What Is Statistics?

Some Definitions of Statistics

This is a course primarily about statistics, but what exactly is *statistics*? In other words, what is this course about?[1] Here are some definitions of statistics from other people:

- a collection of procedures and principles for gaining information in order to make decisions when faced with uncertainty (J. Utts [**Utt05**]),
- a way of taming uncertainty, of turning raw data into arguments that can resolve profound questions (T. Amabile [**fMA89**]),
- the science of drawing conclusions from data with the aid of the mathematics of probability (S. Garfunkel [**fMA86**]),
- the explanation of variation in the context of what remains unexplained (D. Kaplan [**Kap09**]),
- the mathematics of the collection, organization, and interpretation of numerical data, especially the analysis of a population's characteristics by inference from sampling (American Heritage Dictionary [**AmH82**]).

While not exactly the same, these definitions highlight four key elements of statistics.

Data – the raw material

Data are the raw material for doing statistics. We will learn more about different types of data, how to collect data, and how to summarize data as we go along. This will be the primary focus of Chapter 1.

[1] As we will see, the words *statistic* and *statistics* get used in more than one way. More on that later.

From Foundations & App. of Statistics – Randall Pruim

Information – the goal

The goal of doing statistics is to gain some information or to make a decision. Statistics is useful because it helps us answer questions like the following:

- Which of two treatment plans leads to the best clinical outcomes?
- How strong is an I-beam constructed according to a particular design?
- Is my cereal company complying with regulations about the amount of cereal in its cereal boxes?

In this sense, statistics is a science – a method for obtaining new knowledge.

Uncertainty – the context

The tricky thing about statistics is the uncertainty involved. If we measure one box of cereal, how do we know that all the others are similarly filled? If every box of cereal were identical and every measurement perfectly exact, then one measurement would suffice. But the boxes may differ from one another, and even if we measure the same box multiple times, we may get different answers to the question *How much cereal is in the box?*

So we need to answer questions like *How many boxes should we measure?* and *How many times should we measure each box?* Even so, there is no answer to these questions that will give us absolute certainty. So we need to answer questions like *How sure do we need to be?*

Probability – the tool

In order to answer a question like *How sure do we need to be?*, we need some way of measuring our level of certainty. This is where mathematics enters into statistics. Probability is the area of mathematics that deals with reasoning about uncertainty. So before we can answer the statistical questions we just listed, we must first develop some skill in probability. Chapter 2 provides the foundation that we need.

Once we have developed the necessary tools to deal with uncertainty, we will be able to give good answers to our statistical questions. But before we do that, let's take a bird's eye view of the processes involved in a statistical study. We'll come back and fill in the details later.

A First Example: The Lady Tasting Tea

There is a famous story about a lady who claimed that tea with milk tasted different depending on whether the milk was added to the tea or the tea added to the milk. The story is famous because of the setting in which she made this claim. She was attending a party in Cambridge, England, in the 1920s. Also in attendance were a number of university dons and their wives. The scientists in attendance scoffed at the woman and her claim. What, after all, could be the difference?

All the scientists but one, that is. Rather than simply dismiss the woman's claim, he proposed that they decide how one should *test* the claim. The tenor of the conversation changed at this suggestion, and the scientists began to discuss how the claim should be tested. Within a few minutes cups of tea with milk had been prepared and presented to the woman for tasting.

Let's take this simple example as a prototype for a statistical study. What steps are involved?

(1) Determine the question of interest.

Just what is it we want to know? It may take some effort to make a vague idea precise. The precise questions may not exactly correspond to our vague questions, and the very exercise of stating the question precisely may modify our question. Sometimes we cannot come up with any way to answer the question we really want to answer, so we have to live with some other question that is not exactly what we wanted but is something we can study and will (we hope) give us some information about our original question.

In our example this question seems fairly easy to state: Can the lady tell the difference between the two tea preparations? But we need to refine this question. For example, are we asking if she *always* correctly identifies cups of tea or merely if she does better than we could do ourselves (by guessing)?

(2) Determine the population.

Just who or what do we want to know about? Are we only interested in this one woman or women in general or only women who claim to be able to distinguish tea preparations?

(3) Select measurements.

We are going to need some data. We get our data by making some measurements. These might be physical measurements with some device (like a ruler or a scale). But there are other sorts of measurements too, like the answer to a question on a form. Sometimes it is tricky to figure out just what to measure. (How do we measure happiness or intelligence, for example?) Just how we do our measuring will have important consequences for the subsequent statistical analysis.

In our example, a measurement may consist of recording for a given cup of tea whether the woman's claim is correct or incorrect.

(4) Determine the sample.

