AS17: Kapitel 5 - Zufallsvariablen, Verteilungen

Jürgen Hedderich

2020-04-24

Contents

Aktuelle R-Umgebung zu den Beispielen **2**

5.1 Zufallsvariable **3**

5.2 Maßzahlen zur Kennzeichnung einer Verteilung **5**

5.3 Diskrete Verteilungen **11**

5.3.2 Gleichverteilung 11

5.3.3 Binomialverteilung 12

5.3.4 Multinomialverteilung 17

5.3.5 Poisson-Verteilung 18

5.3.6 Negative Binomialverteilung 22

5.3.7 Geometrische Verteilung 24

5.3.8 Hypergeometrische Verteilung 25

5.3.9 Negative Hypergeometrische Verteilung 26

5.4 Stetige Verteilungen **27**

5.4.1 Stetige Gleichverteilung 27

5.4.1 Standard-Beta-Verteilung 28

5.4.3 Normalverteilung 31

5.4.4 Halbnormalverteilung 39

5.4.5 Gestutzte Normalverteilung 41

5.4.6 Lognormalverteilung: 43

5.4.7 Exponentialverteilung 46

5.4.8 Weibull-Verteilung 47

5.4.9 Extremwertverteilung Typ I (Gumbel) 50

5.4.10 Gammaverteilung 55

5.5 Testverteilungen **58**

5.5.1 Student-Verteilung (t-Verteilung) 58

5.5.2 Chiquadrat-Verteilung 61

5.5.3 Fischer-Verteilung (F-Verteilung) 64

5.6 Verteilung zweidimensionaler Zufallsvariablen **68**

5.6.2 Randverteilungen und Unabhängigkeit 68

Korrelationskoeffizient 69

Zweidimensionale Normalverteilung 70

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!

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

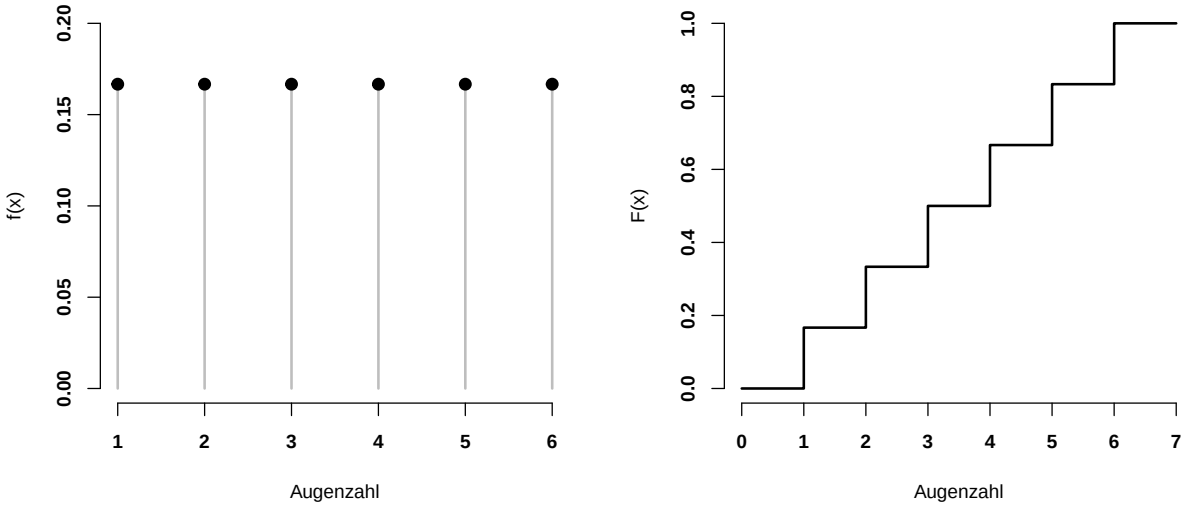
The R Project for Statistical Computing

```
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
```

5.1 Zufallsvariable

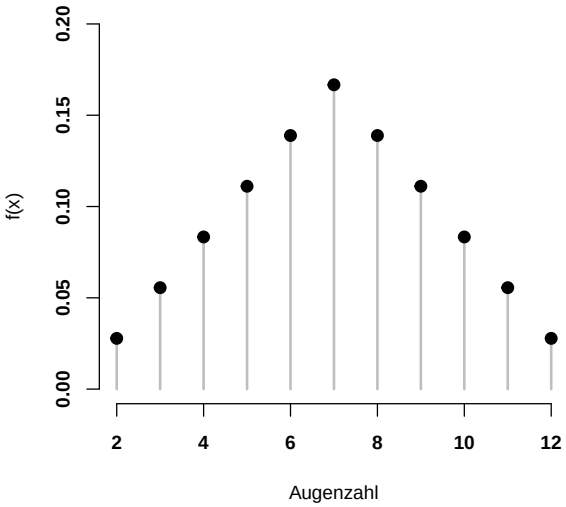
Augenzahl beim Werfen eines Würfels:

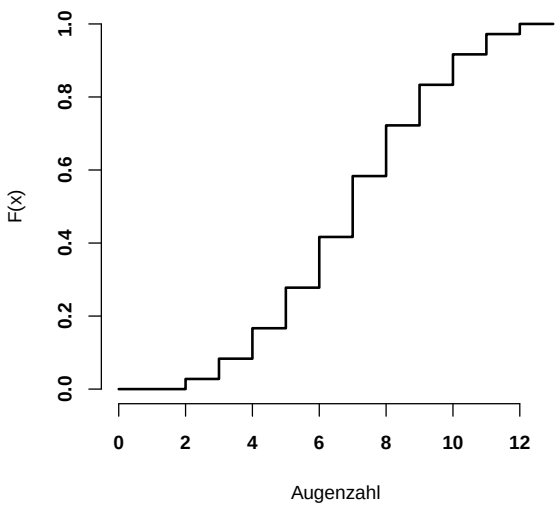
```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
x <- seq(1, 6, by=1); fx <- rep(1/6, 6)
plot(x, fx, type="h", ylim=c(0, 0.2), ylab="f(x)",
                              xlab="Augenzahl", col="grey")
points(x, fx, pch=19, cex=1.1, col="black")
F <- cumsum(fx)
x1 <- c(0, x, 7)
Fx <- c(0, F, 1)
plot(x1, Fx, type="s", ylim=c(0, 1), ylab="F(x)", xlab="Augenzahl")
```



Augenzahl beim Werfen zweier Würfel:

```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
x <- seq(2, 12, by=1)
fx <- c(1/36, 2/36, 3/36, 4/36, 5/36, 6/36, 5/36, 4/36, 3/36, 2/36, 1/36)
plot(x, fx, type="h", ylim=c(0, 0.2), ylab="f(x)",
                                        xlab="Augenzahl", col="grey")
points(x, fx, pch=19, cex=1.1, col="black")
F <- cumsum(fx)
x1 <- c(0, x, 13)
Fx <- c(0, F, 1)
plot(x1, Fx, type="s", ylim=c(0,   1), ylab="F(x)", xlab="Augenzahl")
```


f(x)
0.00
0.05
0.10
0.15
0.20
2
4
6
8
10
12
Augenzahl


F(x)
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
12
Augenzahl

5.2 Maßzahlen zur Kennzeichnung einer Verteilung

5.2.3.1 Momente: Schiefe und Exzess

Funktionen skewness() und kurtosis() in library(e1071)

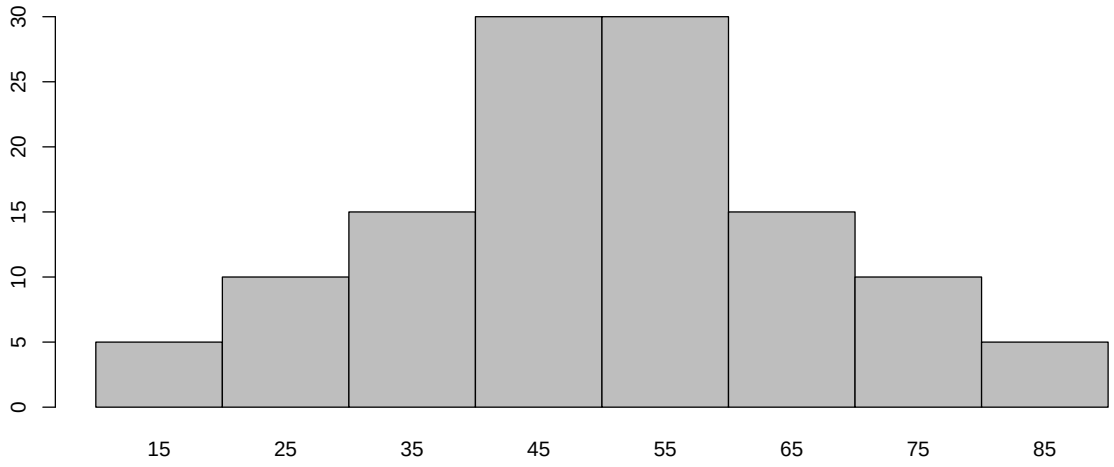
```
x <- c(2, 3, 4, 4, 4, 5, 5, 5, 5, 6, 8, 10, 20, 40)
m <- mean(x); s <- sd(x); n <- length(x)

(1/n)*sum((x-m)^3)/(s^3)                                    # Schiefe
## [1] 2.198071

(1/n)*sum((x-m)^4)/(s^4) - 3                                # Wölbung
## [1] 3.89879

library(e1071)
skewness(x, type = 3)                                       # Schiefe
## [1] 2.198071

kurtosis(x, type = 3)                                       # Wölbung
## [1] 3.89879
```

5.2.3.2 Potenzmomente aus klassierten Häufigkeiten

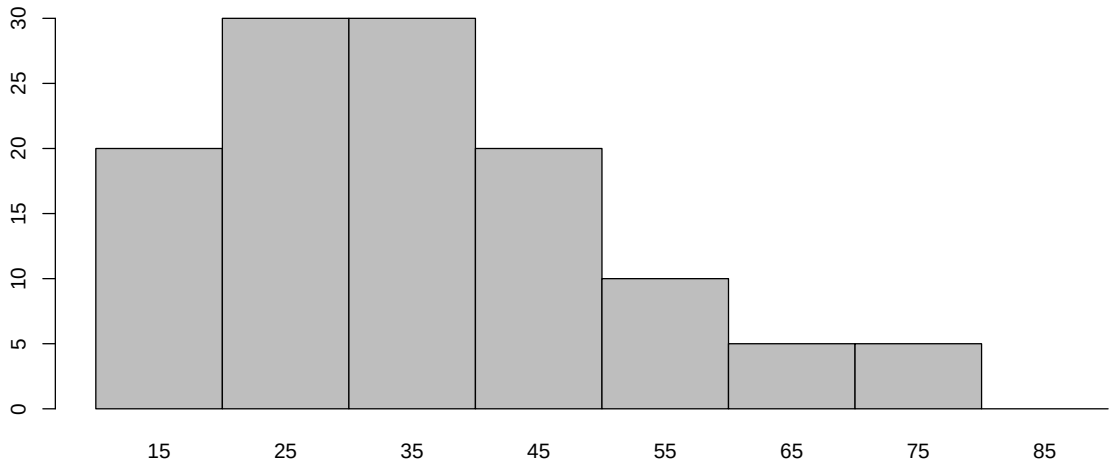
```
momente <- function(x, f, d, b) {                           # x Daten (Vektor)
                                                            # f Klassenhäufigkeiten
                                                            # d Ursprung (Modalwert, Referenz)
                                                            # b Klassenbreite
    n <- sum(f);     z <- (x - d) / b
    m1 <- sum(f * z) / n;      m2 <- sum(f * z^2) / n
    m3 <- sum(f * z^3) / n;    m4 <- sum(f * z^4) / n
    s2 <- b^2 * (m2 - m1^2);   s3 <- (sqrt(s2))^3;     s4 <- s2^2
    mitt <- d + (b * m1)
    vari <- b^2 * (m2 - m1^2)
    schi <- (b^3 * (m3 - 3*m1*m2 + 2*m1^3)) / s3
    woel <- ((b^4 * (m4 - 4*m1*m3 + 6*m1^2*m2 - 3*m1^4)) / s4) - 3
    cat("\n","Momente aus klassierten Beobachtungen:","\n",
        "Mittelwert:",mitt,"\n",        "   Varianz:",vari,"\n",
        "   Schiefe:",schi,"\n",        "   Wölbung:",woel,"\n")
}

x <- c(8.8, 9.3, 9.8, 10.3, 10.8, 11.3, 11.8)
f <- c(  4,   8,  11,    7,    5,    3,    2)
d <- 9.8;   b <- 0.5

momente(x, f, d, b)
##
##  Momente aus klassierten Beobachtungen:
##  Mittelwert: 10.025
##     Varianz: 0.636875
##     Schiefe: 0.4598458
##     Wölbung: -0.4809175
```

Unterschiedliche Verteilungsformen:

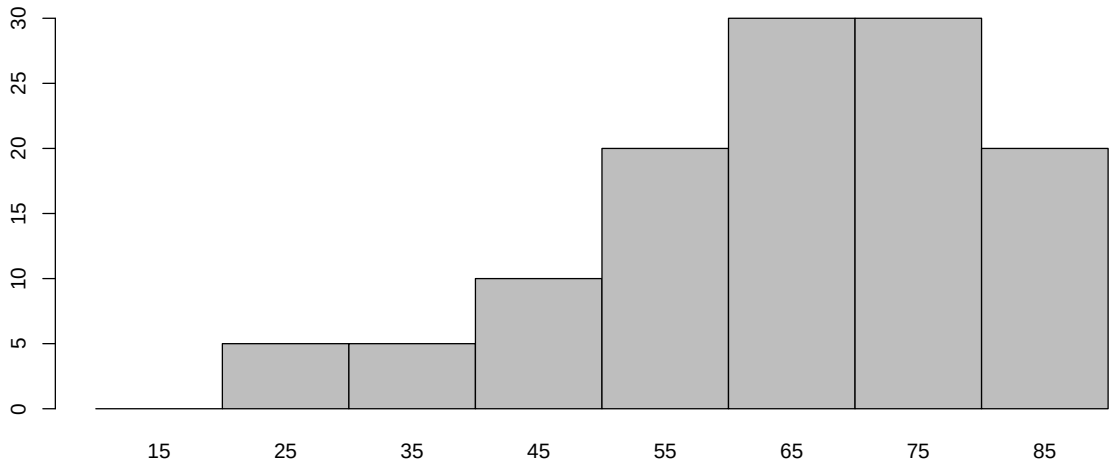
```
b      <- 10                                          # Klassenbreite
d      <- 30                                          # Ursprung
x      <- c(15, 25, 35, 45, 55, 65, 75, 85)           # Klassenmitten

f1     <- c( 5, 10, 15, 30, 30, 15, 10,  5)           # Häufigkeiten
momente(x, f1, d, b); barplot(f1, width=b, space=0, names.arg=x)
##
##  Momente aus klassierten Beobachtungen:
##  Mittelwert: 50
##     Varianz: 275
##     Schiefe: 0
##     Wölbung: -0.3140496
```




```
f2     <- c(20, 30, 30, 20, 10,  5,  5,  0)           # Häufigkeiten
momente(x, f2, d, b); barplot(f2, width=b, space=0, names.arg=x)
##
##  Momente aus klassierten Beobachtungen:
##  Mittelwert: 35.41667
##     Varianz: 245.6597
##     Schiefe: 0.7101991
##     Wölbung: -0.02114747
```

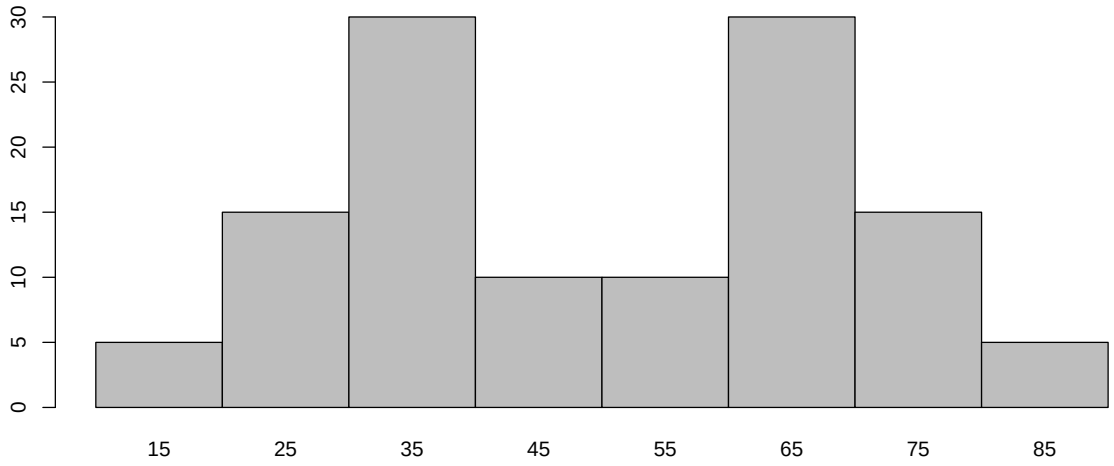



```
f3     <- c( 0,  5,  5, 10, 20, 30, 30, 20)              # Häufigkeiten
momente(x, f3, d, b); barplot(f3, width=b, space=0, names.arg=x)
##
##  Momente aus klassierten Beobachtungen:
##  Mittelwert: 64.58333
##     Varianz: 245.6597
##     Schiefe: -0.7101991
##     Wölbung: -0.02114747
```



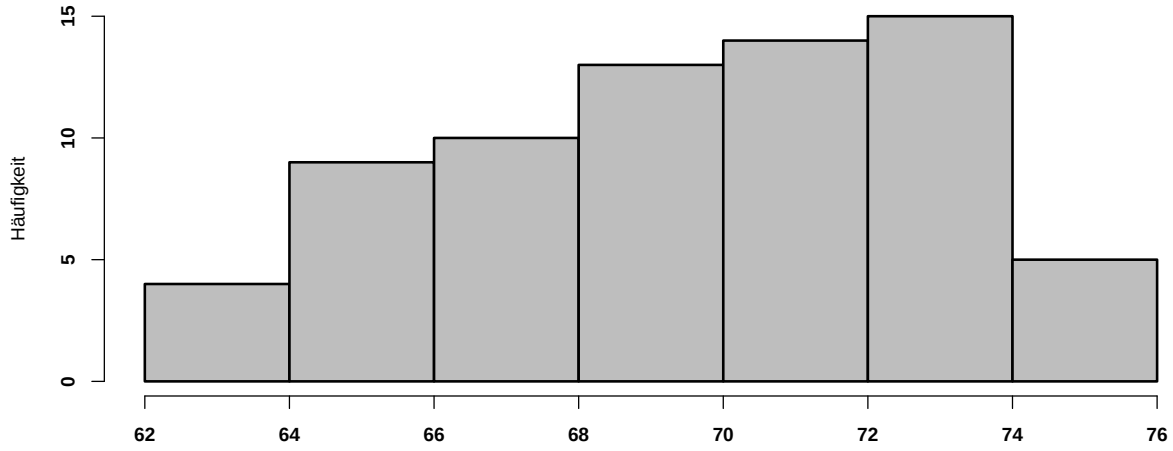

```
f4     <- c( 5, 15, 30, 10, 10, 30, 15,  5)              # Häufigkeiten
momente(x, f4, d, b); barplot(f4, width=b, space=0, names.arg=x)
```

```
##
##  Momente aus klassierten Beobachtungen:
##  Mittelwert: 50
##     Varianz: 375
##     Schiefe: 0
##     Wölbung: -1.235556
```



5.2.3.3 Quantilsmaße zu Schiefe und Exzess

```
                                                    # Körpergröße von 70 Studenten
y <- c(63, 63, 64, 64, rep(65,4), rep(66,5), rep(67,4),
       rep(68,6), rep(69,5), rep(70,8), rep(71,7), rep(72,7),
       rep(73,10), rep(74,5), rep(75,3), rep(76,2))

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
hist(y, breaks=c(seq(62, 76, by=2)), freq=TRUE, col="grey",
     ylim=c(0, 15), xlim=c(62, 76), xlab=" ", main=" ", ylab="Häufigkeit")
```




```
mean(y)                                             # Mittelwert
## [1] 70.04286
var(y)                                              # empirische Varianz
## [1] 11.11408
skewness(y)                                         # empirisches 3tes Moment
## [1] -0.2843902
kurtosis(y)                                         # empirisches 4tes Moment
## [1] -0.8728042

                                                    # Quartile
Q    <- quantile(y, probs=seq(0, 1, 0.25), names=TRUE, type=7); Q
##   0%  25%  50%  75% 100%
##   63   68   70   73   76

Q1   <- as.numeric(Q[2]); Q2 <- as.numeric(Q[3]); Q3 <- as.numeric(Q[4])
skew <- (Q3 + Q1 - 2*Q2)/(Q3-Q1);  skew
## [1] 0.2
                                                    # Oktile
A    <- quantile(y, probs=seq(0, 1, 0.125), names=TRUE, type=7); A
##    0% 12.5%   25% 37.5%   50% 62.5%   75% 87.5%  100%
##    63    66    68    69    70    72    73    74    76
```

```
A7 <- as.numeric(A[8]); A6 <- as.numeric(A[7]); A5 <- as.numeric(A[6])
A3 <- as.numeric(A[4]); A2 <- as.numeric(A[3]); A1 <- as.numeric(A[2])
kurt <- ((A7 - A5) + (A3 - A1))/(A6-A2); kurt
## [1] 1
```

Tukey's Fünfer-Regel:

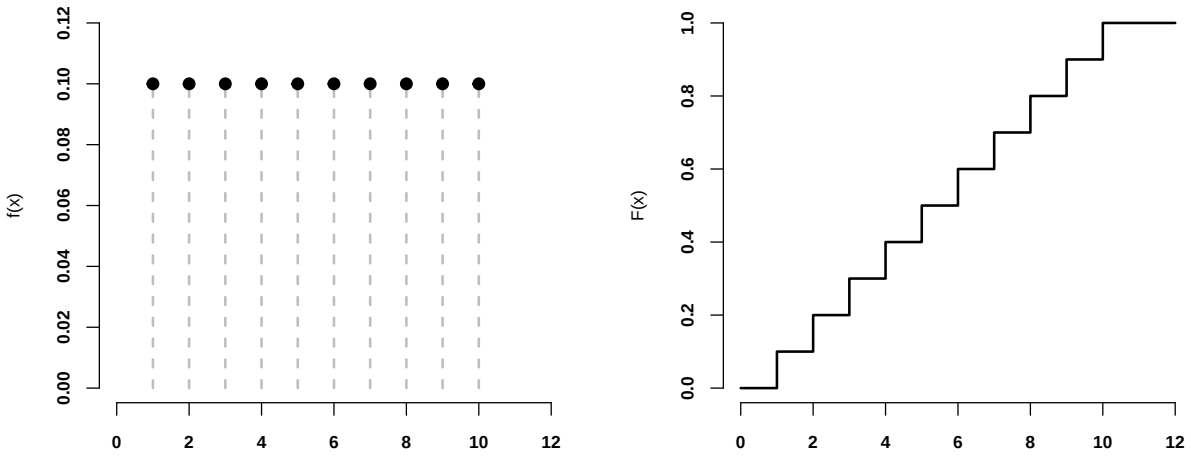
```
                                                    # Körpergröße von 70 Studenten
y <- c(63, 63, 64, 64, rep(65,4), rep(66,5), rep(67,4),
       rep(68,6), rep(69,5), rep(70,8), rep(71,7), rep(72,7),
       rep(73,10), rep(74,5), rep(75,3), rep(76,2))
fivenum(y)                                          # Tukeys five numbers
## [1] 63 68 70 73 76
```

5.3 Diskrete Verteilungen

5.3.2 Gleichverteilung

Funktionen zu diskreten Verteilungen in library(e1071)

