AS17: Kapitel 9 - Das Programm R

Jürgen Hedderich

2020-04-29

Contents

Hinweis: Die Gliederung zu den Befehlen Und Ergebnissen in R orientiert sich an dem Inhaltsverzeichnis der 17. Auflage der 'Angewandten Statistik'. Nähere Hinweise zu den verwendeten Formeln sowie Erklärungen zu den Beispielen sind in dem Buch (Ebook) nachzulesen!

zur Druckversion

Hinweis: Die thematische Gliederung zu den aufgeführten Befehlen und Beispielen orientiert sich an dem Inhaltsverzeichnis der 17. Auflage der 'Angewandten Statistik'. Nähere Hinweise zu den verwendeten Formeln sowie Erklärungen zu den Beispielen sind in dem Buch (Ebook) nachzulesen!

Aktuelle R-Umgebung zu den Beispielen

The R Project for Statistical Computing

```
sessionInfo()
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18363)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=German_Germany.1252  LC_CTYPE=German_Germany.1252
## [3] LC_MONETARY=German_Germany.1252 LC_NUMERIC=C
## [5] LC_TIME=German_Germany.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_3.6.3  magrittr_1.5    tools_3.6.3     htmltools_0.4.0
##  [5] yaml_2.2.1      Rcpp_1.0.4      stringi_1.4.6   rmarkdown_2.1
##  [9] knitr_1.28      stringr_1.4.0   xfun_0.12       digest_0.6.25
## [13] rlang_0.4.5     evaluate_0.14
```

Download von Datensätzen, die in den folgenden Beispielen verwendet werden:

Infarkt-Daten: infarkt

9.1 Befehle im Konsolfenster

```
mean(c(4,6,8,9))                    # Mittelwertberechnung
## [1] 6.75

sqrt(5)                             # Wurzelfunktion
## [1] 2.236068

round(5.23454, digits=3)            # Rundung auf 3 Dezimalziffern
## [1] 5.235
```

9.2 Hilfestellung in R

```
# help()

# help(sqrt)                        # Quadratwurzel

# ?srt

# ?mean

example(mean)                       # arithmetischer Mittelwert
##
## mean> x <- c(0:10, 50)
##
## mean> xm <- mean(x)
##
## mean> c(xm, mean(x, trim = 0.10))
## [1] 8.75 5.50
```

9.3 Objekte in R

```
x <- 1:10                           # numerisch

length(x)
## [1] 10
mode(x)
## [1] "numeric"
```

```
name <- c("Statistik","Mathematik") # Zeichenketten

length(name)
## [1] 2
mode(name)
## [1] "character"
```

```
logic <- c(TRUE, FALSE)             # logisch

mode(logic)
## [1] "logical"
```

```
wurzel.12 <- sqrt(12)               # Variablen

wurzel.12                           # Wurzel aus 12
## [1] 3.464102
```

```
data(Titanic)                       # Datenrahmen (Tabellenstruktur)

str(Titanic)                        #
##  'table' num [1:4, 1:2, 1:2, 1:2] 0 0 35 0 0 0 17 0 118 154 ...
##  - attr(*, "dimnames")=List of 4
##   ..$ Class   : chr [1:4] "1st" "2nd" "3rd" "Crew"
##   ..$ Sex     : chr [1:2] "Male" "Female"
##   ..$ Age     : chr [1:2] "Child" "Adult"
##   ..$ Survived: chr [1:2] "No" "Yes"
```

```
mat <- matrix(c("a","b","c","d"),
              nrow=2)               # Matrix
mat
##      [,1] [,2]
## [1,] "a"  "c"
## [2,] "b"  "d"

is.matrix(mat)
## [1] TRUE

is.numeric(mat)
## [1] FALSE
```

```
library(date)                          # Datumsangaben

dat <- as.date(c("15jul2008","10mar1980"))
dat
## [1] 15Jul2008 10Mar80

(dat[1] - dat[2])/360
## [1] 28.76111

dat1 <- mdy.date(7, 15, 2008)
dat2 <- mdy.date(3, 10, 1980)

str(dat1); str(dat2)
##  'date' int 17728
##  'date' int 7374

(dat1 - dat2)/360
## [1] 28.76111
```

```
ls()                                   # Objekte im 'Arbeitsspeicher'
##  [1] "colorize"  "dat"       "dat1"      "dat2"      "logic"     "mat"
##  [7] "name"      "opar"      "Titanic"   "wurzel.12" "x"         "xm"

rm(list = ls())                        # Löschen aller Objekte im Arbeitspeicher
```

9.3.2 Erzeugen von Vektoren und Dateneingabe

```
1:10
##  [1]  1  2  3  4  5  6  7  8  9 10

20:15
## [1] 20 19 18 17 16 15

seq(1, 5, by=0.5)
## [1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0

seq(1, 5, length=11)
##  [1] 1.0 1.4 1.8 2.2 2.6 3.0 3.4 3.8 4.2 4.6 5.0

rep(5, 20)
##  [1] 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5

c(1, 7:9)
## [1] 1 7 8 9

c(1:5, 10.5, "next")
## [1] "1"    "2"    "3"    "4"    "5"    "10.5" "next"
```

