1 Analyse von Kontigenztafeln: Das loglineare Modell

Aufgabe 12

Analyse des Datensatzes HairEyeColor

(a) Visualisierung der Datenstruktur

```
# ?HairEyeColor
class(HairEyeColor)

## [1] "table"

HairEyeColor

## , , Sex = Male
##
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    32   11    10     3
##   Brown    53   50    25    15
##   Red      10   10     7     7
##   Blond     3   30     5     8
##
## , , Sex = Female
##
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    36    9     5     2
##   Brown    66   34    29    14
##   Red      16    7     7     7
##   Blond     4   64     5     8

# Variablen
# 1 Hair
# 2 Eye
# 3 Sex

# Als "flache" Kontingenztafel
ftable(HairEyeColor)

##             Sex Male Female
## Hair  Eye
## Black Brown       32     36
##       Blue        11      9
##       Hazel       10      5
##       Green        3      2
## Brown Brown       53     66
##       Blue        50     34
##       Hazel       25     29
##       Green       15     14
## Red   Brown       10     16
##       Blue        10      7
```

```
##       Hazel        7      7
##       Green        7      7
## Blond Brown        3      4
##       Blue        30     64
##       Hazel        5      5
##       Green        8      8
```

```
# Randhäufigkeiten
apply(HairEyeColor,1,sum)
```

```
## Black Brown   Red Blond
##   108   286    71   127
```

```
apply(HairEyeColor,2,sum)
```

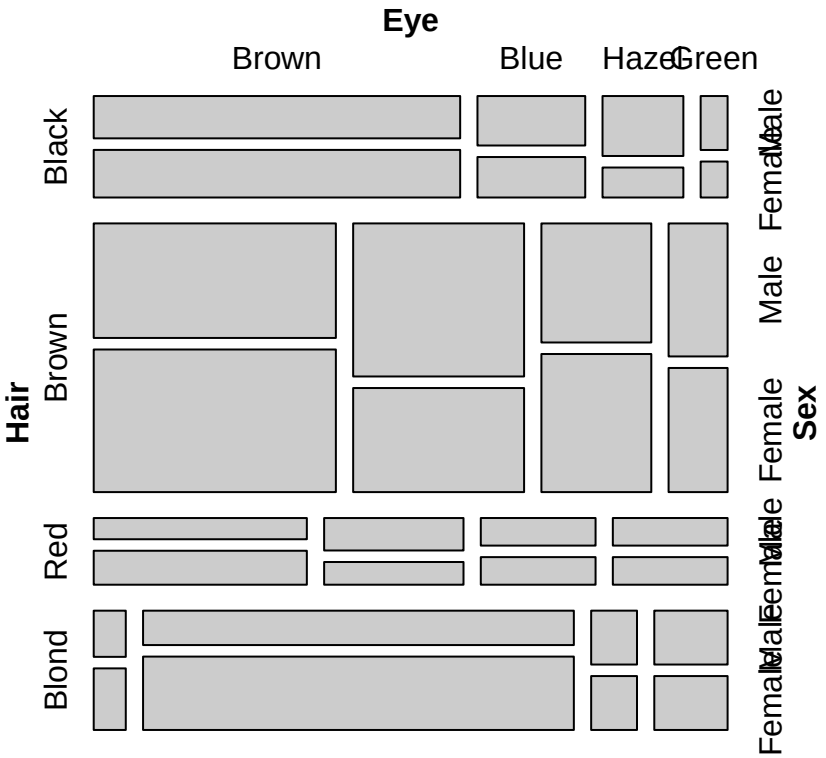
```
## Brown  Blue Hazel Green
##   220   215    93    64
```

```
apply(HairEyeColor,3,sum)
```

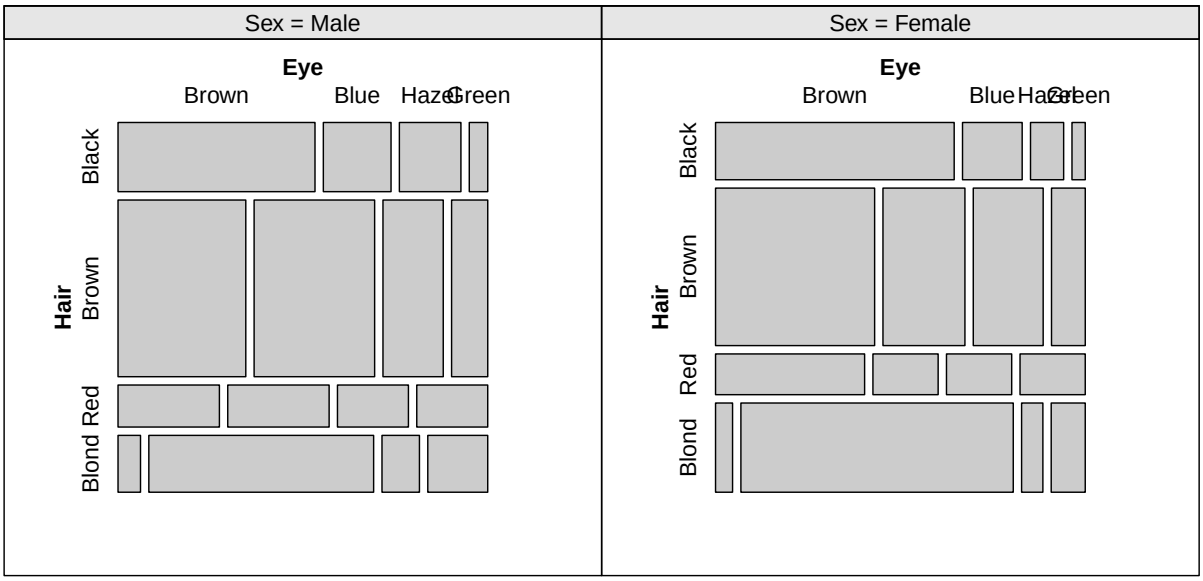
```
##   Male Female
##    279    313
```

