01 EDA

STA120: Introduction to Statistics Spring semester 2022 IMath, University of Zurich, Switzerland

Reinhard Furrer with contributions of various others

February 21, 2022

R-Code:

Loading the lemanHg dataset in R. (Lecture I)

```
Hg.frame <- read.csv('http://user.math.uzh.ch/furrer/download/intro2stat/lemanHg.csv')
#View(Hg.frame)
str( Hg.frame)      # dataframe with 1 numeric column and 293 observations
```

```
## 'data.frame':    293 obs. of  1 variable:
##  $ Hg: num  0.17 0.21 0.06 0.24 0.35 0.14 0.08 0.26 0.23 0.18 ...
```

```
head( Hg.frame, 3)  # column name is 'Hg'
```

```
##     Hg
## 1 0.17
## 2 0.21
## 3 0.06
```

```
Hg <- Hg.frame$Hg   # equivalent to 'Hg.frame[,1]' or `Hg.frame[,"Hg"]`
str( Hg) # now we have a vector
```

```
##  num [1:293] 0.17 0.21 0.06 0.24 0.35 0.14 0.08 0.26 0.23 0.18 ...
```

```
 is.na( Hg)
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
## [205] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [217] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [229] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [241] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [253] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [265] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [277] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [289] FALSE FALSE FALSE FALSE FALSE
```

```
which( is.na( Hg))    # check if there are NAs, alt: 'any( is.na( Hg))'
```

```
## integer(0)
```

```
# str( read.csv('data/lemanHg.csv', header=FALSE))
            # Wrong way to import data. Result is a 'factor' not 'numeric'!
```

R-Code (not in the script):

Working with a vector

```
my.data <- c(2, 3, 5, 7,1)
my.data[1]
```

```
## [1] 2
```

```
my.data[3]
```

```
## [1] 5
```

```
my.data[my.data > 4]
```

```
## [1] 5 7
```

R-Code:

Example of creating ordinal and interval scales in R.

(Lecture II)

```
ordinal <- factor( c("male","female"))
ordinal[1] == ordinal[2]
```

```
## [1] FALSE
```

```
#ordinal[1] > ordinal[2]  # warning ">" not meaningful for factors
interval <- c(2, 3)
interval[1] > interval[2]
```

```
## [1] FALSE
```

R-Code:

A quantitative EDA of the lemanHg dataset. (Lecture III)

```
mean( Hg)
```

```
## [1] 0.4617747
```

```
mean( Hg, trim=0.1)
```

```
## [1] 0.432383
```

```
median( Hg)
```

```
## [1] 0.4
```

```
c( mean=mean( Hg), tr.mean=mean( Hg, trim=0.1), median=median( Hg))
```

```
##      mean   tr.mean    median
## 0.4617747 0.4323830 0.4000000
```

```
c( var=var( Hg), sd=sd( Hg), iqr=IQR( Hg))   # capital letters for IQR!
```

```
##        var         sd        iqr
## 0.09014615 0.30024349 0.38000000
```

```
summary( Hg)              # min, max, quartiles and mean
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0100  0.2500  0.4000  0.4618  0.6300  1.7700
```

```
range( Hg)                # min, max, but not the difference
```

```
## [1] 0.01 1.77
```

```
tail( sort( Hg))          # sorts and then list the 6 largest values
```

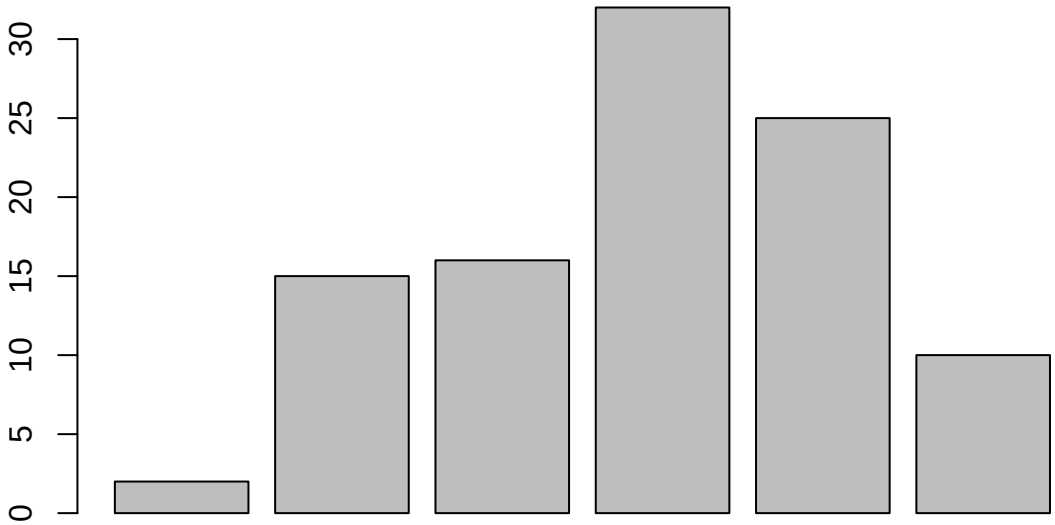
```
## [1] 1.28 1.29 1.30 1.31 1.47 1.77
```

```
## dat <- c(2, 15, 16, 32, 25, 10)   # see Figure 1.11
## emissionsource <- c('Air', 'Transp', 'Manufac', 'Electr', 'Deforest', 'Other')
## barplot( dat, names=emissionsource, ylab="Percent", las=2)
## barplot( cbind('2005'=dat), col=c(2,3,4,5,6,7), legend=emissionsource,
##       args.legend=list(bty='n'), ylab='Percent', xlim=c(0.2,4))
```

