

# Correspondence Analysis

In practice

Stéphane Dray

2024-01-08



# Data

We consider the relationship between hair and eyes colors measured on 592 students in Delaware University.

```
mydata <- read.table("http://pbil.univ-lyon1.fr/R/donnees/snee74.txt",  
  header = TRUE, stringsAsFactors = TRUE)  
names(mydata)
```

```
## [1] "cheveux" "yeux"      "sexe"
```

```
head(mydata)
```

```
##   cheveux   yeux   sexe  
## 1    Noir Marron  Male  
## 2   Blond   Bleu Femelle  
## 3    Noir   Bleu  Male  
## 4 Marron Marron Femelle  
## 5    Roux Marron  Male  
## 6 Marron   Bleu  Male
```

# Factors in R

A qualitative variables is stored as **factor** with different categories (**levels**)

```
is.factor(mydata$yeux)
```

```
## [1] TRUE
```

```
levels(mydata$yeux)
```

```
## [1] "Bleu"      "Marron"    "Noisette"  "Vert"
```

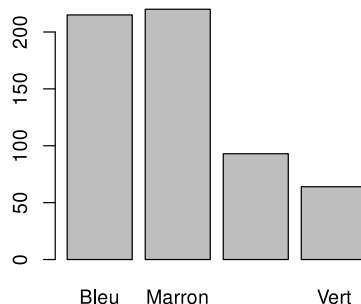
# Univariate analysis

- Compute the vector with the number of individuals for each eye color category and display the results on a plot.

```
summary(mydata$yeux)
```

##	Bleu	Marron	Noisette	Vert
##	215	220	93	64

```
plot(mydata$yeux)
```



# Contingency table

- Build the contingency table crossing hair and eyes colors. See `?table`

```
mytab <- table(mydata$scheveux, mydata$yeux)
```

- Compute row and columns marginal relative frequencies

```
rowSums(mytab)/sum(mytab)
```

```
##      Blond      Marron      Noir      Roux  
## 0.2145270 0.4831081 0.1824324 0.1199324
```

```
colSums(mytab)/sum(mytab)
```

```
##      Bleu      Marron      Noisette      Vert  
## 0.3631757 0.3716216 0.1570946 0.1081081
```

# Bivariate analysis

- Compute and interpret the results of a  $\chi^2$  test

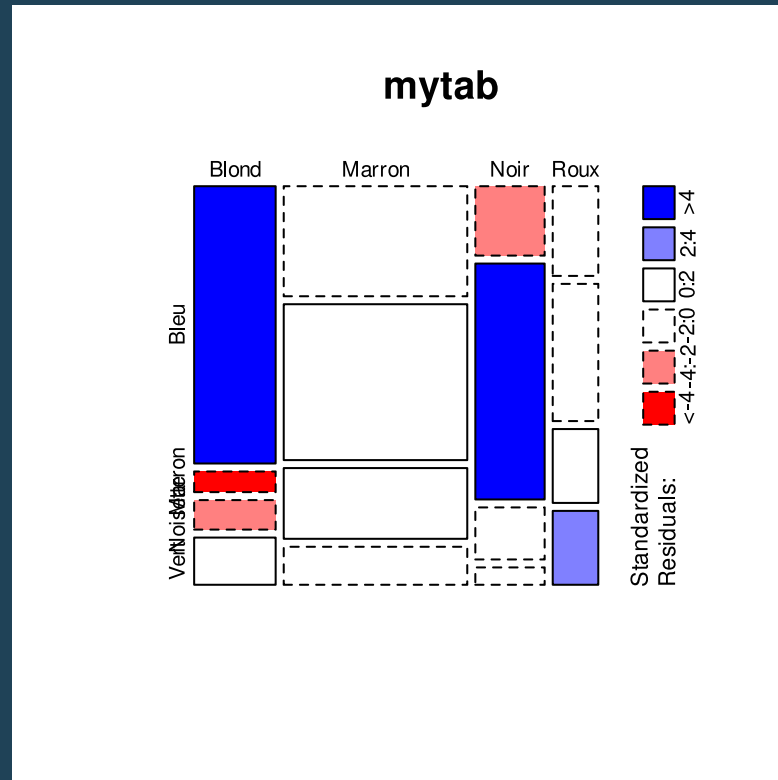
```
chisq.test(mytab)
```

```
##  
##      Pearson's Chi-squared test  
##  
## data:  mytab  
## X-squared = 138.29, df = 9, p-value < 2.2e-16
```

