Applied Statistics and Data Analysis
Lab 2: Exploratory Data Analysis

Luca Grassetti and Paolo Vidoni
Department of Economics and Statistics, University of Udine

September, 2019

1 Tabular and graphical representations

1.1 Example: caffeine and marital status

The creation of a data set is often the first step in a statistical analysis. Here, using the available data on caffeine and marital status, we consider two alternative approaches for specifying a contingency table. A matrix can be directly defined by using the `matrix` function. The vector of the bivariate absolute frequencies is transformed in a matrix with 3 rows. The matrix is filled by rows. The row and column names are added and the names of the matrix dimensions are defined.

```
caff.marital <- matrix(c(652,1537,598,242,36,46,38,21,218,327,106,67),
                       nrow=3,byrow=T)
colnames(caff.marital) <- c("0","1-150","151-300",">300")
rownames(caff.marital) <- c("Married","Prev.married","Single")
names(dimnames(caff.marital)) <-
  c("Marital status","Caffeine consumption in mg/day")
```

The result of these command lines is as follows. It is sufficient to recall the object name to show it on the console window.

```
caff.marital

             Caffeine consumption in mg/day
Marital status   0 1-150 151-300 >300
  Married      652  1537     598  242
  Prev.married  36    46      38   21
  Single       218   327     106   67
```

Another possible solution is to generate a matrix containing the original data by using `rep`, `rbind` and `cbind` functions. A two column data set is saved in the `orig.data` object.

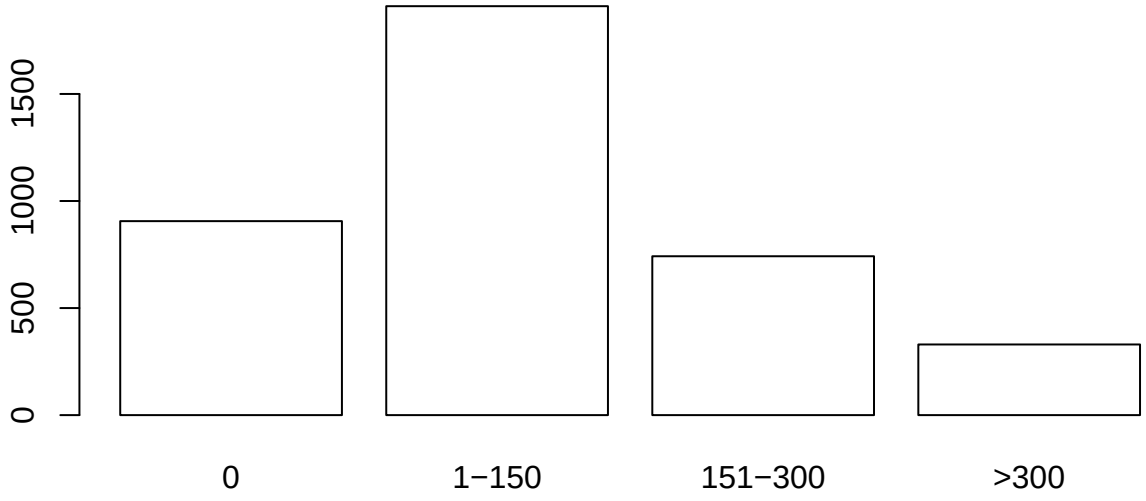
```
orig.data <- rbind(cbind(rep("0",652), rep("Married",652)),
cbind(rep("1-150",1537), rep("Married",1537)),
cbind(rep("151-300",598), rep("Married",598)),
cbind(rep("300+",242), rep("Married",242)),
cbind(rep("0",36), rep("Prev.married",36)),
cbind(rep("1-150",46), rep("Prev.married",46)),
cbind(rep("151-300",38), rep("Prev.married",38)),
cbind(rep("300+",21), rep("Prev.married",21)),
cbind(rep("0",218), rep("Single",218)),
cbind(rep("1-150",327), rep("Single",327)),
cbind(rep("151-300",106), rep("Single",106)),
cbind(rep("300+",67), rep("Single",67)))
orig.data <- as.data.frame(orig.data)
colnames(orig.data) <- c("marital","consumption")
str(orig.data)

'data.frame': 3888 obs. of  2 variables:
 $ marital    : Factor w/ 4 levels "0","1-150","151-300",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ consumption: Factor w/ 3 levels "Married","Prev.married",..: 1 1 1 1 1 1 1 1 1 1 ...
```

The bivariate frequencies distribution of the variables `marital` and `consumption` is then obtained by using the following command.

```
table(orig.data$consumption, orig.data$marital)


                  0 1-150 151-300 300+
  Married       652  1537     598  242
  Prev.married   36    46      38   21
  Single        218   327     106   67
```

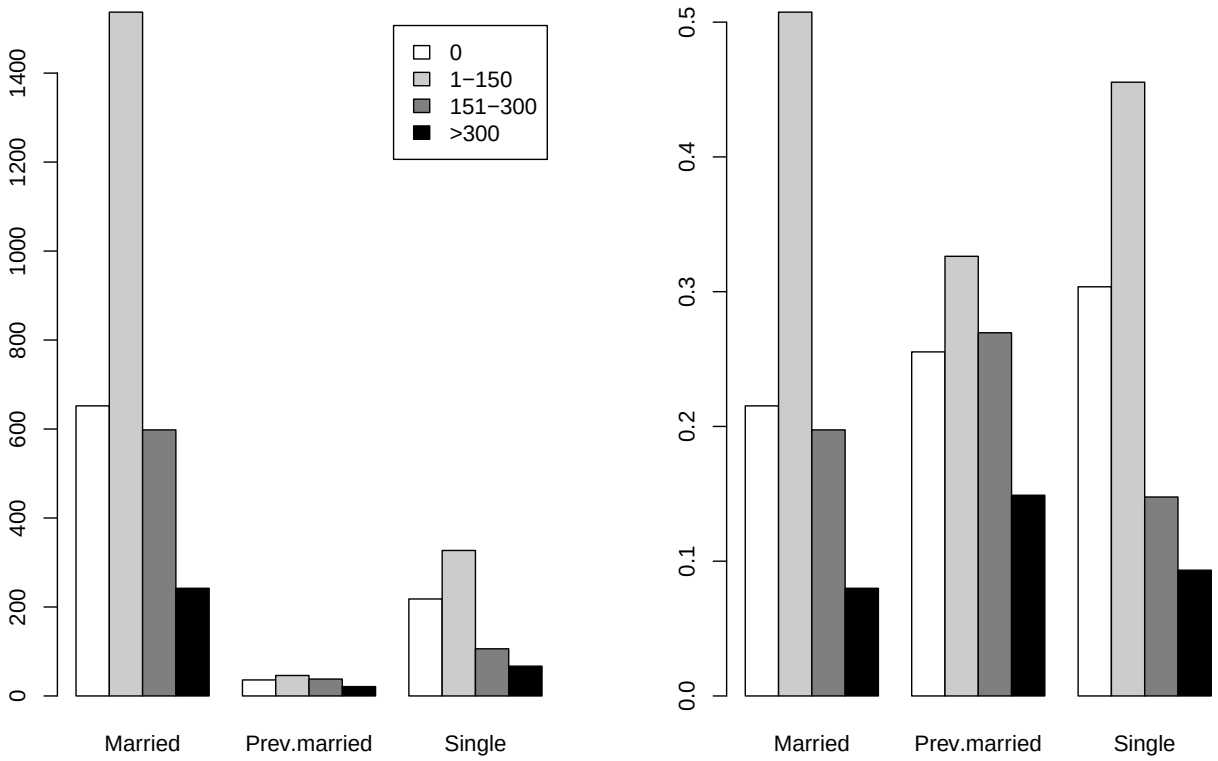
The graphical representation of the caffeine consumption can be obtained by using the `barplot` function. As in many graphical function the `col` argument can be specified in order to modify the color of the bars. In particular, `margin.table` command is used to compute the marginal frequency distribution of caffeine consumption (the argument 2 means that the columns of the matrix have to be summarized).

```
total.caff <- margin.table(caff.marital,2)
barplot(total.caff, col="white") # argument col
```



The relationship between the marital status and the caffeine consumption can be studied by means of the multiple barplots: observed frequencies (left), row percentages (right). In order to obtain this particular graphical representation, it is possible to use the `beside=T` (by default `beside` is set to `FALSE`). The first plot is also completed with the legend. The legend defines the color associated with the classes of consumption in the barplot representation. The last line of code restores the default graphical device.

```
par(mfrow=c(1,2))
barplot(t(caff.marital),beside=T,legend.text=colnames(caff.marital),
        col=c("white","grey80","grey50","black"))
barplot(prop.table(t(caff.marital),2),beside=T,
        col=c("white","grey80","grey50","black"))
```




```
par(mfrow=c(1,1))
```