9.3.3 Faktoren in R - Klasseneinteilungen

```
m   <- rep(1, 15); w   <- rep(2, 20)
sex <- c(m, w)

faktor <- factor(sex, levels=1:2, labels=c("männlich", "weiblich"))
summary(faktor)
## männlich weiblich
##       15       20
```

```
alter   <- rnorm(50, mean=40, sd=10)
summary(alter)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.86   29.27   36.17   36.84   42.17   72.90

alter.k <- cut(alter, breaks=c(10, 20, 30, 40, 50, 60, 70))
summary(alter.k)
## (10,20] (20,30] (30,40] (40,50] (50,60] (60,70]    NA's
##       2      10      19      11       6       0       2
```

9.3.4 Matrizen und Tabellen

```
mat <- matrix(1:9, ncol=3, byrow=T); mat
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9

mat[1,]                                  # Zeilen
## [1] 1 2 3

mat[,2]                                  # Spalten
## [1] 2 5 8

mat[2,2]                                 # Zellen
## [1] 5

margin.table(mat, 1)                     # Zeilensummen
## [1]  6 15 24

margin.table(mat, 2)                     # Spaltensummen
## [1] 12 15 18
```

```
mat <- rbind(mat, c(10, 11, 12)); mat       # Zeilen ergänzen (Verbinden)
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
## [4,]   10   11   12
```

```
mat <- cbind(mat, c(13, 14, 15, 16)); mat   # Spalten ergänzen (verbinden)
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3   13
## [2,]    4    5    6   14
## [3,]    7    8    9   15
## [4,]   10   11   12   16
```

```
count <- matrix(c(12, 6, 5, 10),            # Zeilen- / Spaltenbezeichnungen
         ncol=2, byrow=T,
         dimnames=list(c("Zeile 1", "Zeile 2"), c("Spalte 1", "Spalte 2")))
count
##         Spalte 1 Spalte 2
## Zeile 1       12        6
## Zeile 2        5       10
```

9.3.5 Rechnen mit Matrizen

```
n    <- 10
data <- matrix(c(11, 6,11, 9,  8, 6, 2, 7,  4, 9,10, 4,
                  6, 4, 9, 6,  5, 4, 1, 2, 10,10, 6, 9,
                  6, 8, 2, 1,  2, 2, 8, 4,  8, 2, 4,10,
                  9, 7, 6, 9), nrow=n, byrow=T,
        dimnames = list(
        c("S1","S2","S3","S4","S5","S6","S7","S8","S9","S10"),
        c("V.1", "V.2", "V.3","V.4"))); data
##     V.1 V.2 V.3 V.4
## S1   11   6  11   9
## S2    8   6   2   7
## S3    4   9  10   4
## S4    6   4   9   6
## S5    5   4   1   2
## S6   10  10   6   9
## S7    6   8   2   1
## S8    2   2   8   4
## S9    8   2   4  10
## S10   9   7   6   9

one <- rep(1,10); one           # Vektor mit Einsen "1"
##  [1] 1 1 1 1 1 1 1 1 1 1

summe <- one %*% data; summe    # Summen über "inner product" mit Einsen
##      V.1 V.2 V.3 V.4
## [1,]  69  58  59  61

mittel <- summe / n; mittel     # Mittelwerte
##      V.1 V.2 V.3 V.4
## [1,] 6.9 5.8 5.9 6.1

                                # Abweichungen vom Mittelwert
deviat <- data - one %*% mittel; round(deviat,2)
##      V.1  V.2  V.3  V.4
```

```
## S1   4.1  0.2  5.1  2.9
## S2   1.1  0.2 -3.9  0.9
## S3  -2.9  3.2  4.1 -2.1
## S4  -0.9 -1.8  3.1 -0.1
## S5  -1.9 -1.8 -4.9 -4.1
## S6   3.1  4.2  0.1  2.9
## S7  -0.9  2.2 -3.9 -5.1
## S8  -4.9 -3.8  2.1 -2.1
## S9   1.1 -3.8 -1.9  3.9
## S10  2.1  1.2  0.1  2.9

                                # Kovarianzmatrix
covar <- t(deviat) %*% deviat/(n - 1); round(covar,2)
##      V.1   V.2   V.3   V.4
## V.1 7.88  2.76  0.32  6.79
## V.2 2.76  7.73  0.87 -0.09
## V.3 0.32  0.87 12.77  3.46
## V.4 6.79 -0.09  3.46 10.32

                                # Varianzen aus der Diagonalen
varia <- diag(covar); round(varia,2)
##   V.1   V.2   V.3   V.4
##  7.88  7.73 12.77 10.32

                                # Korrelationskoeffizienten
sdi <- diag(1/sqrt(diag(covar)))
correl <- sdi %*% covar %*% sdi; round(correl,2)
##      [,1]  [,2] [,3]  [,4]
## [1,] 1.00  0.35 0.03  0.75
## [2,] 0.35  1.00 0.09 -0.01
## [3,] 0.03  0.09 1.00  0.30
## [4,] 0.75 -0.01 0.30  1.00
```

9.3.6 Daten im Rahmen (Tabellen)

```
alter      <- c(19, 22, 24)

geschlecht <- c("maennlich","weiblich","maennlich")

groesse    <- c(170, 165, 181)

studenten  <- data.frame(alter, geschlecht, groesse)
studenten
```

alter	geschlecht	groesse
19	maennlich	170
22	weiblich	165
24	maennlich	181

```
studenten[2,]                    # Auswahl 2. Zeile (fallweise)
```

	alter	geschlecht	groesse
2	22	weiblich	165

```
studenten[geschlecht=="maennlich",]  # Auswahl nach dem Geschlecht
```

	alter	geschlecht	groesse
1	19	maennlich	170
3	24	maennlich	181

Einlesen von Daten aus externe Datei (CSv-Format)

```
infarkt <- read.csv2("infarkt.csv")

infarkt[1:10, 1:9]
```

