Medical Statistics with R

Dr. Gulser Caliskan
Prof. Giuseppe Verlato

Unit of Epidemiology and Medical Statistics
Department of Diagnostics and Public Health
University of Verona, Italy

LESSON 2 INDEX

Descriptive statistics are used to describe the basic features of the data in a study. They provide simple summaries about the sample and the measures. Together with simple graphics analysis, they form the basis of virtually every quantitative analysis of data.

In quantitative research, after collecting data, the first step of statistical analysis is to describe characteristics of the responses, such as the average of one variable (e.g., age), or the relation between two variables (e.g., age and creativity).

The next step is inferential statistics, which help you decide whether your data confirms or refutes your hypothesis and whether it is generalizable to a larger population.

Types Of Descriptive Statistics

There are **3 main types** of descriptive statistics:

- The distribution concerns the frequency of each value.
- The central tendency concerns the averages of the values.
- The variability or dispersion concerns how spread out the values are.

The distribution is a summary of the frequency of individual values or ranges of values for a variable. The simplest distribution would list every value of a variable and the number of persons who had each value.

For instance, a typical way to describe the distribution of college students is by year in college, listing the number or percent of students at each of the four years.

Or, we describe gender by listing the number or percent of males and females.

In these cases, the variable has few enough values that we can list each one and summarize how many sample cases had the value.

One of the most common ways to describe a single variable is with a frequency distribution. Depending on the particular variable, all of the data values may be represented, or you may group the values into categories first (e.g., with age, price, or temperature variables, it would usually not be sensible to determine the frequencies for each value.

Rather, the value are grouped into ranges and the frequencies determined.).

Frequency distributions can be depicted in two ways, as a table or as a graph. The table above shows an age frequency distribution with five categories of age ranges defined.

The same frequency distribution can be depicted in a graph. This type of graph is often referred to as a histogram or bar chart.

Measures Of Central Tendency

The central tendency of a distribution is an estimate of the "center" of a distribution of values. There are three major types of estimates of central tendency:

- **Mean**
- **Median**
- **Mode**

The Mean is probably the most commonly used method of describing central tendency. To compute the mean all you do is add up all the values and divide by the number of values.

Population Mean	Sample Mean
$\mu = \frac{\sum_{i=1}^{N} x_i}{N}$	$\overline{X} = \frac{\sum_{i=1}^{n} x_i}{n}$
N = number of items in the population	n = number of items in the sample

$$\bar{X} = \frac{60 + 72 + 57 + 90 + 95 + 72}{6}$$

```
> (60+72+57+90+95+72)/6
[1] 74.33333
```

OR;

```
> weight<-c(60, 72, 57,90, 95, 72)
> mean(weight)
[1] 74.33333
```

The Median is the score found at the exact middle of the set of values. One way to compute the median is to list all scores in numerical order, and then locate the score in the center of the sample.

To **find the median**, first order your data. Then calculate the middle position based on *n*, the number of values in your data set.

- If *n* is an odd number, the median lies at the position **(*n* + 1) / 2**.

- If *n* is an even number, the median is the **mean** of the values at positions ***n* / 2 and (*n* / 2) + 1**.

57 60 72 72 90 95

(6/2)+1= 4

57 60 72 *72* 90 95

```
> weight<-c(60, 72, 57, 90, 95, 72)
> median(weight)
[1] 72
```

The mode of a data set is the most frequently occurring value. A data set can often have no mode, one mode or more than one mode – it all depends on how many different values repeat most frequently.

Your data can be: without any mode, **unimodal,** with one mode, **bimodal,** with two modes, or **multimodal,** with four or more modes.

To find the mode, follow these two steps:

- If your data takes the form of numerical values, order the values from low to high. If it takes the form of categories or groupings, sort the values by group, in any order.

- Identify the value or values that occur most frequently.

57 60 72 72 90 95

Mode:72

When to use the mode?

The **level of measurement** of your variables determines when you should use the mode.

The mode works best with categorical data. It is the only measure of central tendency for **nominal** variables, where it can reflect the most commonly found characteristic (e.g., demographic information).

The mode is also useful with **ordinal** variables – for example, to reflect the most popular answer on a ranked scale (e.g., level of agreement).

Measures Of Variability

Variability describes how far apart data points lie from each other and from the center of a distribution. Along with measures of central tendency, measures of variability give you descriptive statistics that summarize your data.

Variability is also referred to as spread, scatter or dispersion. It is most commonly measured with the following:

- **Range:** the difference between the highest and lowest values
- **Interquartile Range:** the range of the middle half of a distribution
- **Standard Deviation:** average distance from the mean
- **Variance:** average of squared distances from the mean

