# AS17: Kapitel 7.6 - 7.8 Hypothesentest

Jürgen Hedderich

2020-04-27

## 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:

Wiederholte Messungen: repeated

Übereinstimmung (Bland-Altman): blandcor

## 7.6 Mehrstichprobenverfahren

### 7.6.1 Prüfung der Gleichheit mehrerer Varianzen

#### 7.6.1.3 Bartlett-Verfahren

```
x <- c( 9, 11,  6, 11, 14,  7,  7, 11)                    # Bartlett-Test
y <- c(13, 10, 12, 16, 11, 13, 15,  9,  9, 10)
z <- c( 7, 27,  8, 11, 17,  2, 16, 15,  9, 15, 18, 12)

k   <- 3
si  <- c(sd(x), sd(y), sd(z)); si
## [1] 2.725541 2.440401 6.444989
nui <- c(length(x)-1, length(y)-1, length(z)-1); nu <- sum(nui)
c   <- (sum(1/nui)- 1/nu)/(3*(k-1)) +1
ssqr <- sum(nui*si^2)/nu
chisqr <- 1/c*(2.3026 * (nu*log10(ssqr)-sum(nui*log10(si^2)))); chisqr
## [1] 10.36702
qchisq(0.95, k-1)
## [1] 5.991465
pchisq(chisqr, k-1, lower.tail=F)
## [1] 0.005608289

bartlett.test(list(x,y,z))
##
##  Bartlett test of homogeneity of variances
##
## data:  list(x, y, z)
## Bartlett's K-squared = 10.367, df = 2, p-value = 0.005608
```

#### 7.6.1.4 Levene-Test (Brown-Forsythe-Version)

Funktionen leveneTest() in library(car)

```
library(car)
val  <- c(x, y, z)
grp  <- as.factor(c(rep("I", length(x)),
                    rep("II", length(y)), rep("III", length(z))))

leveneTest(val ~ grp)                                    # Levene-test
```

| | Df | F value | Pr(>F) |
|---|---|---|---|
| group | 2 | 3.904317 | 0.0324068 |
| | 27 | NA | NA |

```
fligner.test(val ~ grp)                                  # Fligner-Test
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  val by grp
## Fligner-Killeen:med chi-squared = 7.3235, df = 2, p-value = 0.02569
```

### 7.6.3 Varianzanalyse (ANOVA)

```
gruppe <- c(1, 1, 2, 2, 2, 2, 3, 3, 3)                  # Funktion aov()
wert   <- c(3, 7, 4, 2, 7, 3, 8, 4, 6)
daten  <- data.frame(gruppe=factor(gruppe), wert);


summary(aov(wert ~ gruppe, data=daten))
##             Df Sum Sq Mean Sq F value Pr(>F)
## gruppe       2  6.889   3.444   0.689  0.538
## Residuals    6 30.000   5.000
```

Beispiel (gleichgroße Stichprobenumfänge):

```
gruppe <- c(rep(1,4), rep(2,4), rep(3,4))               # Beispiel
wert   <- c(6, 7, 6, 5, 5, 6, 4, 5, 7, 8, 5, 8)
daten  <- data.frame(gruppe=factor(gruppe), wert)

summary(aov(wert ~ gruppe, daten))
##             Df Sum Sq Mean Sq F value Pr(>F)
## gruppe       2      8   4.000     3.6  0.071 .
## Residuals    9     10   1.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#### 7.6.3.1 Permutationstest zur Varianzanalyse

```
aov_permute <- function(grp, val, B=499) {
  grp <- factor(grp)
  n   <- length(val)
  obs <- anova(lm(val ~ grp))$F[1]
  res <- numeric(B)
  for (i in 1:B) {
    index    <- sample(n);     val.perm <- val[index]
    res[i]   <- anova(lm(val.perm ~ grp))$F[1]  }
  p <- (sum(res > obs) + 1) / (B+1)
  cat("ANOVA-Permutationstest P=",p,"\n")
}

aov_permute(gruppe, wert)
## ANOVA-Permutationstest P= 0.062
```

#### 7.6.3.2 Stichprobenumfänge und Power

```
npwr.ANOVA <- function(effect, groups, alpha=0.05, power=NULL, n=NULL) {
  if (sum(sapply(list(n, power), is.null)) != 1)
    stop("Fehler: Nur eins von n oder power darf NULL sein")
  f.distr <- quote({
    lambda <- effect^2 * n * groups
    q.alpha <- qf(alpha, groups-1, (n-1)*groups, lower.tail=FALSE)
    pf(q.alpha, groups-1, (n-1)*groups, lambda, lower.tail=FALSE) })
  if (is.null(n)) {
    n <- uniroot(function(n) eval(f.distr)-power, c(2, 1e+05))$root  }
  if (is.null(power)) power <- eval(f.distr)
  cat("Stichprobenumfang n  =",round(n, 0),"\n",
      "Anzahl der Gruppen  =",groups,"\n",
      "Effekt f            =",effect,"\n",
      "Signifikanzniveau   =",alpha,"\n",
      "Power               =",round(power*100,2),"%")
}

npwr.ANOVA(effect=0.353, groups=5, alpha=0.05, power=0.80)
## Stichprobenumfang n  = 20
##  Anzahl der Gruppen  = 5
##  Effekt f            = 0.353
##  Signifikanzniveau   = 0.05
##  Power               = 80 %
```

### 7.6.4 Multiple Vergleiche

#### 7.6.4.1 Multiple Vergleich nach Tukey-Kramer

Obere Schranken der SR-Verteilung:

```
p <- 0.05
k <- 2:12
v <- c(2:40, 50, 60, 70, 80, 90, 100, 10000)

tab <- matrix(data = NA, nrow = 46, ncol = 11,
              byrow = FALSE, dimnames = NULL)
for (i in 1:46) tab[i,] <- round(qtukey(0.95, k, v[i]), 2)
tab <- cbind(v, tab)
colnames(tab) <- c("v", 2:12)
as.data.frame(tab[1:10,])
```

| v | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 6.08 | 8.33 | 9.80 | 10.88 | 11.73 | 12.43 | 13.03 | 13.54 | 13.99 | 14.40 | 14.76 |
| 3 | 4.50 | 5.91 | 6.82 | 7.50 | 8.04 | 8.48 | 8.85 | 9.18 | 9.46 | 9.72 | 9.95 |
| 4 | 3.93 | 5.04 | 5.76 | 6.29 | 6.71 | 7.05 | 7.35 | 7.60 | 7.83 | 8.03 | 8.21 |
| 5 | 3.64 | 4.60 | 5.22 | 5.67 | 6.03 | 6.33 | 6.58 | 6.80 | 6.99 | 7.17 | 7.32 |
| 6 | 3.46 | 4.34 | 4.90 | 5.30 | 5.63 | 5.90 | 6.12 | 6.32 | 6.49 | 6.65 | 6.79 |
| 7 | 3.34 | 4.16 | 4.68 | 5.06 | 5.36 | 5.61 | 5.82 | 6.00 | 6.16 | 6.30 | 6.43 |
| 8 | 3.26 | 4.04 | 4.53 | 4.89 | 5.17 | 5.40 | 5.60 | 5.77 | 5.92 | 6.05 | 6.18 |
| 9 | 3.20 | 3.95 | 4.41 | 4.76 | 5.02 | 5.24 | 5.43 | 5.59 | 5.74 | 5.87 | 5.98 |
| 10 | 3.15 | 3.88 | 4.33 | 4.65 | 4.91 | 5.12 | 5.30 | 5.46 | 5.60 | 5.72 | 5.83 |
| 11 | 3.11 | 3.82 | 4.26 | 4.57 | 4.82 | 5.03 | 5.20 | 5.35 | 5.49 | 5.61 | 5.71 |

Beispiel Antibiotika elementar:

```
A <- c(27, 27, 25, 26, 25); nA <- length(A)
B <- c(26, 25, 26, 25, 24); nB <- length(B)
C <- c(21, 21, 20, 20, 22); nC <- length(C)
grp <- c(rep("A", nA), rep("B", nB), rep("C", nC))
d   <- data.frame(Gruppe = factor(grp), Wert = c(A, B, C))

f  <- nA + nB + nC - 3
mA <- mean(A);   mB <- mean(B);   mC <- mean(C)
s  <- sqrt((sum((A-mA)^2)+sum((B-mB)^2)+sum((C-mC)^2)) / f)

T.AB <- (mA - mB) / (s*sqrt(0.5*(1/nA + 1/nB))); T.AB
## [1] 2
T.AC <- (mA - mC) / (s*sqrt(0.5*(1/nA + 1/nC))); T.AC
## [1] 13
T.BC <- (mB - mC) / (s*sqrt(0.5*(1/nB + 1/nC))); T.BC
## [1] 11

q    <- qtukey(0.95, 3, f); q
## [1] 3.7T2929
```

Beispiel Antibiotika mit Funktion aov() und TukeyHSD()::

```r
A <- c(27, 27, 25, 26, 25); nA <- length(A)
B <- c(26, 25, 26, 25, 24); nB <- length(B)
C <- c(21, 21, 20, 20, 22); nC <- length(C)
grp <- c(rep("A", nA), rep("B", nB), rep("C", nC))
d   <- data.frame(Gruppe = factor(grp), Wert = c(A, B, C))

model <- aov(Wert ~ Gruppe, data = d);
TukeyHSD(model)
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Wert ~ Gruppe, data = d)
##
## $Gruppe
##     diff       lwr        upr     p adj
## B-A -0.8 -2.309172  0.7091716 0.3647720
## C-A -5.2 -6.709172 -3.6908284 0.0000025
## C-B -4.4 -5.909172 -2.8908284 0.0000139
```

Beispiel Antibiotika:

Funktion glht() in library(multcomp)

```r
library(multcomp)
grp <- c(rep("A", nA), rep("B", nB), rep("C", nC))
d   <- data.frame(Gruppe = factor(grp), Wert = c(A, B, C))

model <- aov(Wert ~ Gruppe, data = d);
summary(glht(model, linfct = mcp(Gruppe = "Tukey")))       # summary(model)
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = Wert ~ Gruppe, data = d)
##
## Linear Hypotheses:
##            Estimate Std. Error t value Pr(>|t|)
## B - A == 0  -0.8000     0.5657  -1.414    0.365
## C - A == 0  -5.2000     0.5657  -9.192   <0.001 ***
## C - B == 0  -4.4000     0.5657  -7.778   <0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Funktion confint() in library(multcomp)

```r
confint(glht(model, linfct = mcp(Gruppe = "Tukey")))
##
##   Simultaneous Confidence Intervals
##
```

```
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = Wert ~ Gruppe, data = d)
##
## Quantile = 2.6684
## 95% family-wise confidence level
##
##
## Linear Hypotheses:
##            Estimate lwr     upr
## B - A == 0 -0.8000  -2.3095  0.7095
## C - A == 0 -5.2000  -6.7095 -3.6905
## C - B == 0 -4.4000  -5.9095 -2.8905

mA-mB + qtukey(0.95, 3, f)*s*sqrt(0.5*(1/nA + 1/nB))       # Berechnung direkt ...
## [1] 2.309172

mA-mB - qtukey(0.95, 3, f)*s*sqrt(0.5*(1/nA + 1/nB))
## [1] -0.7091716
```

#### 7.6.4.3 Multiple Vergleich nach Dunnett

Beispiel Blutzellen:

```
Kontrolle <- c(7.40, 8.50, 7.20, 8.24, 9.84, 8.32)
Praep.A   <- c(9.76, 8.80, 7.68, 9.36)
Praep.B   <- c(12.80, 9.68, 12.16, 9.20, 10.55)

n0 <- length(Kontrolle); nA <- length(Praep.A); nB <- length(Praep.B)
f  <- n0+nA+nB-(3+1)
m0 <- mean(Kontrolle);   mA <- mean(Praep.A);   mB <- mean(Praep.B)
s  <- sqrt((sum((Kontrolle-m0)^2)+sum((Praep.A-mA)^2)+sum((Praep.B-mB)^2)) / f)

D.A  <- (mA - m0) / (s*sqrt(1/nA + 1/n0)); D.A
## [1] 0.8205458
D.B  <- (mB - m0) / (s*sqrt(1/nB + 1/n0)); D.B
## [1] 3.536499

R    <- sqrt(nA/(n0+nA)) * sqrt(nB/(n0+nB))
cR   <- matrix(c(1, R, R, 1), nrow=2); round(cR,2)
##      [,1] [,2]
## [1,] 1.00 0.43
## [2,] 0.43 1.00

library(mvtnorm)                              # Funktion qmvt() in library(mvtnorm)
qmvt(0.95, tail ="both.tail", df = f, corr = cR)$quantile
## [1] 2.543653
```

Funktion contrMat() in library(multcomp)

```
library(multcomp)
grp <- c(rep("grp.0", n0), rep("grp.1", nA), rep("grp.2", nB))
d   <- data.frame(Gruppe = factor(grp), Wert = c(Kontrolle, Praep.A, Praep.B))

n   <- c(n0, nA, nB); names(n) <- paste("Gruppe", c("K", "A", "B"), sep=".")
K   <- contrMat(n, type="Dunnett", base=1); K
##
##   Multiple Comparisons of Means: Dunnett Contrasts
##
##                     Gruppe.K Gruppe.A Gruppe.B
## Gruppe.A - Gruppe.K       -1        1        0
## Gruppe.B - Gruppe.K       -1        0        1

model <- aov(Wert ~ Gruppe, data = d)
summary(glht(model, linfct = mcp(Gruppe = K), alternative = "greater"))
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: User-defined Contrasts
##
##
## Fit: aov(formula = Wert ~ Gruppe, data = d)
##
```

```
## Linear Hypotheses:
##                            Estimate Std. Error t value  Pr(>t)
## Gruppe.A - Gruppe.K <= 0   0.6500     0.7584   0.857 0.32498
## Gruppe.B - Gruppe.K <= 0   2.6280     0.7115   3.694 0.00291 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Konfidenzintervalle:

```
confint(glht(model, linfct = mcp(Gruppe = K), alternative = "two.sided"))
##
##   Simultaneous Confidence Intervals
##
## Multiple Comparisons of Means: User-defined Contrasts
##
##
## Fit: aov(formula = Wert ~ Gruppe, data = d)
##
## Quantile = 2.5136
## 95% family-wise confidence level
##
##
## Linear Hypotheses:
##                            Estimate lwr     upr
## Gruppe.A - Gruppe.K == 0  0.6500  -1.2564  2.5564
## Gruppe.B - Gruppe.K == 0  2.6280   0.8396  4.4164
```

#### 7.6.4.4 Auswahl des Besten nach HSU bei gleichen Stichprobenumfängen

Obere Schranken der Dunett-Verteilung (Tabelle): Längere Laufzeit.....

Funktion qDunnett() in library(mvtnorm)

```
library(mvtnorm)                                        # Quantile zur Dunnett-Verteilung
qDunnett <-function(p, df, k, tail="both.tails") {
  m    <- k-1
  R    <- matrix(0.5, m, m)                             # Korrelationsmatrix bei gleichen
  for (i in 1:m) R[i, i] <- 1                           # Stichprobenumfängen
  temp <- qmvt(p, interval=c(0, 7), tail=tail, df=df, corr=R)[1]
  return(temp$quantile)
}

fg <- c(5,7,10,12,14,16,20,24,28,30,40,50,60,80,120,5000)
rc <- length(fg)
kn <- 2:8
cc <- length(kn)

# tab <- matrix(rep(0, rc*cc), nrow=rc, dimnames=list(fg,kn))
# for (i in 1:rc) {                                      # alpha = 0.10 zweiseitig
#   for (j in 1:cc) tab[i,j] <- round(qDunnett(0.90, df=fg[i], k=kn[j]), 3)  }
# tab <- cbind(fg, tab); colnames(tab) <- c("FG", 2:8)
# as.data.frame(tab[1:5,])
#
# tab <- matrix(rep(0, rc*cc), nrow=rc, dimnames=list(fg,kn))
# for (i in 1:rc) {                                      # alpha = 0.05 zweiseitig
#   for (j in 1:cc) tab[i,j] <- round(qDunnett(0.95, df=fg[i], k=kn[j]), 3)  }
# tab <- cbind(fg, tab); colnames(tab) <- c("FG", 2:8)
# as.data.frame(tab[1:5,])
```

Beispiel Insektenfallen:

```
insects <- matrix(c(45, 59, 48, 46, 38, 47,
                    21, 12, 14, 17, 13, 17,
                    37, 32, 15, 25, 39, 41,
                    16, 11, 20, 21, 14,  7), byrow=F, nrow=6,
                  dimnames=list(1:6,c("gelb","weiss","rot","blau")))
d <- data.frame(insects); attach(d)
as.data.frame(d)
```

| gelb | weiss | rot | blau |
|---|---|---|---|
| 45 | 21 | 37 | 16 |
| 59 | 12 | 32 | 11 |
| 48 | 14 | 15 | 20 |
| 46 | 17 | 25 | 21 |
| 38 | 13 | 39 | 14 |
| 47 | 17 | 41 | 7 |

```r
gefangen <- c(gelb, weiss, rot, blau)
farbe    <- as.factor(c(rep("gelb",6),rep("weiss",6),rep("rot",6),rep("blau",6)))
anova    <- summary(aov(gefangen ~ farbe)); anova
##             Df Sum Sq Mean Sq F value   Pr(>F)
## farbe        3   4218    1406   30.55 1.15e-07 ***
## Residuals   20    920      46
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
MSE      <- anova[[1]][2,3]; MSE
## [1] 46.025

mi <- apply(insects, 2, mean); mi                          # Mittelwerte nach den Farben
##     gelb    weiss      rot     blau
## 47.16667 15.66667 31.50000 14.83333
n  <- nrow(insects); k  <- ncol(insects)
MSE  <- 0                                                  # berechne MSE (error)
for (i in 1:k) {for (j in 1:n) MSE <- MSE + (insects[j,i]-mi[i])^2}
MSE  <- MSE/(k*(n-1))

############## qd   <- qDunnett(0.95, df=k*(n-1), k, tail="lower.tail") ################
qd   <- 2.191
d    <- qd * sqrt(2*MSE/n)                                 # Distanz mit Dunnett-Quantil

for (i in 1:k) {                                           # Beschränkte Konfidenzintervalle
  m <- mi[i]; mmax <- max(mi[-i])
  lower.ci <- m - mmax - d; lower.cic <- min(lower.ci, 0)
  upper.ci <- m - mmax + d; upper.cic <- max(upper.ci, 0)
  cat("\n\t", round(lower.cic, 3),"\ - \t",round(upper.cic,3),
      "für", colnames(insects)[i]) }
##
##   0  -    24.248 für gelb
##   -40.082  -      0 für weiss
##   -24.248  -      0 für rot
##   -40.915  -      0 für blau
```

#### 7.6.4.6 Maximum-Modulus-Ansatz

Studentisierte Maximum-Modulus-Verteilung (SMM): längere Laufzeit.....

Funktion qmvt() und qmvnorm() in library(mvtnorm)

```
# library(mvtnorm)
# fg <- c(5,10,15,20,25,30,40,50,75,100,200,1000); l <- length(fg)
# k  <- c(1:10, 15, 20);                           m <- length(k)
#
# tab1  <- matrix(rep(NA, l * m), nrow=l, byrow=T)
# alpha <- 0.05
# for (i in 1:(l-1)) {                                  # multivariate t-Verteilung
#     for (j in 1:m) {
# tab1[i, j] <- qmvt(1-alpha, df = fg[i], tail = "both",
#                    corr=diag(k[j]))$quantile              }   }
# for (j in 1:m) {                                      # multivariate Normalvert.
# tab1[l, j] <- qmvnorm(1.alpha,  mean=rep(0,k[j]), tail = "both",
#                       sigma=diag(k[j]))$quantile              }
# tab1 <- cbind(fg, tab1); row <- c(NA, round(k, 0));
# tab1 <- rbind(row, tab1)
# tab1
#
# tab2 <- matrix(rep(NA, l * m), nrow=l, byrow=T)
# alpha <- 0.01
# for (i in 1:(l-1)) {                                  # multivariate t-Verteilung
#     for (j in 1:m) {
# tab2[i, j] <- qmvt(1-alpha, df = fg[i], tail = "both",
#                    corr=diag(k[j]))$quantile     }   }
# for (j in 1:m) {                                      # multivariate Normalvert.
# tab2[l, j] <- qmvnorm(1-alpha,  mean=rep(0,k[j]), tail = "both",
#                       sigma=diag(k[j]))$quantile         }
# tab2 <- cbind(fg, tab2); row <- c(NA, round(k, 0))
# tab2 <- rbind(row, tab2)
# tab2
```

Beispiel Arbeitsunfähigkeit:

