bean, col = "green3", xlab = "area (squarecm)",
```

```
+ ylab = "dry weight (g)", main = "Soybean Data")
> boxplot(x = bean$area, col = "green3",
+ main = "Leaf Area of Soybean Seedlings", ylab = "area (sqarecm)")
> boxplot(x = bean$weight, col = "green3",
+ main = "Dry Weight of Soybean Seedlings", ylab = "dry weight (g)")
```

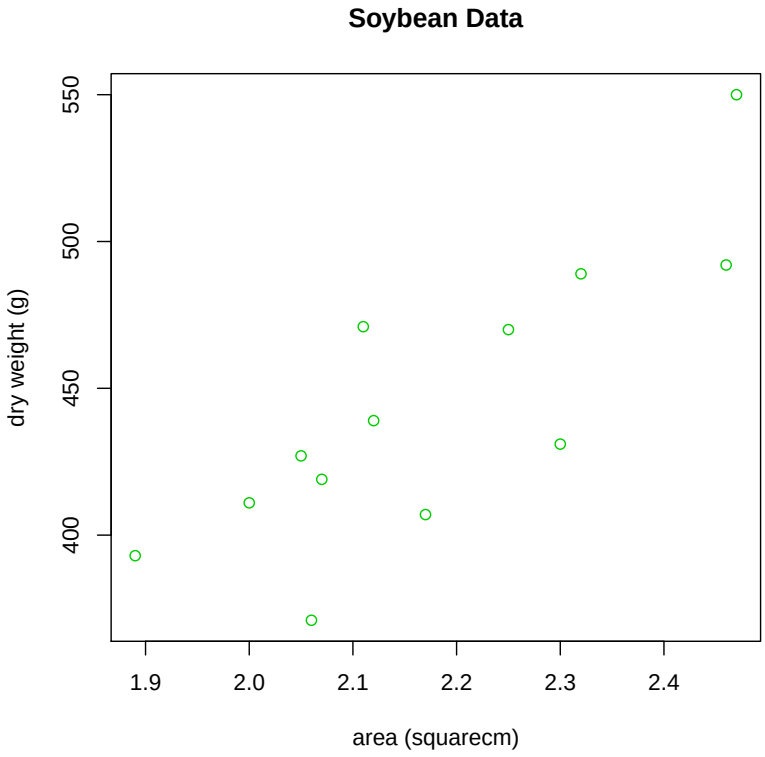


Abbildung 8.3: Scatterplot der Sojabohnendaten.

- ✔ Die Gaußverteilung ist kritisch, da die Mediane nicht mittig in der Box liegen (Abbildungen 8.4 und 8.5).
- ✔ Die Unabhängigkeit der Variablen voneinander ist gegeben.

$\Longrightarrow$ Spearman's Rangkorrelation. Da Abbildung 8.3 einen positiven Korrelationskoeffizienten vermuten lässt, wird einseitig auf Anstieg getestet.

```
> cor.test(formula = ~weight+area, data = bean, method = "spearman",
+ alternative = "greater")

        Spearman's rank correlation rho

data:  weight and area
S = 74, p-value = 0.0009218
alternative hypothesis: true rho is greater than 0
sample estimates:
      rho
0.7967033
```

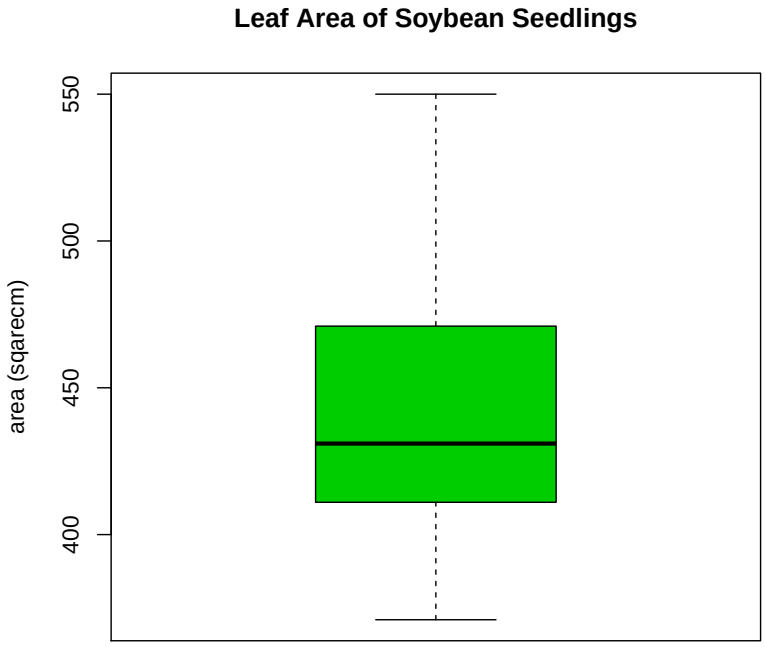


Abbildung 8.4: Boxplot der Blattfläche der Sojabohnensämlinge zur Feststellung der Gaußverteilung.

Der Korrelationskoeffizient ρ ist 0.7967022. Die Korrelation ist mit einer Irrtumswahrscheinlichkeit von 5% signifikant, da der p-Wert von 0.0008658 kleiner als 0.05 ist.

✎ Übungsaufgabe 9

Ascorbinsäure ist quantitiativ messbar durch Entfärbung des blauen Stärke-Iod-Komplexes. Man verwendet dazu ein photoelektrisches Absorbtionmeter. Um die Prozedur zu standardisieren werden zunächst Proben mit bekannter Ascorbinsäurekonzentration gemessen (Ergebnis siehe Data 8.3) (Bishop, 1980, S. 70).

Liegt eine signifkante Korrelation von Ascorbinsäurekonzentration und Gerätemesswert vor?

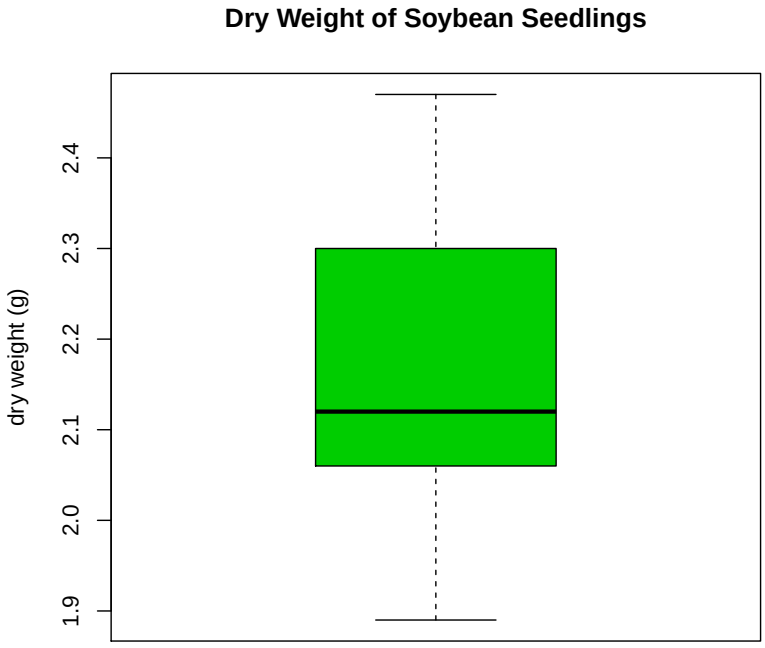


Abbildung 8.5: Boxplot des Trockengewichtes der Sojabohnensämlinge zur Feststellung der Gaußverteilung.

Kapitel 9

Lineare Regressionsanalyse

9.1 Voraussetzungen

Mit der Korrelationsanalyse lässt sich ein linearer Zusammenhang zweier Variablen nachweisen. Durch Regressionsanalyse lässt sich der funktionale Zusammenhang zwischen einer Zielgröße und einer Einflussgröße berechnen.

Im vereinfachten linearen Modell steht α für den Achsenabschnitt, β für die Steigung und ε für den Versuchsfehler (i ist der i-te Messwert):

$$y_i = \alpha + \beta x_i + \varepsilon_i$$

Folgende Voraussetzungen zur Regression sollten erfüllt sein:

- Die **Zahl der Prediktorstufen** muss mindestens zwei sein.
- Die **Zahl der Wiederholungen über alle Messstellen** muss mindestens drei sein.
- **Varianzhomogenität** der Residuen: Im Residuenplot sollte der vertikal eingenommene Bereich, in dem die Residuen liegen, möglichst gleichmäßig sein und nicht an einem Ende eng zulaufen oder in der Mitte extrem schlank werden. Mit dem Levene-Test (Funktion `leveneTest`, enthalten im Paket `car`) lässt sich die Varianzhomogenität von zwei und mehr Gruppen überprüfen.
- **Normalverteilung** der Residuen: Die Residuen sollten im Residuenplot gleichmäßig um die horizontale 0-Linie verteilt sein. In einem Boxplot ist die Normalverteilung der Residuen einfacher zu erkennen, allerdings lässt sich aus einem Boxplot nicht auf die Varianzhomogenität schließen (da es sich um eine einzelne Box handelt).

9.2 Anwendung

9.2.1 Die Funktion `lm()`

`lm()` wird benutzt, um lineare Modelle anzugeben.

```
lm(formula, data, subset, na.action, ...)
```

Mit dem `formula`-Konstrukt lässt sich das Modell angeben (siehe Abschnitt 3.1). Die Funktion gibt den Achsenabschnitt und die Steigung der Geraden aus. Dies allein sagt aber noch nichts über die Signifikanz der Regression aus.

water (mm)	root dry weight (t/ha)
0	9
0	10.3
0	11.5
0	14.2
48	12.2
50	13.8
48	14
50	16.2
88	14
88	14.5
100	15
88	15.3
145	17.8
137	18
150	18.1
153	18.4
177	16.9
189	17.6
200	16.8
200	17
209	18.2
210	17
213	17.5
222	18.5
227	17.2
227	17.4
234	19.2
239	16.8

Data 9.1: Daten zum Ertrag von Zuckerrüben mit unterschiedlichen Bewässerungsstufen.

9.2.2 Die Funktion `summary()`

Mit der Funktion `summary` lässt sich bei Angabe eines linearen Modells (`lm()`) als `object` eine Liste ausgeben, in der u.a. Informationen über die Residuen, den Achsenabschnitt und die Steigung der Geraden gegeben werden.

```
summary(object, ...)
```

9.2.3 Funktionen zur Residuenanalyse

Mit der Funktion `fitted(object, ...)` lassen sich die erwarteten y-Werte eines linearen Modells auf der Regressionsgeraden aufrufen. Mit `resid(object, ...)` werden die tatsächlichen Residuen des Modells aufgerufen.

Um die Verteilung der Residuen graphisch zu überprüfen, lassen sich die Graphikfunktionen `plot()` und `abline()` anwenden (siehe auch Abschnitt 3.3:

```
plot(x, y, ...)
abline(h = 0)
```

Dabei ist `x` der Vektor der Erwartungswerte und `y` ein Vektor mit den tatsächlichen Residuen. Die Punkte in der Graphik sollten gleichmäßig um die 0-Gerade verteilt sein.

Eine andere geeignete Darstellung ist der QQ-Plot (Funktion `qqnorm()`)mit `x` als Vektor der tatsächlichen Residuen:

```
qqnorm(x, ...)
```

Mit der Funktion `qqline()`, angewandt auf das lineare Modell, lässt sich eine Gerade durch die Residuen legen.

Durch das Plotten des linearen Modells (`plot(object = lm(...))`) werden mehrere Graphiken nacheinander ausgegeben: Der oben beschriebene Residuenplot, der QQ-Plot, der Scale-Location[1] Plot und Cook's distance Plot[2] .

```
plot(object, ...)
```

9.2.4 Die Funktion `leveneTest()`

Der Levene-Test kann zur Feststellung der Varianzhomogenität zwischen zwei und mehr Gruppen verwendet werden und ist robuster gegenüber Abweichungen von der Normalverteilung, als der F-Test (nur zwei Stichproben, `var.test`) und Bartlett's Test auf Varianzhomogenität (`bartlett.test()`).

Zur Nutzung der Funktion `leveneTest()` muss das Paket `car` nachinstalliert und mit der Funktion `library()` eingebunden werden!

[1] Der Scale-Location Plot (Streuungsdiagramm) zeichnet die Wurzel der Residuenbeträge gegen die angepassten (fitted) Werte auf. Im Idealfall sollten die Punkte in der Graphik gleichmäßig verteilt sein.

[2] Cook's Distance ist ein anderes Maß für den Einfluss der einzelnen Observation auf die Regressionskoeffizienten. Eine Beobachtung mit einem großen Einfluss verändert die Regressionskoeffizienten stark, wenn sie weggelassen wird.

```
leveneTest(y, group)
```

Vorsicht! Wenn der Gruppierungsvektor den Datentyp `numerical` enthält, dann wird manchmal der p-Wert nicht korrekt berechnet. Dieses Problem lässt sich aber durch `as.character()` lösen. Dies gilt **nur** für die Funktion `leveneTest`.

`y` ist die Zielgröße, z.B. die Residuen, `group` ist ein Gruppierungsvektor. Verschiedene Behandlungen können als Gruppierungsvektor dienen. Es ist jedoch etwas Vorsicht angebracht. Die Verwendung des `formula`-Konstruktes ist derzeit nicht möglich.

Interpretiert wird der Output folgendermaßen: Wenn der p-Wert signifikant ist, dann liegt eine Varianzheterogenität vor. Andernfalls wird von Varianzhomogenität ausgegangen.

9.2.5 Beispiel Zuckerrübe

9.2.5.1 Versuchsbeschreibung

Das Ziel eines Experimentes ist es, herauszufinden ob ein und wenn ja, welcher Zusammenhang zwischen der Bewässerung von Zuckerrüben und ihrem Ernteertrag besteht. Es wurden sieben verschiedene Bewässerungsstufen von 0 bis 250 mm benutzt und jede Behandlung vier mal wiederholt. Die tatsächlich angewendete Wassermenge variierte leicht und die gegebenen Daten berücksichtigen die tatsächliche, nicht die theoretische Wassermenge, Data 9.1 auf der vorhergehenden Seite (Collins and Seeney, 1999, S. 207f, Daten wurden aus Abbildung 6.57 abgelesen und können leicht von den Originaldaten abweichen.).

9.2.5.2 Auswertung der Daten

Die Daten werden aus einer Textdatei im Flatfile-Format eingelesen. Eine Spalte enthält die Wassermenge, die andere den Ertrag.

```
> beets <- read.table(file = "../text/beets.txt", sep = "\t",
+ header = TRUE)
> plot(yield~water, data = beets, col = "turquoise3",
+ xlab = "irrigation (mm)", ylab = "yield (t/ha)",
+ main = "Sugarbeet Irrigation")
```

Das Regressionsmodell wird mit der Funktion `lm()` für lineare Modelle erstellt:

```
> beetmodel <- lm(formula = yield~water, data = beets)
```

Mit der Funktion `abline()` angewandt auf das lineare Modell lässt sich die Regressionsgerade in den Scatterplot einzeichnen (siehe Abbildung 9.1).

```
> abline(reg = beetmodel, col = "turquoise4")
```

9.2.5.3 Residuenanalyse

Residuen entsprechen im Regressionsmodell dem Fehlerterm!

Zur Residuenanalyse wird folgende Graphik erstellt (Abbildung 9.2):

```
> fitted.values <- fitted(object = beetmodel)
> resid.values <- resid(object = beetmodel)
> plot(x = fitted.values, y = resid.values, col = "turquoise3")
> abline(h = 0, col = "turquoise4")
```

Alternativ kann der QQ-Plot (Abbildung 9.3) benutzt werden:

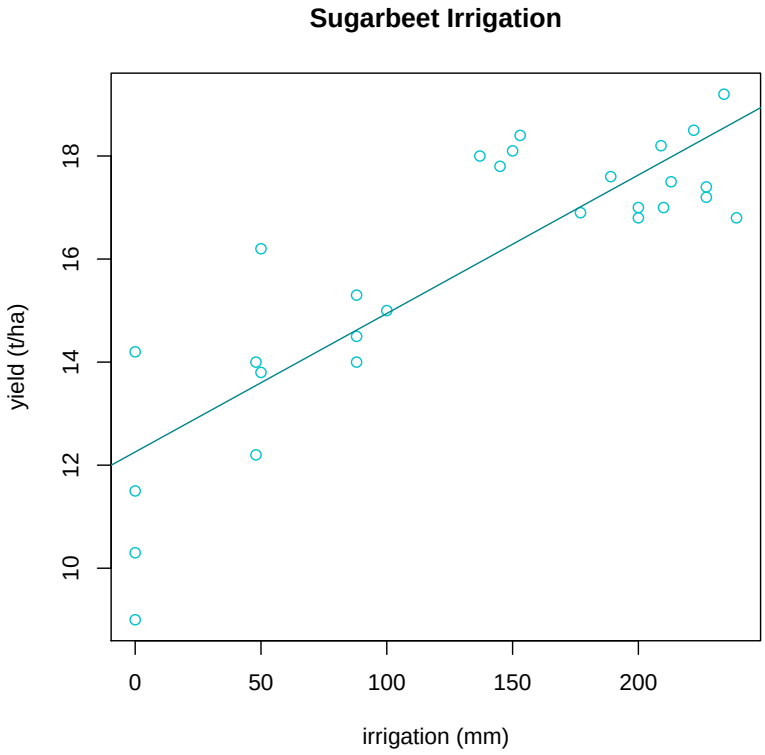


Abbildung 9.1: Die Zuckerrübendaten mit eingezeichneter Regressionsgeraden.

```
> resid.values <- resid(object = beetmodel)
> qqnorm(y = resid.values, col = "turquoise3")
> qqline(y = resid.values, col = "turquoise4")
```

Beide Graphiken zusammen in Begleitung des Scale-Location und Cook's Distance Plot können alternativ durch Plotten des linearen Modells erzeugt werden (Abbildung 9.4):

```
> plot(beetmodel, col = "turquoise3")
```

- ✔ Die **Zahl der X-Werte** ist sieben und damit größer als zwei.
- ✔ Die **Zahl der Wiederholungen** pro Messwert ist vier (das ist größer als drei über alle Messstellen).
- ✔ **Varianzhomogenität** der Residuen (Abbildung 9.2).
- ✔ Eine annähernde **Normalverteilung** der Residuen ist gegeben (Abbildung 9.2 and 9.3).

$\Longrightarrow$ Regressionsanalyse.

```
> summary(object = beetmodel)

Call:
lm(formula = yield ~ water, data = beets)
```

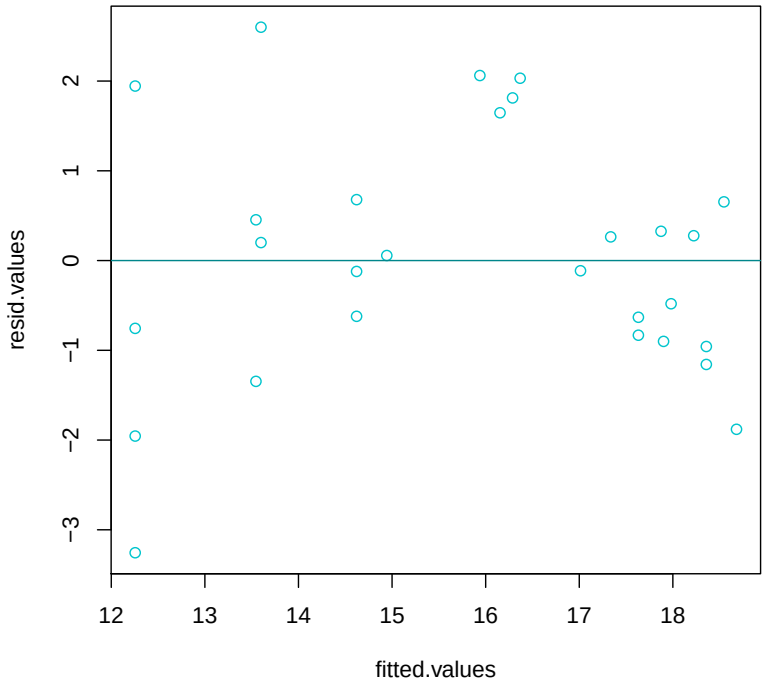


Abbildung 9.2: Residuenplot zum Zuckerrübendatensatz

```
Residuals:
    Min      1Q  Median      3Q     Max
-3.2555 -0.8490 -0.0286  0.6604  2.6004

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.255531   0.505482  24.245   <2e-16 ***
water        0.026881   0.003261   8.244    1e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.407 on 26 degrees of freedom
Multiple R-squared:  0.7233,        Adjusted R-squared:  0.7127
F-statistic: 67.97 on 1 and 26 DF,  p-value: 1.002e-08
```

9.2.5.4 Interpretation der Ausgabetabelle

```
Call:
lm(formula = yield ~ water, data = beets)
```

Das berechnete lineare Modell und der verwendete Datensatz werden wiederholt.