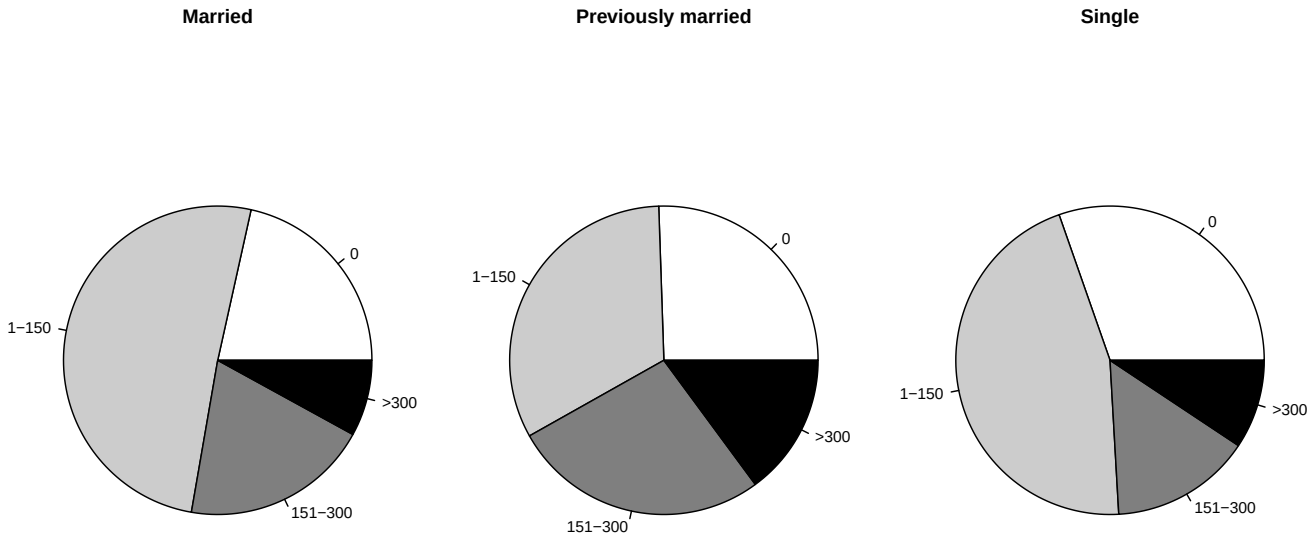
Since the aim of the analysis is to describe and compare the distribution of caffeine consumption in the three marital status, only the second plot is considered to obtain a reasonable comparison. The `col` option defines a vector of colors to be applied to the classes of data that must be represented.

Another possibility is to use the piecharts to represent the caffeine consumption. In particular the following lines of code are used to obtain the three piecharts of caffeine consumption according to marital status. The graphical device is divided into three columns and the plot parameters `mex` and `mar` are also changed. In particular,

- `mex` gives the character size expansion factor which is used to describe coordinates in the margins of plots;
- `mar` specifies the margins of the plots; the vector defines the `bottom`, `left`, `top` and `right` margins of each single plot (in number of text lines).

The colors of the slices are defined in the character vector `slices`. The main title of each single plot can be defined by using the `main="Title"` argument in `pie` function.

```
opar <- par(mfrow=c(1,3),mex=0.8, mar=c(1,1,2,1))
slices <- c("white","grey80","grey50","black")
pie(caff.marital["Married",], main="Married", col=slices)
pie(caff.marital["Prev.married",],
main="Previously married", col=slices)
pie(caff.marital["Single",], main="Single", col=slices)
```




```
par(opar)
```

The last line of the code sets back the graphical parameters to the default ones.

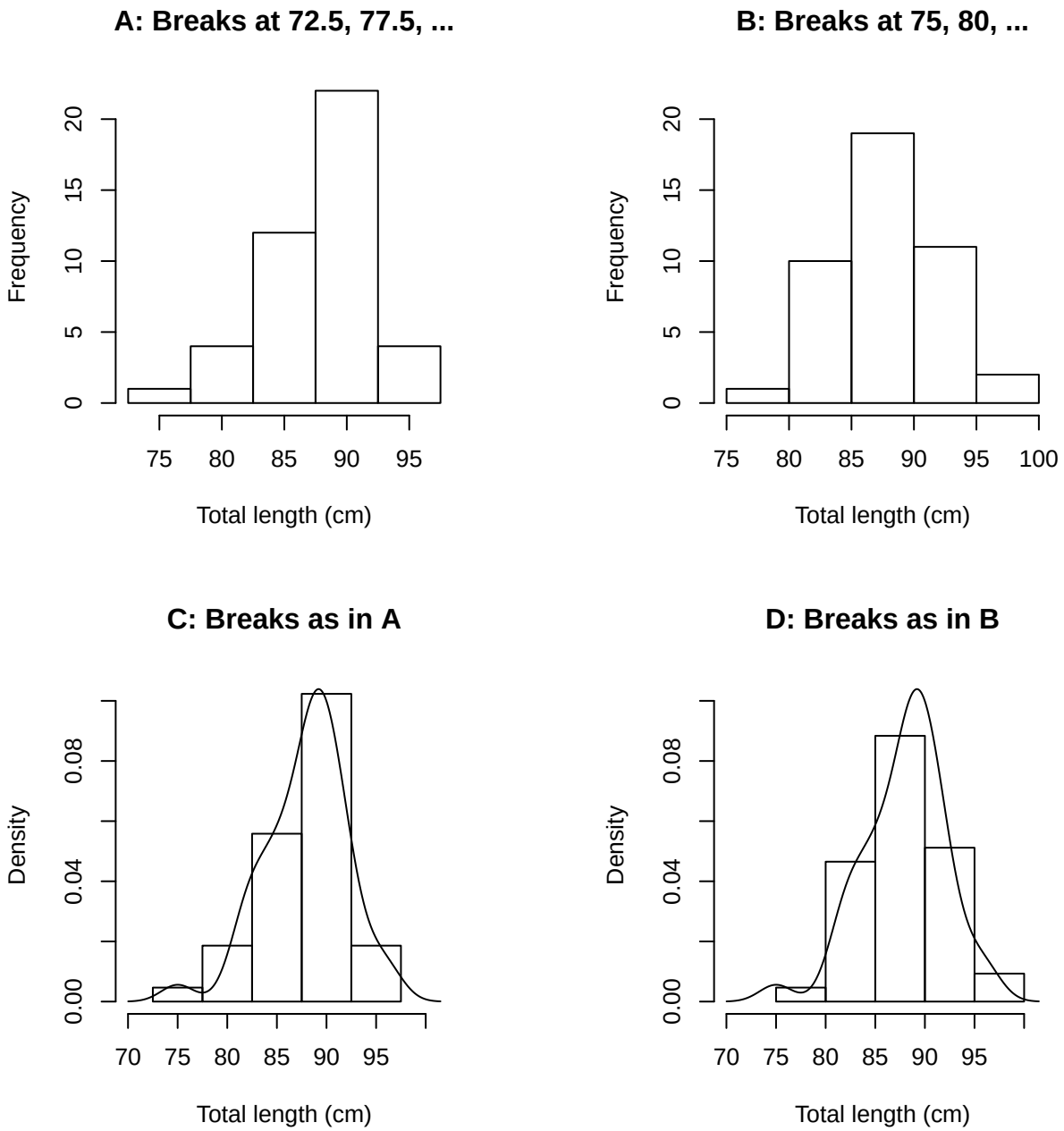
1.2 Example: possum data set

The data set consists of nine morphometric measurements on each of 104 mountain brushtail possums (Australian opossum), trapped at seven sites from Southern Victoria to central Queensland. The data set is included in the `DAAG` library. The first analysis regards the female possum total length. Steps in the analysis are the following:

- install the `DAAG` library and load it;
- load the specific `possum` data set and ask for a description;
- choose the total length variable from the data set and select the female subset only; to this end the `with` function is used and `sex=="f"` is the condition used to select from the `totlngth` variable;
- define the graphical device parameters;

- plot the histograms with different break points (in the `hist` function one can specify the `breaks` option);
- add to the histograms the density function estimate using the `lines` function applied to the `density` function result (which gives the kernel density estimation for numerical variables and it will be studied in more detail later on); in order to obtain overlapped plots it is important to specify the `probability` option as `TRUE`.