```
grp <- c("A","B","C","D","E","F")
n.i <- c( 74, 13, 15, 47, 23, 28); N <- sum(n.i)
m.i <- c(14.30, 13.65, 19.57, 16.91, 13.38, 15.89)

mean   <- sum(n.i*m.i)/N;  s  <- 15.26; mean; s
## [1] 15.38315
## [1] 15.26

d.i  <- round(m.i - mean, 2); sd.i  <- round(sqrt(s/n.i * (N-n.i)/N), 2)
quot <- round(abs(d.i) / sd.i, 2)
SMM  <- rep(NA, 6); k <- rep(NA, 6)

tab <- as.data.frame(cbind(grp, n.i, m.i, d.i, sd.i, quot, SMM, k))
names(tab) <- c("Gruppe","Anzahl","Mittelwert","Differenz",
                "Stdabw","Quotient","SMM","k")
```

```
I <- order(tab$Quotient, decreasing=T); tab <- tab[I, ]    # sortieren

SMM <- c(2.66, 2.59, 2.51, 2.40, 2.25, 1.97)              # k=6,...,1 | FG=n-6=194
tab[,8] <- 6:1; tab[,7] <- SMM; tab
```

| | Gruppe | Anzahl | Mittelwert | Differenz | Stdabw | Quotient | SMM | k |
|---|---|---|---|---|---|---|---|---|
| 3 | C | 15 | 19.57 | 4.19 | 0.97 | 4.32 | 2.66 | 6 |
| 4 | D | 47 | 16.91 | 1.53 | 0.5 | 3.06 | 2.59 | 5 |
| 1 | A | 74 | 14.3 | -1.08 | 0.36 | 3 | 2.51 | 4 |
| 5 | E | 23 | 13.38 | -2 | 0.77 | 2.6 | 2.40 | 3 |
| 2 | B | 13 | 13.65 | -1.73 | 1.05 | 1.65 | 2.25 | 2 |
| 6 | F | 28 | 15.89 | 0.51 | 0.68 | 0.75 | 1.97 | 1 |

#### 7.6.4.7 Lineare Kontraste nach Scheffe

```
x <- c( 4,  8, 11, 14, 10,  9, 11,  6); mean(x)
## [1] 9.125

y <- c(17, 10, 11, 13, 14,  9, 11, 12, 12,  8); mean(y)
## [1] 11.7

z <- c(12, 16, 11, 12, 17, 22, 12, 16, 17, 13, 19, 12); mean(z)
## [1] 14.91667

grp    <- c(rep(1,8), rep(2,10), rep(3,12))
wert   <- c(x, y, z)
daten  <- data.frame(grp=factor(grp), wert)
aov.mod  <- aov(wert ~ grp, daten); summary(aov.mod)
##             Df Sum Sq Mean Sq F value  Pr(>F)
## grp          2  166.4   83.20   8.644 0.00125 **
## Residuals   27  259.9    9.63
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

se.contrast(aov.mod, list(grp=="1", grp=="2", grp=="3"), coef=c(-1, 0, 1))
## [1] 1.416099
```

Funktion fit.contrast() aus library(gmodels)

```
library(gmodels)
fit.contrast(aov.mod, grp, c(-1, 0,  1))
##                  Estimate Std. Error  t value     Pr(>|t|)
## grp c=( -1 0 1 ) 5.791667   1.416099 4.089874 0.0003487793
## attr(,"class")
## [1] "fit_contrast"
```

### 7.6.5 H-Test von Kruskal und Wallis

Kritische Schranken mit Funktion qKruskalWallis() aus library(SuppDists):

```
library(SuppDists)
k <- c(3,4,5,6)
n <- c(3:10, 12, 14, 16, 18, 20, 25, 30, 40, 50, 1000)
alpha <- c(0.10, 0.05, 0.01)

matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
##      [,1]
## [1,]   NA
tab     <- matrix(NA, nrow = 20, ncol = 14, byrow = TRUE)
tab[1,] <- c(NA, rep(0.10, 4), rep(0.05,4), rep(0.01, 4), NA)
tab[2,] <- c(NA, rep(c(3,4,5,6), 3), NA)

for (i in 3:20) {
   tab[i,] <- c(n[i-2],
   round(qKruskalWallis(0.90, k, k*n[i-2], k*(1/n[i-2])), 3),
   round(qKruskalWallis(0.95, k, k*n[i-2], k*(1/n[i-2])), 3),
   round(qKruskalWallis(0.99, k, k*n[i-2], k*(1/n[i-2])), 3), n[i-2]) }
as.data.frame(tab[3:10, 1:6])
```

| V1 | V2 | V3 | V4 | V5 | V6 |
|---|---|---|---|---|---|
| 3 | 4.472 | 5.912 | 7.286 | 8.618 | 5.292 |
| 4 | 4.543 | 6.039 | 7.452 | 8.815 | 5.546 |
| 5 | 4.569 | 6.098 | 7.535 | 8.917 | 5.668 |
| 6 | 4.581 | 6.132 | 7.585 | 8.980 | 5.738 |
| 7 | 4.588 | 6.154 | 7.618 | 9.022 | 5.783 |
| 8 | 4.593 | 6.169 | 7.641 | 9.052 | 5.815 |
| 9 | 4.595 | 6.180 | 7.659 | 9.075 | 5.838 |
| 10 | 4.597 | 6.189 | 7.672 | 9.092 | 5.856 |

Beispiel mit Funktion kruskal.test():

```
A <- c(12.1, 14.8, 15.3, 11.4, 10.8)
B <- c(18.3, 49.6, 10.1, 35.6, 26.2, 8.9)
C <- c(12.7, 25.1, 47.0, 16.3, 30.4)
D <- c( 7.3,  1.9,  5.8, 10.1,  9.4)

x <- c(A, B, C, D)
g <- factor(rep(1:4, c(5, 6, 5,5)),labels = c("A","B","C","D"))

kruskal.test(x, g)
##
##  Kruskal-Wallis rank sum test
##
## data:  x and g
## Kruskal-Wallis chi-squared = 11.53, df = 3, p-value = 0.009179
```

#### 7.6.5.1 Multiple paarweise Vergleiche mittlerer Ränge

Beispiel mit Funktion kruskalmc() aus library(pgirmess):

```
library(pgirmess)
demo <- data.frame(y = c(28, 30, 33, 35, 38, 41,
                         36, 39, 40, 43, 45, 50,
                         44, 45, 47, 49, 53, 54),
                   gruppe = factor(c(rep("A",6),rep("B",6),rep("C",6))))

kruskalmc(y ~ gruppe, data=demo)
## Multiple comparison test after Kruskal-Wallis
## p.value: 0.05
## Comparisons
##       obs.dif critical.dif difference
## A-B  5.583333     7.378741      FALSE
## A-C 10.416667     7.378741       TRUE
## B-C  4.833333     7.378741      FALSE
```

Beispiel mit Funktion oneway_test() aus library(coin):

```
library(coin)
demo <- data.frame(y = c(28, 30, 33, 35, 38, 41,
                         36, 39, 40, 43, 45, 50,
                         44, 45, 47, 49, 53, 54),
                   gruppe = factor(c(rep("A",6),rep("B",6),rep("C",6))))

Nemenyi <- oneway_test(y ~ gruppe, data = demo,
           ytrafo = function(data) trafo(data, numeric_trafo = rank),
           xtrafo = function(data) trafo(data, factor_trafo = function(x)
           model.matrix(~x - 1) %*% t(contrMat(table(x), "Tukey"))),
           teststat = "max")
Nemenyi
##
##  Asymptotic K-Sample Fisher-Pitman Permutation Test
##
## data:  y by gruppe (A, B, C)
## chi-squared = 11.453, df = 2, p-value = 0.003258

drop(pvalue(Nemenyi, method = "single-step"))
## [1] 0.003257908
```

Nemenyi-Damico-Wolfe-Dunn Test - Funktion indepence_test() aus library(coin):

```
library(coin)
demo <- data.frame(y = c(28, 30, 33, 35, 38, 41,
                         36, 39, 40, 43, 45, 50,
                         44, 45, 47, 49, 53, 54),
                   gruppe = factor(c(rep("A",6),rep("B",6),rep("C",6))))

kw <- kruskal_test(y ~ gruppe, data = demo,
                   distribution = approximate(nresample=9999)); kw
```

```
##
##  Approximative Kruskal-Wallis Test
##
## data:  y by gruppe (A, B, C)
## chi-squared = 11.453, p-value = 4e-04

it <- independence_test(y ~ gruppe, data = demo,
                        distribution = approximate(nresample = 50000),
                        ytrafo = function(data) trafo(data, numeric_trafo = rank_trafo),
                        xtrafo = function(data) trafo(data, factor_trafo = function(x)
                          model.matrix(~x - 1) %*% t(contrMat(table(x), "Tukey")))); it
##
##  Approximative General Independence Test
##
## data:  y by gruppe (A, B, C)
## maxT = 3.3814, p-value = 0.00028
## alternative hypothesis: two.sided

pvalue(it, method = "single-step")
##
## B - A 0.17276
## C - A 0.00028
## C - B 0.27752
```

#### 7.6.5.4 Trendtest nach Jonckheere

```
jonckheere.test<-function(x, g)  {                              # Jonckheere-Terpstra Test
    x <- table(g,x); nco <- ncol(x);  nro <- nrow(x);  summe <- 0
    for(j in 1:(nco - 1))
      for(i in 1:(nro - 1))
       summe <- summe + x[i, j] * (0.5 * sum(x[(i + 1):nro, j]) +
                                          sum(x[(i + 1):nro, (j + 1):nco]))
    for(k in 1:(nro - 1))
        summe <- summe + x[k, nco] * 0.5 * sum(x[(k + 1):nro, nco])
    n   <- sum(x);     nip <- apply(x, 1, sum);    npj <- apply(x, 2, sum)
    expect <- (n^2 - sum(nip^2))/4
    u1 <- n * (n - 1) * (2 * n + 5) - sum(nip * (nip - 1) * (2 * nip + 5)) -
                                      sum(npj * (npj - 1) * (2 * npj + 5))
    u2 <- sum(nip * (nip - 1) * (nip - 2)) * sum(npj * (npj - 1) * (npj - 2))
    u3 <- sum(nip * (nip - 1)) * sum(npj * (npj - 1))
    v  <- u1/72 + u2/(36 * n * (n - 1) * (n - 2)) + u3/(8 * n * (n - 1))
    zval <- (summe - expect)/sqrt(v);     pval <-1 - pnorm(zval)
    cat("Jonckheere-Terpstra Test : ", " Statistik =",
                               round(zval, 3)," P =",round(pval, 3),"\n")
}
```

Beispiel 1):

```
A <- c(30, 31, 34, 34, 37, 39)
B <- c(36, 38, 41, 41, 45, 48)
C <- c(44, 45, 47, 49, 50, 50)
```

```
werte <- c(A, B, C);   n <- c(6, 6, 6)
grp   <- as.ordered(factor(rep(1:length(n),n)))

jonckheere.test(werte, grp)
## Jonckheere-Terpstra Test :   Statistik = 3.768  P = 0
```

Beispiel 2):

```
A <- c(106, 114, 116, 127, 145)
B <- c(110, 125, 143, 148, 151)
C <- c(136, 139, 149, 160, 174)
werte <- c(A, B, C);  n <- c(5, 5, 5)

grp   <- as.ordered(factor(rep(1:length(n),n)))

jonckheere.test(werte, grp)
## Jonckheere-Terpstra Test :   Statistik = 2.272  P = 0.012
```

Beispiel 2) mit Funktion oneway_test() aus library(coin):

```
A <- c(106, 114, 116, 127, 145)
B <- c(110, 125, 143, 148, 151)
C <- c(136, 139, 149, 160, 174)
werte <- c(A, B, C);  n <- c(5, 5, 5)

library(coin)
d <- data.frame(werte=werte, grp=factor(c(rep("A",5),rep("B",5),rep("C",5))))

oneway_test(werte ~ grp, data=d, scores = list(grp = c(1, 2, 3)))
##
##  Asymptotic Linear-by-Linear Association Test
##
## data:  werte by grp (A < B < C)
## Z = 2.4246, p-value = 0.01533
## alternative hypothesis: two.sided
```

### 7.6.6 Varianzanalyse für Messwiederholungen

Beispiel Gewichtsreduktion:

```
diet <- data.frame(effect = c(1.5, 1.4, 1.4, 1.2, 1.4,
                              2.7, 2.9, 2.1, 3.0, 3.3,
                              2.1, 2.2, 2.4, 2.0, 2.5,
                              1.3, 1.0, 1.1, 1.3, 1.5),
                   patient   = factor(paste("pat", rep(1:5, 4), sep="")),
                   zeit      = factor(paste("T", rep(c(1, 2, 3, 4),
                                                     c(5, 5, 5, 5)), sep="")),
                   row.names = NULL)

summary(aov(effect ~ zeit + Error(patient), data=diet))
##
## Error: patient
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  4  0.393 0.09825
##
## Error: Within
##           Df Sum Sq Mean Sq F value   Pr(>F)
## zeit       3  8.154  2.7178   41.87 1.24e-06 ***
## Residuals 12  0.779  0.0649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

### 7.6.7 Friedman-Test

Tabelle mit Funktion qFriedman() aus library(SuppDists):

```
library(SuppDists)
k <- c(3,4,5,6)
n <- c(3:10, 12, 14, 16, 18, 20, 25, 30, 35, 40, 45,
                              50, 60, 70, 80, 90, 100, 1000)
alpha <- c(0.05, 0.01)

tab <- matrix(NA, nrow = 27, ncol = 10, byrow = TRUE)
tab[1,] <- c(NA, rep(0.05,4), rep(0.01, 4), NA)
tab[2,] <- c(NA, rep(c(3,4,5,6), 2), NA)

for (i in 3:27) {
   tab[i,] <- c(n[i-2], round(qFriedman(0.95, k, n[i-2]), 3),
                        round(qFriedman(0.99, k, n[i-2]), 3), n[i-2]) }

colnames(tab) <- c("n",3:6,3:6,"n")
as.data.frame(tab[13:20,])
```

| n | 3 | 4 | 5 | 6 | 3 | 4 | 5 | 6 | n |
|---|---|---|---|---|---|---|---|---|---|
| 16 | 6.125 | 7.800 | 9.350 | 10.893 | 9.125 | 10.875 | 12.725 | 14.500 | 16 |
| 18 | 5.778 | 7.800 | 9.378 | 10.921 | 8.778 | 10.933 | 12.800 | 14.571 | 18 |
| 20 | 6.100 | 7.800 | 9.400 | 10.943 | 9.100 | 10.980 | 12.840 | 14.629 | 20 |
| 25 | 6.000 | 7.776 | 9.408 | 10.971 | 8.880 | 11.016 | 12.928 | 14.720 | 25 |
| 30 | 6.067 | 7.800 | 9.413 | 10.990 | 8.867 | 11.120 | 12.973 | 14.781 | 30 |
| 35 | 6.000 | 7.783 | 9.440 | 10.980 | 9.029 | 11.143 | 13.029 | 14.824 | 35 |
| 40 | 6.000 | 7.800 | 9.440 | 11.000 | 9.075 | 11.160 | 13.060 | 14.857 | 40 |
| 45 | 6.000 | 7.787 | 9.440 | 11.010 | 9.111 | 11.173 | 13.084 | 14.883 | 45 |

Beispiel Schokoladensorten:

```
y <- matrix(c( 2.2, 2.0, 1.8,
               2.4, 1.8, 1.6,
               2.5, 1.9, 1.7,
               1.7, 2.5, 1.9),
            nrow = 4, byrow = TRUE,
            dimnames = list(Person = as.character(1:4),
                            Sorte = LETTERS[1:3]))
as.data.frame(y)
```

| A | B | C |
|---|---|---|
| 2.2 | 2.0 | 1.8 |
| 2.4 | 1.8 | 1.6 |
| 2.5 | 1.9 | 1.7 |
| 1.7 | 2.5 | 1.9 |

```
n <- dim(y)[1];    k <- dim(y)[2]
R <- matrix(rep(NA, n*k), nrow=n, byrow=TRUE)                # Rangzahlen
for (i in 1:n) R[i,] <- rank(-y[i,]); R
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    1    2    3
## [3,]    1    2    3
## [4,]    3    1    2
Ri2 <- colSums(R)^2; Ri2
## [1]  36  49 121
                                                             # Teststatistik
stat <- (12/(n*k*(k+1)))*sum(Ri2) - 3*n*(k+1); stat
## [1] 3.5
pval <- 1 - pchisq(stat, k-1); pval
## [1] 0.1737739

friedman.test(y)                                             # Funktion friedman.test()
##
##  Friedman rank sum test
##
## data:  y
## Friedman chi-squared = 3.5, df = 2, p-value = 0.1738
```

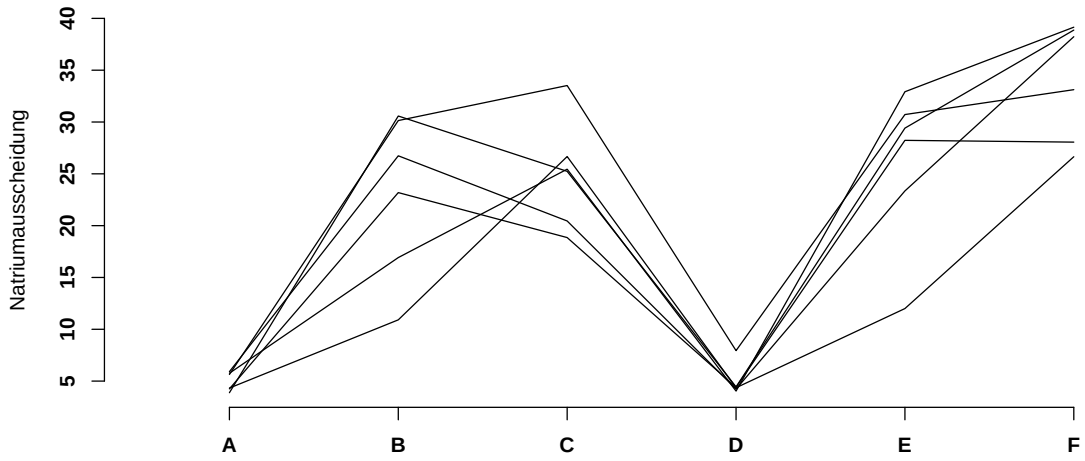
#### 7.6.7.2 Multiple paarweise Vergleiche nach Wilcoxon und Wilcox

Beispiel Diuretika:

```
diuret <- data.frame(block = factor(rep(1:6, rep(6,6))),
                diuretikum = factor(rep(c("A","B","C","D", "E","F"), 6)),
                natrium = c(3.88, 30.58, 25.24, 4.44, 29.41, 38.87,
                            5.64, 30.14, 33.52, 7.94, 30.72, 33.12,
                            5.76, 16.92, 25.45, 4.04, 32.92, 39.15,
                            4.25, 23.19, 18.85, 4.40, 28.23, 28.06,
                            5.91, 26.74, 20.45, 4.23, 23.35, 38.23,
                            4.33, 10.91, 26.67, 4.36, 12.00, 26.65))

reshape(diuret, timevar="diuretikum", idvar="block", direction="wide")
```

| | block | natrium.A | natrium.B | natrium.C | natrium.D | natrium.E | natrium.F |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 3.88 | 30.58 | 25.24 | 4.44 | 29.41 | 38.87 |
| 7 | 2 | 5.64 | 30.14 | 33.52 | 7.94 | 30.72 | 33.12 |
| 13 | 3 | 5.76 | 16.92 | 25.45 | 4.04 | 32.92 | 39.15 |
| 19 | 4 | 4.25 | 23.19 | 18.85 | 4.40 | 28.23 | 28.06 |
| 25 | 5 | 5.91 | 26.74 | 20.45 | 4.23 | 23.35 | 38.23 |
| 31 | 6 | 4.33 | 10.91 | 26.67 | 4.36 | 12.00 | 26.65 |

```
par(mfcol=c(1,1), lwd=3, font.axis=2, bty="l", ps=12)
matplot(t(matrix(diuret$natrium, ncol = 6, byrow = TRUE)),
          type = "l", col = 1, lty = 1, axes = FALSE,
          ylab = "Natriumausscheidung", xlim = c(0.5, 6.5))
axis(1, at = 1:6, labels = levels(diuret$diuretikum)); axis(2)
```