```
Residuals:
     Min       1Q   Median       3Q      Max
-3.25553 -0.84896 -0.02857  0.66045  2.60041
```

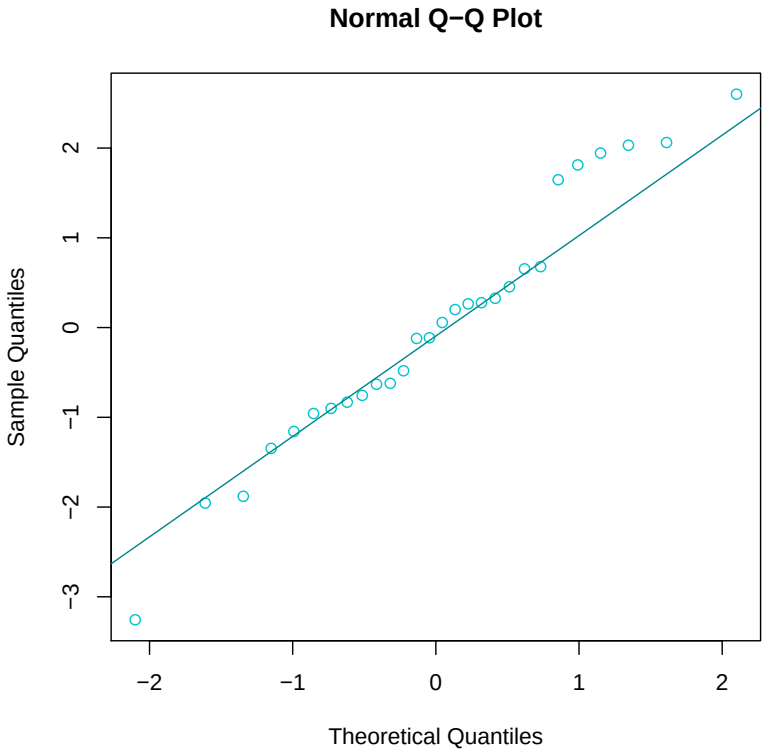


Abbildung 9.3: QQ-Plot der Zuckerrübendaten.

Diese Tabelle gibt Kurzinformationen zu den Residuen. Damit eine Regressionsanalyse aussagekräftig ist, müssen die Residuen normalverteilt sein, d.h. der Minimum- und Maximum-Wert sollte jeweils im Betrag ähnlich sein, der Median etwa um Null liegen. Dies ist hier gegeben.

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.255531   0.505482  24.245  < 2e-16 ***
water        0.026881   0.003261   8.244 1.00e-08 ***
```

Unter `Estimate` – `(Intercept)` wird der Achsenabschnitt der Geraden angezeigt, der Schätzer für `water` gibt die Steigung an. Die Geradengleichung ist also:

$$y = 12.255531 + 0.026881x$$

`Std. Error` gibt die Standardfehler für Achsenabschnitt und Steigung an, unter `t value` verbirgt sich die Teststatistik und `Pr(>|t|)` gibt den p-Wert an. In diesem Fall sind sowohl der Achsenabschnitt als auch die Steigung mit einer Irrtumswahrscheinlichkeit von 5% hoch signifikant.

```
Residual standard error: 1.407 on 26 degrees of freedom
```

Hier wird die Variation der Residuen gezeigt, ein Ausdruck der Variation der Beobachtungen um die Regressionslinie.

```
Multiple R-Squared: 0.7233,     Adjusted R-squared: 0.7127
```

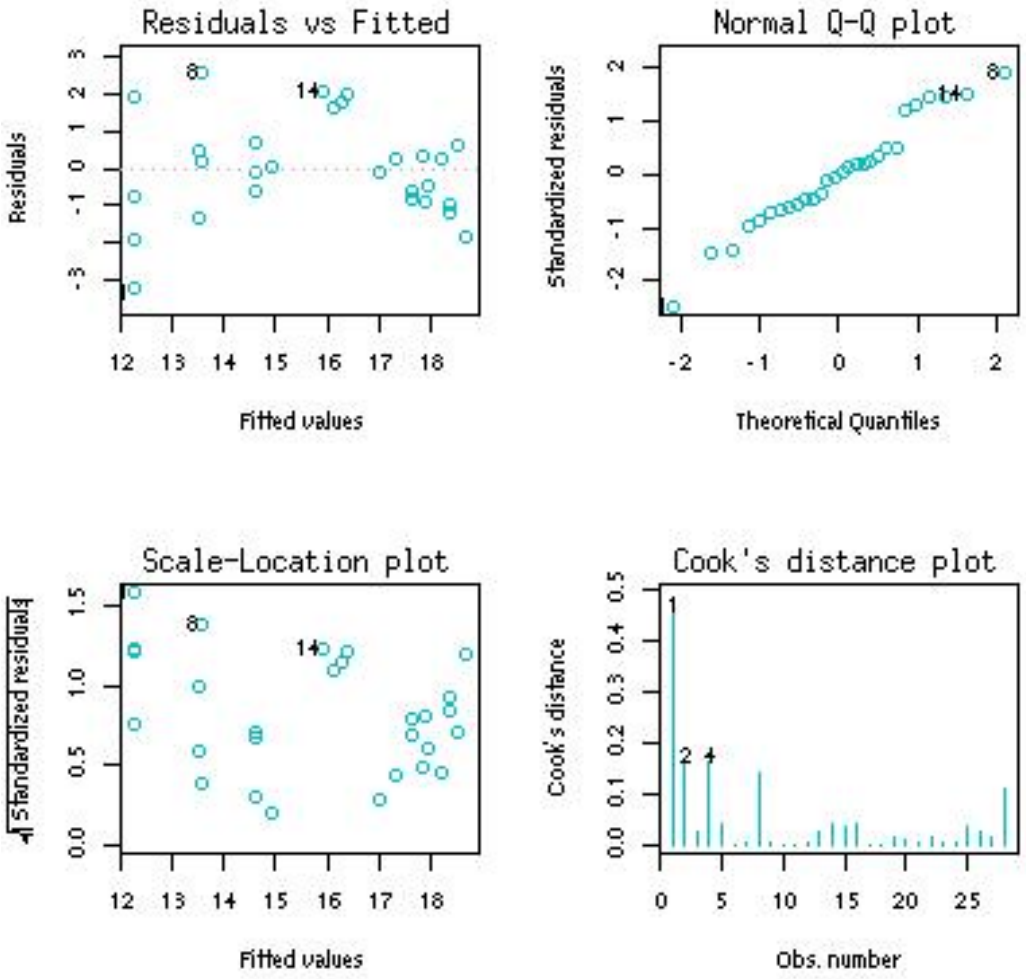


Abbildung 9.4: Ausgabe von `plot(beetmodel)`.

Die erste Angabe ist R^2, der quadrierte Korrelationskoeffizient nach Pearson ($R^2 = r^2$). Das adjustierte R^2 kann als *Prozent Varianzreduktion* interpretiert werden.

```
F-statistic: 67.97 on 1 and 26 DF,  p-value: 1.002e-08
```

Dies ist ein F-Test für die Hypothese, dass der Regressionskoeffizient gleich Null ist. Der Test ist in diesem Fall uninteressant, da er beim linearen Modell nur ohnehin schon vorhandene Information dupliziert. Interessanter wird es, wenn der Einfluss mehrerer Variablen zu klären ist.

9.2.5.5 Konfidenz- und Vorhersagebänder

Abbildung 9.5 zeigt die Konfidenz- und Vorhersagebänder.

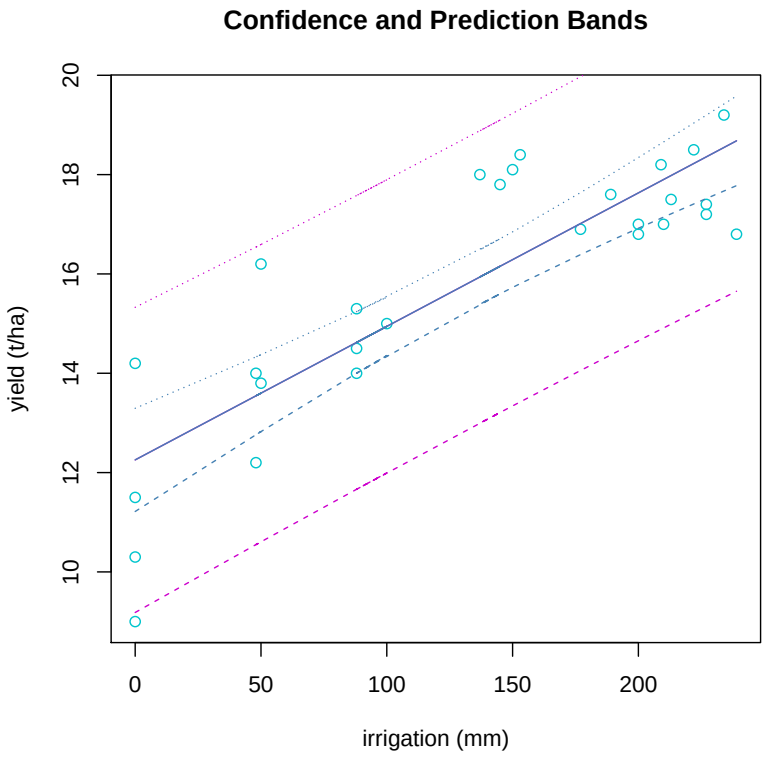


Abbildung 9.5: Darstellung der Konfidenz- und Vorhersagebänder zum Zuckerrübendatensatz. Die weit auseinander liegenden, rosa Linien sind die Vorhersagebänder, die engeren, blauen Linien die Konfidenzbänder.

Mit der Funktion `predict()` lassen sich die Vorhersagedaten des linearen Modells zu den Koordinaten der x-Achse machen. Der Parameter `interval` spezifiziert, welche Art von Konfidenzwerten erstellt werden soll: `confidence` steht für den Bereich, in dem die Gerade aufgrund der gegebenen Versuchsdaten mit einer Sicherheit von 95% liegt. Die mit der Option `prediction` erstellten Konfidenzdaten dienen der Erstellung von Vorhersagebändern. Diese schließen die Mehrheit aller beobachteten Punkte ein und zeigen die Sicherheit, mit der man für weitere Vorhersagen exakte Werte ermitteln kann.

```
> pp <- predict.lm(object = beetmodel, interval = "prediction",
+ data = beets$water)
```

Fertilizer (lb/acre)	Yield
100	24
100	35
100	42
100	47
100	55
200	31
200	40
200	50
200	54
200	61
300	37
300	43
300	53
300	55
300	62
400	47
400	53
400	62
400	70
400	74
500	52
500	61
500	65
500	70
500	80
600	63
600	68
600	74
600	80
600	90
700	67
700	74
700	80
700	84
700	93

Data 9.2: Daten zum Ertrag von Weizen bei unterschiedlichen Düngerstufen.

```
> pc <- predict.lm(object = beetmodel, interval = "confidence",
+ data = beets$water)
```

Mit der Funktion `matlines()` lassen sich die Bänder graphisch darstellen:

```
> plot(x = beets$water, y = beets$yield, ylim = range(beets$yield, pc),
+ col = "turquoise3", xlab = "irrigation (mm)", ylab = "yield (t/ha)",
+ main = "Confidence and Prediction Bands")
> matlines(x = beets$water, y = pp, tly = c(1,3), col = "magenta3")
> matlines(x = beets$water, y = pc, tly = c(1,2,3), col = "steelblue")
```

Das Argument `tly` gibt an, welche Spalten der `predict`-Tabellen geplottet werden sollen.

9.2.6 Beispiel Weizen

9.2.6.1 Versuchsbeschreibung

In einem Experiment wurde die Wirkung unterschiedlicher Düngermengen auf den Ertrag von Weizen untersucht. Die Konzentrationen 100, 200, 300, 400, 500, 600 und 700 lb/acre wurden auf jeweils fünf zufällig ausgewählten Plots angewendet (Data 9.2) (Wonnacott and Wonnacott, 1990, S. 359, die Daten wurden aus Graph 11-1 abgelesen, leichte Abweichungen von den Originaldaten sind möglich.).

9.2.6.2 Auswertung der Daten

```
> wheat <- read.table(file = "../text/wheat.txt", header = TRUE,
+ sep = "\t")
> plot(yield~fertilizer, data = wheat, col = "turquoise3",
+ xlab = "Fertilizer (lb/acre)",
+ main = "Wheat Yield in a Fertilizer Experiment")
```

Mit der Funktion `lm` wird ein Regressionsmodell erstellt:

```
> wheatmodel <- lm(formula = yield~fertilizer, data = wheat)
```

`abline()` fügt eine Regressionsgerade in den Scatterplot ein (Abb. 9.6):

```
> abline(reg = wheatmodel, col = "turquoise4")
```

Ein Boxplot dient der Feststellung über die Normalverteilung der Residuen:

```
> resid.values <- resid(object = wheatmodel)
> boxplot(x = resid.values, col = "turquoise3",
+ main = "Boxplot of Residuals")
```

Da die Varianzhomogenität der Residuen nicht im Boxplot erkannt werden kann (nur eine Box), wird ein Levene-Test durchgeführt. Für die Funktion `leveneTest` muss das Paket `car` installiert sein!

```
> library(car)
> lev <- data.frame(res = resid.values,
+ group = as.character(wheat$fertilizer))
> leveneTest(y = lev$res, group = lev$group)
```

Abbildung 9.6: Der Ertrag von Weizen in Abhängigkeit von der Düngermenge (lb/acre).

```
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  6  0.0623 0.9989
      28
```

- ✔ Die Zahl der **Messstellen** ist sieben (> 2).
- ✔ Die Zahl der Wiederholungen pro Messwert ist fünf (> 3 über alle Messstellen).
- ✔ Varianzhomogenität der Residuen ist aufgrund des nicht signifikanten Levene-Tests gegeben.
- ✔ Die Residuen sind annähernd normalverteilt (Abbildung 9.7).

$\Longrightarrow$ Die Daten sind für eine Regressionsanalyse mit dem linearen Modell `wheatmodel` geeignet.

```
> summary(object = wheatmodel)

Call:
lm(formula = yield ~ fertilizer, data = wheat)

Residuals:
     Min       1Q   Median       3Q      Max
-16.1643  -6.6643   0.6714   7.4179  16.6714
```

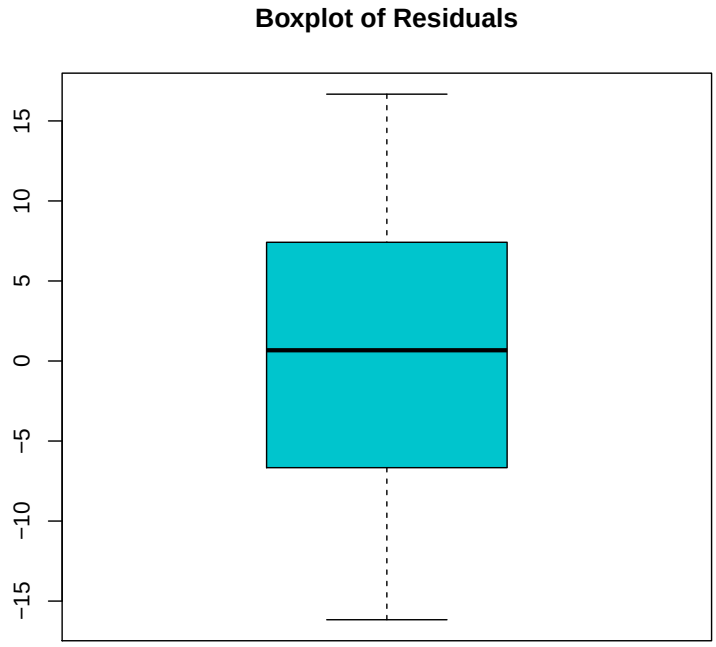


Abbildung 9.7: Boxplot der Residuen zur Untersuchung ihrer Normalverteilung.

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 33.000000   3.822172   8.634 5.60e-10 ***
fertilizer   0.067214   0.008547   7.864 4.57e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.11 on 33 degrees of freedom
Multiple R-squared:  0.6521,        Adjusted R-squared:  0.6415
F-statistic: 61.85 on 1 and 33 DF,  p-value: 4.574e-09
```

9.2.6.3 Interpretation der Ausgabetabelle

Die Gleichung der Regressionsgeraden lautet:

$$y = 0.067214x + 33$$

Sowohl der Achsenabschnitt als auch die Steigung sind zu einem Konfidenzniveau von 95% hoch signifikant.

✎ Übungsaufgabe 10

Sulfur ist ein wirksames Mittel gegen Kartoffelschorf. In einem Experiment wurde der Effekt verschiedener Konzentrationen auf die Kartoffelkrankheit durch Anwendung von

sulphur (pound/acre)	scab (%)
0	18
0	30
0	24
0	29
300	9
300	9
300	16
300	4
600	18
600	10
600	18
600	16
1200	4
1200	10
1200	5
1200	4

Data 10.3: Daten zur Sulfurbehandlung von Kartoffelschorf.

vier Konzentrationsstufen (0, 300, 600 und 1200 pounds/acre) untersucht. Der Schwefel wurde im Herbst auf den Boden ausgebracht. Von allen Versuchsblöcken wurden zufällig 100 Kartoffeln ausgewählt und auf den Prozent an Oberflächenschädigung durch Schorf untersucht (Data 10.3) (Pearce, 1983, S. 46, Datensatz hier unvollständig wiedergegeben. Das tatsächliche Experiment umfasste Applikationen im Frühling und im Herbst.), Originalexperiment publiziert in (Cochran and Cox, 1950).

Sind die Daten für eine Regressionsanalyse geeignet?

Wenn ja, führen Sie eine Regressionsanalyse durch. Wie lautet der Achsenabschnitt und die Steigung? Ist das Regressionsmodell zu einem Konfidenzniveau von 95% signifikant? Sind die Residuen normalverteilt?

Lassen Sie die Konfidenz- und Vorhersagebänder zeichnen!

Kapitel 10

ANOVA

10.1 Voraussetzungen

Die Varianzanalyse (ANOVA) wird verwendet, um den Einfluss ein oder mehrerer gestufter Einflussgrößen auf ein oder mehrere Zielgrößen zu analysieren, z.B. den Einfluss von verschiedenen Sorten und Düngern auf den Ertrag. Zur Unterschiedermittlung werden die Varianzen der einzelnen Gruppen herangezogen. Wenn die Varianzen sich überschneiden, liegt kein Unterschied vor.

Ein Beispiel für ein Modell – zweifaktorielle ANOVA mit Wechselwirkungsterm:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \varepsilon_{ijk}$$

Y_{ijk} ist die Zufallsvariable, μ der Erwartungswert, α_i der Effekt der i-ten Stufe von Faktor A, β_j der Effekt der j-ten Stufe von Faktor B, $(\alpha\beta)_{ij}$ die Wechselwirkung, ε_{ijk} der Versuchsfehler und k die Wiederholungsanzahl.

Testvoraussetzung für die ANOVA sind:

- **Normalverteilung von** ε_{ijk} innerhalb der Gruppen → Residuenplot, Punkte sollten gleichmäßig nach oben und unten um die Linie verteilt sein. Auch der Boxplot ist zur Beurteilung der Normalverteilung geeignet.
- **Varianzhomogenität** der Residuen → Levene-Test und/oder Residuenplot.
- **unabhängige Daten**

Ein Beispiel für die Formulierung der Hypothesen mit drei Stufen des Faktors A und zwei Stufen des Faktors B:

$\exists$ wird als *es existiert* gelesen.

$$\begin{array}{lll} H_0^1: & \mu_{A1} = \mu_{A2} = \mu_{A3} & H_0^2: \mu_{B1} = \mu_{B2} \\ H_1^1: & \exists \text{ mindestens ein } \mu_{A_i} \neq \mu_{A_j} & H_1^2: \mu_{B1} \neq \mu_{B2} \end{array}$$

10.2 Anwendung

10.2.1 Die Erweiterung der Funktion `lm()`

Die Funktion `lm()` wurde bereits in Abschnitt 9.2.1 beschrieben. Zur Varianzanalyse wird das Modell mit dem `formula`-Konstrukt (siehe Abschnitt 3.1) angegeben:

```
lm(target~treatment.1+treatment.2+treatment.1:treatment.2, data = dataset)
```

Mit einem + lassen sich die verschiedenen Einflussfaktoren kombinieren, mit einem Doppelpunkt : wird das Wechselwirkungsmodell zusammengesetzt.

10.2.2 Die Funktion `anova()`

Die Funktion `anova()` berechnet die Varianztabelle.