Nummer	Gruppe	Geschlecht	Alter	RelGew	RRsyst	RRdias	Blutz	Diabet
1	Infarkt	m	39	128	155	95	100	FALSE
2	Infarkt	m	43	106	145	95	140	TRUE
3	Infarkt	m	45	94	160	90	95	FALSE
4	Infarkt	m	50	98	200	110	80	FALSE
5	Infarkt	m	51	132	190	110	90	FALSE
6	Infarkt	m	52	108	195	110	90	FALSE
7	Infarkt	m	53	165	220	105	70	FALSE
8	Infarkt	m	53	116	140	85	85	FALSE
9	Infarkt	m	56	138	180	100	200	TRUE
10	Infarkt	m	56	140	175	95	90	FALSE

```
str(infarkt)                          # Datenstruktur
## 'data.frame':    80 obs. of  14 variables:
##  $ Nummer    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Gruppe    : Factor w/ 2 levels "Infarkt","Kontrolle": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Geschlecht: Factor w/ 2 levels "m","w": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Alter     : int  39 43 45 50 51 52 53 53 56 56 ...
##  $ RelGew    : int  128 106 94 98 132 108 165 116 138 140 ...
##  $ RRsyst    : int  155 145 160 200 190 195 220 140 180 175 ...
##  $ RRdias    : int  95 95 90 110 110 110 105 85 100 95 ...
##  $ Blutz     : int  100 140 95 80 90 90 70 85 200 90 ...
##  $ Diabet    : logi  FALSE TRUE FALSE FALSE FALSE FALSE ...
##  $ Chol      : int  195 205 245 190 260 190 340 195 285 380 ...
##  $ Trigl     : int  146 138 200 101 202 164 150 93 135 127 ...
##  $ Hbdh      : int  410 380 260 165 300 290 605 170 277 523 ...
##  $ Got       : num  28.3 19 15.4 23.2 20.6 18.7 33.1 20.2 18.2 29.7 ...
##  $ Zigarr    : int  30 0 15 15 5 20 0 5 20 10 ...

attach(infarkt)                       # vereinfachter Zugriff über Namen (cave!!!)

# edit(infarkt)                       # Änderungen in der Tabelle
```

```
attach(infarkt)

Chol
##  [1] 195 205 245 190 260 190 340 195 285 380 220 240 235 215 190 275 205 290 200
## [20] 210 220 265 235 200 350 220 800 230 185 295 380 200 485 210 185 210 395 290
## [39] 190 210 220 200 185 220 215 135 220 180 220 135 150 165 150 195 160 190  85
## [58] 180 160 200 205 230 125 195 100 185 180 205 160 195 190 175 140 145 170 150
## [77] 180 190 175 200

mean(Chol)
## [1] 219.75
```

Berechnungen / Transformationen / Rekodierungen:

```
infarkt <- transform(infarkt, lnHbdh=log(Hbdh))

infarkt$lnHbdh
##  [1] 6.016157 5.940171 5.560682 5.105945 5.703782 5.669881 6.405228 5.135798
##  [9] 5.624018 6.259581 6.595781 5.433722 5.880533 5.652489 5.517453 5.783825
## [17] 5.379897 5.666427 4.955827 5.463832 4.934474 6.602588 6.434547 6.495266
## [25] 5.252273 5.541264 6.198479 5.631212 6.200509 5.918894 6.107023 5.560682
## [33] 4.859812 5.609472 5.572154 5.743003 6.177944 6.077642 6.175867 5.799093
## [41] 4.276666 5.247024 4.584967 5.003946 4.532599 4.204693 4.356709 4.007333
## [49] 4.510860 4.882802 4.330733 4.955827 3.713572 4.553877 4.234107 4.615121
## [57] 5.036953 5.111988 4.672829 5.327876 4.700480 4.644391 3.850148 4.682131
## [65] 4.465908 4.905275 4.941642 5.267858 5.575949 4.488636 4.077537 5.017280
## [73] 4.007333 5.043425 4.969813 5.081404 4.890349 4.812184 4.605170 4.605170
```

```
replace(1:10, list=c(2, 4, 6), values=c(20, 40, 60))  #Funktion replace()
##  [1]  1 20  3 40  5 60  7  8  9 10
```

```
re.code <- function(var, alt, neu) {      # Umkodieren einzelner Faktorstufen
            x <- as.vector(var)
            i <- which(x == alt); ni <- length(i)
            x <- replace(x, list=i, values=rep(neu, ni))
            if (is.factor(x)) factor(x) else x }

re.code(as.factor(c("A", "A", "A", "B", "B", "C", "C")), "B", "X")
## [1] "A" "A" "A" "X" "X" "C" "C"

infarkt <- transform(infarkt, Gruppe=re.code(Gruppe, "Infarkt", "1"))

infarkt <- transform(infarkt, Gruppe=re.code(Gruppe, "Kontrolle", "0"))

infarkt$Gruppe
##  [1] "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1"
## [20] "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1"
## [39] "1" "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
## [58] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
## [77] "0" "0" "0" "0"
```