Beispiel Diuretika:

Funktion symmetrie_test() in library(coin)

```
library(coin)                                              # multiple paarweise Vergleiche
friedman_test(natrium ~ diuretikum | block, data = diuret)
##
##  Asymptotic Friedman Test
##
## data:  natrium by
##   diuretikum (A, B, C, D, E, F)
##   stratified by block
## chi-squared = 23.333, df = 5, p-value = 0.0002915
library(multcomp)
friedm <- symmetry_test(natrium ~ diuretikum | block, data = diuret,
       xtrafo = function(data) trafo(data, factor_trafo = function(x)
                model.matrix(~ x - 1) %*% t(contrMat(table(x), "Tukey"))),
       ytrafo = function(data) trafo(data, numeric_trafo = rank,
                                     block = diuret$block),
       teststat = "max",
      )
pvalue(friedm)                                                  # Friedman-Test
## [1] 0.001588278
## 99 percent confidence interval:
##  0.001464350 0.001712206

drop(round(pvalue(friedm, method = "single-step"), 5))
##   B - A   C - A   D - A   E - A   F - A   C - B   D - B   E - B   F - B   D - C
## 0.18815 0.09154 0.99963 0.03960 0.00158 0.99963 0.33868 0.98982 0.63622 0.18815
##   E - C   F - C   E - D   F - D   F - E
## 0.99963 0.81999 0.09154 0.00507 0.93999
```

#### 7.6.7.3 Page-Test für geordnete Alternativen

Funktion page.trend.test() in library(crank)

```
library(crank)
Gutachter <- matrix(c(2,2,1,2,2,1,3,1,1,                    # Objekt B
                      1,3,2,3,1,2,2,2,4,                    # Objekt C
                      4,1,3,1,4,3,1,4,2,                    # Objekt D
                      3,4,4,4,3,4,4,3,3),                   # Objekt A
                      nrow=9,byrow=F)

Page <- page.trend.test(Gutachter, ranks=TRUE)
Page
##
## Page test for ordered alternatives
## L = 252  p(table)  <=.01
```

Geordnete Alternativen mit der Funktion friedman_test() in library(coin):

```
g <- data.frame(block = factor(rep(1:9, rep(4,9))),
                objekt = ordered(rep(c("Obj1","Obj2","Obj3","Obj4"), 9)),
                note = c(2,1,4,3,     2,3,1,4,       1,2,3,4,
                         2,3,1,4,     2,1,4,3,       1,2,3,4,
                         3,2,1,4,     1,2,4,3,       1,4,2,3))

library(coin)
friedman_test(note ~ objekt | block, data = g)
##
##  Asymptotic Page Test
##
## data:  note by
##   objekt (Obj1 < Obj2 < Obj3 < Obj4)
##   stratified by block
## Z = 3.1177, p-value = 0.001823
## alternative hypothesis: two.sided
```

#### 7.6.7.4 Spannweitenrangtest nach Quade

Beispiel Marktanalyse:

```
y <- matrix(c( 5, 4, 7, 10, 12, 1, 3, 1, 0, 2,
               16, 12, 22, 22, 35, 5, 4, 3, 5, 4,
               10, 9, 7, 13, 10, 19, 18, 28, 37, 58,
               10, 7, 6, 8, 7),
            nrow = 7, byrow = TRUE,
            dimnames = list(Store = as.character(1:7),
                            Brand = LETTERS[1:5]))
as.data.frame(y)
```

| A | B | C | D | E |
|---|---|---|---|---|
| 5 | 4 | 7 | 10 | 12 |
| 1 | 3 | 1 | 0 | 2 |
| 16 | 12 | 22 | 22 | 35 |
| 5 | 4 | 3 | 5 | 4 |
| 10 | 9 | 7 | 13 | 10 |
| 19 | 18 | 28 | 37 | 58 |
| 10 | 7 | 6 | 8 | 7 |

```
k <- dim(y)[1];    b <- dim(y)[2]
R <- matrix(rep(NA,k*b), nrow=k, byrow=TRUE)            # Rangzahlen
for (i in 1:k) R[i,] <- rank(y[i,])
as.data.frame(R)
```

| V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|
| 2.0 | 1.0 | 3.0 | 4.0 | 5.0 |
| 2.5 | 5.0 | 2.5 | 1.0 | 4.0 |
| 2.0 | 1.0 | 3.5 | 3.5 | 5.0 |
| 4.5 | 2.5 | 1.0 | 4.5 | 2.5 |
| 3.5 | 2.0 | 1.0 | 5.0 | 3.5 |
| 2.0 | 1.0 | 3.0 | 4.0 | 5.0 |
| 5.0 | 2.5 | 1.0 | 4.0 | 2.5 |

```
                                                        # Spannweiten
range  <- rep(NA, k)
for (i in 1:k) range[i] <- max(y[i,]) - min(y[i,])
Qi <- rank(range)

S  <- Qi * (R - (b + 1)/2)                              # Scores
A2 <- sum(S^2)                                          # Q-gesamt
B  <- sum(colSums(S)^2)/k                               # Q-zwischen
stat <- (k-1)*B / (A2-B); stat                          # Teststatistik
## [1] 3.829252
pval <- 1 - pf(stat, b-1, (b-1)*(k-1)); pval
## [1] 0.01518902
```

```
quade.test(y)                                          # quade.test() in library(stats)
##
##  Quade test
##
## data:  y
## Quade F = 3.8293, num df = 4, denom df = 24, p-value = 0.01519
```

Multiple paarweise Vergleiche mit Funktion posthoc.quade.test() in library(PMCMR)

```
library(PMCMR)
posthoc.quade.test(y, dist="TDist", p.adj="none")
##
##  Pairwise comparisons using posthoc-Quade test with TDist approximation
##
## data:  y
##
##   A      B      C      D
## B 0.2087 -      -      -
## C 0.8401 0.2874 -      -
## D 0.1477 0.0102 0.1021 -
## E 0.0416 0.0021 0.0269 0.5172
##
## P value adjustment method: none
```

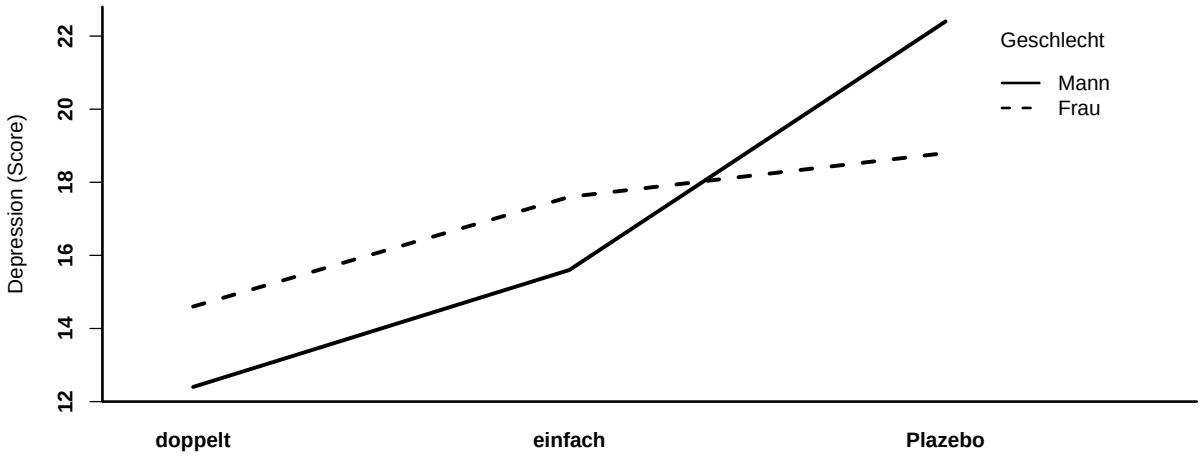
### 7.6.8 Zweifache Varianzanalyse

Beispiel Antidepressiva:

```
depr <- data.frame(
        score = c(22, 25, 22, 21, 22, 16, 16, 16, 15, 15, 13, 12, 12, 13, 12,
                  18, 19, 17, 21, 19, 19, 20, 17, 16, 16, 16, 14, 16, 13, 14),
        geschl = factor(c(rep("Mann", 15), rep("Frau",15))),
        therap = factor(rep(c(rep("Plazebo",5),rep("einfach",5),
                        rep("doppelt",5)),2)))
as.data.frame(depr[1:5,])
```

| score | geschl | therap |
|---|---|---|
| 22 | Mann | Plazebo |
| 25 | Mann | Plazebo |
| 22 | Mann | Plazebo |
| 21 | Mann | Plazebo |
| 22 | Mann | Plazebo |

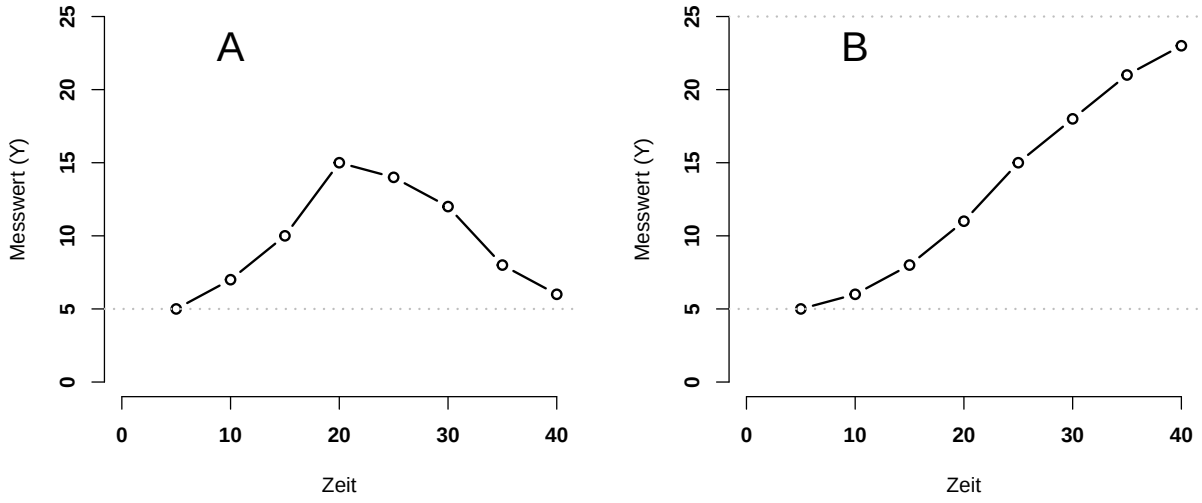
```
summary(aov(score ~ therap + geschl + geschl:therap, depr))
##                Df Sum Sq Mean Sq F value   Pr(>F)
## therap          2  253.4   126.7  74.529 5.06e-11 ***
## geschl          1    0.3     0.3   0.176    0.678
## therap:geschl   2   54.2    27.1  15.941 3.94e-05 ***
## Residuals      24   40.8     1.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

par(mfrow=c(1,1), lwd=2, font.axis=2, bty="l", ps=12)
interaction.plot(depr$therap, depr$geschl, depr$score, main=" ",
      trace.label="Geschlecht", lwd=3,  xlab=" ", ylab="Depression (Score)")
```


Depression (Score)
12
14
16
18
20
22
doppelt
einfach
Plazebo
Geschlecht
Mann
Frau

### 7.6.9 Analyse von wiederholten Messungen

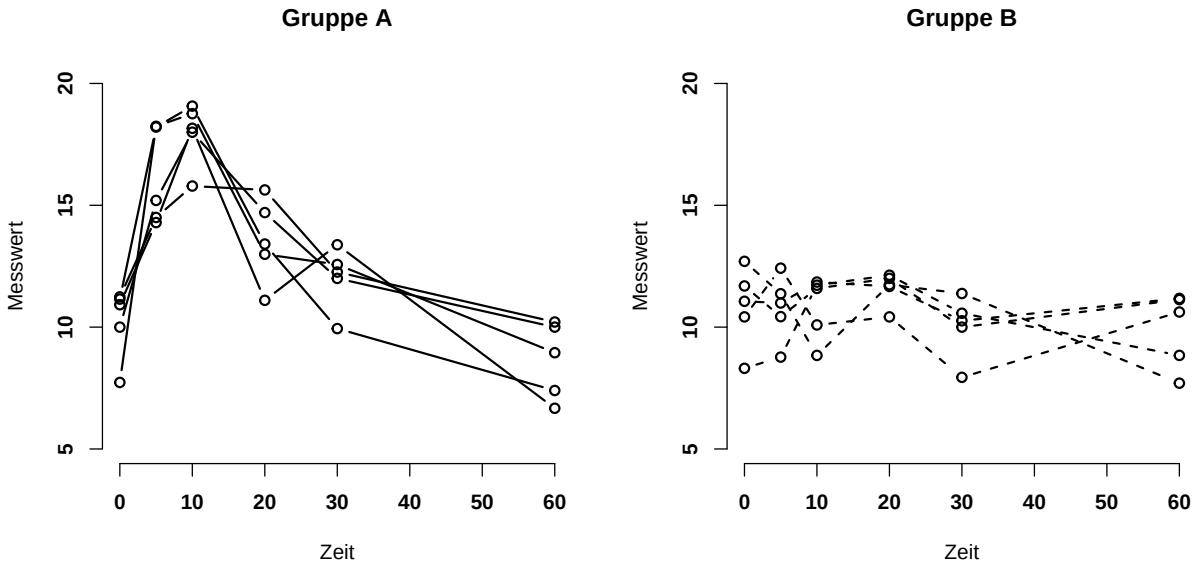
Zeitlicher Verlauf:

```
Zeit <- c(5, 10, 15, 20, 25, 30, 35, 40)
y1   <- c(5,  7, 10, 15, 14, 12,  8,  6)
y2   <- c(5,  6,  8, 11, 15, 18, 21, 23)

par(mfrow=c(1,2), lwd=1.8, font.axis=2, bty="n", ps=12)
plot(Zeit, y1, type="b", ylab="Messwert (Y)", ylim=c(0,25), xlim=c(0,40))
abline(h=5, lty=3, col="grey")
text(10, 23, "A", cex=2)
plot(Zeit, y2, type="b", ylab="Messwert (Y)", ylim=c(0,25), xlim=c(0,40))
text(10, 23, "B", cex=2)
abline(h=5, lty=3, col="grey")
abline(h=25, lty=3, col="grey")
```




Beispiel:

```
daten  <- read.csv2("repeated.csv")
attach(daten)
Zeit <- c(0, 5, 10, 20, 30, 60)

par(mfrow=c(1,2), lwd=1.7, font.axis=2, bty="n", ps=12)
plot(Zeit, daten[1,3:8], type="b", ylab="Messwert", main="Gruppe A",
     ylim=c(5,20))
for (i in 2:5) lines(Zeit, daten[i, 3:8], type="b")
plot(Zeit, daten[6,3:8], type="b", ylab="Messwert", main="Gruppe B",
     ylim=c(5,20), lty=2)
for (i in 7:10) lines(Zeit, daten[i, 3:8], type="b", lty=2)
```




```
AUC   <- function(y, t, n) {                                    # Area Under Curve
         F <- rep(NA, (n-1))
         for (i in 1:(n-1)) F[i] <- (t[i+1]-t[i])*(y[i]+y[i+1])
         sum(F)/2          }

REGR <- function(y, t, n) {                                     # Regressions-Koeffizient
sum((y-mean(y))*(t-mean(t)))/sum((t-mean(t))^2)     }
```

```
daten  <- read.csv2("repeated.csv")
attach(daten)
Zeit <- c(0, 5, 10, 20, 30, 60)
                                                                # Tabelle zum Beispiel
daten$Max   <- apply(daten[,3:8], 1, max, na.rm = T)            # Maximum
daten$AUC   <- apply(daten[,3:8], 1, AUC,  t=Zeit, n=6)         # AUC
daten$REGR  <- apply(daten[,5:8], 1, REGR, t=Zeit[3:6], n=4)    # Regression
daten
```

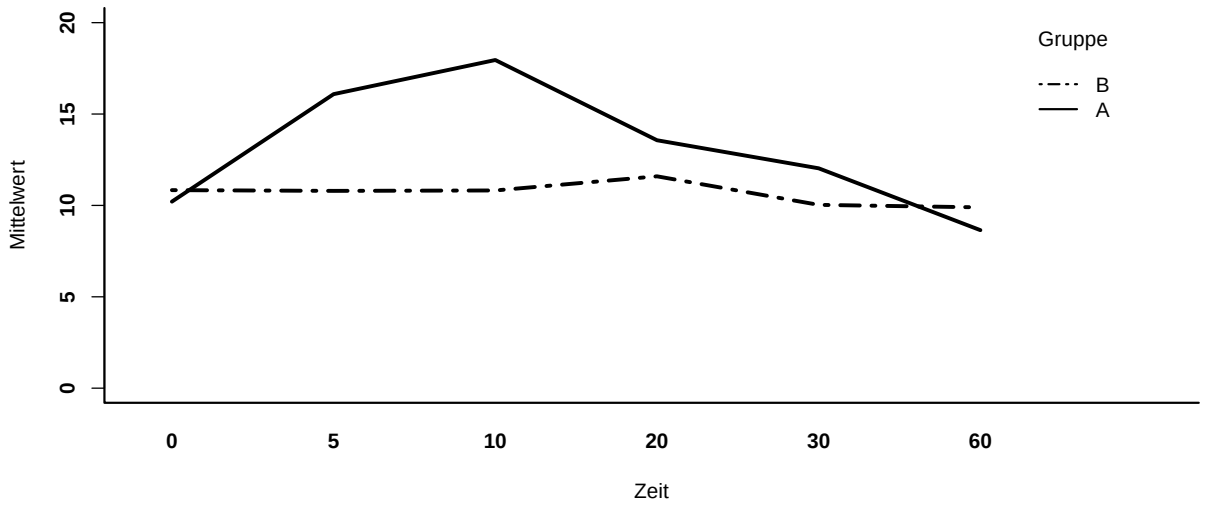
| Gruppe | Prob | t0 | t5 | t10 | t20 | t30 | t60 | Max | AUC | REGR |
|---|---|---|---|---|---|---|---|---|---|---|
| A | 1 | 10.00 | 15.20 | 18.00 | 14.70 | 12.00 | 10.00 | 18.00 | 773.000 | -0.1478571 |
| A | 2 | 10.92 | 14.29 | 18.16 | 11.10 | 13.38 | 6.67 | 18.16 | 713.600 | -0.1957857 |
| A | 3 | 7.73 | 18.24 | 18.76 | 12.99 | 12.57 | 8.95 | 18.76 | 766.775 | -0.1690000 |
| A | 4 | 11.15 | 18.21 | 19.07 | 13.41 | 9.94 | 7.40 | 19.07 | 705.850 | -0.2096429 |
| A | 5 | 11.24 | 14.50 | 15.79 | 15.63 | 12.26 | 10.21 | 15.79 | 773.675 | -0.1184286 |
| B | 6 | 11.69 | 10.43 | 11.59 | 12.00 | 10.00 | 11.13 | 12.00 | 655.250 | -0.0127857 |
| B | 7 | 12.70 | 11.37 | 8.84 | 11.74 | 11.38 | 7.70 | 12.70 | 615.400 | -0.0451429 |
| B | 8 | 8.31 | 8.77 | 11.72 | 12.13 | 10.57 | 8.84 | 12.13 | 617.825 | -0.0646429 |
| B | 9 | 10.42 | 12.42 | 10.09 | 10.42 | 7.94 | 10.62 | 12.42 | 586.125 | 0.0090000 |
| B | 10 | 11.06 | 10.99 | 11.85 | 11.67 | 10.26 | 11.18 | 11.85 | 661.075 | -0.0130714 |

```r
t1  <- t.test(Max ~ Gruppe, data = daten)                    # Maximum
max <- c(t1$estimate,t1$statistic,t1$p.value)
t2  <- t.test(AUC ~ Gruppe, data = daten)                    # AUC
auc <- c(t2$estimate,t2$statistic,t2$p.value)
t3  <- t.test(REGR ~ Gruppe, data = daten)                   # Aenderung
reg <- c(t3$estimate,t3$statistic,t3$p.value)
t4  <- t.test(t60 ~ Gruppe, data = daten)                    # letzter Wert
lst <- c(t4$estimate,t4$statistic,t4$p.value)
tab <- as.data.frame(rbind(max,auc,reg,lst),
                     row.names = c("Max","AUC","REGR","t60"))
colnames(tab) <- c("mean A", "mean B", "t-stat","p-val")
as.data.frame(tab)
```

| | mean A | mean B | t-stat | p-val |
|---|---|---|---|---|
| Max | 17.9560000 | 12.2200000 | 9.636377 | 0.0003354 |
| AUC | 746.5800000 | 627.1350000 | 5.816149 | 0.0004093 |
| REGR | -0.1681429 | -0.0253286 | -6.810586 | 0.0001701 |
| t60 | 8.6460000 | 9.8940000 | -1.265131 | 0.2414327 |