```
anova(object, ...)
```

`object` ist ein lineares Modell. Man kann dieses Modell entweder in einem Objekt speichern und dann mit `anova(objectname)` aufrufen, oder das Modell in die ANOVA-Funktion integrieren: `anova(lm(...))`.

10.2.3 Beispiel Mais

10.2.3.1 Versuchsbeschreibung

Können biologische Methoden der Schadinsektkontrolle bei Mais erfolgreich angewendet werden, und kann der Effekt der Schaderreger auf Mais durch diese Methoden erfolgreich verringert werden? In einem Experiment zu dieser Fragestellung verglichen Forscher das Kolbengewicht von Mais unter fünf verschiedenen biologischen Behandlungen. Verwendet wurden der nützliche Nematode *Steinernema carpocapsae*, die Schlupfwespe *Trichogramma pretiosum*, eine Kombination der ersten beiden, *Bacillus thuringiensis* und eine Kontrollgruppe. Die Maiskolben wurden zufällig aus jedem Plot ausgewählt und gewogen. Das Resultat ist in Tabelle 10.1 abgebildet (Martinez, 1998) zitiert nach (Samuels and Witmer, 2003, S. 463f, die hier angegebenen Daten sind nur eine zufällige Stichprobe aus einer umfassenderen Studie.).

Nematode	Wasp	Nematode & Wasp	Bacterium	Control
16.5	11.0	8.5	16.0	13.0
15.0	15.0	13.0	14.5	10.5
11.5	9.0	12.0	15.0	11.0
12.0	9.0	10.0	9.0	10.0
12.5	11.5	12.5	10.5	14.0
9.0	11.0	8.5	14.0	12.0
16.0	9.0	9.5	12.5	11.0
6.5	10.0	7.0	9.0	18.5
8.0	9.0	10.5	9.0	9.5
14.5	8.5	10.5	9.0	17
7.0	8.0	13.0	6.5	10.0
10.5	5.0	9.0	8.5	11.0

Tabelle 10.1: Gewicht der Maiskolben in Unzen.

```
> corn <- read.table(file = "../text/corn.txt", sep = "\t", header = TRUE)
```

Zur Visualisierung der Daten werden die Boxplots erstellt (Abbildung 10.1):

```
> plot(response~treatment, data = corn, col = "blue3",
+ main = "Plot of the Corn Data", ylab = "weight (ounces)",
+ names = c("nem","wasp","nem+wasp","bac","control"))
```

In diesem Experiment wurde nur ein Einflussfaktor berücksichtigt. Im linearen Modell gibt es deshalb weder additive Einflussgrößen noch eine Wechselwirkung:

```
> corn.model <- lm(formula = response~treatment, data = corn)
```

Graphische Darstellung der Residuen (Abbildung 10.2):

```
> fitted.values <- fitted(object = corn.model)
> resid.values <- resid(object = corn.model)
> plot(x = fitted.values, y = resid.values, col = "blue3")
> abline(h = 0, col = "blue4")
```

Einfacher ist die Prüfung der Residuen auf ihre Verteilung mit einem Boxplot (Abb. 10.3):

```
> boxplot(x = resid.values, col="blue3", main="Residuals")
```

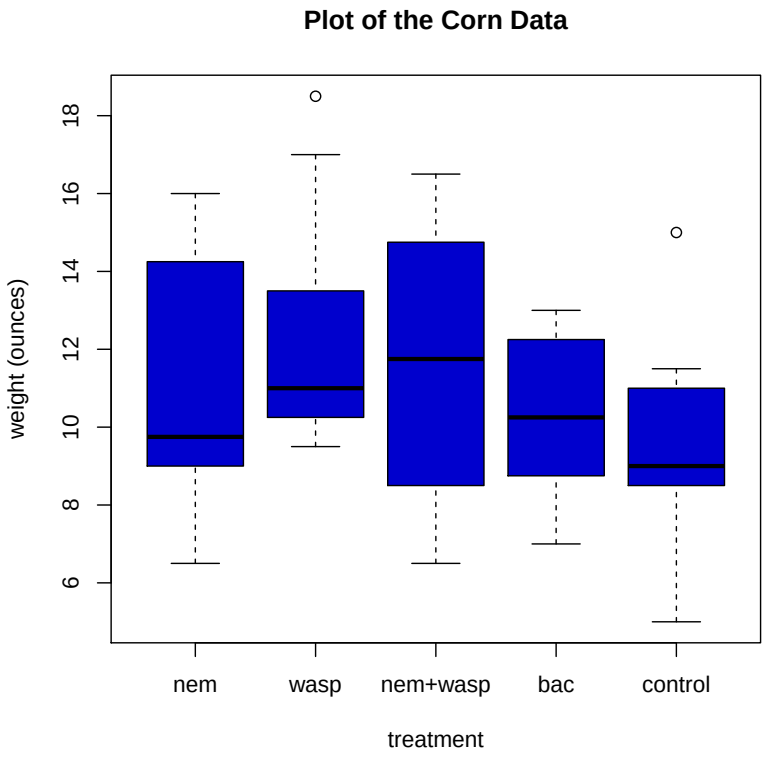


Abbildung 10.1: Plot der Maisdaten.

Benutzung des Levene-Tests zur Prüfung auf Varianzhomogenität der Residuen innerhalb der Gruppen (das Paket `car` muss installiert sein!):

```
> library(car)
> lev <- data.frame(res = resid.values, group = corn$treatment)
> leveneTest(y = lev$res, group = lev$group)

Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  4  1.1028 0.3645
      55
```

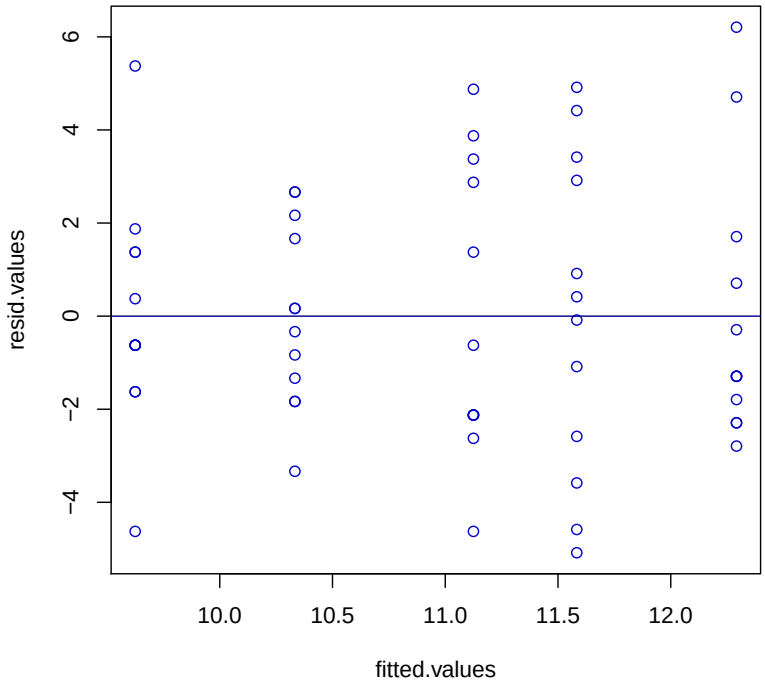


Abbildung 10.2: Residuenplot der Maisdaten zur Feststellung der Normalverteilung des Fehlerterms.

Die Nullhypothese, dass Varianzhomogenität vorliegt, wird beibehalten.

- ✔ Varianzhomogenität (Levene-Test)
- ✔ Normalverteilung des Fehlerterms ist in Abbildung 10.2 und 10.3 gezeigt
- ✔ Unabhängige Daten

$\Longrightarrow$ ANOVA mit einem Einflussfaktor. Die Hypothesenpaare lauten:

$$H_0: \quad \mu_{nem} = \mu_{wasp} = \mu_{nem+wasp} = \mu_{bac} = \mu_{control}$$
$$H_1: \quad \exists \text{ mindestens ein } \mu_{treatment} \neq \mu_{treatment'}$$

```
> anova(object = corn.model)

Analysis of Variance Table

Response: response
          Df Sum Sq Mean Sq F value Pr(>F)
treatment  4  52.31 13.0771  1.6461 0.1758
Residuals 55 436.94  7.9443
```

10.2.3.2 Interpretation des Outputs

Zuerst wird die Überschrift der Varianztabelle sowie die Zielgröße angegeben.

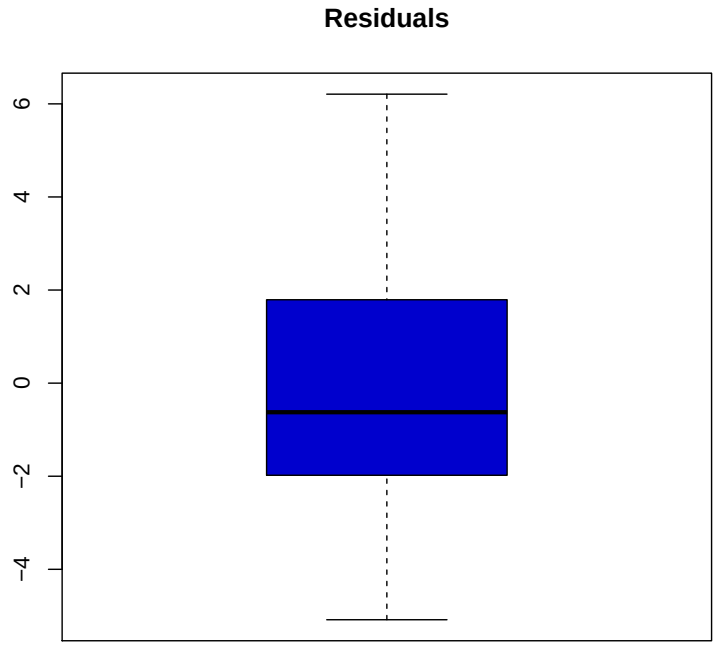


Abbildung 10.3: Boxplot der Residuen.

```
          Df Sum Sq Mean Sq F value Pr(>F)
treatment  4  52.31   13.08  1.6461 0.1758
Residuals 55 436.94    7.94
```

Die erste Spalte der Varianztabelle zeigt die Ursache der Variation auf. Es folgen die jeweiligen Freiheitsgrade. Die Summe der Abweichungsquadrate ist mit `Sum Sq` bezeichnet, die mittlere Quadratabweichung heißt `Mean Sq`. Des Weiteren sind die Teststatistik – der bereits aus `var.test()` bekannte F-Wert – und der p-Wert, welcher mit dem gewünschten Signifikanzniveau verglichen wird (z.B. $\alpha = 0.05$), aufgeführt. In diesem Fall ist der p-Wert größer als 0.05 und damit kann zu einem Konfidenzniveau von 95% kein signifikanter Unterschied zwischen den biologischen Behandlungen zur Schädlingsbekämpfung nachgewiesen werden.

Die Sternchen hinter den Zeilen zeigen auf einen Blick, welches Signifikanzniveau jeweils eingehalten wird:

```
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1
```

Ein Sternchen steht für kleiner als 0.05, zwei Sternchen für kleiner als 0.01 usw..

10.2.4 Beispiel Sojabohnen

10.2.4.1 Versuchsbeschreibung

Ein Pflanzenphysiologe untersuchte den Effekt von mechanischem Stress auf das Wachstum von Sojabohnenpflanzen. Einzeln getopfte Sämlinge wurden zufällig in vier Behand-

lungsgruppen à 13 Pflanzen eingeteilt. Sämlinge in zwei Gruppen wurden zweimal täglich für zwanzig Minuten durch Schütteln gestresst, während die Kontrolle diesem Stress nicht ausgesetzt wurde. Der erste Faktor in diesem Experiment ist der Schüttelstress. Des Weiteren wurden die Pflanzen mit viel und wenig Licht behandelt $\Longrightarrow$ zweiter Faktor. Die Blattfläche der vier Behandlungen ist in Tabelle 10.2 gezeigt (Pappas and Mitchell, 1984), Rohdaten publiziert in (Samuels and Witmer, 2003, S. 491, der Autor gibt an, dass das Originalexperiment mehr als vier Behandlungen umfasste).

Control Low Light	Stress Low Light	Control Moderate Light	Stress Moderate Light
264	235	314	283
200	188	320	312
225	195	310	291
268	205	340	259
215	212	299	216
241	214	268	201
232	182	345	267
256	215	271	326
229	272	285	241
288	163	309	291
253	230	337	269
288	255	282	282
230	202	273	257

Tabelle 10.2: Blattfläche von Sojabohnenpflanzen (cm^2).

```
> soybeans <- read.table(file = "../text/soybeans.txt", sep = "\t",
+ header = TRUE)
```

Aufstellung eines linearen Modells, in dem die Blattfläche durch Licht, seismischen Stress sowie einen Wechselterm aus Licht und seismischem Stress beeinflusst wird:

```
> model <- lm(formula = response~treatment.B+treatment.A+treatment.A:
+ treatment.B, data = soybeans)
```

Graphische Darstellung der Residuen (Abbildung 10.4):

```
> fitted.values <- fitted(object = model)
> resid.values <- resid(object = model)
> plot(x = fitted.values, y = resid.values, col = "blue3")
> abline(h = 0, col = "blue4")
```

Durchführung des Levene-Tests zur Klärung der Varianzhomogenität innerhalb der Gruppen:

```
> library(car)
> lev <- data.frame(res = resid.values, group = rep(c("low.light.c",
+ "low.light.s","mod.light.c","mod.light.s"), each = 13))
> leveneTest(y = lev$res, group = lev$group)

Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  3  0.1963 0.8984
      48
```

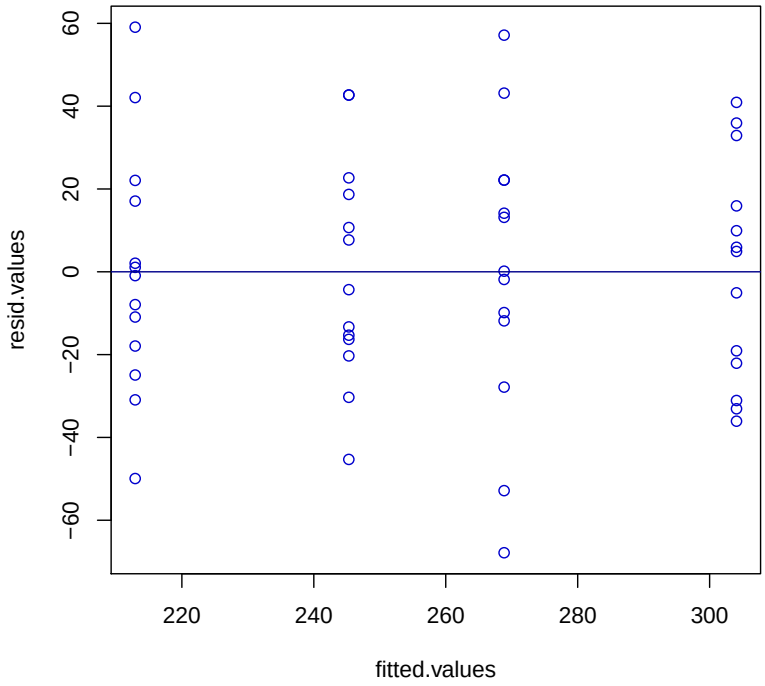


Abbildung 10.4: Residuenplot zu den Sojabohnendaten.

Der p-Wert ist größer als 0.05, damit wird die Nullhypothese (es herrscht Varianzhomogenität) beibehalten.

- ✔ Die Varianzhomogenität der Residuen für die einzelnen Gruppen wird aufgrund des Levene-Tests angenommen.
- ✔ Die annähernde Normalverteilung des Fehlerterms ist in Abbildung 10.4 zu erkennen.
- ✔ Die Daten sind unabhängig (randomisierte Gruppen).

$\Longrightarrow$ Auswertung mit ANOVA. Fragestellung: Führt mechanischer Stress und Lichtentzug zu mindestens einem Unterschied in den Versuchsgruppen? Die Hypothesenpaare (einschließlich Interaktionshypothese) lauten:

$$H_0^A : \mu_{stress} = \mu_{nostress} \qquad H_0^B : \mu_{light} = \mu_{dark}$$
$$H_1^A : \mu_{stress} \neq \mu_{nostress} \qquad H_1^B : \mu_{light} \neq \mu_{dark}$$

$$H_0^{A'B}: \quad \mu_{stressfactor,lightfactor} = \mu_{stressfactor} + \mu_{lightfactor} - \mu$$
$$H_1^{A'B}: \quad \mu_{stressfactor,lightfactor} \neq \mu_{stressfactor} + \mu_{lightfactor} - \mu$$

```
> anova(object = model)

Analysis of Variance Table

Response: response
```

```
                        Df Sum Sq Mean Sq F value    Pr(>F)
treatment.B              1  14858   14858 16.5954 0.0001725 ***
treatment.A              1  42752   42752 47.7490  1.01e-08 ***
treatment.B:treatment.A  1     26      26  0.0294 0.8645695
Residuals               48  42976     895
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

10.2.4.2 Interpretation des Outputs

Aus der Varianztabelle (Interpretation siehe vorhergehendes Beispiel) lässt sich ablesen, dass sowohl ein signifkanter Einfluss der Lichtbehandlung als auch des seismischen Stresses, auf die Blattfläche von Sojabohnensämlingen vorliegt. Es gibt keine signifikante Wechselwirkung. In diesem Fall ist es eindeutig, wo die Signifikanz liegt, da es nur je zwei Alternativhypothesen gibt.

10.2.5 Beispiel Luzerne

10.2.5.1 Versuchsbeschreibung

Forscher waren am Effekt von Säure auf die Wachstumsrate von Luzerne interessiert. Sie untersuchten die drei Gruppen *wenig Säure* (drei Tropfen 1.5 M HCl auf zwei Tropfen Wasser), *viel Säure* (Verwendung von 3 M HCl in derselben Mixtur) und *Kontrolle* (nur Wasser). Der Zielpunkt war die Höhe der Luzernepflanzen in einer Styroporschale nach fünf Tagen. (Nicht die einzelne Pflanze, sondern die Gesamtheit der Pflanzen pro Schale wurde gemessen.) Es gab fünf Styroporschalen pro Behandlung, also eine Gesamtfallzahl von 15.