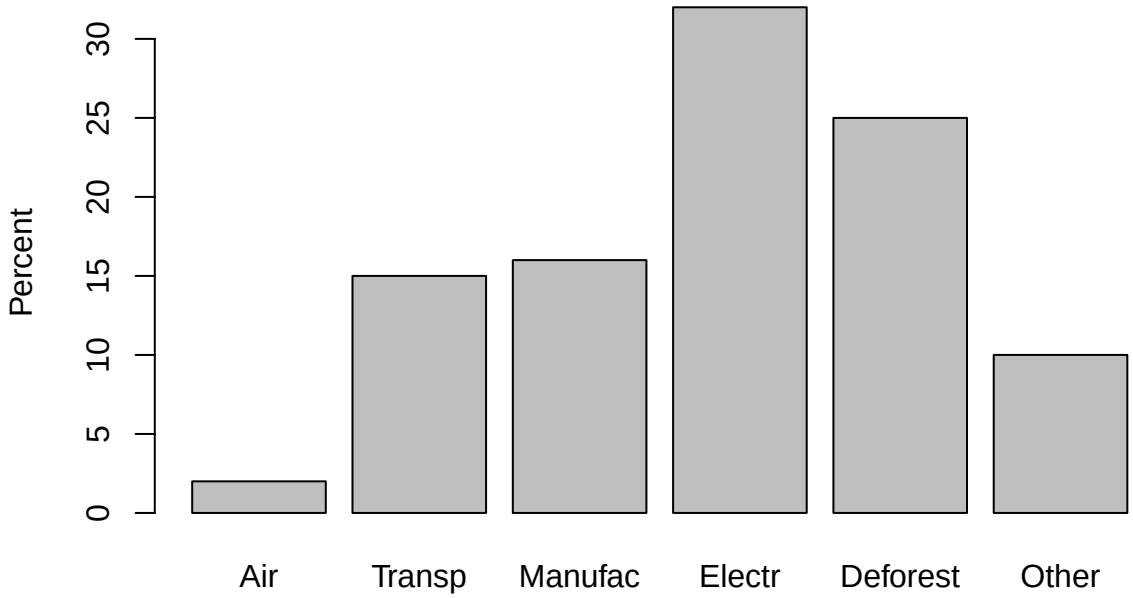
R-Code/Figure:

Bar plots: (Lecture IV)

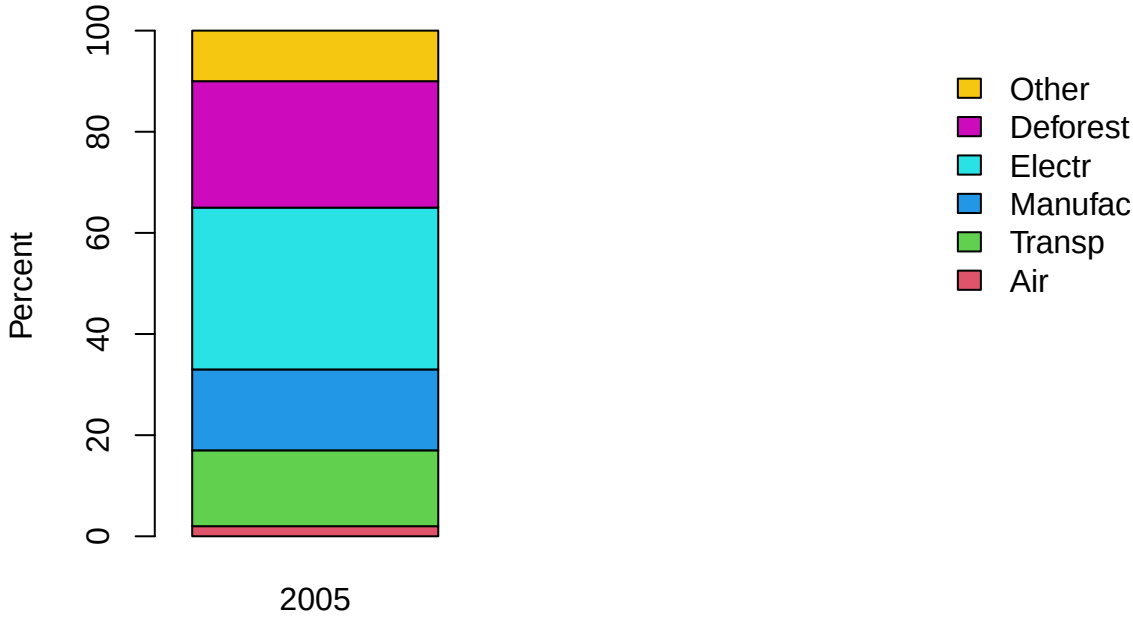
```
dat <- c(2, 15, 16, 32, 25, 10)   # see Figure 1.11
emissionsource <- c('Air', 'Transp', 'Manufac', 'Electr', 'Deforest', 'Other')
barplot(dat)
```



```
barplot( dat, names=emissionsource, ylab="Percent")
```




```
################## (not in the lecture)
barplot( cbind('2005'=dat), col=c(2,3,4,5,6,7), legend=emissionsource,
      args.legend=list(bty='n'), ylab='Percent', xlim=c(0.2,4))
```