#### 7.6.9.2 ANOVA für wiederholte Messungen (gemischte Modelle)

```
daten  <- read.csv2("repeated.csv")
Zeit <- c(0, 5, 10, 20, 30, 60)
neu  <- reshape(daten, varying=list(c("t0","t5","t10","t20","t30","t60")),
         v.names=c("Wert"), timevar="Zeit",
         times=c(0, 5, 10, 20, 30, 60), idvar="Prob", direction="long")

par(mfrow=c(1,1), lwd=1.7, font.axis=2, bty="l", ps=12)
interaction.plot(neu$Zeit, factor(neu$Gruppe), neu$Wert, lty=c(1,12), lwd=3,
             ylim=c(0,20), ylab="Mittelwert", xlab="Zeit", trace.label="Gruppe")
```




```
rep.aov <- aov(neu$Wert ~ factor(neu$Gruppe)*factor(neu$Zeit) + Error(factor(neu$Prob)))
summary(rep.aov)
##
## Error: factor(neu$Prob)
##                       Df Sum Sq Mean Sq F value   Pr(>F)
## factor(neu$Gruppe)  1  87.94   87.94   76.69 2.26e-05 ***
## Residuals           8   9.17    1.15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Error: Within
##                                      Df Sum Sq Mean Sq F value   Pr(>F)
## factor(neu$Zeit)                      5 185.92   37.18   15.97 1.22e-08 ***
## factor(neu$Gruppe):factor(neu$Zeit)  5 134.07   26.81   11.52 6.28e-07 ***
## Residuals                           40  93.11    2.33
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

### 7.6.10 Versuchsplanung

Beispiel Feldversuch:

```
Amm   <- c(rep(0,4),rep(1,4),rep(0,4),rep(1,4),rep(0,4),rep(1,4),rep(0,4),rep(1,4))
Magn  <- c(rep(0,8),rep(1,8),rep(0,8),rep(1,8))
Mist  <- c(rep(0,16),rep(1,16))
yield <- c(19.2,15.5,17.0,11.7,20.6,16.9,19.5,21.9,18.9,20.2,16.7,
           20.7,25.3,27.6,29.1,25.4,20.8,18.5,20.1,19.2,26.8,17.8,
           18.6,19.0,22.2,18.6,22.3,21.1,27.7,28.6,28.7,28.5)
data <- data.frame(block=gl(8,4), Amm=factor(Amm),
         Magn=factor(Magn), Mist=factor(Mist), yield=yield)

yield.aov1 <- aov(yield ~ block, data)
summary(yield.aov1)
##             Df Sum Sq Mean Sq F value   Pr(>F)
## block        7  484.2   69.17   12.92 8.91e-07 ***
## Residuals   24  128.5    5.35
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

yield.aov2 <- aov(yield ~ Amm*Magn*Mist, data)
summary(yield.aov2)
##               Df Sum Sq Mean Sq F value   Pr(>F)
## Amm            1 196.52  196.52  36.700 2.95e-06 ***
## Magn           1 192.57  192.57  35.963 3.43e-06 ***
## Mist           1  32.60   32.60   6.089  0.02112 *
## Amm:Magn       1  52.79   52.79   9.858  0.00444 **
## Amm:Mist       1   5.70    5.70   1.064  0.31267
## Magn:Mist      1   0.69    0.69   0.129  0.72270
## Amm:Magn:Mist  1   3.32    3.32   0.619  0.43907
## Residuals     24 128.51    5.35
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## 7.7 Die Analyse von Häufigkeiten

### 7.7.1 Vergleich relativer Häufigkeiten

Beispiel Präsidentschaftswahlen (Bootstrap)

```
n.obama <- 120; n.mccain <- 80; n.other <- 10
data <- c(rep("Obama",n.obama),
          rep("McCain", n.mccain), rep("andere",n.other))

bootstrap.stat <- function(data) {                                    # Bootstrap
    b.smpl <- sample(data, length(data), replace=TRUE)
    (sum(b.smpl=="Obama") - sum(b.smpl=="McCain"))/length(data)
    }

p.distr <- replicate(500, bootstrap.stat(data))
round(quantile(p.distr, probs=c(0.025, 0.25, 0.50, 0.75, 0.975)), 4)
##   2.5%    25%    50%    75%  97.5%
## 0.0619 0.1429 0.1905 0.2333 0.3143
```

Beispiel Biodiversitäten (Shannon-Index):

```
food      <- c("Eiche","Mais","Brombeere","Buche","Kirsche","Sonstige")
michigan  <- c(47, 35,  7,  5, 3, 2)
louisiana <- c(48, 23, 11, 13, 8, 2)
jay.diet  <- as.data.frame(cbind(food, michigan, louisiana))
jay.diet
```

| food | michigan | louisiana |
|---|---|---|
| Eiche | 47 | 48 |
| Mais | 35 | 23 |
| Brombeere | 7 | 11 |
| Buche | 5 | 13 |
| Kirsche | 3 | 8 |
| Sonstige | 2 | 2 |

```
shannon.index <- function(x) {
  n <- sum(x); k <- length(x)
  (n*log10(n) - sum(x*log10(x)))/n
}

H.1 <- shannon.index(michigan); H.1
## [1] 0.5403328
H.2 <- shannon.index(louisiana); H.2
## [1] 0.6327828
```

```
shannon.test <- function(x, y, alpha) {
  sign <- 1 - alpha/2
  nx  <- sum(x); kx <- length(x)
```

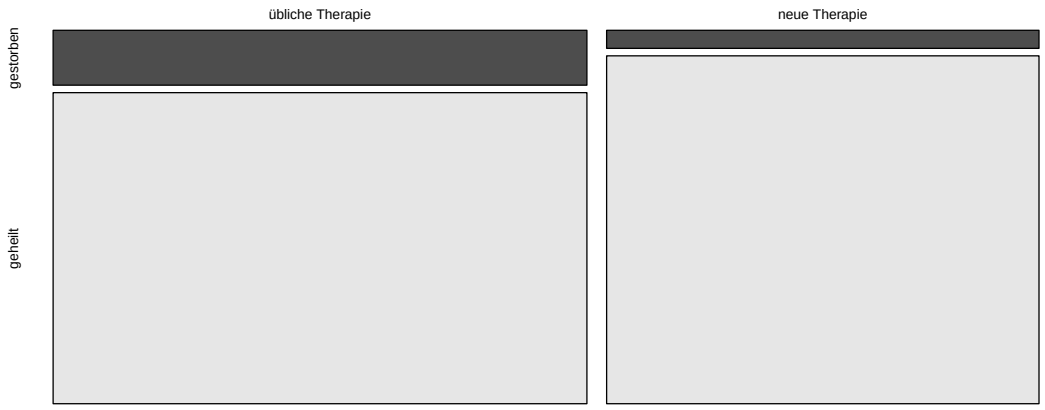
```
  ny  <- sum(y); kx <- length(y)
  Hx  <- (nx*log10(nx) - sum(x*log10(x)))/nx                    # Shannon-Index 1
  Hy  <- (ny*log10(ny) - sum(y*log10(y)))/ny                    # Shannon Index 2
  s2x <- (sum(x*log10(x)^2) - (sum(x*log10(x)))^2/nx)/nx^2
  s2y <- (sum(y*log10(y)^2) - (sum(y*log10(y)))^2/ny)/ny^2
  stat <- abs((Hx - Hy) / sqrt(s2x + s2y))                      # Teststatistik
  fg   <- round(nx*ny*(s2x+s2y)^2 / (ny*s2x^2 + nx*s2y^2), 0)
  p.val <- (1 - pt(stat, fg))*2                                 # P-Wert zweiseitig
  cat(" Shannon-Wiener-Index in Population X =", round(Hx, 4),
      "und in Population y =", round(Hy, 4),"\n",
      "Tesstatistik:", round(stat, 4),
      "t-Verteilung mit",fg,"Freiheitsgraden:",
      round(qt(sign, fg), 4),"- Pwert", round(p.val, 4),"\n")
}

shannon.test(michigan, louisiana, alpha=0.05)
##  Shannon-Wiener-Index in Population X = 0.5403 und in Population y = 0.6328
##  Tesstatistik: 1.909 t-Verteilung mit 196 Freiheitsgraden: 1.9721 - Pwert 0.0577
```

### 7.7.2 Analyse von Vierfeldertafeln

```
tab <- matrix(c(15, 85, 4, 77), nrow=2, ncol=2, byrow=TRUE)
dimnames(tab) <- list(c("übliche Therapie","neue Therapie"),
                      c("gestorben","geheilt"))
as.data.frame(tab)
```

| | gestorben | geheilt |
|---|---|---|
| übliche Therapie | 15 | 85 |
| neue Therapie | 4 | 77 |

```
mosaicplot(tab, col=TRUE, main=" ")
```




```
chisq.test(tab, correct=FALSE)                                    # ohne Korrektur
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 4.8221, df = 1, p-value = 0.0281

chisq.test(tab, correct=TRUE)                                     # mit Korrektur
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  tab
## X-squared = 3.8107, df = 1, p-value = 0.05093
```

#### 7.7.2.1 Fallzahl und Power zum Vierfeldertest

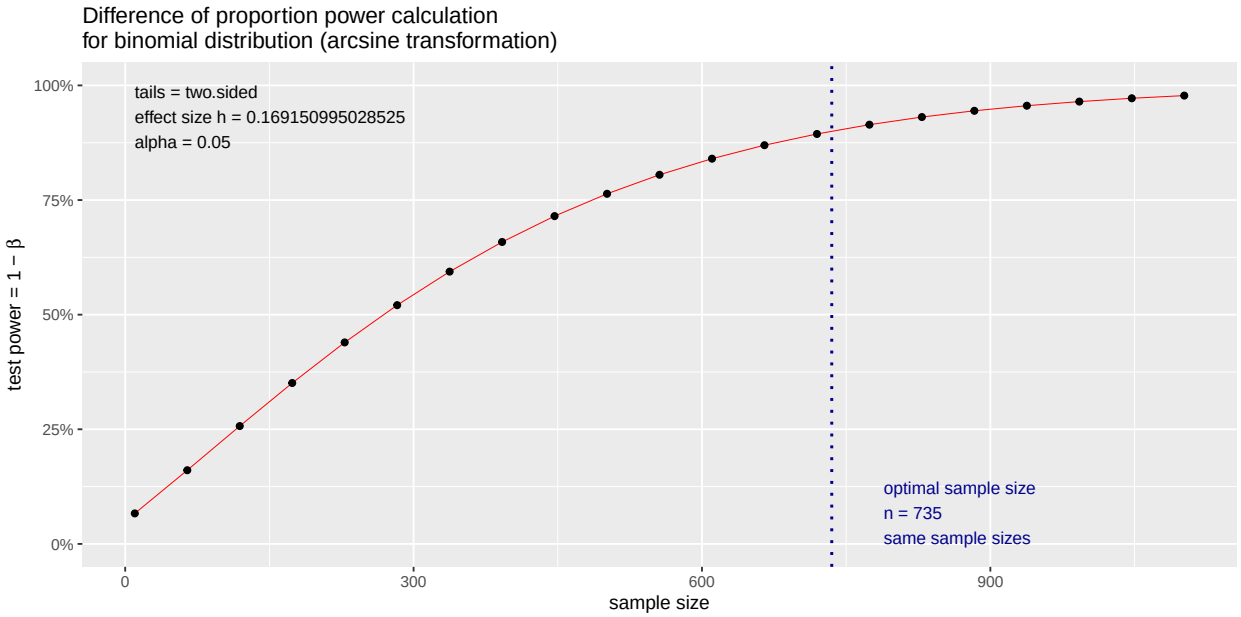
```
z.alpha <- qnorm(0.975); z.beta  <- qnorm(0.90)

p1 <- 0.38;       q1 <- 1 - p1
p2 <- 0.30;       q2 <- 1 - p2
p <- (p1 + p2)/2; q <- 1 - p
n <- (z.alpha * sqrt(2*p*q) + z.beta *
          sqrt(p1*q1+p2*q2))^2 / ((p2 - p1)^2); n
## [1] 734.7537
```

```
power.prop.test(p1=0.3, p2=0.38, sig.level =0.05, power = 0.90)
##
##      Two-sample comparison of proportions power calculation
##
##               n = 734.7537
##              p1 = 0.3
##              p2 = 0.38
##       sig.level = 0.05
##           power = 0.9
##     alternative = two.sided
##
## NOTE: n is number in *each* group

power.prop.test(p1=0.3, p2=0.38, sig.level =0.05, n=735)
##
##      Two-sample comparison of proportions power calculation
##
##               n = 735
##              p1 = 0.3
##              p2 = 0.38
##       sig.level = 0.05
##           power = 0.9000955
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

Funktion ES.h() und pwr.2p.test() in library(pwr)

```
library(pwr)
effect <- ES.h(0.38, 0.30); effect
## [1] 0.169151
p2p    <- pwr.2p.test(h = effect, sig.level = 0.05, power = 0.90);
p2p
##
##      Difference of proportion power calculation for binomial distribution
##
##               h = 0.169151
##               n = 734.4749
##       sig.level = 0.05
##           power = 0.9
##     alternative = two.sided
```

```
##
## NOTE: same sample sizes

plot(p2p)
```



### 7.7.3 Spezielle Risiko- und Effektma0ße

Relatives Risiko und Odds Ratio

```
a <- 24; b <- 96; c <- 48; d <- 592
tab <- matrix(c(a, b, c, d), nrow=2, ncol=2, byrow=TRUE)
dimnames(tab) <- list(c("exponiert","nicht exponiert"),
                      c("krank","nichtkrank"))
as.data.frame(tab)
```

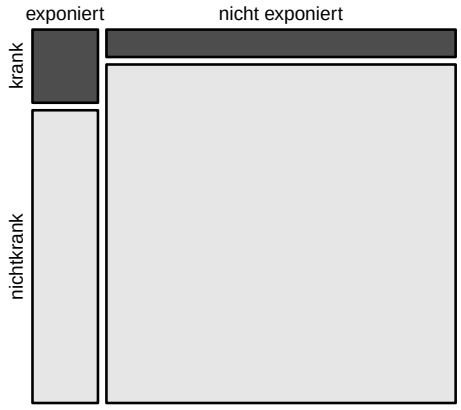
| | krank | nichtkrank |
|---|---|---|
| exponiert | 24 | 96 |
| nicht exponiert | 48 | 592 |

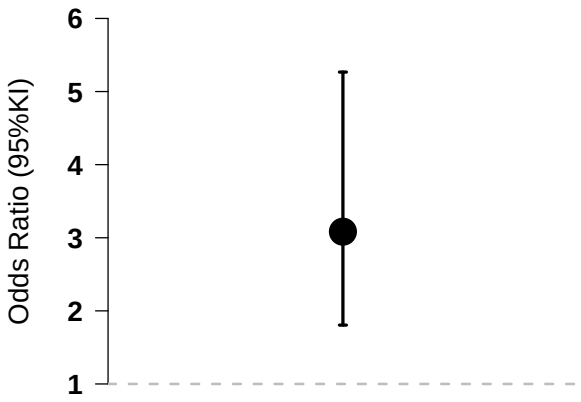
```
IR.exp  <- a / (a+b); IR.exp                                  # Inzidenzrate exponiert
## [1] 0.2

IR.nexp <- c / (c+d); IR.nexp                                 # Inzidenzrate nicht exponiert
## [1] 0.075

delta   <- IR.exp - IR.nexp; delta                            # zuschreibbares Risiko
## [1] 0.125

psi     <- IR.exp / IR.nexp; psi                              # relatives Risiko
## [1] 2.666667

omega   <- (a*d) / (b*c); omega                               # Odds Ratio
## [1] 3.083333
```

Funktion oddsratio() in library(vcd)

```
library(vcd)
library(Hmisc)
a <- 24; b <- 96; c <- 48; d <- 592
tab <- matrix(c(a, b, c, d), nrow=2, ncol=2, byrow=TRUE)
dimnames(tab) <- list(c("exponiert","nicht exponiert"),
                      c("krank","nichtkrank")); tab
##                  krank nichtkrank
## exponiert           24         96
## nicht exponiert     48        592
OR <- loddsratio(tab, log=FALSE)

sor <- summary(OR); sor                                       # OR Schätzung
##
## z test of coefficients:
##
##                                              Estimate Std. Error z value
## exponiert:nicht exponiert/krank:nichtkrank  3.08333    0.84218  3.6611
##                                               Pr(>|z|)
## exponiert:nicht exponiert/krank:nichtkrank 0.0002511 ***
## ---
```

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

cor <- confint(OR); cor                                    # Konfidenzintervall
##                                                 2.5 %   97.5 %
## exponiert:nicht exponiert/krank:nichtkrank 1.805189 5.266454

par(mfrow=c(1,2), lwd=2, font.axis=2.5, bty="n", ps=16)
mosaicplot(tab, col=TRUE, main=" ")
u <- cor[1]; v <- sor[1]; o <- cor[2]
errbar( x=1, v, o, u , xlim=c(0.9, 1.1), ylim=c(1,6), las=1, xaxt="n",
        cex=3, lwd=2.5, xlab=" ", ylab="Odds Ratio (95%KI)")
abline(h=1, lty=2, col="grey")
```




#### 7.7.3.3 Stichprobenumfänge

Fall-Kontroll-Studien mit mehreren Kontrollen:

```
smpl.matched.cc <- function(alpha, beta, MM, OR, ff) {
    zalpha <- qnorm(alpha/2, lower.tail = FALSE)
    zbeta  <- qnorm(beta, lower.tail=FALSE)
    ss <- (OR-1)*ff*(1-ff)/((1-ff)+ ff*OR)
    nn <- (MM+1)*((zalpha*(1+OR) + 2*zbeta*sqrt(OR))^2)/(2*MM*(OR^2 - 1)*ss)
    round(nn, 0)
}

smpl.matched.cc(alpha=0.05, beta=0.10, 1:4, OR=1.5, ff=0.1)
## [1] 1206  905  804  754

result <- matrix(NA, nrow=6, ncol=5)
or     <- c(1.5,2,2.5,3,3.5,4)
for (i in 1:length(or))
result[i, ] <- c(or[i], smpl.matched.cc(0.05, 0.10, 1:4, or[i], 0.1))
```

```r
colnames(result) <- c("OR", "1:1", "1:2", "1:3", "1:4")
as.data.frame(result)
```

| OR | 1:1 | 1:2 | 1:3 | 1:4 |
|---|---|---|---|---|
| 1.5 | 1206 | 905 | 804 | 754 |
| 2.0 | 368 | 276 | 245 | 230 |
| 2.5 | 193 | 145 | 129 | 121 |
| 3.0 | 126 | 94 | 84 | 79 |
| 3.5 | 92 | 69 | 61 | 57 |
| 4.0 | 72 | 54 | 48 | 45 |

