AS17: Kapitel 7.1 - 7.5 Hypothesentest

Jürgen Hedderich

2020-04-26

Contents

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

zur Druckversion

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

Aktuelle R-Umgebung zu den Beispielen

The R Project for Statistical Computing

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

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

Schweissraten : sweat

7.1 Der statistische Test

7.1.5 Wie oft wird eine wahre Nullhypothese abgelehnt?

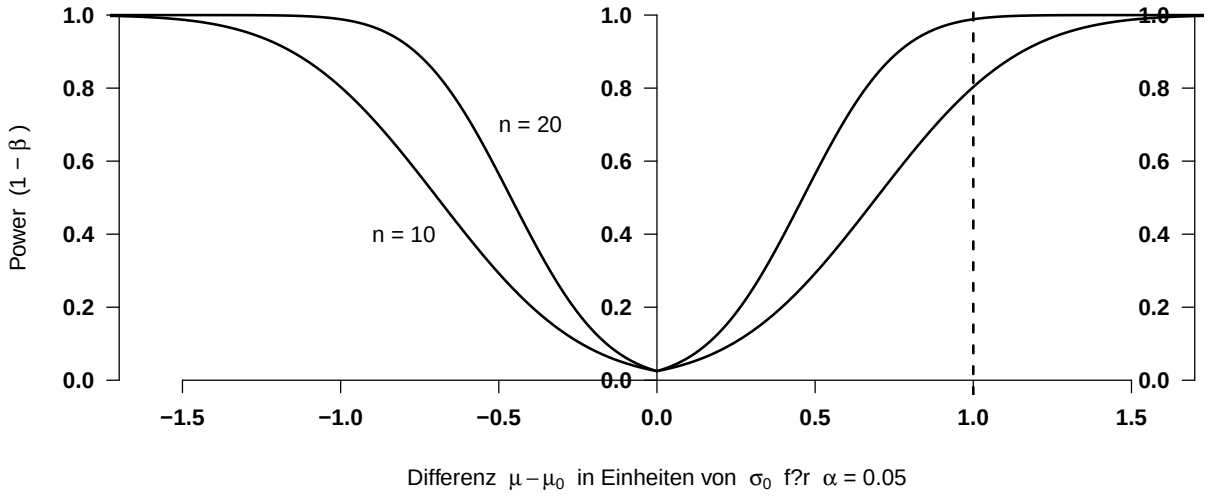
```
r <- 1:5
p <- 0.01
n <- c(22, 83, 154, 231, 310)
P <- round(pnbinom(n-r, r, p), 2); P
## [1] 0.2 0.2 0.2 0.2 0.2
```

7.1.6 Einstichproben-Gauss-Test

```
alpha <- 0.05
mu.0  <- 20
sigma <- 10
n     <- 25

l.a    <- mu.0 - qnorm(1-alpha, mean=0, sd=1)*(sigma/sqrt(n)); l.a
## [1] 16.71029
l.b    <- mu.0 + qnorm(1-alpha, mean=0, sd=1)*(sigma/sqrt(n)); l.b
## [1] 23.28971
l.c1   <- mu.0 - qnorm(1-alpha/2, mean=0, sd=1)*(sigma/sqrt(n)); l.c1
## [1] 16.08007
l.c2   <- mu.0 + qnorm(1-alpha/2, mean=0, sd=1)*(sigma/sqrt(n)); l.c2
## [1] 23.91993

m      <- 16
p.a    <- pnorm(m, mean=mu.0, sd=sigma/sqrt(n)); p.a
## [1] 0.02275013
p.b    <- 1 - pnorm(m, mean=mu.0, sd=sigma/sqrt(n)); p.b
## [1] 0.9772499
p.c    <- 2 * pnorm(m, mean=mu.0, sd=sigma/sqrt(n)); p.c
## [1] 0.04550026
```

7.1.7 Powerfunktion

```
alpha <- 0.05
nro1  <- 10; nro2 <- 20;
diff  <- seq(-4, +4, by=0.01)
stdev <- 2

result1 <- power.t.test(n = nro1, delta = diff, sd = stdev,
    sig.level = alpha, type = "one.sample", alternative = "two.sided")
result2 <- power.t.test(n = nro2, delta = diff, sd = stdev,
    sig.level = alpha, type = "one.sample", alternative = "two.sided")

par(mfcol=c(1,1), lwd=2, font.axis=2, bty="l", ps=13)
plot(diff/stdev, result1$power, type = "l", xlim=c(-1.6, +1.6),
        ylim=c(0,1), axes = FALSE,
     xlab = expression(paste("Differenz  ",mu - mu[0],
     "  in Einheiten von  ",sigma[0],"  f?r  ",alpha," = 0.05")),
     ylab = expression(paste("Power  (1 - ",beta," )")))
axis(2, pos=c(-1.7,0),las=1)    ; axis(2, pos=c(0,0), las=1)
axis(2, pos=c(+1.7,0), las=1)   ; axis(1, pos=c(0,0))
lines(diff/stdev, result2$power); abline(v=1, lty=2)
text(-0.4,0.7,"n = 20")
text(-0.8,0.4,"n = 10")
```

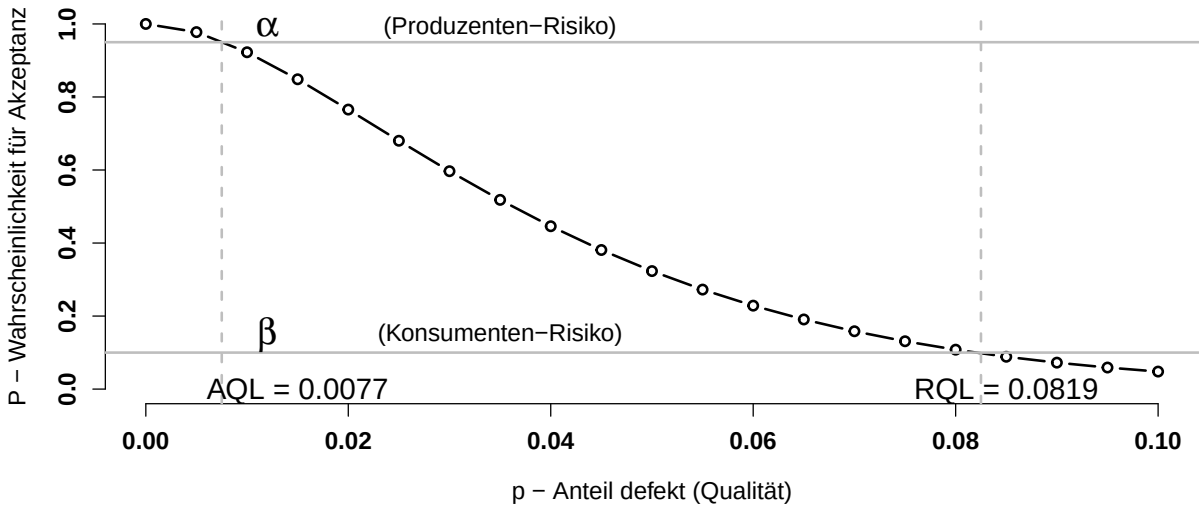



```
power.t.test(n = 10, delta = 5, sd = 5, sig.level = 0.05,
                      type = "one.sample", alternative = "two.sided")
##
##      One-sample t test power calculation
##
##               n = 10
##           delta = 5
##              sd = 5
```

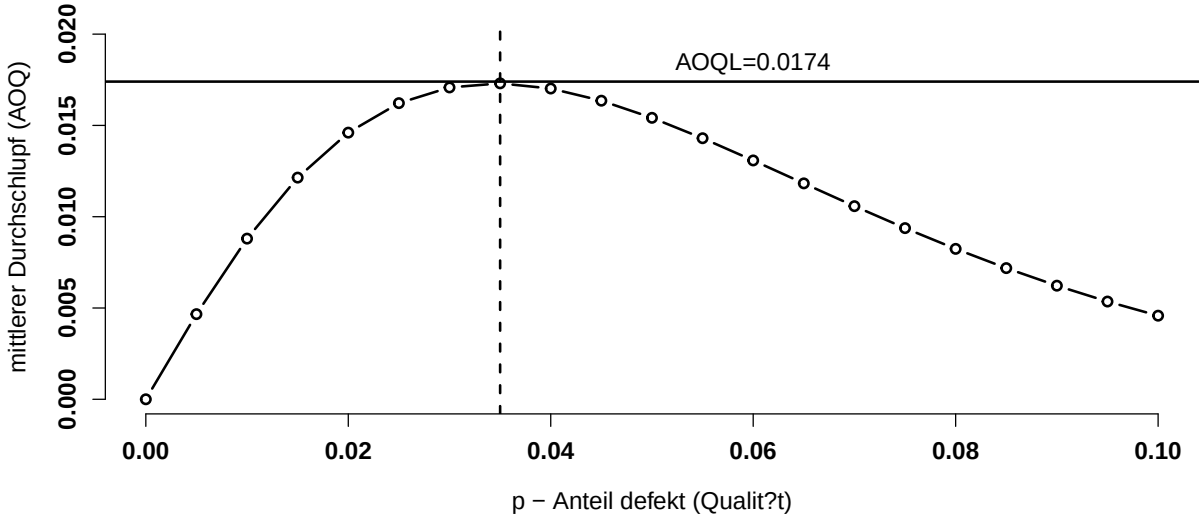
```
##       sig.level = 0.05
##           power = 0.8030962
##     alternative = two.sided

power.t.test(n = 20, delta = 5, sd = 5, sig.level = 0.05,
                       type = "one.sample", alternative = "two.sided")
##
##      One-sample t test power calculation
##
##               n = 20
##           delta = 5
##              sd = 5
##       sig.level = 0.05
##           power = 0.9885913
##     alternative = two.sided
```

7.1.8 Operationscharakteristik

```
n <- 46                                                    # Umfang der Stichprobe
c <- 1                                                     # Annahmezahl (acceptance)
N <- 1000                                                  # Umfang der Charge

                                          # Wahrscheinlichkeit defekter Einheiten (quality)
p <- seq(0, 0.1, by=0.005)
P <- pbinom(c, n, p)

pbinom(1, 46, 0.0077)                                      #  AQL für alpha = 0.05
## [1] 0.9509134
pbinom(1, 46, 0.0819)                                      #  RQL für beta  = 0.10
## [1] 0.1001856

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=14)
plot(p, P, type="b", xlim=c(0, 0.1), ylim=c(0, 1),
     xlab="p - Anteil defekt (Qualität)",
     ylab="P - Wahrscheinlichkeit für Akzeptanz")

abline(h=0.1, lty=1, col="grey")
abline(h=0.95, lty=1, col="grey")

abline(v=0.0075, lty=2, col="grey")
abline(v=0.0825, lty=2, col="grey")

text(0.012, 0.99, expression(alpha), cex=1.5)
text(0.035, 0.99, "(Produzenten-Risiko)")

text(0.012, 0.15, expression(beta), cex=1.5)
text(0.035, 0.15, "(Konsumenten-Risiko)")

text(0.015, 0.00, "AQL = 0.0077", cex=1.2)
text(0.085, 0.00, "RQL = 0.0819", cex=1.2)
```

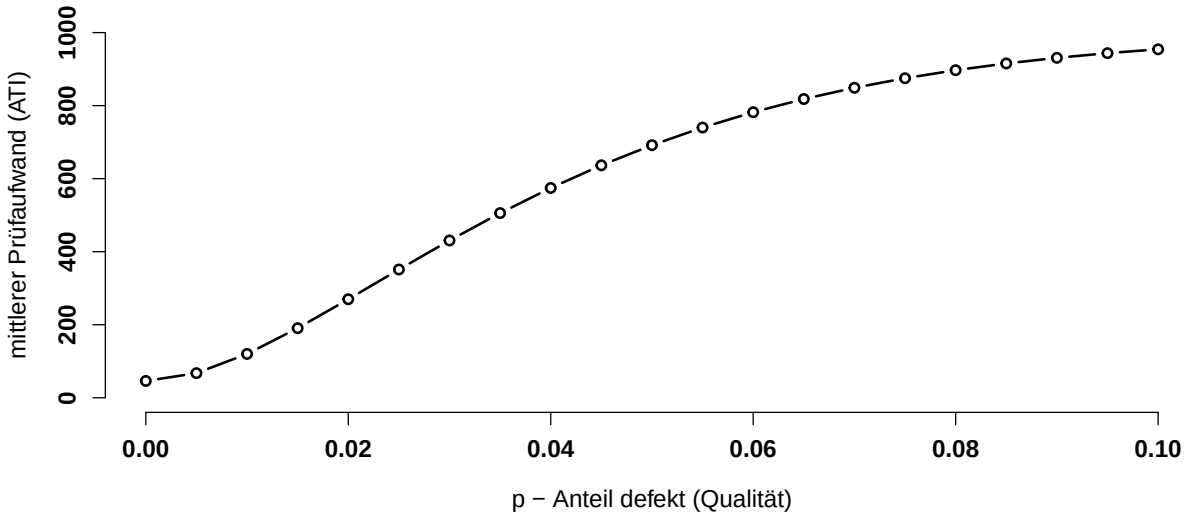



```
AOQ <- P * p * (N-n) / N

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=14, cex.axis=1)
plot(p, AOQ, type="b", xlim=c(0, 0.1), ylim=c(0, 0.02),
     xlab="p - Anteil defekt (Qualit?t)",
     ylab="mittlerer Durchschlupf (AOQ)")
abline(h=0.0174, lty=1)
abline(v=0.035, lty=2)
text(0.06, 0.0185, "AOQL=0.0174")
```




```
ATI <- n + (1-P) * (N-n)
plot(p, ATI, type="b", xlim=c(0, 0.1), ylim=c(0, 1000),
```

```
     xlab="p - Anteil defekt (Qualität)",
     ylab="mittlerer Prüfaufwand (ATI)")
```



7.2 Tests der Verteilung (Anpassungstests)

7.2.2 Überprüfung des 3. und 4. Moments

```
x <- c(rep(30, 16), 50, 70, 90, 110)
n <- length(x);  m <- mean(x)

sqrt(n)*sum((x-m)^3) / sqrt(sum((x-m)^2)^3)                    # skewness
## [1] 2.146625

n * sum((x-m)^4) / (sum((x-m)^2))^2                            # kurtosis
## [1] 6.248
```

Funktionen skewness() und kurtosis in library(e1071)

```
library(e1071)
x <- c(rep(30, 16), 50, 70, 90, 110)
skewness(x)
## [1] 1.987658
kurtosis(x)+3
## [1] 5.63882
```

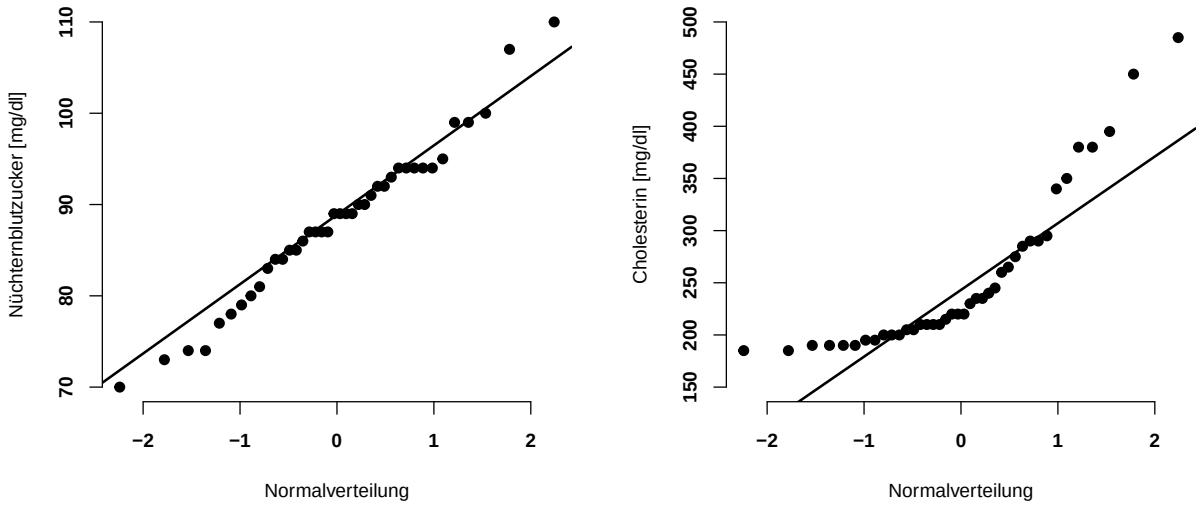
Symmetrie-Vorzeichentest:

```
x <- c(3, 5, 2, 6, 7, 7, 4, 2, 6, 12, 15, 18); sort(x)
##  [1]  2  2  3  4  5  6  6  7  7 12 15 18
n <- length(x); m <- mean(x); I <- ifelse(x<m, 1, 0)
S <- sum(I); Amin <- S; Aplus <- n - S
test <- (Amin - Aplus)^2 / n; test
## [1] 3
qchisq(0.10, 1, lower.tail=F)
## [1] 2.705543
```