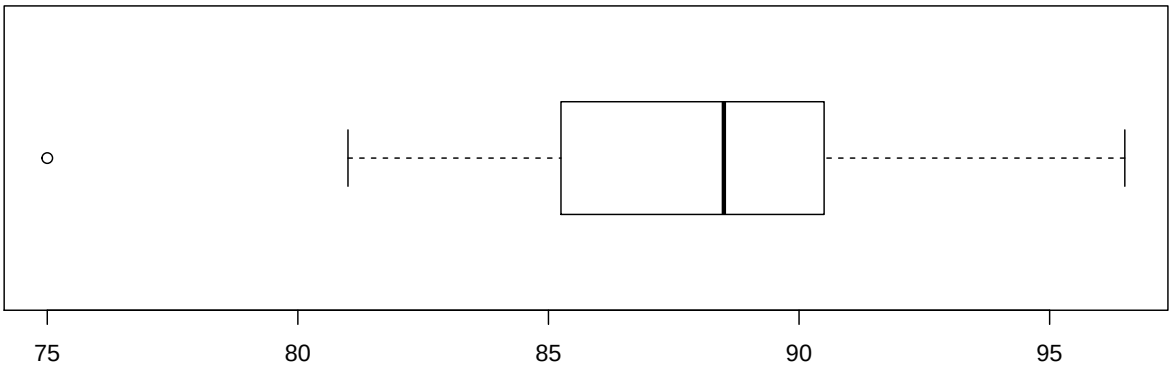
```
# install.packages("DAAG", repos="https://cran.rstudio.com/")
library(DAAG)

Loading required package: lattice
Warning: package 'lattice' was built under R version 3.4.4

data(possum)
# ?possum
ftotlngth <- with(possum, totlngth[sex=="f"])
opar <- par(mfrow = c(2,2), pty="s")
hist(ftotlngth, breaks = 72.5 + (0:5) * 5, ylim = c(0, 22),
     xlab="Total length (cm)", main ="A: Breaks at 72.5, 77.5, ...")
hist(ftotlngth, breaks = 75 + (0:5) * 5, ylim = c(0, 22),
     xlab="Total length (cm)", main="B: Breaks at 75, 80, ...")
dens <- density(ftotlngth)
xlim <- range(dens$x); ylim <- range(dens$y)
hist(ftotlngth, breaks = 72.5 + (0:5) * 5, probability = T,
     xlim = xlim, ylim = ylim, xlab="Total length (cm)",
     main ="C: Breaks as in A")
lines(dens)
hist(ftotlngth, breaks = 75 + (0:5) * 5, probability = T,
     xlim = xlim, ylim = ylim, xlab="Total length (cm)",
     main="D: Breaks as in B")
lines(dens)
```




```
par(opar)
par(mfrow=c(1,1))
```

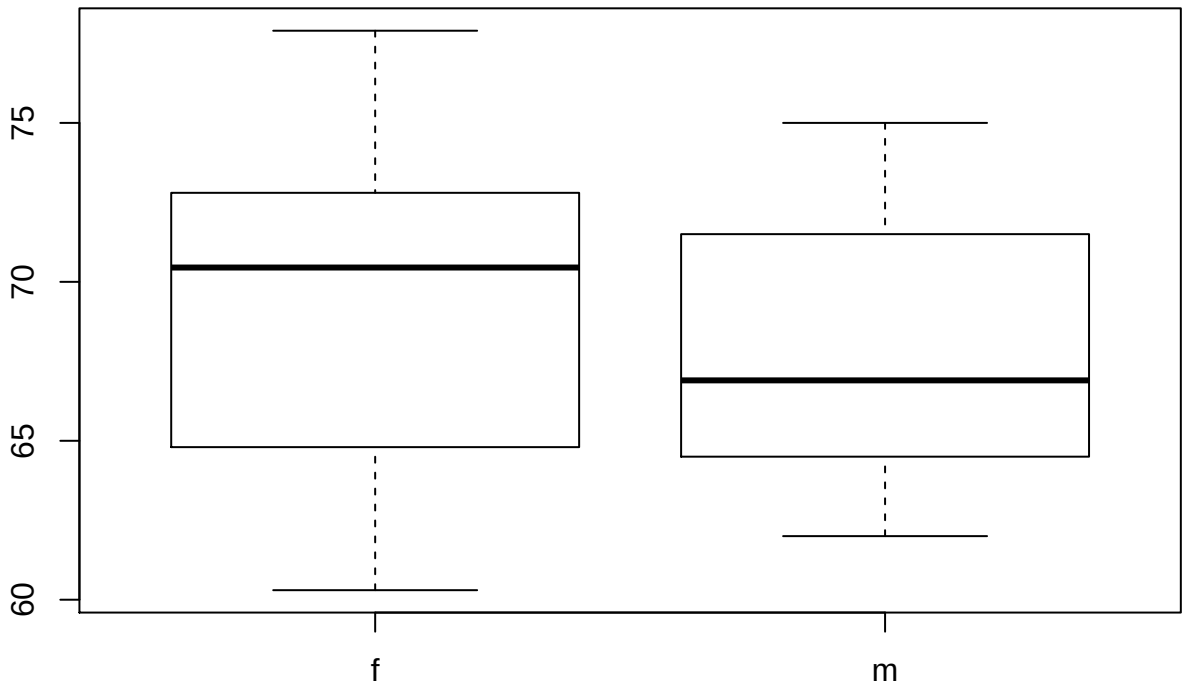
The `boxplot` function can also be applied to the total length measures to represent its empirical distribution. The `horizontal=T` option allows to plot the box horizontally.

```
boxplot(ftotlngth, horizontal=TRUE)
```



Boxplots are very useful to compare the conditional distributions of the numeric variables. Here the foot length (`footlgth`) is studied conditionally to the sex (`sex`).

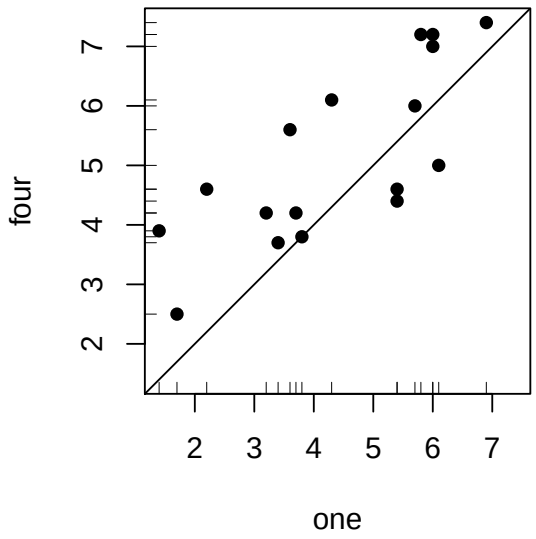
```
boxplot(possum$footlgth~possum$sex)
```



1.3 Example: the milk data set

The milk data set is part of the `DAAG` library. The aim of the analysis is to compare the sweetness of two different milk samples. The function `range` identifies the minimum and the maximum of a numeric vector. The graphical parameter `pty` defines the type of plot region (`"s"` means a squared region). In the plot function the argument `pch` specifies the symbol to be used in the plot (16 is the integer for filled circles) and `rug` is a function used to add the projections of the observations on the axes (the option `side` is used to specify the axis on which the projections have to be represented). A straight line is added to the plot by means of the `abline` function (0 and 1 are the intercept and the slope of the line, respectively).

```
xyrange <- range(milk)
par(pty="s")
plot(four ~ one, data = milk, xlim = xyrange, ylim = xyrange, pch = 16)
rug(milk$one)
rug(milk$four, side = 2)
abline(0, 1)
```



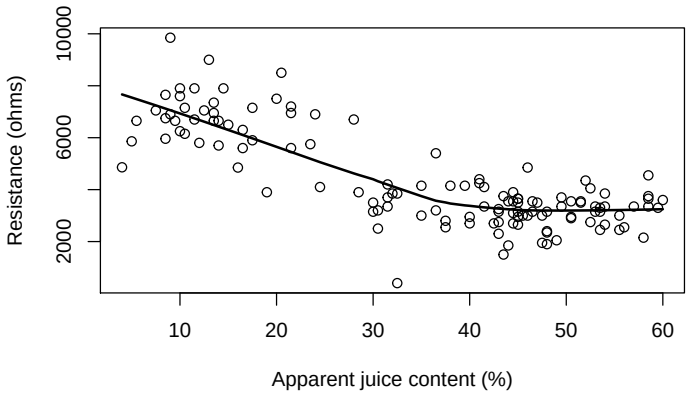

```
par(pty="m")
```

1.4 Example: electrical resistance of kiwifruit

The `fruitohms` data set, included in the `DAAG` library, is used to study the relationship between two variables: the electrical resistance (in ohms) and the apparent juice content (in percent) in some slabs of kiwifruit. The relationship between these two variables is negative, however it is not linear

and, consequently, its graphical representation can be described with the non-linear regression line obtained using the `lowess` function. The lowess smoother uses locally-weighted polynomial regression. The regression lowess is drawn on the scatterplot using the `lines` function. With the `with` command the functions are applied on the specified data set (it is not necessary to recall the name of the data set name). The option `lwd` of function `lines` defines the line width (the default is 1).

```
plot(ohms ~ juice, xlab="Apparent juice content (%)",
     ylab="Resistance (ohms)", data=fruitohms)
with(fruitohms, lines(lowess(juice, ohms), lwd=2))
```



1.5 Data transformation

Sometimes the measured phenomenon presents a peculiar dimension or the data variability is very large, due, for example, to the presence of outliers. Moreover, the frequency distribution could be very skewd (economic measures often present a positive skewed frequency distribution). In all these cases it is necessary to transform data before starting the statistical analysis.

