```
## alles (!) im Workspace löschen
rm(list=ls(all=T))

######################
## allgemeine Plots ##
######################

## Scatterplot
##############

x <- c(1, 2, 3, 4, 6)
y <- c(4, 6, 5, 7, 6)
# einzelne Punkte plotten
plot(x, y)

## Parameter
############

# erzeugt eine 2x2-Matrix von Bildern in einem Fenster (Bilder werden zeilenweise erzeugt)
par(mfrow = c(2,2))

# Beschriftung der x- und y-Achse
plot(x, y, xlab = "x-Achse", ylab = "y-Achse")

# Bereich der x- und y-Achse von Hand vorgeben
plot(x, y, type = "p", xlim = c(1, 10), ylim = c(0,8))

# farbiger Plot / Linienstil / Linienstärke / Titel
plot(x, y, col = "red", type = "l", lty = 2, lwd = 3, main = "Mein Plot")

# Untertitel
plot(x, y, type = "b", lty = 3, sub = "Unterschrift")

## Balkendiagramme
##################

# Standard-Balkendiagramm
barplot(y)

# horizontales Balkendiagramm
barplot(y, horiz = TRUE)
# es gibt viele Parameter, siehe ?barplot

## plots abspeichern
####################

# Plot erzeugen
barplot(y, horiz = TRUE)

# Verzeichnis angeben
dev.print(postscript, "C://bars.ps")

## einfache Plots (öffnen kein eigenes Grafik-Fenster)
```

```
54  #################
55
56  # zuerst normalen Plot erzeugen
57  plot(x, y)
58
59  # zusätzliche Geraden einzeichnen
60  abline(3, 0.75, col = "red", lwd = 3)
61
62  # horizontale Geraden (vgl. auch abline(v = 3))
63  abline(h = 5, col = "blue", lwd = 3)
64
65  # Titel angeben
66  title("Ueberschrift")
67
68  # Liniensegment einzeichnen
69  segments(2, 5.5, 4, 6.5, col = "green", lwd = 3)
70
71  # Legende angeben (Position, Inhalt)
72  legend(x = 4, y = 4.5, legend = c("m", "w"), lty = c(1, 2), col = c("brown", "black"),
    lwd = c(3, 1))
73
74  ########################
75  ## Funktionen plotten ##
76  ########################
77
78  ## Beispiel: Verteilungsfunktion der N(2,9)-Verteilung
79  ######################################################
80
81  # function(x) sagt, dass das was danach kommt (ohne Komma dazwischen) eine Funktion in
    x ist
82  plot(function(x) pnorm(x, mean = 2, sd = 3), -8, 12)
83
84  # zusätzlich die Dichte plotten mit add = TRUE (funktioniert auch bei plot(function(x)
    ...))
85  curve(dnorm(x, mean = 2, sd = 3), -8, 12, col = "red", add = TRUE)
86
87  ## eigene Funktionen plotten (Achtung: Funktion muss Vektoren verarbeiten können!)
88  ##############################
89
90  # Gauss-Klammer (findet die größte ganze Zahl, die kleiner gleich x ist)
91  gauss <- function(x) {
92      k <- 0
93      while (k <= x) {
94          k <- k + 1
95      }
96      return (k - 1)
97  }
98
99  # Fehler! Die Funktion gauss(x) kann keine Vektoren verarbeiten.