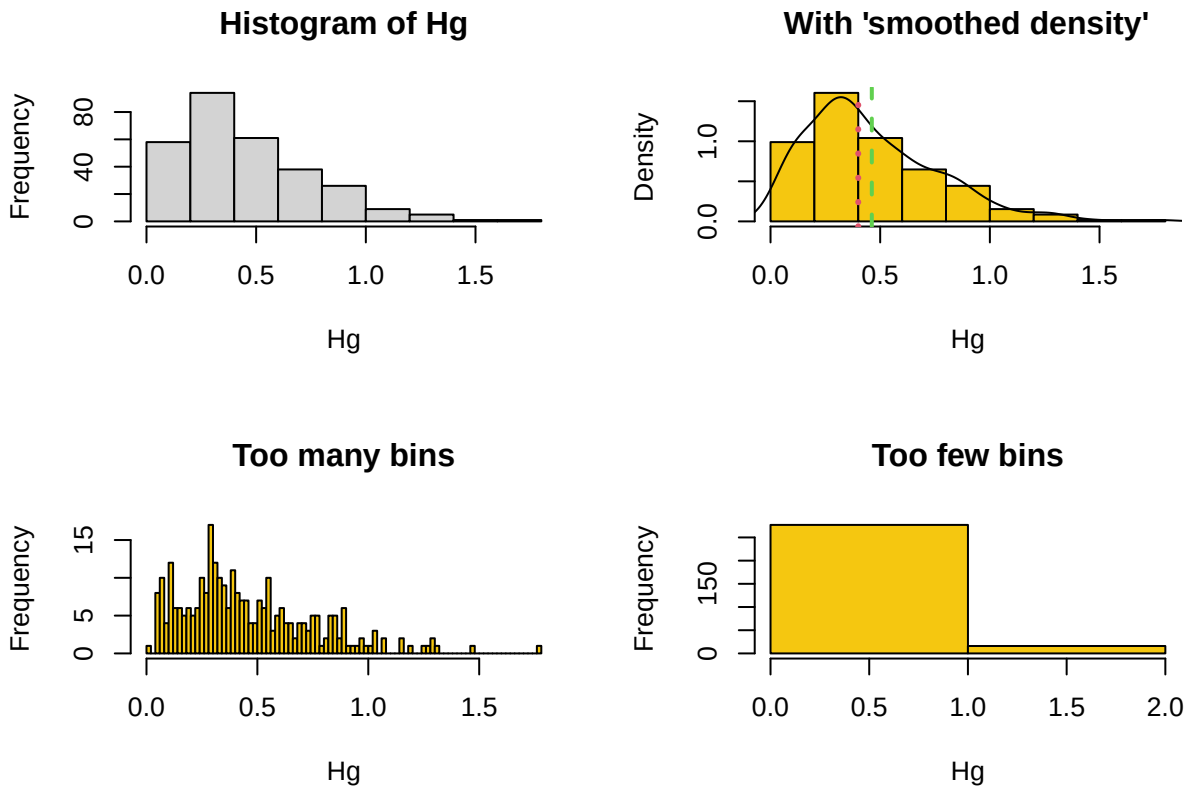


R-Code/Figure:

Histogram: (lecture V)

```
par( mfrow=c(2, 2))
histout <- hist( Hg)     # default histogram
hist( Hg, col=7, probability=TRUE, main="With 'smoothed density'")
lines( density( Hg))     # add smooth version of the histogram
abline( v=c(mean( Hg), median( Hg)), col=3:2, lty=2:3, lwd=2:3)
```

```
hist( Hg, col=7, breaks=90, main="Too many bins")
hist( Hg, col=7, breaks=2, main="Too few bins")
```




```
str( histout[1:3])      # Contains essentially all information of histogram
```

```
## List of 3
##  $ breaks : num [1:10] 0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6 1.8
##  $ counts : int [1:9] 58 94 61 38 26 9 5 1 1
##  $ density: num [1:9] 0.99 1.604 1.041 0.648 0.444 ...
```

R-Code/Figure:

Boxplots: (lecture VI)

```
par( mfcol=c(1, 3)) # to have three plots next to each other.
out <- boxplot( Hg, col="LightBlue", ylab="Hg")

out <- boxplot( Hg, col="LightBlue", ylab="Hg", outlty=1, outpch='')

  # 'out' contains numeric values of the boxplot
quantile( Hg, c(0.25, 0.75))  # compare with summary( Hg) and out["stats"]
```

```
##  25%  75%
## 0.25 0.63
```

```
IQR( Hg)                      # interquartile range
```

```
## [1] 0.38
```

```
quantile(Hg, 0.75) + 1.5 * IQR( Hg)           # upper boundary of the whisker
```

```
## 75%
## 1.2
```