Usually we cannot measure every individual in our population; we have to select some to measure. But how many and which ones? These are important questions that must be answered. Generally speaking, bigger is better, but it is also more expensive. Moreover, no size is large enough if the sample is selected inappropriately.

Suppose we gave the lady one cup of tea. If she correctly identifies the mixing procedure, will we be convinced of her claim? She might just be guessing; so we should probably have her taste more than one cup. Will we be convinced if she correctly identifies 5 cups? 10 cups? 50 cups?

What if she makes a mistake? If we present her with 10 cups and she correctly identifies 9 of the 10, what will we conclude? A success rate of 90% is, it seems, much better than just guessing, and anyone can make a mistake now and then. But what if she correctly identifies 8 out of 10? 80 out of 100?

And how should we prepare the cups? Should we make 5 each way? Does it matter if we tell the woman that there are 5 prepared each way? Should we flip a coin to decide even if that means we might end up with 3 prepared one way and 7 the other way? Do any of these differences matter?

(5) Make and record the measurements.

Once we have the design figured out, we have to do the legwork of data collection. This can be a time-consuming and tedious process. In the case of the lady tasting tea, the scientists decided to present her with ten cups of tea which were quickly prepared. A study of public opinion may require many thousands of phone calls or personal interviews. In a laboratory setting, each measurement might be the result of a carefully performed laboratory experiment.

(6) Organize the data.

Once the data have been collected, it is often necessary or useful to organize them. Data are typically stored in spreadsheets or in other formats that are convenient for processing with statistical packages. Very large data sets are often stored in databases.

Part of the organization of the data may involve producing graphical and numerical summaries of the data. We will discuss some of the most important of these kinds of summaries in Chapter 1. These summaries may give us initial insights into our questions or help us detect errors that may have occurred to this point.

(7) Draw conclusions from data.

Once the data have been collected, organized, and analyzed, we need to reach a conclusion. Do we believe the woman's claim? Or do we think she is merely guessing? How sure are we that this conclusion is correct?

Eventually we will learn a number of important and frequently used methods for drawing inferences from data. More importantly, we will learn the basic framework used for such procedures so that it should become easier and easier to learn new procedures as we become familiar with the framework.

(8) Produce a report.

Typically the results of a statistical study are reported in some manner. This may be as a refereed article in an academic journal, as an internal report to a company, or as a solution to a problem on a homework assignment. These reports may themselves be further distilled into press releases, newspaper articles, advertisements, and the like. The mark of a good report is that it provides the essential information about each of the steps of the study.

As we go along, we will learn some of the standard terminology and procedures that you are likely to see in basic statistical reports and will gain a framework for learning more.

At this point, you may be wondering who the innovative scientist was and what the results of the experiment were. The scientist was R. A. Fisher, who first described this situation as a pedagogical example in his 1925 book on statistical methodology [**Fis25**]. We'll return to this example in Sections 2.4.1 and 2.7.3.

Chapter 1

Summarizing Data

It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts.
Sherlock Holmes [**Doy27**]

Graphs are essential to good statistical analysis.
F. J. Anscombe [**Ans73**]

Data are the raw material of statistics.

We will organize data into a 2-dimensional schema, which we can think of as rows and columns in a spreadsheet. The rows correspond to the **individuals** (also called **cases**, **subjects**, or **units** depending on the context of the study). The columns correspond to variables. In statistics, a **variable** is one of the measurements made for each individual. Each individual has a **value** for each variable. Or at least that is our intent. Very often some of the data are **missing**, meaning that values of some variables are not available for some of the individuals.

How data are collected is critically important, and good statistical analysis requires that the data were collected in an appropriate manner. We will return to the issue of how data are (or should be) collected later. In this chapter we will focus on the data themselves. We will use R to manipulate data and to produce some of the most important numerical and graphical summaries of data. A more complete introduction to R can be found in Appendix A.

1.1. Data in R

Most data sets in R are stored in a structure called a data frame that reflects the 2-dimensional structure described above. A number of data sets are included with the basic installation of R. The `iris` data set, for example, is a famous data set containing a number of physical measurements of three varieties of iris. These data were published by Edgar Anderson in 1935 [**And35**] but are famous because R. A. Fisher [**Fis36**] gave a statistical analysis of these data that appeared a year later.

The `str()` function provides our first overview of the data set.

iris-str

```
> str(iris)
'data.frame':          150 obs. of  5 variables:
$ Sepal.Length:num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
$ Sepal.Width :num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
$ Petal.Length:num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
$ Petal.Width :num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
$ Species :Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1
  1 1 1 ...
```

From this output we learn that our data set has 150 observations (rows) and 5 variables (columns). Also displayed is some information about the type of data stored in each variable and a few sample values.