Hypothesentest in Kohortenstudien:

```r
npwr.RR <- function(P2, RR, N=NULL, alpha=0.05, power=NULL, einseitig=TRUE, r=1) {
  if (sum(sapply(list(N, power), is.null)) != 1)
    stop("Entweder 'N' oder die 'Power' müssen NULL gesetzt werden!")
  if (einseitig) zalph <- qnorm(1-alpha) else zalph <- qnorm(1-alpha/2)
  p.c   <- (P2*(r*RR+1))/(r+1)
  if (is.null(N)) {
    N     <- (r+1)/(r*(RR-1)^2*P2^2) * (zalph*sqrt((r+1)*p.c*(1-p.c))
             + qnorm(power) * sqrt(RR*P2*(1-RR*P2)+r*P2*(1-P2)))^2  }
  if (is.null(power)) {
    zpower <- (P2*abs(RR-1)*sqrt(N*r) - zalph*(r+1)*sqrt(p.c*(1-p.c))) /
      sqrt((r+1)*(RR*P2*(1-RR*P2)+r*P2*(1-P2)))
    power <- pnorm(zpower)  }
  cat("Stichprobenumfang und Power zum relativen Risiko","\n",
      "Stichprobenumfang N:",round(N, 0),"\n",
      "Power:",round(power,5),"\n")
}

npwr.RR(P2=0.2, RR=2.0, alpha=0.05, einseitig=TRUE, power=0.90)
## Stichprobenumfang und Power zum relativen Risiko
##  Stichprobenumfang N: 176
##  Power: 0.9

npwr.RR(P2=0.2, RR=2.0, alpha=0.05, einseitig=TRUE, N=176)
## Stichprobenumfang und Power zum relativen Risiko
##  Stichprobenumfang N: 176
##  Power: 0.8999
```

#### 7.7.3.4 Der expositionsbedingte Anteil (PAR)

Beispiel Framingham-Studie:

```r
a <- 72; b <- 684; c <- 20; d <- 553; N <- a+b+c+d

RR <- ( a*c + a*d ) / ( a*c + b*c ) ; RR
## [1] 2.728571

Pexp <- ( a + b ) / N ; Pexp
```

```
## [1] 0.5688488

PAR <- Pexp*(RR-1) / (1 + Pexp*(RR-1)); PAR
## [1] 0.4957888

var <- ( c*N*( a*d*(N-c )+ b*c ^ 2 ) ) / ( ( a+c ) ^3*( c+d ) ^ 3 )
se <- sqrt(var)
KIu <- PAR - 1.96*se; KIo <- PAR + 1.96*se; KIu; KIo
## [1] 0.3046911
## [1] 0.6868865
```

### 7.7.4 Exakter Test nach R.A. Fisher

Beispiel Tea Tasting Lady:

```
TeaTasting <- matrix(c(3, 1, 1, 3),  nr = 2,
       dimnames = list(Guess = c("Milk", "Tea"), Truth = c("Milk", "Tea")))
TeaTasting
##       Truth
## Guess  Milk Tea
##   Milk    3   1
##   Tea     1   3

fisher.test(TeaTasting, alternative = "greater")            # Funktion fisher.test()
##
##  Fisher's Exact Test for Count Data
##
## data:  TeaTasting
## p-value = 0.2429
## alternative hypothesis: true odds ratio is greater than 1
## 95 percent confidence interval:
##  0.3135693       Inf
## sample estimates:
## odds ratio
##   6.408309
```

```
tab <- matrix(c(2, 8, 10, 4), byrow=TRUE, nr = 2); tab
##      [,1] [,2]
## [1,]    2    8
## [2,]   10    4

fisher.test(tab, alternative="less", conf.level=0.95)       # Funktion fisher.test()
##
##  Fisher's Exact Test for Count Data
##
## data:  tab
## p-value = 0.01804
## alternative hypothesis: true odds ratio is less than 1
## 95 percent confidence interval:
##  0.0000000 0.6965009
## sample estimates:
## odds ratio
##  0.1121872
```

### 7.7.5 Äquivalenztest zweier Binomialwahrscheinlichkeiten

Intervallinklusion (TOST):

```
TOST_int <- function(r1, n1, r2, n2, delta, alpha) {
            level <- (1-2*alpha)*100
            p1 <- r1/n1; p2 <- r2/n2;  z  <- qnorm(1-2*alpha)
            d  <- p1*(1-p1)/n1 + p2*(1-p2)/n2 + ((1/n1)+(1/n2))/2
            lu <- p1 - p2 - z * sqrt(d);     lo <- p1 - p2 + z * sqrt(d)
cat("Das",level,"%-KI zur Differenz",p1-p2,"lautet",lu,"bis",lo,"\n",
    "mit Bezug zum Äquivalenzintervall",-delta,"bis",+delta,"\n") }
```

Ansatz nach Dunnett und Gent:

```
TOST_test <- function(r1, n1, r2, n2, delta) {
             p1 <- r1/n1; p2 <- r2/n2
             p1d <- (r1 + r2 + delta*n2)/(n1+n2); p2d <- p1d - delta
             z1 <- (p1 -p2 - delta)/sqrt((p1d*(1-p1d)/n1)+(p2d*(1-p2d))/n2)
             p1d <- (r1 + r2 - delta*n2)/(n1+n2); p2d <- p1d + delta
             z2 <- (p1 -p2 + delta)/sqrt((p1d*(1-p1d)/n1)+(p2d*(1-p2d))/n2)
             P <- pnorm(z1) + (1-pnorm(z2))
cat("Der P-Wert für den zweiseitigen Test auf Äquivalenz von ",p1,"versus",p2,"\n",
    "mit Delta =",delta,"beträgt P=",P,"\n")  }
```

Beispiel Therapievergleich:

```
TOST_int(120, 200, 57, 100, 0.15, 0.05)
## Das 90 %-KI zur Differenz 0.03 lautet -0.1053297 bis 0.1653297
##  mit Bezug zum Äquivalenzintervall -0.15 bis 0.15

TOST_test(120, 200, 57, 100, 0.15)
## Der P-Wert für den zweiseitigen Test auf Äquivalenz von  0.6 versus 0.57
##  mit Delta = 0.15 beträgt P= 0.02449961
```

Hinweis zur Studienplanung:

```
alpha <- 0.05;  beta  <- 0.20; p1 <- 0.58; p2 <- 0.60; delta <- 0.15
n <- ((qnorm(1-alpha) + qnorm(1-beta))^2 *
                    (p1*(1-p1) + p2*(1-p2))) / (delta-(p1-p2))^2
ceiling(n)
## [1] 104
```

### 7.7.6 McNemar Vorzeichentest

Beispiel Nachweis der Wirksamkeit:

```
wirk <- matrix(c(8, 16, 5,11), nr=2, byrow=TRUE,
        dimnames = list(verum=c("stark","schwach"),
                        placebo=c("stark","schwach")))
as.data.frame(wirk)
```

| | stark | schwach |
|---|---|---|
| stark | 8 | 16 |
| schwach | 5 | 11 |

```
mcnemar.test(wirk, correct=TRUE)
##
##  McNemar's Chi-squared test with continuity correction
##
## data:  wirk
## McNemar's chi-squared = 4.7619, df = 1, p-value = 0.0291
```

```
discordant <- c(wirk[1,2], wirk[2,1])                # diskordante Ergebnisse
nd         <- sum(discordant)
x          <- min(discordant)
pbinom(x, nd, 0.5)                                   # Binomialwahrscheinlichkeit (exakt)
## [1] 0.01330185

binom.test(x, nd, p=0.5)                             # Funktion binom.test()
##
##  Exact binomial test
##
## data:  x and nd
## number of successes = 5, number of trials = 21, p-value = 0.0266
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.08217588 0.47165983
## sample estimates:
## probability of success
##              0.2380952
```

Funktion mcnemar() in library(exact2x2)

```
library(exact2x2)
mcnemar.exact(wirk) # Funktion mcnemar.exact()
##
##  Exact McNemar test (with central confidence intervals)
##
## data:  wirk
## b = 16, c = 5, p-value = 0.0266
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
```

```
##   1.120172 11.169022
## sample estimates:
## odds ratio
##        3.2
```

Beispiel Urlaubsländer (Konfidenzintervall):

```
urlaub <- matrix(c(71, 3, 16, 10), nrow=2, byrow=TRUE,
                 dimnames = list(c("B +","B -"), c("A +","A -")))
as.data.frame(urlaub)
```

| | A + | A - |
|---|---|---|
| B + | 71 | 3 |
| B - | 16 | 10 |

```
mcnemar.test(urlaub)
##
##  McNemar's Chi-squared test with continuity correction
##
## data:  urlaub
## McNemar's chi-squared = 7.5789, df = 1, p-value = 0.005905
```

Funktion binom.confint() in library(binom)

```
library(binom)
binom.confint(3, 19, method="exact")
```

| method | x | n | mean | lower | upper |
|---|---|---|---|---|---|
| exact | 3 | 19 | 0.1578947 | 0.0338262 | 0.3957846 |

Power und Fallzahl zum McNemar-Test:

```
npwr.mcnemar <- function(p, psi, alpha=0.05, n=NULL, power=NULL) {
  if (sum(sapply(list(n, power), is.null)) != 1)
    stop("exactly one of 'n' or 'power' must be NULL")
  zalpha <- qnorm(1-alpha/2)
  if (is.null(n)) {
    zbeta <- qnorm(power)
    n <- (zalpha*sqrt(psi+1) + zbeta*sqrt((psi+1) - (psi-1)^2*p))^2 /
      (p*(psi-1)^2)   }
  if (is.null(power)) {
    zbeta <- (sqrt(n)*sqrt(p)*(psi-1) - qnorm(1-alpha/2)*sqrt(psi + 1)) /
      sqrt((psi + 1) - p * (psi - 1)^2)
    power <- pnorm(zbeta)    }
  cat("Stichprobenumfang und Power zum McNemar-Test","\n",
      "Stichprobenumfang n:",round(n, 0),"\n",
      "Power:",round(power,5),"\n")
```

```
}

npwr.mcnemar(p=0.125, psi=3.2, n=40)
## Stichprobenumfang und Power zum McNemar-Test
##  Stichprobenumfang n: 40
##  Power: 0.68298

npwr.mcnemar(p=0.125, psi=2.0, power=0.90)
## Stichprobenumfang und Power zum McNemar-Test
##  Stichprobenumfang n: 248
##  Power: 0.9
```

### 7.7.7 Test nach Mantel-Haenszel

```
MH.test <- function(tab, conf.level=0.95, correct=T) {
  ##### 3-dim. Tabelle tab[2,2,k]: 1-Exposition / 2-Krankheit / 3-Stratum #####
  zquant <- qnorm(conf.level)                                  # Signifikanzniveau
  cval   <- ifelse(correct==T, 0.5, 0)                         # Kontinuitätskorrektur (?)
                                                               # Zwischenergebnisse
  si <- dim(tab)[3]; OR <- V <- E <- w <- S <- numeric(si)

  for (i in 1:si) {                                            # OR - Statistik je Stratum
    OR[i] <- (tab[1,1,i]*tab[2,2,i]) / (tab[1,2,i]*tab[2,1,i])
    E[i]  <-  colSums(tab[,,i])[1] * rowSums(tab[,,i])[1] / sum(tab[,,i])
    V[i]  <- (colSums(tab[,,i])[1] * colSums(tab[,,i])[2] * rowSums(tab[,,i])[1]
              * rowSums(tab[,,i])[2]) / (sum(tab[,,i])^2 * (sum(tab[,,i]) - 1))
    w[i]  <- (tab[1,2,i] * tab[2,1,i]) / sum(tab[,,i])  }

  ORmh  <- sum(w*OR)/sum(w)                          # Mantel-Haenszel Odds-Ratio/Test)
  CHImh <- (sum(tab[1,1,]) - sum(E) - cval)^2 / sum(V)
  pval1 <- pchisq(CHImh, df=1, lower.tail=F)
  cio   <- ORmh**(1 + 1.96/sqrt(CHImh)); ciu   <- ORmh**(1 - 1.96/sqrt(CHImh))
                                                     # Test Effekt-Modifikation (Confounding)
  for (i in 1:si) S[i] <- (tab[1,1,i]*tab[2,2,i] -
                         ORmh*tab[1,2,i]*tab[2,1,i])^2 / (ORmh*V[i]*sum(tab[,,i])^2)
  CHIefm  <- sum(S); pval2 <- pchisq(CHIefm, df=1, lower.tail=F)

  cat("\n","*** Mantel-Haenszel Test (Effekt-Modifikation/Confounding) ***","\n",
      "Odds-Ratio in Strata:", round(OR, 2),"\n",
      "  Mantel-Haenszel OR:", round(ORmh, 2),"\n",
      "       Chiquadrat-MH:", round(CHImh, 4),"(P = ",round(pval1, 5),")","\n",
      "  Konfidenzintervall:", round(ciu,2),"-",round(cio,2),"\n",
      "   Chiquadrat-Effekt:", round(CHIefm, 4),"(P = ",round(pval2, 5),")","\n")  }
```

Beispiel:

```
tab <- array(c(15, 4, 85, 77, 20, 7, 56, 51), dim = c(2, 2, 2),
             dimnames = list( K = c("I", "II"),  E = c("+", "-"),
                              Geschl = c("maennl", "weibl")))
as.data.frame(tab)
```

| | +.maennl | -.maennl | +.weibl | -.weibl |
|---|---|---|---|---|
| I | 15 | 85 | 20 | 56 |
| II | 4 | 77 | 7 | 51 |

```
MH.test(tab, conf.level=0.95, correct=T)
##
##  *** Mantel-Haenszel Test (Effekt-Modifikation/Confounding) ***
##  Odds-Ratio in Strata: 3.4 2.6
##    Mantel-Haenszel OR: 2.91
##         Chiquadrat-MH: 7.8977 (P =  0.00495 )
```

```
##    Konfidenzintervall: 1.38 - 6.14
##     Chiquadrat-Effekt: 0.1204 (P =  0.7286 )

mantelhaen.test(tab, correct = T, conf.level = 0.95)
##
##  Mantel-Haenszel chi-squared test with continuity correction
##
## data:  tab
## Mantel-Haenszel X-squared = 7.8977, df = 1, p-value = 0.00495
## alternative hypothesis: true common odds ratio is not equal to 1
## 95 percent confidence interval:
##  1.410224 6.016843
## sample estimates:
## common odds ratio
##          2.912919
```

#### 7.7.7.1 Breslow-Day-Test

```
breslow.day.test <- function(x, OR=NA) {

    if(is.na(OR)) {                                          # OR nach Mantel-Haenszel
        OR = mantelhaen.test(x)$estimate
        names(OR) = " " }
  k <- dim(x)[3]                                             # Schichten
  a <- hat.a <- Var.a <- numeric(k)                          # Zwischenergebnisse
  X2.stat <- 0
  for (j in 1:k) {                                           # Randsummen
    mj <- apply(x[,,j], MARGIN=1, sum)
    nj <- apply(x[,,j], MARGIN=2, sum)
                                                             # Schätzung der aj
    coef <- c(-mj[1]*nj[1]*OR, nj[2]-mj[1]+OR*(nj[1]+mj[1]), 1-OR)
    sols <- Re(polyroot(coef))          # 0 < hat.aj <= min(n1_j, m1_j)
    hat.aj <- sols[(0 < sols) &  (sols <= min(nj[1],mj[1]))]
                                                             # weitere Schätzungen
    hat.bj <- mj[1]-hat.aj
    hat.cj <- nj[1]-hat.aj
    hat.dj <- mj[2]-hat.cj
                                                             # Varianz
    Var.aj <- (1/hat.aj + 1/hat.bj + 1/hat.cj + 1/hat.dj)^(-1)
    aj <- x[1,1,j]                                           # beobachtete Häufigkeiten
                                                             # Berechnung der Teststatistik
    X2.stat <- X2.stat + as.numeric((aj - hat.aj)^2 / Var.aj)
                                                             # Zwischenergebnisse
    a[j] <- aj;  hat.a[j] <- hat.aj; Var.a[j] <- Var.aj
  }
                                                             # Korrektur nach Tarrone
  X2.stat <-as.numeric(X2.stat - (sum(a) - sum(hat.a))^2/sum(Var.a))
                                                             # Berechnug des P-Wertes
  p <- 1-pchisq(X2.stat, df=k-1)

  return(unlist(list(OR = round(OR, 3), Statistik = round(X2.stat, 3),
         df = round(k-1, 0), P.Wert = round(p,6))))
}
```

Beispiel Alkoholgenuss:

```
alcohol <- array(c(1,0,9,106,4,5,26,164,25,21,29,138,
                   42,34,27,139,19,36,18,88,5,8,0,31),
           dim=c(2,2,6), dimnames=list(c("exponiert","nicht exponiert"),
           c("Fall","Kontrolle"),
           c("25-34","35-44","45-54","55-64","65-74",">74")))

mantelhaen.test(alcohol, alternative="two.sided", correct=TRUE)
##
##  Mantel-Haenszel chi-squared test with continuity correction
##
## data:  alcohol
## Mantel-Haenszel X-squared = 83.215, df = 1, p-value < 2.2e-16
```

```
## alternative hypothesis: true common odds ratio is not equal to 1
## 95 percent confidence interval:
##  3.562131 7.467743
## sample estimates:
## common odds ratio
##          5.157623

breslow.day.test(alcohol, OR=NA)
##      OR.  Statistik        df    P.Wert
##  5.158000  9.299000  5.000000  0.097704

breslow.day.test(alcohol, OR=2)                              # H0: OR=2
##        OR Statistik        df    P.Wert
##  2.000000 10.783000  5.000000  0.055863
```

### 7.7.8 Der kx2-Felder-Test nach Brandt und Snedecor

Beispiel Therapieerfolg:

```
erfolg <- matrix(c(14, 22, 18, 16, 8, 2), nr=3, byrow=T,
          dimnames = list(heilung=c("geheilt-x","geheilt-x+y","gestorben"),
          therapie=c("symptomatisch","spezifisch")))
as.data.frame(erfolg)
```

| | symptomatisch | spezifisch |
|---|---|---|
| geheilt-x | 14 | 22 |
| geheilt-x+y | 18 | 16 |
| gestorben | 8 | 2 |

```
chisq.test(erfolg, correct = TRUE)
##
##  Pearson's Chi-squared test
##
## data:  erfolg
## X-squared = 5.4954, df = 2, p-value = 0.06407
```

#### 7.7.8.1 Multipler Vergleich von Anteilen - Marascuilo-Prozedur

Beispiel Haarfarben:

```
haare <- matrix(c(32, 43, 16, 9, 55, 65, 64, 16), nrow=2, byrow=T)
colnames(haare) <- c("schwarz","braun","blond","rot")
as.data.frame(haare)
```

| schwarz | braun | blond | rot |
|---|---|---|---|
| 32 | 43 | 16 | 9 |
| 55 | 65 | 64 | 16 |

```
chisq.test(haare)
##
##  Pearson's Chi-squared test
##
## data:  haare
## X-squared = 8.9872, df = 3, p-value = 0.02946

x <- haare[1,]; n <- colSums(haare); p <- x/n; k <- length(p)

dif <- matrix(rep(NA, 4*((k*(k-1)/2))), nrow=4, byrow=T)
row.names(dif) <- c("i","j","Differenz","krit.Wert")

m <- 1
for (i in 1:(k-1)) {
```

```
  for (j in (i+1):k) {
    dif[1, m] <- i; dif[2, m] <- j
    dif[3, m] <- abs(p[i] - p[j])
    dif[4, m] <- sqrt(qchisq(0.95, df=k-1) *
                        (p[i]*(1-p[i])/n[i] + p[j]*(1-p[j])/n[j]))
    m <- m + 1
  }
}
dif[1:2,] <- round(dif[1:2,], 0)
dif[3:4,] <- round(dif[3:4,], 4)

colnames(dif) <- c("schwarz-Braun","schwarz-blond","schwarz-rot",
                   "braun-blond","braun-rot","blond-rot")
as.data.frame(dif)
```

| | schwarz-Braun | schwarz-blond | schwarz-rot | braun-blond | braun-rot | blond-rot |
|---|---|---|---|---|---|---|
| i | 1.0000 | 1.0000 | 1.0000 | 2.0000 | 2.0000 | 3.0000 |
| j | 2.0000 | 3.0000 | 4.0000 | 3.0000 | 4.0000 | 4.0000 |
| Differenz | 0.0303 | 0.1678 | 0.0078 | 0.1981 | 0.0381 | 0.1600 |
| krit.Wert | 0.1955 | 0.1911 | 0.3048 | 0.1816 | 0.2989 | 0.2961 |

```
pairwise.prop.test(x, n, p.adjust.method="holm")          # Adjustierte P-Werte
##
##  Pairwise comparisons using Pairwise comparison of proportions
##
## data:  x out of n
##
##       schwarz braun blond
## braun 1.000   -     -
## blond 0.131   0.037 -
## rot   1.000   1.000 0.682
##
## P value adjustment method: holm
```