7.2.3 Der Quantile-Quantile Plot

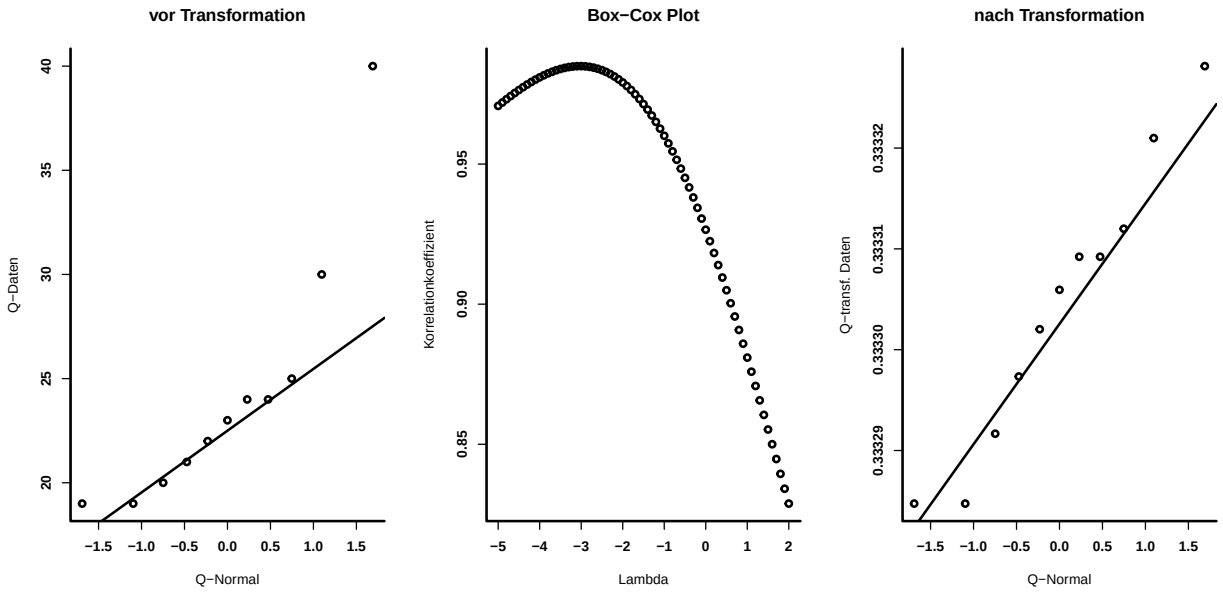
Beispiel Nüchternblutzucker und Cholesterin:

```
nblz <- c(90,  74,  94,  79, 100,  87,  87,  84,  78,  94,
          73,  99,  85,  83,  70,  84,  91,  99,  85,  89,
          80,  89,  81,  95,  89,  94,  77,  87,  89,  86,
          94, 110,  92,  92,  93,  94,  87,  90, 107,  74)

chol <- c(195, 205, 245, 190, 260, 190, 340, 195, 285, 380,
          220, 240, 235, 215, 190, 275, 205, 290, 200, 210,
          220, 265, 235, 200, 350, 220, 450, 230, 185, 295,
          380, 200, 485, 210, 185, 210, 395, 290, 190, 210)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
qqnorm(nblz, ylab="Nüchternblutzucker [mg/dl]", xlab="Normalverteilung",
       main=" ", pch=19, cex=0.9)
qqline(nblz, col = "black", lwd=2)

qqnorm(chol, ylab="Cholesterin [mg/dl]", xlab="Normalverteilung",
             ylim=c(150, 500), main= " ", pch=19, cex=0.9)
qqline(chol, col = "black", lwd=2)
```



7.2.4 Box-Cox Transformation

```
bcplot <- function(x, lambda) {
          n <- length(lambda); v <- rep(NA, n)
          for (i in 1:n) {
              if (lambda[i] == 0) y <- log(x)
              if (lambda[i] != 0) y <- (x^lambda[i] - 1)/lambda[i]
              nppl <- qqnorm(y, plot.it = FALSE); v[i]=cor(nppl$x,nppl$y)      }
          j <- which(v == max(v))
          list("cor"=v, "l.i"=lambda, "lambda"=lambda[j])    }


x <- c(20, 22, 24, 21, 19, 30, 40, 23, 24, 25, 19)
l <- seq(-5, +2, by=0.1)
lp <- bcplot(x, l); lp$lambda
## [1] -3

par(mfrow=c(1,3), lwd=2, font=2, font.axis=2, bty="l", ps=12)
qqnorm(x, main="vor Transformation",xlab="Q-Normal",ylab="Q-Daten")
qqline(x)
plot(lp$l.i, lp$cor, main="Box-Cox Plot", xlab="Lambda",
            ylab="Korrelationkoeffizient")
xtrn <- (x^lp$lambda -1)/lp$lambda
qqnorm(xtrn, main="nach Transformation",xlab="Q-Normal",
            ylab="Q-transf. Daten")
qqline(xtrn)
```

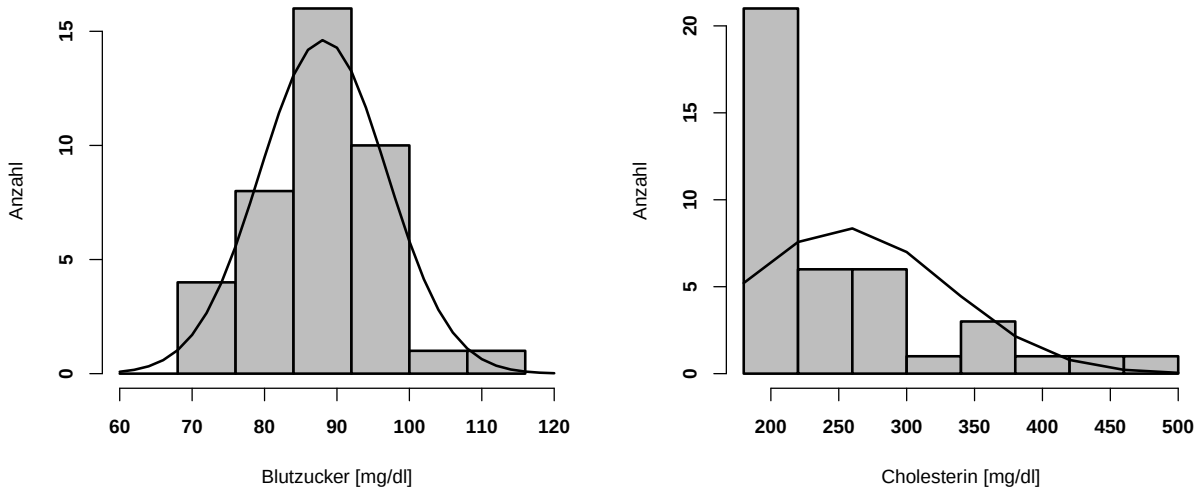


7.2.5 Der Chiquadrat-Anpassungstest

```
obs <- c(7, 16, 8, 17, 3, 9); summe <- sum(obs)
exp <- rep(summe/6, 6)

stat <- sum((obs-exp)^2/exp); stat; qchisq(0.95, 5)
## [1] 14.8
## [1] 11.0705
```

Beispiel Nüchterblutzucker und Cholesterin:

```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
breaks <- seq(60, 120, by=8);  breite <- 8
hist(nblz, breaks, plot=T, col="grey",
     main=" ", xlab="Blutzucker [mg/dl]", ylab="Anzahl",xlim=c(60,120))
x      <- seq(60, 120, by=2)
lines(x, dnorm(x, mean=mean(nblz),
               sd=sd(nblz)) * breite * length(nblz), col="black", lwd=2)

breaks <- seq(180, 500, by=40); breite <- 40
hist(chol, breaks, plot=T, col="grey",
     main=" ", xlab="Cholesterin [mg/dl]", ylab="Anzahl",xlim=c(180, 500))
x      <- seq(180, 500, by=40)
lines(x, dnorm(x, mean=mean(chol),
               sd=sd(chol)) * breite * length(chol), col="black", lwd=2)
```



Funktion pearson.test() in library(nortest)

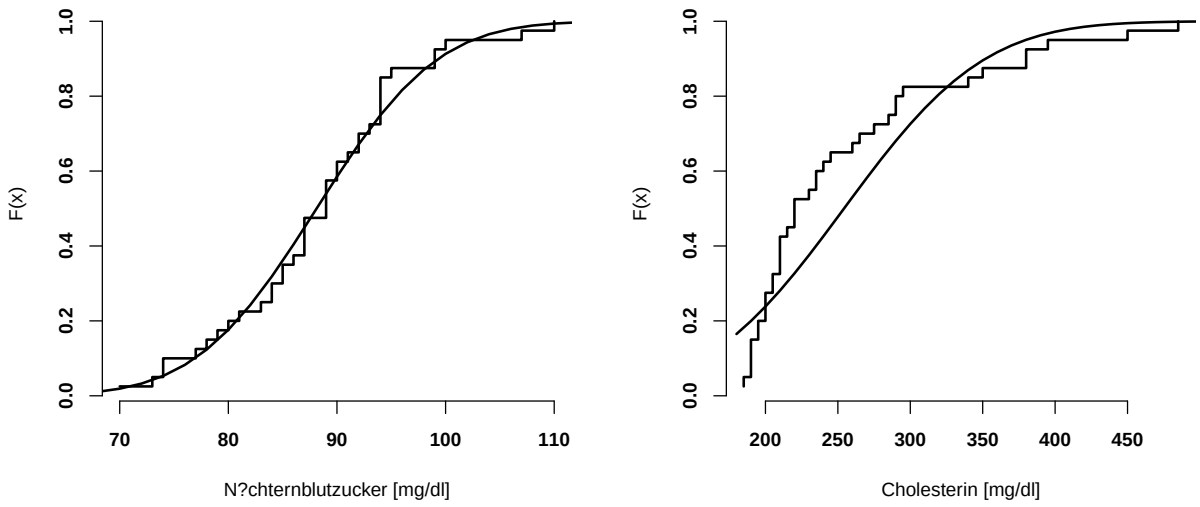
```
library(nortest)

pearson.test(nblz, n.classes=8, adjust=TRUE)
```

```
##
##  Pearson chi-square normality test
##
## data:  nblz
## P = 7.6, p-value = 0.1797

pearson.test(chol, n.classes=8, adjust=TRUE)
##
##  Pearson chi-square normality test
##
## data:  chol
## P = 21.6, p-value = 0.0006237
```

7.2.6 Kolmogoroff-Smirnoff Anpassungstest

Beispiel Nüchterblutzucker und Cholesterin:

```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
n <- length(nblz)
p <- cumsum(rep(1,n)/n)
plot(sort(nblz), p, type="s", xlab="N?chternblutzucker [mg/dl]", ylab="F(x)")
x <- seq(60, 120, by =2)
y <- pnorm(x, mean=mean(nblz), sd=sd(nblz), log=FALSE)
lines(x, y, col="black")

n <- length(chol)
p <- cumsum(rep(1,n)/n)
plot(sort(chol), p, type="s", xlab="Cholesterin [mg/dl]", ylab="F(x)")
x <- seq(180, 500, by =10)
y <- pnorm(x, mean=mean(chol), sd=sd(chol), log=FALSE)
lines(x, y, col="black")
```



Funktionen ks.test() und lilllie.test() in library(nortest)

```
library(nortest)

ks.test(nblz, "pnorm", mean(nblz), sd(nblz))
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  nblz
## D = 0.10063, p-value = 0.8127
## alternative hypothesis: two-sided

ks.test(chol, "pnorm", mean(chol), sd(chol))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  chol
## D = 0.19969, p-value = 0.08232
## alternative hypothesis: two-sided

lillie.test(nblz)
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nblz
## D = 0.10063, p-value = 0.3897

lillie.test(chol)
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  chol
## D = 0.19969, p-value = 0.0003435
```

7.2.7 Shapiro-Wilk Test

Funktion shapiro.test() in library(nortest)

```
library(nortest)

shapiro.test(nblz)
##
##  Shapiro-Wilk normality test
##
## data:  nblz
## W = 0.98006, p-value = 0.6918

shapiro.test(chol)
##
##  Shapiro-Wilk normality test
##
## data:  chol
## W = 0.80629, p-value = 9.187e-06
```

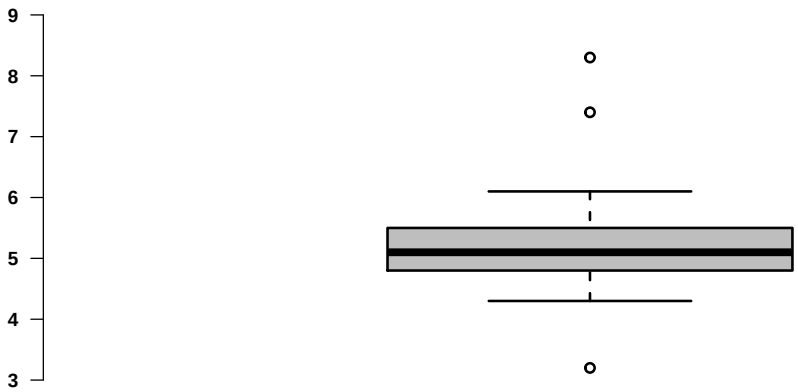
7.2.8 Anderson-Darling Test Anpassungstest

Funktion ad.test() in library(nortest)

```
library(nortest)

ad.test(nblz)
##
##  Anderson-Darling normality test
##
## data:  nblz
## A = 0.30513, p-value = 0.5525

ad.test(chol)
##
##  Anderson-Darling normality test
##
## data:  chol
## A = 2.761, p-value = 4.39e-07
```

7.2.9 Ausreißerproblem

```
par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
x <- c(3.2, 5.1, 5.0, 5.3, 5.1, 5.3, 4.8, 5.7, 4.5, 5.4,
       4.6, 4.8, 6.1, 5.5, 4.3, 4.8, 7.4, 8.3)
bp <- boxplot(x, range=1.5, col="grey", las=1, ylim=c(3,9)); bp$out
```




```
## [1] 3.2 7.4 8.3
```

Äussere Grenzen:

```r
Q <- quantile(x, p=c(0.25,0.5,0.75)); Q1 <- Q[1]; Q3 <- Q[3]
cat("Grenzen:",Q1 - IQR(x)*1.5,"-",Q3 + IQR(x)*1.5,"\n")
## Grenzen: 3.7875 - 6.4875
```

Ausreißer nach Hampel Kriterium:

```r
x <- c(3.2, 5.1, 5.0, 5.3, 5.1, 5.3, 4.8, 5.7, 4.5, 5.4,
       4.6, 4.8, 6.1, 5.5, 4.3, 4.8, 7.4, 8.3)
med.x <- median(x);
mad.x <- mad(x, constant=1)

outlier <- (x < med.x - 5.2*mad.x) | (x > med.x + 5.2*mad.x)
x[outlier]
## [1] 3.2 7.4 8.3
```

7.2.9.1 Grubbs-Test

```r
x <- c(3, 4, 4, 5, 6, 6, 7, 8, 9, 10, 10,
            11, 13, 15, 16, 17, 19, 19, 20, 50)
n <- length(x); m.x <- mean(x); s.x <- sd(x);
alpha <- 0.05; t <- qt(alpha/(2*n), n-2)
G.hat  <- max(abs(x-m.x))/s.x; G.hat
## [1] 3.610448
G.crit <- ((n-1)/sqrt(n)) * sqrt(t^2 / (n-2+t^2)); G.crit
## [1] 2.708246
```

Funktion grubbs.test() in library(outliers)

```r
library(outliers)

grubbs.test(x)
##
##  Grubbs test for one outlier
##
## data:  x
## G = 3.61045, U = 0.27782, p-value = 2.113e-05
## alternative hypothesis: highest value 50 is an outlier
```

7.2.9.2 Q-Test nach Dixon

Funktion dixon.test() in library(outliers)

```r
x <- c(3, 4, 4, 5, 6, 6, 7, 8, 9, 10, 10,
            11, 13, 15, 16, 17, 19, 19, 20, 50)

library(outliers)

dixon.test(x)
```