While we could print the entire data frame to the screen, this is inconvenient for large data sets. We can look at the first few or last few rows of the data set using `head()` and `tail()`. This is enough to give us a feel for how the data look.

iris-head

```
> head(iris,n=3)              # first three fows
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
```

iris-tail

```
> tail(iris,n=3)              # last three rows
    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
148          6.5         3.0          5.2         2.0 virginica
149          6.2         3.4          5.4         2.3 virginica
150          5.9         3.0          5.1         1.8 virginica
```

We can access any subset we want by directly specifying which rows and columns are of interest to us.

iris-subset

```
> iris[c(1:3,148:150),3:5]  # first and last rows, only 3 columns
    Petal.Length Petal.Width   Species
1            1.4         0.2    setosa
2            1.4         0.2    setosa
3            1.3         0.2    setosa
148          5.2         2.0 virginica
149          5.4         2.3 virginica
150          5.1         1.8 virginica
```

It is also possible to look at just one variable using the `$` operator.

iris-vector

```
> iris$Sepal.Length     # get one variable and print as vector
  [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7
 [17] 5.4 5.1 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.7 4.8 5.4
< 5 lines removed >
[113] 6.8 5.7 5.8 6.4 6.5 7.7 7.7 6.0 6.9 5.6 7.7 6.3 6.7 7.2 6.2 6.1
```

> **Box 1.1. Using the snippet( ) function**
>
> If you have installed the `fastR` package (and any other additional packages that may be needed for a particular example) you can execute the code from this book on your own computer using `snippet()`. For example,
>
> ```
> snippet('iris-str')
> ```
>
> will both display and execute the first code block on page 2, and
>
> ```
> snippet('iris-str', exec=FALSE)
> ```
>
> will display the code without executing it. Keep in mind that some code blocks assume that prior blocks have already been executed and will not work as expected if this is not true.

```
[129] 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7 6.3 6.4 6.0 6.9 6.7 6.9 5.8 6.8
[145] 6.7 6.7 6.3 6.5 6.2 5.9
```

iris-vector2

```
> iris$Species          # get one variable and print as vector
  [1] setosa     setosa     setosa     setosa     setosa
  [6] setosa     setosa     setosa     setosa     setosa
 [11] setosa     setosa     setosa     setosa     setosa
< 19 lines removed >
[111] virginica  virginica  virginica  virginica  virginica
[116] virginica  virginica  virginica  virginica  virginica
[121] virginica  virginica  virginica  virginica  virginica
[126] virginica  virginica  virginica  virginica  virginica
[131] virginica  virginica  virginica  virginica  virginica
[136] virginica  virginica  virginica  virginica  virginica
[141] virginica  virginica  virginica  virginica  virginica
[146] virginica  virginica  virginica  virginica  virginica
Levels: setosa versicolor virginica
```

This is not a particularly good way to get a feel for data. There are a number of graphical and numerical summaries of a variable or set of variables that are usually preferred to merely listing all the values – especially if the data set is large. That is the topic of our next section.

It is important to note that the name `iris` is not reserved in R for this data set. There is nothing to prevent you from storing something else with that name. If you do, you will no longer have access to the `iris` data set unless you first reload it, at which point the previous contents of `iris` are lost.

iris-reload

```
> iris <- 'An iris is a beautiful flower.'
> str(iris)
chr "An iris is a beautiful flower."
> data(iris)            # explicitly reload the data set
> str(iris)
'data.frame':       150 obs. of  5 variables:
$ Sepal.Length:num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
$ Sepal.Width :num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
```

```
$ Petal.Length:num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
$ Petal.Width :num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
$ Species :Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1
  1 1 1 ...
```

The `fastR` package includes data sets and other utilities to accompany this text. Instructions for installing `fastR` appear in the preface. We will use data sets from a number of other R packages as well. These include the CRAN packages `alr3`, `car`, `DAAG`, `Devore6`, `faraway`, `Hmisc`, `MASS`, and `multcomp`. Appendix A includes instructions for reading data from various file formats, for entering data manually, for obtaining documentation on R functions and data sets, and for installing packages from CRAN.

1.2. Graphical and Numerical Summaries of Univariate Data

Now that we can get our hands on some data, we would like to develop some tools to help us understand the **distribution** of a variable in a data set. By *distribution* we mean answers to two questions:

- What values does the variable take on?
- With what frequency?

Simply listing all the values of a variable is not an effective way to describe a distribution unless the data set is quite small. For larger data sets, we require some better methods of summarizing a distribution.