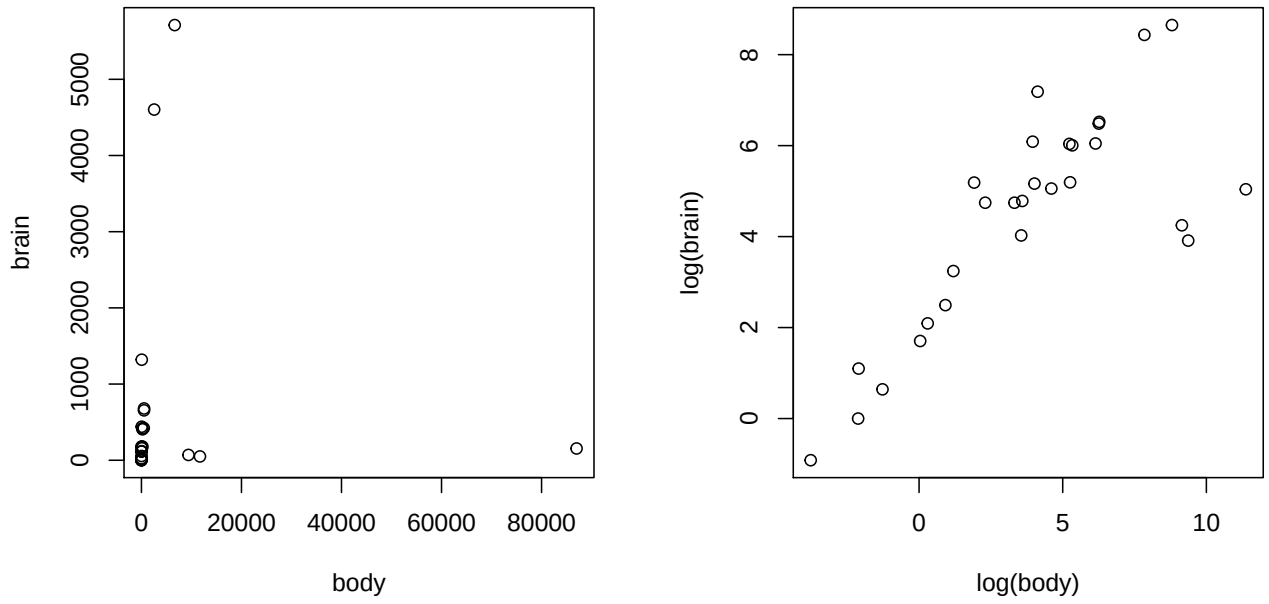
1.5.1 Untransformed scale vs logarithmic scale: the animals example

The following code is used to obtain the scatterplots describing the relationship between brain weight (g) and body weight (kg) for some animals. The data set is called `Animals` and it is included in the library `MASS`, which is a system library. Data are used in both the original scale and the logarithmic scale. The comparison of the results explains why data transformation can so important.

```
library(MASS)

Warning:  package 'MASS' was built under R version 3.4.4

Attaching package:  'MASS'
```

```
The following object is masked from 'package:DAAG':
    hills
```

```
oldpar <- par(mfrow = c(1,2), pty="s")
plot(brain ~ body, data=Animals)
plot(log(brain) ~ log(body), data=Animals)
```




```
par(oldpar)
```

Simple data transformations are commonly admitted in the main functions. The nested syntax allows to use data transformations without modifying the data itself.

1.5.2 Patterns in grouped data: the cuckoos example

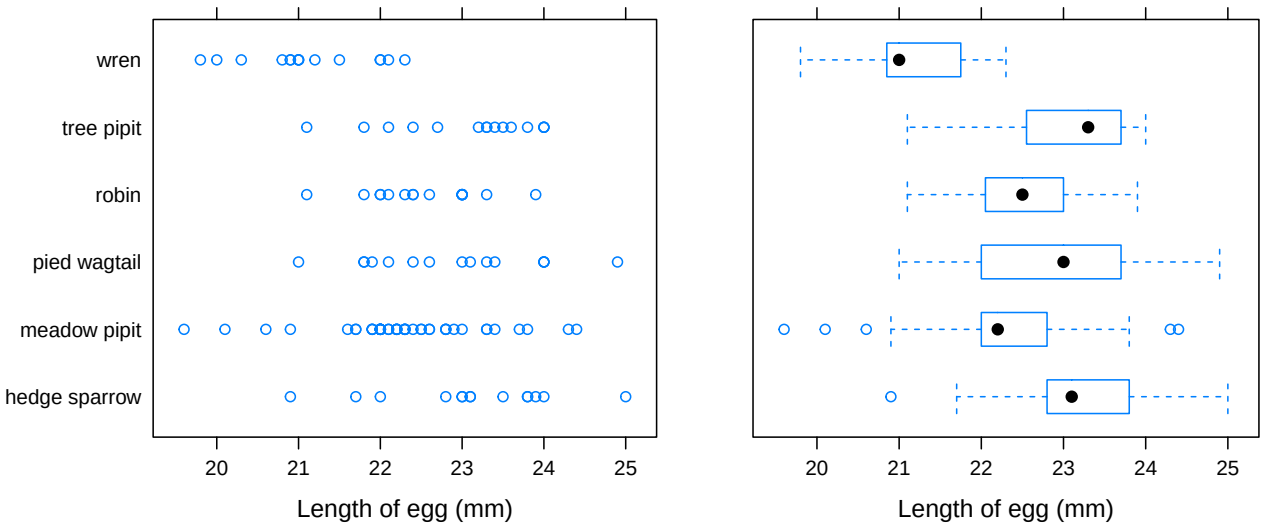
The graphical comparison between numerical observations coming from different groups (or different populations) can be obtained by using the `stripplot` or the `bwplot` functions of the package `lattice`. The second one usually gives the most informative representation. The example is based on the `cuckoos` data set of the `DAAG` library, which collects length and breadth measurements of 120 eggs found in the nests of six different species of host bird. The steps in the plot specification are the following:

- define the vector (`specnam`) with the names of observed species replacing the "." symbol with a space;
- specify the `plt1` object, which is the result of the `stripplot` function;

- print the obtained plot updating it with a new x-axis label (it is possible to define the `xlab` argument from the beginning in the `stripplot` function); the `position` argument is used to define the proportion of the graphical device that must be allocated to the plot `plt1`;
- specify the `plt2` object as the result of the `bwplot` function, where the `scales` argument determines how the x-axis and the y-axis are drawn; the option `list(y=list(alternating=0))` means that, for the y-axis, the tick labels must not be drawn;
- print the `plt2` plot in the graphical device using the position `c(0.55,0,1,1)`, which is specular to the previous one given by `c(0,0,0.55,1)`; the two plots are overlapped in the middle of the graphical device.

The below commented commands can be used to obtain a result that is very similar to the one represented in the following figure.

```
cuckoos$specnam <- with(cuckoos, sub(pattern=".", replacement=" ",
                                     species, fixed=TRUE))
plt1 <- stripplot(specnam ~ length, data=cuckoos)
#specnam <- with(cuckoos, sub(pattern=".", replacement=" ",
#       levels(species), fixed=TRUE))
#plt1 <- stripplot(species ~ length, factor.levels="specnam", data=cuckoos)
print(update(plt1, xlab="Length of egg (mm)"),position=c(0,0,0.55,1))
# xmin, ymin, xmax, ymax
plt2 <- bwplot(specnam ~ length, xlab="Length of egg (mm)",
               scales=list(y=list(alternating=0)), data=cuckoos)
print(plt2,newpage=FALSE, position=c(0.55,0,1,1))
```



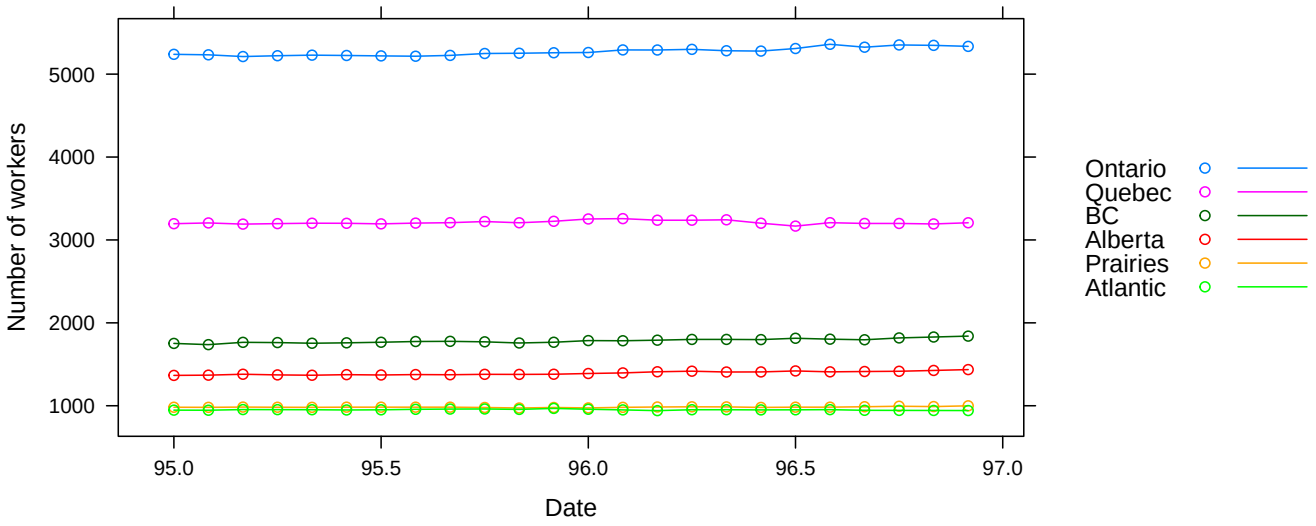
1.5.3 Comparing several time series: the labor force example