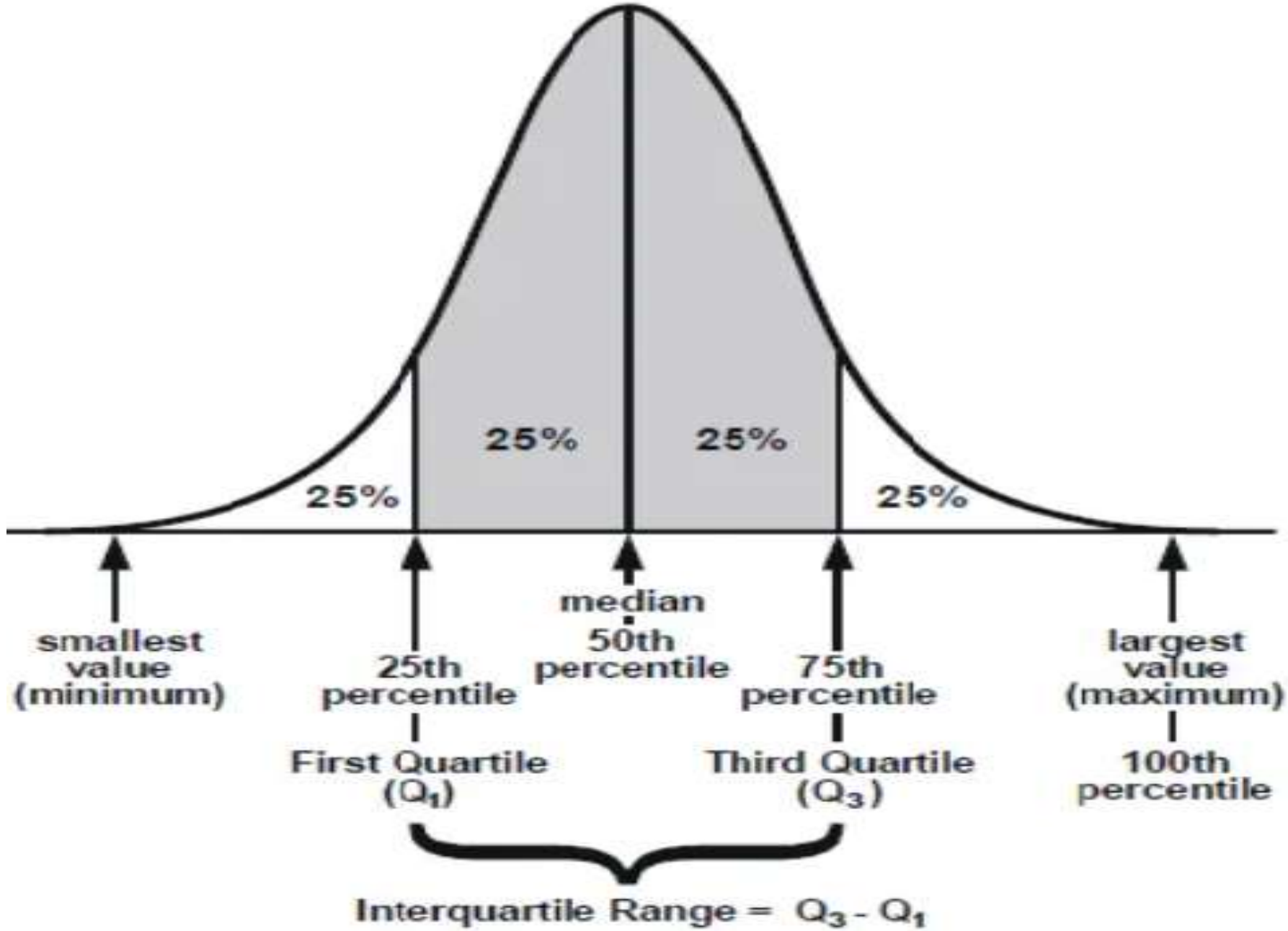
Range

The range is simply the highest value minus the lowest value. In our example distribution, the high value is 36 and the low is 15 , so the range is 36 - 15 = 21 .

The range tells you the spread of your data from the lowest to the highest value in the distribution. It’s the easiest measure of variability to calculate.

Interquartile Range

The interquartile range gives you the spread of the middle of your distribution. For any distribution that’s ordered from low to high, the interquartile range contains half of the values. While the first quartile (Q1) contains the first 25% of values, the fourth quartile (Q4) contains the last 25% of values.


25%
25%
25%
25%
smallest value (minimum)
25th percentile
median
50th percentile
75th percentile
largest value (maximum)
First Quartile (Q_1)
Third Quartile (Q_3)
100th percentile
Interquartile Range = $Q_3 - Q_1$

```
> quantile(weight)
   0%   25%   50%   75% 100%
57.0 63.0 72.0 85.5 95.0
```

Standard Deviation

The Standard Deviation is the average amount of variability in your dataset. It tells you, on average, how far each score lies from the mean. The larger the standard deviation, the more variable the data set is.

If you have data from the entire population, use the population standard deviation formula:

Formula	Explanation
$\sigma = \sqrt{\frac{\Sigma (X - \mu)^2}{N}}$	• σ = population standard deviation • ∑ = sum of... • *X* = each value • μ = population mean • *N* = number of values in the population

Standard deviation formula for samples

If you have data from a sample, use the sample standard deviation formula:

Formula	Explanation
$s = \sqrt{\frac{\Sigma (X - \bar{x})^2}{n - 1}}$	• *s* = sample standard deviation • ∑ = sum of... • *X* = each value • $\bar{x}$ = sample mean • *n* = number of values in the sample

Variance

The **Variance** is the square of the standard deviation. This means that the units of variance are much larger than those of a typical value of a data set.

Variance formula for populations

Formula	Explanation
$\sigma^2 = \frac{\Sigma (X - \mu)^2}{N}$	• σ^2 = population variance • Σ = sum of... • X = each value • μ = population mean • N = number of values in the population

Variance formula for samples

Formula	Explanation
$s^2 = \frac{\Sigma (X - \bar{x})^2}{n - 1}$	• s^2 = sample variance • Σ = sum of... • X = each value • $\bar{x}$ = sample mean • n = number of values in the sample

Variance reflects the degree of spread in the data set. The more spread the data, the larger the variance is in relation to the mean.

While it’s harder to interpret the variance number intuitively, it’s important to calculate variance for comparing different data sets in statistical tests like **ANOVAs**.

```
> sd(weight)
[1] 15.42293
> 15.42293*15.42293
[1] 237.8668
>
>
> var(weight)
[1] 237.8667
```

What’s The Best Measure Of Variability?

The best measure of variability depends on your level of measurement and distribution.

Level of measurement

- For data measured at an ordinal level, the range and interquartile range are the only appropriate measures of variability.

- For more complex interval and ratio levels, the standard deviation and variance are also applicable.

For normal distributions, all measures can be used. The standard deviation and variance are preferred because they take your whole data set into account, but this also means that they are easily influenced by outliers.

For skewed distributions or data sets with outliers, the interquartile range is the best measure. It’s least affected by extreme values because it focuses on the spread in the middle of the data set.

SUMMARY STATISTICS FOR A SINGLE GROUP

Before going into the actual statistical modelling and analysis of a data set, it is often useful to make some simple characterization of the data in terms of summary statistics and graphics.

It is easy to calculate simple summary statistics with R. Here is how to calculate the mean, standard deviation, variance, and median.

```
> x <- rnorm(50)
> mean(x)
[1] -0.01867852
> sd(x)
[1] 1.057685
> var(x)
[1] 1.118698
> median(x)
[1] -0.141053
> quantile(x)
         0%         25%         50%         75%        100%
-3.0102554 -0.6275097 -0.1410530  0.7051967  2.2049396
```

If there are missing values in data, things become a bit more complicated. For illustration, we use the following example: The data set **juul** contains variables from an investigation performed by Anders Juul (Rigshospitalet, Department for Growth and Reproduction) concerning serum IGF-I (insulin-like growth factor) in a group of healthy humans, primarily school children.

The data set is contained in the **ISwR** package and contains a number of variables, of which we only use **igf1** (serum IGF-I) for now, but later in the chapter we also use tanner (Tanner stage of puberty, a classification into five groups, based on appearance of primary and secondary sexual characteristics), sex, and menarche (indicating whether or not a girl has had her first period).

Attempting to calculate the mean of igf1 reveals a problem.

```
> library(ISwR)
> help(package="ISwR")
> data(juul)
> attach(juul)
> mean(igf1)
[1] NA
```

R will not skip missing values unless explicitly requested to do so.