1.2.1. Tabulating Data

The types of summaries used for a variable depend on the kind of variable we are interested in. Some variables, like `iris$Species`, are used to put individuals into categories. Such variables are called **categorical** (or **qualitative**) variables to distinguish them from **quantitative** variables which have numerical values on some numerically meaningful scale. `iris$Sepal.Length` is an example of a quantitative variable.

Usually the categories are either given descriptive names (our preference) or numbered consecutively. In R, a categorical variable is usually stored as a factor. The possible categories of an R factor are called levels, and you can see in the output above that R not only lists out all of the values of `iris$species` but also provides a list of all the possible levels for this variable. A more useful summary of a categorical variable can be obtained using the `table()` function.

iris-table

```
> table(iris$Species)  # make a table of values

    setosa versicolor  virginica
        50         50         50
```

From this we can see that there were 50 of each of three species of iris.

Tables can be used for quantitative data as well, but often this does not work as well as it does for categorical data because there are too many categories.

iris-table2

```
> table(iris$Sepal.Length)  # make a table of values

4.3 4.4 4.5 4.6 4.7 4.8 4.9   5 5.1 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9
  1   3   1   4   2   5   6  10   9   4   1   6   7   6   8   7   3
  6 6.1 6.2 6.3 6.4 6.5 6.6 6.7 6.8 6.9   7 7.1 7.2 7.3 7.4 7.6 7.7
  6   6   4   9   7   5   2   8   3   4   1   1   3   1   1   1   4
7.9
  1
```

Sometimes we may prefer to divide our quantitative data into two groups based on a threshold or some other boolean test.

iris-logical

```
> table(iris$Sepal.Length > 6.0)

FALSE  TRUE
   89    61
```

The cut() function provides a more flexible way to build a table from quantitative data.

iris-cut

```
> table(cut(iris$Sepal.Length,breaks=2:10))

 (2,3]  (3,4]  (4,5]  (5,6]  (6,7]  (7,8]  (8,9] (9,10]
     0      0     32     57     49     12      0      0
```

The cut() function partitions the data into sections, in this case with break points at each integer from 2 to 10. (The breaks argument can be used to set the break points wherever one likes.) The result is a categorical variable with levels describing the interval in which each original quantitative value falls. If we prefer to have the intervals closed on the other end, we can achieve this using right=FALSE.

iris-cut2

```
> table(cut(iris$Sepal.Length,breaks=2:10,right=FALSE))

 [2,3)  [3,4)  [4,5)  [5,6)  [6,7)  [7,8)  [8,9) [9,10)
     0      0     22     61     54     13      0      0
```

Notice too that it is possible to define factors in R that have levels that do not occur. This is why the 0's are listed in the output of table(). See ?factor for details.

A tabular view of data like the example above can be converted into a visual representation called a histogram. There are two R functions that can be used to build a histogram: hist() and histogram(). hist() is part of core R. histogram() can only be used after first loading the lattice graphics package, which now comes standard with all distributions of R. Default versions of each are depicted in Figure 1.1. A number of arguments can be used to modify the resulting plot, set labels, choose break points, and the like.

Looking at the plots generated by histogram() and hist(), we see that they use different scales for the vertical axis. The default for histogram() is to use [illegible] entire data set). By contrast, hist() uses counts. The shapes of

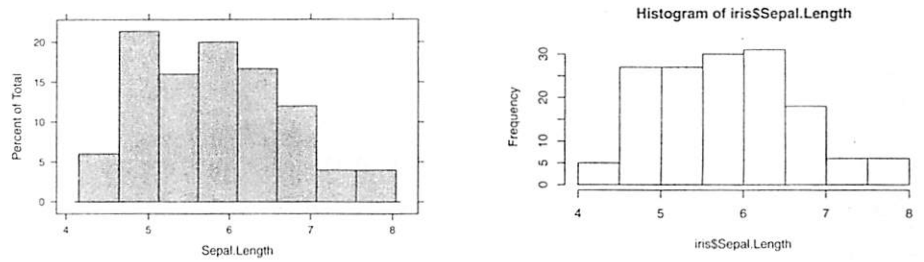


Figure 1.1. Comparing two histogram functions: `histogram()` on the left, and `hist()` on the right.

the two histograms differ because they use slightly different algorithms for choosing the default break points. The user can, of course, override the default break points (using the `breaks` argument). There is a third scale, called the density scale, that is often used for the vertical axis. This scale is designed so that the *area* of each bar is equal to the proportion of data it represents. This is especially useful for histograms that have **bins** (as the intervals between break points are typically called in the context of histograms) of different widths. Figure 1.2 shows an example of such a histogram generated using the following code:

iris-histo-density
```
> histogram(~Sepal.Length,data=iris,type="density",
+     breaks=c(4,5,5.5,6,6.5,7,8,10))
```