# Bivariate analysis

- Display and interpret the associations between categories with `mosaicplot`

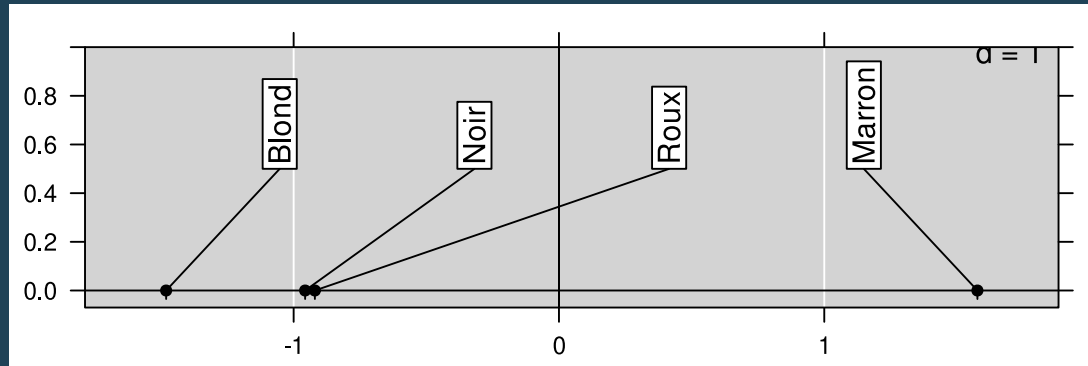
```
mosaicplot(mytab, shade = TRUE)
```



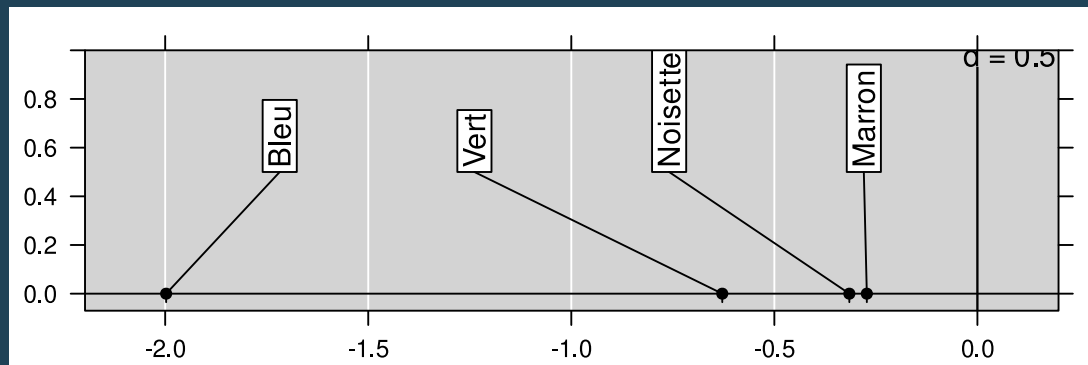


# Scoring

Suppose that we can display hair colors by a score



what would be the best way to display eyes colors? (given the distribution of students)



# Weighted averaging

- Assign a random score to hair colors

```
set.seed(12)
score.h <- rnorm(4)
```

- Compute the position for the eye color 'Vert'

```
sum(score.h * mytab[, 4]/sum(mytab[, 4]))
```

```
## [1] 0.06851621
```

```
## or
```

```
weighted.mean(score.h, w = mytab[, 4])
```

```
## [1] 0.06851621
```

# Compute scores for all eye colors

By repeating the same formulas

```
score.e <- sapply(1:4, function(x) sum(score.h * mytab[,  
  x]/sum(mytab[, x])))
```

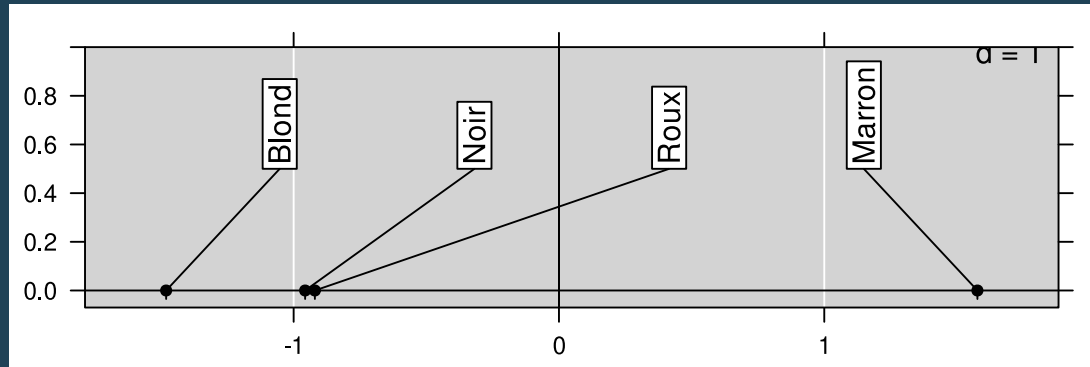
or using matrix algebra

```
t(prop.table(mytab, 2)) %*% score.h
```