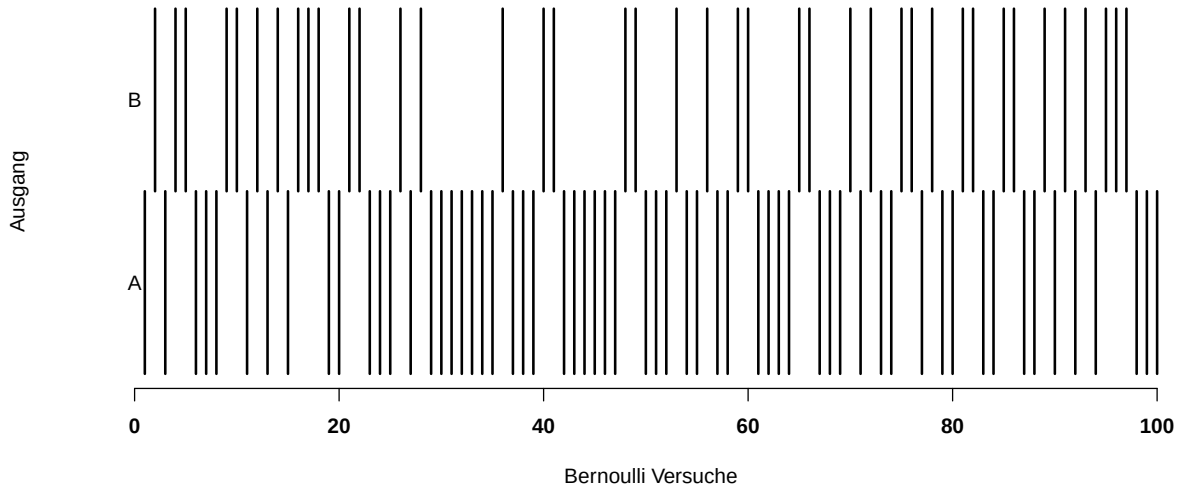
```
library(e1071)
ddiscrete(1:10, rep(0.1, 10))                          # Dichtefunktion
##  [1] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
pdiscrete(1:10, rep(0.1, 10))                          # Verteilungsfunktion
##  [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
qdiscrete(c(0.25, 0.5, 0.75), rep(0.1, 10))            # Quantilfunktion (Quartile)
## [1] 3 5 8
rdiscrete(20, 1:10)                                    # Zufallszahlen
##  [1]  8  8  6  8  8  1  6  5  8  6 10  6  9 10  8  9  9  2  5  9

x  <- 1:10

fx <- ddiscrete(1:10, rep(0.1, 10))                    # Wahrscheinlichkeitsfunktion

Fx <- pdiscrete(1:10, rep(0.1, 10))                    # Verteilungsfunktion
x1 <- c(0, x, 11, 12)
Fx <- c(0, Fx, 1, 1)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=10)
plot(x, fx, type="h", ylim=c(0, 0.12), xlim=c(0,12), ylab="f(x)",
            xlab=" ", lty=2, col="grey")
points(x, fx, pch=19, cex=1.1, col="black")
plot(x1, Fx, type="s", ylim=c(0,1), xlim=c(0,12), ylab="F(x)", xlab=" ")
```



5.3.3 Binomialverteilung

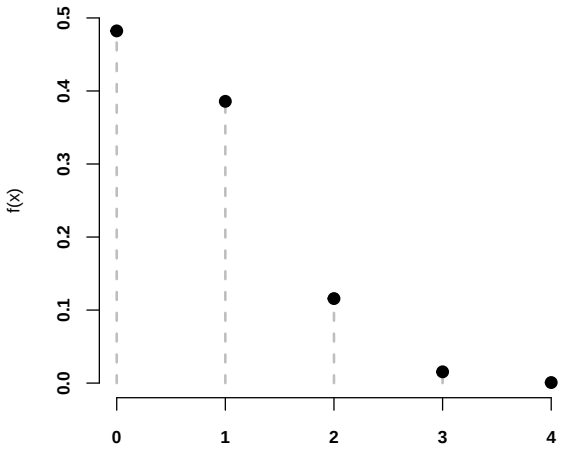
Bernoulli Versuch:

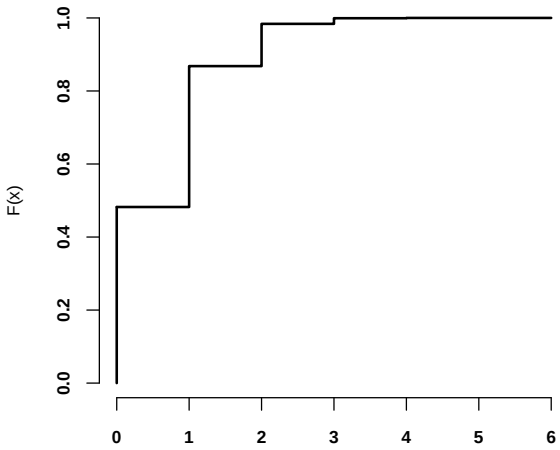
```
n <- 100
x <- sample(c(-1,1), n, replace=T, prob=c(0.6,0.4))

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=12)
plot(x, type="h", axes=FALSE, xlab="Bernoulli Versuche", ylab="Ausgang")
axis(1, at = NULL, labels = TRUE, tick = TRUE)
text(0, -0.5, "A");  text(0, +0.5, "B")
```



Binomialverteilung:

```
n  <- 4; p  <- 1/6;  x  <- 0:n

fx <- dbinom(x, n, p)                                    # Wahrscheinlichkeitsfunktion

Fx <- pbinom(x, n, p); x1 <- c(x, n+1, n+2)              # Verteilungsfunktion
Fx <- c(Fx, 1, 1)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=10)
plot(x, fx, type="h", ylim=c(0, 0.5), xlim=c(0,n), ylab="f(x)",
            xlab=" ", lty=2, col="grey")
points(x, fx, pch=19, cex=1.1, col="black")
plot(x1, Fx, type="s", ylim=c(0,1), xlim=c(0,n+2), ylab="F(x)", xlab=" ")
lines(c(0,0),c(0,Fx[1]))
```






```r
dbinom(3, 3, 1/2)
## [1] 0.125
dbinom(2, 3, 1/2)
## [1] 0.375
```

Tabelle zu Binomialwahrscheinlichkeiten:

```r
prob <- c(0.01, 0.05, 0.10, 0.20, 0.25, 0.30, 0.40, 0.50)
tab  <- matrix(data = NA, nrow = 18, ncol = 8, byrow = FALSE, dimnames = NULL)
row  <- 1
for (n in 2:5) {
  for (i in 0:n) {
    tab[row, ] <- round(dbinom(i, n, prob), 4)
    row <- row +1  }  }
c1 <- c(rep(2,3), rep(3,4), rep(4,5), rep(5,6))
c2 <- c(0:2, 0:3, 0:4, 0:5)
tab <- cbind(c1, c2, tab)
colnames(tab) <- c("n", "x", "p=0.01", "p=0.05", "p=0.10",  "p=0.20", "p=0.25"... )
as.data.frame(tab)
```

n	x	p=0.01	p=0.05	p=0.10	p=0.20	p=0.25	p=0.30	p=0.40	p=0.50
2	0	0.9801	0.9025	0.8100	0.6400	0.5625	0.4900	0.3600	0.2500
2	1	0.0198	0.0950	0.1800	0.3200	0.3750	0.4200	0.4800	0.5000
2	2	0.0001	0.0025	0.0100	0.0400	0.0625	0.0900	0.1600	0.2500
3	0	0.9703	0.8574	0.7290	0.5120	0.4219	0.3430	0.2160	0.1250
3	1	0.0294	0.1354	0.2430	0.3840	0.4219	0.4410	0.4320	0.3750
3	2	0.0003	0.0071	0.0270	0.0960	0.1406	0.1890	0.2880	0.3750
3	3	0.0000	0.0001	0.0010	0.0080	0.0156	0.0270	0.0640	0.1250
4	0	0.9606	0.8145	0.6561	0.4096	0.3164	0.2401	0.1296	0.0625
4	1	0.0388	0.1715	0.2916	0.4096	0.4219	0.4116	0.3456	0.2500
4	2	0.0006	0.0135	0.0486	0.1536	0.2109	0.2646	0.3456	0.3750
4	3	0.0000	0.0005	0.0036	0.0256	0.0469	0.0756	0.1536	0.2500

n	x	p=0.01	p=0.05	p=0.10	p=0.20	p=0.25	p=0.30	p=0.40	p=0.50
4	4	0.0000	0.0000	0.0001	0.0016	0.0039	0.0081	0.0256	0.0625
5	0	0.9510	0.7738	0.5905	0.3277	0.2373	0.1681	0.0778	0.0312
5	1	0.0480	0.2036	0.3280	0.4096	0.3955	0.3601	0.2592	0.1562
5	2	0.0010	0.0214	0.0729	0.2048	0.2637	0.3087	0.3456	0.3125
5	3	0.0000	0.0011	0.0081	0.0512	0.0879	0.1323	0.2304	0.3125
5	4	0.0000	0.0000	0.0005	0.0064	0.0146	0.0284	0.0768	0.1562
5	5	0.0000	0.0000	0.0000	0.0003	0.0010	0.0024	0.0102	0.0312

Beispiele zur Binomialverteilung:

```
dbinom(0, 4, 0.2)                                          # Ausschussware
## [1] 0.4096
dbinom(1, 4, 0.2)
## [1] 0.4096
dbinom(2, 4, 0.2)
## [1] 0.1536

dbinom(0:4, 4, 0.2)
## [1] 0.4096 0.4096 0.1536 0.0256 0.0016
pbinom(2, 4, 0.2)
## [1] 0.9728

1 - pbinom(0, 6, 1/6, lower.tail=TRUE)                     # Chevalier de Mere
## [1] 0.665102
pbinom(1, 12, 1/6, lower.tail=FALSE)
## [1] 0.6186674

pbinom(18, 120, 1/6)                                       # Würfel
## [1] 0.3657008

round(200*dbinom(0:4, 4, 0.465),  2)                       # Mäuse
## [1] 16.38 56.96 74.27 43.03  9.35

dbinom(1, 2, 0.8)                                          # Behandlungserfolge
## [1] 0.32
dbinom(1, 5, 0.8)
## [1] 0.0064
dbinom(5, 5, 0.8)
## [1] 0.32768

dbinom(2, 5, 0.5)                                          # fünf Kinder
## [1] 0.3125
dbinom(5, 5, 0.5)
## [1] 0.03125
```

5.3.3.3 Approximation durch die Standardnormalverteilung

Tabelle zum Vergleich unterschiedlicher Tranformationen:

```
p <- c(0.05, 0.5); q <- 1-p

n <- c(100,   10)
X <- c(10,     8)

P0 <- 1 - pbinom(X, n, p, lower.tail = TRUE)                # exakt binomial

m <- n*p; s <- sqrt(n*p*(1-p))
Z1 <- (X + 0.5 - m) / s                                     # Normalverteilung
P1 <- 1 - pnorm(Z1)

Z2 <- (asin(sqrt(X/n)) - asin(sqrt(p))) / sqrt(1/(4*n))     # ArcSin-Transf.
P2 <- 1 - pnorm(Z2)

Z3 <- 2*(asin(sqrt((X+1)/(n+1))) - asin(sqrt(p))) * sqrt(n+0.5)  # Freeman
P3 <- 1 - pnorm(Z3)

Z4 <- (sqrt(4*q*(X+1)) - sqrt(4*p*(n-X)))                   # Mosteller-Tukey
P4 <- 1 - pnorm(Z4)

Z5 <- (sqrt(q*(4*X+3)) - sqrt(p*(4*n-4*X-1)))               # Freeman-Tukey
P5 <- 1 - pnorm(Z5)

tab <- round(cbind(P0, P1, P2, P3, P4, P5), 4)
c1  <- c(0.05, 0.50); c2  <- c(100, 10); c3 <- c(10, 8)
tab <- cbind(c1, c2, c3, tab)
colnames(tab) <- c("p", "n", "X", "exakt ", "normal",  "arcsin", "Freeman", "...")
as.data.frame(tab)
```

p	n	X	exakt	normal	arcsin	Freeman	Mosteller/Tukey	Freeman/Tukey
0.05	100	10	0.0115	0.0058	0.0271	0.0132	0.0131	0.0156
0.50	10	8	0.0107	0.0134	0.0209	0.0127	0.0125	0.0104

Tabelle zu Winkeltransformationen:

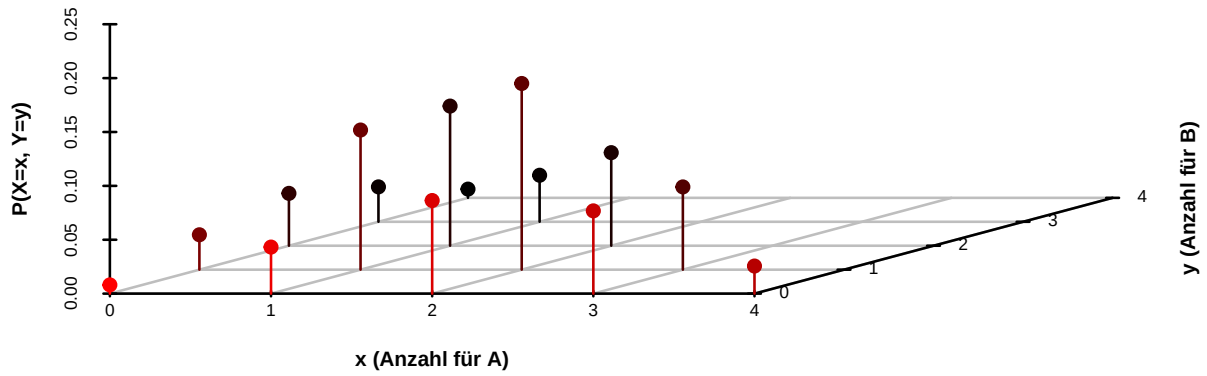
```
p.1 <- seq(0.0, 0.9, by=0.1)
p.2 <- seq(0, 0.09, by=0.01)
p   <- matrix(rep(NA, 100), nrow=10, ncol=10)
for (i in 1:10) {for (k in 1:10) {p[i,k] <- p.1[i] + p.2[k]}}
z   <- round(asin(sqrt(p))*(180/pi), 3)
tab <- cbind(p.1, z)
colnames(tab) <- c("p", seq(0.00, 0.09, 0.01))
as.data.frame(tab)
```

p	0	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.000	5.739	8.130	9.974	11.537	12.921	14.179	15.342	16.430	17.458
0.1	18.435	19.370	20.268	21.134	21.973	22.786	23.578	24.350	25.104	25.842
0.2	26.565	27.275	27.972	28.658	29.334	30.000	30.657	31.306	31.948	32.583
0.3	33.211	33.833	34.450	35.062	35.669	36.271	36.870	37.465	38.057	38.645
0.4	39.232	39.815	40.397	40.976	41.554	42.130	42.706	43.280	43.854	44.427
0.5	45.000	45.573	46.146	46.720	47.294	47.870	48.446	49.024	49.603	50.185
0.6	50.768	51.355	51.943	52.535	53.130	53.729	54.331	54.938	55.550	56.167
0.7	56.789	57.417	58.052	58.694	59.343	60.000	60.666	61.342	62.028	62.725
0.8	63.435	64.158	64.896	65.650	66.422	67.214	68.027	68.866	69.732	70.630
0.9	71.565	72.542	73.570	74.658	75.821	77.079	78.463	80.026	81.870	84.261

5.3.4 Multinomialverteilung

Funktion scatterplot3d() in library(scatterplot3d)

```
n <- 4; p=c(0.4, 0.3, 0.3)
M <- matrix(c(0,0,4, 0,1,3, 0,2,2, 0,3,1, 0,4,0, 1,0,3, 1,1,2, 1,2,1, 1,3,0,
              2,0,2, 2,1,1, 2,2,0, 3,0,1, 3,1,0, 4,0,0),
              byrow=FALSE, nrow=3);
P <- rep(NA, 15); for (i in 1:15) P[i] <- dmultinom(M[,i], prob=p)
round(P, 3)
##  [1] 0.008 0.032 0.049 0.032 0.008 0.043 0.130 0.130 0.043 0.086 0.173 0.086
## [13] 0.077 0.077 0.026

library(scatterplot3d)
par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=12)
z <- P; x <- M[1,]; y <- M[2,]
scatterplot3d(x, y, z, col.axis="black", type="h", highlight.3d=TRUE,
      box=FALSE, col.grid="grey", pch=20, cex.symbols=2,
      xlab="x (Anzahl für A)", ylab="y (Anzahl für B)", zlab="P(X=x, Y=y)")
```



Beispiele zur Multinomialverteilung:

```
dmultinom(c(3,2,1), prob=c(0.50, 0.30, 0.20))          # Beispiel Perlen
## [1] 0.135

dmultinom(c(1,1,1,3,3,3), prob=rep(1/6, 6))            # Beispiel Würfel
## [1] 0.001018751

dmultinom(c(8,1,1), prob=c(1/3, 1/3, 1/3))             # Kandidatenwahl
## [1] 0.001524158
dmultinom(c(3,3,4), prob=c(1/3, 1/3, 1/3))
## [1] 0.07112737
```

5.3.5 Poisson-Verteilung

```
x  <- 0:12
f1 <- dpois(x, 1); f2 <- dpois(x, 2); f3 <- dpois(x, 6)

par(mfrow=c(1,3), lwd=2, font.axis=2, bty="n", ps=14)
plot(x, f1, type="h", ylim=c(0, 0.4), xlim=c(0,12),
     ylab="f(x)", xlab=" ", lty=2, col="grey", lwd=1)
points(x, f1, pch=19, cex=1.2, col="black")
text(6, 0.18, expression(lambda == 1), cex=1.5)
plot(x, f2, type="h", ylim=c(0, 0.3), xlim=c(0,12),
     ylab="f(x)", xlab=" ", lty=2, col="grey", lwd=1)
points(x, f2, pch=19, cex=1.2, col="black")
text(6, 0.28, expression(lambda == 2), cex=1.5)
plot(x, f3, type="h", ylim=c(0, 0.2), xlim=c(0,12),
     ylab="f(x)", xlab=" ", lty=2, col="grey", lwd=1)
points(x, f3, pch=19, cex=1.2, col="black")
text(6, 0.18, expression(lambda == 6), cex=1.5)
```

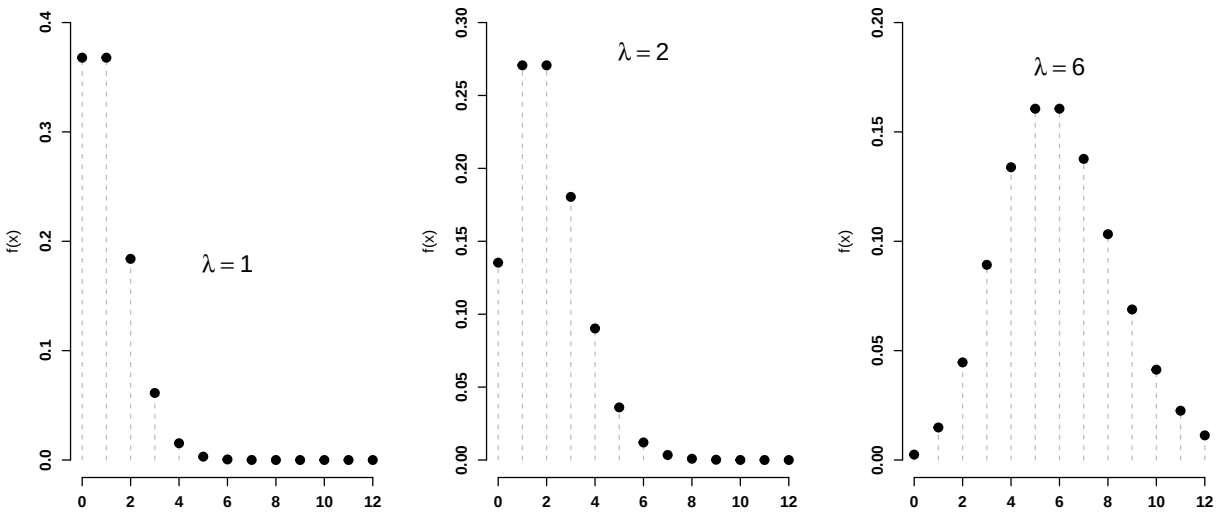


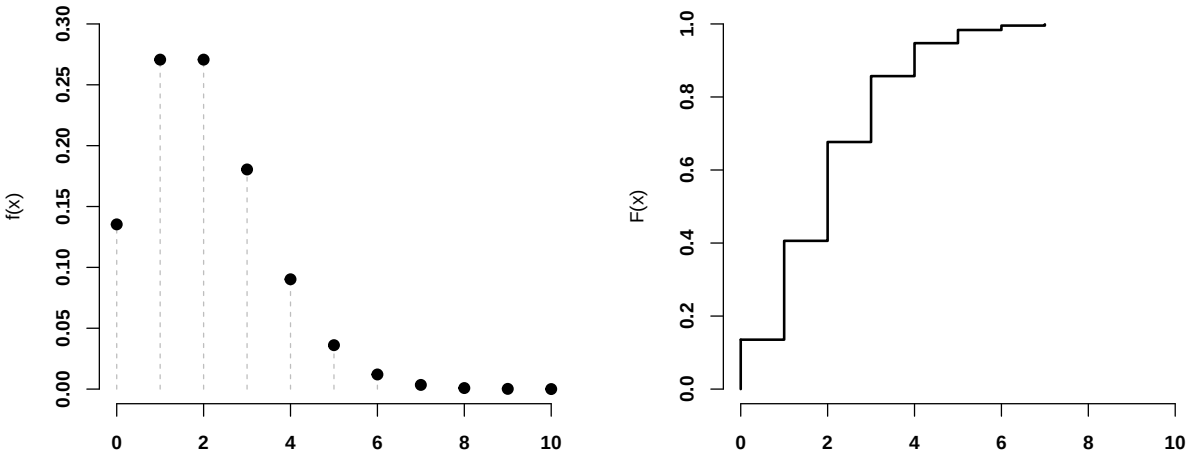
Tabelle zur Poissonverteilung:

```
lambda <- c(0.2, 0.5, 0.8, 1, 3, 5, 8, 12, 20)
tab <- matrix(data = NA, nrow = 30, ncol = 9,
              byrow = FALSE, dimnames = NULL)

for (i in 0:29) {tab[i+1, ] <- round(dpois(i, lambda), 4) }

tab <- cbind(0:29, tab)
colnames(tab) <- c("x","lambda=0.2", "lambda=0.5", "lambda=0.8",
        "lambda=1", "lambda=3", "lambda=5", "lambda=8", "lambda=12", "lambda=20")
as.data.frame(tab[,1:5])
```

x	lambda=0.2	lambda=0.5	lambda=0.8	lambda=1
0	0.8187	0.6065	0.4493	0.3679
1	0.1637	0.3033	0.3595	0.3679
2	0.0164	0.0758	0.1438	0.1839
3	0.0011	0.0126	0.0383	0.0613
4	0.0001	0.0016	0.0077	0.0153
5	0.0000	0.0002	0.0012	0.0031
6	0.0000	0.0000	0.0002	0.0005
7	0.0000	0.0000	0.0000	0.0001
8	0.0000	0.0000	0.0000	0.0000
9	0.0000	0.0000	0.0000	0.0000
10	0.0000	0.0000	0.0000	0.0000
11	0.0000	0.0000	0.0000	0.0000
12	0.0000	0.0000	0.0000	0.0000
13	0.0000	0.0000	0.0000	0.0000
14	0.0000	0.0000	0.0000	0.0000
15	0.0000	0.0000	0.0000	0.0000
16	0.0000	0.0000	0.0000	0.0000
17	0.0000	0.0000	0.0000	0.0000
18	0.0000	0.0000	0.0000	0.0000
19	0.0000	0.0000	0.0000	0.0000
20	0.0000	0.0000	0.0000	0.0000
21	0.0000	0.0000	0.0000	0.0000
22	0.0000	0.0000	0.0000	0.0000
23	0.0000	0.0000	0.0000	0.0000
24	0.0000	0.0000	0.0000	0.0000
25	0.0000	0.0000	0.0000	0.0000
26	0.0000	0.0000	0.0000	0.0000
27	0.0000	0.0000	0.0000	0.0000
28	0.0000	0.0000	0.0000	0.0000
29	0.0000	0.0000	0.0000	0.0000

Beispiele zur Poissonverteilung:

```r
dpois(0:3, 2.7397)                                          # Geburtstagsproblem
## [1] 0.06458972 0.17695646 0.24240380 0.22137123


dpois(3, 2)                                                 # Serumproben
## [1] 0.180447
1-ppois(2, 2,)
## [1] 0.3233236

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(0:10, dpois(0:10, 2), type="h", ylim=c(0, 0.3), xlim=c(0,10),
     ylab="f(x)", xlab=" ", lty=2, col="grey", lwd=1)
points(0:10, dpois(0:10, 2), pch=19, cex=1.0, col="black")
plot(0:7, ppois(0:7, 2), type="s", ylim=c(0,1),
    xlim=c(0,10), ylab="F(x)", xlab=" ")
lines(c(0,0), c(0,dpois(0, 2)))
```



5.3.5.1 Dispersionsindex

```
DI_test <- function(xi, fi) {
  if (length(xi)!=length(fi)) stop("x unf f ungleiche Längen!!!")
  n    <- sum(fi);    mw   <- sum(xi*fi)/n
  var  <- (sum(xi^2*fi) - (sum(xi*fi))^2/n) / (n-1)
  DI   <- mw/var
  stat <- sum(fi*(xi-mw)^2)/mw
  pvalue <- pchisq(stat, df=n-1, lower.tail=F)
  cat("\n","Mittelwert:",round(mw, 4),"\n","   Varianz:",round(var,4),
      "\n","        DI:",round(DI, 4),
      "\n","  Statstik:",stat,"(P-Wert:", pvalue,")","\n")     }

                                                    # Tod durch Pferdehufschlag
x <- c(  0,  1,  2, 3, 4, 5)
f <- c(109, 65, 22, 3, 1, 0)
DI_test(x, f)
##
##  Mittelwert: 0.61
##     Varianz: 0.611
##          DI: 0.9984
##    Statstik: 199.3115 (P-Wert: 0.4804495 )

lambda <- 0.61
n      <- 200
round(dpois(0:5, lambda) * n, 1)
## [1] 108.7  66.3  20.2   4.1   0.6   0.1
```