We will generally use the newer `histogram()` function because it has several nice features. One of these is the ability to split up a plot into subplots called panels. For example, we could build a separate panel for each species in the iris data set. Figure 1.2 suggests that part of the variation in sepal length is associated with the differences in species. Setosa are generally shorter, virginica longer, and versicolor intermediate. The right-hand plot in Figure 1.2 was created using

iris-condition
```
> histogram(~Sepal.Length|Species,data=iris)
```

If we only want to see the data from one species, we can select a subset of the data using the `subset` argument.

iris-histo-subset
```
> histogram(~Sepal.Length|Species,data=iris,
+     subset=Species=="virginica")
```

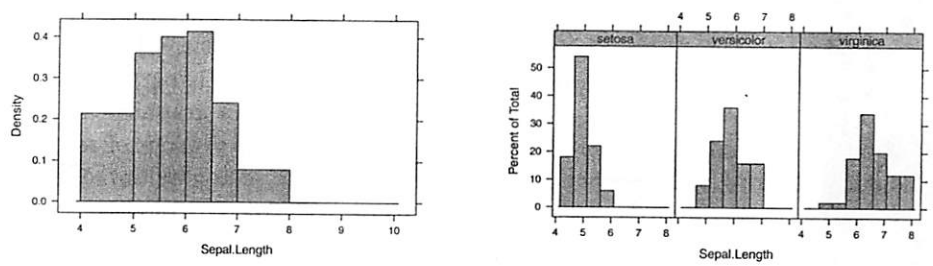


Figure 1.2. Left: A density histogram of sepal length using unequal bin widths. Right: A histogram of sepal length by species.

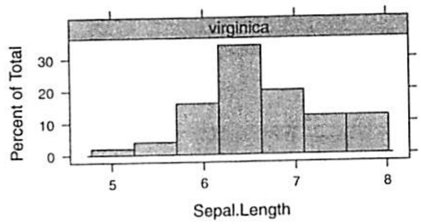


Figure 1.3. This histogram is the result of selecting a subset of the data using the `subset` argument.

By keeping the `groups` argument, our plot will continue to have a strip at the top identifying the species even though there will only be one panel in our plot (Figure 1.3).

The `lattice` graphing functions all use a similar formula interface. The generic form of a formula is

```
y ~ x | z
```

which can often be interpreted as "y modeled by x conditioned on z". For plotting, `y` will typically indicate a variable presented on the vertical axis, and `x` a variable to be plotted along the horizontal axis. In the case of a histogram, the values for the vertical axis are computed from the `x` variable, so `y` is omitted. The condition `z` is a variable that is used to break the data into sections which are plotted in separate panels. When `z` is categorical, there is one panel for each level of `z`. When `z` is quantitative, the data is divided into a number of sections based on the values of `z`. This works much like the `cut()` function, but some data may appear in more than one panel. In R terminology, each panel represents a shingle of the data. The term shingle is supposed to evoke an image of overlapping coverage like the shingles on a roof. Finer control over the number of panels can be obtained by using `equal.count()` or `co.intervals()` to make the shingles directly. See Figure 1.4.

1.2.2. Shapes of Distributions

A histogram gives a shape to a distribution, and distributions are often described in terms of these shapes. The exact shape depicted by a histogram will depend not only on the data but on various other choices, such as how many bins are used, whether the bins are equally spaced across the range of the variable, and just where the divisions between bins are located. But *reasonable* choices of these arguments will usually lead to histograms of similar shape, and we use these shapes to describe the underlying distribution as well as the histogram that represents it.

Some distributions are approximately **symmetrical** with the distribution of the larger values looking like a mirror image of the distribution of the smaller values. We will call a distribution **positively skewed** if the portion of the distribution with larger values (the right of the histogram) is more spread out than the other side. Similarly, a distribution is **negatively skewed** if the distribution deviates from symmetry in the opposite manner. Later we will learn a way to measure