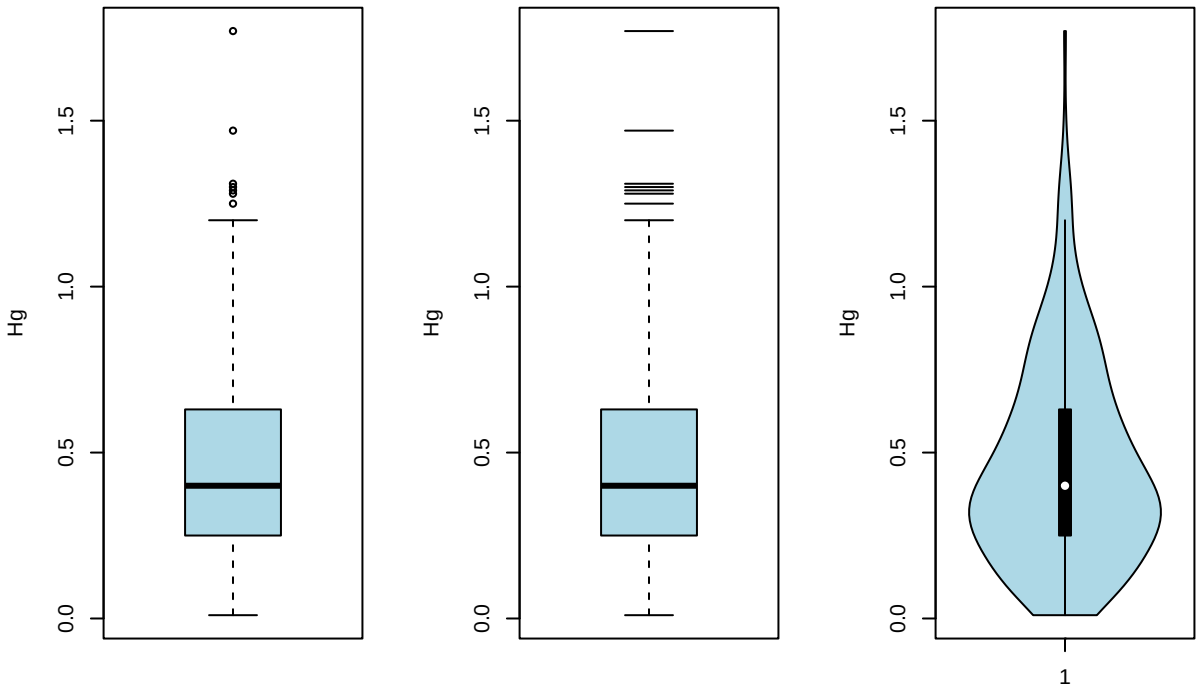
```
Hg[ quantile(Hg, 0.75) + 1.5 * IQR( Hg) < Hg] # points beyond the whisker
```

```
## [1] 1.25 1.30 1.47 1.31 1.28 1.29 1.77
```

```
require(vioplot)                          # R package providing violin plots
```

```
## Loading required package: vioplot
## Loading required package: sm
## Package 'sm', version 2.2-5.7: type help(sm) for summary information
## Loading required package: zoo
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

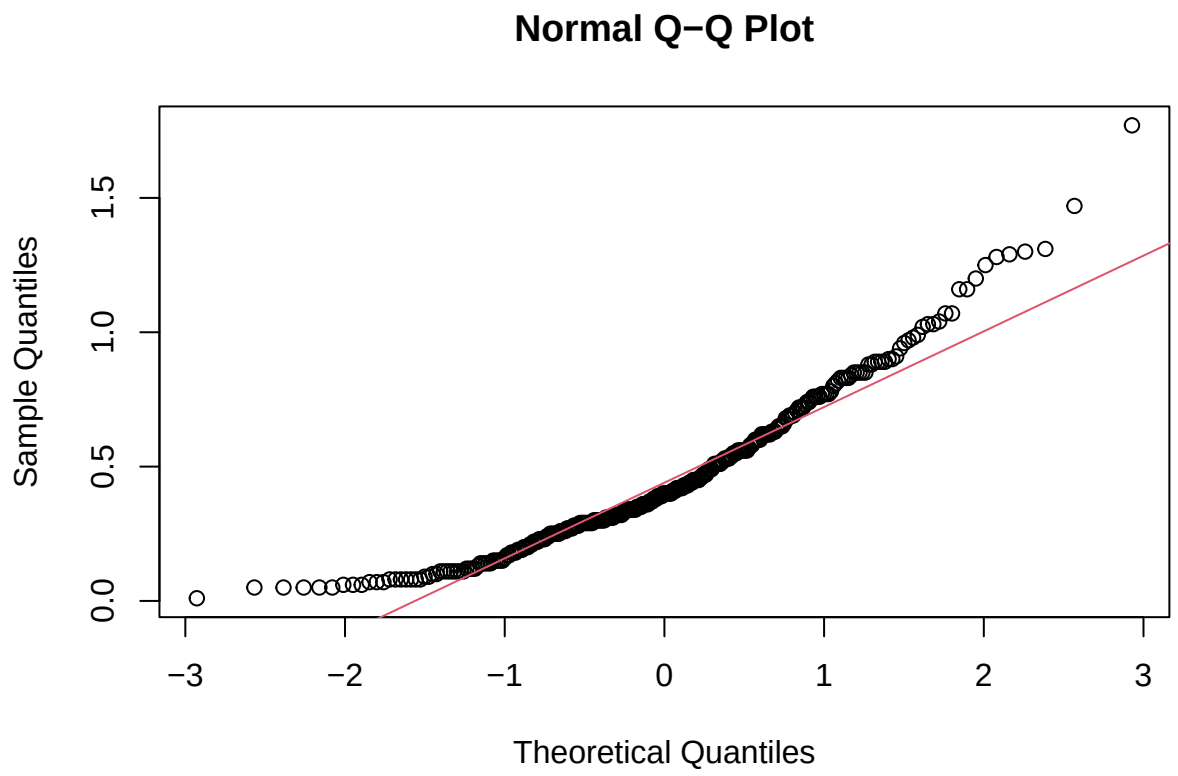
```
vioplot( Hg, col="Lightblue", ylab="Hg")
```



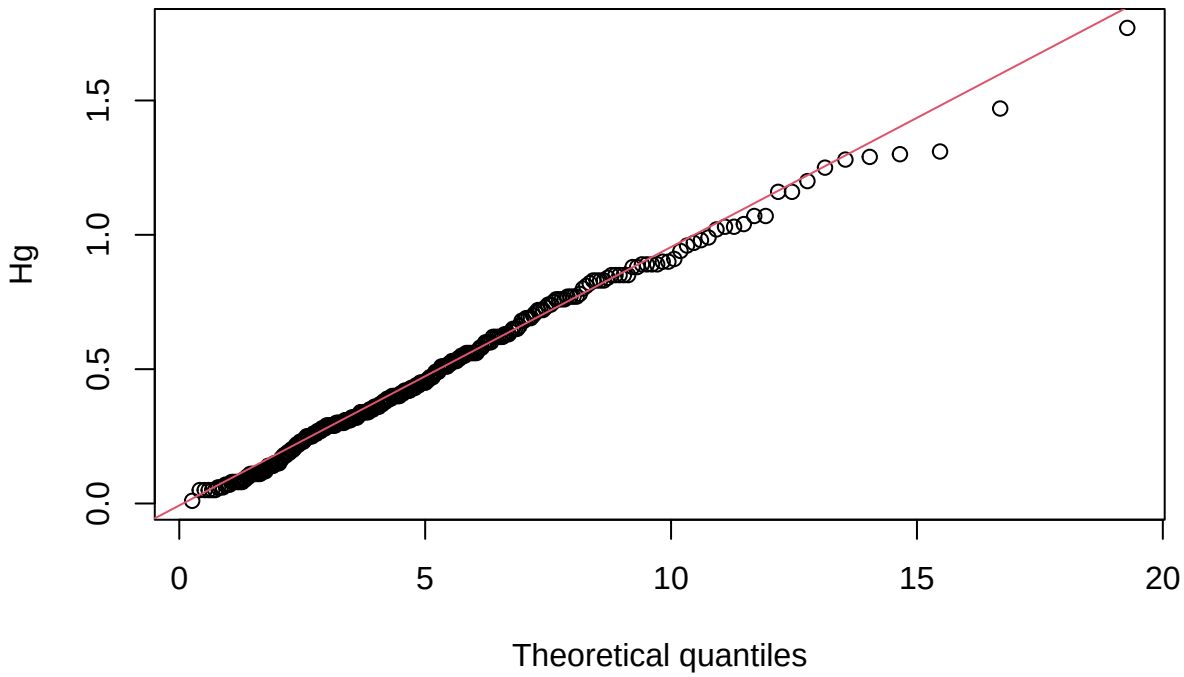
R-Code/Figure:

QQplots: (lecture VII)