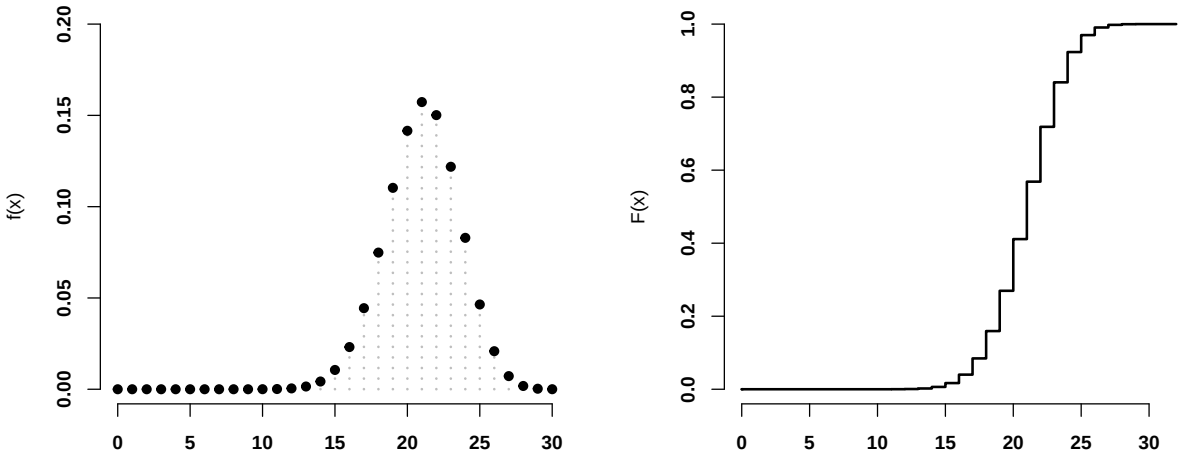
```
##
##  Dixon test for outliers
##
## data:  x
## Q = 0.67391, p-value < 2.2e-16
## alternative hypothesis: highest value 50 is an outlier
```

7.3 Einstichprobenverfahren

7.3.1 Hypothesen zu Wahrscheinlichkeiten

7.3.1.1 Binomialtest

```
par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
n  <- 30
p  <- 0.7
x  <- 0:n
fx <- dbinom(x, n, p)
plot(x, fx, type="h", ylim=c(0, 0.2), xlim=c(0,n), ylab="f(x)",
     xlab=" ", lty=3, col="grey")
points(x, fx, pch=19, cex=0.8, col="black")

Fx <- pbinom(x, n, p)
x1 <- c(x, n+1, n+2)
Fx <- c(Fx, 1, 1)
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]))
```




```
pbinom(25, 30, 0.7, lower.tail=FALSE)                    # Binomialtest
## [1] 0.03015494

binom.test(26, 30, p=0.7, alternative="greater")
##
##  Exact binomial test
##
## data:  26 and 30
## number of successes = 26, number of trials = 30, p-value = 0.03015
## alternative hypothesis: true probability of success is greater than 0.7
```

```
## 95 percent confidence interval:
##  0.7203848 1.0000000
## sample estimates:
## probability of success
##              0.8666667
qbinom(0.95, 30, 0.7)
## [1] 25
```

Beispiel reguläre Münze:

```
binom.test(15, 20, p=0.5, alternative="two.sided")
##
##  Exact binomial test
##
## data:  15 and 20
## number of successes = 15, number of trials = 20, p-value = 0.04139
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.5089541 0.9134285
## sample estimates:
## probability of success
##                   0.75
```

Zweiseitige Hypothesen:

```
n  <- 20; x  <- 15 ; p0 <- 0.5
pbinom(n-x, n, p0, lower.tail=TRUE) + pbinom(x-1, n, p0, lower.tail=FALSE)
## [1] 0.04138947
```

```
n  <- 20; x  <- 15 ; p0 <- 0.2
P <- 0; pH <- dbinom(x, n, p0)
for (i in 0:n) P <- ifelse(dbinom(i,n,p0) <= pH, P+dbinom(i,n,p0), P)
P
## [1] 1.802764e-07
```

7.3.1.2 Approximation durch die Normalverteilung

```
N <- 2000; n <- 400; x <- 184; p0 <- 0.40; p <- x/n  # Beispiel: H?ndler
z <- (abs(p-p0) - 1/(2*n)) / sqrt(((p0*(1-p0))/n)*((N-n)/(N-1))); z
## [1] 2.680888
pnorm(z, lower.tail=F)
## [1] 0.003671356

binom.test(184, 400, p=0.40, alternative="greater")           # exakter Test ...
##
##  Exact binomial test
##
## data:  184 and 400
## number of successes = 184, number of trials = 400, p-value = 0.008542
## alternative hypothesis: true probability of success is greater than 0.4
## 95 percent confidence interval:
##  0.4180829 1.0000000
## sample estimates:
## probability of success
##                   0.46
```

7.3.1.3 Fallzahlabschätzung

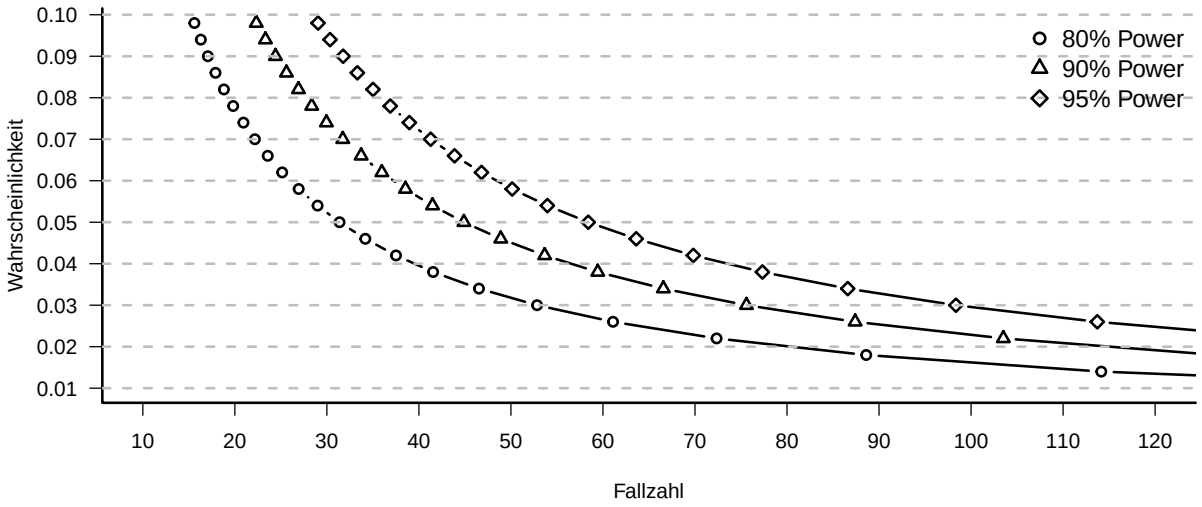
```
alpha <- 0.05; beta <- 0.20
p0 <- c(0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8)
 p <- c(0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9)

ceiling(((qnorm(1-alpha) + qnorm(1-beta))^2 *
            (p0*(1-p0) + p*(1-p))) / (p-p0)^2)
## [1] 155 229 279 303 303 279 229 155

power.prop.test(n=NULL, p1=0.1, p2=0.2, sig.level=0.05,
           power=0.80, alternative="one.sided")
##
##      Two-sample comparison of proportions power calculation
##
##               n = 156.6054
##              p1 = 0.1
##              p2 = 0.2
##       sig.level = 0.05
##           power = 0.8
##     alternative = one.sided
##
## NOTE: n is number in *each* group
```

```
prob <- seq(0.01, 0.10, by=0.004); lp <- length(prob)
conf <- c(0.80, 0.90, 0.95);       lc <- length(conf)
ntab <- matrix(rep(NA, lp*lc), ncol = lp, byrow = TRUE)

for (i in 1:lc)
{ for (j in 1:lp) ntab[i,j] <- (log10(1-conf[i])/log10(1-prob[j])) }
```

```r
rownames(ntab) <- c("80%","90%","95%")
colnames(ntab) <- prob
as.data.frame(ceiling(ntab)[,1:10])
```

	0.01	0.014	0.018	0.022	0.026	0.03	0.034	0.038	0.042	0.046
80%	161	115	89	73	62	53	47	42	38	35
90%	230	164	127	104	88	76	67	60	54	49
95%	299	213	165	135	114	99	87	78	70	64

```r
par(mfrow=c(1,1), lwd=2, bty="l")
plot(ntab[1,], prob, type="b", las=1, lty=1, pch=1, xlim=c(10, 120),
     yaxp=c(0.01, 0.20, 19), xaxp=c(10, 120, 11),
     ylab="Wahrscheinlichkeit", xlab="Fallzahl")
lines(ntab[2,], prob, type="b", lty=1, pch=2)
lines(ntab[3,], prob, type="b", lty=1, pch=5)
abline(h=seq(20,300,20), col="grey", lty=2)
abline(h=seq(0.01, 0.10, 0.01), lty=2, col="grey")
legend("topright", pch = c(1,2,5), bty="n",
       legend=c("80% Power","90% Power","95% Power"), cex=1.2)
```



7.3.1.4 Likelihood-Quotienten-Test

```r
n <- 60; x <- 4; p0 <- 1/6
minus2ll <- 2*(x*log(x/(n*p0)) + (n-x)*log((n-x)/(n-n*p0)))
minus2ll
## [1] 5.362487

pchisq(minus2ll, 1, lower.tail = FALSE)
## [1] 0.02057441

qchisq(0.95, 1)
## [1] 3.841459

binom.test(c(x, n-x), p = p0, alternative="less")              # exakter Test ...
##
##  Exact binomial test
##
## data:  c(x, n - x)
## number of successes = 4, number of trials = 60, p-value = 0.02019
## alternative hypothesis: true probability of success is less than 0.1666667
## 95 percent confidence interval:
##  0.0000000 0.1460972
## sample estimates:
## probability of success
##             0.06666667
```

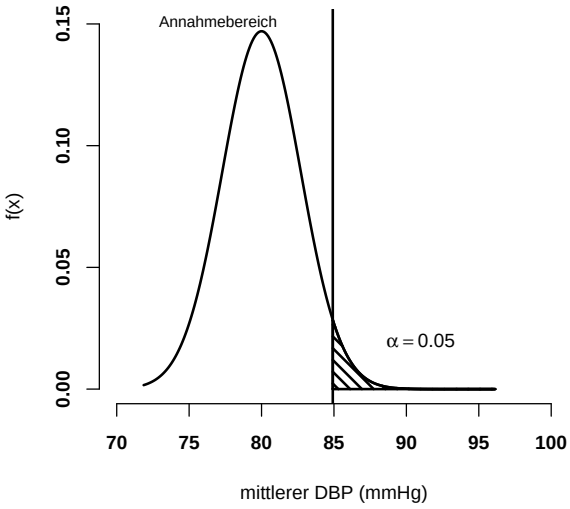
7.3.2 Hypothesen zu Erwartungswerten mit Bezug auf den Mittelwert

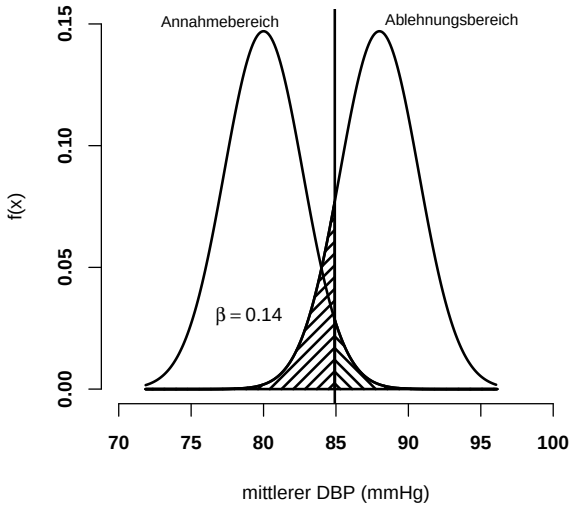
7.3.2.1 Einstichproben t-Test

```
m <- 9; s <- 2; n <- 25
t.hat <- abs(m-10)/(s/sqrt(n)); t.hat
## [1] 2.5

t.krit <- qt(0.975, n-1); t.krit
## [1] 2.063899
```

Beispiel erhöhter Blutdruck:

```
mue        <- 85                                          # Mittelwert Blutdruck
standardab <- 9                                           # Standardabweichung
n          <- 11                                          # Anzahl der Fälle
sem        <- standardab / sqrt(n)

mue.0      <- 80                                          # Erwartungswert unter H.0
tquant     <- qt(0.95, n-1); tquant                       # Quantil Testverteilung
## [1] 1.812461
mkrit      <- mue.0 + tquant*sem;   mkrit                 # kritischer Wert
## [1] 84.9183
p.val <- pt((mue-mue.0)/sem, n-1, lower.tail=FALSE)
mue.a      <- 88                                          # Erwartungswert unter H.A

x.i        <- seq(mue.0 - 3*sem, mue.a + 3*sem, by=0.1)
f0.i       <- dnorm(x.i, mean=mue.0, sd=sem)
fa.i       <- dnorm(x.i, mean=mue.a, sd=sem)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)
plot(x.i, f0.i, col="black", xlim=c(70, 100), ylim=c(0,0.15), yaxs="r",
     type="l", xlab="mittlerer DBP (mmHg)", ylab="f(x)")
abline(v=mkrit, col="black")
text(77, 0.151, "Annahmebereich", col="black", cex=0.8)
xval  <- seq(mkrit, mue.a + 3*sem, by=0.01)
xarea <- c(mkrit, xval); yarea <- c(0, dnorm(xval, mean=mue.0, sd=sem))
polygon(xarea, yarea, col="black", density=15, angle=135)
text(91, 0.02, expression(alpha == 0.05), cex=1)

plot(x.i, f0.i, col="black", xlim=c(70, 100), ylim=c(0,0.15), yaxs="r",
     type="l", xlab="mittlerer DBP (mmHg)", ylab="f(x)")
abline(v=mkrit, col="black")
text(77, 0.151, "Annahmebereich", col="black", cex=0.8)
xval  <- seq(mkrit, mue.a + 3*sem, by=0.01)
xarea <- c(mkrit, xval); yarea <- c(0, dnorm(xval, mean=mue.0, sd=sem))
polygon(xarea, yarea, col="black", density=15, angle=135)
lines(x.i, fa.i, col="black")
text(93, 0.151, "Ablehnungsbereich", col="black", cex=0.8)
xval  <- seq(mue.0 - 3*sem, mkrit, by=0.01)
xarea <- c(xval, mkrit); yarea <- c(dnorm(xval, mean=mue.a, sd=sem), 0)
polygon(xarea, yarea, col="black", density=15)
text(79,0.03, expression(beta == 0.14), cex=1)
```





Abschätzung der Power:

```
tstat <- (86 - mue.0)/sem;   tstat                          # Teststatistik
## [1] 2.211083

pt(tstat, n-1, lower.tail=F)                                # p-Wert für 'größer'
## [1] 0.02573309

beta <- pt((mkrit - mue.a)/sem, n-1, lower.tail=T)          # Power
beta
## [1] 0.1412941

1-beta;
## [1] 0.8587059
```

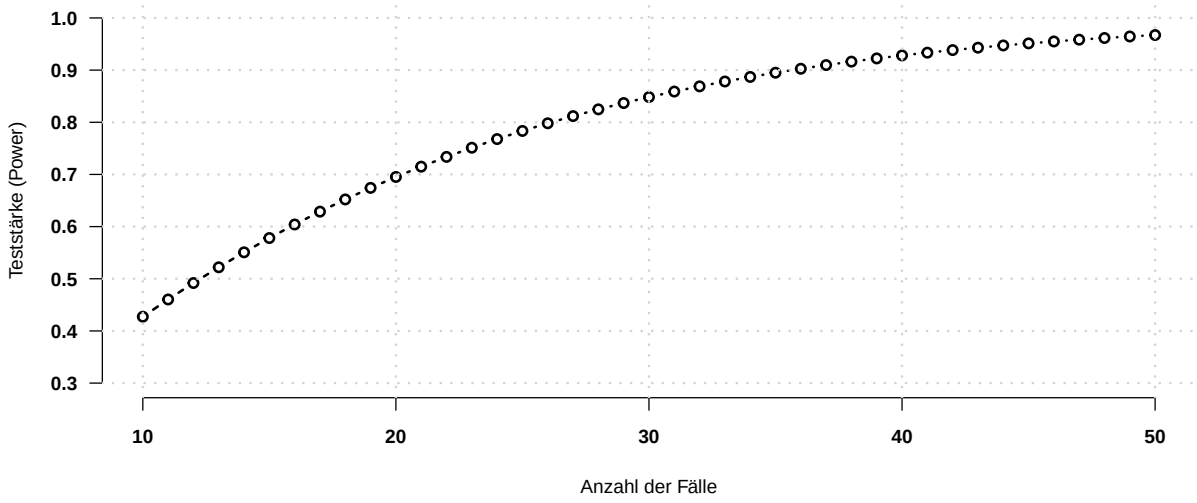
Beispiel Körpertemperatur:

```
temp <- c(36.8, 37.2, 37.5, 37.0, 36.9, 37.4, 37.9, 38.0)

t.test(temp, alternative="greater", mu=37)
##
##  One Sample t-test
##
## data:  temp
## t = 2.1355, df = 7, p-value = 0.03505
## alternative hypothesis: true mean is greater than 37
## 95 percent confidence interval:
##  37.03807      Inf
## sample estimates:
## mean of x
##   37.3375
```