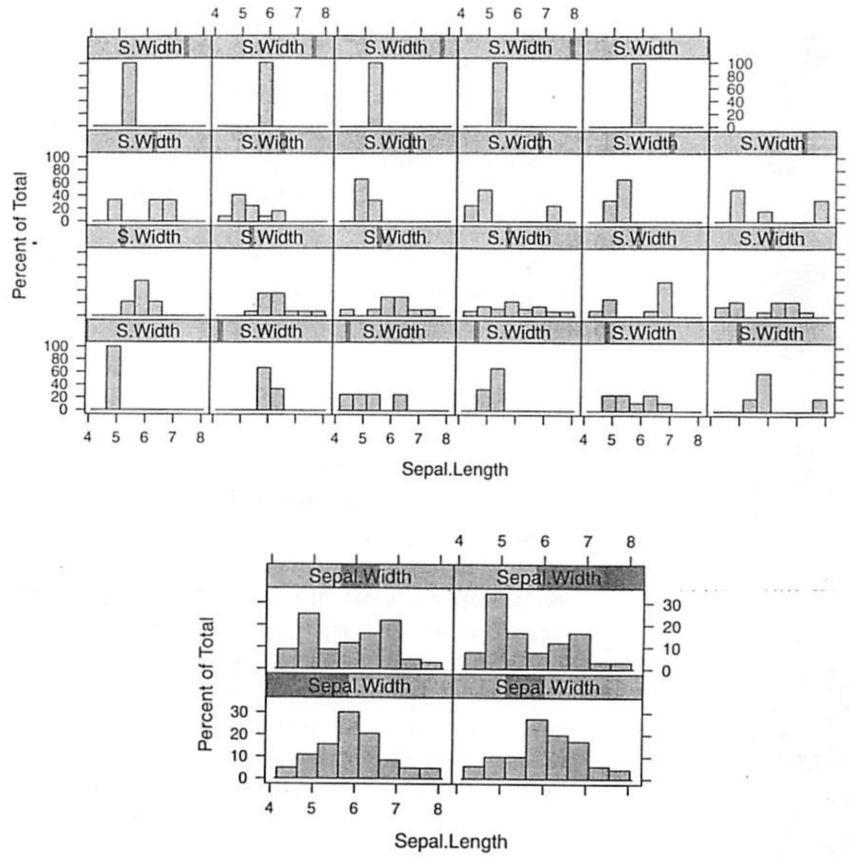


Figure 1.4. The output of histogram(~Sepal.Length|Sepal.Width,iris) and histogram(~Sepal.Length|equal.count(Sepal.Width,number=4),iris).

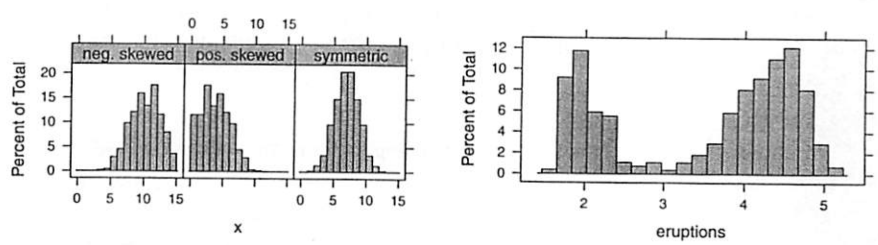


Figure 1.5. Left: Skewed and symmetric distributions. Right: Old Faithful eruption times illustrate a bimodal distribution.

the degree and direction of skewness with a number; for now it is sufficient to describe distributions qualitatively as symmetric or skewed. See Figure 1.5 for some examples of symmetric and skewed distributions.

Notice that each of these distributions is clustered around a center where most of the values are located. We say that such distributions are **unimodal**. Shortly we will discuss ways to summarize the location of the "center" of unimodal distributions numerically. But first we point out that some distributions have other shapes that are not characterized by a strong central tendency. One famous example is eruption times of the Old Faithful geyser in Yellowstone National park.

faithful-histogram

```
> plot <- histogram(~eruptions,faithful,n=20)
```

produces the histogram in Figure 1.5 which shows a good example of a **bimodal** distribution. There appear to be two groups or kinds of eruptions, some lasting about 2 minutes and others lasting between 4 and 5 minutes.

1.2.3. Measures of Central Tendency

Qualitative descriptions of the shape of a distribution are important and useful. But we will often desire the precision of numerical summaries as well. Two aspects of unimodal distributions that we will often want to measure are central tendency (what is a typical value? where do the values cluster?) and the amount of variation (are the data tightly clustered around a central value, or more spread out?).

Two widely used measures of center are the **mean** and the **median**. You are probably already familiar with both. The mean is calculated by adding all the values of a variable and dividing by the number of values. Our usual notation will be to denote the n values as $x_1, x_2, \ldots x_n$, and the mean of these values as $\overline{x}$. Then the formula for the mean becomes

$$\overline{x} = \frac{\sum_{i=1}^{n} x_i}{n} .$$

The median is a value that splits the data in half – half of the values are smaller than the median and half are larger. By this definition, there could be more than one median (when there are an even number of values). This ambiguity is removed by taking the mean of the "two middle numbers" (after sorting the data). See the exercises for some problems that explore aspects of the mean and median that may be less familiar.