```
qqnorm( Hg)             # QQplot with comparing with bell-shaped theoretical
qqline( Hg, col=2, main='')  # add read line
```



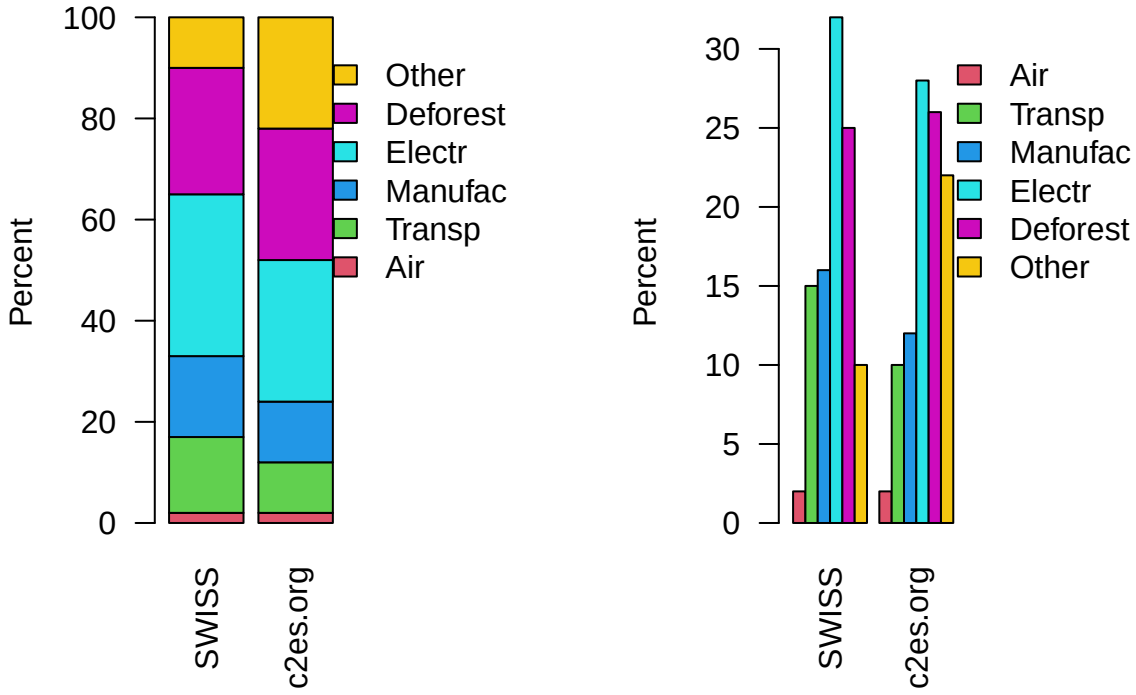

```
theoQuant <- qchisq( ppoints( 293), df=5) # minor mystery for the moment
# hist( theoQuant, prob=TRUE); lines(density(theoQuant)) # convince yourself
qqplot( theoQuant, Hg, xlab="Theoretical quantiles")
# For 'chisq' some a priori knowledge was used, for 'df=5' minimal
# trial and error was used.
qqline( Hg, distribution=function(p) qchisq( p, df=5), col=2)
```



R-Code/Figure:

Barplots:

```
par( mfcol=c(1,2))
dat2 <- c(2, 10, 12, 28, 26, 22) # source c2es.org
mat <- cbind( SWISS=dat, c2es.org=dat2)
barplot(mat, col=c(2,3,4,5,6,7), xlim=c(0.2,5), legend=emissionsource,
        args.legend=list(bty='n'), ylab='Percent', las=2)
barplot(mat, col=c(2,3,4,5,6,7), xlim=c(1,30), legend=emissionsource,
        args.legend=list(bty='n'), ylab='Percent', beside=TRUE, las=2)
```



R-Code (not in the script):

Scatter plot for penguins data set

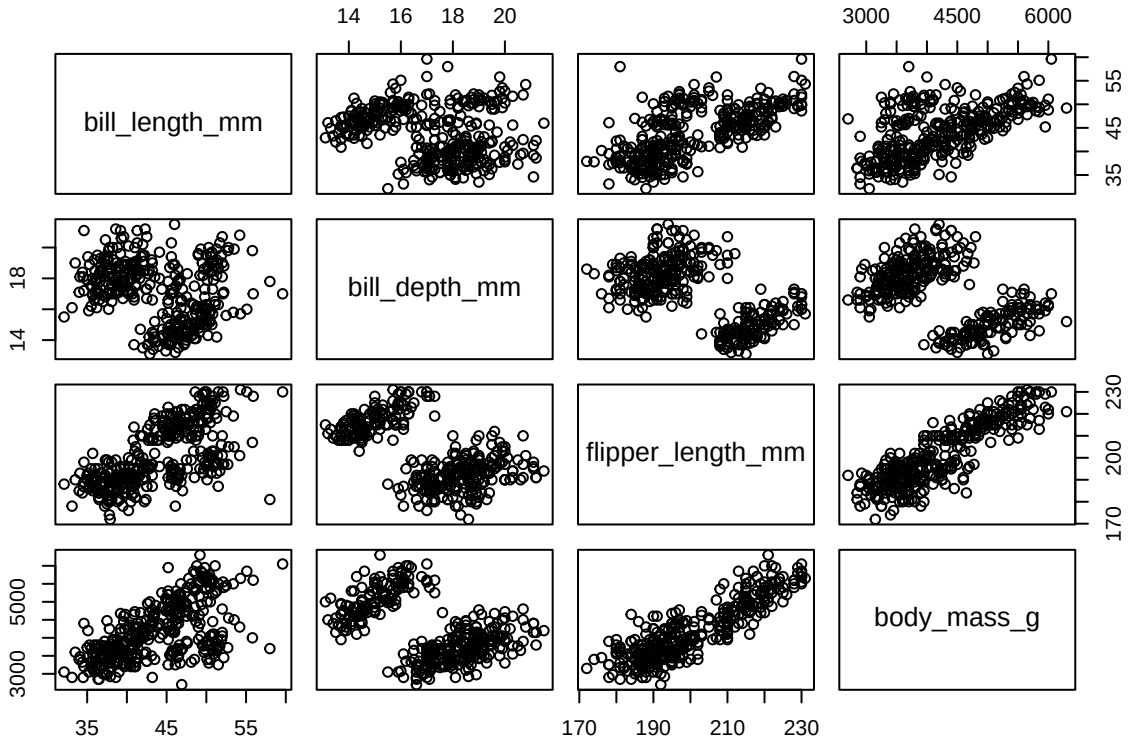
```
require(palmerpenguins)
```