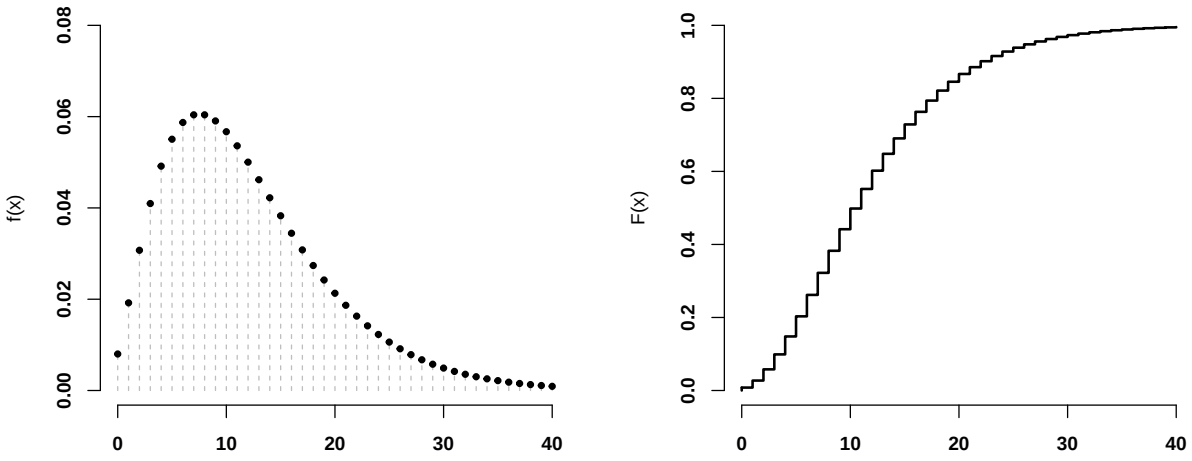
5.3.5.2 Approximation durch die Standardnormalverteilung

```
k <- c(3, 4); lambda <- c(9, 10)                              # Approximationen
z1 <- (k  - lambda) / sqrt(lambda); z
##  [1] 0.0081 0.0324 0.0486 0.0324 0.0081 0.0432 0.1296 0.1296 0.0432 0.0864
## [11] 0.1728 0.0864 0.0768 0.0768 0.0256
ppois(k, lambda)
## [1] 0.02122649 0.02925269
pnorm(z1)
## [1] 0.02275013 0.02888979
```

5.3.6 Negative Binomialverteilung

```
choose(7+3-1, 7) * 0.2^3 * 0.8^7
## [1] 0.06039798
dnbinom(7, 3, 0.2)
## [1] 0.06039798

p <- rep(NA, 8)
for (i in 0:7) p[i+1] <- choose(i+3-1, i) * 0.2^3 * 0.8^i; sum(p)
## [1] 0.3222005
pnbinom(7, 3, 0.2)
## [1] 0.3222005
```

```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(0:40, dnbinom(0:40, 3, 0.2), type="h", ylim=c(0, 0.08),
    xlim=c(0,40), ylab="f(x)", xlab=" ", lty=2, col="grey", lwd=1)
points(0:40, dnbinom(0:40, 3, 0.2), pch=19, cex=0.5, col="black")
plot(0:40, pnbinom(0:40, 3, 0.2), type="s", ylim=c(0,1),
    xlim=c(0,40), ylab="F(x)", xlab=" ")
lines(c(0,0), c(0,pnbinom(0, 3,0.2)))
```



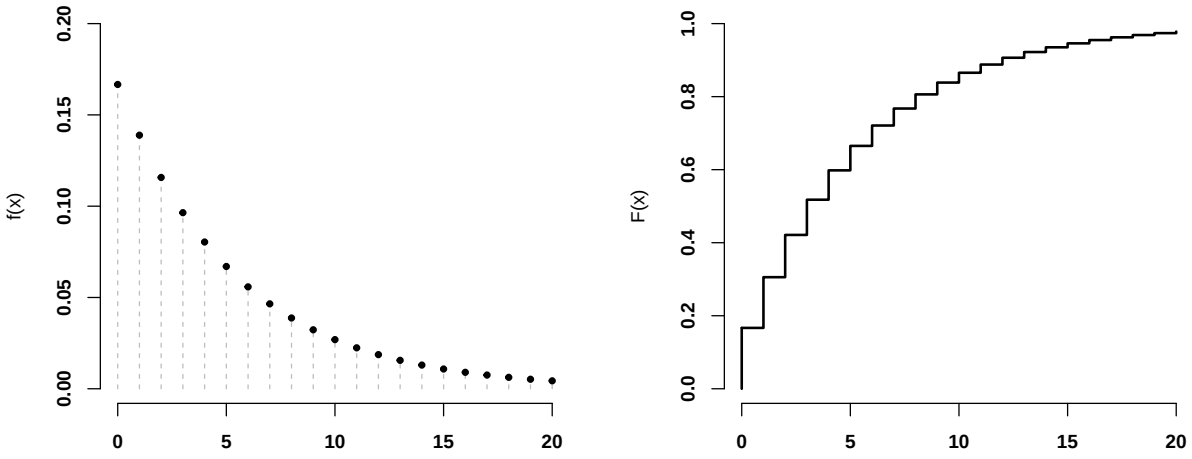
Beispiele zur negativen Binomialverteilung:

```
p <- 0.4; x <- 10; k <- 5                                    # Werfen mit Steinen
choose(x-1, k-1)*p^k*(1-p)^(x-k)
## [1] 0.1003291

dnbinom(x-k, k, 0.4)
## [1] 0.1003291

k   <- c(  0,   1,  2,  3, 4, 5)                             # Unfälle
```

```
obs <- c(447, 132, 42, 21, 3, 2); n <- sum(obs)
m   <- sum(obs*k)/n; round(m, 2)                          # Mittelwert (Erwartungswert)
## [1] 0.47
round(dpois(k, m) * n, 0)                                 # Poisson- Verteilung
## [1] 406 189  44   7   1   0
v   <- sum((obs*(k - m)^2))/(n-1); v                      # (emp.) Varianz
## [1] 0.6919002
p   <- m / v;  r <- m*p/(1-p)                             # Modellparameter
round(dnbinom(k, r, p) * n, 0)                            # negative Binomialverteilung
## [1] 443 139  44  14   5   2

                                                          #  Zecken
beob <- c(rep(0,7), rep(1,9), rep(2,8), rep(3,13), rep(4,8), rep(5,5),
          rep(6,4), rep(7,3), rep(8,0), rep(9,1), 10, 10); n  <- sum(beob)
r.hat <- mean(beob)^2/(var(beob)-mean(beob)); r.hat
## [1] 3.956746
p.hat <- r.hat/(mean(beob)+r.hat); p.hat
## [1] 0.5490336
round(dnbinom(0:11, 3.96, 0.55)*60, 0)
##  [1]  6 10 11 10  8  6  4  2  1  1  1  0

                                                          # Markenartikel
beob <- c(rep(0,39), rep(1,14), rep(2,10), rep(3,6), rep(4,4), rep(5,4),
          rep(6,3), rep(7,3), rep(8,2), rep(9,2), rep(10, 13))
n  <- sum(beob); m  <- mean(beob); m
## [1] 2.91
v  <- var(beob)
r  <- m^2 / (v - m); r
## [1] 0.8536217

m  <- 3.4;  s  <- 0.5;  p  <- s/(s+m)
n=100; x  <- 0:10
round(dnbinom(x, s, p)*n, 0)
##  [1] 36 16 10  7  6  4  4  3  2  2  2
round(dpois(x, m)*n, 0)
##  [1]  3 11 19 22 19 13  7  3  1  1  0
```

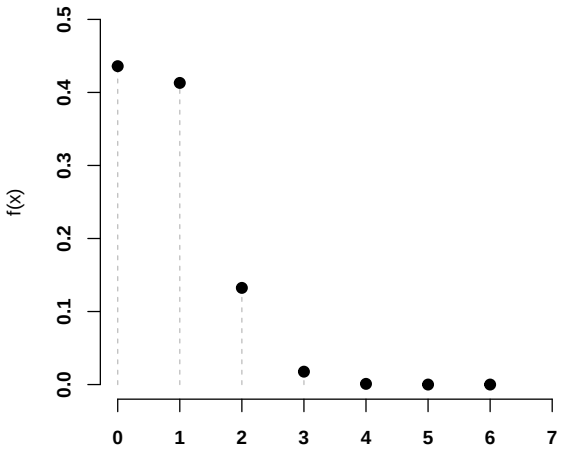
5.3.7 Geometrische Verteilung

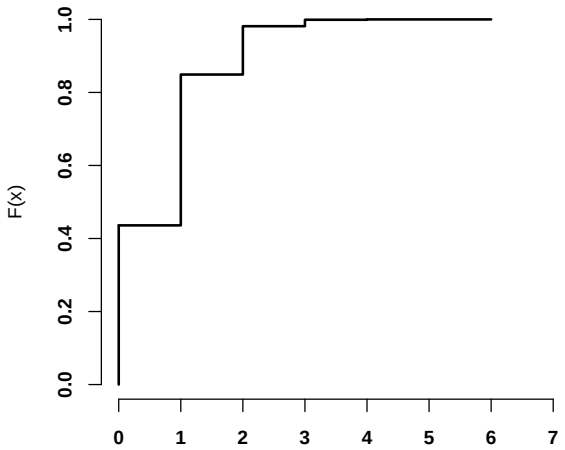
```
p <- 1/6; n <- 0:20
f <- dgeom(n, p)                                    # Wahrscheinlichkeitsfunktion
F <- pgeom(n, p)                                    # Verteilungsfunktion

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(n, f, type="h", ylim=c(0, 0.2),
        xlim=c(0,20), ylab="f(x)", xlab=" ", lty=2, col="grey", lwd=1)
points(n, f, pch=19, cex=0.5, col="black")
plot(n, F, type="s", ylim=c(0,1), xlim=c(0,20), ylab="F(x)", xlab=" ")
lines(c(0,0), c(0,p))
```



5.3.8 Hypergeometrische Verteilung

```
dhyper(2, 5, 10, 5)                                          # Beispiel Urnenmodell
## [1] 0.3996004

dhyper(1:5, 6, 4, 5)                                         # Beispiel Studenten
## [1] 0.02380952 0.23809524 0.47619048 0.23809524 0.02380952
phyper(1:5, 6, 4, 5)
## [1] 0.02380952 0.26190476 0.73809524 0.97619048 1.00000000

dhyper(4, 6, 43, 6)                                          # Beispiel Lotto
## [1] 0.0009686197

x <- 0:6
f <- dhyper(0:6, 6, 43, 6)                                   # Wahrscheinlichkeitsfunktion
F <- phyper(0:6, 6, 43, 6)                                   # Verteilungsfunktion

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, f, type="h", ylim=c(0, 0.5), xlim=c(0,7),
             ylab="f(x)", xlab=" ", lty=2, col="grey", lwd=1)
points(x, f, pch=19, cex=1.0, col="black")
plot(x, F, type="s", ylim=c(0,1), xlim=c(0,7), ylab="F(x)", xlab=" ")
lines(c(0,0), c(0,dhyper(0, 6, 43, 6)))
```






```
dhyper(50, 95, 5, 50)                                        # Beispiel Ausschussware
## [1] 0.02814225
dhyper(49, 95, 5, 50)
## [1] 0.152947

dhyper(0, 10, 42, 15)                                        # Beispiel Annoncen
## [1] 0.02201831
```

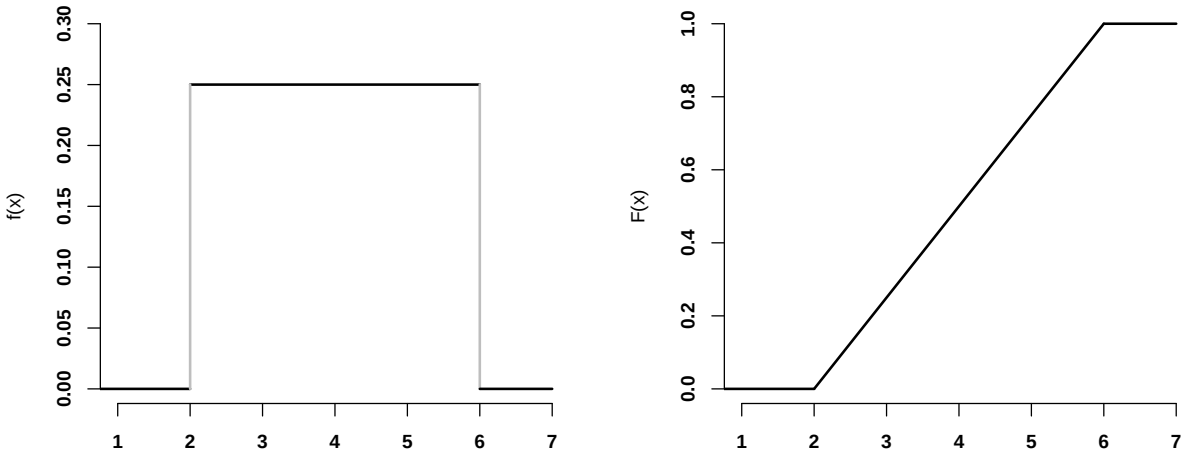
5.3.9 Negative Hypergeometrische Verteilung

```r
nhyper <- function(W, S, k, s) {
  p <- choose(s+k-1, s) * choose(W-k + S-s, W-k) / choose(W+S, W)
  m <- k*S / (W+1)
  v <- k*(S+W+1)*S*(W-k+1) / ((W+1)*(W+1)*(W+2))
  cat("P = ", round(p, 4), "\n","Erwartungswert: ",
                                m,"\n","Varianz.......: ",v,"\n")
}

nhyper(W=2, S=3, k=2, 0:3)
## P =  0.1 0.2 0.3 0.4
##  Erwartungswert:  2
##  Varianz.......:  1

nhyper(W=7, S=10, k=3, 0:10)                                  # Beispiel Urnenmodell
## P =  0.0515 0.1103 0.1527 0.1697 0.162 0.1361 0.1008 0.0648 0.0347 0.0141 0.0034
##  Erwartungswert:  3.75
##  Varianz.......:  4.6875
```

5.4 Stetige Verteilungen

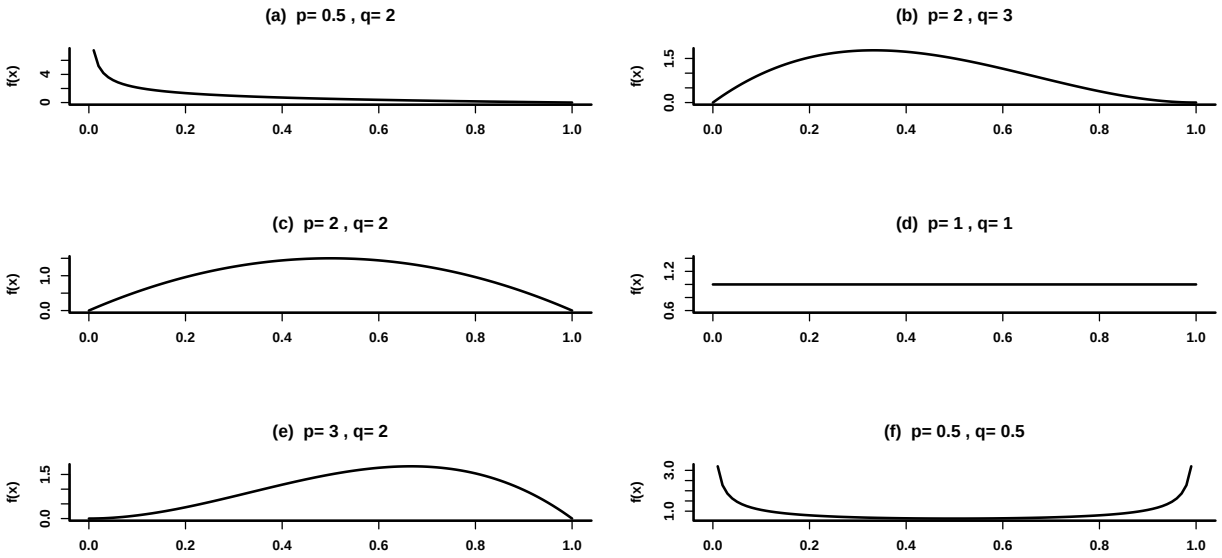
5.4.1 Stetige Gleichverteilung

```
a <- 2; b <- 6
x <- seq(a, b, by=0.01)

f <- dunif(x, a, b)                                    # Wahrscheinlichkeitsdichte
F <- punif(x, a, b)                                    # Verteilungsfunktion

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, f, type="l", ylim=c(0, 0.3), xlim=c(a-1,b+1),
            ylab="f(x)", xlab=" ", lty=1, lwd=2)
lines(c(0,a) , c(0, 0))
lines(c(a,a), c(0, dunif(a, a, b)), col="grey")
lines(c(b,b), c(0, dunif(b, a, b)), col="grey")
lines(c(b, b+1), c(0, 0))
plot(x, F, type="l", ylim=c(0,1), lwd=2,
            xlim=c(a-1,b+1), ylab="F(x)", xlab=" ")
lines(c(0,a), c(0,0))
lines(c(b, b+1), c(1,1))
```



5.4.1 Standard-Beta-Verteilung

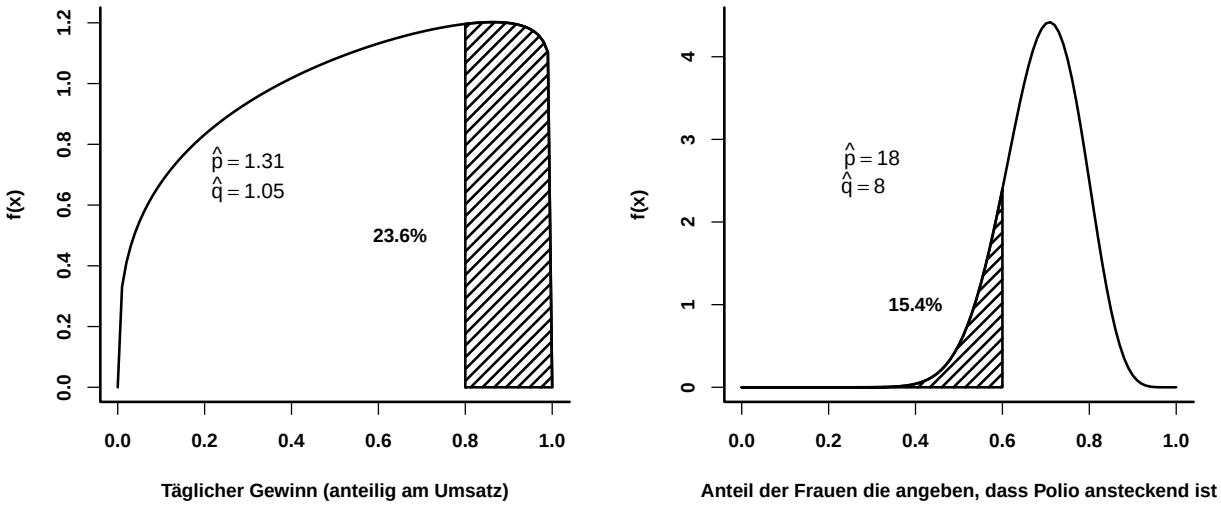
```
x  <- seq(0, 1, by =0.01)

par(mfrow=c(3,2), lwd=2, font=2, font.axis=2, font.lab=2, bty="l", ps=12)

t <- c("(a)","(b)","(c)","(d)","(e)","(f)")
p <- c(0.5, 2, 2, 1, 3, 0.5);    q <- c(  2, 3, 2, 1, 2, 0.5)

for (i in 1:6)  {
  plot(x, dbeta(x, p[i], q[i]), type="l", xlab=" ", ylab="f(x)",
  main=paste(t[i]," p=",p[i],", q=",q[i]))
}
```



Beispiele zur Standard-Beta-Verteilung:

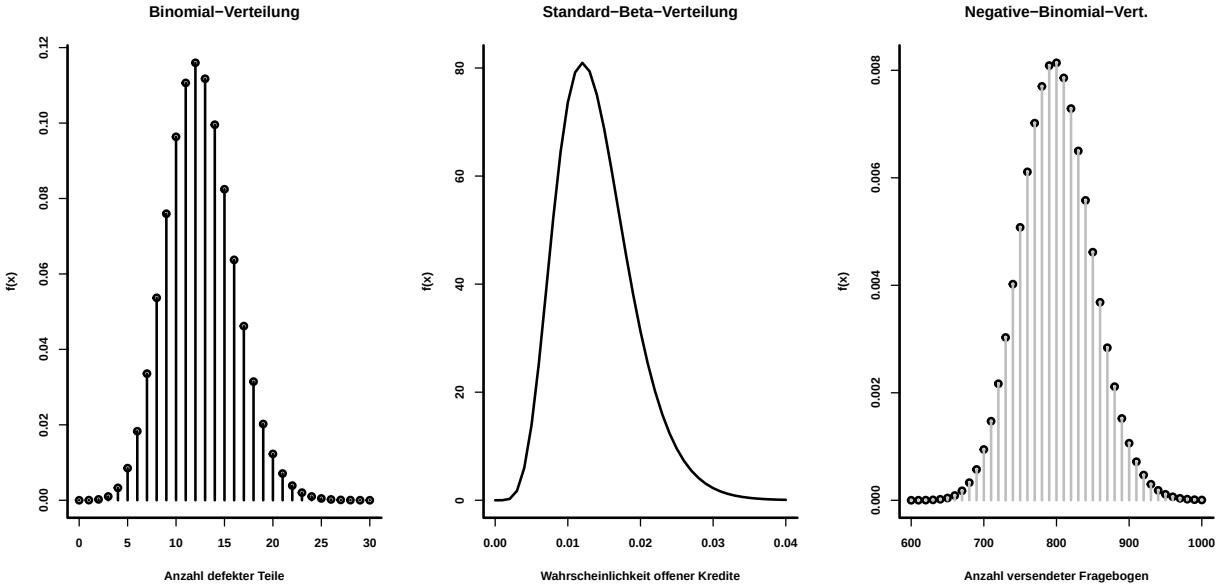
```
                                                    # Gewinn und Umsatz
x <- c(0.06, 0.08, 0.15, 0.32, 0.41, 0.46, 0.47, 0.47, 0.47, 0.51,
       0.59, 0.59, 0.64, 0.69, 0.71, 0.81, 0.89, 0.91, 0.92, 0.96)
m <- mean(x); s <- sd(x)
p1 <- m * ( m*(1-m)/s^2 - 1); p1
## [1] 1.311217
q1 <- (1-m) * ( m*(1-m)/s^2 - 1); q1
## [1] 1.04921
W <- 1-pbeta(0.8, p1, q1); W
## [1] 0.2362548


                                                    # Polio
n <- 24; k <- 17
p2 <- k+1; p2
## [1] 18
q2 <- n - k +1; q2
```

```r
## [1] 8
W <- pbeta(0.6, p2, q2); W
## [1] 0.1535517

x <- seq(0, 1, by=0.01); fbeta1 <- dbeta(x, p1, q1)
x <- seq(0, 1, by=0.01); fbeta2 <- dbeta(x, p2, q2)

par(mfrow=c(1,2), lwd=2, font=2, font.axis=2,
          font.lab=2, bty="l", ps=11)
x <- seq(0, 1, by=0.01)
plot(x, fbeta1, type="l",
        xlab="Täglicher Gewinn (anteilig am Umsatz)", ylab="f(x)")
x1 <- seq(0.8, 1, by=0.01)
p1 <- 1.311217; q1 <- 1.049210
y1 <- dbeta(x1, p1, q1)
x1 <- c(0.8, x1); y1 <- c(0,   y1)
text(0.65, 0.5, "23.6%")
text(0.3,0.75, expression(hat(p) == 1.31),cex=1.1)
text(0.3,0.65, expression(hat(q) == 1.05),cex=1.1)
polygon(x1, y1, density = 15, angle = 45)
plot(x, fbeta2, type="l",
    xlab="Anteil der Frauen die angeben, dass Polio ansteckend ist",
    ylab="f(x)")
x2 <- seq(0, 0.6, by=0.01)
p2 <- 18; q2 <- 8
y2 <- dbeta(x2, p2, q2)
x2 <- c(x2, 0.6); y2 <- c(y2, 0)
text(0.40, 1, "15.4%")
text(0.3,2.80, expression(hat(p) == 18),cex=1.1)
text(0.28,2.45, expression(hat(q) == 8),cex=1.1)
polygon(x2, y2, density = 15, angle = 45)
```