7.3.2.2 Fallzahl und Power

```
d <- 15; s <- 30
effekt <- d / s

alpha <- 0.05; beta <- 0.20
n.1 <- ceiling((qnorm(1-alpha) + qnorm(1-beta))^2 / effekt^2);         n.1
## [1] 25
n.2 <- ceiling((qt(1-alpha, n.1-1) + qt(1-beta, n.1-1))^2 / effekt^2); n.2
## [1] 27
n.3 <- ceiling((qt(1-alpha, n.2-1) + qt(1-beta, n.2-1))^2 / effekt^2); n.3
## [1] 27

                                                          # Funktion power.t.test()
power.t.test(delta=15, sd=30, sig.level=0.05, power=0.80,
             type="one.sample",  alternative="one.sided")
##
##      One-sample t test power calculation
##
##               n = 26.13751
##           delta = 15
##              sd = 30
##       sig.level = 0.05
##           power = 0.8
##     alternative = one.sided
```

```
par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=11)
power <- power.t.test(n=10:50, delta=15, sd=30,
             sig.level=0.05, power=NULL, type="one.sample",
             alternative="one.sided")$power

plot(10:50, power, xlim=c(10,50), ylim=c(0.30, 1.0), type="b", las=1,
     xlab="Anzahl der Fälle", ylab="Teststärke (Power)"); grid()
```



7.3.2.3 Einstichprobentest auf Äquivalenz

Beispiel Mikrozirkulation:

```
                                        # Quantile zur nichtzentralen Fisher-Verteilung
myqf <- function(p, df1, df2, ncp) {
        uniroot(function(x) pf(x, df1, df2, ncp) - p, c(0, 100)) $ root
}

n <- 23; d <- 0.16; s.d <- 4.0; eps <- 0.5

t.hat <- (d/s.d)*sqrt(n); t.hat                          # Teststatistik
## [1] 0.1918333

c <- sqrt(myqf(0.05, 1, n-1, ncp=n*eps^2)); c            # kritischer Wert
## [1] 0.7594587
```

7.3.3 Einstichproben-Median-Test

```
                                                    #  Quantile zur Wilcoxon-Verteilung
qsignrank(0.95, 6:20, lower.tail = TRUE)
##  [1]  18  24  30  36  44  52  60  69  79  89 100 111 123 136 149
```

```
x <- c(12, 16, 18, 24, 26, 31, 38, 40)                  # Wilcoxon signed rank test
wilcox.test(x, alternative="two.sided", mu=30, conf.int=TRUE)
##
##  Wilcoxon signed rank test
##
## data:  x
## V = 10, p-value = 0.3125
## alternative hypothesis: true location is not equal to 30
## 95 percent confidence interval:
##  16.0 35.5
## sample estimates:
## (pseudo)median
##           25.5
```

7.3.6 Prüfung der Zufallsmäßigkeit einer Folge von Alternativdaten

7.3.6.2 Iterationstest / Runs-Test

```
library(tseries)                                    # Iterationstest / Runs-Test
werte <- c(18, 17, 18, 19, 20, 19, 19, 21, 18, 21, 22)
med   <- median(werte)

x     <- as.factor(werte<med); x
##  [1] TRUE  TRUE  TRUE  FALSE FALSE FALSE FALSE FALSE TRUE  FALSE FALSE
## Levels: FALSE TRUE

runs.test(x, alternative="two.sided")
##
##  Runs Test
##
## data:  x
## Standard Normal = -1.4489, p-value = 0.1474
## alternative hypothesis: two.sided
```

7.3.6.4 Vorzeichen-Trendtest von Cox und Stuart

```
cox.stuart.test <- function (x) {                   # Cox-Stuart Test
  method = "Cox-Stuart Test auf Trendänderung"
  leng = length(x);
  apross = round(leng) %% 2
  if (apross == 1) {delete = (length(x)+1)/2;x = x[-delete ] }
  half = length(x)/2
```

```
  x1 = x[1:half];       x2 = x[(half+1):(length(x))]
  difference = x1-x2;   signs = sign(difference)
  signcorr = signs[signs != 0]
  pos = signs[signs>0];   neg = signs[signs<0]
  if (length(pos) < length(neg)) {
    prop = pbinom(length(pos), length(signcorr), 0.5)
    names(prop) = "Aufwärtstrend, P-Wert"
    rval <- list(method = method, statistic = prop)
    class(rval) = "htest";     return(rval)
  }
  else {
    prop = pbinom(length(neg), length(signcorr), 0.5)
    names(prop) = "Abwärtstrend, P-Wert"
    rval <- list(method = method, statistic = prop)
    class(rval) = "htest";     return(rval)
  }
}
```

Beispiel mittlere Laufleistung von Kraftfahrzeugen:

```
mileage <- c(9.8,9.9,10.0,9.8,9.2,9.4,9.5,9.6,9.8,9.3,8.9,8.7,9.2,9.3)
cox.stuart.test(mileage)
##
##  Cox-Stuart Test auf Trendänderung
##
## data:
## Abwärtstrend, P-Wert = 0.0078125
```

7.3.7 Prüfung von Erwartungswerten zu Poisson-Verteilungen

```
1 - ppois(15, .10 * 100, lower.tail = TRUE)          # Einsticprobentest
## [1] 0.0487404
poisson.test(16, 0.10*100, alternative="greater")
##
##  Exact Poisson test
##
## data:  16 time base: 0.1 * 100
## number of events = 16, time base = 10, p-value = 0.04874
## alternative hypothesis: true event rate is greater than 1
## 95 percent confidence interval:
##  1.003596      Inf
## sample estimates:
## event rate
##        1.6

ppois(16, 1.6*10)                                    # Power
## [1] 0.5659624
```

7.3.7.1 Fallzahl und Power

```
lA <- 0.63; l0 <- 1.26
alpha <- 0.05; beta  <- 0.10; power <- 1 - beta
n <- 1/4*((qnorm(1-alpha) + qnorm(1-beta))/(sqrt(lA) - sqrt(l0)))^2
n <- ceiling(n); n
## [1] 20

lA <- 0.63; l0 <- 1.26
alpha <- 0.05; n <- 20
power <- 1-pnorm(2*sqrt(n)*(sqrt(lA)-sqrt(l0)) + qnorm(1-alpha))
power
## [1] 0.9024728

alpha <- 0.05; n <- 15
power <- 1-pnorm(2*sqrt(n)*(sqrt(lA)-sqrt(l0)) + qnorm(1-alpha))
power
## [1] 0.816419
```

7.3.2 Stichprobenumfang zur Prüfung einer Defektrate

```
n_defects <- function(lim, tol=1.5, alpha=0.10, power=0.90) {
  z_alpha <- qnorm(1-alpha);   z_power <- qnorm(power)
  units   <- ceiling(tol/lim * ((z_alpha/sqrt(tol) + z_power)/(tol-1))^2)
  defects <- ceiling(z_alpha*sqrt(units*lim) + units*lim)
  cat("\n","Es sind",units,"Einheiten zu prüfen...!","\n",
      "Die zulässige Fehlerzahl",lim,"wird um das",tol,"-fache","\n",
      "überschritten, wenn mehr als",defects,"Fehler auftreten!")    }
```

```
n_defects(lim=4, tol=1.5, alpha=0.10, power=0.90)
##
##  Es sind 9 Einheiten zu prüfen...!
##  Die zulässige Fehlerzahl 4 wird um das 1.5 -fache
##  überschritten, wenn mehr als 44 Fehler auftreten!
```

7.4 Zweistichprobenverfahren

7.4.1 Vergleich zweier Varianzen

```
n1 <- 41; sq1 <- 25;                                                    # F-Test
n2 <- 31; sq2 <- 16;

f.hat <- sq1 / sq2;                 f.hat
## [1] 1.5625
f.tab <- qf(0.95, n1-1, n2-1);      f.tab
## [1] 1.79179
```

```
x <- round(rnorm(10, mean=90, sd=10)); x
##  [1]  97  88  93 107 105  80  82  93  78 117
y <- round(rnorm(15, mean=90, sd=15)); y
##  [1]  72  56  94 117 102  92  96  79  83  71  70  77 103  76  80
var.test(x, y, ratio=1, altzernative="two.sided", conf.level=0.95)
##
##  F test to compare two variances
##
## data:  x and y
## F = 0.64504, num df = 9, denom df = 14, p-value = 0.5144
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.2009894 2.4498132
## sample estimates:
## ratio of variances
##          0.6450352
```

7.4.1.4 Stichprobenumfang und Power

```
pwr.var <- function(ratio, n=NA, power=NA, alpha=0.05, alternative="einseitig") {
  p.alpha <- ifelse (alternative=="einseitig", 1-alpha, 1-alpha/2)
  if (is.na(n) & !is.na(power)) {                    # Fallzahl
    g <- function(n) {power - (1-pf((1/ratio) * qf(p.alpha, n-1, n-1), n-1, n-1))}
    u <- uniroot(g, c(ratio, 1000 * ratio))$root
    n.r <- ceiling(u);        p.r <- power
  }
  if (!is.na(n) & is.na(power)) {                    # Power
    pwr <- 1 - pf((1/ratio) * qf(p.alpha, n-1, n-1), n-1, n-1)
    p.r <- round(pwr, 5);     n.r <- n
  }
  cat("Fallzahl/Power zum Vergleich zweier Varianzen","\n",
      "Quotient V1/V2  :",ratio,"\n",
      "Signif.niveau   :",alpha,alternative,"\n",
      "Power           :",p.r,"\n",
      "Fallzahl (n1=n2):",n.r,"\n")
}
```

Beispiel Prüfung eines Verhältnisses:

```r
pwr.var(ratio=3, power=0.90, alpha=0.05)
## Fallzahl/Power zum Vergleich zweier Varianzen
##  Quotient V1/V2  : 3
##  Signif.niveau   : 0.05 einseitig
##  Power           : 0.9
##  Fallzahl (n1=n2): 31

pwr.var(ratio=3, n=31, alpha=0.05)
## Fallzahl/Power zum Vergleich zweier Varianzen
##  Quotient V1/V2  : 3
##  Signif.niveau   : 0.05 einseitig
##  Power           : 0.90657
##  Fallzahl (n1=n2): 31
```

7.4.1.6 Vergleich zweier Variationskoeffizienten

```r
cv.test <- function(ni, xi, si){
  N   <- sum(ni); K   <- length(ni); vi <- si / xi
  ui  <- (ni*vi^2)/(1+vi^2)
  sn <- (N-K)*log(sum(ui/(N-K)))-sum((ni-1)*log(ui/(ni-1)))
  sd <- 1 + (1/(3*(K-1)) * sum(1/(ni-1))-(1/(N-K)))
  stat <- sn/sd;  p <- pchisq(stat, df=K-1, lower.tail=F)
  result <- list("chistat" = stat, "pval" = p)
  return(result)
}

ni <- c(54, 23); xi <- c(146.8, 31.2); si <- c(53.8, 31.0)
cvi <- (si/xi); round(cvi, 4)
## [1] 0.3665 0.9936

cv.test(ni, xi, si)
## $chistat
## [1] 18.5667
##
## $pval
## [1] 1.640612e-05
```

Funktion asymptotic_test2() in library(cvequality)

```r
library(cvequality)
asymptotic_test2(k=2, n=c(54, 23), s=c(53.8, 31.0), x=c(146.8, 31.2))
## $D_AD
## [1] 25.13042
##
## $p_value
## [1] 5.358093e-07
```

7.4.2 Rangdispersionstest von Siegel und Tukey

```
siegel.tukey.test = function(x=NA, y=NA, ties=T) {
  n1 <- length(x);   n2 <- length(y);   n  <- n1 + n2

  x <- c(x, y) ;  v <- c( rep(1, n1), rep(0, n2) )
  d <- rbind(x,v)[,order(x)]
  x.levels <- unique (x)
                                                  # Adjustierung bei ungerader Fallzahl
  if (n%%2==1) { d <- d[,c(1:trunc(n/2),(trunc(n/2)+2):n)] ;  n <- n - 1 }

  g = rep(NA, n)                                  # Erzeugen der Rangverteilung
  for (i in 1:n) {
   if                      (i%%2==0 & i  <n & i<=n/2) { g[i] <- 2*i       }
     else if               (i%%2==0 & n/2<i & i<=n  ) { g[i] <- 2*(n-i)+2 }
           else if         (i%%2==1 & 1 <=i & i<=n/2) { g[i] <- 2*i -1    }
                 else if (i%%2==1 & n/2<i & i<n   ) { g[i] <- 2*(n-i)+1 }
               }
  if (ties) {                                     # Bindungen?
    for (xl in x.levels) {
      x.i <- (d[1,]==xl)
      if (sum(x.i)>1) { g[x.i] <- mean(g[x.i]) }
                      }
           }
                                                  # Berechnung der Teststatistik
  ST <- sum(g*d[2,])
                                                  # P-Wert
  eins    <- ifelse (2*ST > n1*(n1+n2+1), -1, +1)
  z.hat   <- (2*ST - n1*(n1+n2+1)+eins)/sqrt(n1*(n1+n2+1)*(n2/3))
  p.value <- 2*pnorm(z.hat, lower.tail=FALSE)
                                  # Ergebnisse
  cat("     Siegel-Tukey-Test ","\n",
      "ST =", ST, "P-Wert (2-seitig) =", p.value, "\n")
 }
```

Beispiel:

```
A  <- c(10.1, 7.3, 12.6, 2.4, 6.1, 8.5, 8.8, 9.4, 10.1, 9.8)
B  <- c(15.3, 3.6, 16.5, 2.9, 3.3, 4.2, 4.9, 7.3, 11.7, 13.1)

siegel.tukey.test(A, B, ties=TRUE)
##      Siegel-Tukey-Test
##  ST = 134.5 P-Wert (2-seitig) = 0.02836551
```

7.4.3 Ansari-Bradley Test / LePage Test

```
A <- c(10.1, 7.3, 12.6, 2.4, 6.1, 8.5, 8.8, 9.4, 10.1,  9.8)
B <- c(15.3, 3.6, 16.5, 2.9, 3.3, 4.2, 4.9, 7.3, 11.7, 13.1)

ansari.test(A, B, alternative="two.sided")              # Ansari-Bradley
##
##  Ansari-Bradley test
##
## data:  A and B
## AB = 70.5, p-value = 0.0183
## alternative hypothesis: true ratio of scales is not equal to 1
```

Beispiel Silbergehalt byzantinischer Münzen:

```
A <- c(5.9, 6.0, 6.4, 7.0, 6.6, 7.7, 7.2, 6.9, 6.2); m <- length(A)
B <- c(5.3, 5.6, 5.5, 5.1, 6.2, 5.8, 5.8);           n <- length(B)
N <- m + n

W <- wilcox.test(A, B); W
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  A and B
## W = 60.5, p-value = 0.002518
## alternative hypothesis: true location shift is not equal to 0
S <- W$statistic                                        # Wilcox-Statistik
S1 <- (S - n*m/2) / sqrt(m*n*(N+1)/12); S1
##        W
## 3.069686

A <- ansari.test(A, B); A;
##
##  Ansari-Bradley test
##
## data:  A and B
## AB = 43.5, p-value = 0.5204
## alternative hypothesis: true ratio of scales is not equal to 1
S <- A$statistic                                        # Ansari-Statistik

if (N%%2==0) {
   S2 <- (S - (m*(N+2)/4)) / sqrt((m*n*(N^2-4)/(48*(N-1))));
   S2 <- (S - (m*(N+1)^2)/(4*N)) / sqrt(m*n*(N+1)*(3+N^2) / (48*N^2))}; S2
##        AB
## 0.6018207

lepage <- S1^2 + S2^2; lepage                           # LePage Test
##        W
## 9.785157

pchisq(lepage, 2, lower.tail=FALSE)
##           W
## 0.007502052
```

7.4.4 t-Test für unabhängige Stichproben

7.4.4.1 Unbekannte aber gleiche Varianzen

```
n1 <- 16; xbar1 <- 14.5; s1 <- 4
n2 <- 14; xbar2 <- 13.0; s2 <- 3
Q1 <- (n1 - 1)*s1
Q2 <- (n2 - 1)*s2

t.hat <- (xbar1 - xbar2) / sqrt(((n1 + n2)/(n1*n2)) *
            ((Q1 + Q2)/(n1+n2-2))); t.hat
## [1] 2.179797

t.krit <- qt(0.975, n1+n2-2); t.krit
## [1] 2.048407
```

Beispiel Gerinnungszeiten:

```
x <- c(8.8, 8.4, 7.9, 8.7, 9.1, 9.6)
y <- c(9.9, 9.0, 11.1, 9.6, 8.7, 10.4, 9.5)

t.test(x, y, alternative="two.sided", var.equal=TRUE)
##
##  Two Sample t-test
##
## data:  x and y
## t = -2.4765, df = 11, p-value = 0.03076
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.8752609 -0.1104534
## sample estimates:
## mean of x mean of y
##  8.750000  9.742857

qt(0.975, 11)
## [1] 2.200985
```