```
## Loading required package: palmerpenguins
```

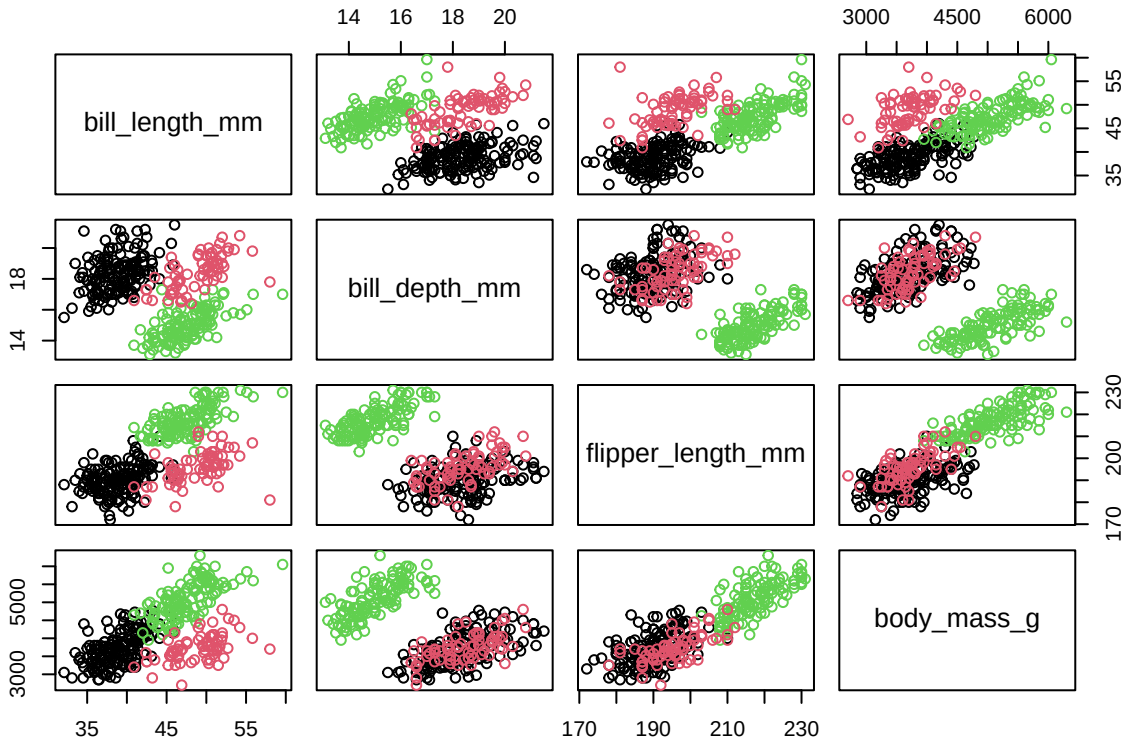
```
data("penguins") # the data is available from the package palmerpenguins
str(penguins)
```

```
## tibble [344 x 8] (S3: tbl_df/tbl/data.frame)
##  $ species          : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ island           : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ bill_length_mm   : num [1:344] 39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
##  $ bill_depth_mm    : num [1:344] 18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
##  $ flipper_length_mm: int [1:344] 181 186 195 NA 193 190 181 195 193 190 ...
##  $ body_mass_g      : int [1:344] 3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
##  $ sex              : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
##  $ year             : int [1:344] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

```
pairs( penguins[,c(3:6)])
```




```
pairs( penguins[,c(3:6)], col=penguins$species)
```



R-Code/Figure:

Histograms of the flipper length of the penguin dataset and further graphics

```
require(palmerpenguins)
penguins <- read.csv(       # 'path.package()' provides local location of pkg
    paste0( path.package('palmerpenguins'),'/extdata/penguins.csv'))
str(penguins, strict.width='cut')
```

```
## 'data.frame':    344 obs. of  8 variables:
##  $ species          : chr  "Adelie" "Adelie" "Adelie" "Adelie" ...
##  $ island           : chr  "Torgersen" "Torgersen" "Torgersen" "Torgersen" ...
##  $ bill_length_mm   : num  39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
##  $ bill_depth_mm    : num  18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
##  $ flipper_length_mm: int  181 186 195 NA 193 190 181 195 193 190 ...
##  $ body_mass_g      : int  3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
##  $ sex              : chr  "male" "female" "female" NA ...
##  $ year             : int  2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

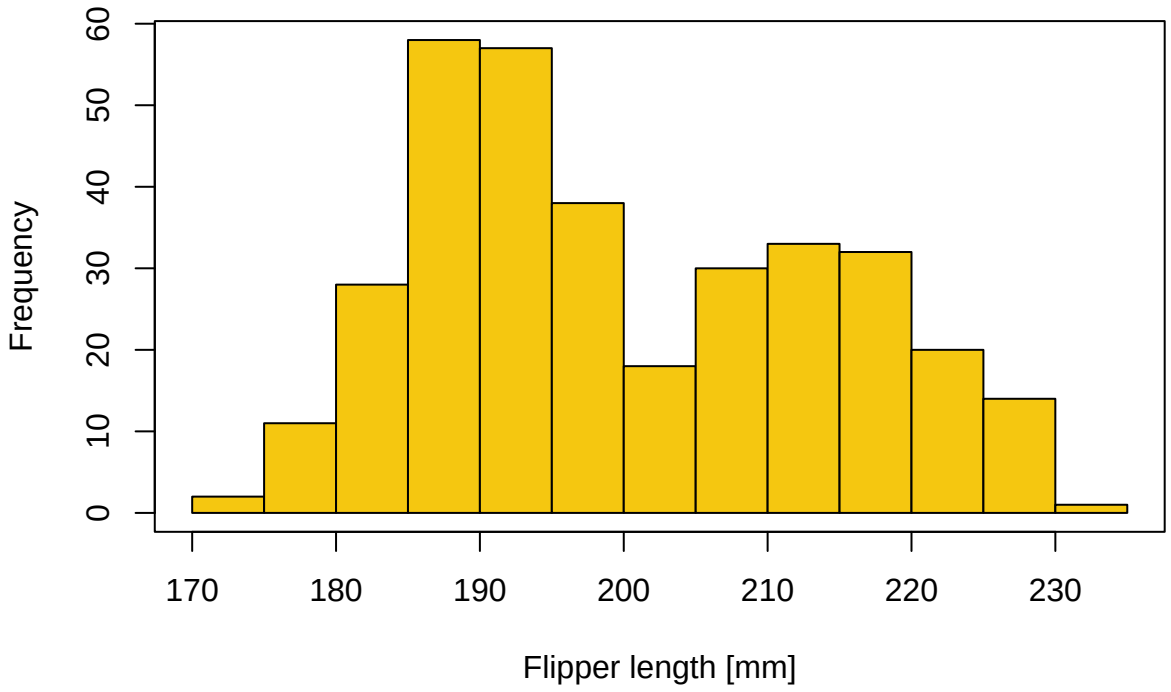
```
summary(penguins[, c(3:6)])  # others variables can be summarized by 'table()'
```