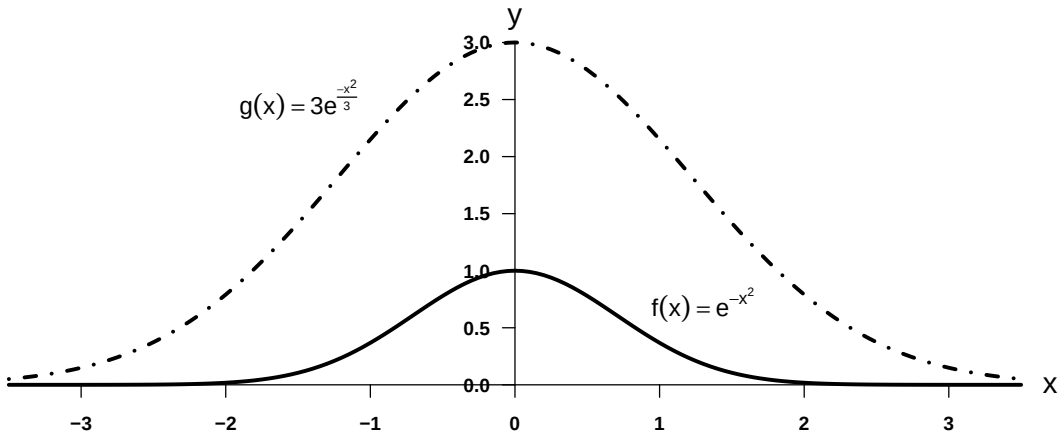
Beispiele zur Standard-Beta-Verteilung:

```r
n <- 250;   p <- 0.05                                          # defekte Maschinenteile
pbinom(15, n, p) - pbinom(10, n, p)
## [1] 0.5203554

k <- 6; p <- k+1                                               # ausstehende Darlehen
n <- 500; q <- n-k+1
pbeta(0.02, p, q) - pbeta(0.01, p, q)
## [1] 0.6350956

pnbinom(850-200, 200, 0.25)                                    # Kundenzufriedenheit
## [1] 0.8485493

par(mfrow=c(1,3), lwd=2, font=2, font.axis=2, font.lab=2, bty="l", ps=11)
n <- 250;   p <- 0.05 ; x <- seq(0, 30, by=1)
fbin <- dbinom(x, n, p)
plot(x, fbin, xlab="Anzahl defekter Teile", ylab="f(x)", main="Binomial-Verteilung")
for (i in 1:n) lines(c(x[i],x[i]),c(0,fbin[i]))

k <- 6; p <- k+1
n <- 500; q <- n-k+1
x <- seq(0, 0.04, by = 0.001)
fbeta <- dbeta(x, p, q)
plot(x, fbeta, type="l", xlab="Wahrscheinlichkeit offener Kredite",
     ylab="f(x)", main="Standard-Beta-Verteilung")

x     <- seq(600, 1000, by = 10); n <- length(x)
fnbin <- dnbinom(x-200, 200, 0.25)
plot(x, fnbin, xlab="Anzahl versendeter Fragebogen", ylab="f(x)",
     main="Negative-Binomial-Vert.")
for (i in 1:n) lines(c(x[i],x[i]),c(0,fnbin[i]),col="grey")
```



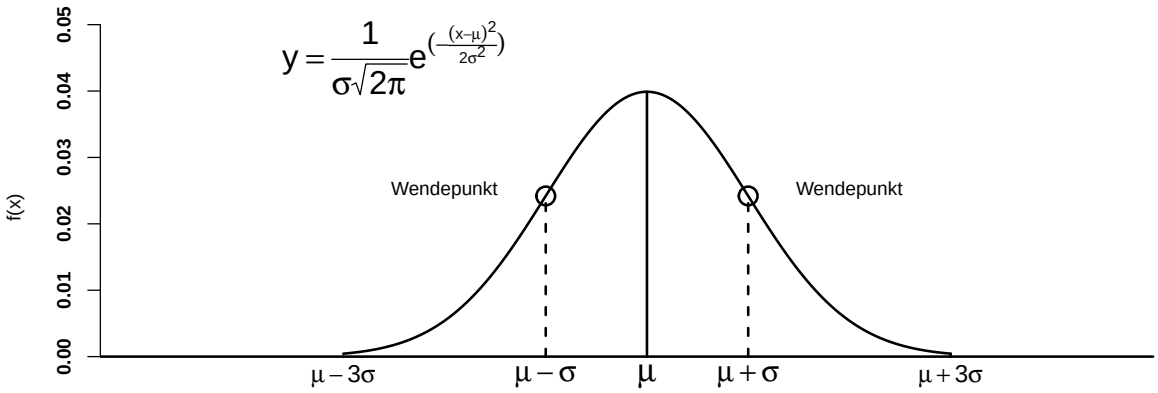
5.4.3 Normalverteilung

```
e <- exp(1); x <- seq(-3.5, +3.5, by=0.05)                    # Normalverteilung
y1 <- 3 * e^(-x^2/3)
y2 <- e^(-x^2)
# Abbildung 5.22
par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, y1, type="l", lty=4, ylab=" ", ylim=c(0, 3.2),
     lwd=3, xlab=" ", xlim=c(-3.5,3.5), axes=FALSE)
axis(1, at=NULL, pos=0, xlab="x")
axis(2, at=NULL, pos=0, las=1)
lines(x, y2, lty=1, lwd=3)
text(-1.5, 2.5, expression(g(x) == 3*e^frac(-x^2, 3)), cex=1.3)
text(0, 3.2, "y", cex=1.5)
text(1.3, 0.7, expression(f(x)==e^-x^2), cex=1.3)
text(3.7, 0, "x", cex=1.5)
```




```
mue <- 90; sig <- 10                                          # Normalverteilung
x   <- seq(60, 120, by=0.01); f   <- dnorm(x, mean=mue, sd=sig)

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, f, type="l", ylim=c(-0.005,0.05), xlim=c(40, 140),
                ylab="f(x)", xlab=" ", lwd=2, axes=FALSE)
axis(2)
text(65, 0.045,
 expression(y == frac(1, sigma*sqrt(2*pi)) * e^(- frac((x-mu)^2, 2*sigma^2))), cex=1.7)
lines(c(0,140), c(0,0))
lines(c(mue,mue), c(0,dnorm(mue, mean=mue, sd=sig)))
text(mue, -0.003, expression(mu), cex=1.7)
lines(c(mue-sig, mue-sig),
      c(0, dnorm(mue-sig, mean=mue, sd=sig)), lty=2)
text(mue-sig, -0.003, expression(mu-sigma), cex=1.5)
```

```
lines(c(mue+sig, mue+sig),
      c(0, dnorm(mue+sig, mean=mue, sd=sig)), lty=2)
text(mue+sig, -0.003, expression(mu+sigma), cex=1.5)
points(mue-sig, dnorm(mue-sig, mean=mue, sd=sig), cex=2)
text(mue-sig-10, 0.025, "Wendepunkt")
points(mue+sig, dnorm(mue+sig, mean=mue, sd=sig), cex=2)
text(mue+sig+10, 0.025, "Wendepunkt")
lines(c(mue-3*sig, mue-3*sig),
      c(0, dnorm(mue-3*sig, mean=mue, sd=sig)), lty=3)
text(mue-3*sig, -0.003, expression(mu-3*sigma), cex=1.3)
lines(c(mue+3*sig, mue+3*sig),
      c(0, dnorm(mue+3*sig, mean=mue, sd=sig)), lty=2)
text(mue+3*sig, -0.003, expression(mu+3*sigma), cex=1.3)
```




```
z <- seq(-3, +3, by=0.01)                                    # Normalverteilung
f <- dnorm(z, mean=0, sd=1); F <- pnorm(z, mean=0, sd=1)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(z, f, type="l", ylim=c(0, 0.4), xlim=c(-3.5, 3.5),
       ylab="f(z)", xlab=" ", lwd=2)
lines(c(-3.5, +3.5), c(0,0))
lines(c(0,0), c(0, dnorm(0,mean=0, sd=1)), lty=2, col="grey")
z1 <- seq(-4, -0.8, by=0.01)
f1 <- dnorm(z1, mean=0, sd=1)
polygon(c(z1,-0.8), c(f1,0), density = 20, angle = 45, col="black")
text(-2.5, 0.2, "F(-0.8)", cex=1.2)

plot(z, F, type="l", ylim=c(0, 1), xlim=c(-3.5, 3.5),
          ylab="F(z)", xlab=" ", lwd=2)
lines(c(-3.5, +3.5), c(0,0))
lines(c(0,0), c(0, pnorm(0,mean=0, sd=1)), lty=2, col="grey")
lines(c(-0.8, -0.8), c(0, pnorm(-0.8, mean=0, sd=1)), col="black")
lines(c(-4, -0.8), c(pnorm(-0.8, mean=0, sd=1),
```

```
                    pnorm(-0.8, mean=0, sd=1)), col="black")
text(-2.3, 0.28, "F(-0.8)", cex=1.2)
```

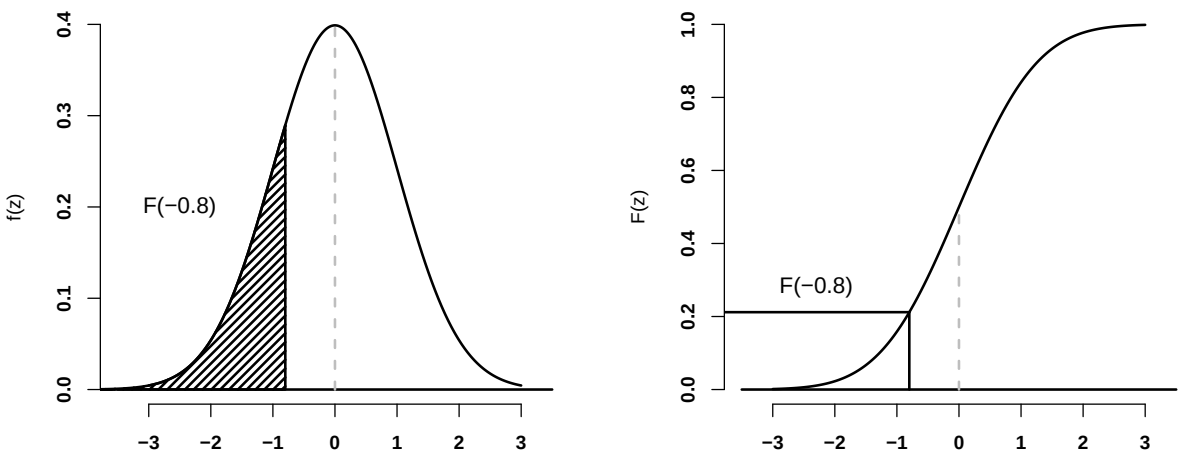


Tabelle zur Standardnormalverteilung:

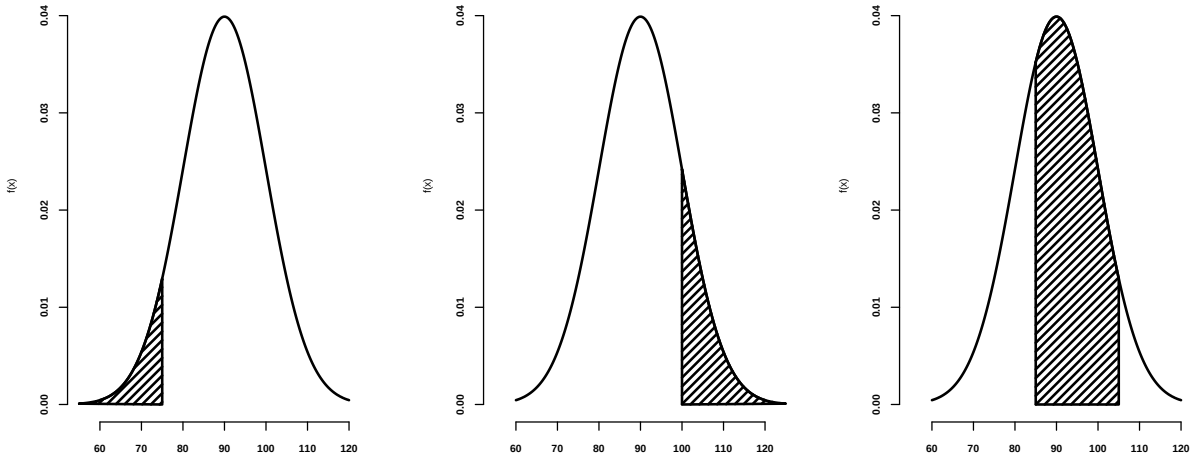
```
z <- seq(0, -3, by=-0.01); F <- pnorm(z, mean=0, sd=1)

tab <- matrix(data = NA, nrow = 30, ncol = 10,
                    byrow = FALSE, dimnames = NULL)
for (i in 1:30) tab[i,] <- round(F[((i-1)*10+1):((i-1)*10+10)],5)

ztr <- seq(0.0, -2.9, by=-0.1)
tab <- cbind(ztr, tab)
colnames(tab) <- c("z","0.00","0.01","0.02","0.03","0.04","0.05",
                   "0.06","0.07","0.08","0.09")
as.data.frame(tab)
```

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.50000	0.49601	0.49202	0.48803	0.48405	0.48006	0.47608	0.47210	0.46812	0.46414
-0.1	0.46017	0.45620	0.45224	0.44828	0.44433	0.44038	0.43644	0.43251	0.42858	0.42465
-0.2	0.42074	0.41683	0.41294	0.40905	0.40517	0.40129	0.39743	0.39358	0.38974	0.38591
-0.3	0.38209	0.37828	0.37448	0.37070	0.36693	0.36317	0.35942	0.35569	0.35197	0.34827
-0.4	0.34458	0.34090	0.33724	0.33360	0.32997	0.32636	0.32276	0.31918	0.31561	0.31207
-0.5	0.30854	0.30503	0.30153	0.29806	0.29460	0.29116	0.28774	0.28434	0.28096	0.27760
-0.6	0.27425	0.27093	0.26763	0.26435	0.26109	0.25785	0.25463	0.25143	0.24825	0.24510
-0.7	0.24196	0.23885	0.23576	0.23270	0.22965	0.22663	0.22363	0.22065	0.21770	0.21476
-0.8	0.21186	0.20897	0.20611	0.20327	0.20045	0.19766	0.19489	0.19215	0.18943	0.18673
-0.9	0.18406	0.18141	0.17879	0.17619	0.17361	0.17106	0.16853	0.16602	0.16354	0.16109
-1.0	0.15866	0.15625	0.15386	0.15151	0.14917	0.14686	0.14457	0.14231	0.14007	0.13786
-1.1	0.13567	0.13350	0.13136	0.12924	0.12714	0.12507	0.12302	0.12100	0.11900	0.11702
-1.2	0.11507	0.11314	0.11123	0.10935	0.10749	0.10565	0.10383	0.10204	0.10027	0.09853

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
-1.3	0.09680	0.09510	0.09342	0.09176	0.09012	0.08851	0.08691	0.08534	0.08379	0.08226
-1.4	0.08076	0.07927	0.07780	0.07636	0.07493	0.07353	0.07215	0.07078	0.06944	0.06811
-1.5	0.06681	0.06552	0.06426	0.06301	0.06178	0.06057	0.05938	0.05821	0.05705	0.05592
-1.6	0.05480	0.05370	0.05262	0.05155	0.05050	0.04947	0.04846	0.04746	0.04648	0.04551
-1.7	0.04457	0.04363	0.04272	0.04182	0.04093	0.04006	0.03920	0.03836	0.03754	0.03673
-1.8	0.03593	0.03515	0.03438	0.03362	0.03288	0.03216	0.03144	0.03074	0.03005	0.02938
-1.9	0.02872	0.02807	0.02743	0.02680	0.02619	0.02559	0.02500	0.02442	0.02385	0.02330
-2.0	0.02275	0.02222	0.02169	0.02118	0.02068	0.02018	0.01970	0.01923	0.01876	0.01831
-2.1	0.01786	0.01743	0.01700	0.01659	0.01618	0.01578	0.01539	0.01500	0.01463	0.01426
-2.2	0.01390	0.01355	0.01321	0.01287	0.01255	0.01222	0.01191	0.01160	0.01130	0.01101
-2.3	0.01072	0.01044	0.01017	0.00990	0.00964	0.00939	0.00914	0.00889	0.00866	0.00842
-2.4	0.00820	0.00798	0.00776	0.00755	0.00734	0.00714	0.00695	0.00676	0.00657	0.00639
-2.5	0.00621	0.00604	0.00587	0.00570	0.00554	0.00539	0.00523	0.00508	0.00494	0.00480
-2.6	0.00466	0.00453	0.00440	0.00427	0.00415	0.00402	0.00391	0.00379	0.00368	0.00357
-2.7	0.00347	0.00336	0.00326	0.00317	0.00307	0.00298	0.00289	0.00280	0.00272	0.00264
-2.8	0.00256	0.00248	0.00240	0.00233	0.00226	0.00219	0.00212	0.00205	0.00199	0.00193
-2.9	0.00187	0.00181	0.00175	0.00169	0.00164	0.00159	0.00154	0.00149	0.00144	0.00139

Beispiel zum Nüchternblutzucker:

```
mue <- 90; sig <- 10
x   <- seq(60, 120, by=0.01); f   <- dnorm(x, mean=mue, sd=sig)

par(mfrow=c(1,3), lwd=2, font.axis=2, bty="n", ps=9)
plot(x, f, type="l", ylim=c(0, 0.045), xlim=c(55,125),
           ylab="f(x)", xlab=" ", lwd=2)
xi <- seq(55, 75, by=0.01)
fi <- dnorm(xi, mean=mue, sd=sig)
polygon(c(xi, 75), c(fi, 0), density=20, angle=45,
        col="black", border=TRUE)
plot(x, f, type="l", ylim=c(0, 0.045), xlim=c(55,125),
           ylab="f(x)", xlab=" ", lwd=2)
xi <- seq(100,125, by=0.01)
fi <- dnorm(xi, mean=mue, sd=sig)
polygon(c(xi, 100), c(fi, 0), density=20, angle=45,
                col="black", border=TRUE)
plot(x, f, type="l", ylim=c(0, 0.045), xlim=c(55,125),
           ylab="f(x)", xlab=" ", lwd=2)
xi <- seq(85, 105, by=0.01); fi <- dnorm(xi, mean=mue, sd=sig)
polygon(c(xi, 105, 85), c(fi, 0, 0), density=20,
                angle=45, col="black", border=TRUE)
```



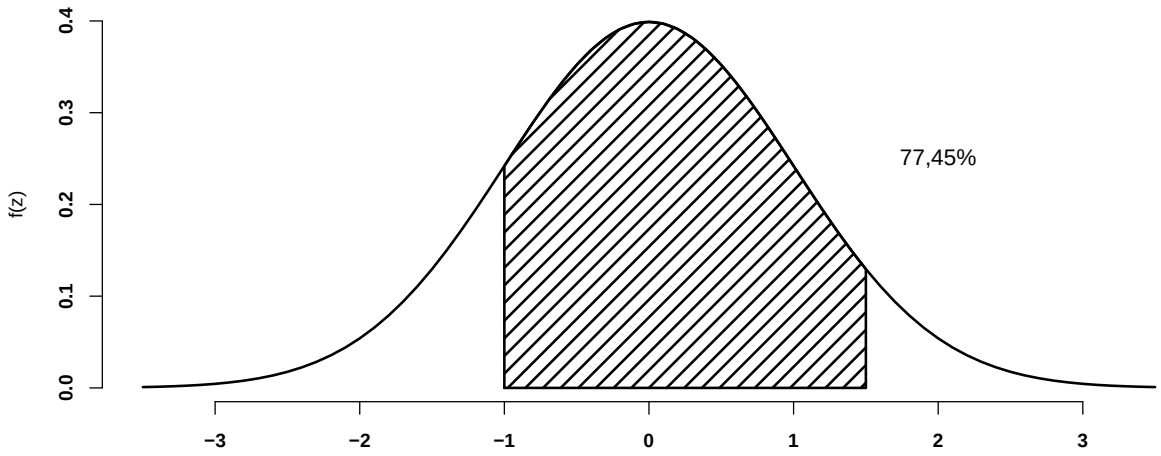

```
pnorm(75, mean=90, sd=10)
## [1] 0.0668072

pnorm(100, mean=90, sd=10, lower.tail=FALSE)
## [1] 0.1586553

pnorm(105, mean=90, sd=10) - pnorm(85, mean=90, sd=10)
## [1] 0.6246553
```

5.4.3.2 Hinweise und Beispiele zur Normalverteilung

```
mue <- 80; sig <-  8                                    # Hinweis 1 und 2 (Längen)
low <- mue - 3.5*sig; upp <- mue + 3.5*sig
x   <- seq(low, upp, by =0.1)
f   <- dnorm(x, mean=mue, sd =sig)

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=10)
plot(x, f, type="l", xlim=c(low, upp), xlab=" ", ylab=" ")
x1 <- seq(mue-3.5*sig, qnorm(0.025, mean=mue, sd=sig), by=0.01)
f1 <- dnorm(x1, mean=mue, sd=sig)
x2 <- seq(qnorm(0.975, mean=mue, sd=sig), mue+3.5*sig, by=0.01)
f2 <- dnorm(x2, mean=mue, sd=sig)
polygon(c(x1,qnorm(0.025, mean=mue, sd=sig)),
        c(f1,0), density = 20, angle = 45)
polygon(c(qnorm(0.975, mean=mue, sd=sig),x2),
        c(0,f2), density = 20, angle = 45)
```

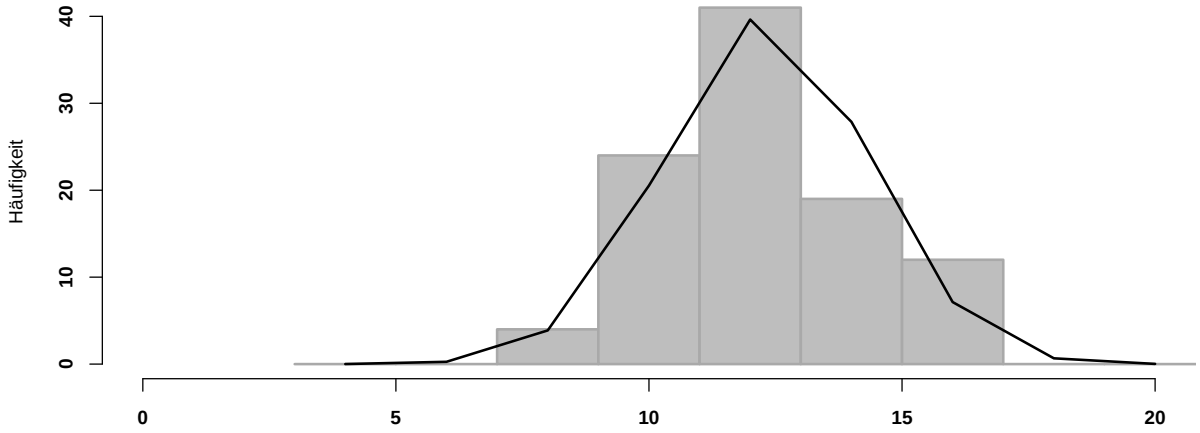

```r
qnorm(0.025, mean=mue, sd=sig)
## [1] 64.32029

qnorm(0.975, mean=mue, sd=sig)
## [1] 95.67971
```

```r
mue <- 0; sig <- 1                                          # Hinweis 3
low <- mue - 3.5*sig; upp <- mue + 3.5*sig
x   <- seq(low, upp, by =0.1); f   <- dnorm(x, mean=mue, sd =sig)

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, f, type="l", xlim=c(low, upp), xlab=" ", ylab="f(z)")
x1 <- seq(-1, 1.5, by=0.01)
f1 <- dnorm(x1, mean=mue, sd=sig)
polygon(c(-1,x1,1.5), c(0,f1,0), density = 10, angle = 45)
text(2, 0.25, "77,45%", cex=1.2)
```




```r
mue <- 150; sig <-  10                                          #  Hinweis 4
qnorm(0.06, mean=mue, sd=sig)
## [1] 134.4523
pnorm(160, mean=mue, sd=sig) - pnorm(130, mean=mue, sd=sig)
## [1] 0.8185946
qnorm(0.025, mean=mue, sd=sig)
## [1] 130.4004
qnorm(0.975, mean=mue, sd=sig)
## [1] 169.5996
```