Funktion recode() in library(car)

```
library(car)
farben <- c("red", "purple", "blue", "blue", "orange", "red", "orange")
farben
## [1] "red"    "purple" "blue"   "blue"   "orange" "red"    "orange"

recode(farben, "'red'='rot'; 'blue'='blau'; 'purple'='violett'")
## [1] "rot"     "violett" "blau"    "blau"    "orange"  "rot"     "orange"
```

Stutzen der Verteilung (Nachweisgrenze)

```
daten <- c(5, 9, 11, 8, 9, 3, 1, 13, 9, 12, 5, 12, 6, 3, 17, 5, 8, 7)

cutoff <- 10
ifelse(daten <= cutoff, daten, cutoff)
##  [1]  5  9 10  8  9  3  1 10  9 10  5 10  6  3 10  5  8  7
```

Klasseneinteilung (kategoriell)

```
infarkt <- read.csv2("infarkt.csv")
infarkt$Alter
##  [1] 39 43 45 50 51 52 53 53 56 56 57 57 59 59 61 61 62 62 67 68 68 71 72 74 75
## [26] 76 79 81 57 58 61 63 66 67 69 69 70 72 79 81 28 40 42 46 51 51 52 53 53 54
## [51] 55 55 57 58 59 60 61 61 62 62 62 64 64 65 65 36 42 42 45 48 49 49 51 52 55
## [76] 56 59 61 63 72

Klassen <- cut(infarkt$Alter, breaks=c(0, 50, 70, Inf),
                    labels=c("-50", ">50-70", ">70"))
Klassen
##  [1] -50    -50    -50    -50    >50-70 >50-70 >50-70 >50-70 >50-70 >50-70
```

```
## [11] >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70
## [21] >50-70 >70    >70    >70    >70    >70    >70    >70    >50-70 >50-70
## [31] >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >70    >70    >70
## [41] -50    -50    -50    -50    >50-70 >50-70 >50-70 >50-70 >50-70 >50-70
## [51] >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70
## [61] >50-70 >50-70 >50-70 >50-70 >50-70 -50    -50    -50    -50    -50
## [71] -50    -50    >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >50-70 >70
## Levels: -50 >50-70 >70
```

Wiederholte Beobachtungen:

```
c1 <- c(1,2,3,4)
c2 <- c(2,5,6,7)
c3 <- c(3,8,9,10)

df <- as.data.frame(rbind(c1,c2,c3))
names(df) <- c("Fall","t1","t2","t3"); df        # weites Format von Daten
```

	Fall	t1	t2	t3
c1	1	2	3	4
c2	2	5	6	7
c3	3	8	9	10

```
reshape(df, timevar="Zeit", idvar="Fall", varying=list(2:4),
        v.names="Wert", direction="long")        # langes Format
```

	Fall	Zeit	Wert
1.1	1	1	2
2.1	2	1	5
3.1	3	1	8
1.2	1	2	3
2.2	2	2	6
3.2	3	2	9
1.3	1	3	4
2.3	2	3	7
3.3	3	3	10

9.4 Fehlende Angaben (not assigned NA)

```
x <- c(1, 2, NA, 4); x
## [1]  1  2 NA  4

y <- x + 1; y
## [1]  2  3 NA  5

is.na(x)
## [1] FALSE FALSE  TRUE FALSE

which(is.na(x))                                    # Auffinden fehlender Angaben
## [1] 3
```

Fehlende Anbgaben in Funktionsaufrufen!

```
x <- c(1, 2, NA, 4); x
## [1]  1  2 NA  4

mean(x)
## [1] NA

mean(x, na.rm=TRUE)
## [1] 2.333333
```

Fehlende Angaben in Datentabellen!

```
any.missing.value <- function(dfr=NA) { any(is.na(dfr)) }

alter      <- c(19,   22,  24,  30,  NA,  25)
geschlecht <- c("maennlich","weiblich","maennlich",NA,"weiblich","maennlich")
groesse    <- c(170, NA, 181, 177, 182, 196)
studenten  <- data.frame(alter, geschlecht, groesse)
studenten
```

alter	geschlecht	groesse
19	maennlich	170
22	weiblich	NA
24	maennlich	181
30	NA	177
NA	weiblich	182
25	maennlich	196

```
y <- studenten[,c("alter","geschlecht","groesse")]

y.na <- !apply(y, 1, any.missing.value); y[y.na,]
```

	alter	geschlecht	groesse
1	19	maennlich	170
3	24	maennlich	181
6	25	maennlich	196

```
colSums(is.na(studenten))
##      alter geschlecht    groesse
##          1          1          1

na.omit(studenten)
```

	alter	geschlecht	groesse
1	19	maennlich	170
3	24	maennlich	181
6	25	maennlich	196

```
studenten[complete.cases(studenten),]
```

	alter	geschlecht	groesse
1	19	maennlich	170
3	24	maennlich	181
6	25	maennlich	196