Die Schalen wurden in der Nähe eines Fensters aufgestellt. Der dadurch bedingten leicht variierenden Lichtmenge sollte Rechnung getragen werden, indem im Blockdesign jeder Block zufällig verteilt die drei Behandlungen enthält (Tabelle 10.3). Die Daten des Experiments sind in Tabelle 10.4 auf Seite 74 dargestellt (Neumann et al., 2001) zitiert nach (Samuels and Witmer, 2003, S. 487f).

	Block 1	**Block 2**	**Block 3**	**Block 4**	**Block 5**
win	high	control	control	control	high
d	control	low	high	low	low
ow	low	high	low	high	control

Tabelle 10.3: Blockdesign des Luzerneexperimentes.

	Low acid	**High Acid**	**Control**
Block 1	1.58	1.10	2.47
Block 2	1.15	1.05	2.15
Block 3	1.27	0.50	1.46
Block 4	1.25	1.00	2.36
Block 5	1.00	1.50	1.00

Tabelle 10.4: Daten des Luzerneexperimentes, Höhe der Pflanzen pro Schale nach fünf Tagen in cm.

```
> alfalfa <- read.table(file = "../text/alfalfa.txt", sep = "\t",
+ header = TRUE)
```

Die Aufstellung eines linearen Modells, in dem die Höhe durch die Säurebehandlung und Block beeinflusst wird:

```
> alfalfa.model <- lm(formula = height~acid+block, data = alfalfa)
```

Boxplot der Residuen (Abbildung 10.5):

```
> resid.values <- resid(object = alfalfa.model)
> boxplot(x = resid.values, col = "blue3", main="Residuals")
```

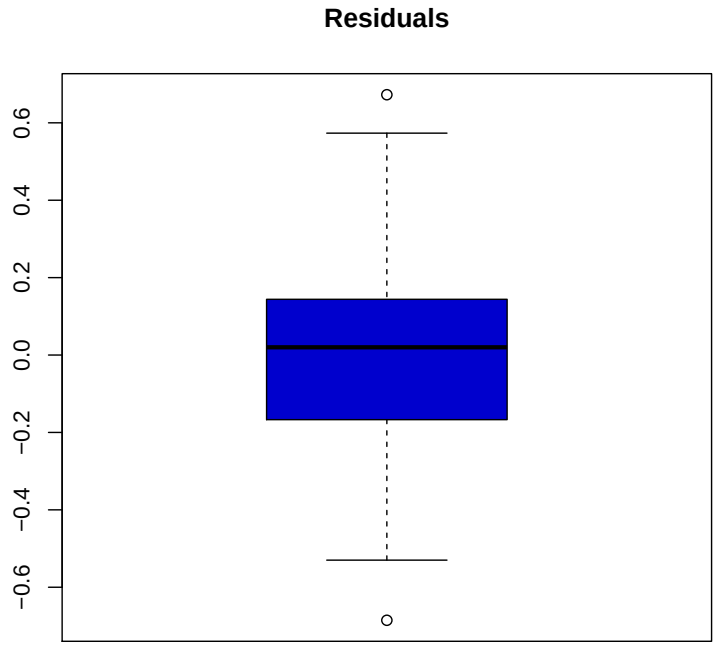


Abbildung 10.5: Boxplot der Residuen zu den Luzernedaten.

Durchführung des Levene-Tests zur Klärung, ob Varianzhomogenität zwischen den Säurebehandlungsgruppen vorliegt:

```
> library(car)
> lev <- data.frame(res = resid.values, group = alfalfa$acid)
> leveneTest(y = lev$res, group = lev$group)
```

```
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  2  1.7928 0.2083
      12
```

✔ Varianzhomogenität wird aufgrund des Levene-Testergebnisses mit einem p-Wert größer als 0.05 angenommen.

- ✔ Die annähernde Normalverteilung des Fehlerterms ist in Abbildung 10.5 zu erkennen.
- ✔ Die Daten sind unabhängig (randomisiertes Blockdesign).

$\Longrightarrow$ ANOVA mit folgenden Hypothesen:

H_0^1 : $\mu_{low} = \mu_{high} = \mu_{control}$

H_1^1 : $\exists$ mindestens ein $\mu_{acid} \neq \mu_{acid'}$

H_{0^2} : $\mu_{block1} = \mu_{block2} = \mu_{block3} = \mu_{block4} = \mu_{block5}$

H_1^2 : $\exists$ mindestens ein $\mu_{block} \neq \mu_{block'}$

```
> anova(object = alfalfa.model)

Analysis of Variance Table

Response: height
          Df  Sum Sq Mean Sq F value  Pr(>F)
acid       2 1.98601 0.99301  5.5066 0.02202 *
block      1 0.30805 0.30805  1.7083 0.21787
Residuals 11 1.98363 0.18033
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Die Ausgabe der Varianzanalyse zeigt, dass zu einem Konfidenzlevel von $\alpha = 5\%$ ein signifikanter Einfluss der Säure auf die Höhe der Luzernepflanzen vorliegt. Wo genau die Signifikanz liegt, kann hier nicht geschlussfolgert werden, da es drei Hypothesenpaare gibt. Dies könnte aber durch einen Multiple Comparison Test zum Mittelwertvergleich (MCP) geklärt werden, da keine Abhängigkeiten vorliegen.

10.2.6 Beispiel Brunnenkresse (1)

10.2.6.1 Versuchsbeschreibung

In einem Studentenexperiment wurde der Einfluss von verschiedenen Lichtqualitäten auf Brunnenkresse (*Lepidium sativum*) untersucht. Es wurden sechs neuartige Lampen nebst der im Gartenbau weitläufig eingesetzten SON-T Lampe verwendet. Aus den drei Blöcken pro Lampe wurden zufällig 15 Pflanzen zur Evaluation ausgewählt. Endpunkt war das Frischgewicht nach acht Tagen (Norlinger and Hoff, 2004), Daten im Anhang B.

10.2.6.2 Auswertung der Daten

Mit Hilfe der ANOVA soll untersucht werden, ob die Lichtqualitäten mindestens einen signifikanten Unterschied im Gewicht der Kressepflanzen bewirken.

```
> cress <- read.table("../text/cress.txt", sep="\t", dec = ",",
+ header = TRUE)
```

Es wird ein lineares Modell für die Höhe in Abhängigkeit von Lichtbehandlung und Block aufgestellt. Abbildung 10.6 zeigt den Residuenplot. Die Residuen sehen normalverteilt aus.

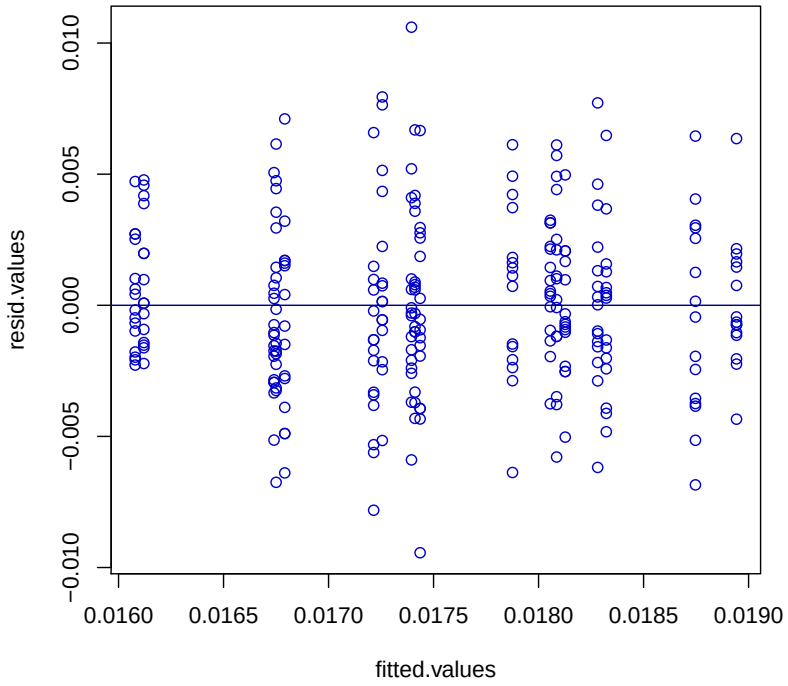


Abbildung 10.6: Residuenplot der Kressedaten zur Feststellung der Normalverteilung des Fehlerterms.

```
> cress.model <- lm(formula = weight~light+block, data = cress)
> fitted.values <- fitted(object = cress.model)
> resid.values <- resid(object = cress.model)
> plot(x = fitted.values, y = resid.values, col = "blue3")
> abline(h = 0, col = "blue4")

> library(car)
> lev <- data.frame(res = resid.values, group = cress$light)
> leveneTest(y = lev$res, group = lev$group)

Levene's Test for Homogeneity of Variance (center = median)
       Df F value Pr(>F)
group   5  1.5068 0.1879
      264
```

Der Levene-Test belegt, dass keine Varianzheterogenität der Residuen vorliegt (Konfidenzniveau 95%).

- ✔ Varianzhomogenität wird aufgrund des Levene-Testergebnisses mit einem p-Wert größer als 0.05 angenommen.
- ✔ Die annähernde Normalverteilung des Fehlerterms ist in Abbildung 10.6 zu erkennen.
- ✔ Die Daten sind unabhängig.

Control	Top	Bottom	Both
86	41	25	13
108	44	35	11
118	40	37	13
79	52	26	13

Tabelle 10.5: Wasserverlust von Cherry Laurel-Blättern ($\frac{mg}{cm^3}$) nach drei Tagen.

$\Longrightarrow$ ANOVA mit folgenden Hypothesen:

H_0^1: $\mu_{red} = \mu_{daylight} = \mu_{SON_T} = \mu_{white} = \mu_{blue} = \mu_{green}$
H_1^1: $\exists$ mindestens ein $\mu_{light_i} \neq \mu_{light_j}$
H_0^2: $\mu_{block1} = \mu_{block2} = \mu_{block3}$
H_1^2: $\exists$ mindestens ein $\mu_{block_i} \neq \mu_{block_j}$

```
> anova(object = cress.model)

Analysis of Variance Table

Response: weight
           Df     Sum Sq    Mean Sq F value  Pr(>F)
light       5 0.00015253 3.0506e-05  2.9121 0.01409 *
block       2 0.00002469 1.2347e-05  1.1787 0.30931
Residuals 262 0.00274459 1.0475e-05
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Zu einem Konfidenzniveau von 95% bewirken die verschiedenen Lichtqualitäten mindestens einen signifikanten Unterschied im Frischgewicht der Kressepflanzen. Es liegt kein signifkanter Blockeinfluss vor.

In einem Multiple Comparison Test kann geklärt werden, wo genau sich der Unterschied befindet (Abschnitt 11.2.5).

✎ Übungsaufgabe 11

In einem Experiment wurde der Einfluss von Petroleumgel - appliziert auf die Fläche von Cherry Laurel-Blättern - auf die Transpiration von Blättern untersucht. Dazu wurden 16 Blätter ausgewählt und zufällig in vier Gruppen aufgeteilt. Die erste Gruppe diente als Kontrollgruppe, während auf die Oberseite der Blätter in der zweiten Gruppe Petroleumgel aufgetragen wurde. Die dritte Gruppe wurde an der Unterseite mit dem Gel behandelt, die vierte Gruppe an Ober- und Unterseite. Die Blätter wurden gewogen, für drei Tage an einem schattigen Ort mit guter Luftzirkulation aufgehangen und wieder gewogen. Die Wasserverluste sind in Tabelle 10.5 wiedergegeben (Bishop, 1980, S. 56).

Werden alle Voraussetzungen zur Varianzanalyse erfüllt? Wenn ja, wie lauten die Hypothesen? Führen Sie gegebenenfalls die ANOVA durch!

Kapitel 11

Multiple Vergleiche

11.1 Voraussetzungen

Mittels der Varianzanalyse lässt sich untersuchen, ob mindestens ein Unterschied zwischen verschiedenen Behandlungen vorliegt. Multiple Vergleiche (engl. Multiple Comparison Tests, MCPs) testen die paarweisen Unterschiede und geben deren Lokalisierung an. (Die ANOVA kann, muss aber nicht, als Vortest für einen MCP verwendet werden. Ergibt die ANOVA eine signifikante Wechselwirkung, ist jedoch die Voraussetzung der Unabhängigkeit nicht mehr erfüllt. In diesem Fall werden die paarweisen Unterschiede des einen Faktors je Stufe des anderen Faktors berechnet!)

Für multiple Vergleiche gelten im Grunde die gleichen Voraussetzungen, wie für den normalen t-Test. Wichtig sind:

- **Normalverteilung** in den einzelnen Gruppen (Boxplots)
- **Varianzhomogenität** zwischen den Behandlungsgruppen (Levene Test, Betrachtung der Boxplots)
- **Unabhängigkeit der Daten** ist beispielsweise erfüllt, wenn in der ANOVA *keine* signifikante Wechselwirkung gefunden wurde, ansonsten wie beim normalen t-Test in Kapitel 5 beschrieben.

11.1.1 Tukey-Prozedur

Beim all pairs Vergleich nach Tukey werden alle Gruppen miteinander verglichen.

11.1.2 Dunnett-Prozedur

Die Variante nach Dunnett ist ein *many to one*-Vergleich, d.h. alle Gruppen werden mit einer einzigen, meist der Kontrollgruppe, verglichen.

11.2 Anwendung

Die Pakete für multiple Vergleichsprozeduren sind zurzeit nicht in der R-Basisinstallation enthalten, daher müssen `mvtnorm` und `multcomp` nachinstalliert und mit der Funktion `library()` eingebunden werden.

11.2.1 Die Funktion `glht()`

Der Name `glht()` ist eine Abkürzung für *general linear hypothesis*. Diese Funktion kann unter anderem für die Durchführung multipler Mittelwertvergleiche in R eingesetzt werden.

```
glht(model, linfct = mcp(grouping.variable = c("Dunnett", "Tukey")),
            alternative = c("two.sided", "less", "greater"))
```

`model` ist ein Modell, z.B. im Fall multipler Mittelwertvergleiche `aov()`.

Mit `lnfct` wird die sogenannte lineare Hypothese angegeben. Diese kann z.B. als Kontrastmatrix eingegeben werden. Wer sich nicht mit Kontrastmatritzen beschäftigen möchte, der kann diese auch durch die Funktion `mcp()` zusammenstellen lassen, wobei `grouping.variable` der Gruppierungsvariable im Modell entsprechen muss. (Es sind über Dunnett oder Tukey hinaus noch weitere, hier nicht diskutierte Varianten verfügbar.)

Das Argument `alternative` ist bereits aus den anderen Tests bekannt und spezifiziert, ob zweiseitig oder einseitig getestet wird, wobei im Fall von Tukey die einseitigen Varianten selten sinnvoll sind.

11.2.2 Die Funktion `confint()`

Die Berechnung der Konfidenzintervalle wird von einer separaten Funktion, `confint()`, übernommen.

```
confint(object, level = 0.95,)
```

Als `object` wird die mit `glht()` erstellte linear hypothesis verwendet.

Mit `level` wird das Konfidenzniveau festgelegt. Der Default sind 95%.

11.2.3 Die Funktion `summary()`

Mit `summary()` lässt sich aus einem `glht()`-Objekt eine detaillierte Ausgabe des Testergebnisses erzeugen.

```
> summary(object)
```

11.2.4 Beispiel Melonen (1)

11.2.4.1 Versuchsbeschreibung

In einem Experiment wurden vier Melonenvarietäten hinsichtlich ihres Ertrags untersucht. Es wurden sechs Blöcke pro Varietät angelegt, das Experiment war vollständig randomisiert. Datensatz in Data 11.1 (Mead et al., 2003, S. 58).

11.2.4.2 Auswertung der Daten

```
> melon <- read.table(file = "../text/melon.txt", sep = "\t", header = TRUE)
```

Zur Überprüfung der Normalverteilung der einzelnen Varietäten werden die Boxplots erstellt (Abbildung 11.1):

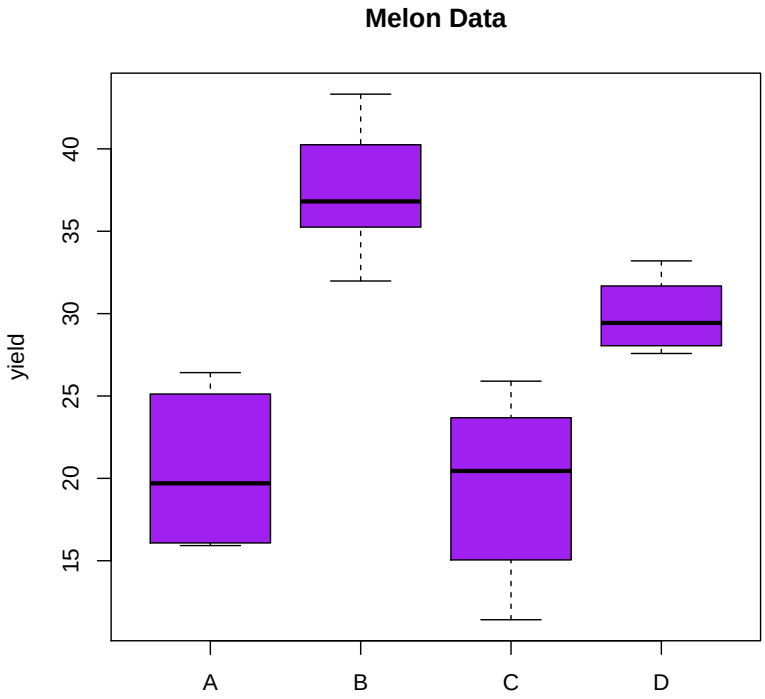


Abbildung 11.1: Boxplots der Melonendaten zur Prüfung der Normalverteilung.

```
> boxplot(formula = yield~variety, data = melon, col = "purple",
+ main = "Melon Data", ylab = "yield")
```

Implementierung des Levene-Tests zur Prüfung auf Varianzhomogenität:

```
> library(car)
> leveneTest(y = melon$yield, group = melon$variety)
```

```
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  3  2.0901 0.1337
      20
```

- ✔ Die annähernde Normalverteilung wird aufgrund der Boxplots angenommen (Abbildung 11.1).
- ✔ Bei einem Konfidenzniveau von 95% wird die H_0-Hypothese des Levene-Tests beibehalten $\rightarrow$ Varianzhomogenität.

$\Longrightarrow$ Die Daten sind für einen MCP geeignet. Die Fragestellung ist, ob sich ein Unterschied zwischen den verschiedenen Varietäten findet, es wird keine Kontrollgruppe genannt. Ein all pairs-Vergleich nach Tukey ist angebracht. Es wird zweiseitig getestet, da nach einem Unterschied gefragt wird und keine Tendenz bekannt ist. Hypothesen:

H_0: $\mu_A = \mu_B = \mu_C = \mu_D$
H_1: $\mu_A \neq \mu_B$
$\mu_A \neq \mu_C$
$\mu_A \neq \mu_D$
$\mu_B \neq \mu_C$
$\mu_B \neq \mu_D$
$\mu_C \neq \mu_D$

variety	yield
A	25.12
A	17.25
A	26.42
A	16.08
A	22.15
A	15.92
B	40.25
B	35.25
B	31.98
B	36.52
B	43.32
B	37.10
C	18.30
C	22.60
C	25.90
C	15.05
C	11.42
C	23.68
D	28.55
D	28.05
D	33.20
D	31.68
D	30.32
D	27.58

Data 11.1: Daten zum Melonenexperiment.