#### 7.7.8.3 Power- und Fallzahlabschätzung

Beispiel fairer Würfel:

```
alpha <- 0.05                                              # Signifikanzniveau
quant <- qchisq(alpha, df=5, lower.tail = FALSE)
n   <- 120                                                 # Anzahl der Würfe
pi0 <- rep(1/6, 6)                                         # Pi unter der Nullhypothese
pi1 <- c(rep(3/20, 5), 1/4)                                # Pi unter der Alternative
effect <- sum((pi1-pi0)^2/pi0)
l <- n * effect; l                                         # Nichtzentralitätsparameter
## [1] 6

power <- pchisq(quant, df=5, ncp=l, lower.tail=FALSE)      # Power
power
## [1] 0.4328759
```

```r
lrange <- seq(6, 20, by=0.01); i <- 0                    # Bereich für lambda
while (power < 0.80) {
    i <- i+1; power <- pchisq(quant, df=5, ncp=lrange[i], lower.tail=FALSE) }

round(power, 4)                                          # Power über 80%
## [1] 0.8001

nc <- lrange[i] / effect                                 # Anzahl der Fälle
ceiling(nc)
## [1] 257
```

### 7.7.9 Cochran-Armitage-Test auf linearen Trend

```
tabtrend <- function(tab, scores, transpose=FALSE) {
    if (any(dim(tab)==2)) {if (transpose==TRUE) {tab <- t(tab)}
    if (dim(tab)[1]!=2)
            {stop("Cochran-Armitage nur in (2,k)-Tafel", call.=FALSE)}
    nidot <- apply(tab,2,sum);   n     <- sum(nidot) # Summen und Scores
    scri  <- scores;             scrq  <- sum(scri*nidot)/n
    p.i   <- tab[1,] / nidot                         # beobachtete Anteile
    p     <- sum(tab[1,])/n
    chi   <- 1/(p*(1-p))*(sum(nidot*((p.i-p)^2))); chi # chi-Quadrat total
    b     <- sum(nidot*(p.i-p)*(scri-scrq))/sum(nidot*(scri-scrq)^2)
    pi.h  <- p + b*(scri-scrq)
    chi.e <- (1/(p*(1-p)))*sum(nidot*(p.i-pi.h)^2); chi.e   # Abweichung
    chi.t <- b^2/(p*(1-p))*sum(nidot*(scri-scrq)^2);  chi.t # Trend
    z     <- sqrt(chi.t)
    p     <- 2*pnorm(abs(z), lower.tail=FALSE)          # P-Wert zweiseitig
cat(name="Cochran-Armitage Test auf Trend","\n",
    "Chi-Quadrat-Trend :", chi.trend=round(chi.t, 3)," p =", p.wert=p, "\n",
    "Chi-Quadrat-Fehler:", chi.err=round(chi.e, 3),"\n",
    "Chi-Quadrat-gesamt:", chi.gesamt=round(chi,3),"\n")
}}
```

Beispiel Alkoholkonsum und Fehlbildungen:

```
malform <- matrix(c(48, 38, 5, 1, 1, 17066, 14464, 788, 126, 37),
          nrow=2, byrow=T,
          dimnames = list(Fehlbildung=c("ja","nein"),
          alkohol=c("0","<1","1-2","3-5",">5")))
as.data.frame(malform)
```

| | 0 | <1 | 1-2 | 3-5 | >5 |
|---|---|---|---|---|---|
| ja | 48 | 38 | 5 | 1 | 1 |
| nein | 17066 | 14464 | 788 | 126 | 37 |

```
tabtrend(malform, c(0,0.5,1.5,4,7), transpose=FALSE)
## Cochran-Armitage Test auf Trend
##  Chi-Quadrat-Trend : 6.57  p = 0.01037041
##  Chi-Quadrat-Fehler: 5.512
##  Chi-Quadrat-gesamt: 12.082
```

Funktion prop.trend.test():

```
x <- malform[1,]; n <- malform[1,] + malform[2,]
prop.trend.test(x, n, c(0, 0.5, 1.5, 4, 7))
##
##  Chi-squared Test for Trend in Proportions
##
## data:  x out of n ,
##  using scores: 0 0.5 1.5 4 7
## X-squared = 6.5701, df = 1, p-value = 0.01037
```

### 7.7.10 Vergleich mehrerer Anteile mit einem Standard

```
ind <- function(d, dir="greater") {                              # Indikatorfunktion
       n <- length(d); s.p <- rep(NA, n); s.n <- rep(NA, n)
       for (i in 1:n) {
            if (d[i]>0) s.p[i]<-1 else s.p[i]<- 0
            if (d[i]<0) s.n[i]<-1 else s.n[i]<- 0 }
       if (dir=="greater") result <- s.p else
          if (dir=="less") result <- s.n else NA
       result
       }

prop.ref.test <- function(p.0, x, n, alternative="zweiseitig") {
    k <- length(x)
    if (alternative=="zweiseitig") {                             # zweiseitig
        B <- sum((x - n*p.0)^2/(n*p.0*(1-p.0)))
        p.val <- pchisq(B, df=k, lower.tail = FALSE)
        cat("B (zweiseitig) =",B,", ( P =",round(p.val,8),")\n")}
    else if (alternative=="größer") {                            # einseitig 'größer'
        diff <- x - n*p.0
        B.plus <- sum((diff*ind(diff, "greater"))^2/(n*p.0*(1-p.0)))
        theta <- pbinom(round(n*p.0), n, prob=p.0, lower.tail = FALSE)
        pv1   <- dbinom(1:k, k, prob=theta)
        pv2   <- pchisq(B.plus, df=1:k, lower.tail = FALSE)
        p.val <- sum(pv1*pv2)
        cat("B (einseitig größer) =",B.plus,", ( P =",round(p.val,8),")\n")}
    else if (alternative=="kleiner") {                          # einseitig 'kleiner'
        diff <- x - n*p.0
        B.min <- sum((diff*ind(diff, "less"))^2/(n*p.0*(1-p.0)))
        theta <- pbinom(round(n*p.0), n, prob=p.0, lower.tail = TRUE)
        pv1   <- dbinom(1:k, k, prob=theta)
        pv2   <- pchisq(B.min, df=1:k, lower.tail = FALSE)
        p.val <- sum(pv1*pv2)
        cat("B (einseitig kleiner) =",B.min,", ( P =",round(p.val,8),")\n")}
    }
```

Beispiel mit drei Anteilen zweiseitig:

```
p.0 <- 0.60
x <- c(59, 107, 48); n <- c(81, 151, 114)

prop.ref.test(p.0, x, n, alternative="zweiseitig")
## B (zweiseitig) = 28.19595 , ( P = 3.3e-06 )
```

Beispiel mit zwei Anteilen größer::

```
p.0 <- 0.60
x <- c(107, 48);  n <- c(151, 114)

prop.ref.test(p.0, x, n, alternative="größer")
## B (einseitig größer) = 7.421634 , ( P = 0.00917102 )
```

### 7.7.11 Die Analyse von Kontingenztafeln

Beispiel Therapieerfolge:

```
erfolg <- matrix(c(14, 22, 32, 18, 16, 8, 8, 2, 0), nr=3, byrow=T,
          dimnames = list(heilung=c("geheilt-x","geheilt-x+y","gestorben"),
          therapie=c("symptomatisch","spezifisch N1","spezifisch N2")))
as.data.frame(erfolg)
```

| | symptomatisch | spezifisch N1 | spezifisch N2 |
|---|---|---|---|
| geheilt-x | 14 | 22 | 32 |
| geheilt-x+y | 18 | 16 | 8 |
| gestorben | 8 | 2 | 0 |

```
chisq.test(erfolg, correct = TRUE)
##
##  Pearson's Chi-squared test
##
## data:  erfolg
## X-squared = 21.576, df = 4, p-value = 0.0002433
```

```
chisq.test(erfolg, simulate.p.value = TRUE, B = 1000)
##
##  Pearson's Chi-squared test with simulated p-value (based on 1000
##  replicates)
##
## data:  erfolg
## X-squared = 21.576, df = NA, p-value = 0.000999
```

Adjustierte Residuen:

```
n        <- sum(erfolg)
erwartet <- outer(rowSums(erfolg), colSums(erfolg), FUN="*")/n
erwartet <- round(erwartet, 2); erwartet
##             symptomatisch spezifisch N1 spezifisch N2
## geheilt-x           22.67         22.67         22.67
## geheilt-x+y         14.00         14.00         14.00
## gestorben            3.33          3.33          3.33
resid    <- erfolg - erwartet; resid
##              therapie
## heilung       symptomatisch spezifisch N1 spezifisch N2
##   geheilt-x           -8.67         -0.67          9.33
##   geheilt-x+y          4.00          2.00         -6.00
##   gestorben            4.67         -1.33         -3.33
p        <- outer(1-rowSums(erfolg)/n, 1-colSums(erfolg)/n, FUN="*")
adjust   <- resid / sqrt(erwartet * p1); round(adjust, 4)
##              therapie
## heilung       symptomatisch spezifisch N1 spezifisch N2
##   geheilt-x         -2.3910       -0.1848        2.5730
##   geheilt-x+y        1.4037        0.7019       -2.1056
##   gestorben          3.3603       -0.9570       -2.3961
stat     <- sum(resid^2/erwartet); stat
## [1] 21.58584
```

#### 7.7.11.1 Kontingenzkoeffizient (Cramer)

```
V.Cramer <- function(tab) {
  n     <- sum(tab)
  nrows <- rowSums(tab); r <- length(nrows)
  ncols <- colSums(tab); c <- length(ncols);   sum   <- 0
  for (i in 1:r) {
    for (j in 1:c) sum <- sum + tab[i,j]^2 / (nrows[i]*ncols[j]) }
  stat <- n * (sum - 1); stat; pval <- 1- pchisq(stat, df=(r-1)*(c-1))
  V <- sqrt(stat / (n * min((r-1), c-1)))
  cat("Chiquadrat =", stat, ", FG =", (r-1)*(c-1), ",
  P-Wert =", pval, "\n","Kontingenzkoeffizient nach Cramer V: =", V, "\n")
}
```

Beispiel Autotypen:

```
cartyp <- matrix(c( 5, 40, 50, 5, 15,  5,  7, 23), nrow=2,
                 ncol=4, byrow=TRUE,
                 dimnames=list(job=c("Arbeiter","Angestellter"),
                               typ=c("Cabrio", "Coupe","Kombi","SUV")))
as.data.frame(cartyp)
```

| | Cabrio | Coupe | Kombi | SUV |
|---|---|---|---|---|
| Arbeiter | 5 | 40 | 50 | 5 |
| Angestellter | 15 | 5 | 7 | 23 |

```
V.Cramer(cartyp)
## Chiquadrat = 67.01128 , FG = 3 ,
##   P-Wert = 1.865175e-14
##  Kontingenzkoeffizient nach Cramer V: = 0.6683875

chisq.test(cartyp)
##
##  Pearson's Chi-squared test
##
## data:  cartyp
## X-squared = 67.011, df = 3, p-value = 1.862e-14
```

#### 7.7.11.2 Fallzahl und Power

```
npow.chisq <- function(w=NULL, n=NULL, r=NULL, c=NULL, alpha=NULL, power=NULL) {
    nu <- (r-1)*(c-1)
    quant   <- qchisq(alpha, df=nu, lower=FALSE)
    p.expr  <- quote(pchisq(quant, df=nu, ncp = n * w^2, lower=FALSE))
    if (is.null(power)) power <- eval(p.expr)
    if (is.null(n))
      n <- uniroot(function(n) eval(p.expr)-power, c(1e-10, 1e+3))$root
    cat("\n","Fallzahl und Power zum Chiquadrat-Test:","\n",
        "Effektindex (w)..:",w,"\n",
        "Freiheitsgrade...:",nu,"\n",
        "Signifikanzniveau:",alpha,"\n",
        "Anzahl n (gesamt):",ceiling(n),"\n",
        "Power........... :",power,"\n")
}

npow.chisq(w=0.3, r=2, c=3, alpha=0.05, power=0.80)
##
##  Fallzahl und Power zum Chiquadrat-Test:
##  Effektindex (w)..: 0.3
##  Freiheitsgrade...: 2
##  Signifikanzniveau: 0.05
##  Anzahl n (gesamt): 108
##  Power........... : 0.8

npow.chisq(w=0.3, r=2, c=3, alpha=0.05, n=108)
##
##  Fallzahl und Power zum Chiquadrat-Test:
##  Effektindex (w)..: 0.3
##  Freiheitsgrade...: 2
##  Signifikanzniveau: 0.05
##  Anzahl n (gesamt): 108
##  Power........... : 0.8036939
```

Funktion pwr.chisq.test() in library(pwr)

```
library(pwr)
pwr.chisq.test(w=0.3 , df=(2-1)*(3-1), sig.level=0.05, power=0.80)
##
##      Chi squared power calculation
##
##               w = 0.3
##               N = 107.0521
##              df = 2
##       sig.level = 0.05
##           power = 0.8
##
## NOTE: N is the number of observations
```

### 7.7.12 Bowker-Test auf Symmetrie

```
bowker.test <- function(tab, k) {
    stat <- 0
    for (j in 1:(k-1)) {
        for (i in (j+1):k) {
    stat <- stat + ((tab[i,j] - tab[j,i])^2 / (tab[i,j] + tab[j,i]))
                       }     }
    pval <- pchisq(stat, df=k*(k-1)/2, lower.tail = FALSE)
    cat("Teststatistik:",stat,"(Signifikanz ",pval,")","\n")   }
```

Beispiel:

```
k <- 4
tab <- matrix(c(15,4,6,1, 16,10,3,1, 10,2,4,4, 0,4,12,8),
            nrow=k, byrow=T); tab
##      [,1] [,2] [,3] [,4]
## [1,]   15    4    6    1
## [2,]   16   10    3    1
## [3,]   10    2    4    4
## [4,]    0    4   12    8

bowker.test(tab, k)
## Teststatistik: 15.2 (Signifikanz  0.01875692 )
```

### 7.7.13 Marginalhomogenitätstest nach Lehmacher

```
lehmacher.test <- function(tab, k) {
    stat <- rep(NA, k)
    r.sum <- apply(tab, 1, sum); c.sum <- apply(tab, 2, sum)
    for (i in 1:k) stat[i] <- (r.sum[i] - c.sum[i])^2 /
                    (r.sum[i] + c.sum[i] - 2*tab[i,i])
    p.val <- round(pchisq(stat, df=1, lower.tail = FALSE), 6)
    p.cor <- round(p.adjust(p.val, "hochberg"), 6)
    cat("Teststatistik:",round(stat, 4),"\n",
        "Signifikanz  :",round(p.val,4),"\n",
        "P-adjustiert :",round(p.cor, 4),"\n")   }

wahl <- matrix(c(400,40,20,10, 50,300,60,20, 10,40,120,5, 5,90,50,80),
    nrow=4, byrow=T, dimnames =
    list(c("Partei A", "Partei B","Partei C","Partei D"),
         c("Partei A", "Partei B","Partei C","Partei D")))
wahl
##          Partei A Partei B Partei C Partei D
## Partei A      400       40       20       10
## Partei B       50      300       60       20
## Partei C       10       40      120        5
## Partei D        5       90       50       80

lehmacher.test(wahl, 4)
```

```
## Teststatistik: 0.1852 5.3333 30.4054 67.2222
##  Signifikanz  : 0.667 0.0209 0 0
##  P-adjustiert : 0.667 0.0418 0 0
bowker.test(wahl, 4)
## Teststatistik: 91.47475 (Signifikanz  1.496264e-17 )
```

Beispiel Wahlergebnisse:

```
wahl <- matrix(c(400,40,20,10, 50,300,60,20, 10,40,120,5, 5,90,50,80),
    nrow=4, byrow=T, dimnames =
    list(c("Partei A", "Partei B","Partei C","Partei D"),
         c("Partei A", "Partei B","Partei C","Partei D")))
as.data.frame(wahl)
```

| | Partei A | Partei B | Partei C | Partei D |
|---|---|---|---|---|
| Partei A | 400 | 40 | 20 | 10 |
| Partei B | 50 | 300 | 60 | 20 |
| Partei C | 10 | 40 | 120 | 5 |
| Partei D | 5 | 90 | 50 | 80 |

```
lehmacher.test(wahl, 4)
## Teststatistik: 0.1852 5.3333 30.4054 67.2222
##  Signifikanz  : 0.667 0.0209 0 0
##  P-adjustiert : 0.667 0.0418 0 0

bowker.test(wahl, 4)
## Teststatistik: 91.47475 (Signifikanz  1.496264e-17 )
```

### 7.7.14 Stuart-Maxwell-Test auf Homogenität in den Randverteilungen

```
stuart.maxwell.test <- function(tab) {
    if(nrow(tab)!=3 | ncol(tab)!=3) {                          # Prüfe Differenzen
        print("Dimension der Tabelle nicht 3*3"); break }
    rs <- rowSums(tab); cs <- colSums(tab)
    d1 <- rs[1] - cs[1]                                        # Randverteilungen
    d2 <- rs[2] - cs[2]
    d3 <- rs[3] - cs[3]
    n12 <- (tab[1,2] + tab[2,1])/2                             # Syymetriefelder
    n13 <- (tab[1,3] + tab[3,1])/2
    n23 <- (tab[2,3] + tab[3,2])/2
    stat <- (n23*d1^2+n13*d2^2+n12*d3^2)/(2*(n12*n13+n12*n23+n13*n23))
    pval <- pchisq(stat, df=2, lower.tail = FALSE)
    cat("Teststatistik:",stat,"(Signifikanz ",pval,")","\n")   }
```

Beispiel psychiatrische Störungen:

```
tab <- matrix(c(35,5,0,  15,20,5,  10,5,5),
              nrow=3, byrow=T, dimnames=list(P1=c("D1","D2","D3"),
                                             P2=c("D1","D2","D3")))
as.data.frame(addmargins(tab))
```

| | D1 | D2 | D3 | Sum |
|---|---|---|---|---|
| D1 | 35 | 5 | 0 | 40 |
| D2 | 15 | 20 | 5 | 40 |
| D3 | 10 | 5 | 5 | 20 |
| Sum | 60 | 30 | 10 | 100 |

```
stuart.maxwell.test(tab)
## Teststatistik: 14 (Signifikanz  0.000911882 )
```

Funktion stuart.maxwell.mh() in library(irr)

```
library(irr)
stuart.maxwell.mh(tab)
##  Stuart-Maxwell marginal homogeneity
##
##  Subjects = 100
##    Raters = 2
##     Chisq = 14
##
##  Chisq(2) = 14
##   p-value = 0.000912
```

Funktion mh_test() in library(coin)

```
library(coin)
mh_test(as.table(tab), distribution=approximate(B=9999))
##
##  Approximative Marginal Homogeneity Test
##
## data:  response by
##   conditions (P1, P2)
##   stratified by block
## chi-squared = 14, p-value = 3e-04
```

### 7.7.15 Q-Test nach Cochran

```
cochranq.test <- function(mat) {
  k <- ncol(mat);             C <- sum(colSums(mat)^2);
  R <- sum(rowSums(mat)^2);   T <- sum(rowSums(mat))
  Q <- (k - 1)*((k*C) - (T^2)) / (k*T - R)
  df <- k - 1;   names(df) <- "FG";   names(Q) <- "Cochran's Q"
  p.val <- pchisq(Q, df, lower = FALSE)
  QVAL <- list(statistic = Q, parameter = df, p.value = p.val,
               method = "Cochran's Q Test for Dependent Samples",
               data.name = deparse(substitute(mat)))
  class(QVAL) <- "htest"
  return(QVAL)
}
```