7.4.4.2 Unbekannte Varianzen - möglicherweise nicht gleich

Beispiel HDL-Werte:

```
aktiv   <- c(29.5, 44.9, 54.2, 55.4, 58.5, 59.8, 60.1, 84.2, 97.5)
inaktiv <- c(32.3, 32.7, 37.4, 38.4, 40.1,
                    40.6, 45.3, 45.6, 52.0, 60.3, 60.5)

t.test(aktiv, inaktiv, alternative="greater", var.equal=FALSE)
##
##  Welch Two Sample t-test
##
## data:  aktiv and inaktiv
## t = 2.2378, df = 11.141, p-value = 0.0233
```

```
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  3.243236      Inf
## sample estimates:
## mean of x mean of y
##  60.45556  44.10909
```

7.4.4.3 Fallzahlabschätzung und Power

```
power.t.test(delta=15, sd=20, sig.level=0.05, power=0.80,
             type="two.sample", alternative="one.sided")
##
##      Two-sample t test power calculation
##
##               n = 22.69032
##           delta = 15
##              sd = 20
##       sig.level = 0.05
##           power = 0.8
##     alternative = one.sided
##
## NOTE: n is number in *each* group
```

Tabelle:

```
beta  <- c(0.3, 0.2, 0.1)                                  # Tabelle 7.24
d     <- c(0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1.0, 1.5)

alpha <- 0.05
n11    <- ceiling(((qnorm(1-alpha)+qnorm(1-0.3))^2 * 2) / d^2); n11
## [1] 942 236 105  59  38  20  10   5
n12    <- ceiling(((qnorm(1-alpha)+qnorm(1-0.2))^2 * 2) / d^2); n12
## [1] 1237  310  138   78   50   26   13    6
n13    <- ceiling(((qnorm(1-alpha)+qnorm(1-0.1))^2 * 2) / d^2); n13
## [1] 1713  429  191  108   69   35   18    8
n21    <- ceiling(((qnorm(1-alpha/2)+qnorm(1-0.3))^2 * 2) / d^2); n21
## [1] 1235  309  138   78   50   26   13    6
n22    <- ceiling(((qnorm(1-alpha/2)+qnorm(1-0.2))^2 * 2) / d^2); n22
## [1] 1570  393  175   99   63   33   16    7
n23    <- ceiling(((qnorm(1-alpha/2)+qnorm(1-0.1))^2 * 2) / d^2); n23
## [1] 2102  526  234  132   85   43   22   10

tab <- matrix(data = NA, nrow = 8, ncol = 7, byrow = FALSE, dimnames = NULL)
tab[,1] <- d
tab[,2] <- n11; tab[,3] <- n12; tab[,4] <- n13
tab[,5] <- n21; tab[,6] <- n22; tab[,7] <- n23
colnames(tab) <- c("effekt","0.7","0.8","0.9","0.7","0.8","0.9")

as.data.frame(tab[, 1:4])                                  # alpha=0.05 einseitig
```

effekt	0.7	0.8	0.9
0.1	942	1237	1713
0.2	236	310	429
0.3	105	138	191
0.4	59	78	108
0.5	38	50	69
0.7	20	26	35
1.0	10	13	18
1.5	5	6	8

```
as.data.frame(tab[, c(1,5:7)])                    # alpha=0.05 zweiseitig
```

effekt	0.7	0.8	0.9
0.1	1235	1570	2102
0.2	309	393	526
0.3	138	175	234
0.4	78	99	132
0.5	50	63	85
0.7	26	33	43
1.0	13	16	22
1.5	6	7	10

```
alpha <- 0.01
n11    <- ceiling(((qnorm(1-alpha)+qnorm(1-0.3))^2 * 2) / d^2); n11
## [1] 1626  407  181  102   66   34   17    8
n12    <- ceiling(((qnorm(1-alpha)+qnorm(1-0.2))^2 * 2) / d^2); n12
## [1] 2008  502  224  126   81   41   21    9
n13    <- ceiling(((qnorm(1-alpha)+qnorm(1-0.1))^2 * 2) / d^2); n13
## [1] 2604  651  290  163  105   54   27   12
n21    <- ceiling(((qnorm(1-alpha/2)+qnorm(1-0.3))^2 * 2) / d^2); n21
## [1] 1923  481  214  121   77   40   20    9
n22    <- ceiling(((qnorm(1-alpha/2)+qnorm(1-0.2))^2 * 2) / d^2); n22
## [1] 2336  584  260  146   94   48   24   11
n23    <- ceiling(((qnorm(1-alpha/2)+qnorm(1-0.1))^2 * 2) / d^2); n23
## [1] 2976  744  331  186  120   61   30   14

tab <- matrix(data = NA, nrow = 8, ncol = 7, byrow = FALSE, dimnames = NULL)
tab[,1] <- d
tab[,2] <- n11; tab[,3] <- n12; tab[,4] <- n13
tab[,5] <- n21; tab[,6] <- n22; tab[,7] <- n23
colnames(tab) <- c("effekt","0.7","0.8","0.9","0.7","0.8","0.9")

as.data.frame(tab[, 1:4])                         # alpha=0.01 einseitig
```

effekt	0.7	0.8	0.9
0.1	1626	2008	2604
0.2	407	502	651
0.3	181	224	290

effekt	0.7	0.8	0.9
0.4	102	126	163
0.5	66	81	105
0.7	34	41	54
1.0	17	21	27
1.5	8	9	12

```
as.data.frame(tab[, c(1,5:7)])                    # alpha=0.01 zweiseitig
```

effekt	0.7	0.8	0.9
0.1	1923	2336	2976
0.2	481	584	744
0.3	214	260	331
0.4	121	146	186
0.5	77	94	120
0.7	40	48	61
1.0	20	24	30
1.5	9	11	14

Beispiel Gerinnung:

```
power.t.test(n=20, sd=sqrt(0.905), sig.level=0.05, power=0.90,
             type="two.sample", alternative="two.sided")
##
##      Two-sample t test power calculation
##
##               n = 20
##           delta = 1.000755
##              sd = 0.9513149
##       sig.level = 0.05
##           power = 0.9
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

Beispiel Sauerstoffaufnahme:

```
m1 <- 43.5;  m2 <- 46.2;  sp <- 2.8;   n  <- 15; alpha <- 0.05
nu    <- (n - 1)*2
quant <- qt(alpha/2, df=nu, lower.tail = FALSE)
nonc  <- sqrt(n/2) * abs(m1-m2)/sp; round(nonc, 2)
## [1] 2.64
power <- pt(quant, df=nu, ncp=nonc, lower.tail = FALSE) +
               pt(-quant, df=nu, ncp=nonc, lower.tail = TRUE)

round(power, 4)
## [1] 0.7222

power.t.test(n=n, delta=abs(m1-m2), sd=sp, sig.level=alpha, alternative="two.sided")
```

```
##
##      Two-sample t test power calculation
##
##               n = 15
##           delta = 2.7
##              sd = 2.8
##       sig.level = 0.05
##           power = 0.7222012
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

7.4.4.4 Bootstrap: t-Test Variante

```
two.sample.bootstrap <- function(x, y, B=999) {
  x <- x[complete.cases(x)];  y <- y[complete.cases(y)]
  n.x <- length(x);           n.y <- length(y)
  mean.x <- mean(x, na.rm=T); mean.y <- mean(y, na.rm=T)
  stdv.x <- sd(x, na.rm=T);   stdv.y <- sd(y, na.rm=T)
  d.x    <- stdv.x^2/n.x;        d.y <- stdv.y^2/n.y
  W      <- rep(NA, B)
  for (i in 1:B) {
    xi <- sample(x, n.x, replace=T);   yi <- sample(y, n.y, replace=T)
    mean.xi <- mean(xi, na.rm=T); mean.yi <- mean(yi, na.rm=T)
    stdv.xi <- sd(xi, na.rm=T);   stdv.yi <- sd(yi, na.rm=T)
    d.xi    <- stdv.xi^2/n.x;        d.yi <- stdv.yi^2/n.y
    W[i]    <- ((mean.xi-mean.yi)-(mean.x-mean.y))/sqrt(d.xi+d.yi)  }
  W <- sort(W); L <- round(0.025*B,0); U <- round(0.975*B,0)
  ci95.L <- round(mean.x - mean.y + W[L]*sqrt(d.x+d.y),3)
  ci95.U <- round(mean.x - mean.y + W[U]*sqrt(d.x+d.y),3)
  cat("       Bootstrap: t-Test Variante","\n",
      "Mittelwerte:",round(mean.x,3),"und",round(mean.y,3),"\n",
      "Differenz  :",round(mean.x-mean.y,3),
      "mit den 95%-Konfidenzgrenzen: [",ci95.L,",",ci95.U,"]","\n")
}
```

Beispiel HDL-Werte:

```
aktiv   <- c(29.5, 44.9, 54.2, 55.4, 58.5, 59.8, 60.1, 84.2, 97.5)
inaktiv <- c(32.3, 32.7, 37.4, 38.4, 40.1, 40.6, 45.3, 45.6, 52.0, 60.3, 60.5)

two.sample.bootstrap(aktiv, inaktiv, B=999)
##        Bootstrap: t-Test Variante
##  Mittelwerte: 60.456 und 44.109
##  Differenz  : 16.346 mit den 95%-Konfidenzgrenzen: [ -0.63 , 31.085 ]
```

```
t.test(aktiv, inaktiv, alternative="two.sided", var.equal=FALSE)
##
##  Welch Two Sample t-test
##
```

```
## data:  aktiv and inaktiv
## t = 2.2378, df = 11.141, p-value = 0.04659
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   0.2937054 32.3992239
## sample estimates:
## mean of x mean of y
##  60.45556  44.10909
```

```
wilcox.test(aktiv, inaktiv, alternative = "two.sided", exact = TRUE, correct = FALSE)
##
##  Wilcoxon rank sum test
##
## data:  aktiv and inaktiv
## W = 73, p-value = 0.0804
## alternative hypothesis: true location shift is not equal to 0
```

7.4.4.5 Multivariater t-Test (Hotelling)

```
hotelling <- function(x1, x2) {
  k1 <- ncol(x1);  k2 <- ncol(x2)          # Dimensionen in x1 und x2
  if (k2 != k1) stop("Fehler in den Dimensionen: x1 hat ", k1,
                     " und x2 hat ", k2, "Spalten - Dimensionen!")
  p <- k1                                  # Anzahl Variablen
  n1    <- nrow(x1);              n2 <- nrow(x2)
  xbar1 <- apply(x1, 2, mean); xbar2 <- apply(x2, 2, mean)
  diffbar <- xbar1 - xbar2
  v <- ((n1-1)*var(x1) + (n2-1)*var(x2)) / (n1+n2-2)
  # Hotelling's T2
  T2 <- n1*n2/(n1+n2) * (diffbar %*% solve(v) %*% diffbar)
  F  <- ((n1+n2-p-1)/((n1+n2-2)*p))*T2     # transform zu Fisher
  pvalue <- 1-pf(F, p, n1+n2-p-1)
  D2 <- diffbar %*% solve(v) %*% diffbar   # Mahalanobis Distanz
  cat(" Hotelling's multivariate T-Statistik","\n",
      " Differenzen    : ", diffbar,"\n",
      " Hotelling's T2 : ", T2,"\n",
      " F-Statistik    : ", F,"(P-Wert = ",pvalue,")","\n",
      " Mahalanobis D2 : ", D2,"\n")
}
```

Beispiel Schweißrate:

```
schweiss <- read.csv("sweat.csv", header=T, sep=";", dec=",")
str(schweiss)
## 'data.frame':    20 obs. of  5 variables:
##  $ id    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Gruppe: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Sweat : num  1.5 2.4 2.7 3.2 3.7 3.8 3.9 4.6 4.8 6.7 ...
##  $ Na    : num  13.5 24.8 55.5 53.2 48.5 47.2 36.9 36.1 54.1 47.4 ...
##  $ Ka    : num  10.1 14 19.7 12 9.3 10.9 12.7 17.9 11.3 8.5 ...
```

```
data      <- with(schweiss, cbind(Sweat, Na, Ka))
trained   <- subset(data, schweiss$Gruppe==1)
untrained <- subset(data, schweiss$Gruppe==2)
hotelling(trained, untrained)
##  Hotelling's multivariate T-Statistik
##   Differenzen    :  -1.81 -10.36 3.35
##   Hotelling's T2 :  11.30816
##   F-Statistik    :  3.350565 (P-Wert =  0.04544368 )
##   Mahalanobis D2 :  2.261631
```

Funktion hotelling.test() in library(Hotelling)

```
library(Hotelling)
print(hotelling.test(trained, untrained))
## Test stat:  3.3506
## Numerator df:  3
## Denominator df:  16
## P-value:  0.04544
```

7.4.5 t-Test für Paardifferenzen

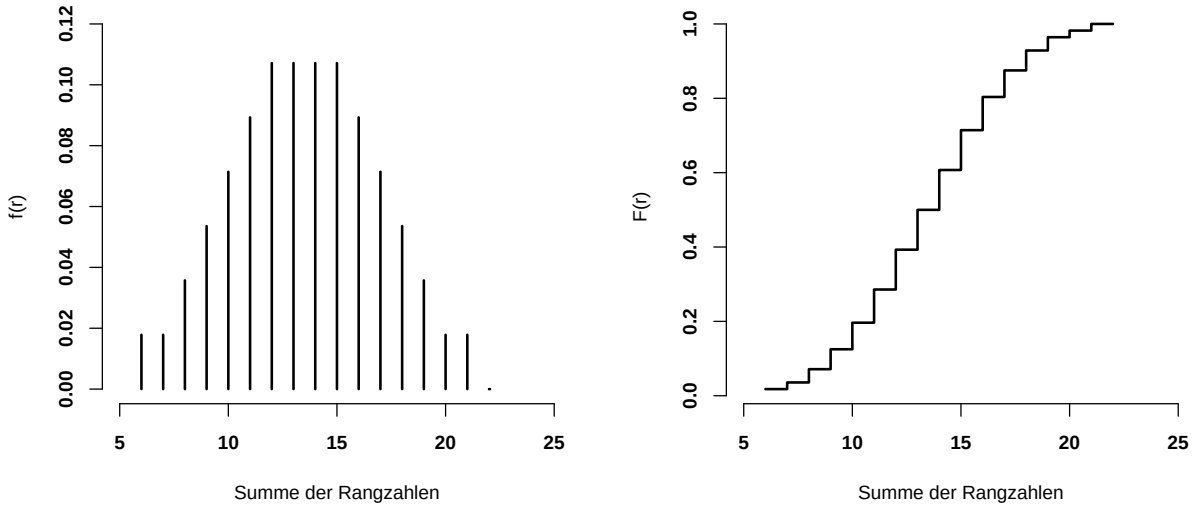
7.4.5.3 t-Test für paarweise angeordnete Messwerte

Beispiel Behandlungseffekt:

```
behandelt   <- c(4.0, 3.5, 4.1, 5.5, 4.6, 6.0, 5.1, 4.3)
unbehandelt <- c(3.0, 3.0, 3.8, 2.1, 4.9, 5.3, 3.1, 2.7)

t.test(behandelt, unbehandelt, alternative = c("two.sided"), paired = TRUE)
##
##  Paired t-test
##
## data:  behandelt and unbehandelt
## t = 2.798, df = 7, p-value = 0.0266
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.17811777 2.1218823
## sample estimates:
## mean of the differences
##                    1.15
```

7.4.6 Wilcoxon-Rangsummentrest (U-Test)

```
n1 <- 3;  n2 <- 5
U  <- 0:(n1*n2 +1)
W  <- U + n1*(n1+1)/2
f  <- dwilcox(U, n1, n2)
F  <- pwilcox(U, n1, n2)

par(mfrow=c(1,2), lwd=2, font.axis=2, bty="n", ps=11)

plot(W, f, type='h', col="black", xlab="Summe der Rangzahlen",
     ylab="f(r)", xlim=c(5,25), ylim=c(0,0.12))
plot(W, F, type="s", col="black", xlab="Summe der Rangzahlen",
     ylab="F(r)", xlim=c(5,25))
```



Untere / obere Quantile zur Wilcoxon-Verteilung (alpha=0.025 einseitig)

```
m <- 2:10; n <- 10
utab.l <- qwilcox(0.025, m, n, lower.tail=TRUE)
rtab.l <- utab.l + m*(m+1)/2
utab.u <- qwilcox(0.975, m, n, lower.tail=TRUE)
rtab.u <- utab.u + m*(m+1)/2

utab.l; utab.u                                  # untere / obere Quantile zur U-Statistik
## [1]  1  4  6  9 12 15 18 21 24
## [1] 19 26 34 41 48 55 62 69 76

rtab.l; rtab.u                                  # untere / obere Quantile zu den Rangsummen
## [1]  4 10 16 24 33 43 54 66 79
## [1]  22  32  44  56  69  83  98 114 131
```