9.5 Auswahl und Sortierung von Daten

```
Zahl1bis20 <- 1:20

Zahl1bis20[6:10]
## [1]  6  7  8  9 10

Zahl1bis20[Zahl1bis20 > 13]
## [1] 14 15 16 17 18 19 20

blut <- c("A","B","AB","0")

blut[3]
## [1] "AB"
```

```
infarkt <- read.csv2("infarkt.csv")
attach(infarkt)

infarkt[, 10]
##  [1] 195 205 245 190 260 190 340 195 285 380 220 240 235 215 190 275 205 290 200
## [20] 210 220 265 235 200 350 220 800 230 185 295 380 200 485 210 185 210 395 290
## [39] 190 210 220 200 185 220 215 135 220 180 220 135 150 165 150 195 160 190  85
## [58] 180 160 200 205 230 125 195 100 185 180 205 160 195 190 175 140 145 170 150
## [77] 180 190 175 200

Chol[5]
## [1] 260

set <- infarkt[Gruppe=="Infarkt" & Blutz>100, ]
set[, 1:9]
```

	Nummer	Gruppe	Geschlecht	Alter	RelGew	RRsyst	RRdias	Blutz	Diabet
2	2	Infarkt	m	43	106	145	95	140	TRUE
9	9	Infarkt	m	56	138	180	100	200	TRUE
14	14	Infarkt	m	59	114	190	120	110	FALSE
16	16	Infarkt	m	61	136	140	80	130	TRUE
20	20	Infarkt	m	68	114	180	105	105	FALSE
31	31	Infarkt	w	61	152	165	105	160	TRUE
37	37	Infarkt	w	70	150	165	95	130	TRUE
38	38	Infarkt	w	72	121	160	95	110	FALSE

```
subset(infarkt, Geschlecht=="w" & Alter<45, select = c(Chol, Trigl))
```

	Chol	Trigl
66	185	103
67	180	207
68	205	65

```r
alter      <- c(19,   22,  24,  30,  NA,  25)
geschlecht <- c("maennlich","weiblich","maennlich",NA,"weiblich","maennlich")
groesse    <- c(170, NA, 181, 177, 182, 196)
studenten  <- data.frame(alter, geschlecht, groesse)
studenten
```

alter	geschlecht	groesse
19	maennlich	170
22	weiblich	NA
24	maennlich	181
30	NA	177
NA	weiblich	182
25	maennlich	196

```r
studenten[geschlecht=="maennlich",]
```

	alter	geschlecht	groesse
1	19	maennlich	170
3	24	maennlich	181
NA	NA	NA	NA
6	25	maennlich	196

Sortieren / Rangzahlen

```r
a <- c(3, 7, 2, 8, 5, 10, 4); a
## [1]  3  7  2  8  5 10  4

sort(a)
## [1]  2  3  4  5  7  8 10

sort(a, decreasing=TRUE)
## [1] 10  8  7  5  4  3  2

a <- c(3, 7, 2, 8, 5, 10, 4)
rank(a)
## [1] 2 5 1 6 4 7 3

b <- c(3, 5, 7, 3, 6, 5)
rank(b)
## [1] 1.5 3.5 6.0 1.5 5.0 3.5

rank(b, ties.method = "min")
## [1] 1 3 6 1 5 3

o <- order(a)
a
## [1]  3  7  2  8  5 10  4
o
## [1] 3 1 7 5 2 4 6
```

```
a[o]
## [1]  2  3  4  5  7  8 10
```

Sortieren in Matrizen und Datentabellen

```
SortMat <- function(Mat, Sort)  {
        m <- do.call("order", as.data.frame(Mat[, Sort]))
        Mat[m, ]                }
```

```
alter      <- c(19,   22,  24,  30,  NA,  25)
geschlecht <- c("maennlich","weiblich","maennlich",NA,"weiblich","maennlich")
groesse    <- c(170, NA, 181, 177, 182, 196)
studenten  <- data.frame(alter, geschlecht, groesse)
studenten
```

alter	geschlecht	groesse
19	maennlich	170
22	weiblich	NA
24	maennlich	181
30	NA	177
NA	weiblich	182
25	maennlich	196

```
SortMat(studenten, 3)
```

	alter	geschlecht	groesse
1	19	maennlich	170
4	30	NA	177
3	24	maennlich	181
5	NA	weiblich	182
6	25	maennlich	196
2	22	weiblich	NA

```
studenten[order(studenten[,3]),]
```

	alter	geschlecht	groesse
1	19	maennlich	170
4	30	NA	177
3	24	maennlich	181
5	NA	weiblich	182
6	25	maennlich	196
2	22	weiblich	NA

```r
m <- matrix(c(2,5,4, 5,2,3, 2,6,7), byrow=T, nrow=3); m
##      [,1] [,2] [,3]
## [1,]    2    5    4
## [2,]    5    2    3
## [3,]    2    6    7
SortMat(m, 2)
##      [,1] [,2] [,3]
## [1,]    5    2    3
## [2,]    2    5    4
## [3,]    2    6    7
```