Paket `vcd` zur Visualisierung kategorialer Daten

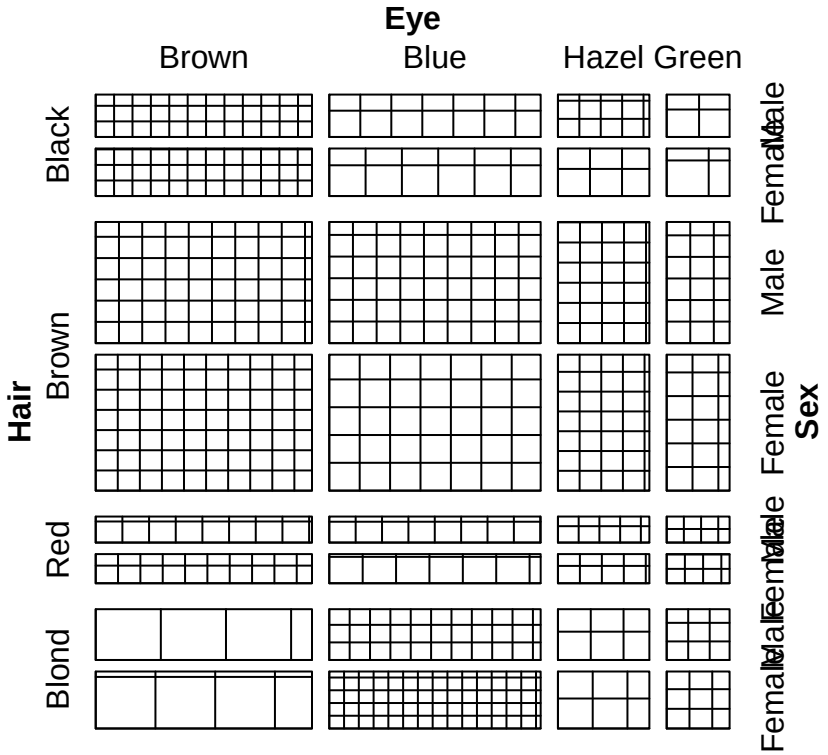
```
library(vcd)

# Darstellung der tatsächlichen Häufigkeiten
mosaic(HairEyeColor)
```



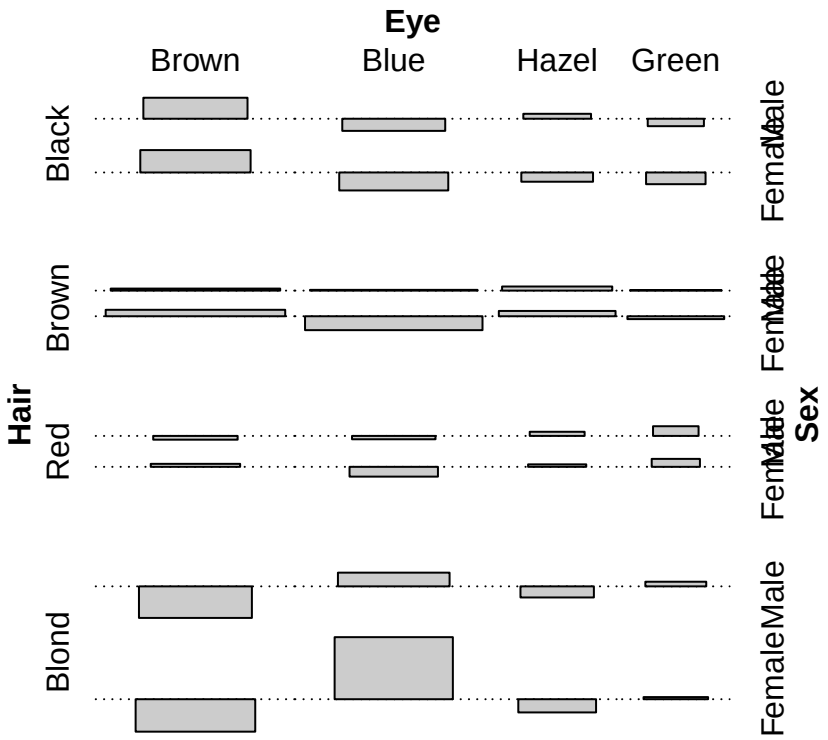
```r
cotabplot(~ Hair + Eye | Sex, data=HairEyeColor)
```