Beispiel 1:

```
A <- c(7, 14, 22, 36, 40, 48, 49, 52); n1 <- length(A)
B <- c(3, 5, 6, 10, 17, 18, 20, 39);   n2 <- length(B)

All <- c(A, B)                                  # verbinden der Stichproben
grp <- c(rep(1, n1), rep(2, n2))                # kennzeichnen der Gruppe
rnk <- rank(All)                                # zuordnen der Rangzahlen

xdata <- matrix(c(grp,All,rnk),ncol=3)          # Aufbau der Matrix
Namen <- c("Gruppe","Wert","Rang")              # Namen der Datenspalten
dimnames(xdata) <- list(NULL, Namen)
data <- as.data.frame(xdata)

attach(data)
r1 <- sum(Rang[Gruppe==1]); r1
```

```
## [1] 89
r2 <- sum(Rang[Gruppe==2]); r2
## [1] 47

u1 <- r2 - n2*(n2+1)/2; u1
## [1] 11
u2 <- r1 - n1*(n1+1)/2; u2
## [1] 53
```

```
wilcox.test(A, B, alternative="greater", conf.int=TRUE, conf.level=0.95)
##
##  Wilcoxon rank sum test
##
## data:  A and B
## W = 53, p-value = 0.01406
## alternative hypothesis: true location shift is greater than 0
## 95 percent confidence interval:
##    4 Inf
## sample estimates:
## difference in location
##                   19.5
```

Beispiel 2:

```
A <- c(63, 68, 70, 81, 97, 104)
B <- c(91, 92, 95, 96, 99)

wilcox.test(A, B, alternative="two.sided")
##
##  Wilcoxon rank sum test
##
## data:  A and B
## W = 9, p-value = 0.329
## alternative hypothesis: true location shift is not equal to 0
```

7.4.6.1 Der U-Test bei Rangaufteilung

```
A <- c(5, 5, 8, 9, 13, 13, 13, 15)
B <- c(3, 3, 4, 5,  5,  8, 10, 16)

wilcox.test(A, B, alternative="two.sided")
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  A and B
## W = 47.5, p-value = 0.1109
## alternative hypothesis: true location shift is not equal to 0
```

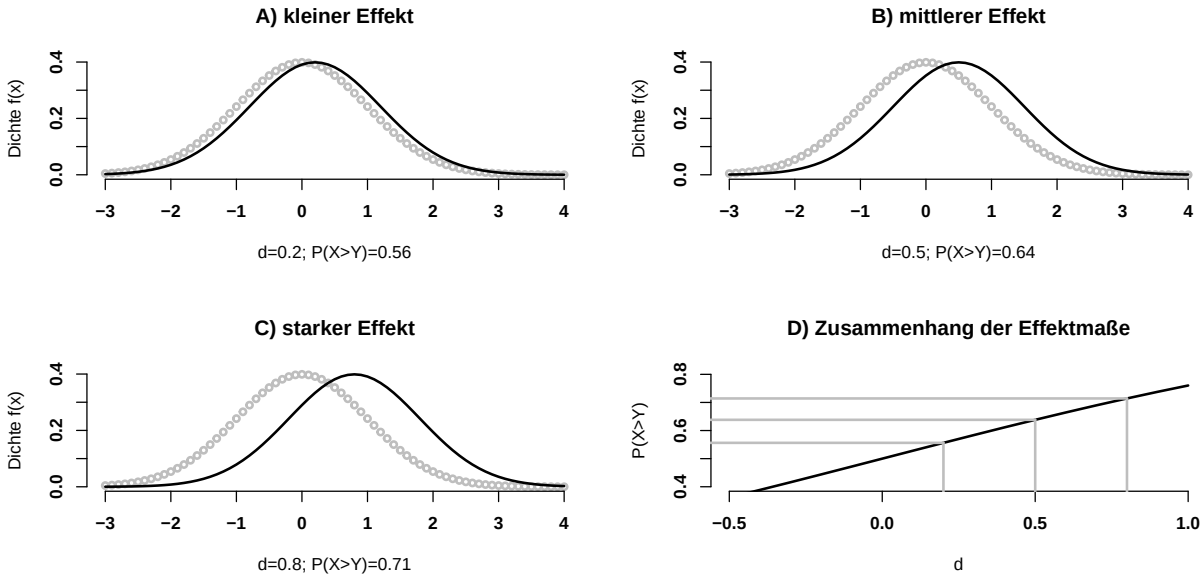
Funktion wilcox_test() in library(coin)

```
A <- c(5, 5, 8, 9, 13, 13, 13, 15)
B <- c(3, 3, 4, 5,  5,  8, 10, 16)

library(coin)
wert <- c(A, B)
grp  <- as.factor(c(rep("A",length(A)),rep("B",length(B))))
dat  <- as.data.frame(c(wert, grp))

wilcox_test(wert ~ grp, data = dat, alternative="two.side",
                        distribution = "exact", conf.int = TRUE)
##
##  Exact Wilcoxon-Mann-Whitney Test
##
## data:  wert by grp (A, B)
## Z = 1.6473, p-value = 0.1071
## alternative hypothesis: true mu is not equal to 0
## 95 percent confidence interval:
##  -1  9
## sample estimates:
## difference in location
##                      4
```

7.4.6.2 Effektstärken im Vergleich unabhängiger Stichproben

```
mue1  <- 0; sigma <- 1
d     <- c(0.2, 0.5, 0.8)
mue2  <- d * sigma
                                                    # Abbildung 7.14
par(mfrow=c(2,2), lwd=2, font.axis=2, bty="n", ps=12)
x <- seq(-3, 4, by=0.1)
f  <- dnorm(x, mean=mue1, sd=sigma)
f1 <- dnorm(x, mean=mue2[1], sd=sigma)
plot(x, f, type="b", xlab="d=0.2; P(X>Y)=0.56", ylab="Dichte f(x)",
     main="A) kleiner Effekt", col="grey", cex=0.8)
lines(x, f1)
```

```
f2 <- dnorm(x, mean=mue2[2], sd=sigma)
plot(x, f, type="b", xlab="d=0.5; P(X>Y)=0.64", ylab="Dichte f(x)",
     main="B) mittlerer Effekt", col="grey", cex=0.8)
lines(x, f2)
f3 <- dnorm(x, mean=mue2[3], sd=sigma)
plot(x, f, type="b", xlab="d=0.8; P(X>Y)=0.71", ylab="Dichte f(x)",
     main="C) starker Effekt", col="grey", cex=0.8)
lines(x, f3)
delta <- seq(-1, 1, b=0.1); F <- pnorm(delta/sqrt(2), mean=0, sd=1)
plot(delta, F, type="l", ylim=c(0.4, 0.8),
            xlim=c(-0.5, 1), xlab="d", ylab="P(X>Y)",
            main="D) Zusammenhang der Effektmaße", cex=0.8)
y <- pnorm(0.2/sqrt(2), mean=0, sd=1)
lines(c(0.2, 0.2), c(0, y), col="grey")
lines(c(-2, 0.2), c(y, y), col="grey")
y <- pnorm(0.5/sqrt(2), mean=0, sd=1)
lines(c(0.5, 0.5), c(0, y), col="grey")
lines(c(-2, 0.5), c(y, y), col="grey")
y <- pnorm(0.8/sqrt(2), mean=0, sd=1)
lines(c(0.8, 0.8), c(0, y), col="grey")
lines(c(-2, 0.8), c(y, y), col="grey")
```



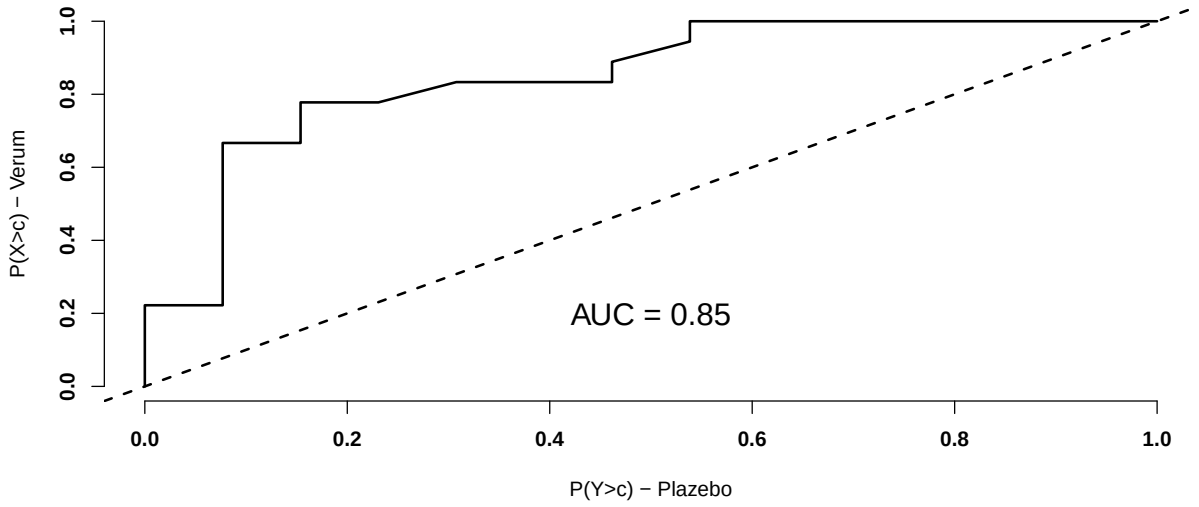
Beispiel Toxizität:

```
placebo <- c(1.69, 1.96, 1.76, 1.88, 2.30, 1.97, 1.69, 1.63, 2.01,
             1.92, 1.93, 1.56, 1.71); n <- length(placebo)
verum   <- c(2.12, 1.88, 2.15, 1.96, 1.83, 2.03, 2.19, 2.10, 2.15,
             2.00, 2.25, 2.49, 2.43, 1.89, 2.38, 2.37, 2.05, 2.00)
m <- length(verum)
d <- matrix(rep(NA, n*m), nrow=n)
for (i in 1:n) {
     for (j in 1:m) {
```

```
     if (placebo[i]  < verum[j]) d[i, j] <- 1
     if (placebo[i] == verum[j]) d[i, j] <- 0.5
     if (placebo[i]  > verum[j]) d[i, j] <- 0
}}
P.hat <- sum(apply(d, 1, sum))/(m*n); P.hat
## [1] 0.8504274
s     <- sqrt(((n-1)*var(verum) + (m-1)*var(placebo))/(n+m-2))
d.hat <- (mean(verum)-mean(placebo))/ s; d.hat
## [1] 1.411237
```

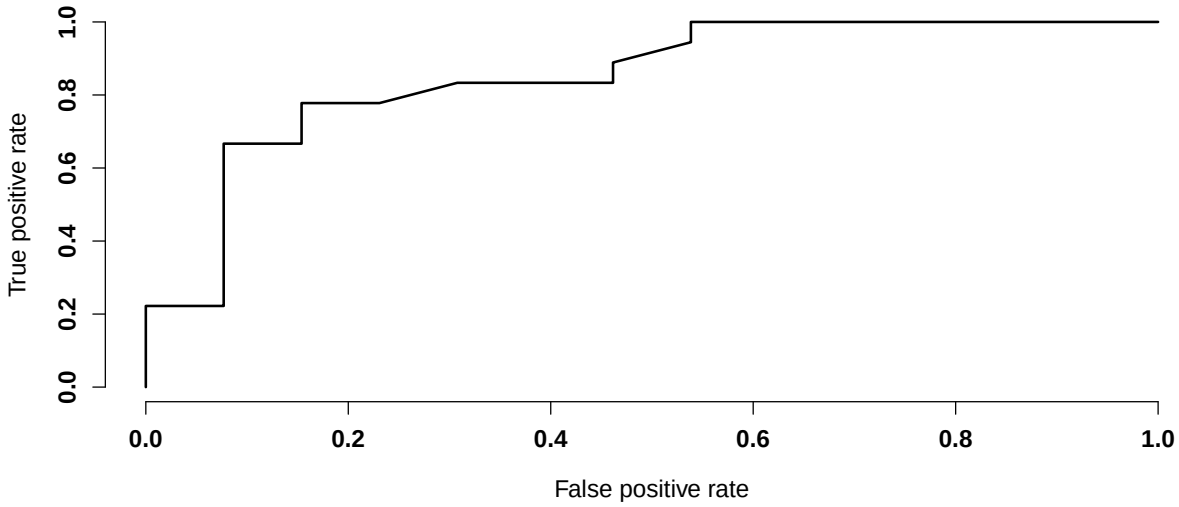
ROC-Analyse:

```
c <- seq(1.5, 2.5, by=0.01); nc <- length(c)                    # ROC-Kurve
rocx <- rep(NA, nc); rocy <- rep(NA,nc)
for (i in 1:nc)   {
    x <- 0; y <- 0
    for (k in 1:m) {if (verum[k]   > c[i]) y <- y + 1}
    for (k in 1:n) {if (placebo[k] > c[i]) x <- x + 1}
    rocx[i] <- x/n; rocy[i] <- y/m
}

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="n", ps=12)
plot(rocx, rocy, type="l", xlab="P(Y>c) - Plazebo", ylab="P(X>c) - Verum")
abline(0,1, lty=2); text(0.5, 0.2, "AUC = 0.85", cex=1.5)
```



Funktion performance() in library(ROCR)

```
library(ROCR)
par(mfcol=c(1,1),lwd=2, font.axis=2, bty="n", ps=14, las=0)
pred <- prediction(c(verum, placebo),c(rep(1, m), rep(0, n)))
perf <- performance( pred, "tpr", "fpr" )
plot(perf)
```



7.4.6.3 Fallzahlabschätzung für den U-Test

```
npwr.Utest <- function(effect, alpha=0.05, alternative="zweiseitig",
                       power=NULL, N=NULL, c=0.5) {
  if (sum(sapply(list(N, power, alpha), is.null)) != 1)
    stop("Fehler: Nur eins von N oder power darf NULL sein")
  sig.level <- ifelse (alternative=="einseitig", alpha*2, alpha)
  if (is.null(N)) {
    N  <- (qnorm(1-power) + qnorm(sig.level/2))**2 / (12*c*(1-c)*(effect-0.5)**2)
    n1 <- ceiling(N*c); n2 <- ceiling(N*(1-c))
    Power <- power   }
  if (is.null(power)) {
    sigma0 <- sqrt(1/(12*c*(1-c)))
    Power <- pnorm((sqrt(N)*abs(effect-0.5)+qnorm(sig.level/2) * sigma0) / sigma0)
    n1 <- round(N*c,0); n2 <- N - n1   }
  cat("Stichprobenumfang n1=",n1,"und n2=",n2,"\n",
      "Effekt P(X>Y)       =",effect,"\n",
      "Signifikanzniveau   =",alpha,"\n",
      "Alternativhypothese =",alternative,"\n",
      "Power               =",round(Power*100,2),"%")
}
```

Beispiel:

```
npwr.Utest(effect=2/3, alpha=0.05, alternative="einseitig", power=0.80)
## Stichprobenumfang n1= 38 und n2= 38
##  Effekt P(X>Y)       = 0.6666667
##  Signifikanzniveau   = 0.05
##  Alternativhypothese = einseitig
##  Power               = 80 %
```

```r
npwr.Utest(effect=2/3, alpha=0.05, alternative="einseitig", N=76)
## Stichprobenumfang n1= 38 und n2= 38
##  Effekt P(X>Y)       = 0.6666667
##  Signifikanzniveau   = 0.05
##  Alternativhypothese = einseitig
##  Power               = 80.83 %
```

Funktion noether() in library(rankFD)

```r
library(rankFD)
noether(alpha=0.10, power=0.80, t=0.5, p=2/3)
```

	Results
alpha (2-sided)	0.1000000
Power	0.8000000
relevant relative effect p	0.6666667
N (total sample size needed)	74.1906868
t=n1/N	0.5000000
n1 in Group 1	37.0953434
n2 in Group 2	37.0953434
N rounded	76.0000000
n1 rounded	38.0000000
n2 rounded	38.0000000

7.4.7 Wilcoxon-Paardifferenzentest

```
M1 <- c(0.47, 1.02, 0.33, 0.70, 0.94, 0.85, 0.39, 0.52, 0.47)
M2 <- c(0.41, 1.00, 0.46, 0.61, 0.84, 0.87, 0.36, 0.52, 0.51)
D  <- M1 - M2; D
## [1]  0.06  0.02 -0.13  0.09  0.10 -0.02  0.03  0.00 -0.04
wilcox.test(M1, M2, alternative="two.sided", paired=TRUE)
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  M1 and M2
## V = 22.5, p-value = 0.5749
## alternative hypothesis: true location shift is not equal to 0
```

Funktion wilcoxsign_test() in library(coin)

```
M1 <- c(0.47, 1.02, 0.33, 0.70, 0.94, 0.85, 0.39, 0.52, 0.47)
M2 <- c(0.41, 1.00, 0.46, 0.61, 0.84, 0.87, 0.36, 0.52, 0.51)

library(coin)
wilcoxsign_test(M1 ~ M2, alternative = "two.sided", distribution = exact())
##
##  Exact Wilcoxon-Pratt Signed-Rank Test
##
## data:  y by x (pos, neg)
##   stratified by block
## Z = 0.65331, p-value = 0.5469
## alternative hypothesis: true mu is not equal to 0
```