The mean of a vector with an unknown value is unknown. However, you can give the na.rm argument (not available, remove) to request that missing values be removed:

```
> mean(igfl,na.rm=T)
[1] 340.168
```

There is one slightly annoying exception: The length function will not understand na.rm, so we cannot use it to count the number of nonmissing measurements of igf1. However, WE can use

```
> sum(!is.na(igf1))
[1] 1018
```

The above construction uses the fact that if logical values are used in arithmetic, then TRUE is converted to 1 and FALSE to 0.

A nice summary display of a numeric variable is obtained from the summary function:

```
> summary(igfl)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   25.0   202.2   313.5   340.2   462.8   915.0     321
>
```

In fact, it is possible to summarize an entire data frame with

```
> summary(juul)
      age            menarche          sex             igf1       
 Min.   : 0.170   Min.   :1.000   Min.   :1.000   Min.   : 25.0  
 1st Qu.: 9.053   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:202.2  
 Median :12.560   Median :1.000   Median :2.000   Median :313.5  
 Mean   :15.095   Mean   :1.476   Mean   :1.534   Mean   :340.2  
 3rd Qu.:16.855   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:462.8  
 Max.   :83.000   Max.   :2.000   Max.   :2.000   Max.   :915.0  
 NA's   :5        NA's   :635     NA's   :5       NA's   :321    
     tanner         testvol      
 Min.   :1.00   Min.   : 1.000  
 1st Qu.:1.00   1st Qu.: 1.000  
 Median :2.00   Median : 3.000  
 Mean   :2.64   Mean   : 7.896  
 3rd Qu.:5.00   3rd Qu.:15.000  
 Max.   :5.00   Max.   :30.000  
 NA's   :240    NA's   :859     
```

Notice that this data set has menarche, sex, and tanner coded as numeric variables even though they are clearly categorical. This can be mended as follows:

```
> detach(juul)
> juul$sex <- factor(juul$sex,labels=c("M","F"))
> juul$menarche <- factor(juul$menarche,labels=c("No","Yes"))
> juul$tanner <- factor(juul$tanner, labels=c("I","II","III","IV","V"))
> attach(juul)
> summary(juul)
```

```
      age          menarche     sex           igf1          tanner
 Min.   : 0.170   No  :369   M   :621   Min.   : 25.0   I   :515
 1st Qu.: 9.053   Yes :335   F   :713   1st Qu.:202.2   II  :103
 Median :12.560   NA's:635   NA's:  5   Median :313.5   III : 72
 Mean   :15.095                         Mean   :340.2   IV  : 81
 3rd Qu.:16.855                         3rd Qu.:462.8   V   :328
 Max.   :83.000                         Max.   :915.0   NA's:240
 NA's   :5                              NA's   :321
    testvol
 Min.   : 1.000
 1st Qu.: 1.000
 Median : 3.000
 Mean   : 7.896
 3rd Qu.:15.000
 Max.   :30.000
 NA's   :859
```

Notice how the display changes for the factor variables. Note also that **juul** was detached and reattached after the modification.

This is because modifying a data frame does not affect any attached version.

It was not strictly necessary to do it here, because summary works directly on the data frame whether attached or not.

In the above the variables sex, menarche, and tanner were converted to factors with suitable level names (in the raw data these are represented using numeric codes).

The syntax x <- factor(x,labels=...) is a short form for x <- factor(x) followed by levels(x) <- ....

The converted variables were put back into the data frame **juul** replacing the original sex, tanner, and menarche variables. We might also have used the transform function:

```
> juul<-transform(juul,  sex=factor(sex,labels=c("M","F")),menarche=factor(menarche,labels=c("No","Yes")),  tanner=factor(tanner,labels=c("I","II","III","IV","V")))
>
> summary(juul)
      age          menarche     sex          igf1         tanner
 Min.   : 0.170   No  :369   M   :621   Min.   : 25.0   I   :515
 1st Qu.: 9.053   Yes :335   F   :713   1st Qu.:202.2   II  :103
 Median :12.560   NA's:635   NA's:  5   Median :313.5   III : 72
 Mean   :15.095                         Mean   :340.2   IV  : 81
 3rd Qu.:16.855                         3rd Qu.:462.8   V   :328
 Max.   :83.000                         Max.   :915.0   NA's:240
 NA's   :5                              NA's   :321
    testvol
 Min.   : 1.000
 1st Qu.: 1.000
 Median : 3.000
 Mean   : 7.896
 3rd Qu.:15.000
 Max.   :30.000
 NA's   :859
```

Summary Statistics By Groups

When dealing with grouped data, you will often want to have various summary statistics computed within groups. For example, a table of means and standard deviations.

To this end you can use **tapply**. Here is an example concerning the folate concentration in red blood cells according to three types of ventilation during anesthesia (Altman, 1991, p. 208).

We return to this example, which also contains the explanation of the category names.

```
> data(red.cell.folate)
> attach(red.cell.folate)
> tapply(folate,ventilation,mean)
N2O+O2,24h   N2O+O2,op      O2,24h
  316.6250    256.4444    278.0000
```

The **tapply** call takes the folate variable, splits it according to ventilation, and computes the mean for each group.

In the same way, standard deviations and number of observations in the groups can be computed.

```
> tapply(folate,ventilation,sd)
N2O+O2,24h  N2O+O2,op      O2,24h
  58.71709   37.12180   33.75648
>
>
> tapply(folate,ventilation,length)
N2O+O2,24h  N2O+O2,op      O2,24h
         8          9           5
>
```

Try something like this for a nicer display:

```
> xbar <- tapply(folate, ventilation, mean)
> s <- tapply(folate, ventilation, sd)
> n <- tapply(folate, ventilation, length)
> cbind(mean=xbar, std.dev=s, n=n)
                  mean   std.dev n
N2O+O2,24h 316.6250 58.71709 8
N2O+O2,op  256.4444 37.12180 9
O2,24h     278.0000 33.75648 5
```

For the juul data we might want the mean igf1 by tanner group, but of course we run into the problem of missing values again:

```
> tapply(igf1, tanner, mean)
  I   II  III   IV    V
 NA   NA   NA   NA   NA
```

We need to get **tapply** to pass **na.rm=T** as a parameter to mean to make it exclude the missing values.

This is achieved simply by passing it as an additional argument to **tapply**.

```
> tapply(igf1, tanner, mean, na.rm=T)
       I        II       III        IV         V
207.4727 352.6714 483.2222 513.0172 465.3344
```

CONTINGENCY TABLES

Categorical data are usually described in the form of tables. This section outlines how you can create tables from your data and calculate relative frequencies.