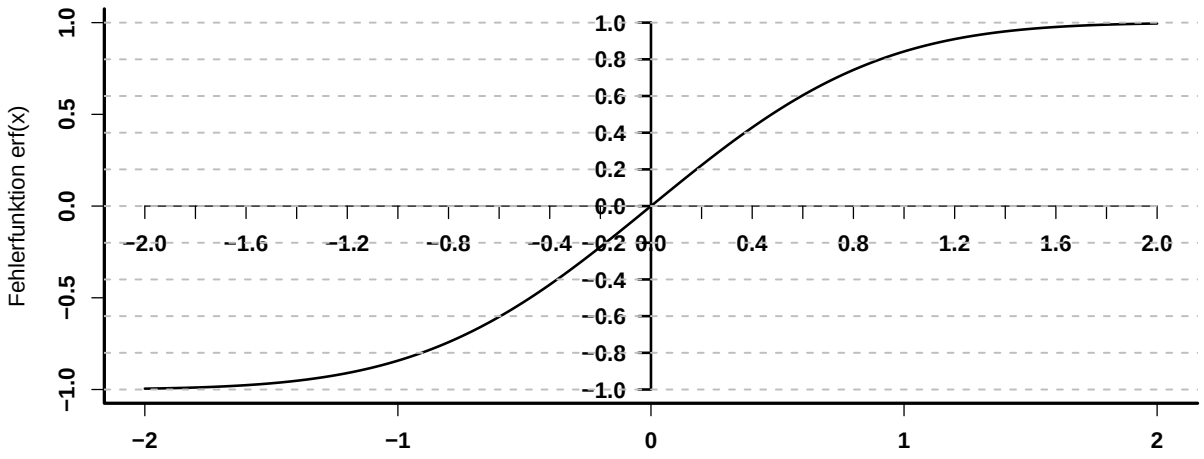
```r
mue <- 12; sig <-  2;  n <- 100;                                # Hinweis 6
y.val <- rnorm(n, mean=mue, sd=sig)
brk <- c(3, 5, 7, 9, 11, 13, 15, 17, 19, 21)

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
hist(y.val, breaks = brk, ylim=c(0,42), xlim=c(0,20), main=" ",
     border="darkgrey", xlab=" ", ylab="Häufigkeit", col="grey")
mid <- c(4, 6, 8, 10, 12, 14, 16, 18, 20)
z.val <- (mid - mean(y.val)) / sd(y.val)
f.val <- dnorm(z.val, mean=0, sd=1)
y.est <- (2 * n / sd(y.val)) * f.val
lines(mid, y.est)
```


Häufigkeit
0
10
20
30
40
0
5
10
15
20

5.4.4 Halbnormalverteilung

Fehlerfunktion:

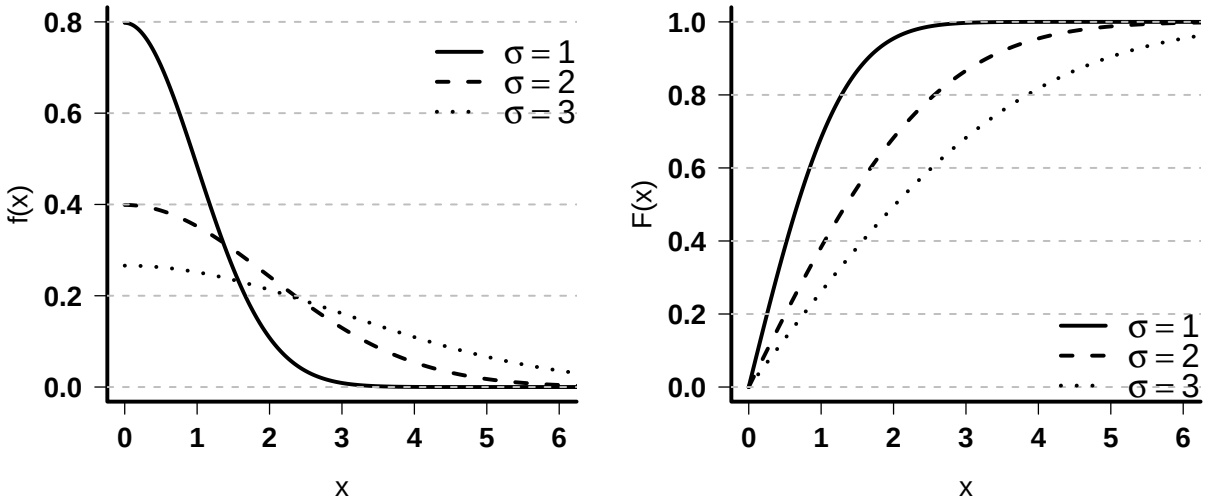
```
erf       <- function(x) 2 * pnorm(x * sqrt(2)) - 1           # Fehlerfunktion

sigma <- 1
x     <- seq(-2, +2, by=0.01)

par(mfcol=c(1,1), lwd=2.5, font.axis=2, bty="l", ps=13)
plot(x, erf(x), type="l", xlab=" ", ylab="Fehlerfunktion erf(x)", axes=F, lwd= 2)
axis(1, pos=0, lty=1, lwd=1, las=1, xlim=c(-2, +2), xaxp=c(-2, +2, 20))
axis(2, pos=0, lty=1, lwd=2, las=1, ylim=c(-1, +2), yaxp=c(-1, +1, 10))
abline(h=seq(-1,+1,0.2), col="grey", lty=2, lwd=1.5)
```



Dichte- und Verteilungsfunktion:

```
dhalfnorm <- function(x, s)  (sqrt(2)/(s*sqrt(pi)))*exp(-x^2/(2*s^2))  # Dichte
phalfnorm <- function(x, s) erf(x/(s*sqrt(2)))                # Verteilung
qhalfnorm <- function(p, s) s * qnorm((1+p)/2)                # Quantile

sigma <- c(1, 2, 3)
x     <- seq(0, 8, by=0.02)

par(mfcol=c(1,2), lwd=3, font.axis=2, bty="l", ps=16)
plot(x, dhalfnorm(x, sigma[1]), type="l", las=1, xlab="x", ylab="f(x)",
     xlim=c(0,6), ylim=c(0,0.8), lty=1)
lines(x, dhalfnorm(x, sigma[2]), lty=2)
lines(x, dhalfnorm(x, sigma[3]), lty=3)
abline(h=seq(0,0.8,0.2), col="grey", lty=2, lwd=1.5)
ltxt <- c(expression(sigma == 1),expression(sigma == 2),expression(sigma == 3))
```

```
legend(4, 0.8, legend = ltxt, lty=1:3, bty="n", cex=1.2)

plot(x, phalfnorm(x, sigma[1]), type="l", las=1, xlab="x", ylab="F(x)",
     xlim=c(0,6), ylim=c(0,1), lty=1)
lines(x, phalfnorm(x, sigma[2]), lty=2)
lines(x, phalfnorm(x, sigma[3]), lty=3)
abline(h=seq(0,1,0.2), col="grey", lty=2, lwd=1.5)
legend(4, 0.25, legend = ltxt, lty=1:3, bty="n", cex=1.2)
```

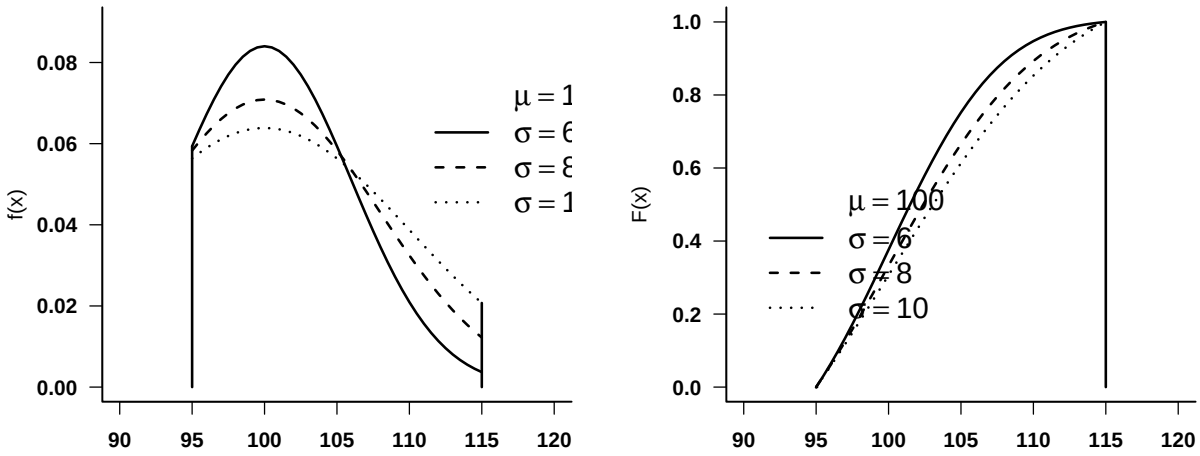


Kiefermodell-Messungen:

```
x <- c(-0.0554, -0.0165, 0.0156,  0.0115, -0.0544,
       0.0094,  0.0076, 0.0253, -0.0226, -0.0306)
n <- length(x)
s.hat <- sqrt(sum(x^2)/n); round(s.hat, 4)
## [1] 0.0298

sigma <- 0.03
E <- sigma*sqrt(2/pi); E                                    # Erwartungswert
## [1] 0.02393654
qhalfnorm(0.5, sigma)                                       # Medianwert
## [1] 0.02023469
qhalfnorm(0.10, sigma); qhalfnorm(0.95, sigma)
## [1] 0.00376984
## [1] 0.05879892
```

5.4.5 Gestutzte Normalverteilung

```
truncnorm <- function(x, a=-Inf, b=+Inf, mu, sigma) {
    za <- (a-mu)/sigma; zb <- (b-mu)/sigma; z <- (x-mu)/sigma
    dz <- dnorm(z); da <- dnorm(za); db <- dnorm(zb)
    pz <- pnorm(z); pa <- pnorm(za); pb <- pnorm(zb)
    f  <- dz / (sigma*(pb - pa))                              # Dichtefunktion
    F  <- (pz - pa) / (pb-pa)                                 # Verteilungsfunktion
    EX <- mu + ((da - db)/(pb - pa))*sigma                    # Erwartungswert
    VX <- sigma^2*(1 + (za*da - zb*db)/(pb - pa)              # Varianz
                   - ((da-db)/(pb-pa))^2)
    return(list(pdf=f, cdf=F, M=EX, V=VX))
}

mu    <- 100;  sigma <- c(6, 8, 10)
a <- 95; b <- 115
x     <- seq(a, b, 0.5)
n     <- length(x)
distr1 <- truncnorm(x, a, b, mu, sigma[1])
distr2 <- truncnorm(x, a, b, mu, sigma[2])
distr3 <- truncnorm(x, a, b, mu, sigma[3])

par(mfrow=c(1,2), lwd=1.5, font.axis=2, bty="l", ps=12)
plot(x, distr1$pdf, las=1, type="l", lwd=2, lty=1,
     xlim=c(90, 120), ylim=c(0, 0.09),
     xlab=" ", ylab="f(x)")
lines(x, distr2$pdf, lty=2, lwd=2)
lines(x, distr3$pdf, lty=3, lwd=2)
lines(c(x[1],x[1]), c(0,distr1$pdf[1]), lwd=2)
lines(c(x[n],x[n]), c(0,distr3$pdf[n]), lwd=2)
ltxt <- c(expression(mu == 100), expression(sigma == 6),
          expression(sigma == 8), expression(sigma == 10))
legend(110, 0.08, legend = ltxt, lty=0:3, bty="n", cex=1.4, lwd=2)

plot(x, distr1$cdf, las=1, type="l",lwd=2,
     xlim=c(90, 120), ylim=c(0,1),
     xlab=" ", ylab="F(x)")
lines(x, distr2$cdf, lty=2, lwd=2)
lines(x, distr3$cdf, lty=3, lwd=2)
lines(c(x[1],x[1]), c(0,distr1$cdf[1]), lwd=2)
lines(c(x[n],x[n]), c(0,distr1$cdf[n]), lwd=2)
legend(90, 0.6, legend = ltxt, lty=0:3, bty="n", cex=1.4, lwd=2)
```



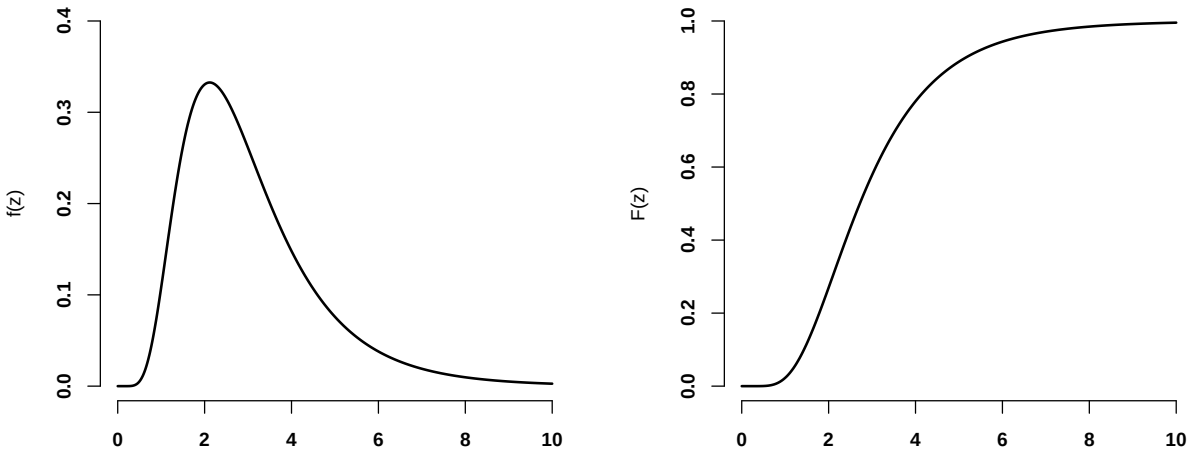
Beispiel Schraubenlängen:

Funktionen zur gestutzten Normalverteilung in library(truncnorm)

```
truncnorm(x=10.4, a=9.5, b=10.5, mu=10, sigma=0.3)
## $pdf
## [1] 0.6044765
##
## $cdf
## [1] 0.9519903
##
## $M
## [1] 10
##
## $V
## [1] 0.05700298

library(truncnorm)
ptruncnorm(10.4, a=9.5, b=10.5, mean=10, sd=0.3)
## [1] 0.9519903
etruncnorm( a=9.5, b=10.5, mean=10, sd=0.3)
## [1] 10
vtruncnorm( a=9.5, b=10.5, mean=10, sd=0.3)
## [1] 0.05700298
```

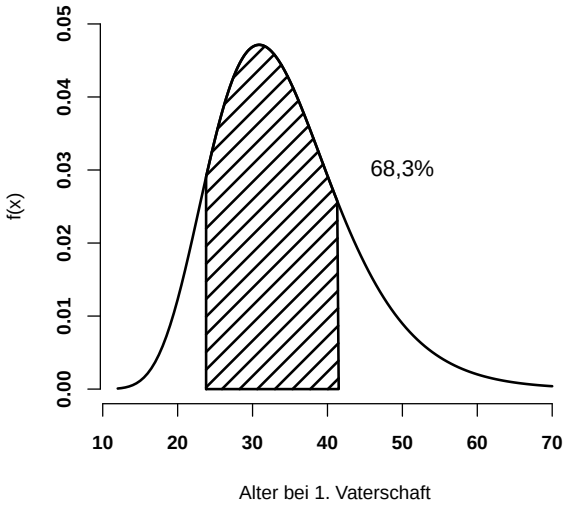
5.4.6 Lognormalverteilung:

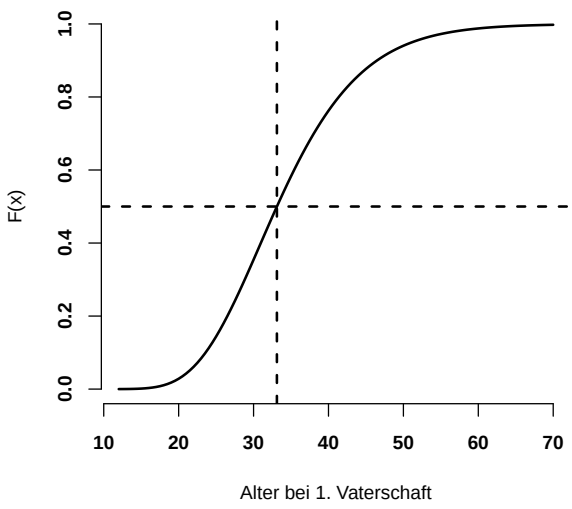
```
x <- seq(0, 10, by=0.01)
f <- dlnorm(x, meanlog=1, sdlog=0.5)
F <- plnorm(x, meanlog=1, sdlog=0.5)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, f, type="l", ylim=c(0, 0.4), xlim=c(0, 10),
     ylab="f(z)", xlab=" ",  lwd=2)
plot(x, F, type="l", ylim=c(0, 1), xlim=c(0, 10),
     ylab="F(z)", xlab=" ",  lwd=2)
```



Beispiel Alter bei 1. Vaterschaft:

```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
x    <- seq(12, 70, by=0.5)
f    <- dlnorm(x, meanlog=3.5, sdlog=sqrt(0.07))
plot(x, f, xlab="Alter bei 1. Vaterschaft", ylab="f(x)",
            type="l", ylim=c(0,0.05))
x1 <- seq(23.8, 41.5, by=0.5)
f1 <- dlnorm(x1, meanlog=3.5, sdlog=sqrt(0.07))
polygon(c(23.8,x1,41.5), c(0,f1,0), density = 10, angle = 45)
text(50, 0.03, "68,3%", cex=1.2)
F    <- plnorm(x, meanlog=3.5, sdlog=sqrt(0.07))
plot(x, F, xlab="Alter bei 1. Vaterschaft", ylab="F(x)",
            type="l", ylim=c(0,1))
abline(h=0.5, lty=2)
abline(v=exp(3.5), lty=2)
```






```
age <- c(36, 44, 38, 41, 26, 59, 27, 26, 34, 30,
         42, 29, 31, 21, 28, 20, 24, 32, 40, 23)
n   <- length(age)
```

Funktion fitdistr() in library(MASS)

```
library(MASS)
fitdistr(age, "lognormal")                                  # fitdistr() in library(MASS)
##     meanlog       sdlog
##   3.44577837   0.26813478
##  (0.05995676) (0.04239583)

mue  <- sum(log(age))/n; mue                                # Schätzung Erwartungswert
## [1] 3.445778
s2   <- sum((log(age)-mue)^2)/n; s2                         # Schätzung Varianz
## [1] 0.07189626

mue.age <- exp(mue + 0.5*s2); mue.age                       # Erwartungswert von X
## [1] 32.51581
s2.age  <- exp(2*mue + s2)*(exp(s2)-1); sqrt(s2.age)        # Varianz von X
## [1] 8.877702

ms.age <- exp(mue); ms.age                                  # Limpert xq-Stern
## [1] 31.36769
ss.age <- exp(sqrt(sum(log(age/ms.age)^2)/(n-1)))
ss.age                                                      # Limpert sd-Stern
## [1] 1.316663
```

Beispiel mit 20 Messwerten:

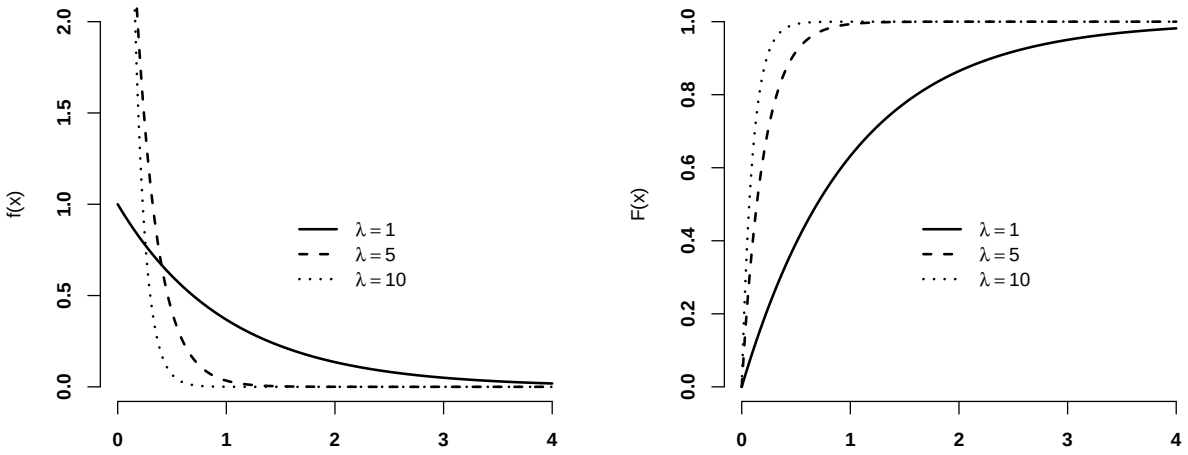
```
x    <- c(3, 4, 5, 5, 5, 5, 5, 6, 7, 7, 7, 7, 8, 8, 9, 9, 10, 11, 12, 14)
lgx  <- log10(x); lgx2 <- lgx^2
tab  <- cbind(x, lgx, lgx2)
colnames(tab) <- c("x", "lg(x)", "(lg(x)^2")
as.data.frame(tab)
```

x	lg(x)	(lg(x)^2
3	0.4771213	0.2276447
4	0.6020600	0.3624762
5	0.6989700	0.4885591
5	0.6989700	0.4885591
5	0.6989700	0.4885591
5	0.6989700	0.4885591
5	0.6989700	0.4885591
6	0.7781513	0.6055194
7	0.8450980	0.7141907
7	0.8450980	0.7141907
7	0.8450980	0.7141907
7	0.8450980	0.7141907
8	0.9030900	0.8155715
8	0.9030900	0.8155715
9	0.9542425	0.9105788
9	0.9542425	0.9105788
10	1.0000000	1.0000000
11	1.0413927	1.0844987
12	1.0791812	1.1646322
14	1.1461280	1.3136095

```
median.L       <- 10^mean(lgx);                    median.L
## [1] 6.850103
streufaktor    <- 10^(sqrt(sd(lgx)^2));            streufaktor
## [1] 1.475594
mittelwert.L   <- 10^(mean(lgx)+1.1513*sd(lgx)^2); mittelwert.L
## [1] 7.388674
dichtemittel.L <- 10^(mean(lgx)-2.3026*sd(lgx)^2); dichtemittel.L
## [1] 5.88787
```

5.4.7 Exponentialverteilung

```
x <- seq(0, 4, by=0.01)
fx1 <- dexp(x, rate=1);  Fx1 <- pexp(x, rate=1)
fx2 <- dexp(x, rate=5);  Fx2 <- pexp(x, rate=5)
fx3 <- dexp(x, rate=10); Fx3 <- pexp(x, rate=10)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, fx1, type="l", ylim=c(0, 2), xlim=c(0, 4),
            ylab="f(x)", xlab=" ", lwd=2, lty=1)
lines(x, fx2, type="l", lwd=2, lty=2)
lines(x, fx3, type="l", lwd=2, lty=3)
legend(1.5, 1, bty="n", c(expression(lambda ==1), expression(lambda ==5),
            expression(lambda ==10)), lty=c(1,2,3), cex=1)

plot(x, Fx1, type="l", ylim=c(0, 1), xlim=c(0, 4),
            ylab="F(x)", xlab=" ", lwd=2, lty=1)
lines(x, Fx2, type="l", lwd=2, lty=2)
lines(x, Fx3, type="l", lwd=2, lty=3)
legend(1.5, 0.5, bty="n", c(expression(lambda ==1),
            expression(lambda ==5), expression(lambda ==10)),
       lty=c(1,2,3), cex=1)
```

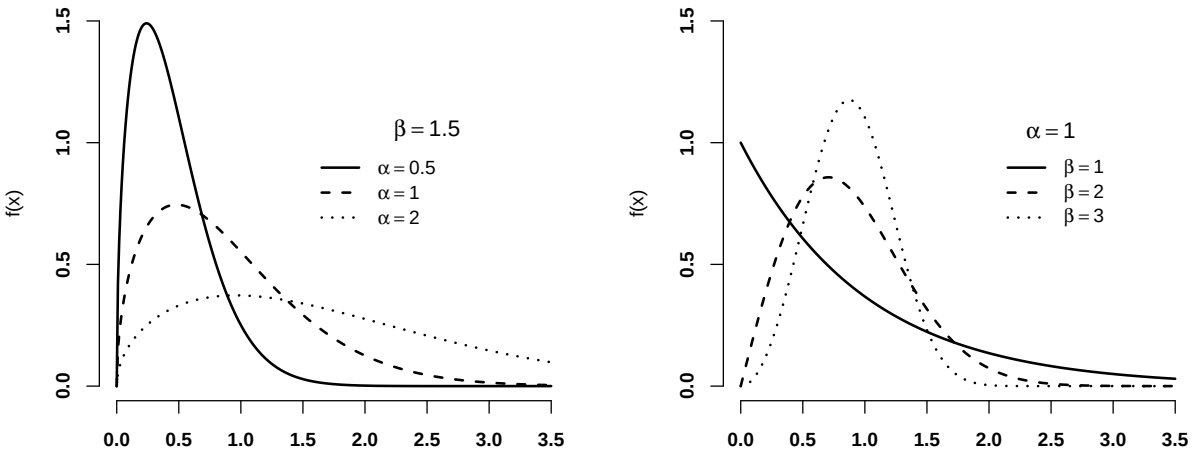


Beispiel zu Wartezeiten und Brenndauer von Gühbirnen:

```
1 - pexp(4, rate = 0.5)                                    # Wartezeiten
## [1] 0.1353353

1 - pexp(110, rate=0.01)                                   # Glühbirnen
## [1] 0.3328711
```