The mean and median are easily computed in R. For example,

iris-mean-median

```
> mean(iris$Sepal.Length); median(iris$Sepal.Length)
[1] 5.8433
[1] 5.8
```

Of course, we have already seen (by looking at histograms), that there are some differences in sepal length between the various species, so it would be better to compute the mean and median separately for each species. While one can use the built-in `aggregate()` function, we prefer to use the `summary()` function from the `Hmisc` package. This function uses the same kind of formula notation that the `lattice` graphics functions use.

iris-Hmisc-summary

```
> require(Hmisc) # load Hmisc package
> summary(Sepal.Length~Species,iris)          # default function is mean
Sepal.Length    N=150
```

> **Box 1.2. R packages used in this text**
>
> From now on we will assume that the `lattice`, `Hmisc`, and `fastR` packages have been loaded and will not show the loading of these packages in our examples. If you try an example in this book and R reports that it cannot find a function or data set, it is likely that you have failed to load one of these packages. You can set up R to automatically load these packages every time you launch R if you like. (See Appendix A for details.)
>
> Other packages will be used from time to time as well. In this case, we will show the `require()` statement explicitly. The documentation for the `fastR` package includes a list of required and recommended packages.

```
+-------+----------+---+------------+
|       |          |N  |Sepal.Length|
+-------+----------+---+------------+
|Species|setosa    | 50|5.0060      |
|       |versicolor| 50|5.9360      |
|       |virginica | 50|6.5880      |
+-------+----------+---+------------+
|Overall|          |150|5.8433      |
+-------+----------+---+------------+
> summary(Sepal.Length~Species,iris,fun=median)  # median instead
Sepal.Length    N=150

+-------+----------+---+------------+
|       |          |N  |Sepal.Length|
+-------+----------+---+------------+
|Species|setosa    | 50|5.0         |
|       |versicolor| 50|5.9         |
|       |virginica | 50|6.5         |
+-------+----------+---+------------+
|Overall|          |150|5.8         |
+-------+----------+---+------------+
```

Comparing with the histograms in Figure 1.2, we see that these numbers are indeed good descriptions of the center of the distribution for each species.

We can also compute the mean and median of the Old Faithful eruption times.

faithful-mean-median

```
> mean(faithful$eruptions)
[1] 3.4878
> median(faithful$eruptions)
[1] 4
```

Notice, however, that in the Old Faithful eruption times histogram (Figure 1.5) there are very few eruptions that last between 3.5 and 4 minutes. So although these numbers are the mean and median, neither is a very good description of the typical eruption time(s) of Old Faithful. It will often be the case that the mean and median are not very good descriptions of a data set that is not unimodal.

faithful-stem

```
> stem(faithful$eruptions)

  The decimal point is 1 digit(s) to the left of the |

  16 | 070355555588
  18 | 000022233333335577777777888822335777888
  20 | 00002223378800035778
  22 | 0002335578023578
  24 | 00228
  26 | 23
  28 | 080
  30 | 7
  32 | 2337
  34 | 250077
  36 | 0000823577
  38 | 2333335582225577
  40 | 0000003357788888002233555577778
  42 | 03335555778800233333555577778
  44 | 02222335557780000000023333357778888
  46 | 0000233357700000023578
  48 | 00000022335800333
  50 | 0370
```

Figure 1.6. Stemplot of Old Faithful eruption times using `stem()`.

In the case of our Old Faithful data, there seem to be two predominant peaks, but unlike in the case of the iris data, we do not have another variable in our data that lets us partition the eruption times into two corresponding groups. This observation could, however, lead to some hypotheses about Old Faithful eruption times. Perhaps eruption times at night are different from those during the day. Perhaps there are other differences in the eruptions. Subsequent data collection (and statistical analysis of the resulting data) might help us determine whether our hypotheses appear correct.

One disadvantage of a histogram is that the actual data values are lost. For a large data set, this is probably unavoidable. But for more modestly sized data sets, a stemplot can reveal the shape of a distribution without losing the actual (perhaps rounded) data values. A stemplot divides each value into a stem and a leaf at some place value. The leaf is rounded so that it requires only a single digit. The values are then recorded as in Figure 1.6.

From this output we can readily see that the shortest recorded eruption time was 1.60 minutes. The second 0 in the first row represents 1.70 minutes. Note that the output of `stem()` can be ambiguous when there are not enough data values in a row.