We deal mainly with two-way tables. In the first example we enter a table directly, as is required for tables taken from a book or a journal article. A **two-way** table needs to be in a matrix object.

Altman (1991, p. 242) contains an example on caffeine consumption by marital status among women giving birth. That table may be input as follows:

```
> caff.marital<- matrix(c(652,1537,598,242,36,46,38,21,218  ,327,106,67),  nrow=3,byrow=T)
> caff.marital
     [,1] [,2] [,3] [,4]
[1,]  652 1537  598  242
[2,]   36   46   38   21
[3,]  218  327  106   67
```

The matrix function needs an argument containing the table values as a single vector and also the number of rows in the argument nrow. By default, the values are entered columnwise; if rowwise entry is desired, then you need to specify **byrow=T**.

You might also give the number of columns instead of rows using ncol. If exactly one of **ncol** and **nrow** is given, R will compute the other one so that it fits the number of values.

If both **ncol** and **nrow** are given and it does not fit the number of values, the values will be “recycled”, which in some (other!) circumstances can be useful.

To get readable printouts, you can add row and column names to the matrices.

```
> colnames(caff.marital) <- c("0","1-150","151-300",">300")
> rownames(caff.marital) <- c("Married","Prev.married","Single")
> caff.marital
              0 1-150 151-300 >300
Married     652  1537     598  242
Prev.married 36    46      38   21
Single      218   327     106   67
```

In practice, the more frequent case is that you have a database of variables for each person in a data set. In that case, you should do the tabulation with table, xtabs, or ftable.

These functions will generally work for tabulating numeric vectors as well as factor variables, but the latter will have their levels used for row and column names automatically.

Hence, it is recommended to convert numerically coded categorical data into factors. The table function is the oldest and most basic of the three. The other two offer formula-based interfaces and better printing of multiway tables.

Here we look at some other variables in that juul data set, namely sex and menarche; the latter indicates whether or not a girl has had her first period. We can generate some simple tables as follows:

```
> table(sex)
sex
  M   F
621 713
```

```
> table(sex,menarche)
    menarche
sex   No Yes
  M    0   0
  F  369 335
```

```
> table(menarche,tanner)
        tanner
menarche   I  II III  IV   V
     No  221  43  32  14   2
     Yes   1   1   5  26 202
```

Of course, the table of menarche versus sex is just a check on internal consistency of the data.

The table of menarche versus Tanner stage of puberty is more interesting. There are also tables with more than two sides, but not many simple statistical functions use them.

Briefly, to tabulate such data just write, for example, table(factor1,factor2,factor3). To input a table of cell counts, use the array function (an analog of matrix).

Like any matrix, a table can be transposed with the t function:

```
> t(caff.marital)
         Married Prev.married Single
0            652            36    218
1-150       1537            46    327
151-300      598            38    106
>300         242            21     67
```

Marginal Tables And Relative Frequency

It is often desired to compute marginal tables, that is, the sums of the counts along one or the other dimension of a table. Due to missing values, this might not coincide with just tabulating a single factor.

This is done fairly easily using the apply function, but there is also a simplified version called **margin.table**, described below.

First we need to generate the table itself:

tanner.sex is an arbitrarily chosen variable name, which is used for the crosstable of tanner and sex.

```
> tanner.sex <- table(tanner,sex)
> tanner.sex
       sex
tanner   M   F
   I   291 224
   II   55  48
   III  34  38
   IV   41  40
   V   124 204
```

Then we compute the marginal tables:

```
> margin.table(tanner.sex,1)
tanner
  I  II III  IV   V
515 103  72  81 328
```

```
> margin.table(tanner.sex,2)
sex
  M   F
545 554
```

The second argument to **margin.table** is the number of the marginal index: 1 and 2 give row and column totals, respectively.

Relative frequencies in a table are generally expressed as proportions of the row or column totals. Tables of relative frequencies can be constructed using ***prop.table***, as follows:

```
> prop.table(tanner.sex,1)
       sex
tanner         M         F
   I   0.5650485 0.4349515
   II  0.5339806 0.4660194
   III 0.4722222 0.5277778
   IV  0.5061728 0.4938272
   V   0.3780488 0.6219512
```

Note that the rows (1st index) sum to 1. If a table of percentages is desired, just multiply the entire table by 100.

Graphical Display Of Tables

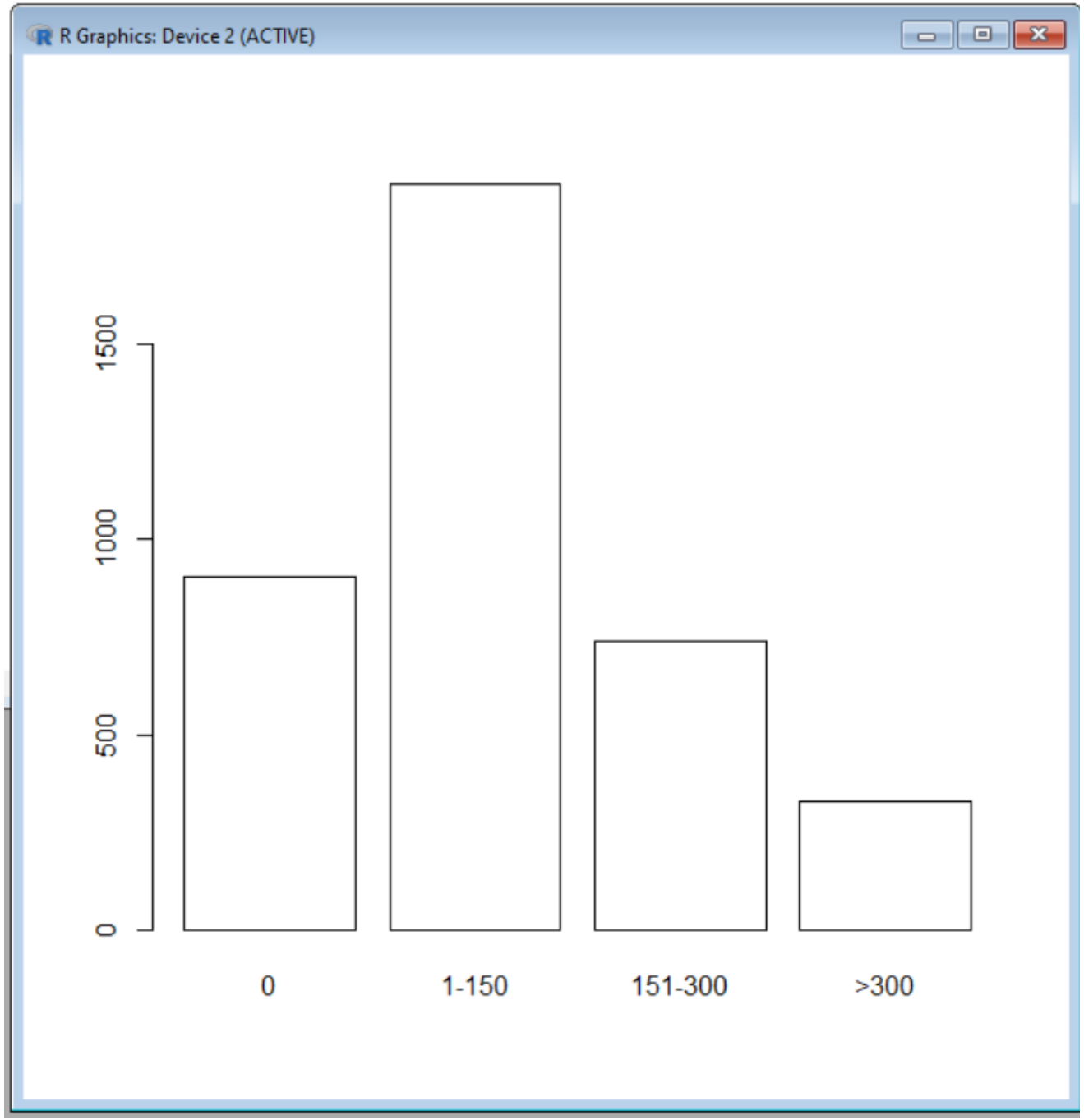
For presentation purposes, it may be desirable to display a graph rather than a table of counts or percentages. In this section the main methods for this are described.