Another possible comparison may regard the behavior of different time series regarding different observational units (economic phenomena in different countries, different prices of different products

and so on). The `jobs` data set of the library `DAAG` collects the monthly observations of the number of workers in Canadian regions. The `xyplot` function of the package `lattice` is used here with a particular syntax where the observations related to the six regions are represented as a function of `Date`, which is a numeric variable. The specific arguments, used in the following lines of code, are:

- `outer=FALSE` which means that the time series have to be plotted on the same plot (alternatively, set `outer=T` and observe the result);
- `type="b"` which adds segments connecting the observed points;
- `auto.key=list(space="right",lines=TRUE)` which is used to automatically produce a suitable legend in conjunction with a grouping variable; the legend will be on the right of the plot (but the option can be changed in `left`, `top` or `bottom`) and it will include both symbols and lines, with the factor names.

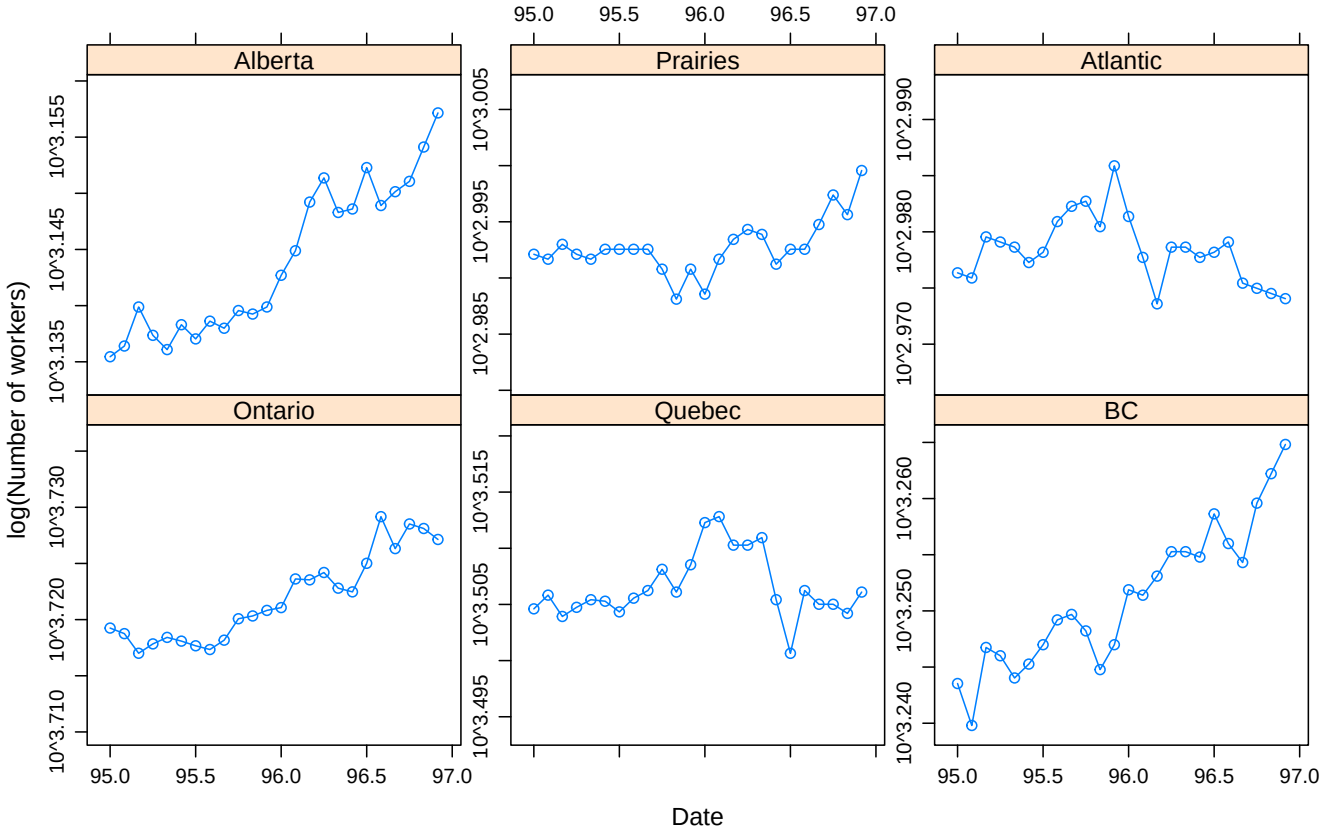
```
xyplot(Ontario+Quebec+BC+Alberta+Prairies+Atlantic ~ Date,
       outer=FALSE, data=jobs, type="b", ylab="Number of workers",
       auto.key=list(space="right", lines=TRUE))
```



As just introduced, a similar syntax is used to obtain separated plots. The additional arguments that are used here are the following:

- `layout=c(3,2)` which defines the layout of the plot in the graphical device (3 columns and 2 rows);
- `scales=list(y=list(relation="sliced", log=TRUE))` which constrains the length of the y-axis to be equal in the different plots and represents the data in the logarithmic scale; other possible values for the `relation` option are `free` and `same` (which is the default value).

```
xyplot(Ontario+Quebec+BC+Alberta+Prairies+Atlantic ~ Date,
       data=jobs, type="b", layout=c(3,2), ylab="log(Number of workers)",
       scales=list(y=list(relation="sliced", log=TRUE)),outer=TRUE)
```



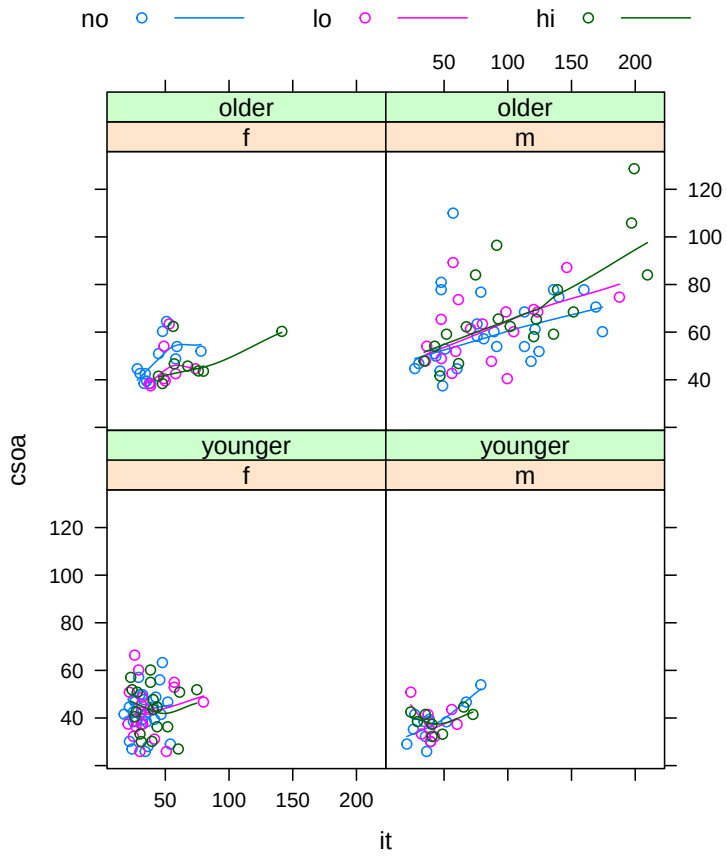
1.5.4 Comparing several scatterplots: the tinting example

Using once again a `DAAG` library data sets, we represent the joint behavior of two variables: the time (msec) `csoa` to recognize an alphanumeric target, on the y-axis, and the time (msec) `it` required to perform a simple discrimination task, on the x-axis. The gender `sex` and the age `agegp` are two levels factors and they are jointly considered in order to obtaining four different scatterplots. Moreover, the `tint` factor is used to distinguish the observations in the plots using three different colors. Thus, the arguments used are the following:

- `aspect=1` which gives the specific rectangular shape of the panel; the parameter represents the ratio between height and width, so that 1 means that a square panel will be plotted;
- `type=c("p","smooth")` which specifies the type of plot to be considered: `"p"` means points and `"shooth"` indicates a smooth regression line;
- `span=1.25` which defines the span of the smoothing function.

