

# Correspondence Analysis

In practice

Stéphane Dray

2021-11-12



# 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 variable 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.

# Contingency table

- Build the contingency table crossing hair and eyes colors. See `?table`
- Compute row and columns marginal relative frequencies

# Bivariate analysis

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

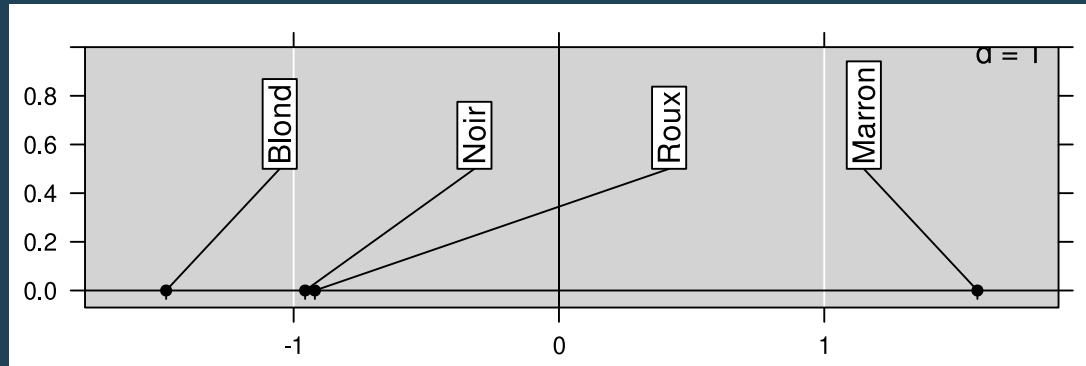
# Bivariate analysis

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

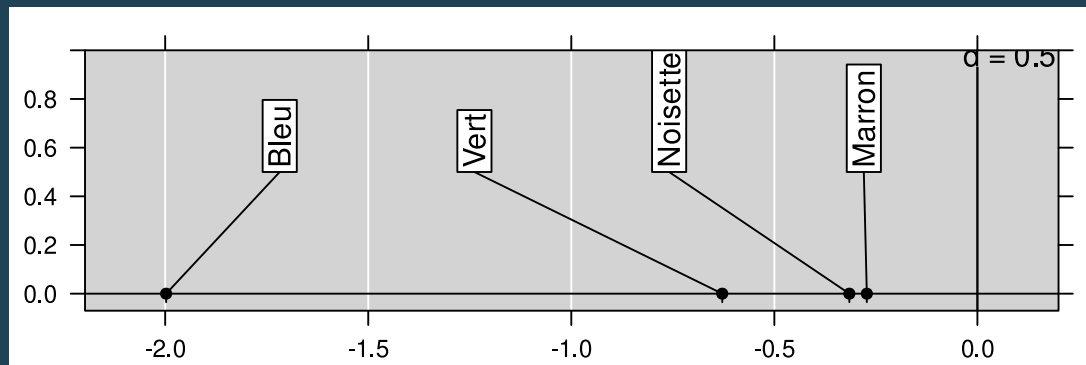


# Scoring

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
- Compute the position for the eye color 'Vert'

# 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])))
```

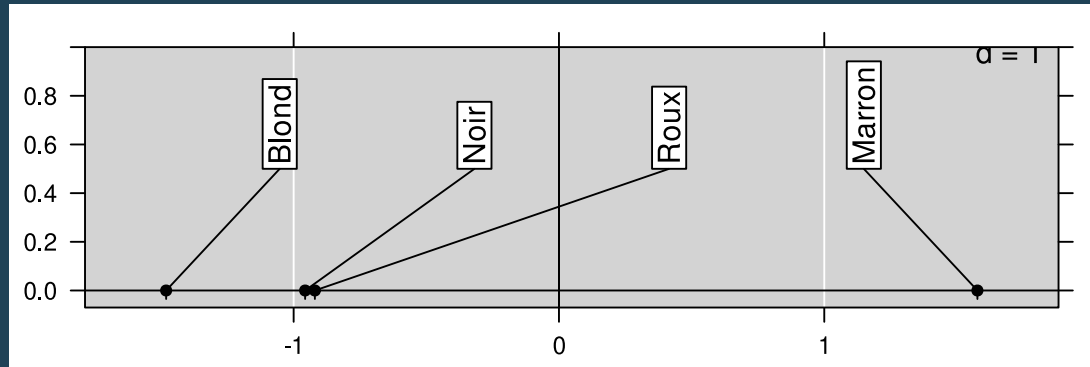
or using matrix algebra

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

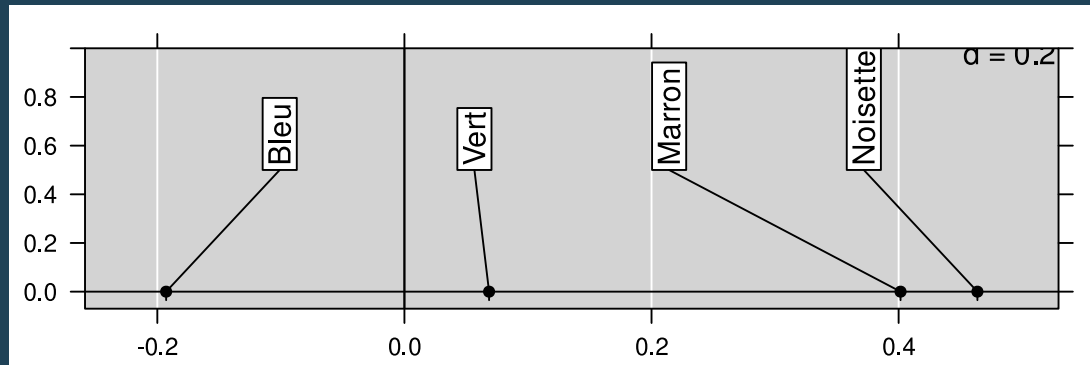
```
##  
##           [,1]  
## Bleu      -0.19286557  
## Marron     0.40154742  
## Noisette   0.46376597  
## Vert       0.06851621
```

# Best representation

if



then



# Reciprocal representation

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

# 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

# Interpretation

- Compute a row score with unit norm ( $\mathbf{l}_1$ ) to obtain a column score by weighted averaging ( $\mathbf{c}_o$ ) with maximal variance ( $\mathbf{e}_i$ )
- Compute a row score with unit norm ( $\mathbf{c}_1$ ) to obtain a column score by weighted averaging ( $\mathbf{l}_i$ ) with maximal variance ( $\mathbf{e}_i$ )
- Check both results



# Graphical representation

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

# Inertia statistics

- Use the `inertia.dudi` function to compute inertia statistics for rows and columns
- Represent absolute contributions using the `plot` function (see ?  
`plot.inertia`)