Bar Plots

Bar plots are made using barplot. This function takes an argument, which can be a vector or a matrix. The simplest variant goes as follows (Figure 3.9):

```
> total.caff <- margin.table(caff.marital,2)
> total.caff
      0   1-150 151-300    >300
    906    1910     742     330
```

```
> barplot(total.caff, col="white")
```



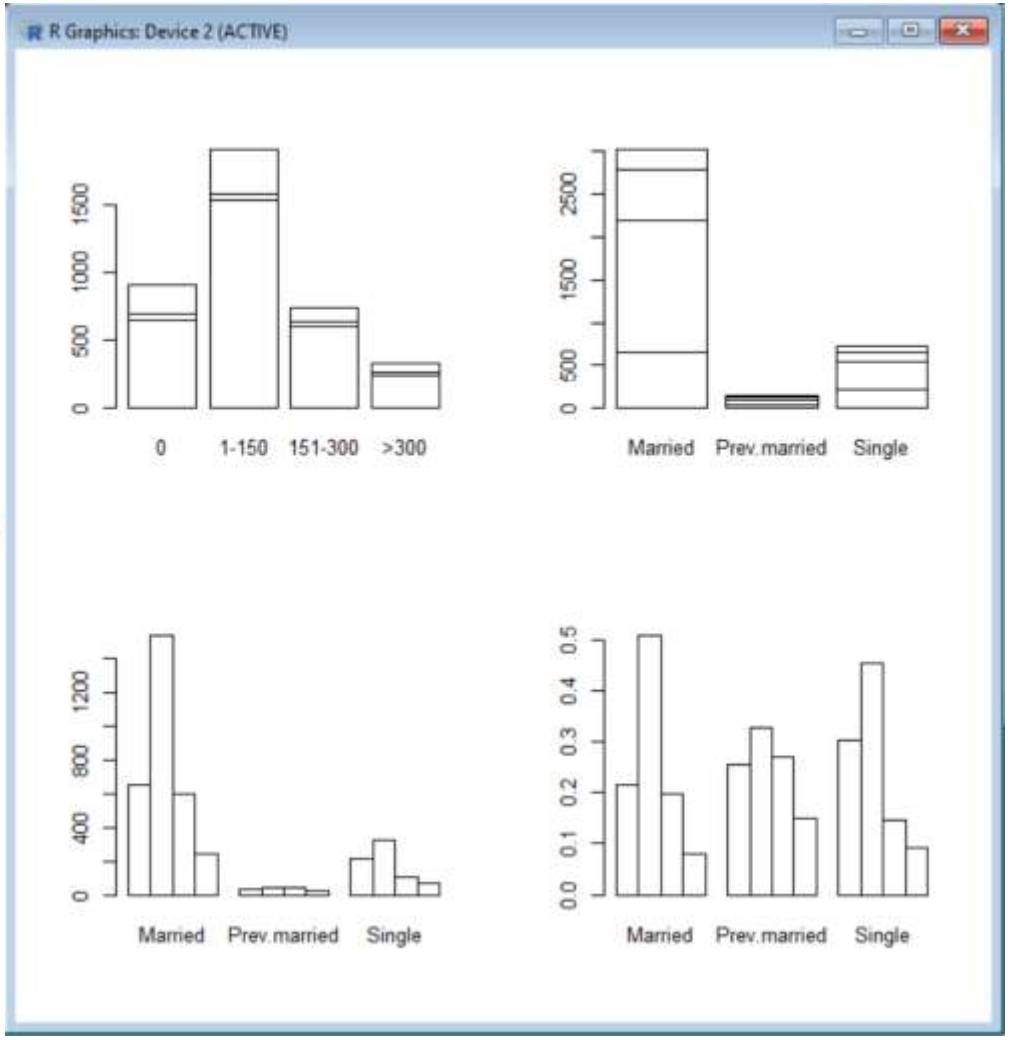
Without the col="white" argument, the plot comes out in colour, but this is not suitable for a black and white book illustration.

If the argument is a matrix, then barplot creates by default a “stacked bar plot”, where the columns are partitioned according to the contributions from different rows of the table.