In the `update` function the `auto.key` argument is set to obtain the legend on the top (the default position), with a three column layout; more precisely, the option is `auto.key=list(columns=3)`.

```
tint.xyplot <-xyplot(csoa ~ it|sex*agegp, groups=tint,
       data=tinting, aspect=1,type=c("p","smooth"), span=1.25)
update(tint.xyplot, legend=NULL, auto.key=list(columns=3,
       points=TRUE, lines=TRUE))
```



2 Numerical summaries

In addition to the graphical description of an interesting phenomenon, it is useful to consider the computation of some simple descriptive statistical summaries. These are simple numerical summaries of the observed data that give a synthetic quantitative description of the phenomenon, focusing in particular on the location, the variability, the skewness and the kurtosis. Also the relationship between numerical variables can be studied as well as the association between qualitative factors.

2.1 Measures of location: the possum example

The main location indices can be easily obtained by considering the `summary` function (a general function which gives different results according to the particular object considered as argument).

```
summary(possum$footlgth)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  60.30   64.60   68.00   68.46   72.50   77.90       1
```

The output of this function includes

```
min(possum$footlgth, na.rm=T) # quantile(possum£footlgth,0, na.rm=T)

[1] 60.3

quantile(possum$footlgth,0.25, na.rm=T)

 25%
64.6

median(possum$footlgth, na.rm=T) # quantile(possum£footlgth,0.5, na.rm=T)

[1] 68

mean(possum$footlgth, na.rm=T)

[1] 68.45922

quantile(possum$footlgth,0.75, na.rm=T)

 75%
72.5

max(possum$footlgth, na.rm=T) # quantile(possum£footlgth,1, na.rm=T)

[1] 77.9
```

The number of possible missing values is also reported, if they are present. A very similar result, to that given by `summary`, can be obtained by using the `Summarize` function included in the `FSA` library.

2.2 Measures of variability: the possum example

The computation of variability indices in R is very simple and it can be obtained considering the following commands (for quantitative variables).

```
var(possum$footlgth, na.rm=T) # the corrected sample variance

[1] 19.31871
```

```
sd(possum$footlgth, na.rm=T)

[1] 4.395306

range(possum$footlgth, na.rm=T)

[1] 60.3 77.9

IQR(possum$footlgth, na.rm=T)

[1] 7.9

# the difference between the third and the first quantiles
```

The R commands apply the sample statistics to a given data set. In particular, the standard deviation is obtained using the following code.

```
cond <- !is.na(possum$footlgth)
numb <- length(possum$footlgth[cond])
sqrt(sum((possum$footlgth[cond]-sum(possum$footlgth[cond])/numb)^2)/(numb-1))

[1] 4.395306
```

Moreover, summary statistics can be computed using proper R commands. For instance, in order to compute the uncorrected variance, we can consider

- the adjusted variance

```
var(possum$footlgth, na.rm=T)*((numb-1)/numb)

[1] 19.13115
```

- the mean of the squared differences from the mean

```
mean((possum$footlgth - mean(possum$footlgth, na.rm=T))^2, na.rm=T)

[1] 19.13115
```

- the mean of the squared values minus the square of the mean

```
mean(possum$footlgth^2, na.rm=T) - mean(possum$footlgth, na.rm=T)^2

[1] 19.13115
```

The results are, obviously, the same.

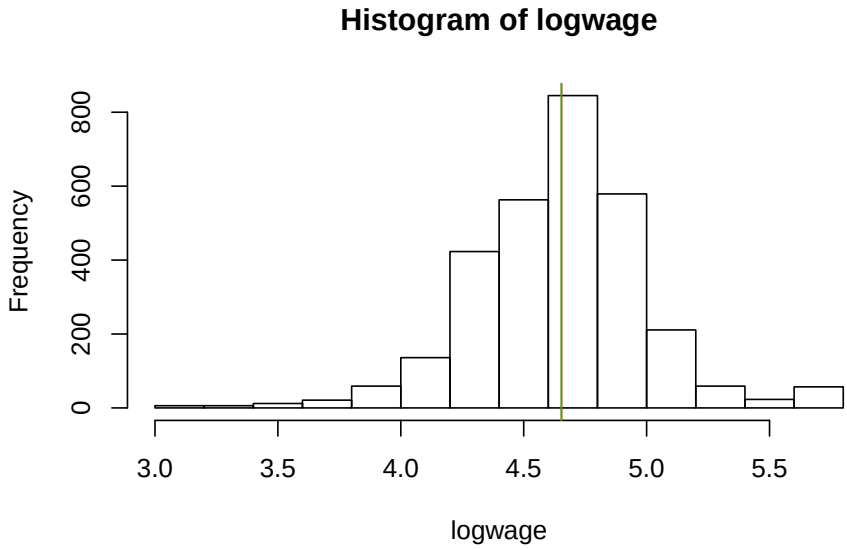
2.3 Measures of skewness and kurtosis: the wage example

In order to further analyze the frequency distribution of an interest phenomenon, it is possible to measure its potential symmetry and the eventual tail weight and peakedness. Thus, skewness and kurtosis indices can be used. The base R software does not include the two corresponding commands but they are included in the `moments` library.

```
# install.packages("moments", repos="https://cran.rstudio.com/")
library(moments)
# install.packages("ISLR", repos="https://cran.rstudio.com/")
library(ISLR)

Warning:  package 'ISLR' was built under R version 3.4.4
```

The graphical representation for the frequency distribution of the `wage` variable, defined within the `Wage` data set of the `ISLR` library, identifies the clear presence of skewness. The logarithmic transformation of the data reduces the skewness. The histograms for the `wage` variable, both in the original and in the logarithmic scale, can be obtained by using the `hist` function. The mean and the median are also represented with a red and a green vertical line, respectively, using the `abline` function. The two plots are included in the same picture. The graphical device considers two rows and one column, that is `par(mfrow=c(2,1))`.

```
attach(Wage)
par(mfrow=c(2,1))
hist(wage)
abline(v=mean(wage), col=2)
abline(v=median(wage), col=3)
hist(logwage)
abline(v=mean(logwage), col=2)
abline(v=median(logwage), col=3)
```



The values of the summary statistic confirm the graphical evidence of a positive asymmetry for the original data and of a slight negative asymmetry for the log data.

```
skewness(wage)

[1] 1.681489

skewness(logwage)

[1] -0.1235535
```

The kurtosis is not so simple to identify graphically but the observed values of the related summary statistic are greater than 3, which is the value identifying a normal-shaped distribution.

```
kurtosis(wage)

[1] 7.828952

kurtosis(logwage)

[1] 4.728038
```

3 Multivariate exploratory data analysis

We shall consider simple data analysis tools for investigating the relationships among two or more variables. The case of two variables is investigated. The conditional analysis can be useful for studying the way in which a factor affect the behavior of a quantitative variable. When the relationship between two quantitative variables is considered, we will adopt the so called correlation analysis. When the variables are both qualitative, the same kind of analysis is called association analysis. We present some simple examples of multivariate exploratory data analysis, focusing on the simple case with two interest variables.

3.1 Correlation analysis

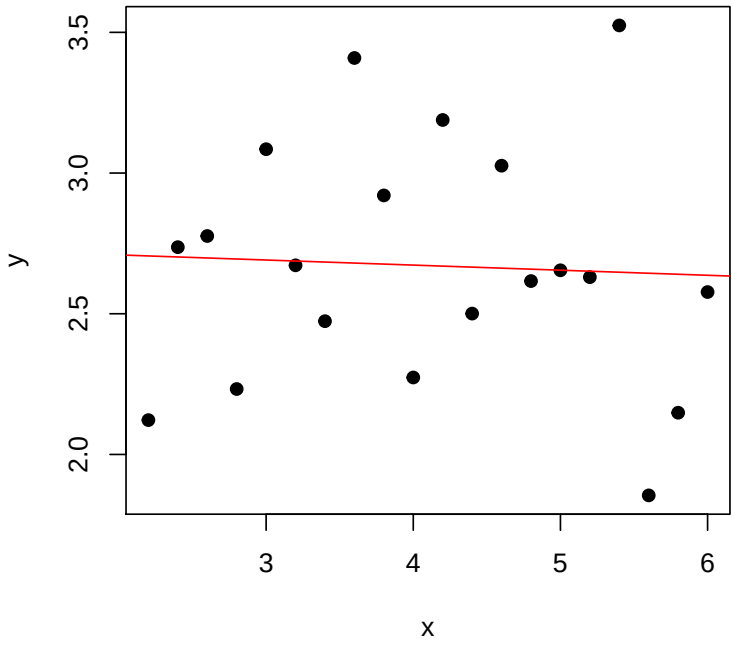
When considering two quantitative variables it is possible to study their relationship using both graphical representations and specific summary statistics. The examples are based on simulated data sets. In R it is very simple to obtain the sample covariance and the sample correlation index, using the functions `cov` and `cor`, respectively.
With regard to the following code lines,

- `set.seed(104)` defines the starting point for the data simulation procedure;
- `(11:30)/5` returns a vector `x` of integer values from 11 to 30, divided by 5;
- `y` is a real vector containing values which are obtained as a linear function of the corresponding `x` values with an additional error term; the magnitude of this error term determines the value of the correlation index and the strength of the linear relationship between `x` and `y`.