100 plot(function(x) gauss(x), 0, 5.5)
101
102 # Gauss-Klammer die mit Vektoren zurechtkommt
103 gauss2 <- function(x) {
104     # Rückgabewert soll ein Vektor mit der gleichen Länge wie x sein
```

```
105       k <- rep(0, length(x))
106       # Schleife über alle Elemente von x
107       for (i in 1:length(x)) {
108           while (k[i] <= x[i]) {
109               # überall den Index i hinzufügen
110               k[i] <- k[i] + 1
111           }
112       }
113       # den Vektor k - 1 zurückgeben
114       return (k - 1)
115   }
116
117   plot(function(x) gauss2(x), 0, 5.5) # klappt
118
119   ## Dreidimensionale Plots
120   ##########################
121
122   # Vektor der x-Werte
123   x <- seq(0, 1, 0.01)
124
125   # Vektor der y-Werte
126   y <- seq(0, 1, 0.01)
127
128   # einfache Funktion zur Berechnung der z-Werte
129   f <- function(x, y) {
130       return (x^2 * y)
131   }
132
133   z <- outer(x, y, f)
134   # Schema zur Berechnung von z:
135   # x/y |   1    |   2    |   3
136   #   1 | f(1,1) | f(1,2) | f(1,3)
137   #   2 | f(2,1) | f(2,2) | f(2,3)
138
139   # perspektivischer Plot, Blickwinkel als Parameter
140   # theta = Azimut (Drehung in x/y-Ebene), phi = Kobreite = 90° - Breite (Kippen nach
      oben))
141   persp(x, y, z, theta = 30, phi = 15)
142
143   # Konturlinien, Anzahl / Dichte der Linien als Parameter
144   contour(x, y, z, nlevels = 15)
145
146   # Bild, d.h. z wird als Höhe interpretiert
147   image(x, y, z)
148
149   # Farbschema als Parameter angeben
150   par(mfrow=c(2,4))
151   image(x, y, z, col=heat.colors(5))
152   image(x, y, z, col=topo.colors(5))
153   image(x, y, z, col=terrain.colors(5))
154   image(x, y, z, col=rainbow(5))
155   image(x, y, z, col=heat.colors(5000))
156   image(x, y, z, col=topo.colors(5000))
157   image(x, y, z, col=terrain.colors(5000))
```

```
158  image(x, y, z, col=rainbow(5000))
159
160  ########################
161  ## Statistische Plots ##
162  ########################
163
164  ## Histogramme
165  ##############
166
167  # Erzeugen von 1000 standardnormalverteilten Pseudozufallszahlen
168  x <- rnorm(1000, 0, 1)
169
170  # freq = FALSE plottet relative Häufigkeiten
171  hist(x, freq = FALSE)
172
173  ## Exkurs: Pakete installieren und laden
174  ########################################
175
176  # Im Menü: Pakete -> Installiere Paket(e) ... -> Mirror wählen -> Paket wählen
177  # Paket laden: library(...)
178
179  ## Histogramm plotten (2)
180  #########################
181
182  # Paket MASS laden
183  library(MASS)
184
185  # Histogramm mit relativen Häufigkeiten, Breite der Balken = 0.2
186  truehist(x, h = 0.2)
187
188  ## Boxplot
189  ##########
190
191  # erzeuge drei Gruppen von Werten
192  x1 <- c(rnorm(100, 0, 1), 4)
193  x2 <- c(rnorm(100, 2, 1), -1)
194  x3 <- c(rnorm(100, 1, 1), 5)
195
196  # Dataframe erzeugen
197  x <- data.frame(x1, x2, x3)
198
199  # Boxplot
200  boxplot(x)
201
202  # kein Plot, nur Statistiken ausgeben
203  boxplot(x, plot = FALSE)
204
205  ## QQ-Plot
206  ##########
207
208  # Stichprobe x1 vs. Standardnormalverteilung
209  qqnorm(x1)
210
211  # zusätzliche Linie durch 1. und 3. Quartil zeichnen
```

```
qqline(x1)

z1 <- rexp(100, 2)
z2 <- rgamma(100, 4, 5)

# QQ-Plot mit zwei Stichproben
qqplot(z1, z2)

#########################
## Schiefe und Wölbung ##
#########################

# Schiefe (Skewness): E((X - EX)^3) / (Var(X))^(3/2)
# Wölbung (Kurtosis): E((X - EX)^4) / (Var(X))^(4/2)
# Normalverteilung hat Schiefe 0 und Wölbung 3
# Schiefe > 0: rechtsschief / linkssteil
# Schiefe < 0: linksschief / rechtssteil
# Wölbung > 0: steilgipflig
# Wölbung < 0: flachgipflig

# Paket moments laden
library(moments)

# Schiefe und Wölbung berechnen
x <- rnorm(100, 0, 1)
skewness(x)
kurtosis(x)

x <- rt(100, 2)
skewness(x)
kurtosis(x)

# t-Verteilung nähert sich für wachsende Freiheitsgrade der Standardnormalverteilung an
x <- rt(100, 20)
skewness(x)
kurtosis(x)

```