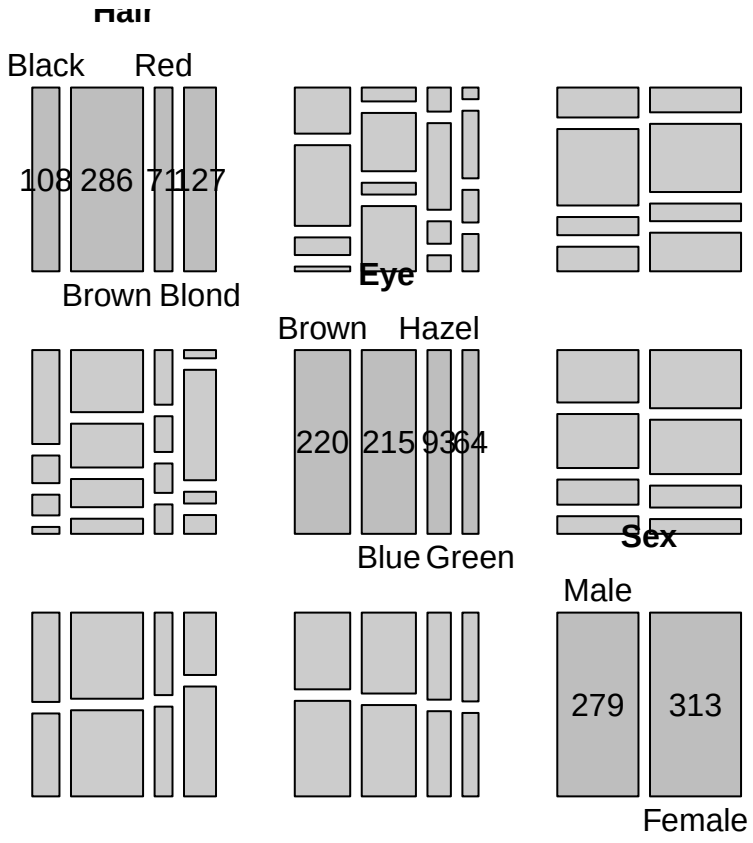

```r
# Plots, die tatsächliche Häufigkeiten mit unter (totaler) Unabhängigkeit zu
# erwartenden Häufigkeiten vergleichen
sieve(HairEyeColor)
```



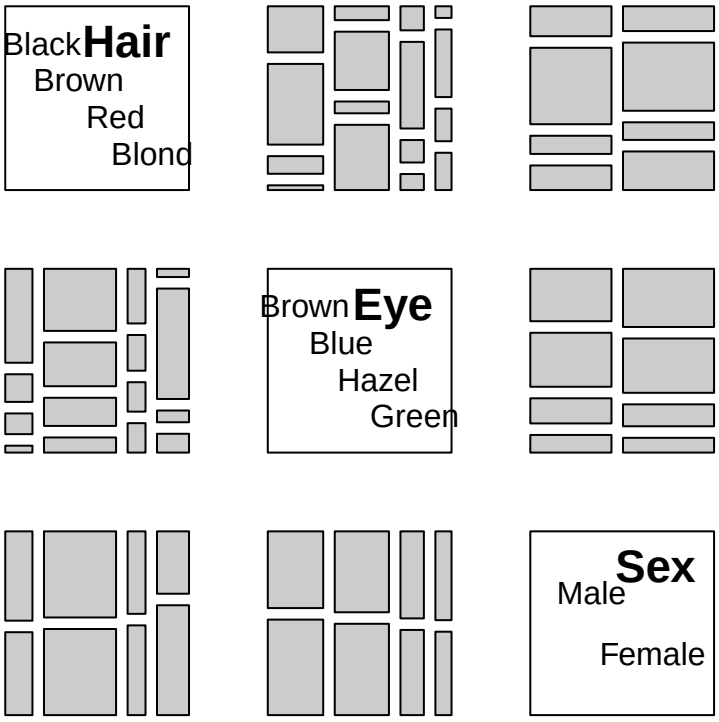

```r
assoc(HairEyeColor)
```



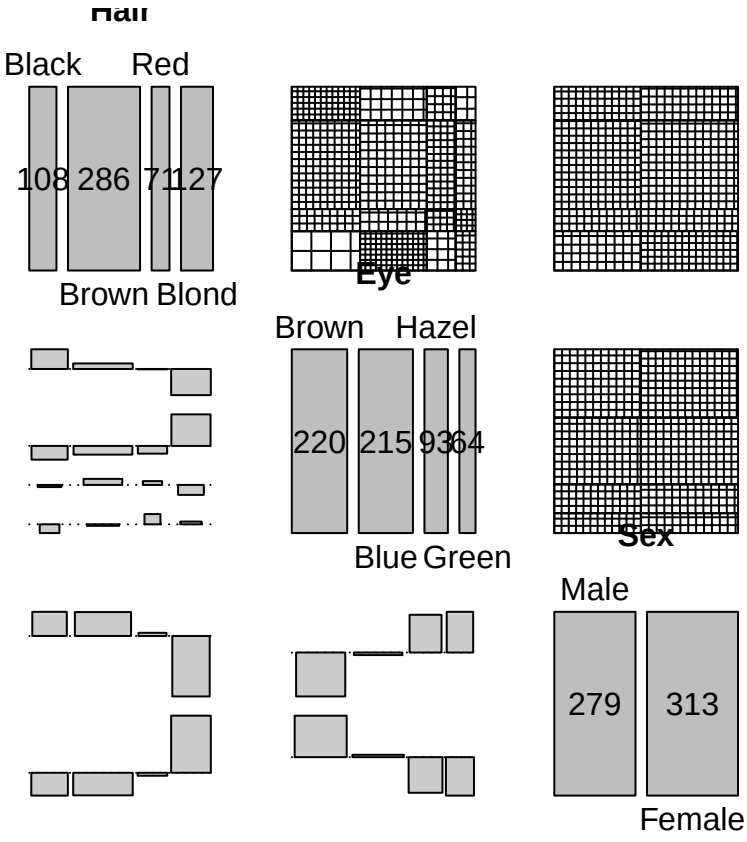

```
# Kombination von paarweisen Plots
pairs(HairEyeColor)
```




```
pairs(HairEyeColor, diag_panel=pairs_diagonal_text())
```