```
##  bill_length_mm  bill_depth_mm   flipper_length_mm  body_mass_g
##  Min.   :32.10   Min.   :13.10   Min.   :172.0     Min.   :2700
##  1st Qu.:39.23   1st Qu.:15.60   1st Qu.:190.0     1st Qu.:3550
##  Median :44.45   Median :17.30   Median :197.0     Median :4050
##  Mean   :43.92   Mean   :17.15   Mean   :200.9     Mean   :4202
##  3rd Qu.:48.50   3rd Qu.:18.70   3rd Qu.:213.0     3rd Qu.:4750
##  Max.   :59.60   Max.   :21.50   Max.   :231.0     Max.   :6300
##  NA's   :2       NA's   :2       NA's   :2         NA's   :2
```

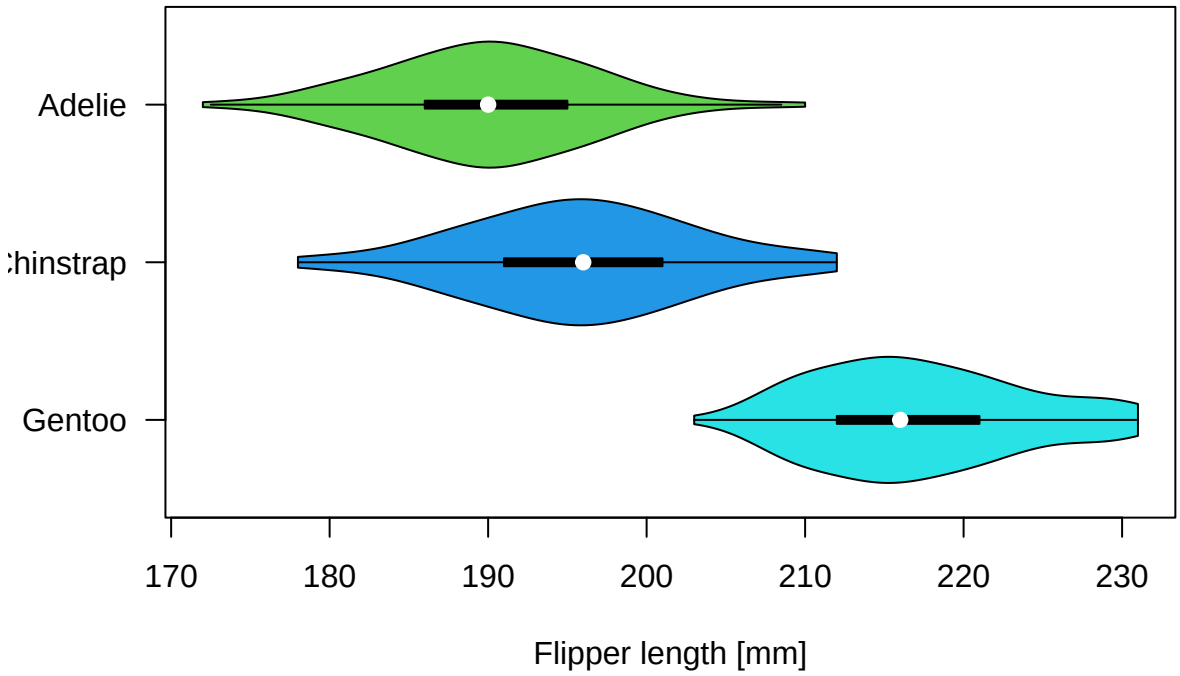
```
table(penguins[,c(2,1)])     # tabulating species on each island
```

```
##            species
## island      Adelie Chinstrap Gentoo
##   Biscoe        44         0    124
##   Dream         56        68      0
##   Torgersen     52         0      0
```

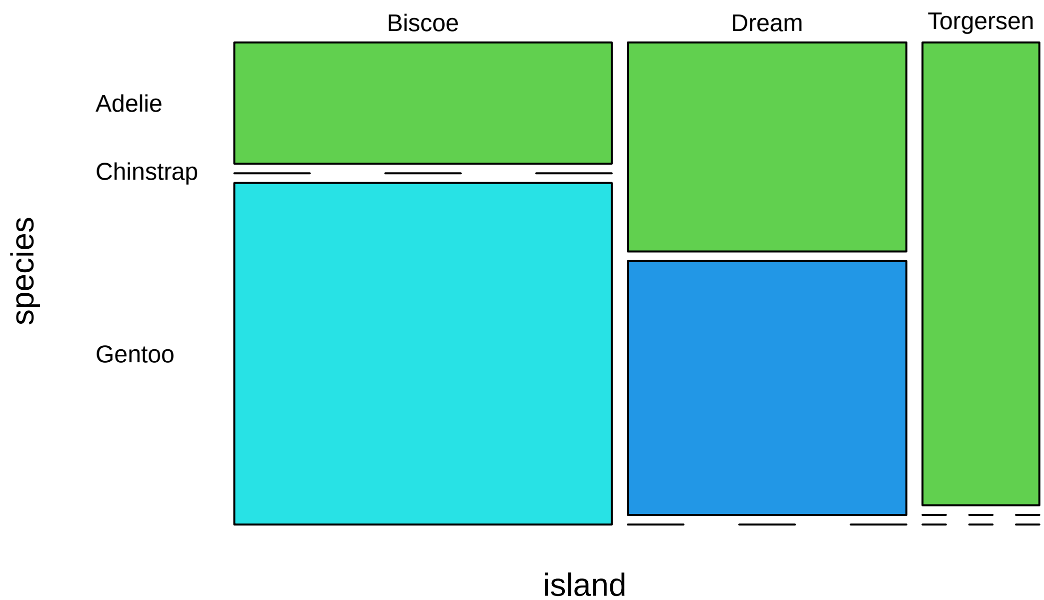
```
hist( penguins$flipper_length_mm, main='', xlab="Flipper length [mm]", col=7)
box()                        # rectangle around for similar view with others
```