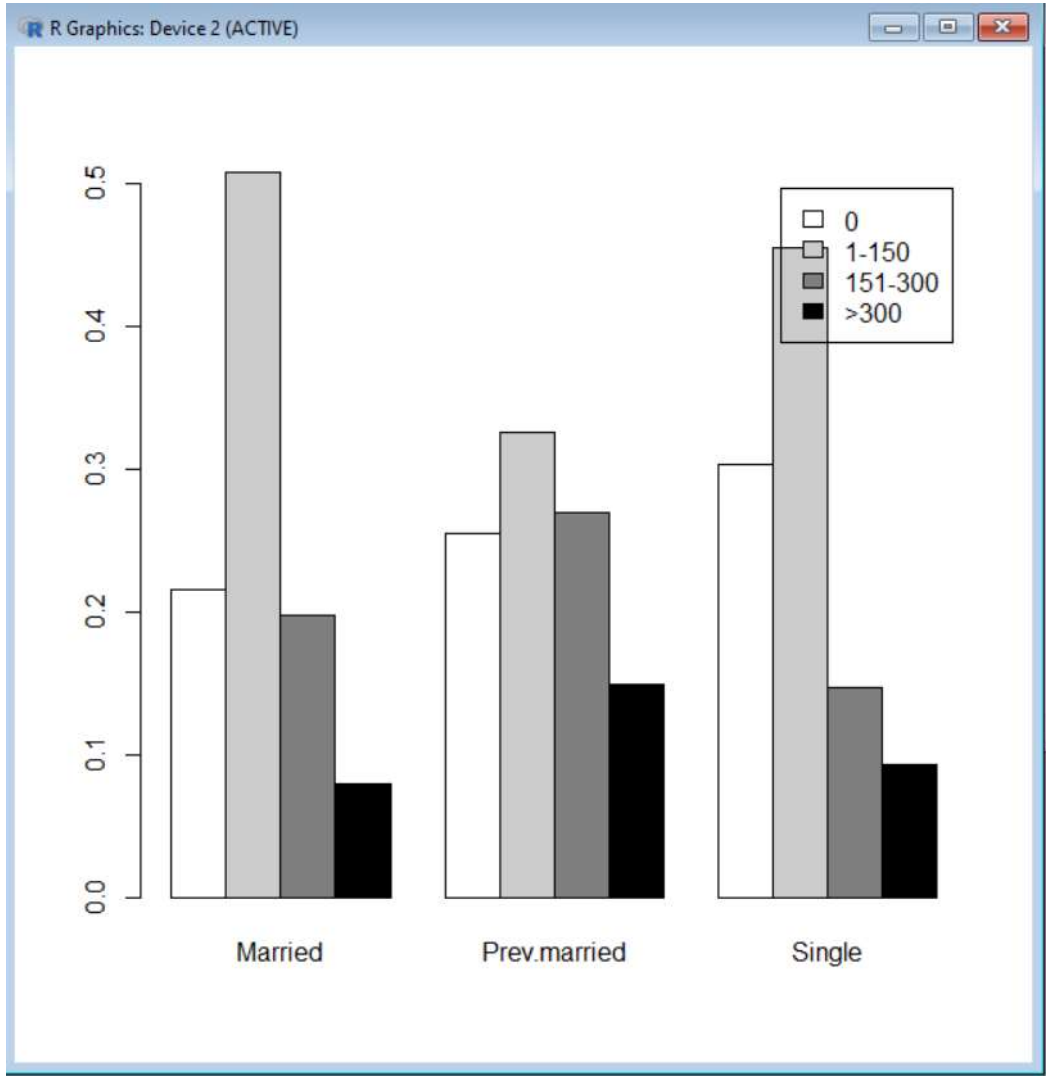
If you want to place the row contributions beside each other instead, you can use the argument beside=T.

A series of variants is found in Figure 3.10, which is constructed as follows:

```
> par(mfrow=c(2,2))
> barplot(caff.marital, col="white")
> barplot(t(caff.marital), col="white")
> barplot(t(caff.marital), col="white", beside=T)
> barplot(prop.table(t(caff.marital),2), col="white", beside=T)
```

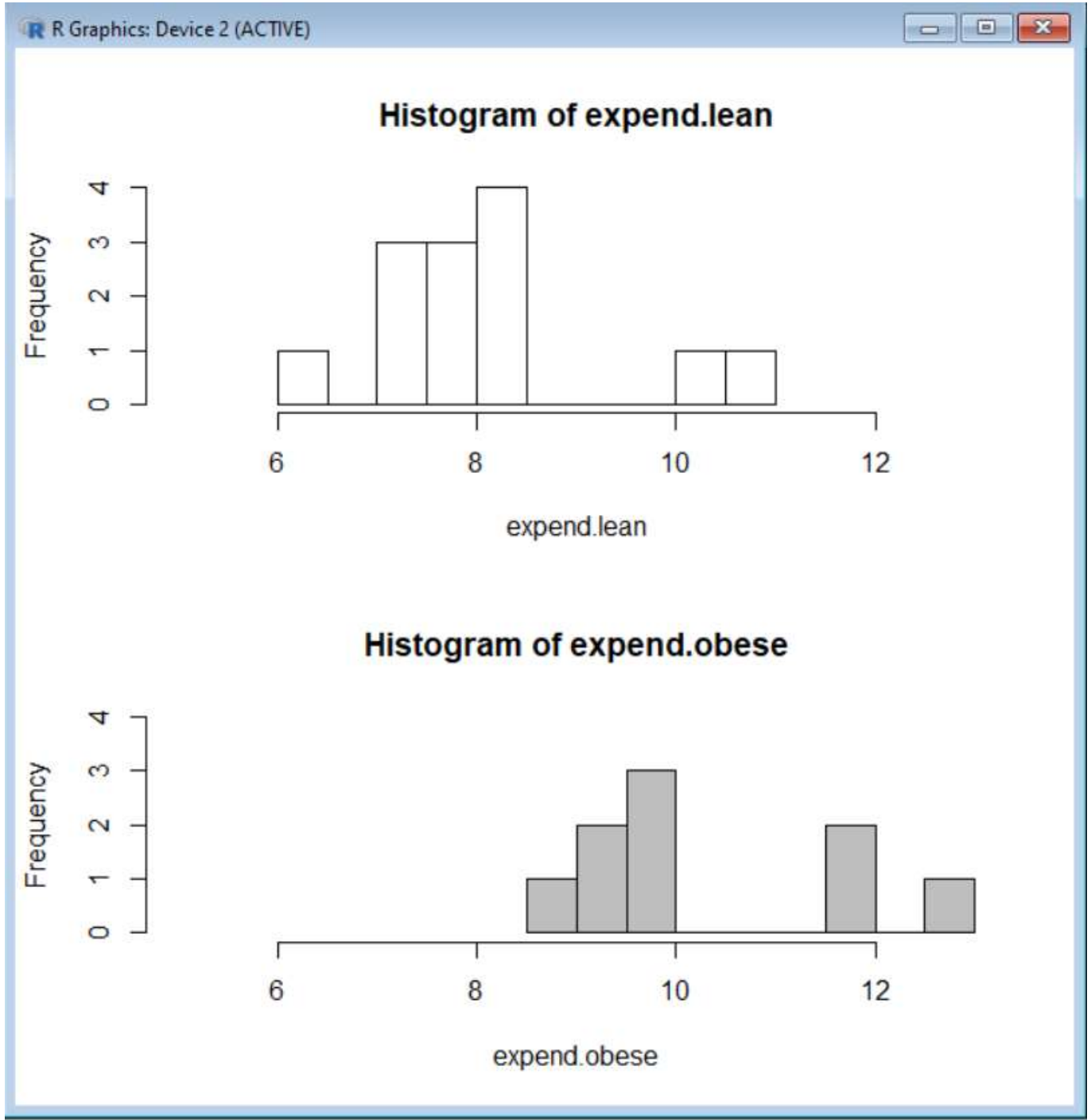


```
> par(mfrow=c(1,1))
> barplot(prop.table(t(caff.marital),2),beside=T, legend.text=colnames(caff.marital), col=c("white","grey80","grey50","black"))
```