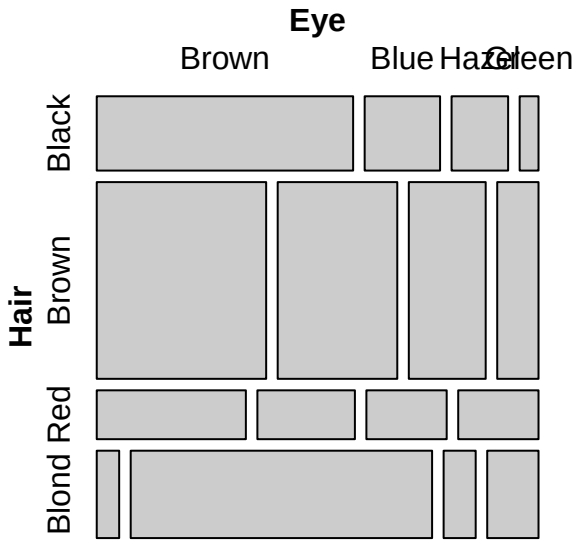



```
pairs(HairEyeColor, diag_panel=pairs_diagonal_mosaic(),
lower_panel=pairs_assoc(), upper_panel=pairs_sieve())
```



(b) Analyse ohne Variable *Sex*

```
# Aggregieren
HairEye <- HairEyeColor[,,1] + HairEyeColor[,,2]
mosaic(HairEye)
```



Loglineares Modell mit `loglin()`

```
# ?loglin
# Saturiertes modell
m0 <- loglin(HairEye, list(c(1,2)), fit=T, param=T)

## 2 iterations: deviation 0

m0

## $lrt
## [1] 0
##
## $pearson
## [1] 0
##
## $df
## [1] 0
##
## $margin
## $margin[[1]]
## [1] "Hair" "Eye"
##
##
## $fit
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    68   20    15     5
##   Brown   119   84    54    29
##   Red      26   17    14    14
##   Blond     7   94    10    16
##
## $param
```

```
## $param$`(Intercept)`
## [1] 3.189547
##
## $param$Hair
##      Black      Brown        Red      Blond
## -0.3063649  0.9520081 -0.3471908 -0.2984523
##
## $param$Eye
##      Brown       Blue      Hazel      Green
##  0.3611125  0.5112173 -0.2798778 -0.5924520
##
## $param$Hair.Eye
##        Eye
## Hair         Brown       Blue      Hazel      Green
##   Black  0.9752132 -0.3986671  0.1047460 -0.6812921
##   Brown  0.2764560 -0.2219556  0.1273068 -0.1818072
##   Red    0.0546279 -0.5203601  0.0765790  0.3891532
##   Blond -1.3062970  1.1409828 -0.3086318  0.4739461

# Haupteffekte
m1 <- loglin(HairEye, list(c(1),c(2)), fit=T, param=T)

## 2 iterations: deviation 0

m1

## $lrt
## [1] 146.4436
##
## $pearson
## [1] 138.2898
##
## $df
## [1] 9
##
## $margin
## $margin[[1]]
## [1] "Hair"
##
## $margin[[2]]
## [1] "Eye"
##
##
## $fit
##        Eye
## Hair        Brown      Blue    Hazel     Green
##   Black  40.13514  39.22297 16.96622 11.675676
##   Brown 106.28378 103.86824 44.92905 30.918919
##   Red    26.38514  25.78547 11.15372  7.675676
##   Blond  47.19595  46.12331 19.95101 13.729730
##
## $param
## $param$`(Intercept)`
## [1] 3.341678
##
## $param$Hair
##       Black       Brown         Red       Blond
## -0.17911627  0.79474431 -0.59856762 -0.01706041
##
## $param$Eye
##      Brown       Blue      Hazel      Green
##  0.5296905  0.5067010 -0.3313375 -0.7050540
```

```
# Test auf Unabhängigkeit über Pearson-Statistik
m1$pearson>qchisq(0.95,9)

## [1] TRUE

# Test auf Unabhängigkeit über Devianz
m1$lrt>qchisq(0.95,9)

## [1] TRUE
```

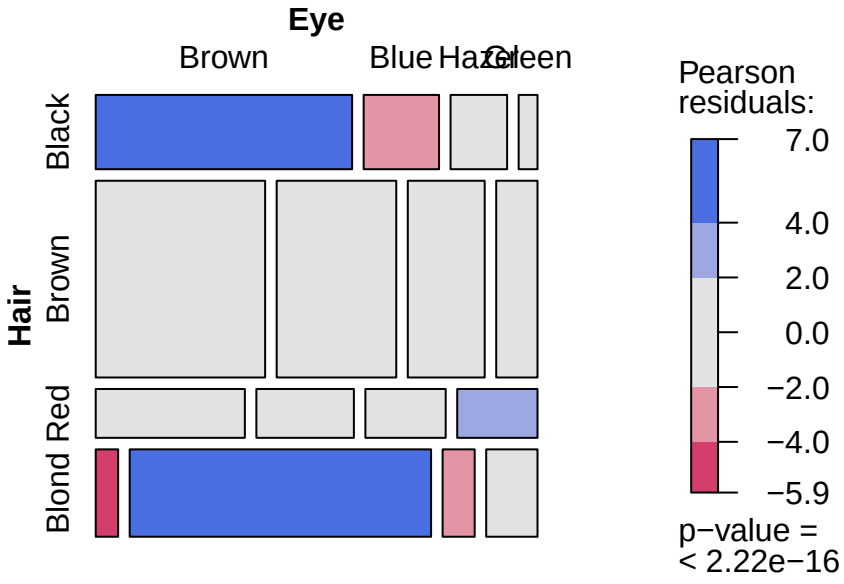
→Jeweils, Vergleich der Teststatistik mit dem 95%-Quantil der Chi-Quadrat-Verteilung mit 9 Freiheitsgraden