Pseudomedian:

```
pseudo.median <- function(x) {
    d <- sort(x); n <- length(d); W <- rep(NA, (n*(n+1))/2); k <- 0
    for (i in 1 : n) {
    for (j in i : n) {
           k <- k+1; W[k] <- (d[i]+d[j])/2 } }
    median(W)
}

data <- c(-2, 4, 8, 25, -5, 16, 3, 1, 12, 17, 20, 9)
pseudo.median(data); median(data)
## [1] 9
## [1] 8.5
```

Walsh averages:

```
data <- c(-2, 4, 8, 25, -5, 16, 3, 1, 12, 17, 20, 9)
Wmat <- outer(data, data, "+")/2
Walsh_Averages <- c(Wmat[lower.tri(Wmat)], diag(Wmat));
pseudo_Median  <- median(Walsh_Averages); pseudo_Median
## [1] 9
```

```r
Wilcoxon_Stat  <- length(Walsh_Averages[Walsh_Averages>0]); Wilcoxon_Stat
## [1] 71

n  <- length(data); Walsh_Averages <- sort(Walsh_Averages)
q1 <- (n*(n+1))/2 + 1 - qsignrank(0.975,n)
q2 <- qsignrank(0.975, n)
KI_u <- Walsh_Averages[q1]; KI_u
## [1] 3
KI_o <- Walsh_Averages[q2]; KI_o
## [1] 14.5

wilcox.test(data, conf.int=TRUE)
##
##  Wilcoxon signed rank test
##
## data:  data
## V = 71, p-value = 0.009277
## alternative hypothesis: true location is not equal to 0
## 95 percent confidence interval:
##   2.5 16.0
## sample estimates:
## (pseudo)median
##              9
```

7.4.7.3 Vorzeichentest von Dixon und Mood

```r
vz_test <- function(x=0, y=NULL, alternative="two.sided") {
  n <- sum((x-y)!=0)                                    # Differenzen ungleich Null
  T <- sum(x < y)
  if (alternative=="less")    p.val <- pbinom(T, n, 0.5)
  if (alternative=="greater") p.val <- 1-pbinom(T-1, n, 0.5)
  if (alternative=="two.sided")
    p.val <- 2*min(1-pbinom(T - 1, n, 0.5), pbinom(T, n, 0.5))
  cat("\n","Vorzeichentest zu",n,"Wertepaaren: T=",T,"; P =",p.val,"\n") }

ohne <- c(202, 182, 207, 184, 190, 197, 224, 171, 194, 203, 192, 215, 204, 178)
mit  <- c(211, 194, 200, 201, 204, 209, 230, 186, 206, 218, 192, 231, 199, 197)
as.data.frame(rbind(ohne=ohne, mit=mit))
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14
ohne	202	182	207	184	190	197	224	171	194	203	192	215	204	178
mit	211	194	200	201	204	209	230	186	206	218	192	231	199	197

```r
vz_test(ohne, mit, alternativ="two.sided")
##
##  Vorzeichentest zu 13 Wertepaaren: T= 11 ; P = 0.02246094
```

7.7.7.4 Fallzahlabschätzung

Vorzeichentest:

```
beta <- c(0.10, 0.20); alpha <- c(0.10, 0.05, 0.01)
p    <- c(3/5, 2/3, 3/4, 4/5); odds <- p / (1-p)
z.alpha <- qnorm(1-alpha); z.beta  <- qnorm(1-beta)

tab <- array(0, dim=c(4,3,2), dimnames=list(round(p, 2),alpha,beta))
for (i in 1:4) { for (j in 1:3) { for (k in 1:2)
  tab[i,j,k] <- ceiling((z.alpha[j] + z.beta[k])^2 / (4*(p[i] - 0.5)^2)) } }
out <- cbind(odds, tab[,,1], tab[,,2])
as.data.frame(out)
```

	odds	0.1	0.05	0.01	0.1	0.05	0.01
0.6	1.5	165	215	326	113	155	251
0.67	2.0	60	78	118	41	56	91
0.75	3.0	27	35	53	19	25	41
0.8	4.0	19	24	37	13	18	28

Wilcoxon Paardifferenzentest

```
beta <- c(0.10, 0.20); alpha <- c(0.10, 0.05, 0.01)
p    <- c(3/5, 2/3, 3/4, 4/5); odds <- p / (1-p)
z.alpha <- qnorm(1-alpha); z.beta  <- qnorm(1-beta)

tab <- array(0, dim=c(4,3,2), dimnames=list(round(p, 2),alpha,beta))
for (i in 1:4) { for (j in 1:3) { for (k in 1:2)
  tab[i,j,k] <- ceiling((z.alpha[j] + z.beta[k])^2 / (3*(p[i] - 0.5)^2)) } }
out <- cbind(odds, tab[,,1], tab[,,2])
as.data.frame(out)
```

	odds	0.1	0.05	0.01	0.1	0.05	0.01
0.6	1.5	219	286	434	151	207	335
0.67	2.0	79	103	157	55	75	121
0.75	3.0	36	46	70	25	33	54
0.8	4.0	25	32	49	17	23	38

7.4.8 Vergleich der Erwartungswerte aus Poisson-Verteilungen

Beispiel Salmonellen:

```
x <- 10;  n1 <- 100
y <- 2;   n2 <- 100

q0 <- 1; pq <- (n1/n2)*q0 / (1 + (n1/n2)*q0)

p1 <- binom.test(x, x+y, pq, alternative="greater")$p.value
p2 <- binom.test(x, x+y, pq, alternative="less")$p.value
p  <- 2*min(p1, p2)
p
## [1] 0.03857422
```

```
poisson.test(x=c(x, y), T=c(n1, n2), r=1, alternative="two.sided")
##
##  Comparison of Poisson rates
##
## data:  c(x, y) time base: c(n1, n2)
## count1 = 10, expected count1 = 6, p-value = 0.03857
## alternative hypothesis: true rate ratio is not equal to 1
## 95 percent confidence interval:
##   1.065528 46.932835
## sample estimates:
## rate ratio
##          5
```

Beispiel Rechnungsprüfung:

```
poisson.test(x=c(6, 16), T=c(500, 500), alternative="two.sided")
##
##  Comparison of Poisson rates
##
## data:  c(6, 16) time base: c(500, 500)
## count1 = 6, expected count1 = 11, p-value = 0.05248
## alternative hypothesis: true rate ratio is not equal to 1
## 95 percent confidence interval:
##  0.1201837 1.0089245
## sample estimates:
## rate ratio
##      0.375

poisson.test(x=c(5, 17), T=c(500, 500), alternative="two.sided")
##
##  Comparison of Poisson rates
##
## data:  c(5, 17) time base: c(500, 500)
## count1 = 5, expected count1 = 11, p-value = 0.0169
## alternative hypothesis: true rate ratio is not equal to 1
## 95 percent confidence interval:
##  0.08484139 0.83050821
## sample estimates:
```

```
## rate ratio
##  0.2941176
```

Beispiel Inzidenzrate:

```
x <- 50; n1 <- 3000
y <- 30; n2 <- 3000
l1 <- x/n1; l2 <- y/n2

z <- log(l1/l2) / (sqrt(1/x + 1/y)); z
## [1] 2.21194
p <- 2*pnorm(z, lower.tail=F);  p                    # zweiseitig
## [1] 0.02697082

poisson.test(x=c(x, y), T=c(n1, n2), r=1, alternative="two.sided")
##
##  Comparison of Poisson rates
##
## data:  c(x, y) time base: c(n1, n2)
## count1 = 50, expected count1 = 40, p-value = 0.03299
## alternative hypothesis: true rate ratio is not equal to 1
## 95 percent confidence interval:
##  1.039340 2.714805
## sample estimates:
## rate ratio
##   1.666667
```

Powerabschätzung:

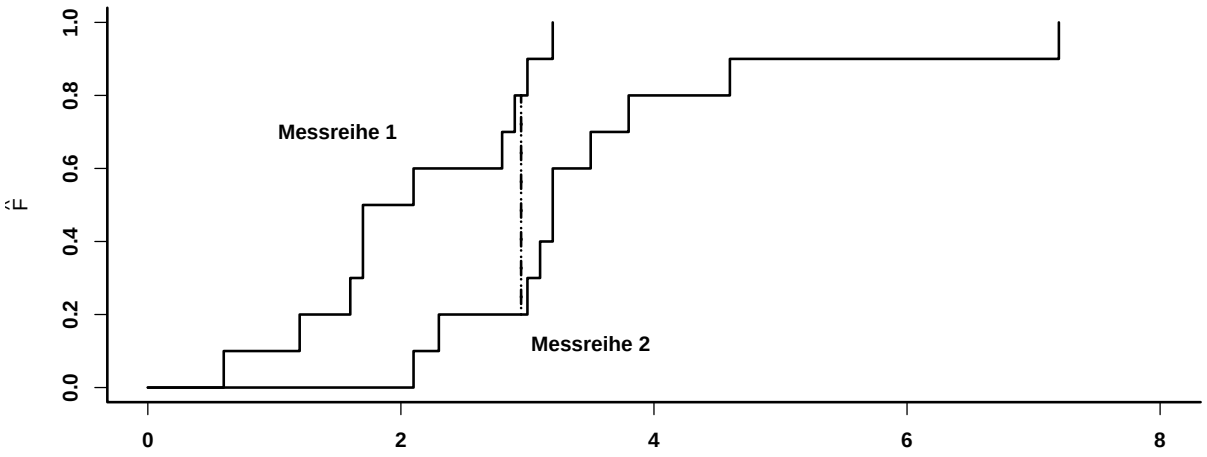
```
x <- 50; n1 <- 3000
y <- 30; n2 <- 3000
l1 <- x/n1; l2 <- y/n2

z.alpha <- z/2                                       # Power (zweiseitig)
z.beta  <- log(l1/l2) / (sqrt(1/x + 1/y)) - z.alpha
power   <- pnorm(z.beta, lower.tail=T); power
## [1] 0.8656302
```

Fallzahlabschätzung:

```
l1 <- 0.010; l2 <- 0.005; ratio <- l1/l2
alpha <- 0.05; z.alpha <- qnorm(1-alpha)             # einseitig
power <- 0.80; z.beta  <- qnorm(power)

x <- (1 + ratio)*((z.alpha + z.beta)/log(ratio))^2; y <- x / ratio
ceiling(x); ceiling(y)
## [1] 39
## [1] 20

n1 <- x / l1; n2 <- y / l2
ceiling(n1); ceiling(n2)
## [1] 3861
## [1] 3861
```

7.4.9 Kolmogoroff-Smirnoff Test

```
m1 <- c(2.1, 3.0, 1.2, 2.9, 0.6, 2.8, 1.6, 1.7, 3.2, 1.7)
m2 <- c(3.2, 3.8, 2.1, 7.2, 2.3, 3.5, 3.0, 3.1, 4.6, 3.2)

par(lwd=2, mfrow=c(1,1), font=2, font.axis=2, bty="l", ps=12)
n1   <- length(m1)
p1   <- cumsum(rep(1,n1)/n1)
plot(c(0,sort(m1)), c(0, p1), type="s",  xlim=c(0,8),
                xlab=" ", ylab=expression(hat(F)))
text(1.5, 0.7, "Messreihe 1", cex=1.0)
n2   <- length(m2)
p2   <- cumsum(rep(1,n2)/n2)
lines(c(0, sort(m2)), c(0,p2), type="s")
text(3.5, 0.12, "Messreihe 2",  cex=1.0)
polygon(c(2.95, 2.95), c(0.2, 0.8), lty=4)
```




```
ks.test(m1, m2, alternative="two.sided")
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  m1 and m2
## D = 0.6, p-value = 0.05465
## alternative hypothesis: two-sided
```

7.4.9.1 Cramer-vonMises Test

Beispiel 1:

```
m1  <- c(0.6, 1.2, 1.6, 1.7, 1.7, 2.1, 2.8, 2.9, 3.0, 3.2)
m2  <- c(2.1, 2.3, 3.0, 3.1, 3.2, 3.2, 3.5, 3.8, 4.6, 7.2)
n1  <- 10; n2 <- 10; x <- seq(0,8,by=0.1)
hm1 <- hist(m1, breaks=x, plot=F);  F <- cumsum(hm1$counts)/n1
hm2 <- hist(m2, breaks=x, plot=F);  G <- cumsum(hm2$counts)/n2

KS  <- max(abs(F-G)); KS                              # Kolmogoroff-Smirnoff-Test
## [1] 0.6

C   <- (n1*n2)/(n1+n2)^2 * sum((hm1$counts+hm2$counts)*((F-G)^2)); C
## [1] 0.875
```

Bootstrap Funktion cvm_test() in library(twosamples)

```
library(twosamples)
cvm_test(m1, m2)
## Test Stat   P-Value
##     2.320     0.006
## attr(,"details")
##      n1      n2 n.boots
##      10      10    2000
```

Beispiel 2:

```
X <- c(7.3, 7.9, 8.7, 9.0, 9.4, 10.2, 10.5, 11.1, 12.6)
Y <- c(4.3, 4.8, 5.2, 5.7, 6.0, 6.9, 8.0, 9.6, 11.8,
       12.8, 13.1, 13.4, 13.7, 14.5, 14.9)
n1 <- length(X); n2 <- length(Y); x <- seq(0, 15, by=0.1)
hX <-hist(X, breaks=x, plot=F);  F <- cumsum(hX$counts)/n1
hY <-hist(Y, breaks=x, plot=F);  G <- cumsum(hY$counts)/n2

KS <- max(abs(F-G)); KS                               # Kolmogoroff-Smirnoff-Test
## [1] 0.4

C  <- (n1*n2)/(n1+n2)^2 * sum((hX$counts+hY$counts)*((F-G)^2)); C
## [1] 0.3041667
```

Bootstrap Funktion cvm_test() in library(twosamples)

```
library(twosamples)
cvm_test(X, Y)
## Test Stat   P-Value
##  1.297778  0.124000
## attr(,"details")
##      n1      n2 n.boots
##       9      15    2000
```

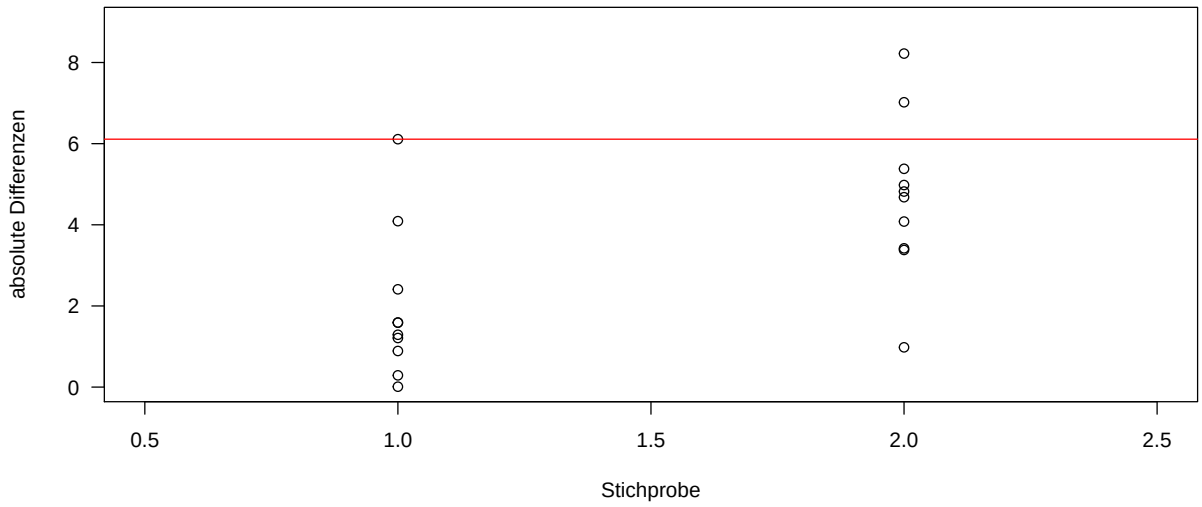
7.4.10 Weitere verteilungsunabhängige Verfahren

7.4.10.1 Count Five: Zweistichproben Dispersionstest

```
x <- c(2.4, 6.1, 7.3, 8.5, 8.8, 9.4,  9.8, 10.1, 10.1, 12.6)
y <- c(2.9, 3.3, 3.6, 4.2, 4.9, 7.3, 11.7, 13.1, 15.3, 16.5)

x.m <- mean(x);  y.m <- mean(y); n <- length(x)

max.y <- max(abs(y-y.m)); max.x <- max(abs(x-x.m))
                abs(x-x.m)>max.y; abs(y-y.m)>max.x
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
tab <- rbind(x, abs(x-x.m), y, abs(y-y.m),
                abs(x-x.m)>max.y, abs(y-y.m)>max.x)