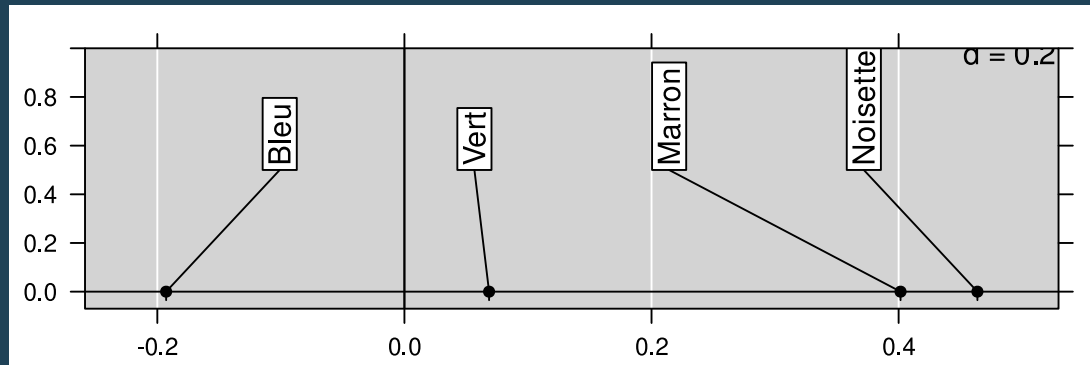
```
##  
##           [,1]  
## Bleu      -0.19286557  
## Marron     0.40154742  
## Noisette   0.46376597  
## Vert       0.06851621
```

# Best representation

if



then



# Reciprocal representation

- Start from a random score for eyes colors to position hair colors by averaging

```
set.seed(13)
score.e <- rnorm(4)
prop.table(mytab, 1) %*% score.e
```

```
##
##           [,1]
## Blond  0.5582172
## Marron 0.4003574
## Noir   0.1814083
## Roux   0.4170599
```

# Reciprocal averaging

1. Set a random score for columns
2. Use the column score to compute a score for rows by weighted averaging
3. Compute a new score for columns by weighted averaging of row score
4. Center and scale the two scores using row and column weights
5. Repeat steps 2-4 until convergence

# Correspondence analysis

The solution of the iterative algorithm can also be obtained by the diagonalization procedure provided by the `dudi.coa` function

```
library(ade4)
coa1 <- dudi.coa(unclass(mytab), scannf = FALSE)
```

- Check the link between  $\chi^2$  statistic and total inertia

```
sum(coa1$eig)
```

```
## [1] 0.2335977
```

```
chisq.test(mytab)$statistic/sum(mytab)
```

```
## X-squared
```

```
## 0.2335977
```

# Interpretation

- Compute a row score with unit norm ( $\$l1$ ) to obtain a column score by weighted averaging ( $\$co$ ) with maximal variance ( $\$eig$ )
- Compute a row score with unit norm ( $\$c1$ ) to obtain a column score by weighted averaging ( $\$li$ ) with maximal variance ( $\$eig$ )
- Check both results

```
prop.table(mytab, 1) %*% coal$c1[, 1]
```

```
##  
##           [,1]  
## Blond    0.8353478  
## Marron  -0.1482527  
## Noir     -0.5045624  
## Roux     -0.1295233
```

```
coal$li[, 1]
```

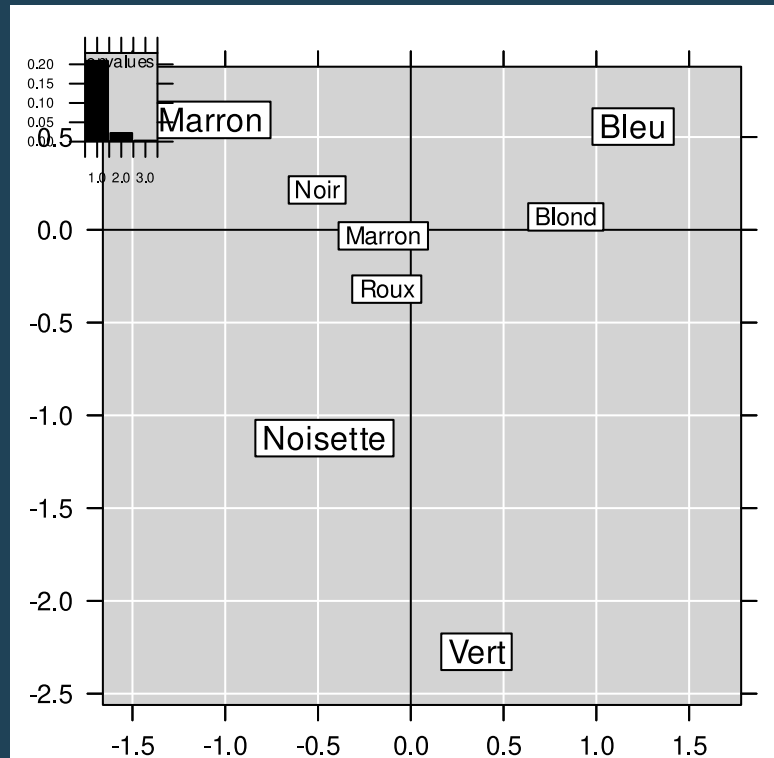
```
## [1] 0.8353478 -0.1482527 -0.5045624 -0.1295233
```



# Graphical representation

- Represent and interpret the results using the `scatter` function (check the `method` argument of `scatter.coa`)

```
scatter(coa1, method = 2)
```



# Inertia statistics

- Use the `inertia.dudi` function to compute inertia statistics for rows and columns

```
ic = inertia.dudi(coal, row.inertia = TRUE, col.inertia = TRUE)
```

- Represent absolute contributions using the `plot` function (see ?  
`plot.inertia`)

```
plot(ic, contrib = "abs")
```