The following clusters of code lines are used to generate variables `x` and `y` that present weak, moderate and strong positive correlations, respectively. The data generation uses the same fixed component for each case, while the magnitude of the random error is progressively reduced. The scatterplots with the regression line are reported below.

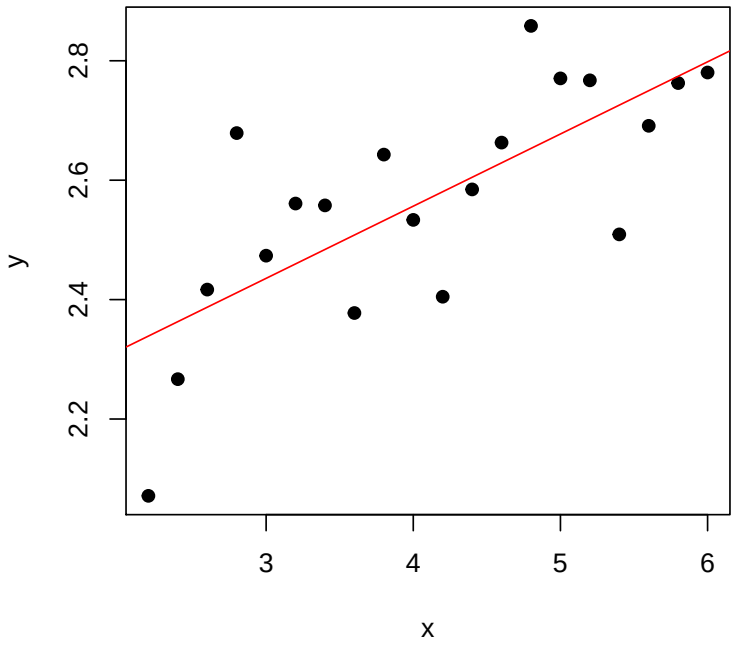
Weak correlation

```r
set.seed(104)
x <- (11:30)/5
y <- 2 + 0.15 * x + 0.6 * rnorm(20)
plot(x,y,pch=19)
cor(x,y)

[1] -0.04919971

abline(lm(y~x),col="red")
```



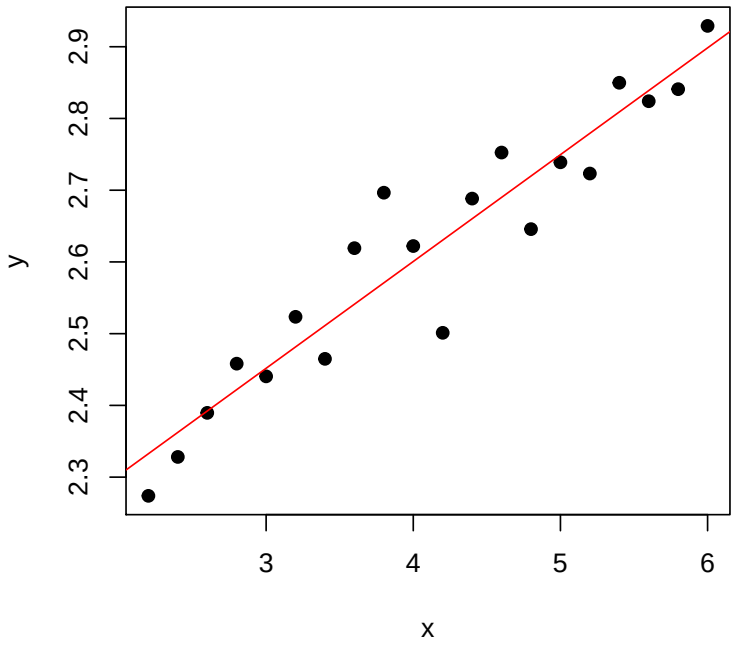
Moderate positive correlation

```r
set.seed(105)
x <- (11:30)/5
y <- 2 + 0.15 * x + 0.2 * rnorm(20)
plot(x,y,pch=19)
cor(x,y)

[1] 0.7310665

abline(lm(y~x),col="red")
```



Strong positive correlation

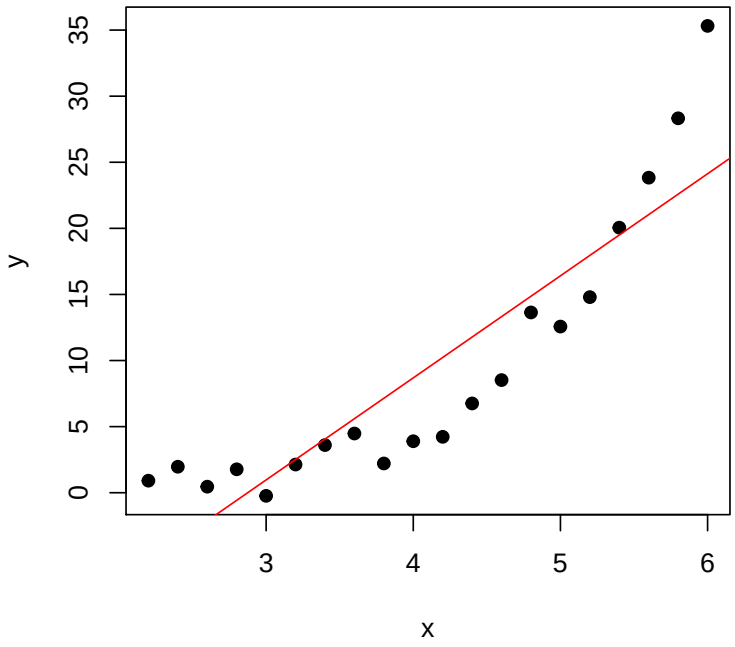
```
set.seed(106)
x <- (11:30)/5
y <- 2 + 0.15 * x + 0.06 * rnorm(20)
plot(x,y,pch=19)
cor(x,y)

[1] 0.9482955

abline(lm(y~x),col="red")
```



3.2 Non-linearity in correlation analysis

The following example is for emphasizing the difference between the Spearman and the Pearson correlation indices. Here the data is generated by considering a monotonic function of **x** as the deterministic part of the model.

```
set.seed(103)
x <- (11:30)/5
y <- 2 - 0.05 * x + 0.1 * ((x - 1.75))^4 + 1.25 * rnorm(20)
plot(x,y,pch=19)
abline(lm(y~x),col="red")
```



Then the two variables are positively related but the relationship is not linear and this is clearly represented in the plot. We shall consider the computation of the correlation index by means of the function `cor`. Both the simple linear correlation index (using the default option `method="pearson"`) and the rank correlation index (using the option `method="spearman"`) are taken into account.

```
cor(x,y,method="pearson")

[1] 0.8917358

cor(x,y,method="spearman")

[1] 0.9639098
```

The Spearman correlation index provides a more satisfactory description of the positive, non-linear relationship between `x` and `y`. There is a third option (`method="kendall"`) which gives a further non-parametric and robust measure of correlation based on the number of concordant and discordant pairs. It can be useful as-well for skewed bivariate data or in case of non-linear relationship.

3.3 The relationship between two qualitative variables

3.3.1 The hitters data set

Considering the `ISLR::Hitters` data set (namely, the data set `Hitters` of the library `ISLR`), it is possible to analyze the relationship between the salary class (a new variable that must be defined)

and the baseball league where the hitter plays. The following commands are specified in order to

- define the new variable `SalaryCL` (salary classes) by means of the `cut` function which arguments are the numerical variable and the edges of classes that we need to define;
- compute the contingency table using the function `table`;
- obtain, by means of function `chisq.test`, the association statistic χ^2, called the Pearson's Chi-squared statistic (values closed to zero describes a low association/dependence).

```
SalaryCl <- cut(ISLR::Hitters$Salary, c(0,200,500,700,2500))
contTable <- table(ISLR::Hitters$League, SalaryCl)
chisq.test(contTable)


Pearson's Chi-squared test

data:  contTable
X-squared = 0.87545, df = 3, p-value = 0.8313
```

The results of the Pearson's Chi-squared test are coherent with the default hypothesis that the salary (in its class version) is not associated (namely, it is independent) to the baseball league (the p-value is high). The notion of p-value, which is a suitable probability with values in $(0, 1)$, is a key concept in statistical inference and it will be reviewed late on.

3.3.2 The caffeine consumption example

In this case, the association measure χ^2 can be obtained by using the `chisq.test` function as well.

```
chisq.test(caff.marital)


Pearson's Chi-squared test

data:  caff.marital
X-squared = 51.656, df = 6, p-value = 2.187e-09
```

Thus, we are able to conclude that there is a significant relationship between caffeine consumption and marital status (the p-value is low).