→Unabhängigkeitshypothese wird in beiden Fällen abgelehnt

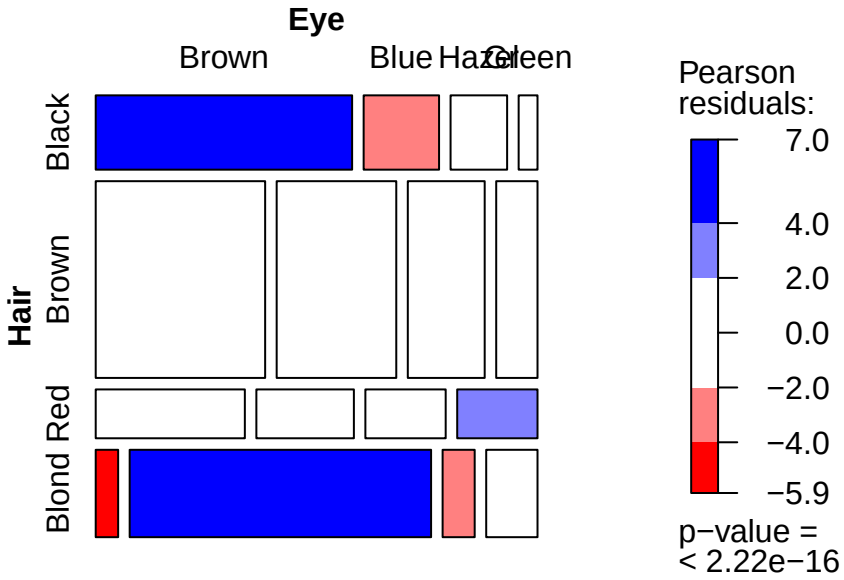
```
# Alternative: Berechnung der p-Werte:
1 - pchisq(m1$pearson,m1$df)

## [1] 0

1 - pchisq(m1$lrt,m1$df)

## [1] 0
```

→Unabhängigkeitshypothese wird in beiden Fällen abgelehnt, da p-Werte~ 0.

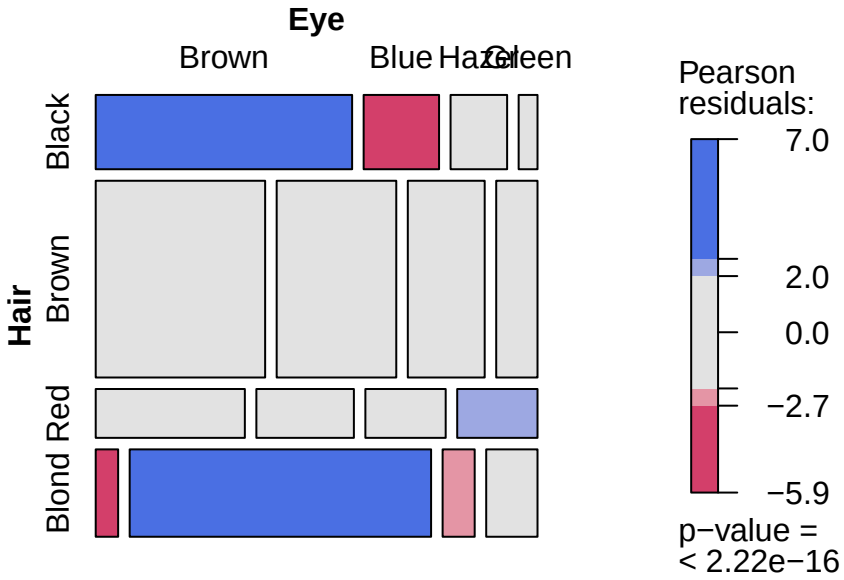
```
# Identifizierung der Zellen, die stark gegen Unabhängigkeitshypothese sprechen
mosaic(HairEye, shade=T) # shade=T äquivalent zu gp=shading_hcl, Defaultmethode
```



```
mosaic(HairEye, gp=shading_hsv)
```




```
mosaic(HairEye, gp=shading_max)
```