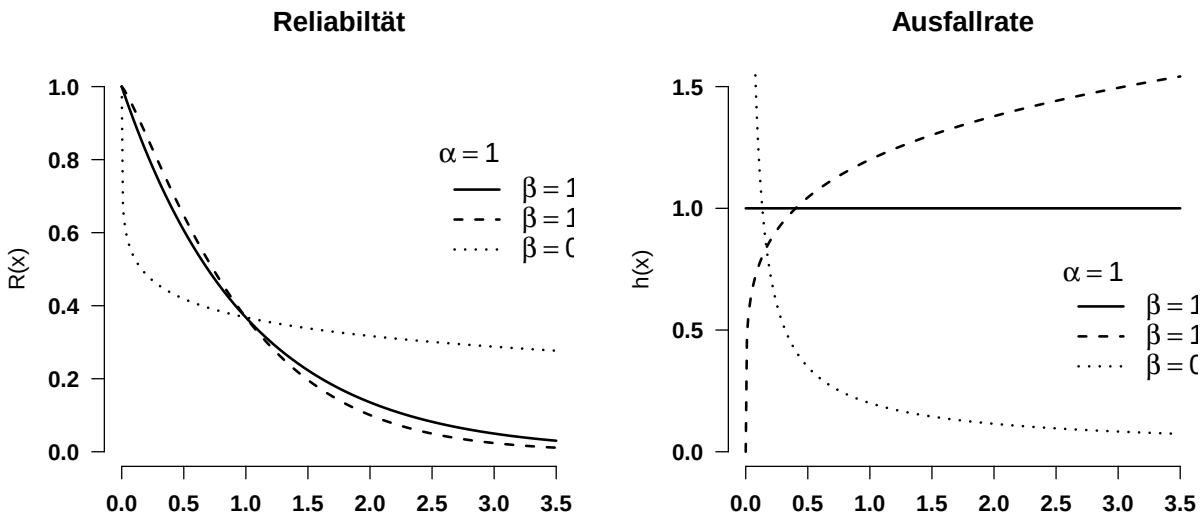
5.4.8 Weibull-Verteilung

```
x <- seq(0, 3.5, by=0.01)

fxa1 <- dweibull(x, 1.5, 0.5)
fxa2 <- dweibull(x, 1.5, 1)
fxa3 <- dweibull(x, 1.5, 2)

fxb1 <- dweibull(x, 1, 1)
fxb2 <- dweibull(x, 2, 1)
fxb3 <- dweibull(x, 3, 1)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, fxa1, type="l", ylim=c(0, 1.5), xlim=c(0, 3.5),
            ylab="f(x)", xlab=" ", lwd=2, lty=1)
lines(x, fxa2, type="l", lwd=2, lty=2)
lines(x, fxa3, type="l", lwd=2, lty=3)
legend(1.5, 1, bty="n", c(expression(alpha==0.5), expression(alpha==1),
            expression(alpha==2)), lty=c(1,2,3), cex=1)
text(2.5,1.05,expression(beta==1.5), cex=1.2)

plot(x, fxb1, type="l", ylim=c(0, 1.5), xlim=c(0, 3.5),
            ylab="f(x)", xlab=" ", lwd=2, lty=1)
lines(x, fxb2, type="l", lwd=2, lty=2)
lines(x, fxb3, type="l", lwd=2, lty=3)
legend(2, 1, bty="n", c(expression(beta==1), expression(beta==2),
            expression(beta==3)), lty=c(1,2,3), cex=1)
text(2.5,1.05,expression(alpha==1.0), cex=1.2)
```



Reliabilität und Ausfallraten:

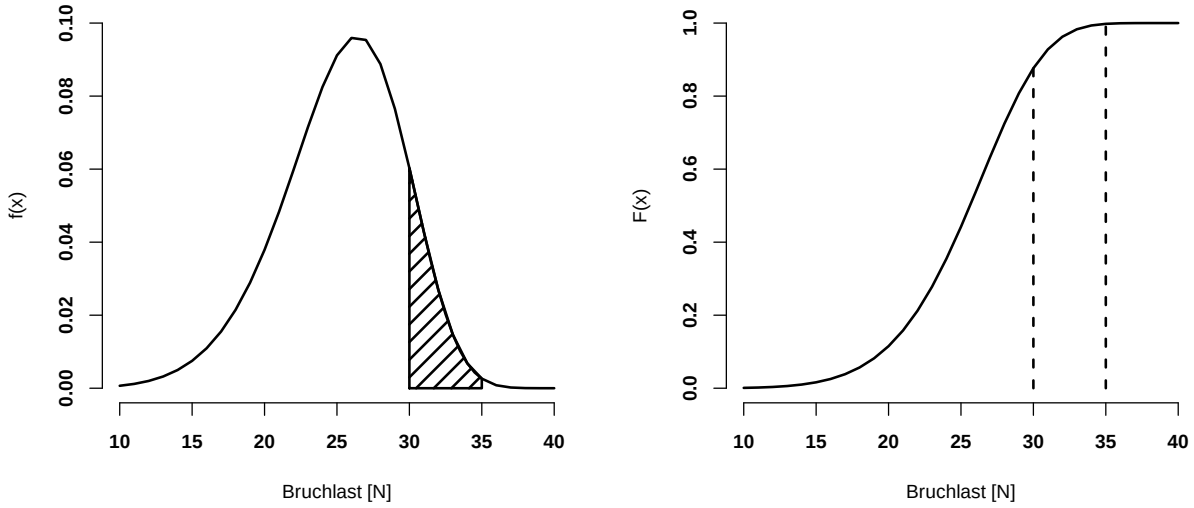
```
x <- seq(0, 3.5, by=0.01)
beta <- c(1, 1.2, .2); alpha <- 1

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=13)
Rx   <- matrix( 0, nrow=3, ncol=length(x))
for (i in 1:3) Rx[i,]= exp(-(x/alpha)^beta[i])
plot(x, Rx[1,], type="l", ylim=c(0, 1.0), xlim=c(0, 3.5), las=1,
     ylab="R(x)", xlab=" ", main="Reliabiltät", lwd=2, lty=1)
lines(x, Rx[2,], type="l", lwd=2, lty=2)
lines(x, Rx[3,], type="l", lwd=2, lty=3)
legend(2.5, 0.8, bty="n", c(expression(beta==1.0), expression(beta==1.2),
                            expression(beta==0.2)), lty=c(1,2,3), cex=1.2)
text(2.8, 0.82,expression(alpha==1), cex=1.2)

hx   <- matrix( 0, nrow=3, ncol=length(x))
for (i in 1:3) hx[i,] <- beta[i]/alpha*(x/alpha)^(beta[i]-1)
plot(x, hx[1,], type="l", ylim=c(0, 1.5), xlim=c(0, 3.5), las=1,
     ylab="h(x)", xlab=" ", main="Ausfallrate", lwd=2, lty=1)
lines(x, hx[2,], type="l", lwd=2, lty=2)
lines(x, hx[3,], type="l", lwd=2, lty=3)
legend(2.5, 0.72, bty="n", c(expression(beta==1.0), expression(beta==1.2),
                             expression(beta==0.2)), lty=c(1,2,3), cex=1.2)
text(2.8, 0.74, expression(alpha==1), cex=1.2)
```



Beispiel zur Bruchfestigkeit keramischer Werkstoffe:

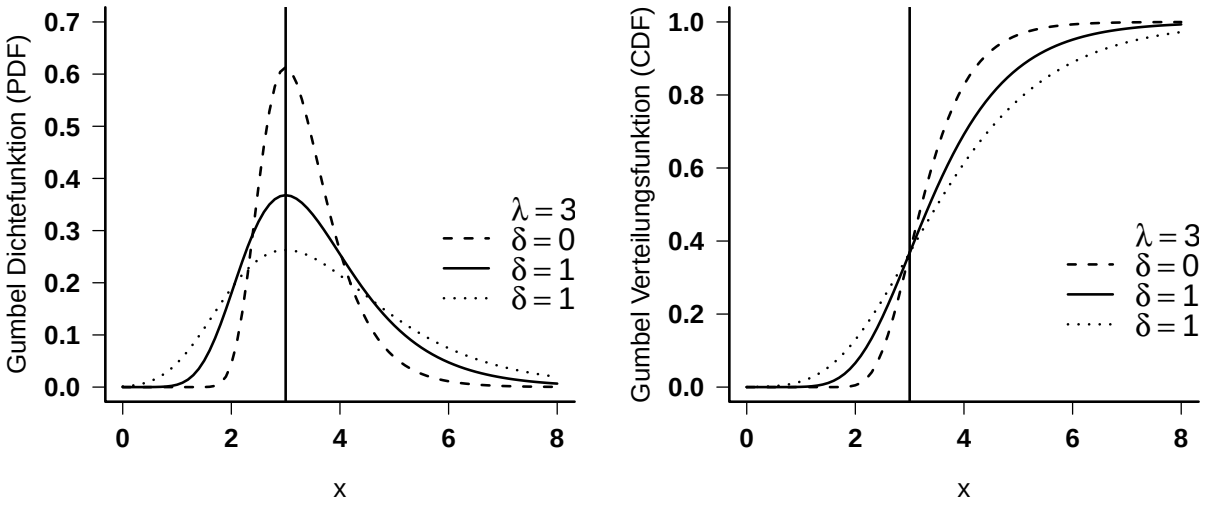
```
x <- seq(10, 40, by=1)
fx <- dweibull(x, 7, 27)
Fx <- pweibull(x, 7, 27, lower.tail = TRUE, log.p = FALSE)

pweibull(35, 7, 27) - pweibull(30, 7, 27)
## [1] 0.1214624
```

```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, fx , type="l", ylim=c(0, 0.1), xlim=c(10, 40),
           ylab="f(x)", xlab="Bruchlast [N]", lwd=2, lty=1)
x1 <- seq(30, 35, by=0.5)
f1 <- dweibull(x1, 7, 27)
polygon(c(30,x1,35), c(0,f1,0), density = 10, angle = 45)
plot(x, Fx , type="l", ylim=c(0, 1), xlim=c(10, 40),
             ylab="F(x)", xlab="Bruchlast [N]", lwd=2, lty=1)
lines(c(30, 30), c(0,pweibull(30, 7, 27)), type="l", lwd=2, lty=2)
lines(c(35, 35), c(0,pweibull(35, 7, 27)), type="l", lwd=2, lty=2)
```



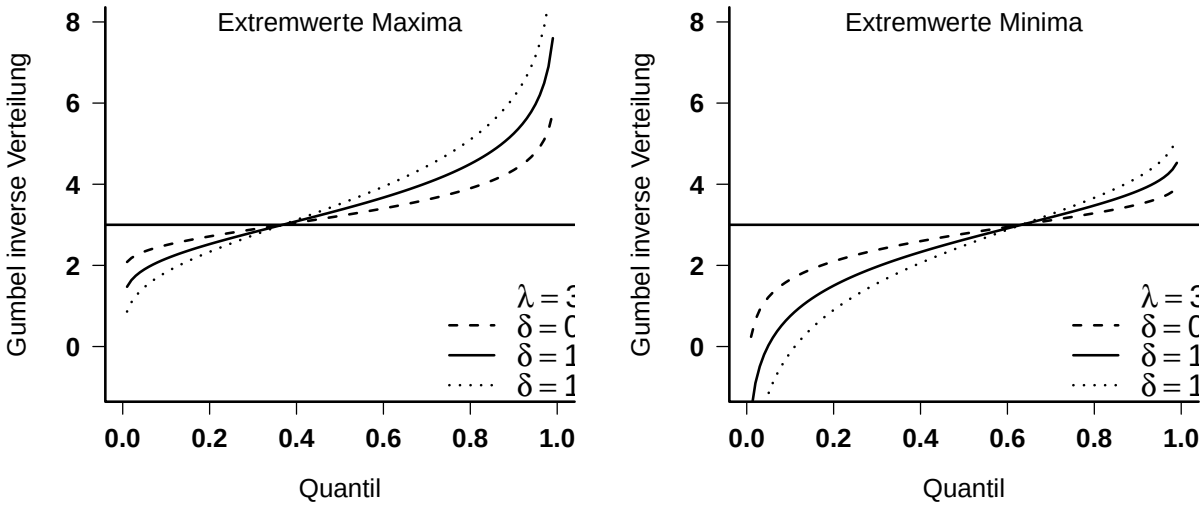
5.4.9 Extremwertverteilung Typ I (Gumbel)

```
dgumbel <- function(x, lambda, delta, dir="max") {          # Dichtefunktion
    z <- (x - lambda)/delta
    if (dir=="max") return(exp(-z - exp(-z))/delta)         # Maximum
    if (dir=="min") return(exp(z - exp(z))/delta)           # Minimum
}
pgumbel <- function(q, lambda, delta, dir="max") {          # Verteilungsfunktion
    z <- (q - lambda)/delta
    if (dir=="max") return(exp(-exp(-(z))))                 # Maximum
    if (dir=="min") return(1 - exp(-exp(z)))                # Minimum
}
qgumbel <- function(p, lambda, delta, dir="max") {          # Quantilfunktion
    if (dir=="max") return(lambda-delta*log(-log(p)))       # Maximum
    if (dir=="min") return(lambda+delta*log(-log(1-p)))     # Minimum
}
rgumbel <- function(n, lambda, delta) {                     # Zufallszahlen
    z <- runif(n, 0, 1)
    lambda - delta*log(-log(z))
}

lambda <- 3
delta  <- c(0.6, 1.0, 1.4)
x <- seq(0, 8, 0.1)

par(mfrow=c(1,2), lwd=2.0, font.axis=2.0, bty="l", ps=15)
plot(x, dgumbel(x, lambda, delta[1]), type ="l", las=1, lty=2,
     ylab="Gumbel Dichtefunktion (PDF)", ylim=c(0,0.7))
lines(x, dgumbel(x, lambda, delta[2]), lty=1)
lines(x, dgumbel(x, lambda, delta[3]), lty=3)
legend(5.5, 0.4, bty="n",
       c(expression(lambda==3.0),expression(delta ==0.6),
         expression(delta ==1.0), expression(delta ==1.4)),
       lty=c(0,2,1,3), cex=1.2)
abline(v=lambda)

plot(x, pgumbel(x, lambda, delta[1]), type ="l", las=1, lty=2,
     ylab="Gumbel Verteilungsfunktion (CDF)", ylim=c(0,1.0))
lines(x, pgumbel(x, lambda, delta[2]), lty=1)
lines(x, pgumbel(x, lambda, delta[3]), lty=3)
legend(5.5, 0.5, bty="n",
       c(expression(lambda==3.0),expression(delta ==0.6),
         expression(delta ==1.0), expression(delta ==1.4)),
       lty=c(0,2,1,3), cex=1.2)
abline(v=lambda)
```



Extremwerte - Maxima / Minima

```
q <- seq(0,1,by=0.01)
par(mfrow=c(1,2), lwd=2.0, font.axis=2.0, bty="l", ps=15)
plot(q, qgumbel(q, lambda, delta[1], dir="max"), type ="l", las=1, lty=2,
     xlab="Quantil", ylab="Gumbel inverse Verteilung", ylim=c(-1,8))
lines(q, qgumbel(q, lambda, delta[2], dir="max"), lty=1)
lines(q, qgumbel(q, lambda, delta[3], dir="max"), lty=3)
legend(0.7, 2, bty="n",
       c(expression(lambda==3.0),expression(delta ==0.6),
         expression(delta ==1.0), expression(delta ==1.4)),
       lty=c(0,2,1,3), cex=1.2)
abline(h=lambda)
text(0.5, 8, "Extremwerte Maxima")

plot(q, qgumbel(q, lambda, delta[1], dir="min"), type ="l", las=1, lty=2,
     xlab="Quantil", ylab="Gumbel inverse Verteilung", ylim=c(-1,8))
lines(q, qgumbel(q, lambda, delta[2], dir="min"), lty=1)
lines(q, qgumbel(q, lambda, delta[3], dir="min"), lty=3)
legend(0.7, 2, bty="n",
       c(expression(lambda==3.0),expression(delta ==0.6),
         expression(delta ==1.0), expression(delta ==1.4)),
       lty=c(0,2,1,3), cex=1.2)
abline(h=lambda)
text(0.5, 8, "Extremwerte Minima")
```

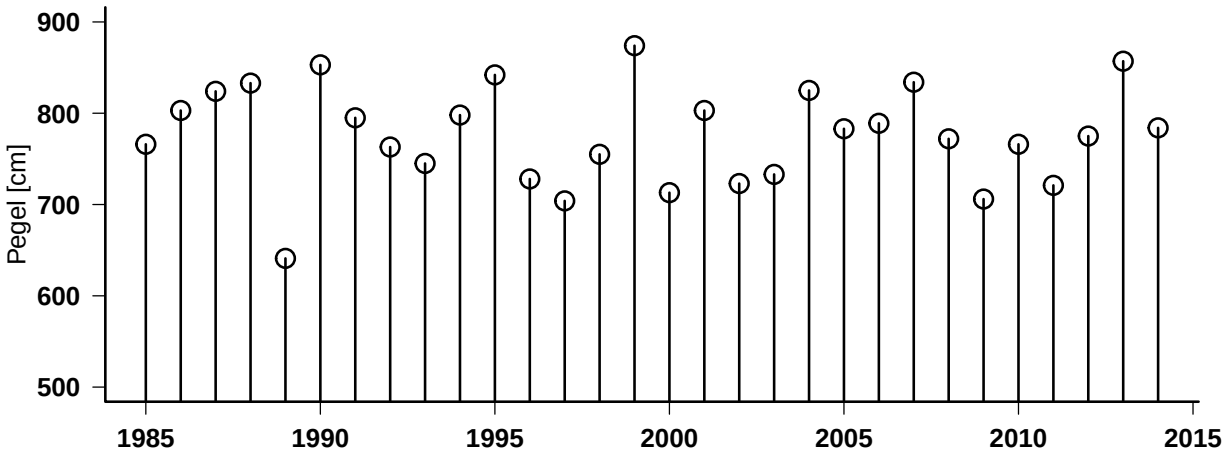


Beispiel Pegelstände am Rhein:

```
hydro <- c(766, 803, 824, 833, 641, 853, 795, 763, 745, 798,
           842, 728, 704, 755, 874, 713, 803, 723, 733, 825,
           783, 789, 834, 772, 706, 766, 721, 775, 857, 784)


tab <- matrix(hydro, nrow=3)
rownames(tab) <- c("1986-1994","1995-2004","2005-2014")
as.data.frame(tab)
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
1986-1994	766	833	795	798	704	713	733	789	706	775
1995-2004	803	641	763	842	755	803	825	834	766	857
2005-2014	824	853	745	728	874	723	783	772	721	784

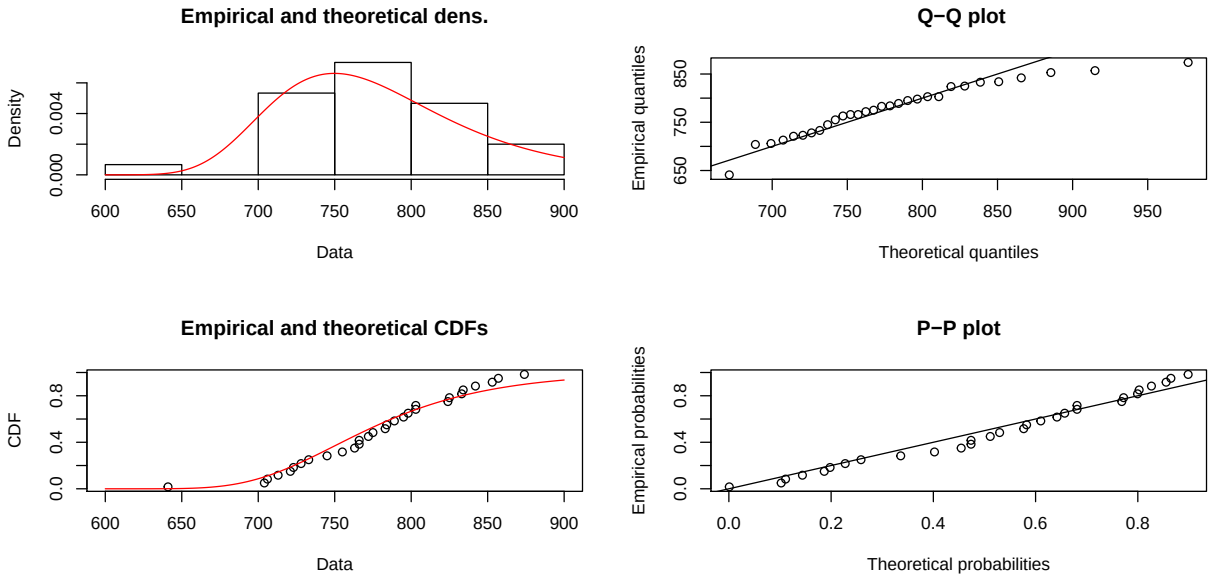
```
n    <- length(hydro)
jahr <- 1985:2014
par(mfrow=c(1,1), lwd=2.0, font.axis=2.0, bty="l", ps=15)
plot(jahr, hydro, cex=2, xlab=" ", ylab="Pegel [cm]", las=1,
     ylim=c(500,900))
for (i in 1:n) lines(c(jahr[i],jahr[i]),c(0,hydro[i]))
```




```
x <- hydro                                                    # Schätzung
n <- length(x); m <- mean(x)
dhat  <- uniroot(function(dhat)
    m-(sum(x*exp(-x/dhat)))/(sum(exp(-x/dhat)))-dhat,
    interval=c(0.5*sd(x), 2*sd(x)), tol = 1e-9)$root
lhat  <- -dhat*log(sum(exp(-x/dhat))/n)
cat("\n","Sch?tzung f?r delta =",dhat,"und lambda =",lhat,"\n")
##
##  Sch?tzung f?r delta = 55.49749 und lambda = 749.7549
```

Funktion fitdist() in library(fitdistrplus)

```
library(fitdistrplus)
fit <- fitdist(x, "gumbel", start=list(lambda=5, delta=5), method="mle")
summary(fit)
## Fitting of the distribution ' gumbel ' by maximum likelihood
## Parameters :
##        estimate Std. Error
## lambda 749.7983  10.778887
## delta   55.5849   7.141288
## Loglikelihood:  -165.1819   AIC:  334.3638   BIC:  337.1662
## Correlation matrix:
##           lambda     delta
## lambda 1.0000000 0.3369727
## delta  0.3369727 1.0000000
plot(fit)
```

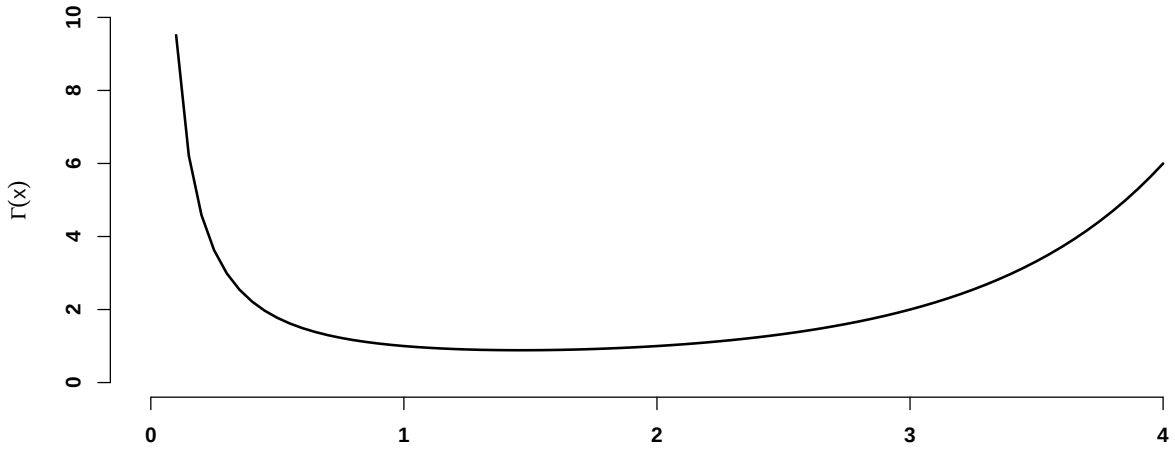