For instance, multiples of 0.5 MJ are chosen in the following example using the energy data introduced in energy dataset on the 24-hour energy expenditure for two groups of women: In this example some further techniques of general use are illustrated. The end result is seen in Figure, but first we must fetch the data:

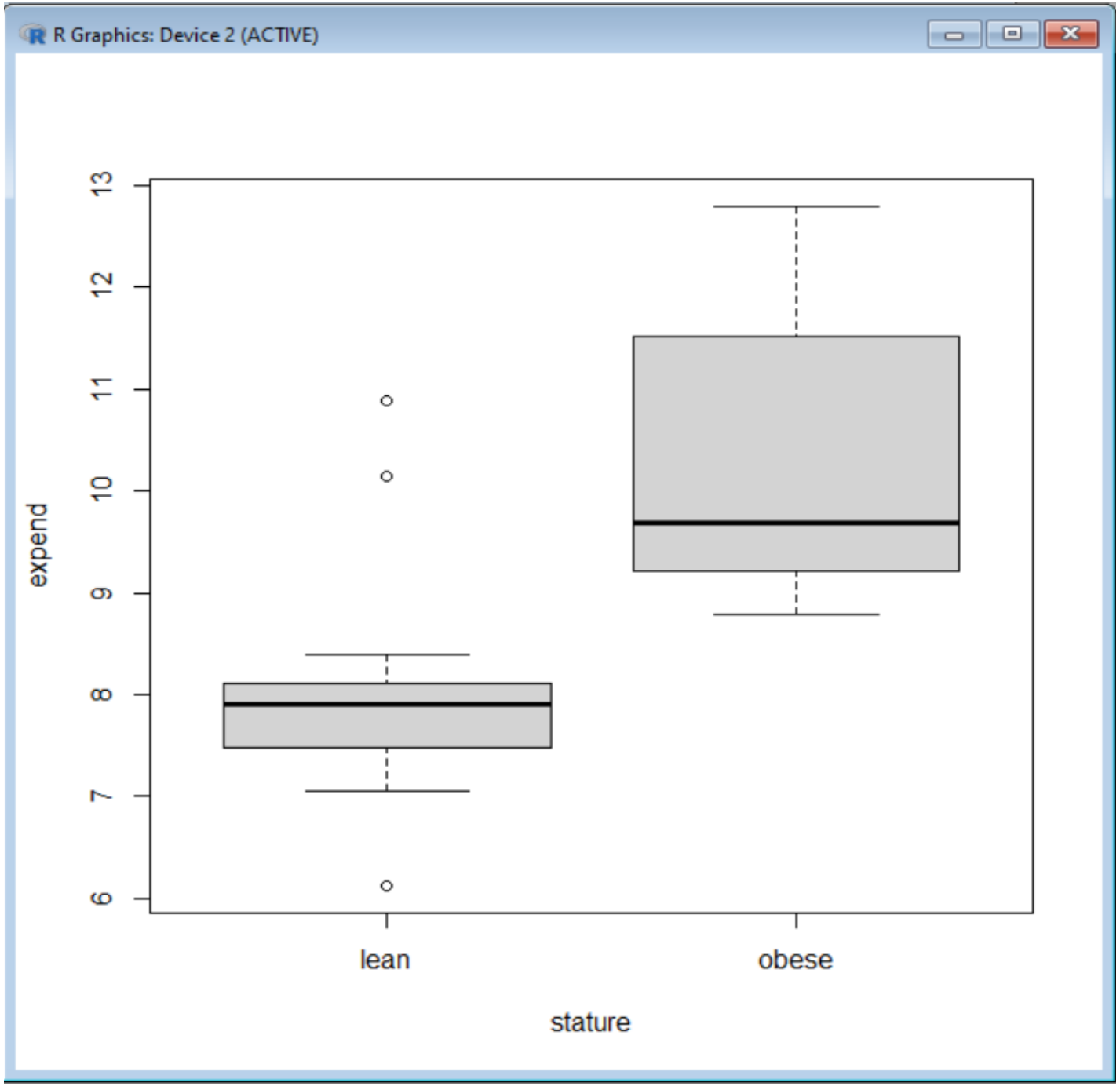
```
> data(energy)
> attach(energy)
The following objects are masked from energy (pos = 3):

    expend, stature

> expend.lean <- expend[stature=="lean"]
> expend.obese <- expend[stature=="obese"]
> par(mfrow=c(2,1))
> hist(expend.lean,breaks=10,xlim=c(5,13),ylim=c(0,4),col="white")
> hist(expend.obese,breaks=10,xlim=c(5,13),ylim=c(0,4),col="grey")
```


R Graphics: Device 2 (ACTIVE)
Histogram of expend.lean
Frequency
0
1
2
3
4
6
8
10
12
expend.lean
Histogram of expend.obese
Frequency
0
1
2
3
4
6
8
10
12
expend.obese