(c) Analyse der dreidimensionalen Tafel

```r
# Saturiertes Modell
fm0 <- loglin(HairEyeColor, list(c(1, 2, 3)), fit=T)

## 2 iterations: deviation 0

fm0

## $lrt
## [1] 0
##
## $pearson
## [1] 0
##
## $df
## [1] 0
##
## $margin
## $margin[[1]]
## [1] "Hair" "Eye"  "Sex"
##
##
## $fit
## , , Sex = Male
##
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    32   11    10     3
##   Brown    53   50    25    15
##   Red      10   10     7     7
##   Blond     3   30     5     8
##
## , , Sex = Female
##
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    36    9     5     2
##   Brown    66   34    29    14
##   Red      16    7     7     7
##   Blond     4   64     5     8

# Ohne 3-Faktor Interaktion
fm1 <- loglin(HairEyeColor, list(c(1, 2), c(1, 3), c(2, 3)), fit=T)

## 5 iterations: deviation 0.04093795

fm1

## $lrt
## [1] 6.761258
##
## $pearson
## [1] 6.868292
##
## $df
## [1] 9
##
## $margin
## $margin[[1]]
## [1] "Hair" "Eye"
```

```
##
## $margin[[2]]
## [1] "Hair" "Sex"
##
## $margin[[3]]
## [1] "Eye" "Sex"
##
##
## $fit
## , , Sex = Male
##
##        Eye
## Hair        Brown      Blue     Hazel     Green
##   Black 32.789022 11.743604  8.443926  3.018481
##   Brown 52.523050 45.931148 28.195250 16.348213
##   Red   10.760485  8.819775  6.916602  7.502671
##   Blond  1.927443 34.505472  3.444222  6.130636
##
## , , Sex = Female
##
##        Eye
## Hair        Brown      Blue     Hazel     Green
##   Black 35.208836  8.258511  6.556304  1.981916
##   Brown 66.477232 38.072151 25.804202 12.652225
##   Red   15.239924  8.180608  7.083293  6.497414
##   Blond  5.074007 59.488731  6.556201  9.868446
```

```
1 - pchisq(fm1$lrt,fm1$df)                                    # H_0 wird beibehalten
```

```
## [1] 0.66196
```

```
# Bedingte Unabhängigkeiten
fm2a <- loglin(HairEyeColor, list(c(1, 2), c(1, 3)))
```

```
## 2 iterations: deviation 7.105427e-15
```

```
fm2a
```

```
## $lrt
## [1] 11.76372
##
## $pearson
## [1] 11.77059
##
## $df
## [1] 12
##
## $margin
## $margin[[1]]
## [1] "Hair" "Eye"
##
## $margin[[2]]
## [1] "Hair" "Sex"
```

```
1 - pchisq(fm2a$lrt-fm1$lrt,fm2a$df-fm1$df)                   # H_0 wird beibehalten
```

```
## [1] 0.1716167
```

```
fm2b <- loglin(HairEyeColor, list(c(1, 2), c(2, 3)))
```

```
## 2 iterations: deviation 1.421085e-14
```

```
fm2b
```

```
## $lrt
## [1] 18.32715
##
## $pearson
## [1] 18.0411
##
## $df
## [1] 12
##
## $margin
## $margin[[1]]
## [1] "Hair" "Eye"
##
## $margin[[2]]
## [1] "Eye" "Sex"
```

```
1 - pchisq(fm2b$lrt-fm1$lrt,fm2b$df-fm1$df)              # H_0 wird abgelehnt
```

```
## [1] 0.009028299
```

```
fm2c <- loglin(HairEyeColor, list(c(1, 3), c(2, 3)))
```

```
## 2 iterations: deviation 2.842171e-14
```

```
fm2c
```

```
## $lrt
## [1] 156.6779
##
## $pearson
## [1] 147.944
##
## $df
## [1] 18
##
## $margin
## $margin[[1]]
## [1] "Hair" "Sex"
##
## $margin[[2]]
## [1] "Eye" "Sex"
```

```
1 - pchisq(fm2c$lrt-fm1$lrt,fm2c$df-fm1$df)              # H_0 wird abgelehnt
```

```
## [1] 0
```

```
# 2 Variablen gemeinsam unahb. von der dritten
fm3a <- loglin(HairEyeColor, list(c(1, 2), c(3)))
```

```
## 2 iterations: deviation 5.684342e-14
```

```
fm3a
```

```
## $lrt
## [1] 19.85656
##
## $pearson
## [1] 19.56712
##
## $df
## [1] 15
##
## $margin
## $margin[[1]]
## [1] "Hair" "Eye"
##
## $margin[[2]]
## [1] "Sex"
```

```
1 - pchisq(fm3a$lrt-fm2a$lrt,fm3a$df-fm2a$df)             # H_0 wird abgelehnt
```

```
## [1] 0.04413148
```

```
fm3b <- loglin(HairEyeColor, list(c(1, 3), c(2)))
```

```
## 2 iterations: deviation 7.105427e-15
```

```
fm3b
```

```
## $lrt
## [1] 158.2073
##
## $pearson
## [1] 150.0845
##
## $df
## [1] 21
##
## $margin
## $margin[[1]]
## [1] "Hair" "Sex"
##
## $margin[[2]]
## [1] "Eye"
```

```
1 - pchisq(fm3b$lrt-fm2a$lrt,fm3b$df-fm2a$df)             # H_0 wird abgelehnt
```

```
## [1] 0
```

```
# -> Endmodell 1 2 | 1 3 also Hair Eye | Hair Sex
```

Aufgabe 13

Analyse der Unfalldaten

(a) Speichern der Daten als R-Objekt

```
accident <- data.frame(expand.grid(
injury=factor(c("nonfatal","fatal"),levels=c("nonfatal","fatal")),
ejected=factor(c("yes","no"),levels=c("yes","no")),
safety=factor(c("belt","none"),levels=c("belt","none"))),
count=c(1105,14,411111,483,4624,497,157342,1008))
accident
```

```
##     injury ejected safety  count
## 1 nonfatal     yes   belt   1105
## 2    fatal     yes   belt     14
## 3 nonfatal      no   belt 411111
## 4    fatal      no   belt    483
## 5 nonfatal     yes   none   4624
## 6    fatal     yes   none    497
## 7 nonfatal      no   none 157342
## 8    fatal      no   none   1008
```