9.6 Ablaufsteuerung: logische Bedingungen und Funktionen

for (...) und if (...)

```
a <- rep(NA, 10)                                          # for (...) if (...)
a
##  [1] NA NA NA NA NA NA NA NA NA NA

for (i in 1:10) if (i<6) a[i]<-"unten" else a[i]<-"oben"

a
##  [1] "unten" "unten" "unten" "unten" "unten" "oben"  "oben"  "oben"  "oben"
## [10] "oben"
```

ifelse()

```
zaehler <- c(2, 4,NA, 8, 7, 5, 3, 1)                      # ifelse()
nenner  <- c(4, 3, 2, 1, 0, 3, 2, 0)

quot <- round(ifelse(nenner != 0, zaehler/nenner, NA), 3)

round(quot, 2)
## [1] 0.50 1.33   NA 8.00   NA 1.67 1.50   NA
```

while (...)

```
i <- 0; summe <- 0                                        # while()
while (i < 10) {i <- i+1;  summe <- summe + i }
summe
## [1] 55

sum(1:10)
## [1] 55
```

function()

```
stdabw <- function(x) {                                   # function()
          anzahl <- length(x)
          summe  <- sum(x)
          mittel <- summe / anzahl
          saq    <- sum((x-mittel)^2)
          return(sqrt(saq/(anzahl-1)))
                      }

x <- c (2,3,4,5,6,7)
stdabw(x)
## [1] 1.870829
```

apply()

```
                                                    # apply functions
X <- matrix(sample(1:20, 20, replace=T), ncol=4, byrow=T)
X
##      [,1] [,2] [,3] [,4]
## [1,]   15   11   13   19
## [2,]    3   17    8    3
## [3,]   13    1   14    7
## [4,]   13    3    6    3
## [5,]   19   12   14   20

apply(X, 1, sum)                                    # Zeilensummen
## [1] 58 31 35 25 65

apply(X, 2, sum)                                    # Spaltensummen
## [1] 63 44 55 52

apply(X, 2, which.max)                              # Pos. max. Wert je Spalte
## [1] 5 2 3 5

apply(X, 1, function(x) {max(x) - min(x)})          # Spannweite je Zeile
## [1]  8 14 13 10  8

apply(X, 1:2, function(x) sqrt(x))                  # Quadratwurzel elementweise
##          [,1]     [,2]     [,3]     [,4]
## [1,] 3.872983 3.316625 3.605551 4.358899
## [2,] 1.732051 4.123106 2.828427 1.732051
## [3,] 3.605551 1.000000 3.741657 2.645751
## [4,] 3.605551 1.732051 2.449490 1.732051
## [5,] 4.358899 3.464102 3.741657 4.472136
```

by()

```
                                                    # by functions
infarkt <- read.csv2("infarkt.csv")

by(infarkt[, 10:12], infarkt$Gruppe, summary)
## infarkt$Gruppe: Infarkt
##       Chol            Trigl             Hbdh
##  Min.   :185.0   Min.   : 70.00   Min.   :129.0
##  1st Qu.:200.0   1st Qu.: 99.75   1st Qu.:253.5
##  Median :220.0   Median :125.00   Median :295.0
##  Mean   :263.0   Mean   :127.53   Mean   :352.5
##  3rd Qu.:286.2   3rd Qu.:141.50   3rd Qu.:457.0
##  Max.   :800.0   Max.   :203.00   Max.   :737.0
## ------------------------------------------------------------
## infarkt$Gruppe: Kontrolle
##       Chol            Trigl            Hbdh
##  Min.   : 85.0   Min.   : 65.0   Min.   : 41.00
##  1st Qu.:157.5   1st Qu.: 99.5   1st Qu.: 84.75
##  Median :180.0   Median :123.5   Median :105.50
##  Mean   :176.5   Mean   :130.1   Mean   :116.03
##  3rd Qu.:200.0   3rd Qu.:158.8   3rd Qu.:145.25
##  Max.   :230.0   Max.   :239.0   Max.   :264.00
```

tapply()

```
                                                    # tapply functions
x <- 1:12
g <- factor(sample(1:3, 12, replace=T))
rbind(x,g)
##   [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
## x    1    2    3    4    5    6    7    8    9    10    11    12
## g    3    1    2    3    3    1    1    3    3     2     3     3

round(tapply(x, g, mean), 2)
##    1    2    3
## 5.00 6.50 7.14
```

```
infarkt <- read.csv2("Infarkt.csv")

tapply(infarkt$Chol, infarkt$Gruppe, mean)
##   Infarkt Kontrolle
##     263.0     176.5
```

sapply()

```
                                                  # sapply functions
l <- list(groesse=c(160,170,177,182), gewicht=c(70,75,80,90,65,78))

sapply(l, mean)
##   groesse   gewicht
## 172.25000  76.33333
```

replicate()

```
normal <- replicate(8, rnorm(5))
round(normal, 3)
##        [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]
## [1,] -0.070 -0.774 -0.125 -0.447 -1.200  0.207  0.065  0.277
## [2,] -0.237 -0.925  0.867  0.930  0.326 -1.322 -0.342 -0.416
## [3,]  0.954  0.625 -1.001 -0.520  1.296 -0.619 -1.407  0.637
## [4,] -0.630 -2.510  0.558  0.299 -1.873 -1.178 -0.076  1.128
## [5,] -0.962  1.360  1.518 -0.251  0.590 -0.848 -0.736 -1.057
```

9.7 Mathematische und statistische Funktionen

```
vect <- c(1.42, 4.84, -2.55, - 1.24)
vect
## [1]  1.42  4.84 -2.55 -1.24

abs(vect)
## [1] 1.42 4.84 2.55 1.24

round(vect, digits=1)
## [1]  1.4  4.8 -2.5 -1.2

ceiling(vect)
## [1]  2  5 -2 -1

floor(vect)
## [1]  1  4 -3 -2

trunc(vect)
## [1]  1  4 -2 -1

round(vect, digits=1)
## [1]  1.4  4.8 -2.5 -1.2

max(vect)
## [1] 4.84

min(vect)
## [1] -2.55

exp(5)
## [1] 148.4132

round(sin(seq(0, 2*pi, by=(pi/4))), digits=3)
## [1]  0.000  0.707  1.000  0.707  0.000 -0.707 -1.000 -0.707  0.000

sqrt(7)
## [1] 2.645751
```