11.2.4.3 Die Implementierung von `glht()`

Mit den Funktionen `glht()` und `summary()` werden die p-Werte berechnet:

```
> library(mvtnorm)
> library(multcomp)
> melon.model <- aov(formula = yield~variety, data = melon)
> mcmp <- glht(melon.model, linfct = mcp(variety = "Tukey"))
> summary(mcmp)

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = yield ~ variety, data = melon)

Linear Hypotheses:
           Estimate Std. Error t value Pr(>|t|)
B - A == 0  16.9133     2.4754   6.833  < 0.001 ***
C - A == 0  -0.9983     2.4754  -0.403  0.97722
D - A == 0   9.4067     2.4754   3.800  0.00559 **
C - B == 0 -17.9117     2.4754  -7.236  < 0.001 ***
D - B == 0  -7.5067     2.4754  -3.033  0.03078 *
D - C == 0  10.4050     2.4754   4.203  0.00214 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)
```

Unter der Testüberschrift wird die Art des Kontrasts angegeben, darauf folgt das analysierte ANOVA-Modell. Die erste Spalte der Ausgabetabelle enthält die entsprechende Paar-Hypothese. Die zweite Spalte enthält einen Schätzer für den Mittelwertunterschied zwischen den beiden Stichproben. Die Spalte `Std. Error` gibt die Standardabweichung zwischen Schätzer und wahren Werten an. Die Spalte `p value` gibt adjustierte p-Werte an.

Interpretiert ewrden die p-Werte wie üblich: Sind sie kleiner als 0.05, so wird die Alternativhypothese mit einer Irrtumswahrscheinlichkeit von 5% angenommen. In diesem Fall unterscheiden sich A und B, A und D, B und C, B und D sowie C und D signifikant voneinander.

11.2.4.4 Die Implementierung von `confint()`

Die Implementierung der Funktion `confint()` erzeugt die Ausgabe der multiplizitätsadustierten Konfidenzintervalle:

```
> melon.int <- confint(mcmp)
> melon.int

         Simultaneous Confidence Intervals

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = yield ~ variety, data = melon)

Quantile = 2.8009
95% family-wise confidence level


Linear Hypotheses:
           Estimate lwr      upr
B - A == 0  16.9133   9.9799  23.8467
C - A == 0  -0.9983  -7.9317   5.9351
D - A == 0   9.4067   2.4733  16.3401
C - B == 0 -17.9117 -24.8451 -10.9783
D - B == 0  -7.5067 -14.4401  -0.5733
D - C == 0  10.4050   3.4716  17.3384
```

11.2.4.5 Interpretation

Auf die Überschrift, welche den aufgerufenen Test wiedergibt, folgt eine Tabelle. In der dritten (`lwr`) und vierten (`upr`) Spalte finden sich die untere und obere Grenze des jeweiligen Konfidenzintervalls.

11.2.4.6 Graphische Darstellung der Konfidenzintervalle

Die graphische Darstellung der Konfidenzintervalle ist sehr einfach durch `plot(confint())` möglich (Abbildung 11.2):

```
> plot(x = melon.int, col = "purple")
```

11.2.4.7 Schlussfolgerung

Die ursprüngliche Fragestellung war, ob und wo sich signifikante Mittelwertunterschiede mit einer Irrtumswahrscheinlichkeit von 5% finden lassen. Aus den simultanen Konfidenzintervallen lässt sich ablesen, dass sich die Varietät A hinsichtlich des Ernteertrags von den Varietäten B und D unterscheidet. Varietät B unterscheidet sich außerdem von den Varietäten C und D. Die Varietäten C und D unterscheiden sich auch voneinander. Dieses Resultat deckt sich mit den nach Bonferroni adjustierten p-Werten.

Fragt man nun, welche Varietät den höchsten Ertrag aufweist, eine einseitige Fragestellung, so lässt sich das auch aus den Konfidenzintervallen ablesen, ohne einen weiteren Test durchzuführen. Die positiven Konfidenzintervalle können als *ist größer als* gewertet werden, die negativen Konfidenzintervalle als *ist kleiner als*. Dadurch gelangt man zu folgenden Feststellungen:

B $>$ A, D $>$ A, D $>$ C, C $<$ B und D $<$ B.

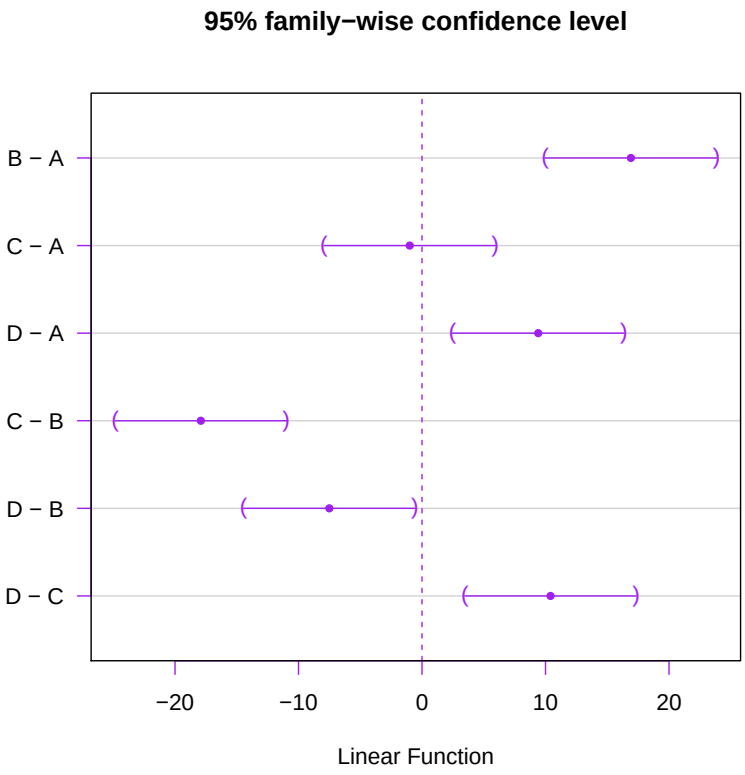


Abbildung 11.2: Graphische Darstellung der Konfidenzintervalle/MCP nach Tukey zum Melonendatensatz.

Die Signifikanz für diese Hypothesen zu einem Konfidenzniveau von 95% wird eingehalten, da die p-Werte für einen einseitigen Test halbiert werden können und damit auf jeden Fall kleiner als 0.05 sind.

Aus dieser Herleitung ist zu schließen, dass die Varietät B den höchsten Ertrag aufweist.

Die Durchführung eines neuen einseitigen Tests mit Konfidenzintervallen ect. ist natürlich auch möglich.

11.2.5 Beispiel Brunnenkresse (2)

In Abschnitt 10.2.6 wurde mit Hilfe der Varianzanalyse festgestellt, dass unterschiedliche Lichtqualitäten mindestens einen Unterschied im Frischgewicht von Kressepflanzen bewirken. Mit einem multiplen Vergleich nach Tukey soll festgestellt werden, wo genau Unterschiede lokalisiert sind.

Zur Überprüfung von Normalverteilung der Daten werden die Boxplots (Abbildung 11.3) erstellt und ergänzend zur Prüfung auf Varianzhomogenität ein Levene-Test durchgeführt:

```
> boxplot(formula = weight~light, data = cress, col = "purple",
+ main = "Cress Data", ylab = "fresh weight (mg)")
> library(car)
> leveneTest(y = cress$weight, group = cress$light)

Levene's Test for Homogeneity of Variance (center = median)
       Df F value  Pr(>F)
```

```
group   5  1.8985 0.09489 .
      264
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

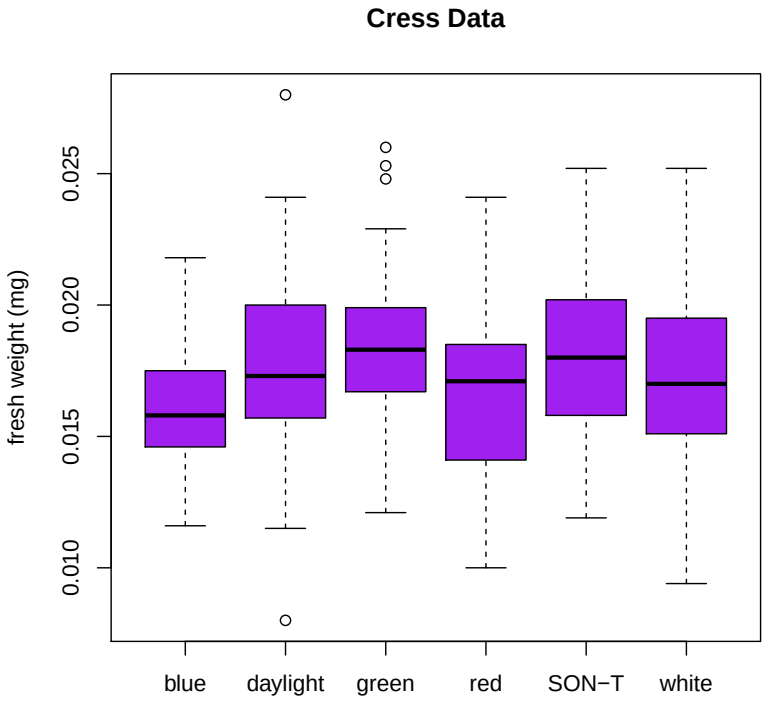


Abbildung 11.3: Boxplots der Kressedaten zur Prüfung der Normalverteilung.

- ✔ Aufgrund der Ausreißer in Abbildung 11.3 wird die annähernde Normalverteilung nur unter Vorbehalt angenommen (Abbildung 11.3).
- ✔ Bei einem Konfidenzniveau von 95% wird die H_0-Hypothese des Levene-Tests beibehalten → Varianzhomogenität.
- ✔ Die Daten sind unabhängig, da das Experiment vollständig randomisiert wurde.

Ein zweiseitiger multipler Vergleich nach Tukey mit folgenden Hypothesen wird ausgeführt:

H_0: $\mu_{red} = \mu_{blue} = \mu_{green} = \mu_{white} = \mu_{daylight} = \mu_{SON-T}$
H_1: $\mu_{red} \neq \mu_{blue}$
$\mu_{red} \neq \mu_{green}$
$\mu_{red} \neq \mu_{white}$
$\mu_{red} \neq \mu_{daylight}$
$\mu_{red} \neq \mu_{SON-T}$
$\mu_{blue} \neq \mu_{green}$
$\mu_{blue} \neq \mu_{white}$
$\mu_{blue} \neq \mu_{daylight}$
$\mu_{blue} \neq \mu_{SON-T}$
$\mu_{green} \neq \mu_{white}$
$\mu_{green} \neq \mu_{daylight}$
$\mu_{green} \neq \mu_{SON-T}$
$\mu_{white} \neq \mu_{daylight}$
$\mu_{white} \neq \mu_{SON-T}$
$\mu_{daylight} \neq \mu_{SON-T}$

```
> library(mvtnorm)
> library(multcomp)
> cress.model <- aov(formula = weight~light, data = cress)
> cress.test <- glht(cress.model, linfct = mcp(light = "Tukey"),
+ alternative = "two.sided")
> summary(cress.test)

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = weight ~ light, data = cress)

Linear Hypotheses:
                       Estimate Std. Error t value Pr(>|t|)
daylight - blue == 0   0.0013156  0.0006828   1.927   0.3882
green - blue == 0      0.0022022  0.0006828   3.225   0.0175 *
red - blue == 0        0.0006711  0.0006828   0.983   0.9231
SON-T - blue == 0      0.0020067  0.0006828   2.939   0.0414 *
white - blue == 0      0.0011356  0.0006828   1.663   0.5576
green - daylight == 0  0.0008867  0.0006828   1.299   0.7857
red - daylight == 0   -0.0006444  0.0006828  -0.944   0.9347
SON-T - daylight == 0  0.0006911  0.0006828   1.012   0.9136
white - daylight == 0 -0.0001800  0.0006828  -0.264   0.9998
red - green == 0      -0.0015311  0.0006828  -2.242   0.2221
SON-T - green == 0    -0.0001956  0.0006828  -0.286   0.9997
white - green == 0    -0.0010667  0.0006828  -1.562   0.6241
SON-T - red == 0       0.0013356  0.0006828   1.956   0.3706
white - red == 0       0.0004644  0.0006828   0.680   0.9840
white - SON-T == 0    -0.0008711  0.0006828  -1.276   0.7980
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)
```

Vom adjustierten p-Wert ausgehend liegen zu einem Konfidenzniveau von 95% zwei signifikante Unterschiede im Frischgewicht vor:

- a) Zwischen grünem und blauem Licht,

- b) zwischen SON-T und blauem Licht.

11.2.5.1 Weitere Untersuchung

Studien an verschiedenen Pflanzenarten haben ergeben, dass blaues Licht eine verringerte Stängelelongation bewirkt, als beispielsweise rotes Licht. Blaues Licht führt deshalb häufig zu kompakterem Pflanzenwachstum und einem leicht verringerten Frischgewicht. Trifft dies auch auf Brunnenkresse zu? Einseitiger Test auf Abfall nach Dunnett (Kontrollgruppe `blue`):

```
> library(mvtnorm)
> library(multcomp)
> cress.test <- glht(cress.model, linfct = mcp(light = "Dunnett"),
+ alternative = "greater")
> summary(cress.test)

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Dunnett Contrasts


Fit: aov(formula = weight ~ light, data = cress)

Linear Hypotheses:
                       Estimate Std. Error t value  Pr(>t)
daylight - blue <= 0 0.0013156  0.0006828   1.927 0.09946 .
green - blue <= 0    0.0022022  0.0006828   3.225 0.00328 **
red - blue <= 0      0.0006711  0.0006828   0.983 0.42232
SON-T - blue <= 0    0.0020067  0.0006828   2.939 0.00790 **
white - blue <= 0    0.0011356  0.0006828   1.663 0.16232
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)
```

Der Output zeigt, dass mit einer Irrtumswahrscheinlichkeit von 5% die Pflanzen unter sowohl grüner als auch SON-T Lichtbehandlung ein höheres Frischgewicht aufweisen, als die mit blauem Licht behandelten Kressepflanzen. Dieses Ergebnis deckt sich nur zum Teil mit den bisher bekannten Forschungsergebnissen zum Einfluss der Lichtqualität auf Stängelelongation von Pflanzen.

11.2.6 Beispiel Dünger

11.2.6.1 Versuchsbeschreibung

Zwölf Versuchsplots wurden zufällig in drei Gruppen geteilt. Die ersten beiden Gruppen wurden mit den Düngern A bzw. B behandelt, die dritte Gruppe blieb als Kontrolle unbehandelt (Tabelle 11.1) (Wonnacott and Wonnacott, 1990, S. 334)

11.2.6.2 Auswertung der Daten

```
> fertilizer <- read.table(file = "../text/fertilizer.txt", sep = "\t",
+ header = TRUE)
```

Fertilizer A	Fertilizer B	Control C
75	74	60
70	78	64
66	72	65
69	68	55

Tabelle 11.1: Ertrag verschiedener Düngerbehandlungen.

Zur Feststellung von Normalverteilung und Varianzhomogenität werden Boxplots (Abb. 11.4 erstellt und ein Levene-Test durchgeführt:

```
> boxplot(formula = yield~fertilizer, data = fertilizer, col = "purple",
+ main = "Fertilizer Data")
> library(car)
> leveneTest(y = fertilizer$yield, group = fertilizer$fertilizer)

Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  2  0.1765 0.8411
       9
```

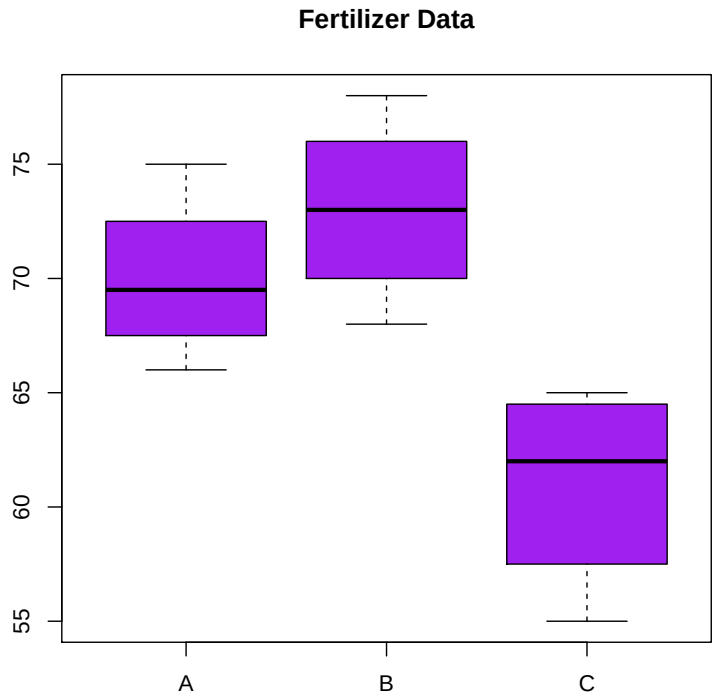


Abbildung 11.4: Boxplots der Düngerdaten zur Feststellung über die Normalverteilung der Daten.

- ✔ Die Daten sind annähernd gaußverteilt (Abbildung 11.4).
- ✔ Mit dem nicht signifikanten Levene-Test und anhand der Boxplots lässt sich feststellen, dass die Varianzen homogen sind.

✔ Die Daten sind unabhängig voneinander: randomisiertes Blockdesign.

Die Fragestellung in diesem Experiment ist, ob der Einfluss der beiden Dünger sich signifikant von der Kontrolle unterscheiden. Daher wird ein einseitiger Test auf Anstieg (denn von Dünger ist eine Steigerung des Erntegewichtes zu erwarten) mit folgenden Hypothesen durchgeführt (im ersten Schritt werden die Levels des Gruppierungsvektors so angepasst, dass die `alternative` im Kontext mit den Hypothesen Sinn ergibt):

$$H_0: \quad \mu_C \geq \mu_A \quad \mu_C \geq \mu_B$$

$$H_1: \quad \mu_C < \mu_A \quad \mu_C < \mu_B$$

```
> fertilizer$fertilizer <- ordered(fertilizer$fertilizer,
+ levels = c("C", "A", "B"))
> fert.model <- aov(formula = yield~fertilizer, data = fertilizer)
> fert.test <- glht(fert.model, linfct = mcp(fertilizer = "Dunnett"),
+ alternative = "greater")
> summary(fert.test)

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Dunnett Contrasts


Fit: aov(formula = yield ~ fertilizer, data = fertilizer)

Linear Hypotheses:
           Estimate Std. Error t value  Pr(>t)
A - C <= 0    9.000      2.944   3.057 0.01233 *
B - C <= 0   12.000      2.944   4.076 0.00256 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)
```

Aufgrund der zu einem Konfidenzniveau von 95% signifikanten p-Werte kann ausgesagt werden, dass beide Dünger zu einer deutlichen Ertragssteigerung führen.

11.2.7 Beispiel Melonen (2)

Zur Anerkennung von neuen Sorten muss nachgewiesen werden, ob die neue Sorte in mindestens einem Punkt besser ist, als die schon vorhandenen Sorten.

Auf das Beispiel mit den Melonen aus Abschnitt 11.2.4 zurückgreifend, treffe ich die Annahme, dass Varietät A die neue Sorte ist, die mit allen anderen vorhanden Sorten B, C und D auf Verbesserung verglichen werden soll. In diesem Fall ist ein MCP nach `Dunnett` angebracht. Die Implementierung erfolgt exakt wie in Abschnitt 11.2.4 beschrieben, außer dass das Argument `type` auf Dunnett gesetzt wird. Es handelt sich um ein einseitiges Testproblem (Anstieg).