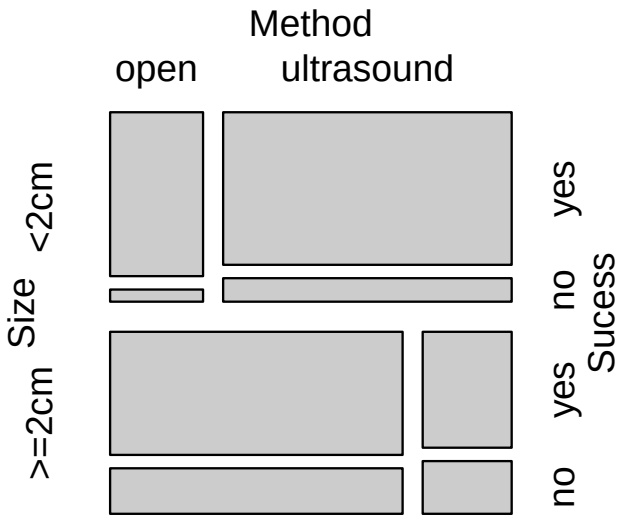
3.4 Multiple phenomena representation: some further examples

3.4.1 The mosaic plot

Using the library `vcd` it is possible to obtain the so called mosaic plot. In particular, the `mosaic` function can be used. First we install and load the library `vcd`. Then, a peculiar contingency table

is defined. The object created by the third line of the subsequent code is a three dimensional array (a generalization of a matrix in three dimensions) with 8 cells and size 2 for the first, the second and the third dimensions. The labels of the factors defining the three dimensions are also supplied in the array command line.
The `mosaic` function is applied to the result of the `aperm` function, which permits the permutation of the plotting order of the factors. The `labeling_args` argument is used to define the fontsize of the labels included in the plot. The mosaic plot is an area-proportional visualization of a higher-dimension frequency table.

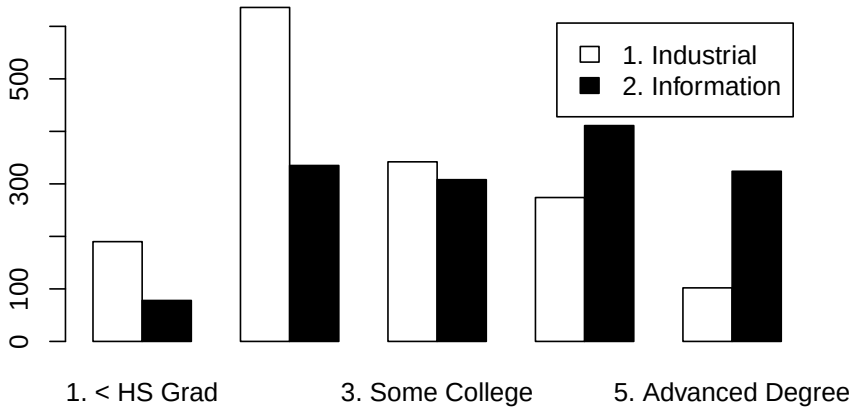
```
# install.packages("vcd", repos="https://cran.rstudio.com/")
library(vcd)

Loading required package: grid

Attaching package: 'vcd'
The following object is masked from 'package:ISLR':
    Hitters

stones <- array(c(81,6,234,36,192,71,55,25), dim=c(2,2,2),
                dimnames=list(Sucess=c("yes","no"),
Method=c("open","ultrasound"),Size=c("<2cm\n", ">=2cm\n")))
stones

, , Size = <2cm


      Method
Sucess open ultrasound
   yes   81        234
   no     6         36

, , Size = >=2cm


      Method
Sucess open ultrasound
   yes  192         55
   no    71         25

mosaic(aperm(stones, 3:1), main=NULL,
       labeling_args=list(gp_labels=gpar(fontsize=12),
                          gp_varnames=gpar(fontsize=12)),
       legend_args=list(fontsize=12))
```



3.4.2 The barplot

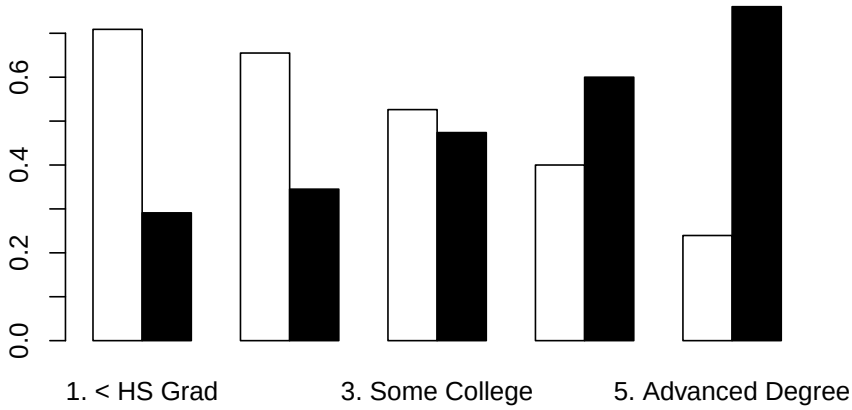
Working on the `Wage` data set of the library `ISLR` we can study the relationship between two of the variables involved: the education level `education` (a factor with five levels: `< HS Grad`, `HS Grad`, `Some College`, `College Grad` and `Advanced Degree`) and the job class `jobclass`, indicating the type of job (a factor with two levels: `Industrial` and `Information`). The following barplot describes the frequency distribution of `jobclass` conditionally to `education`.

```
attach(Wage)

The following objects are masked from Wage (pos = 5):
    age, education, health, health_ins, jobclass, logwage,
maritl, race, region, wage, year

par(mfrow=c(1,1))
table(education, jobclass) -> edu.jclass
barplot(t(edu.jclass),2,beside=T,legend.text=colnames(edu.jclass),
        col=c("white","black"))
```



It is clearly better to use the relative frequencies instead of the absolute ones. The subsequent plot shows an evident relationship between the two variables.

```
barplot(prop.table(t(edu.jclass),2),beside=T,col=c("white","black"))
```

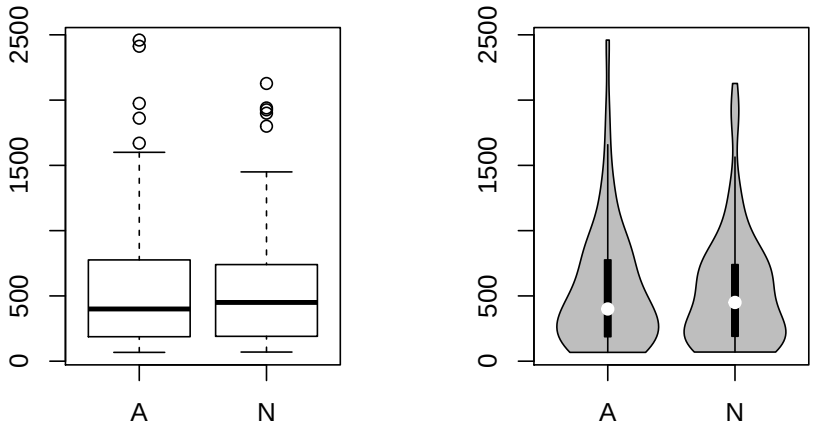


3.4.3 The violin plot

The violin plot is a valid alternative to the boxplot. With regard to the data set `ISLR::Hitters`, we shall investigate if there is a difference in salary level between the two baseball leagues. The data set `Hitters` is included in the `ISLR` library. A different data set, with the same name `Hitters`, is also included in the `vcd` library.

The use of violin plots allows to observe the frequency distribution of the salaries in a more detailed way. We consider function `vioplot` of the library `vioplot`.

```
# install.packages("vioplot", repos="https://cran.rstudio.com/")
library(vioplot)

Loading required package:  sm
Package 'sm', version 2.2-5.4:  type help(sm) for summary information

Attaching package:  'sm'
The following object is masked from 'package:MASS':
    muscle
The following object is masked from 'package:DAAG':
    pause

library(ISLR)
par(mfrow=c(1,2))
boxplot(ISLR::Hitters$Salary~ISLR::Hitters$League)
vioplot(ISLR::Hitters$Salary[ISLR::Hitters$League=="A"],
        ISLR::Hitters$Salary[ISLR::Hitters$League=="N"])

Error in quantile.default(data, 0.25):  missing values and NaN's not allowed if 'na.rm'
is FALSE

# there is an issue with NAs
cond <- !is.na(ISLR::Hitters$Salary)
vioplot(ISLR::Hitters$Salary[cond][ISLR::Hitters$League[cond]=="A"],
        ISLR::Hitters$Salary[cond][ISLR::Hitters$League[cond]=="N"],
        names=c("A","N"), col="grey")
```



3.4.4 Conditional measures of location and variability

Using the `ISLR::Hitters` data set we can also consider the following conditional descriptive summaries. Adopting the `tapply` function one can apply a specific R function to subsets of values defined by the factor included as second argument in the function call. Here the functions to be applied are `summary`, `var` and `sd`.

```
tapply(ISLR::Hitters$Salary, ISLR::Hitters$League, summary)

$A
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   67.5   187.5   400.0   542.0   775.8  2460.0      36

$N
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   70.0   190.8   450.0   529.1   740.0  2127.3      23

tapply(ISLR::Hitters$Salary[cond], ISLR::Hitters$League[cond], var)

     A      N
216023 191033

tapply(ISLR::Hitters$Salary[cond], ISLR::Hitters$League[cond], sd)

       A        N
464.7828 437.0732
```