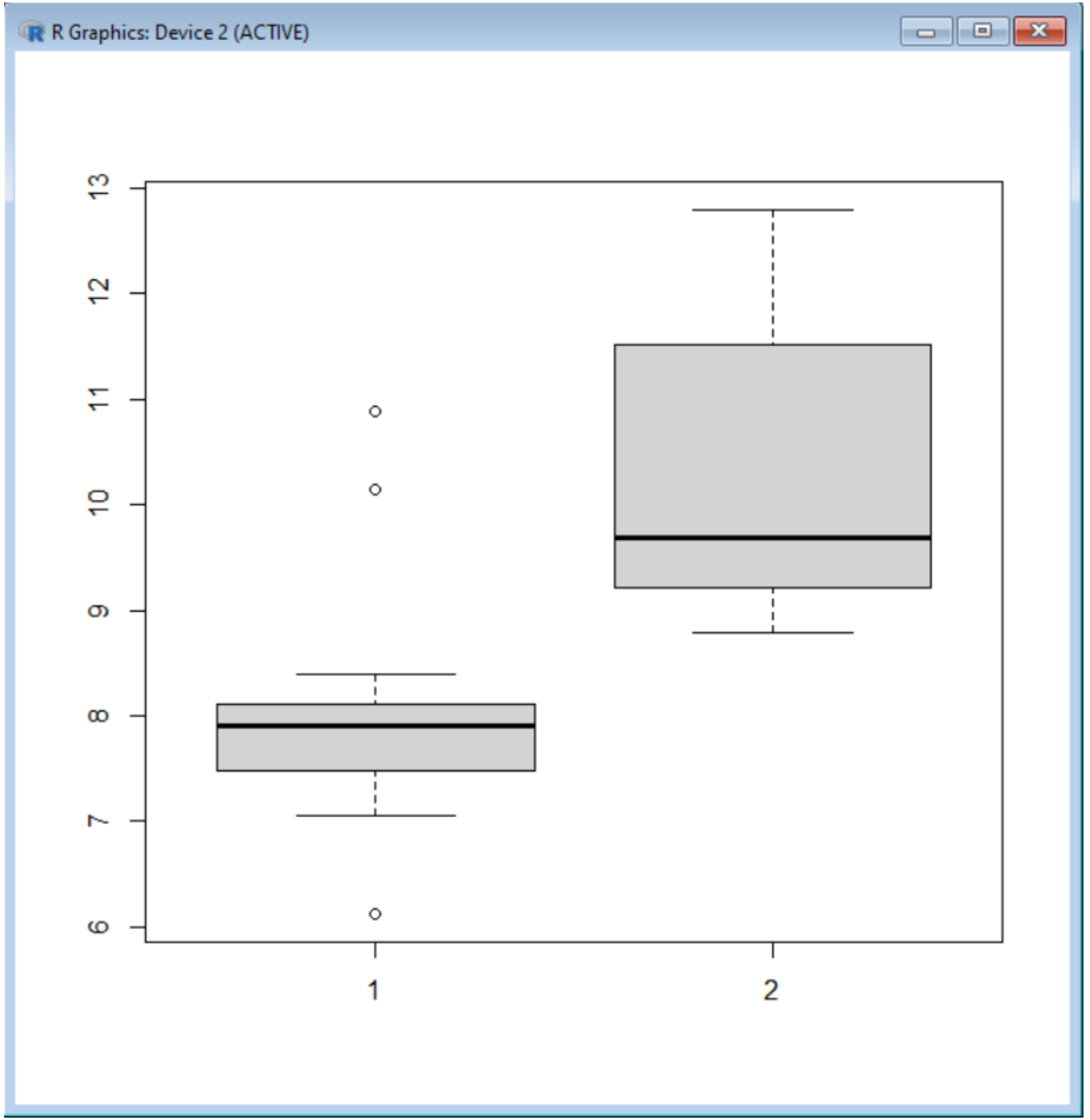
Parallel Boxplots

You might want a set of boxplots from several groups in the same frame. boxplot can handle this, both when data are given in the form of separate vectors from each group and when data are in one long vector and a parallel vector or factor defines the grouping.

```
> boxplot(expend ~ stature)
```



```
> boxplot(expend.lean,expend.obese)
```



The bottom plot has been made using the complete expend vector and the grouping variable fstature. Notation of the type **y~x** should be read “**y described using x**”. This is the first example we see of a model formula.

Chi-Square Test & Fisher Exact Test

A **chi-square test for independence** compares two **variables** in a **contingency table** to see if they are related. In a more general sense, it tests to see whether distributions of **categorical variables** differ from each another.

For the analysis of tables with more than two classes on both sides, you can use **chisq.test** or **fisher.test** although you should note that the latter can be very computationally demanding if the cell counts are large and there are more than two rows or columns.

We have already seen **chisq.test** in a simple example, but with larger tables, some additional features are of interest. An r × c table looks like this:

n_{11}	n_{12}	$\cdots$	n_{1c}	$n_{1.}$
n_{21}	n_{22}	$\cdots$	n_{2c}	$n_{2.}$
$\vdots$	$\vdots$		$\vdots$	$\vdots$
n_{r1}	n_{r2}	$\cdots$	n_{rc}	$n_{r.}$
$n_{.1}$	$n_{.2}$	$\cdots$	$n_{.c}$	$n_{..}$

Such a table can arise from several different sampling plans, and the notion of "**no relation between rows and columns**" is correspondingly different.

The total in each row might be fixed in advance and you would be interested in testing whether the distribution over columns is the same for each row, or vice versa if the column totals were fixed.

It might also be the case that only the total number is chosen and the individuals are grouped randomly according to the row and column criteria.

In the latter case you would be interested in testing the hypothesis of statistical independence, that the probability of an individual falling into the ijth cell is the product pi·p·j of the marginal probabilities.

However, the analysis of the table turns out to be the same in all cases. If there is no relation between rows and columns, then you would expect to have the following cell values:

$$E_{ij} = \frac{n_{i.} \times n_{.j}}{n_{..}}$$

This can be interpreted as distributing each row total according to the proportions in each column (or vice versa) or as distributing the grand total according to the products of the row and column proportions. The test statistic

$$X^2 = \sum \frac{(O - E)^2}{E}$$

has an approximate $\chi 2$ distribution with $(r - 1) \times (c - 1)$ degrees of freedom.

We consider the table with caffeine consumption and marital status from.

```
> caff.marital
               0 1-150 151-300 >300
Married      652  1537     598  242
Prev.married  36    46      38   21
Single       218   327     106   67
> E <- chisq.test(caff.marital)$expected
> E
                     0      1-150   151-300      >300
Married      705.83179 1488.01183 578.06533 257.09105
Prev.married  32.85648   69.26698  26.90895  11.96759
Single       167.31173  352.72119 137.02572  60.94136
> O <- chisq.test(caff.marital)$observed
> O
               0 1-150 151-300 >300
Married      652  1537     598  242
Prev.married  36    46      38   21
Single       218   327     106   67
>
> (O-E)^2/E
                      0    1-150   151-300      >300
Married       4.1055981 1.612783 0.6874502 0.8858331
Prev.married  0.3007537 7.815444 4.5713926 6.8171090
Single       15.3563704 1.875645 7.0249243 0.6023355
```

```
> prop.table(caff.marital,1)
                     0     1-150   151-300       >300
Married      0.2152526 0.5074282 0.1974249 0.07989435
Prev.married 0.2553191 0.3262411 0.2695035 0.14893617
Single       0.3036212 0.4554318 0.1476323 0.09331476
```

```
> prop.table(caff.marital,2)
                     0     1-150    151-300       >300
Married      0.7196468 0.80471204 0.80592992 0.73333333
Prev.married 0.0397351 0.02408377 0.05121294 0.06363636
Single       0.2406181 0.17120419 0.14285714 0.20303030
```