```
> melon.model <- aov(formula = yield~variety, data = melon)
> mcmp <- glht(melon.model, linfct = mcp(variety = "Dunnett"),
+ alternative = "greater")
> summary(mcmp)
```

```
         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Dunnett Contrasts


Fit: aov(formula = yield ~ variety, data = melon)

Linear Hypotheses:
           Estimate Std. Error t value  Pr(>t)
B - A <= 0  16.9133     2.4754   6.833 < 0.001 ***
C - A <= 0  -0.9983     2.4754  -0.403 0.87050
D - A <= 0   9.4067     2.4754   3.800 0.00143 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)
```

```
> plot(confint(mcmp), col = "purple")
```

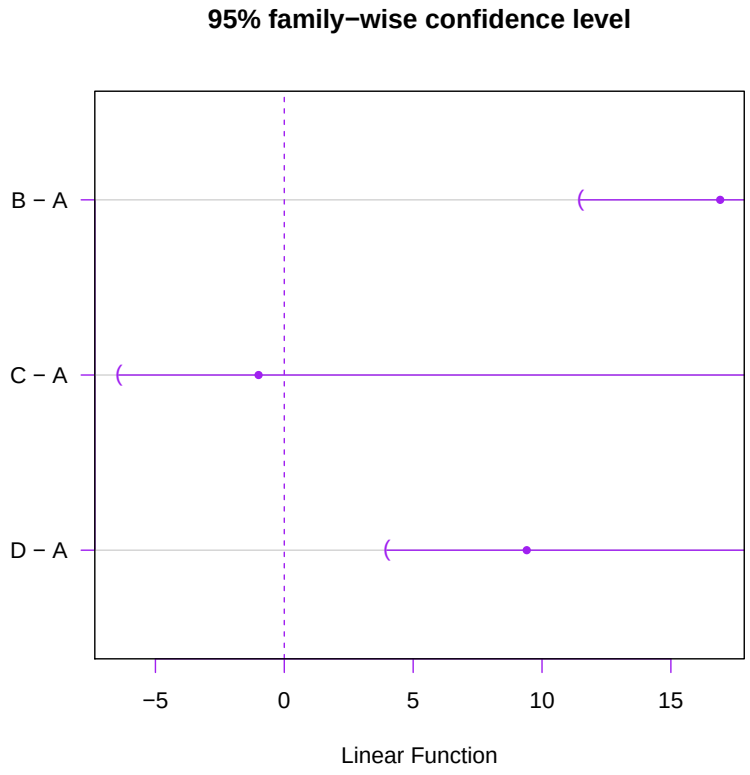


Abbildung 11.5: Einseitige Konfidenzintervalle für den Dunnett MCP angewandt auf den Melonendatensatz.

Den Konfidenzintervallen lässt sich entnehmen, dass die Varietät A zu einer Irrtumswahrscheinlichkeit von 5% hinsichtlich des Ertrags signifikant besser ist, als die Varietäten B und D. Für die Zulassung als neue Sorte wäre die fehlende Signifkanz gegenüber C eventuell ein Problem.

11.2.8 Elementare Berechnung der p-Werte nach Holm

Zur Adjustierung der lokalen p-Werte (`p raw`) wird wie folgt vorgegangen:

Bonferroni: Die rohen p-Werte werden mit der Zahl der Vergleiche multipliziert.

Bonferroni-Holm: Die p-Werte werden der Größe nach aufsteigend sortiert. Der erste p-Wert wird mit der Gesamtanzahl z der Vergleiche multipliziert. Ist der adjustierte p-Wert signifikant, so wird der nächste p-Wert mit z-1 multipliziert usw. Das Verfahren bricht ab, wenn ein nicht signifikanter p-Wert auftaucht (Holm, 1979).

Diese Berechnungen sind in Tabelle 11.2 am Beispiel des Melonentestproblems veranschaulicht.

Hypoth.	$\mathbf{p}_{raw}$	$\mathbf{p}_{Bonf}$	$\mathbf{p}_{Bonf-Holm}$	**Sign.**
B - A	0.000	0.000*3 = 0	0.000*3 = 0	ja/ja
D - A	0.001	0.001*3 = 0.003	0.001*2 = 0.002	ja/ja
C - A	0.654	0.654*3 $\Rightarrow$ 1	0.654*1 = 0.654, stop	nein/nein

Tabelle 11.2: Adjustierung der p-Werte nach Bonferroni und Bonferroni-Holm.

11.2.8.1 Implementierung in R

Die Adjustierungen von p-Werten sind nach diversen Autoren in der Funktion `p.adjust()` implementiert:

```
p.adjust(raw.p.vector, method = c("holm", "hochberg", "hommel",
         "bonferroni", "BH", "BY", "fdr", "none")
```

```
> raw.p.values <- c(0, 0.001, 0.654)
> bonf.p.values <- p.adjust(raw.p.values, method = "bonferroni")
> bonf.p.values

[1] 0.000 0.003 1.000

> holm.p.values <- p.adjust(raw.p.values, method = "holm")
> holm.p.values

[1] 0.000 0.002 0.654
```

✎ Übungsaufgabe 12

Der Feuchtigkeitsgehalt vier verschiedener Bodentypen wurde durch Messung von jeweils zehn Proben untersucht (Daten in der Tabelle 11.3) (Mead et al., 2003, S. 62).

Sind die Daten für einen MCP geeignet? Wenn ja, welche Variante bietet sich hier an? Stellen Sie die Hypothesenpaare auf! Implementieren Sie den Test und das Zeichnen der Konfidenzintervalle; Interpretieren Sie den Output!

Soil A	Soil B	Soil C	Soil D
12.8	8.1	9.8	16.4
13.4	10.3	10.6	8.2
11.2	4.2	9.1	15.1
11.6	7.8	4.3	10.4
9.4	5.6	11.2	7.8
10.3	8.1	11.6	9.2
14.1	12.7	8.3	12.6
11.9	6.8	8.9	11.0
10.5	6.9	9.2	8.0
10.4	6.4	6.4	9.8

Tabelle 11.3: Der Feuchtigkeitsgehalt vier verschiedener Bodentypen.

Schlusswort

Mit Hilfe dieses Handbuches sollen Studenten der Biometrie-Einführungsveranstaltungen biowissenschaftlicher Fakultäten in der Lage sein, die Benutzungsweise von R nachzuvollziehen und die Software selbstständig zur Auswertung wissenschaftlicher Experimente zu verwenden.

Aus hunderten Funktionen in R wurden einige sehr nützliche ausgewählt und erläutert. Reale Datensätze wahren einen engen Bezug zur gartenbauwissenschaftlichen Praxis. Behandelt wurden parametrische und nicht parametrische Zweistichprobentests, Korrelation, lineare Regression und ANOVA.

Für deutschsprachige Studenten erleichtert dieses Einführungshandbuch das Lesen und Verstehen der englischsprachigen Hilfe.

Anhang A

Lösungen der Übungsaufgaben

✍ Lösung 1

1. ```
   > a <- 12
   > b <- 7
   > result.2.binom <- (a-b)^2
   > result.2.binom

   [1] 25
   ```

2. ```
   > zahlenkette <- (28:-34)
   > zahlenkette

    [1]  28  27  26  25  24  23  22  21  20  19  18  17  16  15  14
   [16]  13  12  11  10   9   8   7   6   5   4   3   2   1   0  -1
   [31]  -2  -3  -4  -5  -6  -7  -8  -9 -10 -11 -12 -13 -14 -15 -16
   [46] -17 -18 -19 -20 -21 -22 -23 -24 -25 -26 -27 -28 -29 -30 -31
   [61] -32 -33 -34
   ```

3. ```
   > ?objects()
   ```

   Unter Windows das Hilfefenster wie üblich schließen, unter Linux zur Rückkehr in die R-Konsole „q“drücken.

   ```
   > objects()

   [1] "a"              "b"              "result.2.binom"
   [4] "zahlenkette"

   > rm(object = a)
   ```

Data-Frame zu Sonnenblumen in Flüssigkultur:

```
> sunflowers <- data.frame(solution = rep(c("complete","1.Mg","1.N","1.mn"),
+ each = 3), dry.weight = c(1172,750,784,67,95,59,148,234,92,297,243,263))
> sunflowers

   solution dry.weight
1  complete       1172
2  complete        750
3  complete        784
```
```

```
4      l.Mg         67
5      l.Mg         95
6      l.Mg         59
7       l.N        148
8       l.N        234
9       l.N         92
10     l.mn        297
11     l.mn        243
12     l.mn        263
```

✍ Lösung 2

```
> salad <- data.frame(weight = c(3.06, 2.78, 2.87, 3.52, 3.81, 3.60, 3.3, 2.77,
+ 3.62, 1.31, 1.17, 1.72, 1.20, 1.55, 1.53), group = c(rep(c("bowl"), times = 9),
+ rep(c("bibb"), times = 6)))
> tapply(X = salad$weight, INDEX = salad$group, FUN = mean)

    bibb     bowl
1.413333 3.258889

> tapply(X = salad$weight, INDEX = salad$group, FUN = sd)

     bibb      bowl
0.2198788 0.3999201

> tapply(X = salad$weight, INDEX = salad$group, FUN = median)

bibb bowl
1.42 3.30

> tapply(X = salad$weight, INDEX = salad$group, FUN = var)

      bibb       bowl
0.04834667 0.15993611

> tapply(X = salad$weight, INDEX = salad$group, FUN = min)

bibb bowl
1.17 2.77

> tapply(X = salad$weight, INDEX = salad$group, FUN = max)

bibb bowl
1.72 3.81

> tapply(X = salad$weight, INDEX = salad$group, FUN = quantile)

$bibb
    0%    25%    50%    75%   100%
1.1700 1.2275 1.4200 1.5450 1.7200

$bowl
  0%  25%  50%  75% 100%
2.77 2.87 3.30 3.60 3.81
```

```
> tapply(X = salad$weight, INDEX = salad$group, FUN = sum)

 bibb  bowl
 8.48 29.33

> tapply(X = salad$weight, INDEX = salad$group, FUN = IQR)

  bibb   bowl
0.3175 0.7300
```

✍ Lösung 3

Die Eingabe von folgendem Quelltext ergibt Abbildung A.1:

```
> boxplot(formula = weight~group, data = salad, col = "red1",
+ main = "Dryweight of Lettuce Varieties", ylab = "dryweight (g)")
```

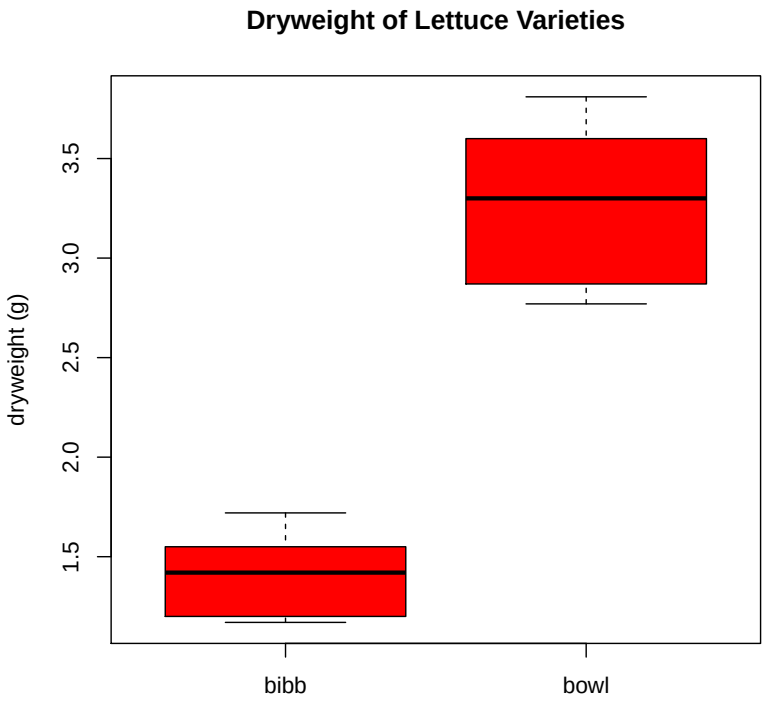


Abbildung A.1: Boxplots zum Trockengewicht der Blätter zweier Salatvarietäten.

✍ Lösung 4

Die Auswirkung des Additivs auf die Pflanzen ist unbekannt, daher ein zweiseitiges Hypothesenpaar:

$$H_0 : \mu_{standard} = \mu_{additiv}$$
$$H_1 : \mu_{standard} \neq \mu_{additiv}$$

```
> strawberry<- read.table(file = "../text/strawberry.txt", sep = "\t",
+ header = TRUE)
```

Zur Feststellung der Verteilung werden die Boxplots erstellt (Abb. A.2:

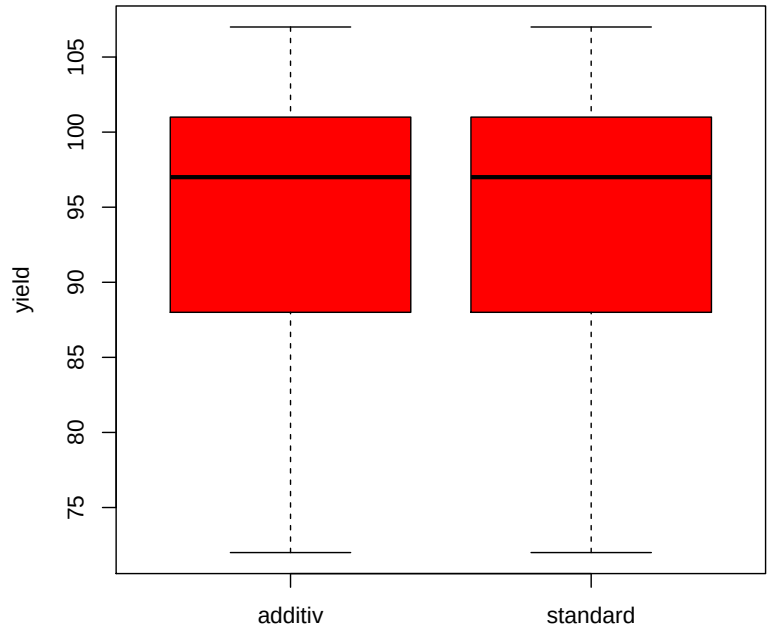


Abbildung A.2: Boxplots zum Ertrag von Erdbeeren mit und ohne Zusatz eines Additivs zum Desinfektionsmittel gegen kleine weiße Würmer.

```
> boxplot(formula = yield ~ treatment, data = strawberry, col = "red",
+ ylab = "yield")
```

Es ist von einer Normalverteilung auszugehen.

Zur Feststellung der Varianzhomogenität wird ein F-Test durchgeführt:

```
> var.test(formula = yield ~ treatment, data = strawberry)

        F test to compare two variances

data:  yield by treatment
F = 1, num df = 4, denom df = 4, p-value = 1
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
```

```
 0.1041175 9.6045299
sample estimates:
ratio of variances
                 1
```

Die Varianzen unterscheiden sich nicht signifikant voneinander.

Die Daten sind für eine Auswertung mit dem t-Test geeignet.

```
> t.test(formula = yield ~ treatment, data = strawberry,
+ alternative = "two.sided", var.equal = TRUE)

        Two Sample t-test

data:  yield by treatment
t = 0, df = 8, p-value = 1
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -19.86379  19.86379
sample estimates:
 mean in group additiv mean in group standard
                    93                     93
```

Es lässt sich zu einem Konfidenzniveau von 95% kein signifikanter Unterschied feststellen. Im Gegenteil: Der sehr große p-Wert indiziert, dass die Stichproben sich recht ähnlich sind. Dies kann für den Versuch als Erfolg gewertet werden, da ein Einfluss des Additivs auf die Erdbeerpflanzen vermieden werden sollte.

✍ Lösung 5

Es wird zweiseitig getestet, da a priori keine Tendenz erkennbar ist:

$$H_0 : \mu_{bowl} \neg \mu_{bibb}$$
$$H_1 : \mu_{bowl} \neq \mu_{bibb}$$

```
> read.table(file = "../text/lettuce.txt", sep = "\t",
+ header = TRUE)

   variety weight
1     bowl   3.06
2     bowl   2.78
3     bowl   2.87
4     bowl   3.52
5     bowl   3.81
6     bowl   3.60
7     bowl   3.30
8     bowl   2.77
9     bowl   3.62
10    bibb   1.31
11    bibb   1.17
12    bibb   1.72
13    bibb   1.20
14    bibb   1.55
15    bibb   1.53
```

```
> boxplot(formula = weight~variety, data = lettuce, col = "orange")
```

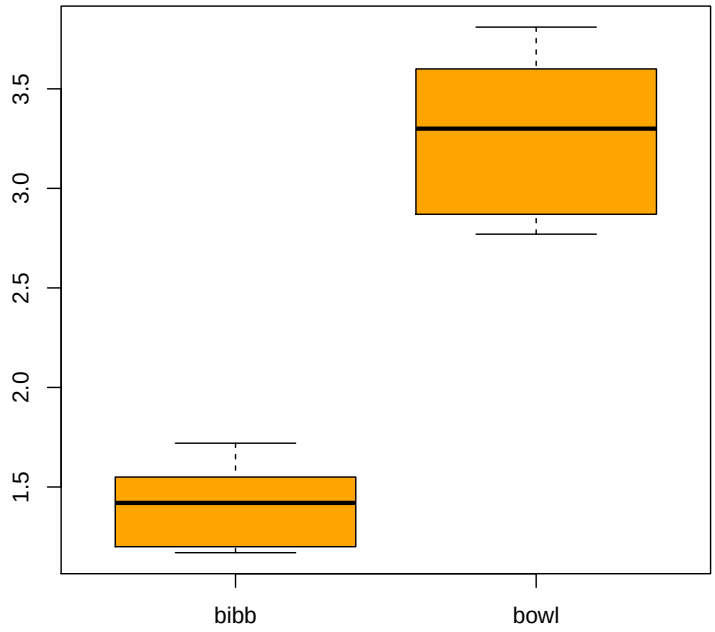


Abbildung A.3: Boxplots zum Trockengewicht von Blättern zweier Salatvarietäten.

- ✔ Die Daten sind annähernd normal verteilt (Abbildung A.3).
- ✔ Die Boxen sind unterschiedlich lang (Boxplots A.3), deshalb wird von Varianzheterogenität ausgegangen.
- ✔ Die Daten sind unabhängig voneinander.

$\Longrightarrow$ t-Welch Test.