(b) Loglineare Modell mit *loglm()*

```
library(MASS)
# ?loglm

# Saturiertes Modell
fm0 <- loglm(count ~ injury*ejected*safety,data=accident,param=T,fit=T)
# Modell ohne 3-Faktor Interaktion
fm1 <- update(fm0,  ~ . - injury:ejected:safety)
# . steht für bisheriges Modell fm0: injury*ejected*safety,
# davon werden 3-Faktor-Interaktionen entfernt

# Geschätzte Parameter
fm0$param
```

```
## $`(Intercept)`
## [1] 7.78535
##
## $injury
##  nonfatal     fatal
##  2.299503 -2.299503
##
## $ejected
##       yes        no
## -1.711784  1.711784
##
## $safety
##       belt       none
## -0.5970266  0.5970266
##
## $injury.ejected
##           ejected
## injury            yes         no
##   nonfatal -0.6497609  0.6497609
##   fatal     0.6497609 -0.6497609
##
```

```
## $injury.safety
##           safety
## injury           belt       none
##   nonfatal  0.4792833 -0.4792833
##   fatal    -0.4792833  0.4792833
##
## $ejected.safety
##        safety
## ejected       belt       none
##     yes -0.6532103  0.6532103
##     no   0.6532103 -0.6532103
##
## $injury.ejected.safety
## , , safety = belt
##
##           ejected
## injury            yes         no
##   nonfatal  0.0552462 -0.0552462
##   fatal    -0.0552462  0.0552462
##
## , , safety = none
##
##           ejected
## injury            yes         no
##   nonfatal -0.0552462  0.0552462
##   fatal     0.0552462 -0.0552462
```

```
fm1$param
```

```
## $`(Intercept)`
## [1] 7.831967
##
## $injury
##  nonfatal     fatal
##  2.251692 -2.251692
##
## $ejected
##       yes        no
## -1.663278  1.663278
##
## $safety
##       belt       none
## -0.5489884  0.5489884
##
## $injury.ejected
##           ejected
## injury            yes         no
##   nonfatal -0.6994479  0.6994479
##   fatal     0.6994479 -0.6994479
##
## $injury.safety
##           safety
## injury           belt       none
##   nonfatal  0.4293303 -0.4293303
##   fatal    -0.4293303  0.4293303
##
## $ejected.safety
##        safety
## ejected      belt      none
##     yes -0.599909  0.599909
##     no   0.599909 -0.599909
```

```r
# Gefittete Werte
fm0$fit
```

```
## , , safety = belt
##
##           ejected
## injury      yes     no
##   nonfatal 1105 411111
##   fatal      14    483
##
## , , safety = none
##
##           ejected
## injury      yes     no
##   nonfatal 4624 157342
##   fatal     497   1008
```

```r
fm1$fit
```

```
## , , safety = belt
##
##           ejected
## injury            yes         no
##   nonfatal 1098.13199 411117.868
##   fatal      20.86801    476.132
##
## , , safety = none
##
##           ejected
## injury          yes         no
##   nonfatal 4630.869 157335.131
##   fatal     490.131   1014.869
```

Test über Devianz, ob Modell ohne 3-Faktor Interaktion passt

```r
1 - pchisq(fm1$lrt,fm1$df)             # H_0 wird beibehalten
```

```
## [1] 0.09114565
```

Interpretation über bedingte odds ratios

```r
OR <- function(X)
  {
  return((X[1,1]*X[2,2])/(X[1,2]*X[2,1]))
  }


# Verwende gefittete Werte:

# bedingt auf injury
fm1$fit[1,,]
```

```
##        safety
## ejected       belt       none
##     yes   1098.132   4630.869
##     no  411117.868 157335.131
```

```r
fm1$fit[2,,]
```

```
##        safety
## ejected      belt     none
##     yes  20.86801  490.131
##     no  476.13203 1014.869
```

```
apply(fm1$fit,1,OR)
```

```
## nonfatal    fatal
## 0.090751 0.090751
```

```
# bedingt auf ejected
fm1$fit[,1,]
```

```
##           safety
## injury           belt     none
##   nonfatal 1098.13199 4630.869
##   fatal      20.86801  490.131
```

```
fm1$fit[,2,]
```

```
##           safety
## injury           belt       none
##   nonfatal 411117.868 157335.131
##   fatal       476.132   1014.869
```

```
apply(fm1$fit,2,OR)
```

```
##      yes       no
## 5.569589 5.569589
```

```
# bedingt auf safety
fm1$fit[,,1]
```

```
##           ejected
## injury            yes         no
##   nonfatal 1098.13199 411117.868
##   fatal      20.86801    476.132
```

```
fm1$fit[,,2]
```

```
##           ejected
## injury          yes         no
##   nonfatal 4630.869 157335.131
##   fatal     490.131   1014.869
```

```
apply(fm1$fit,3,OR)
```

```
##      belt      none
## 0.0609445 0.0609445
```

→Bedingte Odds Ratios zweier Variablen auf jeder Stufe der dritten gleich (in allen drei Fällen)!

```
# Verwende reale Werte:

# bedingt auf injury
fatal <- matrix(accident[accident$injury=="fatal",][,4],nrow=2)
```

```r
nonfatal <- matrix(accident[accident$injury=="nonfatal",][,4],nrow=2)
OR(fatal)
```

```
## [1] 0.05878751
```

```r
OR(nonfatal)
```

```
## [1] 0.09145975
```

```r
# bedingt auf ejected
yes <- matrix(accident[accident$ejected=="yes",][,4],nrow=2)
no <- matrix(accident[accident$ejected=="no",][,4],nrow=2)
OR(yes)
```

```
## [1] 8.483456
```

```r
OR(no)
```

```
## [1] 5.452904
```

```r
# bedingt auf safety
belt <- matrix(accident[accident$safety=="belt",][,4],nrow=2)
none <- matrix(accident[accident$safety=="none",][,4],nrow=2)
OR(belt)
```

```
## [1] 0.09273043
```

```r
OR(none)
```

```
## [1] 0.05960426
```

→Beziehung von oben für reale Werte nur approximativ erfüllt!