Kontingenztafeln

```
infarkt <- read.csv2("infarkt.csv")

table(infarkt$Gruppe, infarkt$Geschlecht)
##
##              m  w
##   Infarkt   28 12
##   Kontrolle 25 15

tab <- as.data.frame(table(infarkt$Gruppe, infarkt$Geschlecht))
```

```
names(tab) <- c("Gruppe","Geschlecht","H?ufigkeit")
tab
```

Gruppe	Geschlecht	H?ufigkeit
Infarkt	m	28
Kontrolle	m	25
Infarkt	w	12
Kontrolle	w	15

Statistische Maßzahlen:

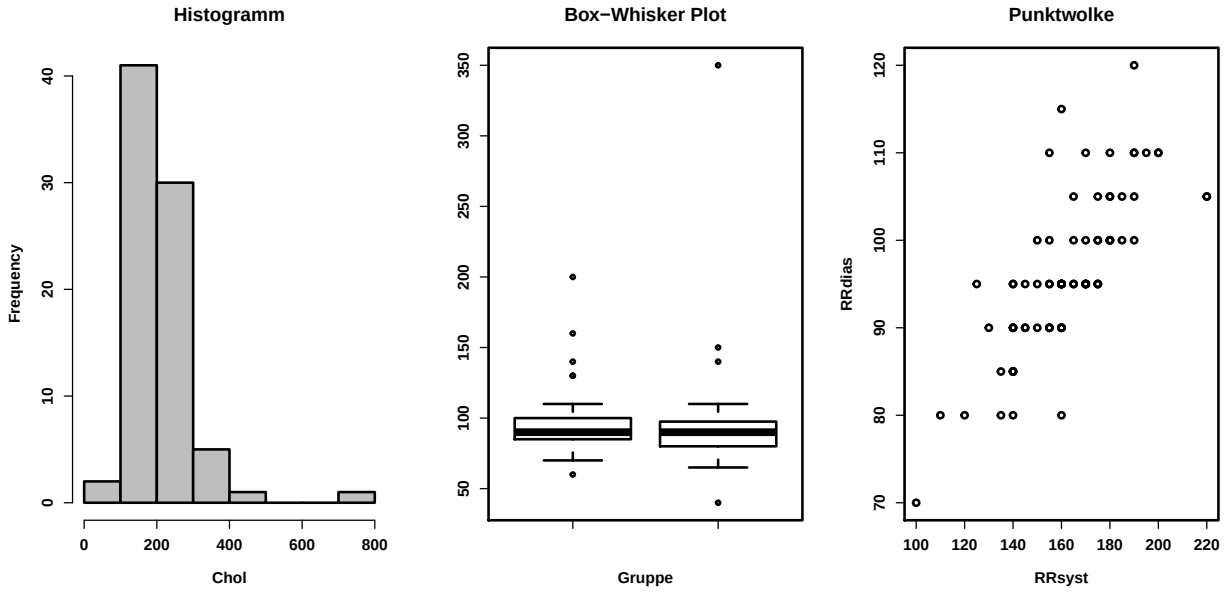
```
infarkt <- read.csv2("infarkt.CSV")
attach(infarkt)

mean(Alter)
## [1] 58.4875

sd(Alter)
## [1] 10.70549

max(Blutz)
## [1] 350

quantile(RRsyst, prob=c(0.10,0.25,0.50,0.75,0.90))
##    10%    25%    50%    75%    90%
## 140.00 148.75 160.00 175.00 190.00

summary(Chol)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    85.0   180.0   200.0   219.8   222.5   800.0
```

aggregate()

```
infarkt <- read.csv2("infarkt.csv")
attach(infarkt)

stat <- aggregate(Chol, by=data.frame(Geschlecht), summary, na.rm=T)

print(stat)
##   Geschlecht   x.Min. x.1st Qu. x.Median   x.Mean x.3rd Qu.   x.Max.
## 1          m  85.0000  190.0000 205.0000 220.8491  230.0000 800.0000
## 2          w 140.0000  177.5000 190.0000 217.5926  210.0000 485.0000

stat[[2]][,3]                                  # Teilergebnisse: Medianwerte
## [1] 205 190
```