```
> t.test(formula = weight~variety, data = lettuce)

        Welch Two Sample t-test

data:  weight by variety
t = -11.484, df = 12.716, p-value = 4.422e-08
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -2.193542 -1.497569
sample estimates:
mean in group bibb mean in group bowl
          1.413333           3.258889
```

Die beiden Salatvarietäten unterschieden sich mit einer Irrtumswahrscheinlichkeit von 5% hinsichtlich ihres Trockengewichtes signifikant voneinander, da der p-Wert kleiner als 0.05 ist. Aus dem Konfidenzintervall lässt sich ablesen, dass die Blätter der Varietät Bibb in 95% der Fälle mindestens 1.4 bis 2.1 g leichter sind, als die der Varietät Bowl.

✍ Lösung 6

```
> light <- read.table(file = "../text/light.txt", sep = "\t",
+ header = TRUE)
> boxplot(formula = height~color, data = light, col = "yellow",
+ ylab = "height in inches", main ="Effect of Light on Soybeans")
> var.test(formula = height~color, data = light)

        F test to compare two variances

data:  height by color
F = 1.4026, num df = 24, denom df = 16, p-value = 0.4892
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.5342844 3.3809995
sample estimates:
ratio of variances
          1.402585
```

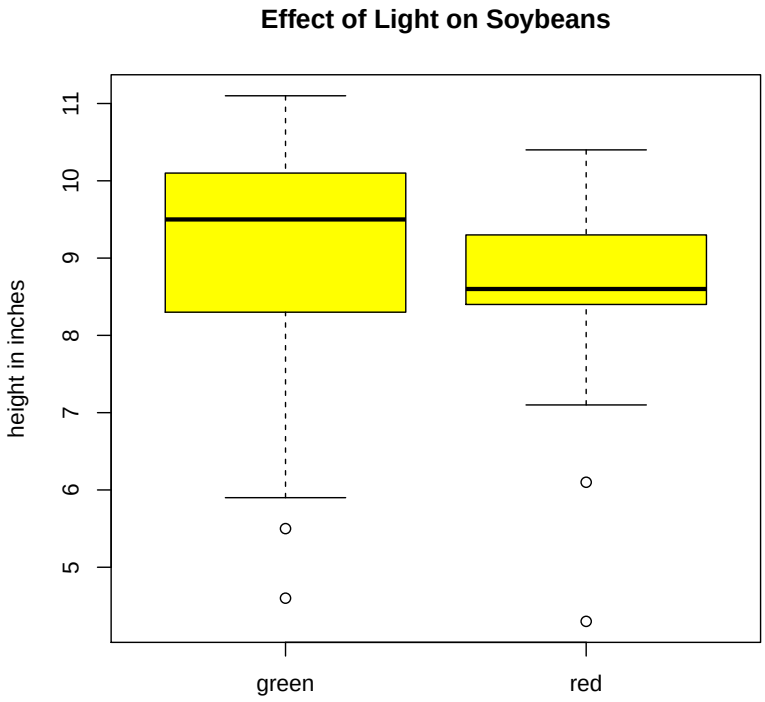


Abbildung A.4: Boxplots der Auswirkung von zwei Lichtfarben auf das Wachstum von Sojabohnenpflanzen.

- ✔ Unbekannte Verteilung, keine Gaußverteilung (siehe Ausreißer und asymmetrisch liegenden Median in den Boxplots A.4).
- ✔ Varianzhomogenität ist kritisch zu betrachten. Mit dem F-Test wurde keine signifikante Varianzheterogenität nachgewiesen.

✔ Stetige Daten (Höhe gemessen in Inches).

$\Longrightarrow$ Zweiseitiger Wilcoxon-Rangsummentest, da a priori keine Tendenz klar ist, Verwendung von `wilcox.exact()`, weil Bindungen vorliegen.

$$H_0 : F_{red}(y) = F_{green}(y)$$
$$H_1 : F_{red}(y) \neq F_{green}(y)$$

```
> library(exactRankTests)
> wilcox.exact(formula = height~color, data = light,
+ alternative = "two.sided", correct = FALSE, exact = TRUE)

        Exact Wilcoxon rank sum test

data:  height by color
W = 272, p-value = 0.1296
alternative hypothesis: true mu is not equal to 0
```

Mit rotem Licht behandelte Sojabohnenpflanzen unterscheiden sich in ihrer Höhe mit einer Irrtumswahrscheinlichkeit von 10% nicht signifikant von mit grünem Licht behandelten Sojabohnenpflanzen.

✍ Lösung 7

χ^2-Anpassungstest nach Pearson (Fallzahl größer 20).

```
> flax <- c(15,26,15,0,8,8)
> genetic.model <- c(3,6,3,1,2,1)/16
> chisq.test(x = flax, p = genetic.model)

        Chi-squared test for given probabilities

data:  flax
X-squared = 7.7037, df = 5, p-value = 0.1733
```

Die beobachtete Verteilung unterscheidet sich zu einem Konfidenzniveau von 90% nicht signifikant von der hypothetischen Verteilung. H_0 wird nicht abgelehnt.

✍ Lösung 8

χ^2-Homogenitätstest nach Pearson (Fallzahl größer 20).

```
> biotope <- matrix(c(25,25,75,75), ncol = 2)
> chisq.test(biotope, correct = FALSE)

        Pearson's Chi-squared test

data:  biotope
X-squared = 0, df = 1, p-value = 1
```

Mit 1 ist der p-Wert deutlich größer als 0.1. H_0 wird beibehalten. Es findet sich kein signifikanter Unterschied in der prozentualen Verteilung von Spezies A in Abhängigkeit von Spezies B. Es ist also nicht von einer Assoziation auszugehen.

✍ Lösung 9

```
> ascorbic.acid <- read.table(file = "../text/ascorbic.txt", sep = "\t",
+ header = TRUE)
> plot(acid~response, data = ascorbic.acid, col = "green3",
+ xlab = "Ascorbic acid concentration (mug/cm^3)", ylab = "response",
+ main = "Photometric Data")
> boxplot(x = ascorbic.acid$acid, col = "green3",
+ ylab = "conenctration (mug/cm^3)", main = "Boxplot of Acid Concentration")
> boxplot(ascorbic.acid$response, col = "green3",
+ main = "Photometer Response")
```

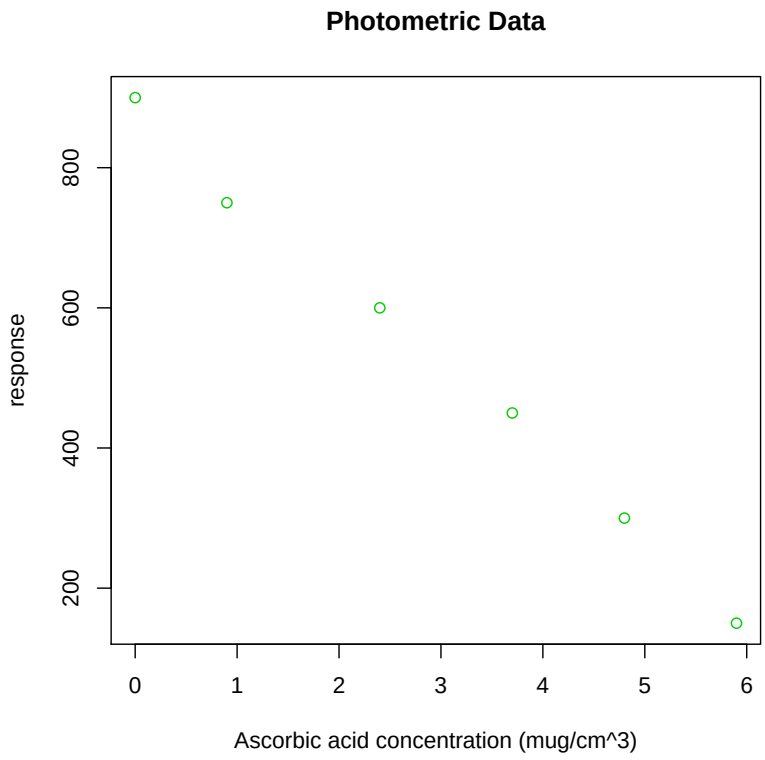


Abbildung A.5: Scatterplot der photometrischen Daten zum Ascorbinsäuregehalt einer Lösung.

Im Scatterplot (Abbildung A.5) lässt sich ein negativer, signifikanter Korrelationskoeffizient vermuten. Aus Abbildungen A.6 und A.7 lässt sich die Gaußverteilung der Variablen ablesen. Deshalb wird eine Korrelation nach Pearson mit einseitigem Test auf Abfall durchgeführt:

```
> cor.test(formula = ~acid+response, data = ascorbic.acid,
+ method = "pearson", alternative = "less")

        Pearson's product-moment correlation

data:  acid and response
t = -33.599, df = 4, p-value = 2.34e-06
```

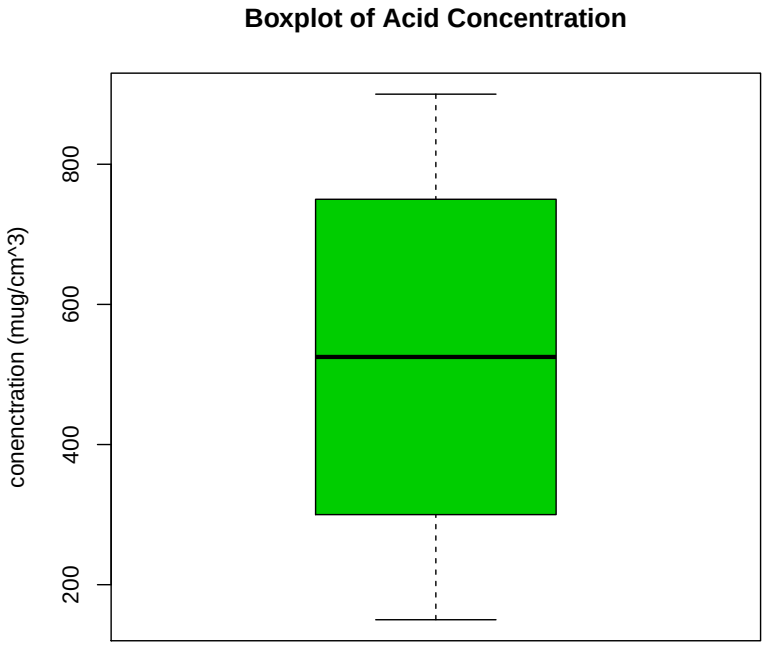


Abbildung A.6: Boxplot der Ascorbinsäurekonzentration zur Feststellung der Gaußverteilung.

```
alternative hypothesis: true correlation is less than 0
95 percent confidence interval:
 -1.0000000 -0.9882535
sample estimates:
       cor
-0.9982331
```

Mit einem Korrelationskoeffizienten von -0.998233 liegt eine nahezu perfekte Korrelation vor. Der kleine p-Wert zeigt, dass die Korrelation zu einem Konfidenzniveau von 95% hoch signifikant ist.

✍ Lösung 10

```
> sulphur <- read.table(file = "../text/sulphur.txt", sep = "\t",
+ header = TRUE)
> plot(scab~concentration, data = sulphur, col = "turquoise3",
+ xlab = "sulphur (pounds/acre)", ylab = "percentage scab damage",
+ main = "Scab Treatment with Sulphur")
> scabmodel <- lm(formula = scab~concentration, data = sulphur)
> abline(reg = scabmodel, col = "turquoise4")
```

Mit folgenden Funktionen lassen sich die Residuen graphisch darstellen (siehe Abbildung A.9). Die Residuen werden als normalverteilt und annähernd varianzhomogen angenommen.

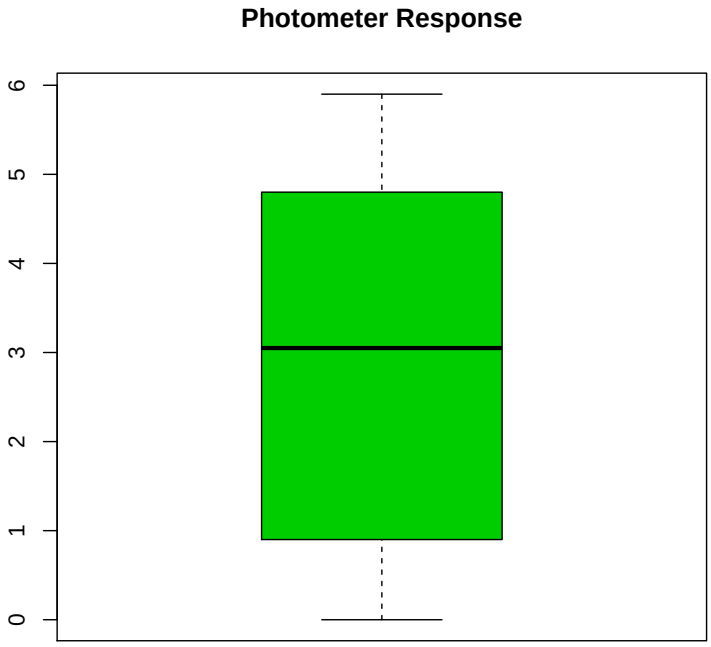


Abbildung A.7: Boxplot der photometrischen Antwortdaten zur Feststellung der Gaußverteilung.

```
> fitted.values <- fitted(object = scabmodel)
> resid.values <- resid(object = scabmodel)
> plot(x = fitted.values, y = resid.values, col = "turquoise3")
> abline(h = 0, col = "turquoise4")
```

- ✔ Die **Anzahl der Prediktorstufen** vier ist größer zwei.
- ✔ Die **Zahl der Wiederholungen** über alle Messwerte ist mit vier pro Messstelle größer als drei
- ✔ **Varianzhomogenität** der Residuen (Abbildung A.9)
- ✔ **Normalverteilung** der Residuen (Abbildung A.9)

$\Longrightarrow$ Daten können zur Regression verwendet werden.

```
> summary(object = scabmodel)

Call:
lm(formula = scab ~ concentration, data = sulphur)

Residuals:
     Min       1Q   Median       3Q      Max
-12.9571  -2.9286  -0.5429   4.9000   9.1000
```

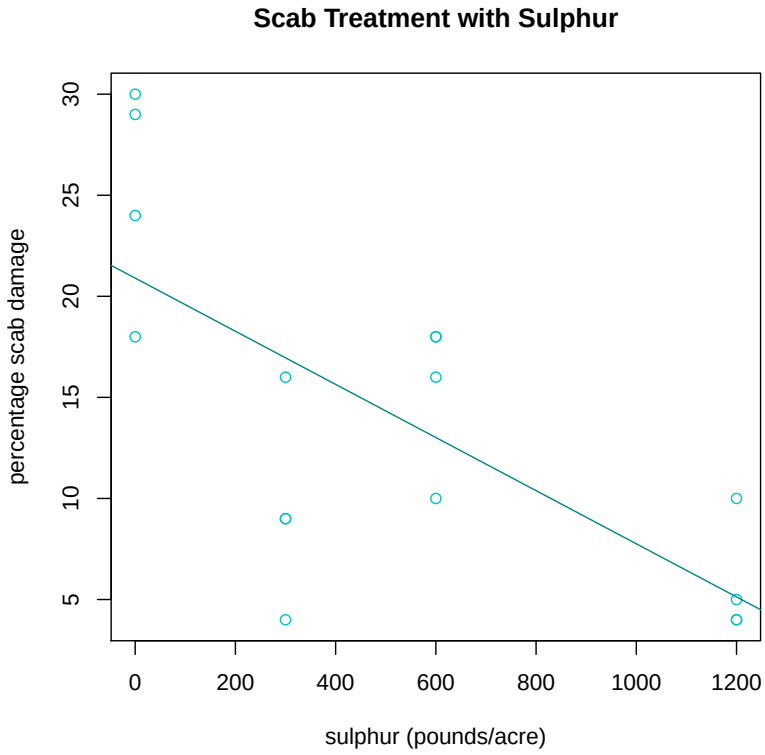


Abbildung A.8: Scatterplot der Kartoffelschorfdaten.

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   20.90000    2.44048   8.564 6.14e-07 ***
concentration -0.01314    0.00355  -3.702  0.00237 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.301 on 14 degrees of freedom
Multiple R-squared:  0.4946,        Adjusted R-squared:  0.4586
F-statistic:  13.7 on 1 and 14 DF,  p-value: 0.002369
```

Die Residuen sehen anhand der Quantile annährend normalverteilt aus. Die Geradengleichung lautet:

$$y = 20.9 - 0.01314x$$

Sowohl für den Achsenabschnitt als auch für die Steigung liegt eine hohe Signifikanz vor.

Mit folgenden Befehlen lassen sich die Konfidenz- und Vorhersagebänder zeichnen:

```
> pp <- predict(object = scabmodel, interval = "prediction",
+ data = sulphur$concentration)
> pc <- predict(object = scabmodel, interval = "confidence",
+ data = sulphur$concentration)
> plot(x = sulphur$concentration, y = sulphur$scab,
+ ylim = range(sulphur$scab, pc), col = "turquoise3",
```

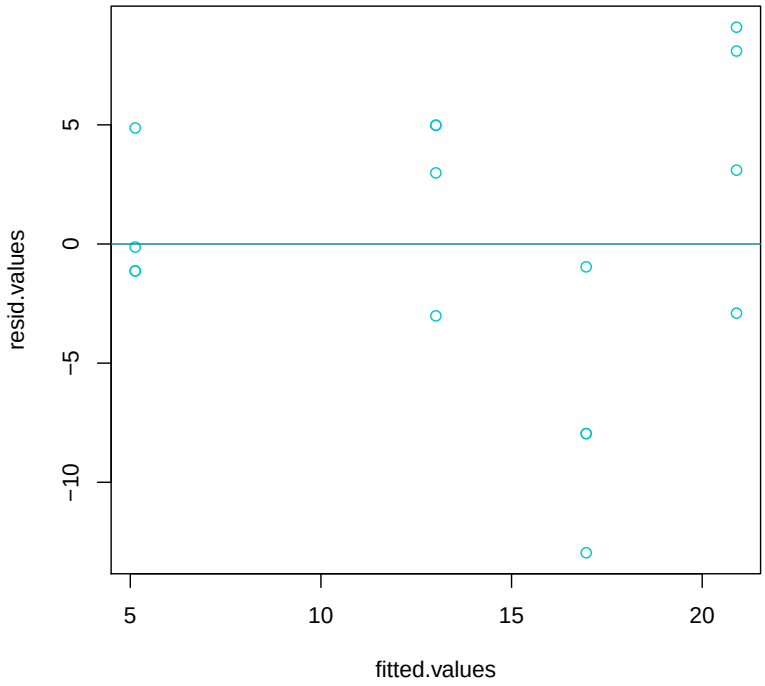


Abbildung A.9: Residuenplot der Kartoffelschorfdaten.

```
+ xlab = "application (pounds/acre)", ylab = "percentage scab damage",
+ main = "Confidence and Prediction Bands")
> matlines(x = sulphur$concentration, pp, tly = c(1,3),
+ col = "magenta3")
> matlines(x = sulphur$concentration, pc, tly = c(1,2,3),
+ col = "steelblue")
```

✍ Lösung 11

```
> cherry <- read.table(file = "../text/cherry.txt", sep = "\t",
+ header = TRUE)
```

Aufstellen eines linearen Modells:

```
> cherry.model <- lm(formula = response~treatment, data = cherry)
```

Graphische Darstellung der Residuen (Abbildung A.11):

```
> fitted.values <- fitted(object = cherry.model)
> resid.values <- resid(object = cherry.model)
> plot(x = fitted.values, y = resid.values, col = "blue3")
> abline(h = 0, col = "blue4")
```

Levene-Test zur Feststellung der Varianzhomogenität der Residuen innerhalb der Gruppen:

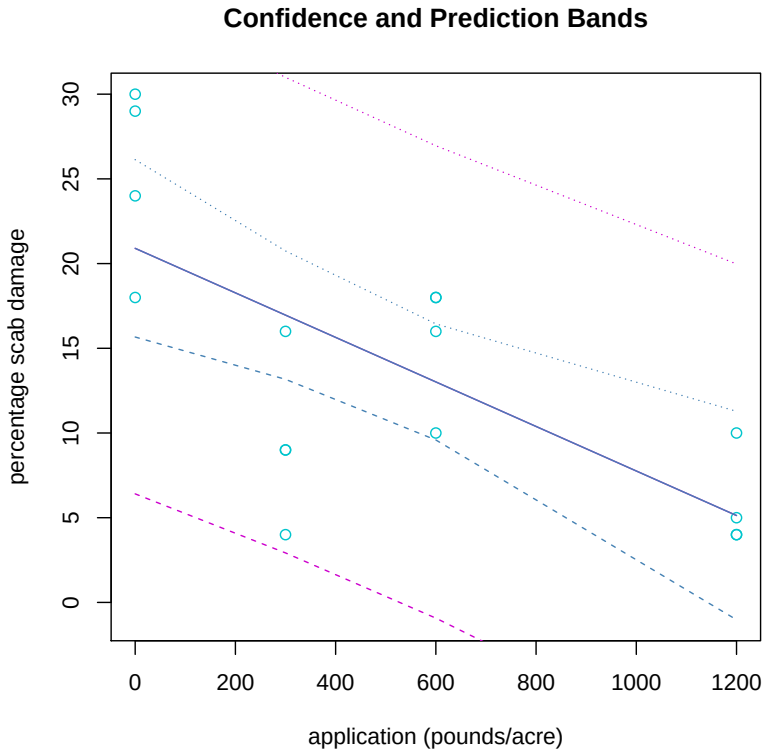


Abbildung A.10: Konfidenz und Vorhersagebänder zum linearen Regressionsmodell vom Kartoffelschorf.

```
> library(car)
> lev <- data.frame(res = resid.values, group = cherry$treatment)
> attach(lev)
> leveneTest(y = res, group = group)

Levene's Test for Homogeneity of Variance (center = median)
      Df F value    Pr(>F)
group  3  14.912 0.0002376 ***
      12
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> detach(lev)
```