Beispiel Expertenurteil zu Weinen:

```
wein <- as.table(matrix(c(1,0,1,1,0, 1,1,1,0,1, 0,0,1,1,1, 1,0,1,0,0,
                          0,0,0,1,1, 1,0,1,1,0), byrow=TRUE, ncol=5,
                          dimnames = list("Person"=c("1","2","3","4","5","6"),
                                          "Wein" = c("A","B","C","D","E"))))
cochranq.test(wein)
##
##  Cochran's Q Test for Dependent Samples
##
## data:  wein
## Cochran's Q = 5.4118, FG = 4, p-value = 0.2476
```

### 7.7.16 Cohens's Kappa-Koeffizient

Funktion Kappa() in library(vcd)

```
library(vcd)
attention <- matrix(c(14, 3, 5, 18), nrow=2, ncol=2, byrow=TRUE)
attention
##      [,1] [,2]
## [1,]   14    3
## [2,]    5   18

Kappa(attention)
##             value    ASE     z  Pr(>|z|)
## Unweighted 0.597 0.1267 4.711 2.459e-06
## Weighted   0.597 0.1267 4.711 2.459e-06

confint(Kappa(attention))
##
## Kappa              lwr       upr
##   Unweighted 0.3486363 0.8453184
##   Weighted   0.3486363 0.8453184
```

Hinweis zu Konfidenzintervallen (Bootstrap):

Funktion ckappa() in library(psy)

```
library(psy); library(boot)
b1 <- c(rep(0,17), rep(1,23)); b2 <- c(rep(0,14),
                            rep(1,3), rep(0,5), rep(1,18))
attention <- as.data.frame(cbind(b1, b2))

ckappa(attention)$kappa
## [1] 0.5969773

ckappa.boot <- function(data, x) {ckappa(data[x,])[[2]]}
res  <- boot(attention, ckappa.boot, 500)

boot.ci(res, type="bca")
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 500 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = res, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 0.3052,  0.8000 )
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

#### 7.7.16.1 Das gewichtete Kappa

Beispiel Botulinum:

Funktion Kappa() in library(vcd)

```
botulin <- matrix(c(5,2,0,1,0, 1,7,2,2,0, 1,2,10,5,1,
                    0,0,3,4,0, 0,0,0,0,3),
                  nrow=5, ncol=5, byrow=TRUE,
                  dimnames = list("U1"=c("0","I","II","III","IV"),
                                  "U2"=c("0","I","II","III","IV")))
as.data.frame(botulin)
```

| | 0 | I | II | III | IV |
|---|---|---|---|---|---|
| 0 | 5 | 2 | 0 | 1 | 0 |
| I | 1 | 7 | 2 | 2 | 0 |
| II | 1 | 2 | 10 | 5 | 1 |
| III | 0 | 0 | 3 | 4 | 0 |
| IV | 0 | 0 | 0 | 0 | 3 |

```
library(vcd)
Kappa(botulin, weights = "Fleiss-Cohen")
##             value     ASE     z  Pr(>|z|)
## Unweighted 0.4651 0.09339 4.980 6.370e-07
## Weighted   0.6849 0.09569 7.158 8.216e-13

confint(Kappa(botulin, weights = "Fleiss-Cohen"))
##
## Kappa               lwr       upr
##   Unweighted 0.2820184 0.6481126
##   Weighted   0.4973423 0.8724326
```

#### 7.7.16.2 Das Kappa für mehrfache Beurteilungen (Multi-Rater)

Bewertung von Röntgenaufnamen:

```
radiol <- matrix(c(1,4,0,  2,0,3,   0,0,5,   4,0,1,    3,0,2,
                   1,4,0,  5,0,0,   0,4,1,   1,0,4,    3,0,2),
                   nrow=10, ncol=3, byrow=TRUE)

n     <- 10; R <- 5; k <- 3;
p.i   <- rep(NA, n);

for (i in 1:n) p.i[i] <- sum(radiol[i,]*(radiol[i,]-1))/(R*(R-1))
p.bar <- sum(p.i)/n; p.bar
## [1] 0.62

p.j   <- rep(NA, k); for (j in 1:k) p.j[j] <- sum(radiol[,j])/(n*R)
p.e   <- sum(p.j^2); p.e
```

```
## [1] 0.3472

kappa.m <- (p.bar - p.e)/(1-p.e)
kappa.m
## [1] 0.4178922

var <- (2/(n*R*(R-1))) * (p.e-(2*R-3)*p.e^2+2*(R-2)*sum(p.j^3))/(1-p.e)^2
var
## [1] 0.005872261

z   <- kappa.m / sqrt(var)
z
## [1] 5.453327

2*pnorm(z, lower.tail=FALSE)
## [1] 4.943598e-08
```

Funktion kappam.fleiss() in library(irr)

```
library(irr)
data <- matrix(c(1,2,2,2,2, 1,1,3,3,3, 3,3,3,3,3, 1,1,1,1,3, 1,1,1,3,3,
                 1,2,2,2,2, 1,1,1,1,1, 2,2,2,2,3, 1,3,3,3,3, 1,1,1,3,3),
                 nrow=10, byrow=T,
        dimnames=list(Bild=1:10,
                      Untersucher=c("U1","U2","U3","U4","U5")))

kappam.fleiss(data, exact = FALSE, detail = FALSE)
##  Fleiss' Kappa for m Raters
##
##  Subjects = 10
##    Raters = 5
##     Kappa = 0.418
##
##         z = 5.83
##   p-value = 5.47e-09
```

### 7.7.17 Krippendorff's Alpha

Beispiel Reliabilität mit Funktion kripp.alpha() in library(irr):

```
o1 <- c(1,  2, 3, 3, 2,  1, 4, 1, 2, NA, NA, NA)
o2 <- c(1,  2, 3, 3, 2,  2, 4, 1, 2,  5, NA,  3)
o3 <- c(NA, 3, 3, 3, 2,  3, 4, 2, 2,  5,  1, NA)
o4 <- c(1,  2, 3, 3, 2,  4, 4, 1, 2,  5,  1, NA)
dat <- rbind(Obs_A=o1, Obs_B=o2, Obs_C=o3, Obs_D=o4)
dat
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
## Obs_A    1    2    3    3    2    1    4    1    2    NA    NA    NA
## Obs_B    1    2    3    3    2    2    4    1    2     5    NA     3
## Obs_C   NA    3    3    3    2    3    4    2    2     5     1    NA
## Obs_D    1    2    3    3    2    4    4    1    2     5     1    NA

library(irr)
kripp.alpha(dat, method="ordinal")
##  Krippendorff's alpha
##
##  Subjects = 12
##    Raters = 4
##     alpha = 0.815
```

### 7.7.18 Kendall's Konkordanzkoeffizient

Beispiel Schönheitswettbewerb mit Funktion kendall() in library(irr)

```
library(irr)
ranking <- matrix(c(3,1,4,6,5,7,8,2,
                    2,3,5,7,4,6,8,1,
                    3,2,5,4,6,8,7,1), nrow=3, byrow=T)
kendall(t(ranking), correct=TRUE)
##  Kendall's coefficient of concordance Wt
##
##  Subjects = 8
##    Raters = 3
##        Wt = 0.894
##
##  Chisq(7) = 18.8
##   p-value = 0.00891

pchisq(18.8, 7, ncp=0, lower.tail = F, log.p = FALSE)
## [1] 0.008837491
```

## 7.8 Hypothesentests zur Korrelaton und Regression

### 7.8.1 Korrelationskoeffizient nach Pearson

```
x <- c( 4 , 6 , 8 , 3 , 9 , 5 , 10 , 2 , 7 , 8)
y <- c( 5 , 5 , 7 , 4 , 11 , 7 , 9 , 5 , 8 , 10)
n <- length(x)

r <- cor(x, y, method="pearson"); r
## [1] 0.8401183

t_hat <- r * sqrt((n-2)/(1-r^2)); t_hat
## [1] 4.380899

cor.test(x, y, method="pearson")
##
##  Pearson's product-moment correlation
##
## data:  x and y
## t = 4.3809, df = 8, p-value = 0.002346
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4468671 0.9612704
## sample estimates:
##       cor
## 0.8401183
```

Fisher-Transformtion von r auf z:

```
r <- seq(0, 0.99, 0.01); zp <- function(r) 0.5*log((1+r)/(1-r))
tab <- matrix(round(zp(r),4), byrow=T, nrow=10);
colnames(tab) <- seq(0.00, 0.09, 0.01)
rownames(tab) <- seq(0.0, 0.9, 0.1)
as.data.frame(tab[1:10,])
```

| | 0 | 0.01 | 0.02 | 0.03 | 0.04 | 0.05 | 0.06 | 0.07 | 0.08 | 0.09 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.0000 | 0.0100 | 0.0200 | 0.0300 | 0.0400 | 0.0500 | 0.0601 | 0.0701 | 0.0802 | 0.0902 |
| 0.1 | 0.1003 | 0.1104 | 0.1206 | 0.1307 | 0.1409 | 0.1511 | 0.1614 | 0.1717 | 0.1820 | 0.1923 |
| 0.2 | 0.2027 | 0.2132 | 0.2237 | 0.2342 | 0.2448 | 0.2554 | 0.2661 | 0.2769 | 0.2877 | 0.2986 |
| 0.3 | 0.3095 | 0.3205 | 0.3316 | 0.3428 | 0.3541 | 0.3654 | 0.3769 | 0.3884 | 0.4001 | 0.4118 |
| 0.4 | 0.4236 | 0.4356 | 0.4477 | 0.4599 | 0.4722 | 0.4847 | 0.4973 | 0.5101 | 0.5230 | 0.5361 |
| 0.5 | 0.5493 | 0.5627 | 0.5763 | 0.5901 | 0.6042 | 0.6184 | 0.6328 | 0.6475 | 0.6625 | 0.6777 |
| 0.6 | 0.6931 | 0.7089 | 0.7250 | 0.7414 | 0.7582 | 0.7753 | 0.7928 | 0.8107 | 0.8291 | 0.8480 |
| 0.7 | 0.8673 | 0.8872 | 0.9076 | 0.9287 | 0.9505 | 0.9730 | 0.9962 | 1.0203 | 1.0454 | 1.0714 |
| 0.8 | 1.0986 | 1.1270 | 1.1568 | 1.1881 | 1.2212 | 1.2562 | 1.2933 | 1.3331 | 1.3758 | 1.4219 |
| 0.9 | 1.4722 | 1.5275 | 1.5890 | 1.6584 | 1.7380 | 1.8318 | 1.9459 | 2.0923 | 2.2976 | 2.6467 |

Fisher-Transformtion von z auf r:

```
z <- seq(0, 3, 0.01); zr <- function(z) (exp(2*z)-1)/(exp(2*z)+1)
tab <- matrix(round(zr(z), 4), byrow=T, ncol=10)
colnames(tab) <- seq(0.00, 0.09, 0.01)
rownames(tab) <- seq(0.0, 3, 0.1)
as.data.frame(tab[1:10,])
```

| | 0 | 0.01 | 0.02 | 0.03 | 0.04 | 0.05 | 0.06 | 0.07 | 0.08 | 0.09 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.0000 | 0.0100 | 0.0200 | 0.0300 | 0.0400 | 0.0500 | 0.0599 | 0.0699 | 0.0798 | 0.0898 |
| 0.1 | 0.0997 | 0.1096 | 0.1194 | 0.1293 | 0.1391 | 0.1489 | 0.1586 | 0.1684 | 0.1781 | 0.1877 |
| 0.2 | 0.1974 | 0.2070 | 0.2165 | 0.2260 | 0.2355 | 0.2449 | 0.2543 | 0.2636 | 0.2729 | 0.2821 |
| 0.3 | 0.2913 | 0.3004 | 0.3095 | 0.3185 | 0.3275 | 0.3364 | 0.3452 | 0.3540 | 0.3627 | 0.3714 |
| 0.4 | 0.3799 | 0.3885 | 0.3969 | 0.4053 | 0.4136 | 0.4219 | 0.4301 | 0.4382 | 0.4462 | 0.4542 |
| 0.5 | 0.4621 | 0.4699 | 0.4777 | 0.4854 | 0.4930 | 0.5005 | 0.5080 | 0.5154 | 0.5227 | 0.5299 |
| 0.6 | 0.5370 | 0.5441 | 0.5511 | 0.5581 | 0.5649 | 0.5717 | 0.5784 | 0.5850 | 0.5915 | 0.5980 |
| 0.7 | 0.6044 | 0.6107 | 0.6169 | 0.6231 | 0.6291 | 0.6351 | 0.6411 | 0.6469 | 0.6527 | 0.6584 |
| 0.8 | 0.6640 | 0.6696 | 0.6751 | 0.6805 | 0.6858 | 0.6911 | 0.6963 | 0.7014 | 0.7064 | 0.7114 |
| 0.9 | 0.7163 | 0.7211 | 0.7259 | 0.7306 | 0.7352 | 0.7398 | 0.7443 | 0.7487 | 0.7531 | 0.7574 |

#### 7.8.1.2 Korrelation bei Mehrfachbeobachtungen

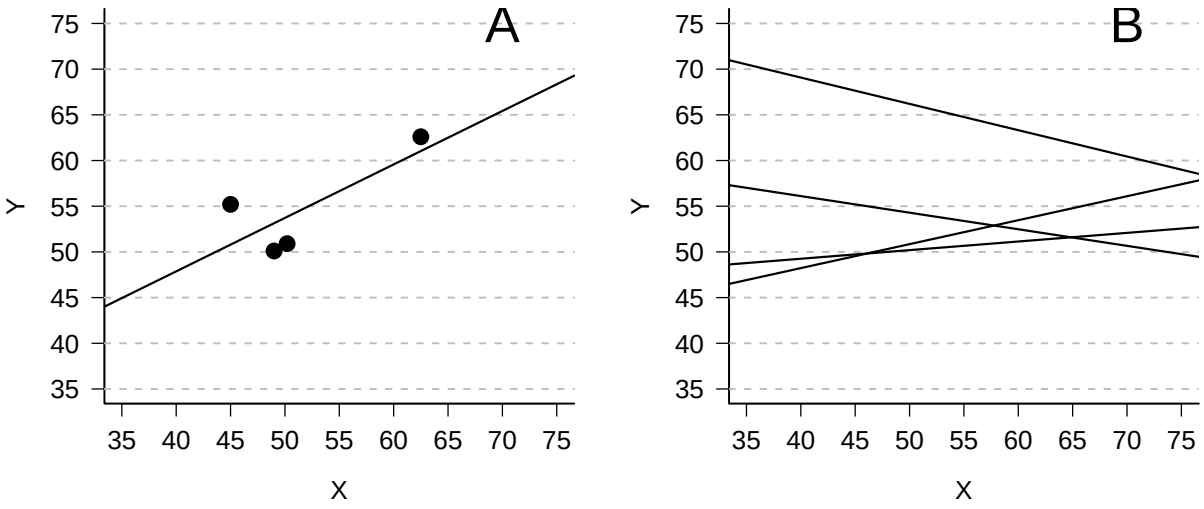
Beispieldatensatz nach J.M. Bland:

```
daten <- read.csv2("blandcor.csv")
subj  <- 1:4

attach(daten)
cor.test(y, x)                                          # Pearson Korrelation
##
##  Pearson's product-moment correlation
##
## data:  y and x
## t = 4.3809, df = 8, p-value = 0.002346
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4468671 0.9612704
## sample estimates:
##       cor
## 0.8401183

tmp  <- by(daten, id, function(d) cor(d$y, d$x))
korr <- tmp[1:4]; round(korr,3)                         # Einzelfälle
## id
##      1      2      3      4
## -0.333  0.486  0.065 -0.389
                                                        # Mittelwerte für jeden Fall
tmp.y <- by(daten, id, function(d) mean(d$y))
tmp.x <- by(daten, id, function(d) mean(d$x))
tmp.n <- by(daten, id, function(d) length(d$x))
cor.test(tmp.y, tmp.x)                                  # Korrelation aus Mittelwerten
```

```
##
##  Pearson's product-moment correlation
##
## data:  tmp.y and tmp.x
## t = 1.7177, df = 2, p-value = 0.228
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.7327773  0.9949069
## sample estimates:
##       cor
## 0.7720022
```

```
mx <- round(tmp.x, 2); my <- round(tmp.y, 2); w  <- tmp.n
korr.w <- (sum(w*mx*my) - sum(w*mx)*sum(w*my) / sum(w)) /
    sqrt( (sum(w*mx^2) - (sum(w*mx)^2/sum(w))) *
              (sum(w*my^2) - (sum(w*my)^2/sum(w))) )
korr.w                                                  # Pearson Korrelation gewichtet
## [1] 0.7720022
```

```
par(mfcol=c(1,2), lwd=1.5, font.axis=1.5, bty="l", ps=15)
plot(x[id==subj[1]], y[id==subj[1]], las=1,
     xlab="X", ylab="Y", xlim=c(35, 75),    ylim=c(35, 75),
     xaxp=c(35, 75, 8), yaxp=c(35, 75, 8), cex=1.0, pch=1)
for (i in 2:4) points(x[id==subj[i]], y[id==subj[i]], cex=1.0, pch=i)
abline(h=seq(35, 75, 5), lty=2, col="grey")
text(70,75,"A", cex=2)
points(tmp.x, tmp.y, pch=16, cex=1.8)                   # Regression zu Mittelwerten
abline(lm(tmp.y ~ tmp.x), lty=1, lwd=1.7)

plot(x[id==subj[1]], y[id==subj[1]], las=1,
     xlab="X", ylab="Y", xlim=c(35, 75),    ylim=c(35, 75),
     xaxp=c(35, 75, 8), yaxp=c(35, 75, 8), cex=1.5, pch=1)
for (i in 2:4) points(x[id==subj[i]], y[id==subj[i]], cex=1.5, pch=i)
abline(h=seq(35, 75, 5), lty=2, col="grey")
                                                        # Regression zu einzelnen Faellen
tmp  <- with(daten, by(daten, id, function(d) lm(y ~ x, data = d)))
regr <- sapply(tmp, coef)
for (i in 1:4) abline(a=regr[1,i], b=regr[2,i], lty=1, lwd=1.7)
text(70,75,"B", cex=2)
```




Korrelation innerhalb der Fälle:

```
vartab <- summary(aov(y ~ as.factor(id) + x, data=daten))
vartab
##               Df Sum Sq Mean Sq F value   Pr(>F)
## as.factor(id)  3  982.6   327.5  26.317 4.34e-09 ***
## x              1    0.2     0.2   0.016    0.901
## Residuals     35  435.6    12.4
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

saq    <- as.vector(vartab[[1]][2])
korr.w <- sqrt(saq[2,1] / (saq[2,1]+saq[3,1]))
korr.w
## [1] 0.02118205
```

#### 7.8.1.3 Fallzahl und Power zum Korrelationskoeffizienten

```
npwr.rho <- function(n=NULL, r, sig.lev=0.05, power=NULL) {
  if (sum(sapply(list(n, power), is.null)) != 1)
    stop("nur n oder power d?rfen fehlen")
  z.alpha <- qnorm(1-sig.lev/2)   # nur zweiseitige Hypothesenstellung
  if (is.null(n))         {
    z.beta  <- qnorm(power)
    n <- ((z.alpha + z.beta)/atanh(r))^2 + 3
    return(ceiling(n))   }
  if (is.null(power))       {
    z.beta <- sqrt(n-3)*atanh(r) - z.alpha
    power <- pnorm(z.beta)
    return(round(power, 4)) }
}
```

```
npwr.rho(r=0.60, power=0.90)
## [1] 25
```

Tabelle:

```
rho <- seq(0.1, 0.9, by=0.1); pwr <- seq(0.5, 0.9, by=0.1)
tab <- matrix(rep(0, 5*9), byrow=T, nrow=9)

for (i in 1:5) tab[,i] <- npwr.rho(r=rho, power=pwr[i], sig.lev=0.05)  # Alpha = 0.05
colnames(tab) <- seq(0.50, 0.90, 0.10)
rownames(tab) <- seq(0.10, 0.90, 0.10)
as.data.frame(tab)
```

| | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
|---|---|---|---|---|---|
| 0.1 | 385 | 490 | 617 | 783 | 1047 |
| 0.2 | 97 | 123 | 154 | 194 | 259 |
| 0.3 | 44 | 55 | 68 | 85 | 113 |
| 0.4 | 25 | 31 | 38 | 47 | 62 |
| 0.5 | 16 | 20 | 24 | 30 | 38 |
| 0.6 | 11 | 14 | 16 | 20 | 25 |
| 0.7 | 9 | 10 | 12 | 14 | 17 |
| 0.8 | 7 | 8 | 9 | 10 | 12 |
| 0.9 | 5 | 6 | 6 | 7 | 8 |

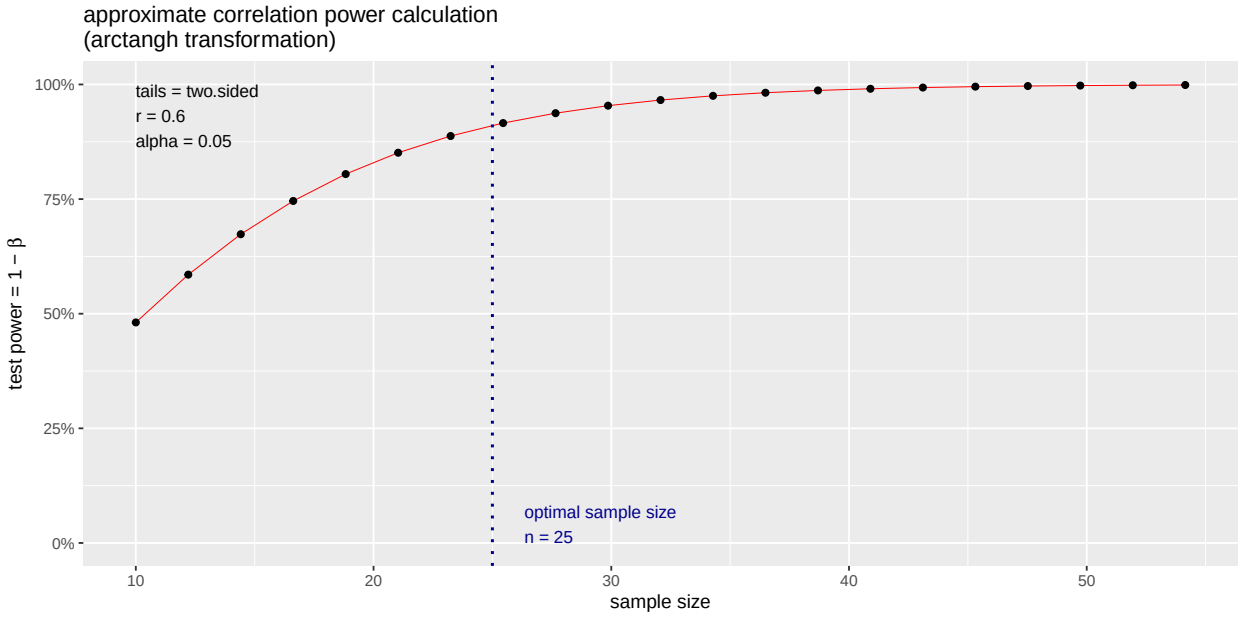
```
for (i in 1:5) tab[,i] <- npwr.rho(r=rho, power=pwr[i], sig.lev=0.01)  # Alpha = 0.01
colnames(tab) <- seq(0.50, 0.90, 0.10)
rownames(tab) <- seq(0.10, 0.90, 0.10)
as.data.frame(tab)
```

| | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
|---|---|---|---|---|---|
| 0.1 | 663 | 799 | 958 | 1164 | 1482 |
| 0.2 | 165 | 198 | 237 | 288 | 366 |
| 0.3 | 73 | 87 | 104 | 125 | 159 |
| 0.4 | 40 | 48 | 57 | 69 | 86 |
| 0.5 | 25 | 30 | 35 | 42 | 53 |
| 0.6 | 17 | 20 | 24 | 28 | 34 |
| 0.7 | 12 | 14 | 16 | 19 | 23 |
| 0.8 | 9 | 10 | 11 | 13 | 16 |
| 0.9 | 7 | 7 | 8 | 9 | 10 |

Funktion pwr.r.test() in library(pwr)

```
library(pwr)
prp <- pwr.r.test(n=NULL, r=0.60, power=0.90, alternative="two.sided")
prp
##
##      approximate correlation power calculation (arctangh transformation)
##
```

```
##                n = 24.14439
##                r = 0.6
##        sig.level = 0.05
##            power = 0.9
##      alternative = two.sided
plot(prp)
```



### 7.8.2 Prüfung der Rangkorrelationskoeffizienten

Schranken für Spearman und Kendall:

Funktion q.Kendall() und q.Spearm() in library(SuppDists)

```
library(SuppDists)                                    # einseitige obere Schranken

alpha <- c(0.005, 0.025, 0.05);  q <- 1 - alpha
n     <- 4:30

q.Kendall <- matrix(NA, nrow=27, ncol=3)
q.Spearm  <- matrix(NA, nrow=27, ncol=3)

for (i in 4:30) {
  for(j in 1:3) {
    q.Kendall[i-3, j] <- qKendall(q[j], n[i-3], lower.tail=TRUE)
    q.Spearm[i-3, j]  <- qSpearman(q[j], n[i-3], lower.tail=TRUE)
  } }
tab <- cbind(round(q.Spearm, 3), round(q.Kendall, 3))
colnames(tab) <- c("Spearman 0.01","Spearman 0.05","Spearman 0.10",
                   "Kendall 0.01","Kendall 0.05","Kendall 0.10")
rownames(tab) <- 4:30
as.data.frame(tab)
```

| | Spearman 0.01 | Spearman 0.05 | Spearman 0.10 | Kendall 0.01 | Kendall 0.05 | Kendall 0.10 |
|---|---|---|---|---|---|---|
| 4 | 1.000 | 1.000 | 0.800 | 1.000 | 1.000 | 0.667 |
| 5 | 1.000 | 0.900 | 0.800 | 1.000 | 0.800 | 0.600 |
| 6 | 1.000 | 0.943 | 0.829 | 0.867 | 0.733 | 0.600 |
| 7 | 0.929 | 0.821 | 0.750 | 0.810 | 0.619 | 0.524 |
| 8 | 0.881 | 0.762 | 0.667 | 0.714 | 0.571 | 0.500 |
| 9 | 0.833 | 0.700 | 0.617 | 0.667 | 0.500 | 0.444 |
| 10 | 0.782 | 0.648 | 0.576 | 0.600 | 0.467 | 0.422 |
| 11 | 0.755 | 0.618 | 0.536 | 0.564 | 0.455 | 0.382 |
| 12 | 0.720 | 0.587 | 0.507 | 0.545 | 0.424 | 0.364 |
| 13 | 0.692 | 0.560 | 0.484 | 0.538 | 0.410 | 0.359 |
| 14 | 0.670 | 0.538 | 0.464 | 0.495 | 0.385 | 0.341 |
| 15 | 0.645 | 0.521 | 0.446 | 0.486 | 0.371 | 0.314 |
| 16 | 0.626 | 0.503 | 0.432 | 0.467 | 0.367 | 0.300 |
| 17 | 0.610 | 0.485 | 0.417 | 0.456 | 0.353 | 0.294 |
| 18 | 0.593 | 0.472 | 0.404 | 0.438 | 0.333 | 0.281 |
| 19 | 0.578 | 0.458 | 0.391 | 0.427 | 0.322 | 0.275 |
| 20 | 0.564 | 0.447 | 0.380 | 0.411 | 0.316 | 0.263 |
| 21 | 0.550 | 0.434 | 0.371 | 0.400 | 0.305 | 0.257 |
| 22 | 0.539 | 0.425 | 0.362 | 0.385 | 0.299 | 0.255 |
| 23 | 0.528 | 0.415 | 0.353 | 0.383 | 0.289 | 0.249 |
| 24 | 0.516 | 0.406 | 0.345 | 0.370 | 0.283 | 0.239 |
| 25 | 0.506 | 0.398 | 0.338 | 0.360 | 0.280 | 0.233 |
| 26 | 0.497 | 0.389 | 0.331 | 0.354 | 0.274 | 0.231 |
| 27 | 0.487 | 0.382 | 0.324 | 0.350 | 0.265 | 0.225 |
| 28 | 0.479 | 0.375 | 0.318 | 0.339 | 0.259 | 0.222 |
| 29 | 0.471 | 0.368 | 0.312 | 0.335 | 0.256 | 0.217 |
| 30 | 0.464 | 0.362 | 0.307 | 0.329 | 0.251 | 0.214 |

Beispiel Weinsorten - Spearman:

```
x <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
y <- c(2, 1, 5, 3, 4, 6, 7, 9, 8, 10)

cor.test(x, y, method="spearman")
##
##  Spearman's rank correlation rho
##
## data:  x and y
## S = 10, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.9393939
```

Beispiel Weinsorten - Kendall:

```
x <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
y <- c(2, 1, 5, 3, 4, 6, 7, 9, 8, 10)

cor.test(x, y, method="kendall")
##
##  Kendall's rank correlation tau
##
## data:  x and y
## T = 41, p-value = 0.0003577
## alternative hypothesis: true tau is not equal to 0
## sample estimates:
##       tau
## 0.8222222
```

### 7.8.4 Hypothesentests zu den Regressionskoeffizienten

#### 7.8.4.1 Prüfung der Linearität

```
xi <- c(1, 5, 9, 13);  k <- length(xi)
ni <- c(2, 3, 1,  2);  n <- sum(ni)
yij <- matrix(c(1,2,NA, 2,3,3, 4,NA,NA, 5,6,NA), ncol=k, byrow=FALSE)

yisum <- rep(0, k)                                          # Gruppenmittelwerte
for (j in 1:k) {for (i in 1:ni[j]) yisum[j] <- yisum[j] + yij[i,j]}
yibar <- yisum / ni
                                                            # lineare Regression (x, y)
x <- NULL; for (j in 1:k) x <- c(x, rep(xi[j], ni[j]))
y <- NULL; for (j in 1:k) {for (i in 1:ni[j]) y <- c(y, yij[i,j])}
linreg <- lm(y~x); a <- linreg$coeff[1]; b <- linreg$coeff[2]
yihat    <- a + b*xi                                        # Schätzung

ZF <- (1/(k-2))*sum(ni*(yibar-yihat)^2)                     # Abweichungen
sn <- 0                                                     # vom Gruppenmittelwert
for (j in 1:k) {for (i in 1:ni[j]) sn <- sn + (yij[i,j] - yibar[j])^2}
NF <- (1/(n-k))*sn

F  <- ZF/NF; F                                              # Teststatistik F
## [1] 0.06582278
```

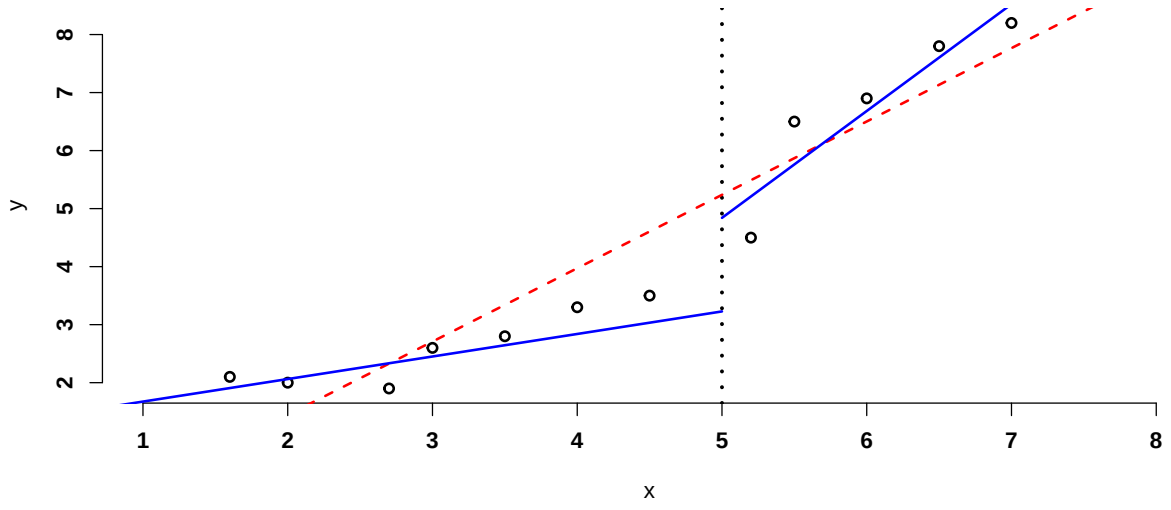
#### 7.8.4.2 Chow-Test: Strukturbruch

```
chow.test <- function(x, y, gruppe)  {
  n  <- length(x); k <- 2
  if (length(y) != n | length(gruppe) != n)
    stop("unterschiedliche Laengen in 'x', 'y' oder 'gruppe' ")
  x1 <- x[gruppe==1]; x2 <- x[gruppe==2]
  y1 <- y[gruppe==1]; y2 <- y[gruppe==2]
  ur.reg <- lm(y ~ x)                   # lineare Regressionsmodelle lm()
  r.reg1 <- lm(y1 ~ x1);  r.reg2 <- lm(y2 ~ x2)
  SSR = NULL                            # Summe der quadrierten. Residuen
  SSR$ur <- ur.reg$residuals^2; sum(SSR$ur)
  SSR$r1 <- r.reg1$residuals^2; sum(SSR$r1)
  SSR$r2 <- r.reg2$residuals^2; sum(SSR$r2)
  zaehler <- (sum(SSR$ur) - (sum(SSR$r1) + sum(SSR$r2))) / k
  nenner  <- (sum(SSR$r1) +  sum(SSR$r2)) / (n - 2*k)
  chow    <- zaehler / nenner;   P <- 1 - pf(chow, k, (n - 2*k))
  cat(sep="","\n","Chow-Test: F=",chow," mit ",k,";",n-2*k,
      " Freiheitsgraden (P-Wert=",P,")","\n")
}
```

Beispiel:

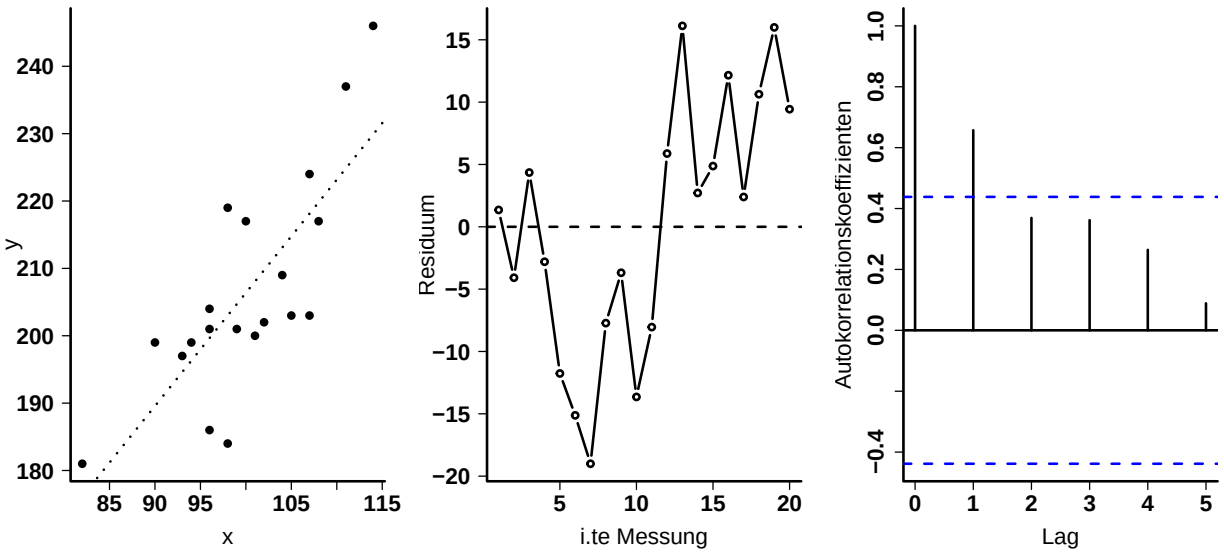
```
x <- c(1.6, 2.0, 2.7, 3.0, 3.5, 4.0, 4.5, 5.2, 5.5, 6.0, 6.5, 7.0)
y <- c(2.1, 2.0, 1.9, 2.6, 2.8, 3.3, 3.5, 4.5, 6.5, 6.9, 7.8, 8.2)

grp <- c(rep(1, 7), rep(2, 5))
chow.test(x, y, grp)
##
## Chow-Test: F=10.72839 mit 2;8 Freiheitsgraden (P-Wert=0.005440255)
```

```
n   <- length(x); cut <- 5; ol <- 8
groups <- c(rep(1, cut), rep(2, n-cut))
dfr <- as.data.frame(cbind(x,y,groups))

ur.reg <- lm(y ~ x, data = dfr)                              # Regression (alle Daten)
r.reg1 <- lm(y ~ x, data = dfr[1:cut,])                      # beschränkt auf  1:19
r.reg2 <- lm(y ~ x, data = dfr[cut+1:n,])                    # beschränkt auf 20:38

par(mfcol=c(1,1),lwd=2, font.axis=2, bty="n", ps=13)
plot(x, y, xlim=c(1,ol), ylim=range(y))
abline(v=cut, lwd=3, lty=3)
abline(ur.reg, col = "red", lwd = 2, lty = "dashed")         # volles Modell
segments(0, r.reg1$coefficients[1], cut,                     # Modell 1 / 2
   r.reg1$coefficients[1] + cut * r.reg1$coefficients[2], col= 'blue')
segments(cut, r.reg2$coefficients[1] + cut * r.reg2$coefficients[2],
   ol, r.reg2$coefficients[1] + ol * r.reg2$coefficients[2], col= 'blue')
```



#### 7.8.4.3 Durbin-Watson-Test: Autokorrelation in den Residuen

```
x <- c( 96,104, 96,108,105,107, 98,102, 99, 96,101,
        107,114, 94, 82,111, 93,100, 98, 90)
y <- c(201,209,204,217,203,203,184,202,201,186,200,
        224,246,199,181,237,197,217,219,199)
n <- length(x)

mod <- lm(y~x); e <- residuals(mod);                              # summary(mod)

par(mfcol=c(1,3), lwd=2, font.axis=2, bty="l", ps=20)
plot(x, y, las=1, xlab="x", ylab="y", cex=1.4, pch=16, col="black")
abline(mod, lty=3, col="black")
plot(1:n, e, type="b", las=1, xlab="i.te Messung", ylab="Residuum")
abline(h=0, lty=2, col="black")
acorr <- acf(e, lag.max=5, main=" ", ylab="Autokorrelationskoeffizienten")
```




```
mod <- lm(y~x); e <- residuals(mod)
d   <- rep(NA, n-1)
for (i in 2:n) d[i-1] <- (e[i] - e[i-1])
dw_stat <- sum(d^2)/sum(e^2); dw_stat                          # Durbin-Watson Test
## [1] 0.6408784
```

Funktion dwtest() in library(lmtest)

```
library(lmtest)
dwtest(mod)
##
##  Durbin-Watson test
##
```

```
## data:  mod
## DW = 0.64088, p-value = 0.0001659
## alternative hypothesis: true autocorrelation is greater than 0
```

Korrektur für Autokorrelation:

```
lag_e <- rep(NA, n-1); lag_y <- rep(NA,n-1); lag_x <- rep(NA, n-1)
for (i in 2:n) {lag_e[i] <- e[i-1];
                lag_y[i] <- y[i-1];
                lag_x[i] <- x[i-1] }
mc  <- lm(e ~ lag_e - 1)
rho <- mc$coefficients; rho                                  # Autokorrelationskoeffizient
##     lag_e
## 0.6872998
y_c <- y - rho*lag_y;   x_c <- x - rho*lag_x
modc<- lm(y_c ~ x_c); summary(modc)
##
## Call:
## lm(formula = y_c ~ x_c)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.3635  -7.1165   0.8942   7.1390   9.0663
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.2821     5.8068   0.393    0.699
## x_c           2.0007     0.1783  11.219  2.8e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.414 on 17 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.881,  Adjusted R-squared:  0.874
## F-statistic: 125.9 on 1 and 17 DF,  p-value: 2.796e-09
```