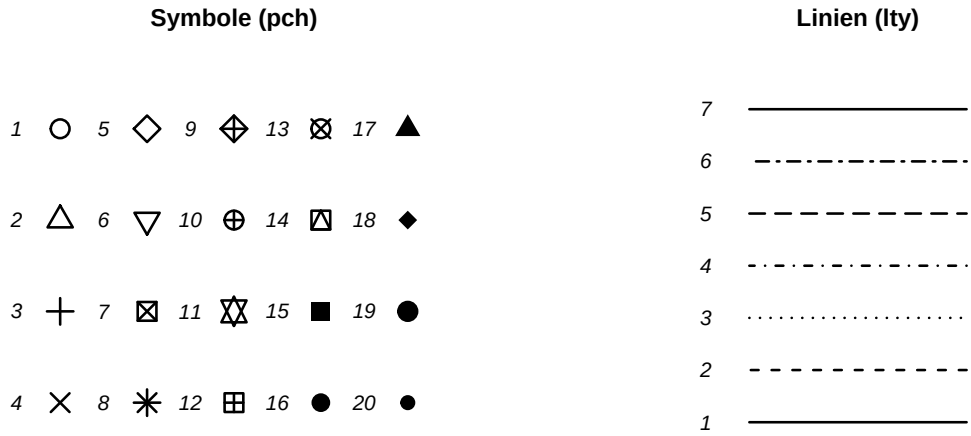
9.8 Graphikfunktionen

```
infarkt <- read.csv2("infarkt.CSV")
attach(infarkt)

par(mfrow=c(1,3), ps=14, font=2, font.axis=2, font.lab=2,
    font.main=2, font.sub=2, lwd=2)

hist(Chol, main="Histogramm", col="grey")

boxplot(Blutz[Gruppe=="Infarkt"], Blutz[Gruppe=="Kontrolle"],
            xlab="Gruppe", main="Box-Whisker Plot")

plot(RRsyst, RRdias, main="Punktwolke")
```

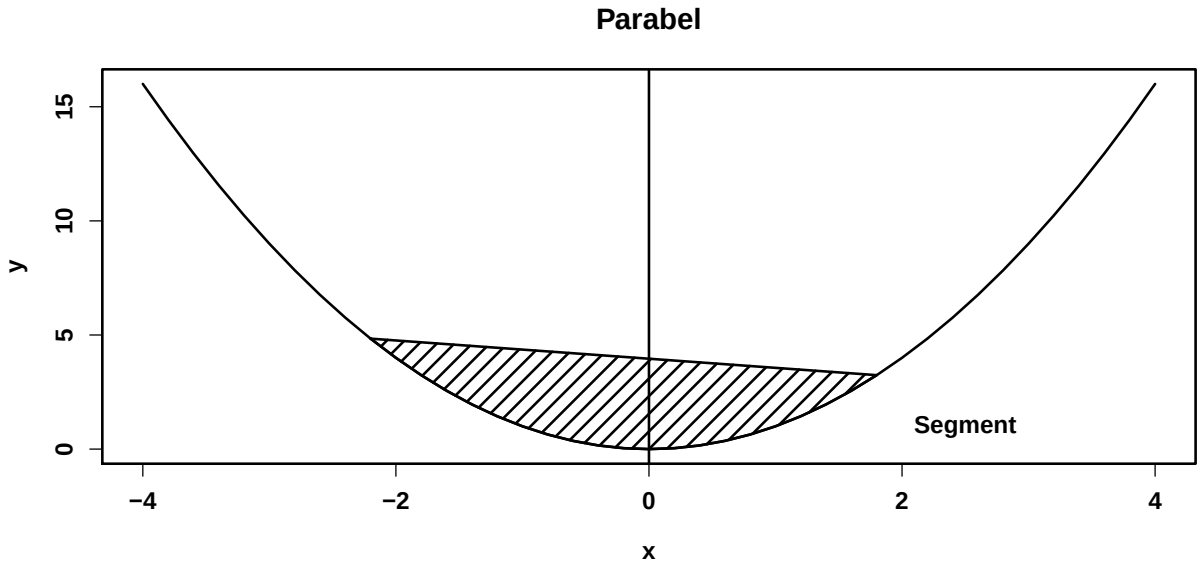



```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="l", ps=12)

Pex <- 2.0
ipch <- 1:20
k <- floor(sqrt(20))
dd <- c(-1, 1)/2
rx <- dd + range(ix <- (ipch - 1)%/%k)
ry <- dd + range(iy <- 3 + (k - 1) - (ipch - 1)%%k)
pch <- as.list(ipch)
plot(rx, ry, type = "n", axes = FALSE, xlab = "", ylab = "",
     main = "Symbole (pch)")
for (i in 1:20) {
    pc <- pch[[i]]
    points(ix[i], iy[i], pch = pc, col = "black",
           bg = "yellow", cex = Pex)
    text(ix[i] - 0.5, iy[i], pc, col = "black", font=3, cex = 1.0)
}
```

```
Pex <- 3
ity <- 7
dd <- c(-1, 1)/2
rx <- dd + range(ix <- rep(c(1,2), 7))
ry <- dd + range(iy <- seq(1, 7, by=1))
plot(rx, ry, type = "n", axes = FALSE, xlab = "", ylab = "",
     main = "Linien (lty)")
for (i in 1:7) {
    pc <- i
    lines(c(ix[i], ix[i+1]), c(iy[i], iy[i]), lty=i, lwd=2, cex = Pex)
    text(1 - 0.2, iy[i], pc, col = "black", font=3, cex = 1.0)
}
```




```
par(mfrow=c(1,1),ps=14, font=2, font.axis=2, font.lab=2,
                                  font.main=2, font.sub=2, lwd=2)

x <- seq(-4, +4, by=0.2)
y <- x^2
plot(x, y, type="l")
abline(v=0)
polygon(x[10:30], y[10:30], density=10)
text(2.5,1, "Segment")
title("Parabel")
```


Parabel
y
0
5
10
15
−4
−2
0
2
4
x
Segment