```
with(penguins,  vioplot(flipper_length_mm[species=="Gentoo"],
      flipper_length_mm[species=="Chinstrap"], flipper_length_mm[
      species=="Adelie"],  names=c("Gentoo", "Chinstrap", "Adelie"),
      col=5:3, xlab="Flipper length [mm]", horizontal=TRUE, las=1))
```



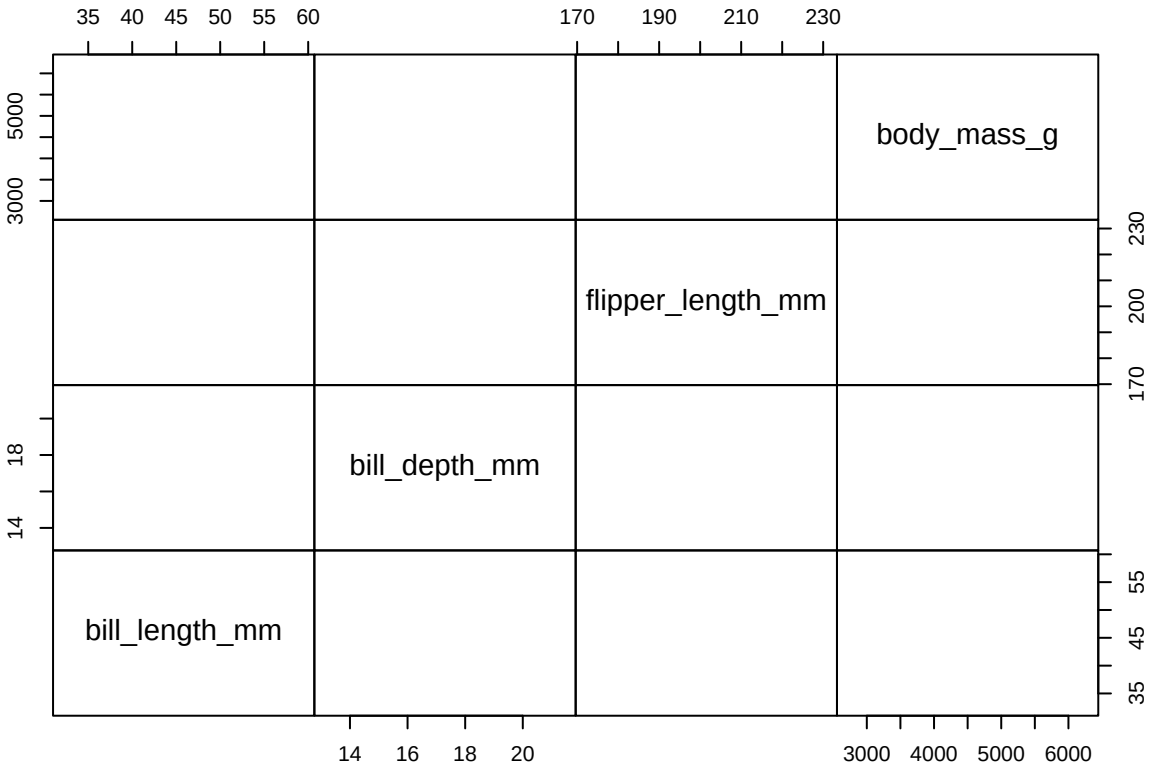

```
mosaicplot( table( penguins[,c(2,1)]), col=3:5, main='', cex.axis=.75, las=1)
```




```
upper.panel <- function( x,y, ...)   # see '?pairs' for a better and more
  points( x, y, col=as.numeric(penguins$species)+2,...) # complete way
lower.panel <- function( x,y, ...)
  points( x, y, col=as.numeric(penguins$sex))
pairs( penguins[,c(3:6)], gap=0, row1attop=FALSE,   # scatterplot
      lower.panel=lower.panel, upper.panel=upper.panel)
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
```

```
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion

## Warning in plot.xy(xy.coords(x, y), type = type, ...): NAs introduced by
## coercion
```




```
 #   pch=as.numeric(penguins$island) # would clutter and not be helpful
```

R-Code/Figure:

Parallel coordinate plot of the swiss dataset:

```
dim( swiss)   # in   package:datasets, available without the need to load.
```

```
## [1] 47  6
```

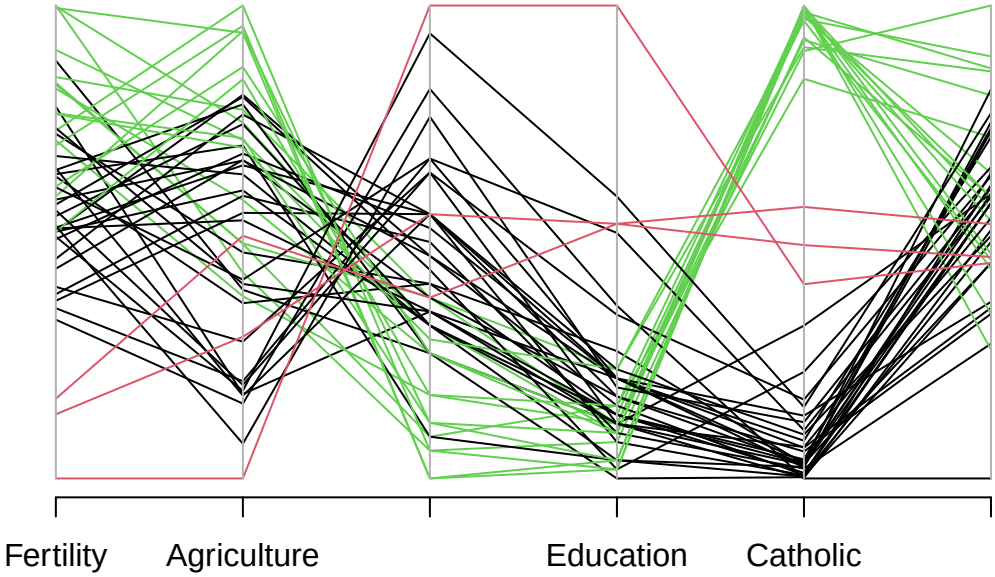
```
# str( swiss, strict.width='cut')    # or even:
# head( swiss);  summary( swiss)
require( MASS)    # package providing the function `parcoord()`
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:sm':
##
##     muscle
```

```
parcoord( swiss, col=2-(swiss$Catholic<40) + (swiss$Catholic>60))
```



Reinhard Furrer, Spring 2022