- ✔ Varianzhomogenität wird aufgrund des Levene-Tests mit einem p-Wert größer als 0.05 angenommen
- ✔ Die Normalverteilung des Fehlerterms ist kritisch, hier wird Robustheit vorausgesetzt.
- ✔ unabhängige Daten

$\Longrightarrow$ ANOVA, die Daten sind ausreichend geeignet. Hypothesenpaare:

H_0: $\mu_{control} = \mu_{top}\mu_{bottom}\mu_{both}$
H_1: $\exists$ mindestens ein $\mu_{location} \neq \mu_{location'}$

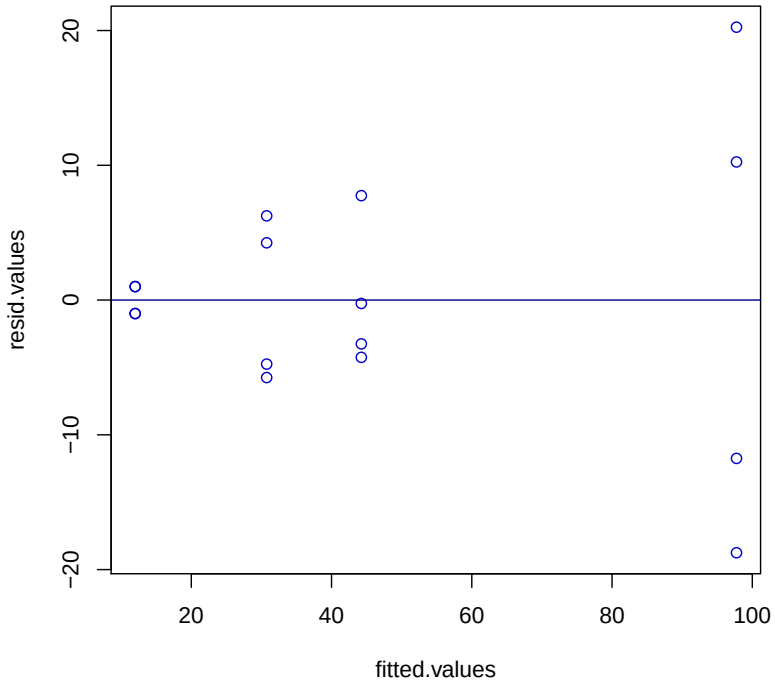


Abbildung A.11: Residuenplot der Cherrydaten.

```
> anova(object = cherry.model)

Analysis of Variance Table

Response: response
          Df  Sum Sq Mean Sq F value    Pr(>F)
treatment  3 16278.2  5426.1  53.801 3.124e-07 ***
Residuals 12  1210.3   100.9
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Der hoch signifikante p-Wert belegt, dass mindestens ein Unterschied bezüglich des Wasserverlustes bei den unterschiedlichen Behandlungsvarianten vorliegt.

✍ Lösung 12

```
> soil <- read.table(file = "../text/soil.txt", sep = "\t",
+ header = TRUE)
> boxplot(formula = moisture~treatment, data = soil, col = "purple",
+ ylab = "moisture", main = "Soil Moisture in Different Plots")


> library(car)
> leveneTest(y = soil$moisture, group = soil$treatment)
```

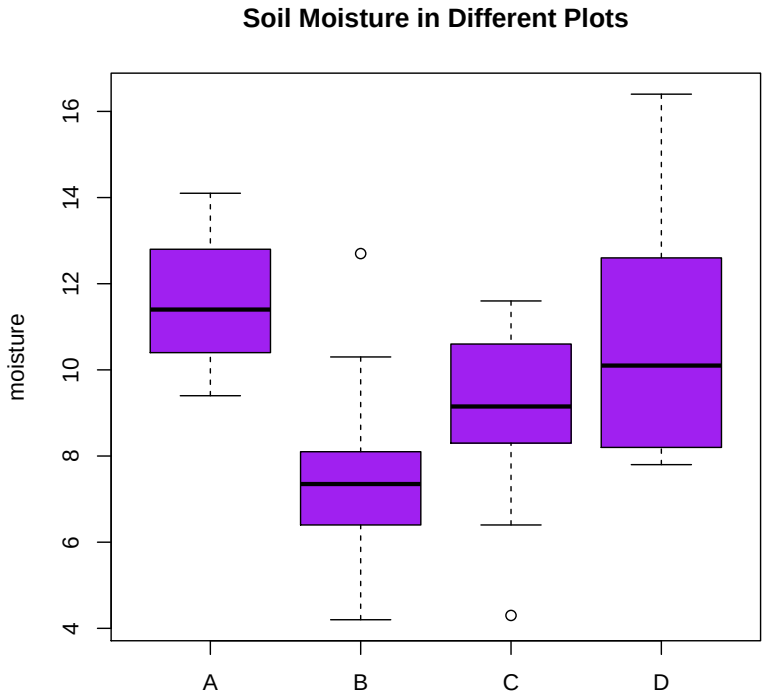


Abbildung A.12: Boxplots zur Bodenfeuchtigkeit vier verschiedener Bodentypen.

```
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  3  0.8003 0.5019
      36
```

- ✔ Annähernde **Normalverteilung** (Abbildung A.12) wird trotz Ausreißer angenommen.
- ✔ **Varianzhomogenität** wird mit einem p-Wert des Levene-Tests von 0.5 als belegt angesehen.

$\Longrightarrow$ Die Daten sind für einen Multiple Comparison Test geeignet. Ein all pairs-Vergleich nach Tukey bietet sich an, da keine Kontrollgruppe genannt wurde. Es wird zweiseitig auf Unterschied getestet. Hypothesenpaare:

H_0: $\mu_A = \mu_B = \mu_C = \mu_D$

H_1: $\mu_A \neq \mu_B$
$\mu_A \neq \mu_C$
$\mu_A \neq \mu_D$
$\mu_B \neq \mu_C$
$\mu_B \neq \mu_D$
$\mu_C \neq \mu_D$

```
> library(mvtnorm)
> library(multcomp)
```

```
> soil.model <- aov(formula = moisture~treatment, data = soil)
> soil.test <- glht(soil.model, linfct = mcp(treatment = "Tukey"),
+ alternative = "two.sided")
> summary(soil.test)

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: aov(formula = moisture ~ treatment, data = soil)

Linear Hypotheses:
           Estimate Std. Error t value Pr(>|t|)
B - A == 0   -3.870      1.046  -3.701  0.00386 **
C - A == 0   -2.620      1.046  -2.506  0.07604 .
D - A == 0   -0.710      1.046  -0.679  0.90440
C - B == 0    1.250      1.046   1.195  0.63368
D - B == 0    3.160      1.046   3.022  0.02285 *
D - C == 0    1.910      1.046   1.827  0.27782
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Adjusted p values reported -- single-step method)

> plot(confint(soil.test), col = "purple")
```

Mit einer Irrtumswahrscheinlichkeit von 5% wird für die Bodentypen A und B sowie D und B ein signifikanter Unterschied festgestellt. Für die anderen Bodentypen konnte kein Unterschied nachgewiesen werden.

In Abbildung A.13 ist erkennbar, dass der Feuchtigkeitsgehalt in Bodentyp D höher ist, als in Bodentyp D und dass er in A höher ist als in Typ B.

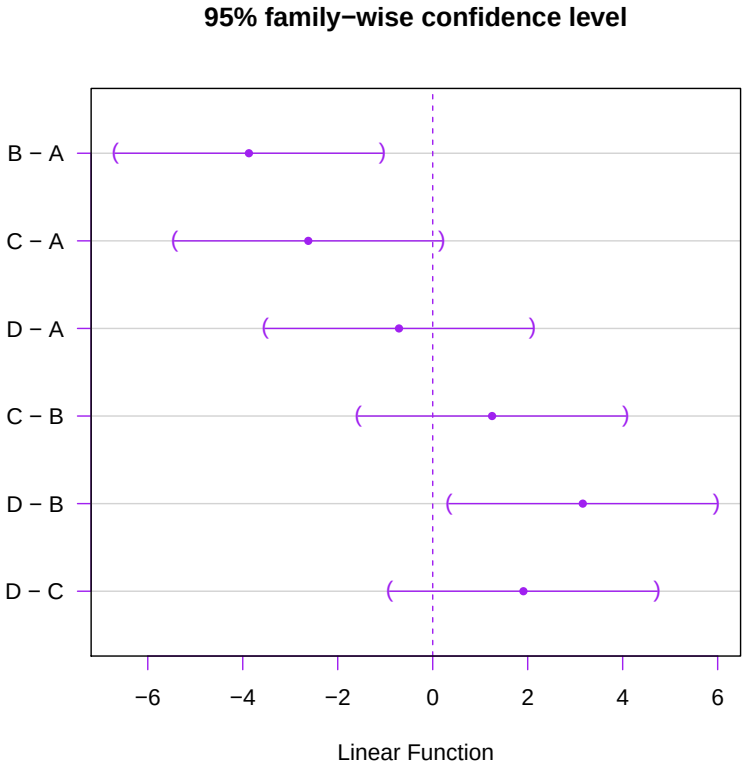


Abbildung A.13: Die Konfidenzintervalle zum Feuchtigkeitsgehalt von vier veschiedenen Bodentypen.

Anhang B

Cress Data

Light	Block	Height	Light	Block	Height	Light	Block	Height
red	A	2,1	SON-T	A	2,5	blue	A	2,15
red	A	2,2	SON-T	A	2,6	blue	A	2,4
red	A	2,7	SON-T	A	2,5	blue	A	2,15
red	A	1,85	SON-T	A	2,6	blue	A	2
red	A	2,4	SON-T	A	2,25	blue	A	2,1
red	A	2,2	SON-T	A	2,6	blue	A	2,1
red	A	2,55	SON-T	A	2,75	blue	A	2,35
red	A	2,55	SON-T	A	2,7	blue	A	1,95
red	A	2,6	SON-T	A	2,5	blue	A	2,2
red	A	3,05	SON-T	A	3,2	blue	A	2,4
red	A	2,45	SON-T	A	2,1	blue	A	2,1
red	A	2,75	SON-T	A	3,15	blue	A	2,2
red	A	2,55	SON-T	A	2,55	blue	A	2,1
red	A	2,65	SON-T	A	2,75	blue	A	2,45
red	A	2,6	SON-T	A	1,85	blue	A	2
red	B	2,2	SON-T	B	2,95	blue	B	1,95
red	B	2,5	SON-T	B	3,4	blue	B	2
red	B	2,4	SON-T	B	2,35	blue	B	2,3
red	B	2,95	SON-T	B	3,1	blue	B	1,8
red	B	2,8	SON-T	B	3,25	blue	B	2,5
red	B	3,2	SON-T	B	2,9	blue	B	2,4
red	B	2,25	SON-T	B	2,6	blue	B	2,1
red	B	2,7	SON-T	B	2,45	blue	B	2,15
red	B	2,4	SON-T	B	2,95	blue	B	2,3
red	B	2,35	SON-T	B	3,05	blue	B	2,5
red	B	2,6	SON-T	B	3,5	blue	B	2,1
red	B	2,8	SON-T	B	3,4	blue	B	2,3
red	B	2,1	SON-T	B	2,7	blue	B	2,35
red	B	2,75	SON-T	B	2,9	blue	B	2,3
red	B	2,3	SON-T	B	2,5	blue	B	1,95
red	C	2,5	SON-T	C	2,4	blue	C	2,05
red	C	2,9	SON-T	C	2,6	blue	C	2,35
red	C	2,8	SON-T	C	3,15	blue	C	2,1

red	C	2,5	SON-T	C	2,6	blue	C	2,1
red	C	2,7	SON-T	C	2,7	blue	C	1,75
red	C	3,05	SON-T	C	2,8	blue	C	1,95
red	C	2,5	SON-T	C	2,7	blue	C	2,35
red	C	2	SON-T	C	3,35	blue	C	2,2
red	C	2,7	SON-T	C	2,4	blue	C	2,6
red	C	2,7	SON-T	C	2,8	blue	C	1,65
red	C	2,8	SON-T	C	2,85	blue	C	1,75
red	C	2,6	SON-T	C	2,5	blue	C	2,2
red	C	2,9	SON-T	C	2,9	blue	C	2,1
red	C	3,1	SON-T	C	2,7	blue	C	1,9
red	C	2,5	SON-T	C	2,8	blue	C	2,25
daylight	A	2,5	white	A	2,5	green	A	2,55
daylight	A	2,4	white	A	2,8	green	A	2,35
daylight	A	2,3	white	A	2,2	green	A	2,8
daylight	A	2,15	white	A	3	green	A	2,55
daylight	A	1,6	white	A	2,7	green	A	2,9
daylight	A	2,35	white	A	2,7	green	A	2,4
daylight	A	1,95	white	A	2,75	green	A	2,3
daylight	A	2,5	white	A	2,7	green	A	2,75
daylight	A	2,7	white	A	2,35	green	A	3,1
daylight	A	2,75	white	A	2,8	green	A	2,9
daylight	A	2,6	white	A	2,5	green	A	2,8
daylight	A	2,8	white	A	2,8	green	A	2,75
daylight	A	2,4	white	A	3,4	green	A	2,5
daylight	A	2,15	white	A	3,3	green	A	2,4
daylight	A	2,25	white	A	2,55	green	A	3
daylight	B	2,4	white	B	2,65	green	B	2,5
daylight	B	2,55	white	B	2,2	green	B	2,7
daylight	B	2,3	white	B	2,7	green	B	3,2
daylight	B	2,9	white	B	2,2	green	B	2,9
daylight	B	2,75	white	B	2,5	green	B	2,6
daylight	B	2,85	white	B	2,5	green	B	3,4
daylight	B	2,3	white	B	1,5	green	B	3
daylight	B	2,85	white	B	2,25	green	B	2,2
daylight	B	2,2	white	B	2,15	green	B	2,5
daylight	B	2,45	white	B	2,5	green	B	2,15
daylight	B	2,2	white	B	1,85	green	B	2,8
daylight	B	2,55	white	B	3	green	B	2,8
daylight	B	2,3	white	B	2,2	green	B	2,8
daylight	B	2,3	white	B	2,1	green	B	2,75
daylight	B	2,2	white	B	2,7	green	B	3,2
daylight	C	2,55	white	C	3	green	C	2,95
daylight	C	2,45	white	C	2,95	green	C	2,9
daylight	C	2,35	white	C	2,25	green	C	2,6
daylight	C	2,35	white	C	2,95	green	C	3,2
daylight	C	2,7	white	C	2,7	green	C	2,8
daylight	C	2,6	white	C	2,7	green	C	2,8

daylight	C	2,1	white	C	2,5	green	C	2,9
daylight	C	2,15	white	C	2,3	green	C	2,95
daylight	C	2,55	white	C	2,2	green	C	2,75
daylight	C	2,45	white	C	2,8	green	C	2,75
daylight	C	2,5	white	C	3	green	C	2,8
daylight	C	2,65	white	C	2,5	green	C	2,4
daylight	C	2,65	white	C	2,9	green	C	2,6
daylight	C	2,65	white	C	1,8	green	C	2,6
daylight	C	2,2	white	C	2,45	green	C	3,2

Danksagung

Ich möchte Prof. Dr. A. L. Hothorn und Universitetslektor J.-E. Englund für die Unterstützung und Betreuung meiner Bachelor-Arbeit danken.

Besonderer Dank gebührt auch Dr. Frank Bretz, der das Thema ausschrieb und in der Anfangsphase einen Großteil der Betreuung übernahm.

Für geduldiges Korrekturlesen danke ich Cornelia Froemke, Alexandra Hoff, Xuefei Mi und Barbara Zinck.

Ohne die schnelle Hilfe von Brian Fynn, der mir bei Versagen meines Computers ein Notebook zur Verfügung stellte, wäre die Arbeit nicht bis zum Abgabetermin fertig geworden.

Prof. Dr. Klaus Hoff, Linus Masumbuko, Prof. Dr. Jan Petersen und Richard Zinck danke ich für Diskussionen, Unterstützung und Motivation.

Literaturverzeichnis

Ahern, T. (1998). *Statistical analysis of EIN plants treated with ancymidol and H_20.* Oberlin College. Unpublished manuscript.

Baur, E., Fischer, E., and Lenz, F. (1931). *Human Heredity, 3rd edition.* Macmillan, New York.

Bishop, O. N. (1980). *Statistics for biology - A practical guide fo the experimental biologist, 3rd edition.* Longman, Longman House, Burnt Mill, Harlow, Essex.

Cochran, W. G. and Cox, G. M. (1950). *Experimental designs.* John Wiley & Sons, Ltd, New York, Second Edition 1957.

Collins, C. and Seeney, F. (1999). *Statistical Experiment Design and Interpretation - An Introduction with Agricultural Examples.* John Wiley & Sons, Ltd, Baffins Lane, Chichester, West Sussex PO19 1UD, England.

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

Fierer, N. (1994). *Statistical analysis of soil respiration rates in a light gap and surrounding old-growth forest.* Oberlin College. Unpublished manuscript.

Froemke, C. (2004). *Einführung in die Biometrie für Gartenbauer.* Lehrgebiet für Bioinformatik, Universität Hannover. Unveröffentlichtes Übungsskript.

Gent, A. (1999). Oberlin College. Unpublished data collected at Oberlin College.

Gentleman, R. (2005). Reproducible research: A bioinformatics case study. *bepress (http://www.bepress.com/sagmb)*, 4, Issue 1.

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and Ostrowski, E. (1994). *A Handbook of Small Data Sets.* Chapman & Hall, Great Britain.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6:65–70.

Knight, S. L. and Mitchell, C. A. (2000). Enhancement of lettuce yield by manipulation of light and nitrogen nutrition. *Journal of the American Society for Horticultural Science*, 108:750 – 754.

Martinez, J. (1998). *Organic practices for the cultivation of sweet corn.* Oberlin College. Unpublished manuscript.

Mead, R., Curnow, R. N., and Hasted, A. M. (2003). *Statistical Methods in Agriculture and Experimental Biology.* Chapman & Hall/CRC, CRC Press LLC, 2000 N. W. Corporate Blvd., Boca Raton, Florida 33431.

Neumann, A., Richards, A.-L., and Randa, J. (2001). *Effects of acid rain on alfalfa plants.* Oberlin College. Unpublished manuscript.

Norlinger, C. and Hoff, K. J. (2004). *The effect of light quality on garden cress*. Swedish University of Agricultural Sciences. Unpublished project report.

Pappas, T. and Mitchell, C. A. (1984). Effects of seismic stress on the vegetative growth of glycine max (l.) merr. cv. wells ii. *Plant, Cell and Environment*, 8:143 – 148.

Pearce, S. C. (1983). *The Agricultural Field Experiment*. John Wiley & Sons, Ltd, Chicester, New York, Brisbane, Toronto, Singapore.

R Development Core Team (2004a). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-00-3.

R Development Core Team (2004b). *R Installation and Administration (Version 2.0.1., 2004-11-15)*. 17.01.2004 http://www.r-project.org.

Saedi, G. and Rowland, G. G. (1997). The inheritance of variegated seed color and palmitic acid in flax. *Journal of Heredity*, 88:466 – 468.

Samuels, M. L. and Witmer, J. A. (2003). *Statistics for the Life Sciences, 3rd edition*. Pearson Education, Inc., Upper Saddle River, New Jersey 07458.

Stallman, R. (1991). *GNU General Public License, 2nd edition 1991*. 59 Temple Place, Suite 330, Boston, USA.

Wonnacott, T. H. and Wonnacott, R. J. (1990). *Introductory Statistics*. John Wiley & Sons, New York, Chichester, Brisbane, Toronto, Singapore. 5th edition.