```r
q <- c(0.90, 0.95, 0.99)                                    # Quantile
round(qgumbel(q, lhat, dhat),0)
## [1]  875  915 1005

m     <- c(10, 20, 30); gamma <- 0.57722                    # Vorhersage
prd <- lhat + (gamma + log(m)) * dhat
round(prd, 0)
## [1] 910 948 971
```

5.4.10 Gammaverteilung

Gammafunktion

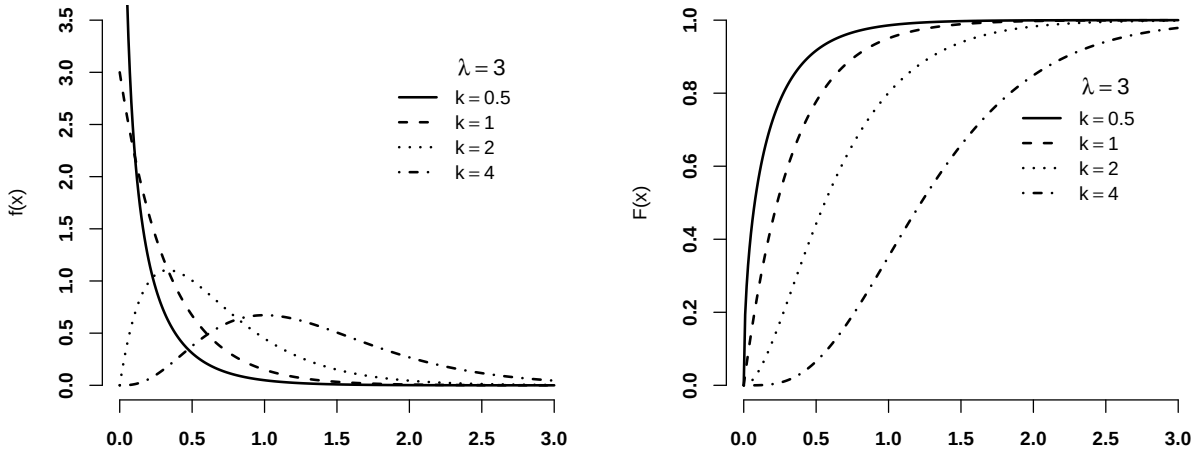
```
x <- seq(0.1, 4, by=0.05)                                    # Gammafunktion
y <- gamma(x)

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=12)
plot(x, y, type="l", xlab=" ", ylab=expression(Gamma(x)), ylim=c(0,10), xlim=c(0, 4))
```



- Gammaverteilung

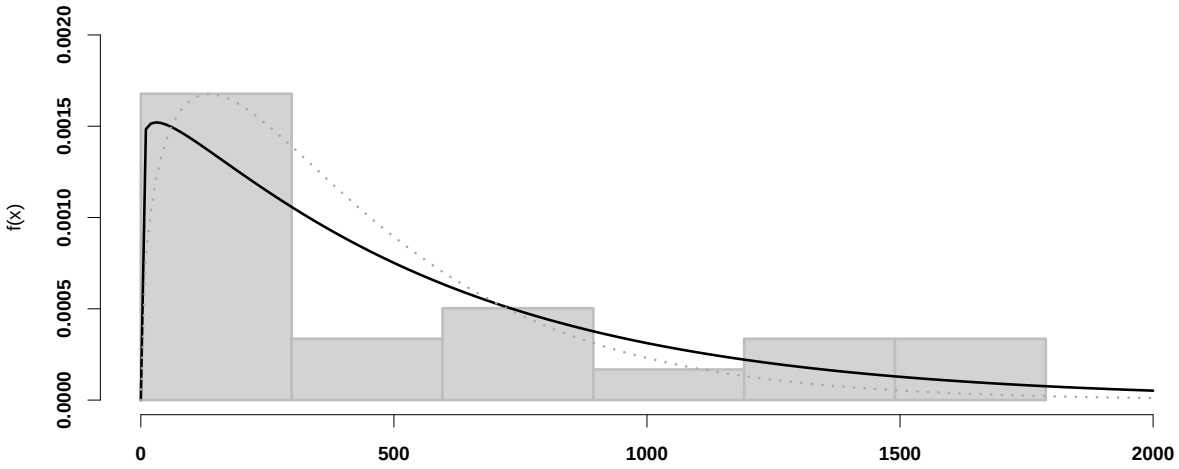
```
x <- seq(0, 3, by=0.01)
fx1 <- dgamma(x, 0.5, rate = 3)
fx2 <- dgamma(x, 1, rate = 3)
fx3 <- dgamma(x, 2, rate = 3)
fx4 <- dgamma(x, 4, rate = 3)

Fx1 <- pgamma(x, 0.5, rate = 3)
Fx2 <- pgamma(x, 1, rate = 3)
Fx3 <- pgamma(x, 2, rate = 3)
Fx4 <- pgamma(x, 4, rate = 3)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, fx1, type="l", ylim=c(0, 3.5), xlim=c(0, 3),
     ylab="f(x)", xlab=" ",  lwd=2, lty=1)
lines(x, fx2, type="l", lwd=2, lty=2)
lines(x, fx3, type="l", lwd=2, lty=3)
lines(x, fx4, type="l", lwd=2, lty=4)
legend(1.8, 3, bty="n", c(expression(k==0.5), expression(k==1),
                          expression(k==2),expression(k==4)), lty=c(1,2,3,4), cex=1)
text(2.5,3.05, expression(lambda==3), cex=1.2)
```

```r
plot(x, Fx1, type="l", ylim=c(0, 1), xlim=c(0, 3),
     ylab="F(x)", xlab=" ", lwd=2, lty=1)
lines(x, Fx2, type="l", lwd=2, lty=2)
lines(x, Fx3, type="l", lwd=2, lty=3)
lines(x, Fx4, type="l", lwd=2, lty=4)
legend(1.8, 0.8, bty="n", c(expression(k==0.5), expression(k==1),
                            expression(k==2),expression(k==4)), lty=c(1,2,3,4), cex=1)
text(2.5, 0.82, expression(lambda==3), cex=1.2)
```



Beispiel zu Haltbarkeit von Druckgefäßen

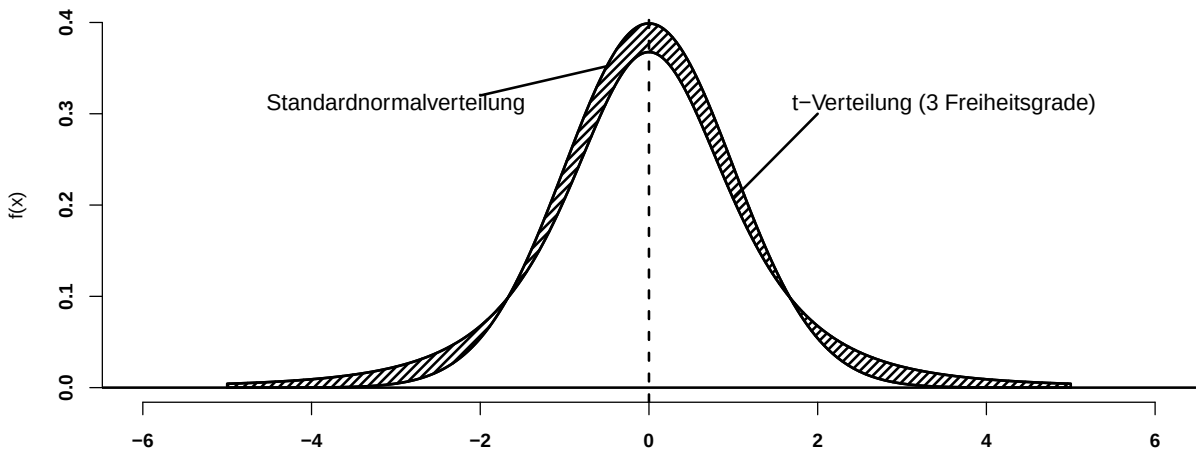
```r
time <- c(274.0,   1.7,  871.0, 1311.0, 236.0,  458.0,  54.9, 1787.0,
          0.75, 776.0,    28.5,  20.8, 363.0, 1661.0, 828.0,  290.0,
          175.0,  970.0, 1278.0, 126.0)
m.t  <- mean(time); n <- length(time)
k.hat <- (n * (m.t)^2) / sum((time - m.t)^2); round(k.hat, 4)
## [1] 1.0559
l.hat <- (n *  m.t   ) / sum((time - m.t)^2); round(l.hat, 4)
## [1] 0.0018

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
hist(time, breaks=c(0, 298, 596, 894, 1192, 1490, 1788), lwd=0.5,
     main=" ", xlab=" ", ylab="f(x)", col="lightgrey", border="grey",
     xlim=c(0,2000), ylim=c(0,0.002), freq=FALSE)

x <- seq(0, 2000, by=10)
f <- dgamma(x, k.hat,  rate = l.hat)
lines(x, f)
f <- dgamma(x, 1.45,  rate = 1/300)
lines(x, f, lty=3, col="darkgrey")
```


f(x)
0.0000
0.0005
0.0010
0.0015
0.0020
0
500
1000
1500
2000

5.5 Testverteilungen

5.5.1 Student-Verteilung (t-Verteilung)

```
x <- seq(-5, +5, by=0.01)
fn <- dnorm(x, mean=0, sd=1); ft <- dt(x, 3)

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, fn, type="l", ylim=c(0, 0.4), xlim=c(-6, +6),
     ylab="f(x)", xlab=" ", lwd=2)
lines(x, ft, type="l", lwd=2)
abline(h=0)
abline(v=0, lty=2)
x1 <- c(seq(-5,5,by=0.01), seq(+5,-5,by=-0.01))
f1 <- c(dnorm(seq(-5,5,by=0.01), mean=0, sd=1),
        dt(seq(+5,-5,by=-0.01), 3))
polygon(x1,f1, density =20, angle = 45)
lines(c(1,2),c(dt(1,3),0.3), lwd=2)
text(3.5,0.31, "t-Verteilung (3 Freiheitsgrade)", cex=1.2)
lines(c(-0.5,-2),c(dnorm(-0.5,mean=0,sd=1),0.32), lwd=2)
text(-3, 0.31, "Standardnormalverteilung", cex=1.2)
```




```
x <- seq(-6, +6, by=0.01)
ft1 <- dt(x, 1); Ft1 <- pt(x, 1)
ft2 <- dt(x, 3); Ft2 <- pt(x, 3)
ft3 <- dt(x, 8); Ft3 <- pt(x, 8)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, ft1, type="l", ylim=c(0, 0.4), xlim=c(-6, +6),
     ylab="f(x)", xlab=" ", lwd=2, lty=3)
abline(v=0, lty=2, col="grey")
```

```
lines(x, ft2, type="l", lwd=2, lty=2)
lines(x, ft3, type="l", lwd=2, lty=1)
legend(1.5, 0.4, bty="n", c("FG=1","FG=3","FG=8"),
       lty=c(3,2,1), cex=0.9)
plot(x, Ft1, type="l", ylim=c(0, 1), xlim=c(-6, +6),
     ylab="f(x)", xlab=" ", lwd=2, lty=3)
abline(v=0, lty=2, col="grey")
lines(x, Ft2, type="l", lwd=2, lty=2)
lines(x, Ft3, type="l", lwd=2, lty=1)
legend(1.5, 0.3, bty="n", c("FG=1","FG=3","FG=8"), lty=c(3,2,1), cex=0.9)
```

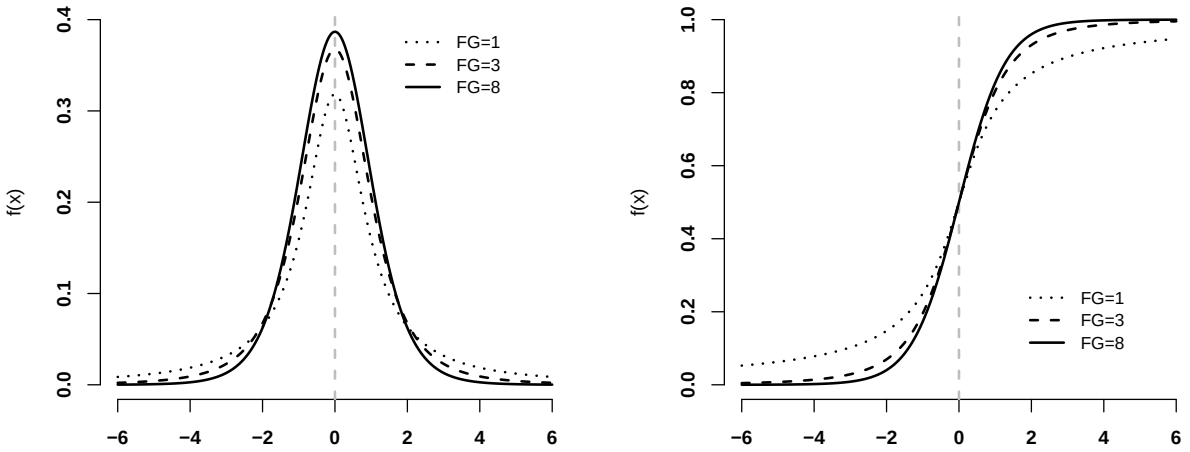


Tabelle zur t-Verteilung

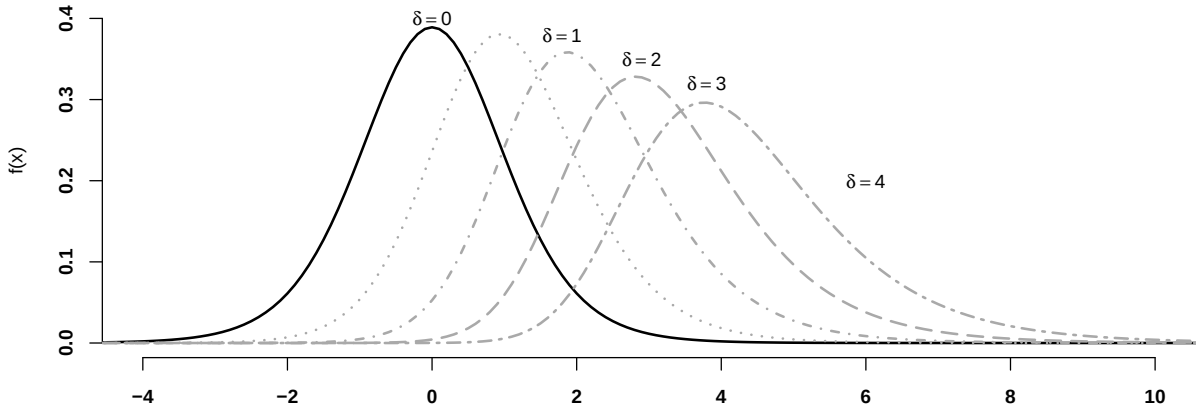
```
fg    <- c(seq(1, 20, by=1), seq(22, 50, by =2), seq(55, 100, by=5), 250, 500, 1000)
alpha <- c(0.99, 0.975, 0.95, 0.90)

tab <- matrix(NA, nrow=length(fg), ncol=length(alpha)+1)
tab[,1] <- fg
for (i in 2:5) tab[,i]=qt(alpha[i-1], fg)
colnames(tab) <- c("FG","0.99","0.975","0.95","0.90")
as.data.frame(tab[1:20,])
```

FG	0.99	0.975	0.95	0.90
1	31.820516	12.706205	6.313752	3.077684
2	6.964557	4.302653	2.919986	1.885618
3	4.540703	3.182446	2.353363	1.637744
4	3.746947	2.776445	2.131847	1.533206
5	3.364930	2.570582	2.015048	1.475884
6	3.142668	2.446912	1.943180	1.439756
7	2.997952	2.364624	1.894579	1.414924
8	2.896459	2.306004	1.859548	1.396815

FG	0.99	0.975	0.95	0.90
9	2.821438	2.262157	1.833113	1.383029
10	2.763769	2.228139	1.812461	1.372184
11	2.718079	2.200985	1.795885	1.363430
12	2.680998	2.178813	1.782288	1.356217
13	2.650309	2.160369	1.770933	1.350171
14	2.624494	2.144787	1.761310	1.345030
15	2.602480	2.131449	1.753050	1.340606
16	2.583487	2.119905	1.745884	1.336757
17	2.566934	2.109816	1.739607	1.333379
18	2.552380	2.100922	1.734064	1.330391
19	2.539483	2.093024	1.729133	1.327728
20	2.527977	2.085963	1.724718	1.325341

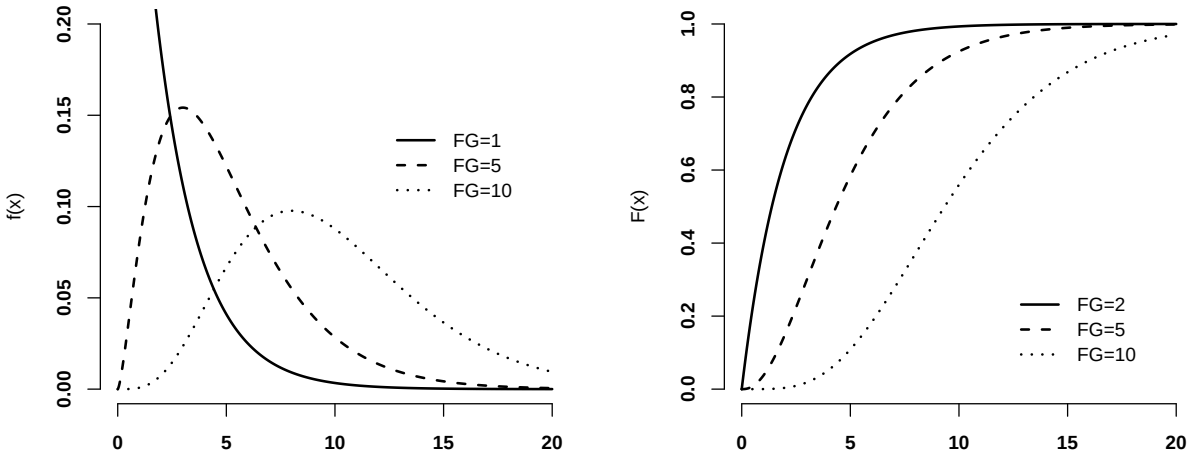
5.5.1.1 Nichtzentrale t-Verteilung

```
par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
x <- seq(-6, 12, by=0.1); d <- c(1, 2, 3, 4)
f <- dt(x, 10, ncp=0, log = FALSE)
plot(x, f, type="l", ylim=c(0, 0.45), xlim=c(-4, 10),
                              ylab="f(x)", xlab=" ", lwd=2)
text(0, 0.4, expression(delta == 0))
text(1.8, 0.38, expression(delta == 1))
text(2.9, 0.35, expression(delta == 2))
text(3.8, 0.32, expression(delta == 3))
text(6, 0.2, expression(delta == 4))
lt <- c(3, 4, 5, 6)
for (i in 1:4) {
f <- dt(x, 10, ncp=d[i], log = FALSE)
lines(x, f, type="l", lwd=2, lty=lt[i], col="darkgrey") }
```



5.5.2 Chiquadrat-Verteilung

```
x <- seq(0, +20, by=0.01)
fx1 <- dchisq(x, 2);  Fx1 <- pchisq(x, 2)
fx2 <- dchisq(x, 5);  Fx2 <- pchisq(x, 5)
fx3 <- dchisq(x, 10); Fx3 <- pchisq(x, 10)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, fx1, type="l", ylim=c(0, 0.2), xlim=c(0, 20),
            ylab="f(x)", xlab=" ", lwd=2, lty=1)
lines(x, fx2, type="l", lwd=2, lty=2)
lines(x, fx3, type="l", lwd=2, lty=3)
legend(12, 0.15, bty="n", c("FG=1","FG=5","FG=10"), lty=c(1,2,3), cex=1)

plot(x, Fx1, type="l", ylim=c(0, 1), xlim=c(0, 20),
            ylab="F(x)", xlab=" ", lwd=2, lty=1)
lines(x, Fx2, type="l", lwd=2, lty=2)
lines(x, Fx3, type="l", lwd=2, lty=3)
legend(12, 0.3, bty="n", c("FG=2","FG=5","FG=10"), lty=c(1,2,3), cex=1)
```




```
pchisq(2, 5, lower.tail = TRUE)
## [1] 0.150855

pchisq( 3.841458, 1, lower.tail=FALSE)
## [1] 0.05000002
```

Tabelle zur Chiquadrat-Verteilung

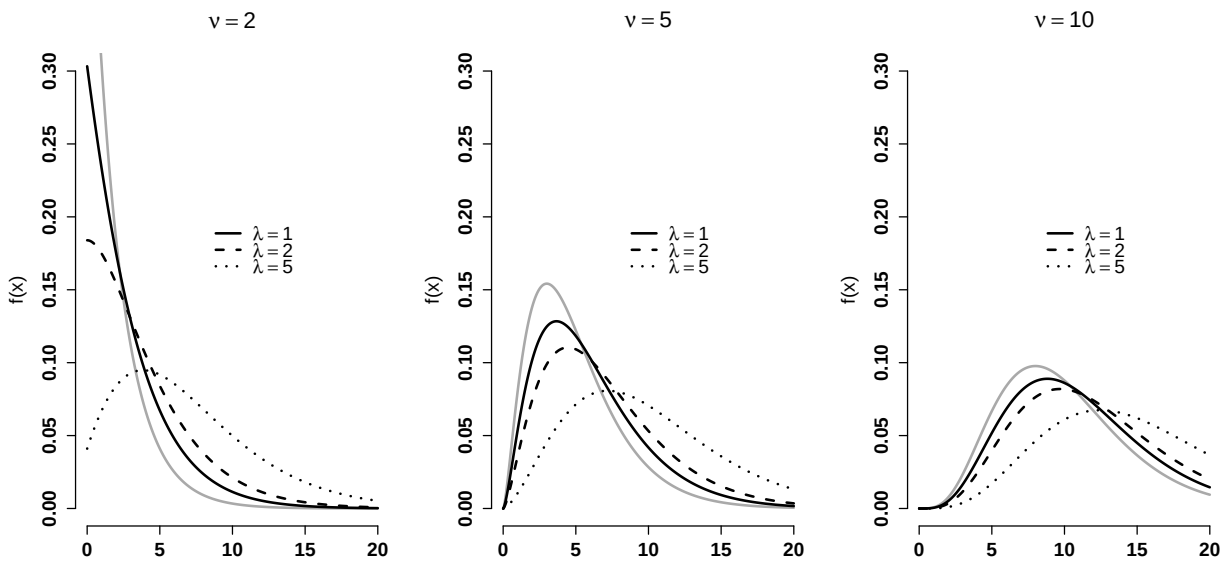
```
fg    <- c(seq(1, 20, by=1), seq(22, 50, by =2),
           seq(55, 100, by=5), 250, 500, 1000)
alpha <- c(0.01, 0.025, 0.05, 0.10, 0.90, 0.95, 0.975, 0.99)
```

```
tab <- matrix(NA, nrow=length(fg), ncol=length(alpha)+1)
tab[,1] <- fg
for (i in 2:9) tab[,i]=qchisq(alpha[i-1], fg)
colnames(tab) <- c("FG","0.01","0.025","0.05","0.10","0.90","0.95","0.975","0.99")
as.data.frame(tab[1:20,])
```

FG	0.01	0.025	0.05	0.10	0.90	0.95	0.975	0.99
1	0.0001571	0.0009821	0.0039321	0.0157908	2.705544	3.841459	5.023886	6.634897
2	0.0201007	0.0506356	0.1025866	0.2107210	4.605170	5.991465	7.377759	9.210340
3	0.1148318	0.2157953	0.3518463	0.5843744	6.251389	7.814728	9.348404	11.344867
4	0.2971095	0.4844186	0.7107230	1.0636232	7.779440	9.487729	11.143287	13.276704
5	0.5542981	0.8312116	1.1454762	1.6103080	9.236357	11.070498	12.832502	15.086272
6	0.8720903	1.2373442	1.6353829	2.2041307	10.644641	12.591587	14.449375	16.811894
7	1.2390423	1.6898692	2.1673499	2.8331069	12.017037	14.067140	16.012764	18.475307
8	1.6464974	2.1797307	2.7326368	3.4895391	13.361566	15.507313	17.534546	20.090235
9	2.0879007	2.7003895	3.3251128	4.1681590	14.683657	16.918978	19.022768	21.665994
10	2.5582122	3.2469728	3.9402991	4.8651821	15.987179	18.307038	20.483177	23.209251
11	3.0534841	3.8157483	4.5748131	5.5777848	17.275008	19.675138	21.920049	24.724970
12	3.5705690	4.4037885	5.2260295	6.3037961	18.549348	21.026070	23.336664	26.216967
13	4.1069155	5.0087505	5.8918643	7.0415046	19.811929	22.362033	24.735605	27.688250
14	4.6604251	5.6287261	6.5706314	7.7895336	21.064144	23.684791	26.118948	29.141238
15	5.2293489	6.2621378	7.2609439	8.5467562	22.307130	24.995790	27.488393	30.577914
16	5.8122125	6.9076644	7.9616456	9.3122364	23.541829	26.296228	28.845351	31.999927
17	6.4077598	7.5641864	8.6717602	10.0851863	24.769035	27.587112	30.191009	33.408664
18	7.0149109	8.2307462	9.3904551	10.8649361	25.989423	28.869299	31.526378	34.805306
19	7.6327296	8.9065165	10.1170131	11.6509100	27.203571	30.143527	32.852327	36.190869
20	8.2603983	9.5907774	10.8508114	12.4426092	28.411981	31.410433	34.169607	37.566235

5.5.2.1 Nichtzentrale Chiquadrat-Verteilung