C_x <- sum(tab[5,]); C_x
## [1] 0
C_y <- sum(tab[6,]); C_y
## [1] 2

colnames(tab) <- c(1:10)
rownames(tab) <- c("x","d_x","y","d_y","C_x","C_y")
as.data.frame(tab[1:6,])
```

	1	2	3	4	5	6	7	8	9	10
x	2.40	6.10	7.30	8.50	8.80	9.40	9.80	10.10	10.10	12.60
d_x	6.11	2.41	1.21	0.01	0.29	0.89	1.29	1.59	1.59	4.09
y	2.90	3.30	3.60	4.20	4.90	7.30	11.70	13.10	15.30	16.50
d_y	5.38	4.98	4.68	4.08	3.38	0.98	3.42	4.82	7.02	8.22
C_x	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C_y	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00

```
plot(c(rep(1,n),rep(2,n)), c(abs(x-x.m),abs(y-y.m)), las=1,
     xlim=c(0.5, 2.5), ylim=c(0,9),
     xlab="Stichprobe", ylab="absolute Differenzen")
abline(h=min(max.x, max.y), col="red")
```




```
ansari.test(x, y, alternative="two.sided")
##
##  Ansari-Bradley test
##
## data:  x and y
## AB = 70.5, p-value = 0.0183
## alternative hypothesis: true ratio of scales is not equal to 1
var.test(x, y, alternative="two.sided")
##
##  F test to compare two variances
##
## data:  x and y
## F = 0.26878, num df = 9, denom df = 9, p-value = 0.06348
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.06675995 1.08208719
## sample estimates:
## ratio of variances
##          0.2687752
```

7.4.10.3 Permutationstest, Randomisierungstest

Funktion oneway_test() in library(coin)

```
x1 <- c(20, 23, 30);                    n1 <- length(x1)
x2 <- c(27, 29, 35, 38, 40, 40, 45);    n2 <- length(x2)

sum(x1)                                    # Summe in der ersten Stichprobe
## [1] 73

choose(n1 + n2, n1)            # Anzahl möglicher Summen mit drei Summanden
## [1] 120
```

```
library(coin)
x <- c(x1,x2); grp <- as.factor(c(rep(1, n1), rep(2, n2)))
dat <- as.data.frame(c(grp, x))
oneway_test(x ~ grp, distribution="exact", data=dat, alternative="less")
##
##  Exact Two-Sample Fisher-Pitman Permutation Test
##
## data:  x by grp (1, 2)
## Z = -2.1212, p-value = 0.025
## alternative hypothesis: true mu is less than 0
```

Beispiel (a) - unabhängige Stichproben:

```
A <- c(65, 79, 90, 75, 61, 98, 80, 75); na <- length(A)
B <- c(90, 98, 73, 79, 84, 98, 90, 88); nb <- length(B)

T.o   <- sum(A); T.o                      # Teststatistik
## [1] 623
N     <- 500                              # Anzahl Wiederholungen
count <- 0

combine <- c(A, B)
for (i in 1:N) {
    if (sum(sample(combine, na)) <= T.o) count <- count + 1
    }
count/N                                  # P-Wert (einseitig)
## [1] 0.05
```

Funktion wilcox_test() in library(coin)

```
library(coin)
x <- c(A, B)
g <- as.factor(c(rep("A", length(A)), rep("B", length(B))))
wilcox_test(x ~ g, distribution = "exact", alternative="less")
##
##  Exact Wilcoxon-Mann-Whitney Test
##
## data:  x by g (A, B)
## Z = -1.5341, p-value = 0.06729
## alternative hypothesis: true mu is less than 0
```

Beispiel (b) - verbundene Stichproben:

```
prae <- c(90, 115,  98, 120, 93, 95, 102, 92)
post <- c(80,  95, 105, 110, 88, 92,  95, 88)
n    <- length(prae); stat <- numeric(n)

diff  <- prae - post
T.o   <- sum(diff); T.o                   # Teststatistik
## [1] 52
N     <- 500                              # Anzahl Wiederholungen
```

```
count <- 0

for (i in 1:N) {
    for (j in 1:n) stat[j]=ifelse(runif(1) < 0.5, diff[j], -diff[j])
    if (sum(stat) >= T.o) count <- count + 1
    }
count/N                                  # P-Wert (einseitig)
## [1] 0.028
```

Funktion wilcoxsign_test() in library(coin)

```
library(coin)
wilcoxsign_test(prae ~ post, alternative = "greater", distribution = exact())
##
##  Exact Wilcoxon-Pratt Signed-Rank Test
##
## data:  y by x (pos, neg)
##   stratified by block
## Z = 1.895, p-value = 0.03125
## alternative hypothesis: true mu is greater than 0
```

7.4.10.5 Mediantest

```
y1 <- c(2, 3, 4, 5, 4, 6, 7, 4, 3)
y2 <- c(7, 5, 6, 7, 6, 5, 4, 8, 9)

y <- c(y1, y2)
g <- as.factor(c(rep("I", length(y1)), rep("II", length(y2))))

tab <- matrix(c(6, 1, 3, 8),  nrow = 2,
       dimnames = list(Stichprobe = c("I", "II"), Median = c("<5", ">=5")))
as.data.frame(tab)
```

	<5	>=5
I	6	3
II	1	8

```
fisher.test(as.factor(y < median(y)), g)$p.value
## [1] 0.04977376
```

Funktion oneway_test() in library(coin)

```
y1 <- c(2, 3, 4, 5, 4, 6, 7, 4, 3)
y2 <- c(7, 5, 6, 7, 6, 5, 4, 8, 9)

y <- c(y1, y2)
g <- as.factor(c(rep("I", length(y1)), rep("II", length(y2))))
```

```r
library(coin)
dat <- as.data.frame(c(g, y))

oneway_test(y ~ g, distribution="exact", alternative="two.sided")
##
##  Exact Two-Sample Fisher-Pitman Permutation Test
##
## data:  y by g (I, II)
## Z = -2.3915, p-value = 0.01896
## alternative hypothesis: true mu is not equal to 0

median_test(y ~ g, conf.int = FALSE, distribution="exact")
##
##  Exact Two-Sample Brown-Mood Median Test
##
## data:  y by g (I, II)
## Z = -1.8439, p-value = 0.1534
## alternative hypothesis: true mu is not equal to 0
```

7.4.11 Zweistichprobentest auf Äquivalenz

```
                        # Quantile zur nichtzentralen Fisher-Verteilung
myqf <- function(p, df1, df2, ncp) {
uniroot(function(x) pf(x, df1, df2, ncp) - p, c(0, 100)) $ root }

x <- c(59.3, 58.8, 62.0, 42.6, 73.3, 54.2, 50.5, 38.0, 45.3, 50.0)
y <- c(34.9, 44.9, 52.0, 65.4, 52.5, 52.2, 68.6, 47.7, 55.9, 55.7, 53.5, 56.6)

m.x <- mean(x); s.x <- sd(x); m=length(x)
m.y <- mean(y); s.y <- sd(y); n=length(y)
T <- ((m.x - m.y) / sqrt(sum((x-m.x)^2)+ sum((y-m.y)^2)))*sqrt((m*n*(m+n-2))/(m+n));T
## [1] 0.0183523

eps <- 0.5                                              # kritischer Wert
c <- sqrt(myqf(0.05, 1, m+n-2, ncp=(m*n/(m+n))*eps^2))
c
## [1] 0.125252
```

Funktion tost() in library(equivalence)

```
x <- c(59.3, 58.8, 62.0, 42.6, 73.3, 54.2, 50.5, 38.0, 45.3, 50.0)
y <- c(34.9, 44.9, 52.0, 65.4, 52.5, 52.2, 68.6, 47.7, 55.9, 55.7, 53.5, 56.6)

library(equivalence)
tost(x, y, alpha = 0.05, epsilon=0.50)
##
##  Welch Two Sample TOST
##
## data:  x and y
## df = 17.741
## sample estimates:
## mean of x mean of y
##    53.400    53.325
##
## Epsilon: 0.5
## 95 percent two one-sided confidence interval (TOST interval):
##  -7.131907  7.281907
## Null hypothesis of statistical difference is: not rejected
## TOST p-value: 0.4598171
```

7.4.11.1 Test auf Bioäquivalenz

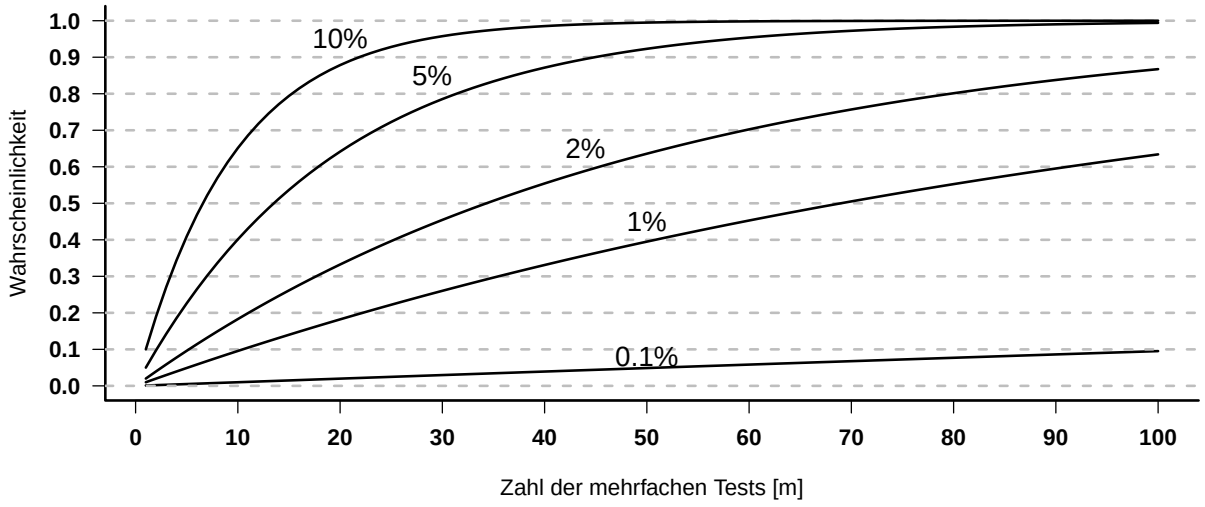
Beispiel Allopurinol:

```
R1 <- c(3.648, 8.531, 4.318, 6.974, 5.862, 3.082)
R2 <- c(4.894, 6.504, 7.372, 4.105, 2.368, 6.229)
T1 <- c(3.881, 4.835, 6.914, 5.236, 3.058, 5.722)
T2 <- c(3.671, 7.693, 4.481, 5.591, 5.311, 3.165)

RT <- log(R1) - log(T2); nRT <- length(RT); mRT <- mean(RT); sRT <- sd(RT)
TR <- log(R2) - log(T1); nTR <- length(TR); mTR <- mean(TR); sTR <- sd(TR)

mD <- (mRT + mTR)/2; mD
## [1] 0.044304
sD <- sqrt(((nRT-1)*sRT^2 + (nTR-1)*sTR^2)/(nRT + nTR -2)); sD
## [1] 0.1797106

alpha <- 0.05
l.u <- mD - qt(1-alpha, nTR + nRT -2)* (sD * sqrt((1/nRT + 1/nTR) * 0.5)); l.u
## [1] -0.08866999
l.o <- mD + qt(1-alpha, nTR + nRT -2)* (sD * sqrt((1/nRT + 1/nTR) * 0.5)); l.o
## [1] 0.177278
```

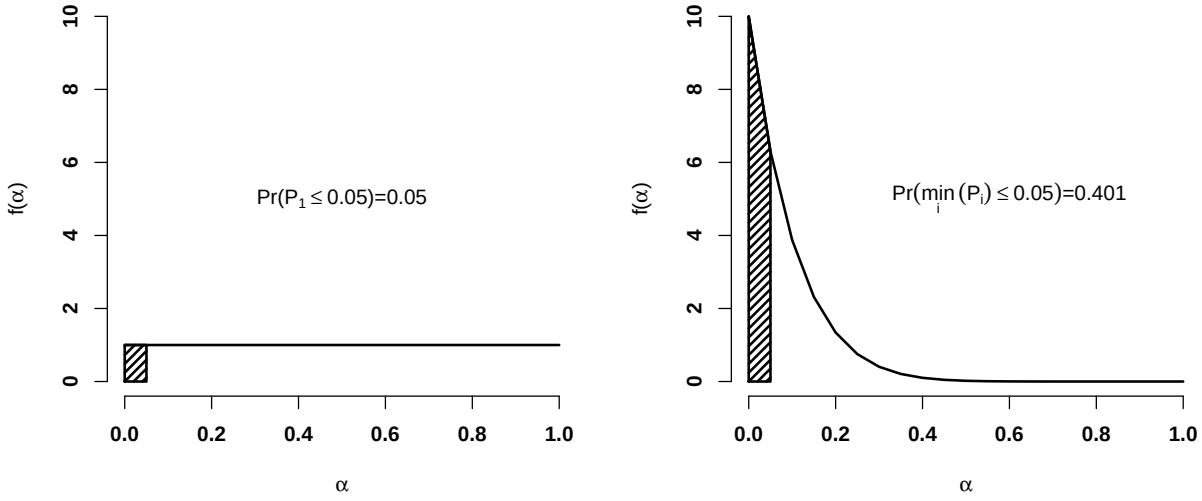
7.5 Mehrfacher Hypothesentest

7.5.1 Multiples Testproblem

```
m <- 1:100
p <- c(0.001, 0.01, 0.02, 0.05, 0.10)
y <- matrix(rep(NA, 5*100), nrow=5, byrow=T)
for (i in 1:5) y[i,] <- 1 - (1-p[i])^m

par(mfcol=c(1,1), lwd=2, font.axis=2, bty="l", ps=13)
plot(m, y[1,], ylim=c(0,1), type="l", las=1, lwd=2,
     xaxp=c(0,100,10), yaxp=c(0,1,10),
     xlab="Zahl der mehrfachen Tests [m]",
     ylab="Wahrscheinlichkeit")
abline(h=seq(0,1,0.1), lty=2, col="grey")
for (i in 2:5) lines(m, y[i,], lwd=2)
text(50, 0.08, "0.1%", cex=1.2); text(50, 0.45, "1%", cex=1.2)
text(44, 0.65, "2%", cex=1.2);   text(29, 0.85, "5%", cex=1.2)
text(20, 0.95, "10%", cex=1.2)
```




```
alpha <- seq(0, 1, by=0.05); m=10
p1    <- rep(1, length(alpha)); p2 <- m*(1-alpha)^(m-1)

par(mfrow=c(1,2), lwd=2, font.axis=2.5, bty="n", ps=12)
plot(alpha, p1, type ="l", ylim=c(0,10), xlim=c(0, 1),
     ylab=expression(paste("f(",alpha,")")), xlab=expression(alpha))
polygon(c(0,0,0.05,0.05), c(0,1,1,0), density=20)
text(0.5, 5, expression(paste("Pr(",P[1] <= 0.05,")=0.05")))
plot(alpha, p2, type ="l", ylim=c(0,10), xlim=c(0, 1),
     ylab=expression(paste("f(",alpha,")")), xlab=expression(alpha))
polygon(c(0,0,0.05,0.05), c(0,10,m*(1-0.05)^(m-1),0), density=20)
text(0.6, 5, expression(paste("Pr",(min((P[i]),i) <= 0.05),"=0.401")))
```



7.5.2 Adjustierung von P-Werten

```
test <- 1:5; m <- length(test)
p    <- c(0.011, 0.062, 0.015, 0.040, 0.002)
p.B  <- p.adjust(p, method="bonferroni")
R.p  <- rank(p)
p.H  <- p.adjust(p, method="holm")
p.SH <- p.adjust(p, method="hochberg")
p.BH <- p.adjust(p, method="BH")

adjust <- cbind(test, p, p.B, R.p, p.H, p.SH, p.BH)
as.data.frame(adjust)
```

test	p	p.B	R.p	p.H	p.SH	p.BH
1	0.011	0.055	2	0.044	0.044	0.025
2	0.062	0.310	5	0.080	0.062	0.062
3	0.015	0.075	3	0.045	0.045	0.025
4	0.040	0.200	4	0.080	0.062	0.050
5	0.002	0.010	1	0.010	0.010	0.010

7.5.3 Kombination von P-Werten

```
stouffer_test <- function(p, w) {                    # Stouffer-Verfahren
  if (missing(w)) w <- rep(1, length(p))/length(p)
  else if (length(w)!=length(p)) stop("p und w unterschiedlich lang (?)")
  zi <- qnorm(1-p); z <- sum(w*zi)/sqrt(sum(w^2))
  P.Wert <- 1 - pnorm(z)
  cat("z =",z," und kombinierter P-Wert =",P.Wert,"\n")
}

stouffer_test(p=c(0.04, 0.07, 0.10))
## z = 2.602712  und kombinierter P-Wert = 0.004624487
```