(c) Analyse mit *glm()*

```r
# Poisson-Modell
glm1 <- glm(count ~ injury*ejected*safety - injury:ejected:safety,
data=accident, family=poisson)
summary(glm1)
```

```
##
## Call:
## glm(formula = count ~ injury * ejected * safety - injury:ejected:safety,
##     family = poisson, data = accident)
##
## Deviance Residuals:
##        1         2         3         4         5         6         7         8
##  0.20704  -1.59987  -0.01071   0.31400  -0.10095   0.30951   0.01731  -0.21583
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)              7.00137    0.02992  233.99   <2e-16 ***
## injuryfatal             -3.96315    0.06944  -57.07   <2e-16 ***
## ejectedno                5.92527    0.02996  197.76   <2e-16 ***
## safetynone               1.43913    0.03321   43.33   <2e-16 ***
## injuryfatal:ejectedno   -2.79779    0.05526  -50.63   <2e-16 ***
## injuryfatal:safetynone   1.71732    0.05402   31.79   <2e-16 ***
```

```
## ejectedno:safetynone   -2.39964    0.03334  -71.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1.6249e+06  on 7  degrees of freedom
## Residual deviance: 2.8540e+00  on 1  degrees of freedom
## AIC: 93.853
##
## Number of Fisher Scoring iterations: 3

# Vergleich gefittete Werte
accident

##     injury ejected safety  count
## 1 nonfatal     yes   belt   1105
## 2    fatal     yes   belt     14
## 3 nonfatal      no   belt 411111
## 4    fatal      no   belt    483
## 5 nonfatal     yes   none   4624
## 6    fatal     yes   none    497
## 7 nonfatal      no   none 157342
## 8    fatal      no   none   1008

glm1$fit

##            1            2            3            4            5            6
##   1098.13193     20.86807 411117.86807    476.13193   4630.86807    490.13193
##            7            8
## 157335.13193   1014.86807

fm1$fit

## , , safety = belt
##
##           ejected
## injury            yes         no
##   nonfatal 1098.13199 411117.868
##   fatal      20.86801    476.132
##
## , , safety = none
##
##           ejected
## injury          yes         no
##   nonfatal 4630.869 157335.131
##   fatal     490.131   1014.869

# Vergleich Parameter
glm1$coef

##            (Intercept)            injuryfatal              ejectedno
##               7.001366              -3.963146               5.925269
##             safetynone  injuryfatal:ejectedno injuryfatal:safetynone
##               1.439134              -2.797795               1.717321
##   ejectedno:safetynone
##              -2.399636

fm1$param
```

```
## $`(Intercept)`
## [1] 7.831967
##
## $injury
##  nonfatal     fatal
##  2.251692 -2.251692
##
## $ejected
##       yes        no
## -1.663278  1.663278
##
## $safety
##       belt       none
## -0.5489884  0.5489884
##
## $injury.ejected
##           ejected
## injury            yes         no
##   nonfatal -0.6994479  0.6994479
##   fatal     0.6994479 -0.6994479
##
## $injury.safety
##           safety
## injury           belt       none
##   nonfatal  0.4293303 -0.4293303
##   fatal    -0.4293303  0.4293303
##
## $ejected.safety
##        safety
## ejected      belt      none
##     yes -0.599909  0.599909
##     no   0.599909 -0.599909
```

→Gefittete Werte sind identisch, Parameter unterscheiden sich aber, da loglm() Effektkodierung verwendet, glm() per Default jedoch Dummy-Kodierung mit Referenzkategorien safety=belt, ejected=yes, injury=nonfatal.

⇒ Poisson-Modell mit Effektkodierung:

```
# Modell mit Effektkodierung
glm2 <- glm(count ~ injury*ejected*safety - injury:ejected:safety,
            data=accident, family=poisson,
            contrast=list(safety=contr.sum(2),ejected=contr.sum(2),injury=contr.sum(2)))

# Vergleich Parameter
glm2$coef
```

```
##      (Intercept)          injury1         ejected1          safety1
##        7.8319672        2.2516913       -1.6632772       -0.5489882
## injury1:ejected1  injury1:safety1 ejected1:safety1
##       -0.6994486        0.4293302       -0.5999089
```

```
fm1$param
```

```
## $`(Intercept)`
## [1] 7.831967
##
## $injury
##  nonfatal     fatal
##  2.251692 -2.251692
```

```
##
## $ejected
##       yes        no
## -1.663278  1.663278
##
## $safety
##       belt       none
## -0.5489884  0.5489884
##
## $injury.ejected
##           ejected
## injury            yes         no
##   nonfatal -0.6994479  0.6994479
##   fatal     0.6994479 -0.6994479
##
## $injury.safety
##           safety
## injury           belt       none
##   nonfatal  0.4293303 -0.4293303
##   fatal    -0.4293303  0.4293303
##
## $ejected.safety
##        safety
## ejected      belt      none
##     yes -0.599909  0.599909
##     no   0.599909 -0.599909

# -> Modelle nun identisch
```