```
nu  <- c(2, 5, 10)
tt  <- c(expression(nu == 2), expression(nu == 5), expression(nu == 10))
x   <- seq(0, +20, by=0.01)

par(mfrow=c(1,3), lwd=2, font.axis=2, bty="n", ps=16)
for (k in 1:3) {
plot(x, dchisq(x, nu[k]), type="l", ylim=c(0, 0.3), xlim=c(0, 20),
          main=tt[k], col="darkgray", ylab="f(x)", xlab=" ", lwd=2)
lt <- c(1,2,3); lambda <- c(1, 2, 5)
l1 <- expression(lambda == 1)
l2 <- expression(lambda == 2)
l3 <- expression(lambda == 5)
legend(8, 0.2, bty="n", c(l1, l2, l3), lty=c(1,2,3), cex=1)
for (i in 1:3) lines(x, dchisq(x, nu[k], ncp=lambda[i]),
                     type="l", lwd=2, lty=lt[i], col="black")
}
```


ν = 2
ν = 5
ν = 10
f(x)
λ = 1
λ = 2
λ = 5
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
5
10
15
20

5.5.3 Fischer-Verteilung (F-Verteilung)

```
x <- seq(0, 4, by=0.01)
fx1 <- df(x, 2, 5);  Fx1 <- pf(x, 2, 5)
fx2 <- df(x, 10, 10);  Fx2 <- pf(x, 10, 10)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x, fx1, type="l", ylim=c(0, 1), xlim=c(0, 4),
            ylab="f(x)", xlab=" ", lwd=2, lty=1)
lines(x, fx2, type="l", lwd=2, lty=2)
legend(2, 0.4, c("FG=(2, 5)","FG=(10, 10)"), bty="n",
       lty=c(1,2), cex=0.8)
plot(x, Fx1, type="l", ylim=c(0, 1), xlim=c(0, 4),
            ylab="F(x)", xlab=" ", lwd=2, lty=1)
lines(x, Fx2, type="l", lwd=2, lty=2)
legend(2, 0.4, c("FG=(2, 5)","FG=(10, 10)"), bty="n",
       lty=c(1,2), cex=0.8)
```

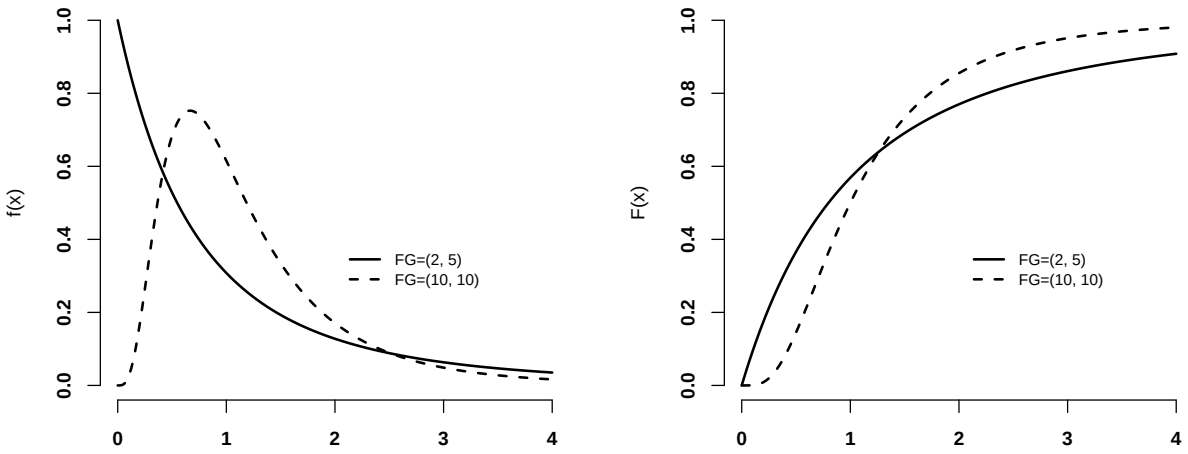


Tabelle zur F-Verteilung

```
fg1 <- c(seq(1, 10, by=1), seq(12, 20, by =2), 25, 30, 40, 50, 100)
fg2 <- c(seq(1, 10, by=1), seq(12, 20, by =2), 25, 30, 40, 50, 100)

print("Tabelle zu 0.95 Quantilen"); alpha <- 0.95
## [1] "Tabelle zu 0.95 Quantilen"

tab1 <- matrix(NA, nrow=21, ncol=11)
tab1[1,] <- c(NA, fg1[1:10])
tab1[,1] <- c(NA, fg2)
for (i in 1:20) {
    for (j in 1:10) tab1[i+1,j+1]=qf(alpha, fg1[i], fg2[j]) }
```

```
colnames(tab1) <- c("m/n",1:10)
as.data.frame(round(tab1, 2))
```

m/n	1	2	3	4	5	6	7	8	9	10
NA	1.00	2.00	3.00	4.00	5.00	6.00	7.00	8.00	9.00	10.00
1	161.45	18.51	10.13	7.71	6.61	5.99	5.59	5.32	5.12	4.96
2	199.50	19.00	9.55	6.94	5.79	5.14	4.74	4.46	4.26	4.10
3	215.71	19.16	9.28	6.59	5.41	4.76	4.35	4.07	3.86	3.71
4	224.58	19.25	9.12	6.39	5.19	4.53	4.12	3.84	3.63	3.48
5	230.16	19.30	9.01	6.26	5.05	4.39	3.97	3.69	3.48	3.33
6	233.99	19.33	8.94	6.16	4.95	4.28	3.87	3.58	3.37	3.22
7	236.77	19.35	8.89	6.09	4.88	4.21	3.79	3.50	3.29	3.14
8	238.88	19.37	8.85	6.04	4.82	4.15	3.73	3.44	3.23	3.07
9	240.54	19.38	8.81	6.00	4.77	4.10	3.68	3.39	3.18	3.02
10	241.88	19.40	8.79	5.96	4.74	4.06	3.64	3.35	3.14	2.98
12	243.91	19.41	8.74	5.91	4.68	4.00	3.57	3.28	3.07	2.91
14	245.36	19.42	8.71	5.87	4.64	3.96	3.53	3.24	3.03	2.86
16	246.46	19.43	8.69	5.84	4.60	3.92	3.49	3.20	2.99	2.83
18	247.32	19.44	8.67	5.82	4.58	3.90	3.47	3.17	2.96	2.80
20	248.01	19.45	8.66	5.80	4.56	3.87	3.44	3.15	2.94	2.77
25	249.26	19.46	8.63	5.77	4.52	3.83	3.40	3.11	2.89	2.73
30	250.10	19.46	8.62	5.75	4.50	3.81	3.38	3.08	2.86	2.70
40	251.14	19.47	8.59	5.72	4.46	3.77	3.34	3.04	2.83	2.66
50	251.77	19.48	8.58	5.70	4.44	3.75	3.32	3.02	2.80	2.64
100	253.04	19.49	8.55	5.66	4.41	3.71	3.27	2.97	2.76	2.59

```
print("Tabelle zu 0.95 Quantilen"); alpha <- 0.95
## [1] "Tabelle zu 0.95 Quantilen"

tab2 <- matrix(NA, nrow=21, ncol=11)
tab2[1,] <- c(NA, fg1[11:20])
tab2[,1] <- c(NA, fg2)
for (i in 1:20) {
    for (j in 1:10) tab2[i+1,j+1]=qf(alpha, fg1[i], fg2[j+10]) }
colnames(tab2) <- c("m/n",12,14,16,18,20,25,30, 40, 50, 100)
as.data.frame(round(tab2, 2))
```

m/n	12	14	16	18	20	25	30	40	50	100
NA	12.00	14.00	16.00	18.00	20.00	25.00	30.00	40.00	50.00	100.00
1	4.75	4.60	4.49	4.41	4.35	4.24	4.17	4.08	4.03	3.94
2	3.89	3.74	3.63	3.55	3.49	3.39	3.32	3.23	3.18	3.09
3	3.49	3.34	3.24	3.16	3.10	2.99	2.92	2.84	2.79	2.70
4	3.26	3.11	3.01	2.93	2.87	2.76	2.69	2.61	2.56	2.46
5	3.11	2.96	2.85	2.77	2.71	2.60	2.53	2.45	2.40	2.31
6	3.00	2.85	2.74	2.66	2.60	2.49	2.42	2.34	2.29	2.19
7	2.91	2.76	2.66	2.58	2.51	2.40	2.33	2.25	2.20	2.10
8	2.85	2.70	2.59	2.51	2.45	2.34	2.27	2.18	2.13	2.03
9	2.80	2.65	2.54	2.46	2.39	2.28	2.21	2.12	2.07	1.97
10	2.75	2.60	2.49	2.41	2.35	2.24	2.16	2.08	2.03	1.93
12	2.69	2.53	2.42	2.34	2.28	2.16	2.09	2.00	1.95	1.85

m/n	12	14	16	18	20	25	30	40	50	100
14	2.64	2.48	2.37	2.29	2.22	2.11	2.04	1.95	1.89	1.79
16	2.60	2.44	2.33	2.25	2.18	2.07	1.99	1.90	1.85	1.75
18	2.57	2.41	2.30	2.22	2.15	2.04	1.96	1.87	1.81	1.71
20	2.54	2.39	2.28	2.19	2.12	2.01	1.93	1.84	1.78	1.68
25	2.50	2.34	2.23	2.14	2.07	1.96	1.88	1.78	1.73	1.62
30	2.47	2.31	2.19	2.11	2.04	1.92	1.84	1.74	1.69	1.57
40	2.43	2.27	2.15	2.06	1.99	1.87	1.79	1.69	1.63	1.52
50	2.40	2.24	2.12	2.04	1.97	1.84	1.76	1.66	1.60	1.48
100	2.35	2.19	2.07	1.98	1.91	1.78	1.70	1.59	1.52	1.39

```
print("Tabelle zu 0.975 Quantilen"); alpha <- 0.975
## [1] "Tabelle zu 0.975 Quantilen"

tab3 <- matrix(NA, nrow=21, ncol=11)
tab3[1,] <- c(NA, fg1[1:10])
tab3[,1] <- c(NA, fg2)
for (i in 1:20) {
    for (j in 1:10) tab3[i+1,j+1]=qf(alpha, fg1[i], fg2[j]) }
colnames(tab3) <- c("m/n",1:10)
as.data.frame(round(tab3, 2))
```

m/n	1	2	3	4	5	6	7	8	9	10
NA	1.00	2.00	3.00	4.00	5.00	6.00	7.00	8.00	9.00	10.00
1	647.79	38.51	17.44	12.22	10.01	8.81	8.07	7.57	7.21	6.94
2	799.50	39.00	16.04	10.65	8.43	7.26	6.54	6.06	5.71	5.46
3	864.16	39.17	15.44	9.98	7.76	6.60	5.89	5.42	5.08	4.83
4	899.58	39.25	15.10	9.60	7.39	6.23	5.52	5.05	4.72	4.47
5	921.85	39.30	14.88	9.36	7.15	5.99	5.29	4.82	4.48	4.24
6	937.11	39.33	14.73	9.20	6.98	5.82	5.12	4.65	4.32	4.07
7	948.22	39.36	14.62	9.07	6.85	5.70	4.99	4.53	4.20	3.95
8	956.66	39.37	14.54	8.98	6.76	5.60	4.90	4.43	4.10	3.85
9	963.28	39.39	14.47	8.90	6.68	5.52	4.82	4.36	4.03	3.78
10	968.63	39.40	14.42	8.84	6.62	5.46	4.76	4.30	3.96	3.72
12	976.71	39.41	14.34	8.75	6.52	5.37	4.67	4.20	3.87	3.62
14	982.53	39.43	14.28	8.68	6.46	5.30	4.60	4.13	3.80	3.55
16	986.92	39.44	14.23	8.63	6.40	5.24	4.54	4.08	3.74	3.50
18	990.35	39.44	14.20	8.59	6.36	5.20	4.50	4.03	3.70	3.45
20	993.10	39.45	14.17	8.56	6.33	5.17	4.47	4.00	3.67	3.42
25	998.08	39.46	14.12	8.50	6.27	5.11	4.40	3.94	3.60	3.35
30	1001.41	39.46	14.08	8.46	6.23	5.07	4.36	3.89	3.56	3.31
40	1005.60	39.47	14.04	8.41	6.18	5.01	4.31	3.84	3.51	3.26
50	1008.12	39.48	14.01	8.38	6.14	4.98	4.28	3.81	3.47	3.22
100	1013.17	39.49	13.96	8.32	6.08	4.92	4.21	3.74	3.40	3.15

```
print("Tabelle zu 0.975 Quantilen"); alpha <- 0.975
## [1] "Tabelle zu 0.975 Quantilen"

tab4<- matrix(NA, nrow=21, ncol=11)
```

```
tab4[1,] <- c(NA, fg1[11:20])
tab4[,1] <- c(NA, fg2)
for (i in 1:20) {
    for (j in 1:10) tab4[i+1,j+1]=qf(alpha, fg1[i], fg2[j+10]) }
colnames(tab4) <- c("m/n",12,14,16,18,20,25,30, 40, 50, 100)
as.data.frame(round(tab4, 2))
```

m/n	12	14	16	18	20	25	30	40	50	100
NA	12.00	14.00	16.00	18.00	20.00	25.00	30.00	40.00	50.00	100.00
1	6.55	6.30	6.12	5.98	5.87	5.69	5.57	5.42	5.34	5.18
2	5.10	4.86	4.69	4.56	4.46	4.29	4.18	4.05	3.97	3.83
3	4.47	4.24	4.08	3.95	3.86	3.69	3.59	3.46	3.39	3.25
4	4.12	3.89	3.73	3.61	3.51	3.35	3.25	3.13	3.05	2.92
5	3.89	3.66	3.50	3.38	3.29	3.13	3.03	2.90	2.83	2.70
6	3.73	3.50	3.34	3.22	3.13	2.97	2.87	2.74	2.67	2.54
7	3.61	3.38	3.22	3.10	3.01	2.85	2.75	2.62	2.55	2.42
8	3.51	3.29	3.12	3.01	2.91	2.75	2.65	2.53	2.46	2.32
9	3.44	3.21	3.05	2.93	2.84	2.68	2.57	2.45	2.38	2.24
10	3.37	3.15	2.99	2.87	2.77	2.61	2.51	2.39	2.32	2.18
12	3.28	3.05	2.89	2.77	2.68	2.51	2.41	2.29	2.22	2.08
14	3.21	2.98	2.82	2.70	2.60	2.44	2.34	2.21	2.14	2.00
16	3.15	2.92	2.76	2.64	2.55	2.38	2.28	2.15	2.08	1.94
18	3.11	2.88	2.72	2.60	2.50	2.34	2.23	2.11	2.03	1.89
20	3.07	2.84	2.68	2.56	2.46	2.30	2.20	2.07	1.99	1.85
25	3.01	2.78	2.61	2.49	2.40	2.23	2.12	1.99	1.92	1.77
30	2.96	2.73	2.57	2.44	2.35	2.18	2.07	1.94	1.87	1.71
40	2.91	2.67	2.51	2.38	2.29	2.12	2.01	1.88	1.80	1.64
50	2.87	2.64	2.47	2.35	2.25	2.08	1.97	1.83	1.75	1.59
100	2.80	2.56	2.40	2.27	2.17	2.00	1.88	1.74	1.66	1.48

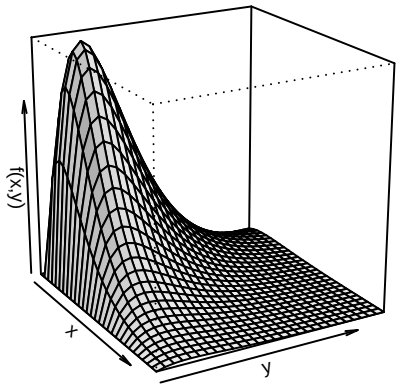
5.5.4.1 Interpolation von Zahlenwerten der F-Verteilung

```
approx(x=c(2,7), y=c(2,3), xout=c(5,6), method="linear")
## $x
## [1] 5 6
##
## $y
## [1] 2.6 2.8

approx(x=c(50, 100), y=c(4.03, 3.94), xout=70, method="linear")$y
## [1] 3.994
```

5.6 Verteilung zweidimensionaler Zufallsvariablen

Beispiel Teenager Allüren:

```
f  <- function(x, y) { x*y*exp(-(x+y)) }
x  <- seq(0, 7, by=0.25)
y  <- seq(0, 7, by=0.25)
z <- outer(x, y, f)

par(mfrow=c(1,1), lwd=1.5, font.axis=2, bty="n", ps=11)
persp(x, y, z, theta = 60, phi = 20, r=10, shade=0.1,
      xlab="x", ylab="y", zlab="f(x,y)")
```

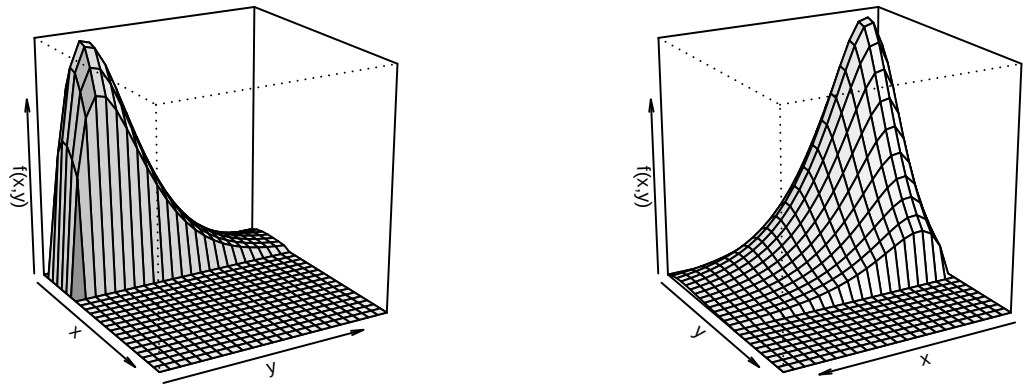


5.6.2 Randverteilungen und Unabhängigkeit

5.6.2.1 Bedingte Verteilung und Unabhängigkeit

Beispiel Teenager Allüren:

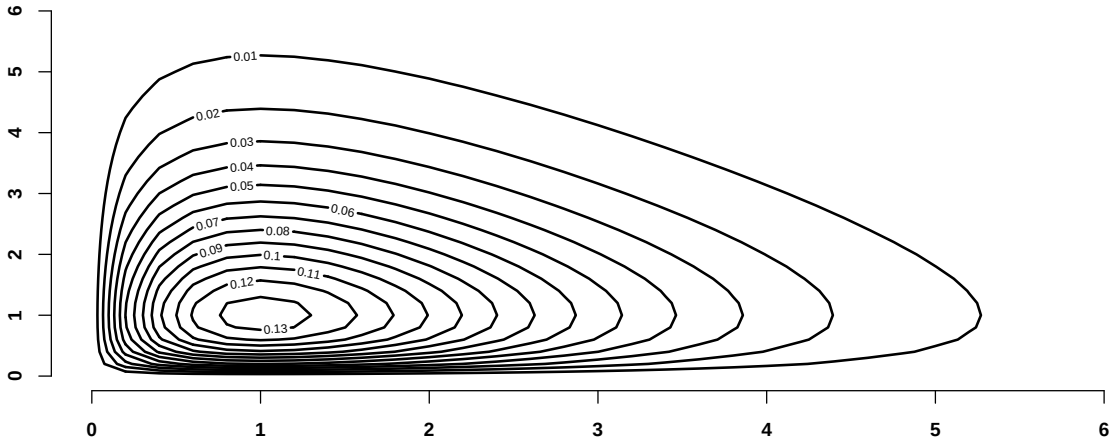
```
x  <- seq(0, 7, by=0.3)
y  <- seq(0, 7, by=0.3)

par(mfrow=c(1,2), lwd=1.5, font.axis=2, bty="n", ps=11)
f <- function(x, y) { ifelse(x>2, 0, x*y*exp(-(x+y))) }
z <- outer(x, y, f)
persp(x, y, z, theta = 60, phi = 20, r=10, shade=0.1,
            xlab="x", ylab="y", zlab="f(x,y)")
f <- function(x, y) { ifelse(y>4, 0, x*y*exp(-(x+y))) }
z <- outer(x, y, f)
persp(x, y, z, theta = 150, phi = 20, r=10, shade=0.1,
            xlab="x", ylab="y", zlab="f(x,y)")
```



Korrelationskoeffizient

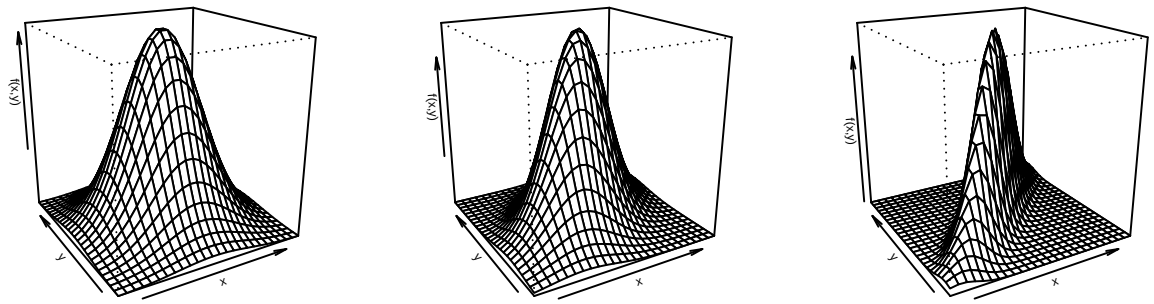
Höhenlinien zum Beispiel Teenager Allüren:

```
f     <- function(x, y) { x*y*exp(-(x+y)) }
x  <- seq(0, 7, by=0.2)
y  <- seq(0, 7, by=0.2)
z <- outer(x, y, f)

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
contour(x, y , z,  nlevels = 10, cex=2,
      xlim=c(0,6), ylim=c(0,6), xlab=" ", ylab=" ")
```



Zweidimensionale Normalverteilung

```
f <- function(x, y, rho) {
  t.x   <- (x-m.x)/s.x
  t.y   <- (y-m.y)/s.y
  rho.h <- 1-rho^2
  (1/(2*pi*s.x*s.y*sqrt(rho.h))) * exp(-(1/(2*rho.h)) *
                            ((t.x^2 - 2*rho*t.x*t.y + t.y^2)))
}

m.x <- 0;  s.x <- 1
m.y <- 0;  s.y <- 1
rho <- 0

l.x <- m.x-2.5*s.x;    u.x <- m.x+2.5*s.x; x   <- seq(l.x, u.x, by=0.2)
l.y <- m.y-2.5*s.y;    u.y <- m.y+2.5*s.y; y   <- seq(l.y, u.y, by=0.2)

par(mfrow=c(1,3), lwd=1.5, font.axis=2, bty="n", ps=11)
z <- outer(x, y, f, 0)
persp(x, y, z, theta = -30, phi = 20, r=4,
                      xlab="x", ylab="y", zlab="f(x,y)")
z <- outer(x, y, f, 0.5)
persp(x, y, z, theta = -30, phi = 20, r=4,
                      xlab="x", ylab="y", zlab="f(x,y)")
z <- outer(x, y, f, 0.9)
persp(x, y, z, theta = -30, phi = 20, r=4,
                      xlab="x", ylab="y", zlab="f(x,y)")
```



Höhenlinien zu drei standardisierten Normalverteilungen

```
f <- function(x, y, rho) {
  t.x   <- (x-m.x)/s.x
  t.y   <- (y-m.y)/s.y
  rho.h <- 1-rho^2
  (1/(2*pi*s.x*s.y*sqrt(rho.h))) * exp(-(1/(2*rho.h)) *
                            ((t.x^2 - 2*rho*t.x*t.y + t.y^2)))
}

l.x <- m.x-2.5*s.x;    u.x <- m.x+2.5*s.x
x   <- seq(l.x, u.x, by=0.15)
l.y <- m.y-2.5*s.y;    u.y <- m.y+2.5*s.y
y   <- seq(l.y, u.y, by=0.15)

par(mfrow=c(1,3), lwd=1.5, font.axis=2, bty="n", ps=11)
z <- outer(x, y, f, 0)
contour(x, y , z,  nlevels = 10, cex=2,
           xlim=c(-3,+3), ylim=c(-3,+3), xlab="x", ylab="y")
z <- outer(x, y, f, 0.5)
contour(x, y , z,  nlevels = 10, cex=2,
           xlim=c(-3,+3), ylim=c(-3,+3), xlab="x", ylab="y")
z <- outer(x, y, f, 0.9)
contour(x, y , z,  nlevels = 10, cex=2,
           xlim=c(-3,+3), ylim=c(-3,+3), xlab="x", ylab="y")
```