Comparing mean and median

Why bother with two different measures of central tendency? The short answer is that they measure different things. If a distribution is (approximately) symmetric, the mean and median will be (approximately) the same (see Exercise 1.5). If the distribution is not symmetric, however, the mean and median may be very different, and one measure may provide a more useful summary than the other.

For example, if we begin with a symmetric distribution and add in one additional value that is very much larger than the other values (an **outlier**), then the median will not change very much (if at all), but the mean will increase substantially. We say that the median is **resistant** to outliers while the mean is not. A similar thing happens with a skewed, unimodal distribution. If a distribution is positively skewed, the large values in the tail of the distribution increase the mean (as compared to a symmetric distribution) but not the median, so the mean will be larger than the median. Similarly, the mean of a negatively skewed distribution will be smaller than the median.

Whether a resistant measure is desirable or not depends on context. If we are looking at the income of employees of a local business, the median may give us a much better indication of what a typical worker earns, since there may be a few large salaries (the business owner's, for example) that inflate the mean. This is also why the government reports median household income and median housing costs.

On the other hand, if we compare the median and mean of the value of raffle prizes, the mean is probably more interesting. The median is probably 0, since typically the majority of raffle tickets do not win anything. This is independent of the values of any of the prizes. The mean will tell us something about the overall value of the prizes involved. In particular, we might want to compare the mean prize value with the cost of the raffle ticket when we decide whether or not to purchase one.

The trimmed mean compromise

There is another measure of central tendency that is less well known and represents a kind of compromise between the mean and the median. In particular, it is more sensitive to the extreme values of a distribution than the median is, but less sensitive than the mean. The idea of a **trimmed mean** is very simple. Before calculating the mean, we remove the largest and smallest values from the data. The percentage of the data removed from each end is called the trimming percentage. A 0% trimmed mean is just the mean; a 50% trimmed mean is the median; a 10% trimmed mean is the mean of the middle 80% of the data (after removing the largest and smallest 10%). A trimmed mean is calculated in R by setting the `trim` argument of `mean()`, e.g., `mean(x,trim=0.10)`. Although a trimmed mean in some sense combines the advantages of both the mean and median, it is less common than either the mean or the median. This is partly due to the mathematical theory that has been developed for working with the median and especially the mean of sample data.

1.2.4. Measures of Dispersion

It is often useful to characterize a distribution in terms of its center, but that is not the whole story. Consider the distributions depicted in the histograms in Figure 1.7. In each case the mean and median are approximately 10, but the distributions clearly have very different shapes. The difference is that distribution B is much more "spread out". "Almost all" of the data in distribution A is quite close to 10; a much larger proportion of distribution B is "far away" from 10. The intuitive (and not very precise) statement in the preceding sentence can be quantified by means of **quantiles**. The idea of quantiles is probably familiar to you since **percentiles** are a special case of quantiles.

Definition 1.2.1 (Quantile). Let $p \in [0, 1]$. A p-**quantile** of a quantitative distribution is a number q such that the (approximate) proportion of the distribution that is less than q is p.

So, for example, the 0.2-quantile divides a distribution into 20% below and 80% above. This is the same as the 20th percentile. The median is the 0.5-quantile (and the 50th percentile).

The idea of a quantile is quite straightforward. In practice there are a few wrinkles to be ironed out. Suppose your data set has 15 values. What is the 0.30-quantile? Exactly 30% of the data would be $(0.30)(15) = 4.5$ values. Of course, there is no number that has 4.5 values below it and 11.5 values above it. This is the reason for the parenthetical word *approximate* in Definition 1.2.1. Different schemes have been proposed for giving quantiles a precise value, and R implements several such methods. They are similar in many ways to the decision we had to make when computing the median of a variable with an even number of values.

Two important methods can be described by imagining that the sorted data have been placed along a ruler, one value at every unit mark and also at each end. To find the p-quantile, we simply snap the ruler so that proportion p is to the left and $1-p$ to the right. If the break point happens to fall precisely where a data value is located (i.e., at one of the unit marks of our ruler), that value is the p-quantile. If the break point is between two data values, then the p-quantile is a weighted mean of those two values.

Example 1.2.1. Suppose we have 10 data values: $1, 4, 9, 16, 25, 36, 49, 64, 81, 100$. The 0-quantile is 1, the 1-quantile is 100, the 0.5-quantile (median) is midway between 25 and 36, that is, 30.5. Since our ruler is 9 units long, the 0.25-quantile is located $9/4 = 2.25$ units from the left edge. That would be one quarter of the way from 9 to 16, which is $9 + 0.25(16 - 9) = 9 + 1.75 = 10.75$. (See Figure 1.8.) Other quantiles are found similarly. This is precisely the default method used by `quantile()`.

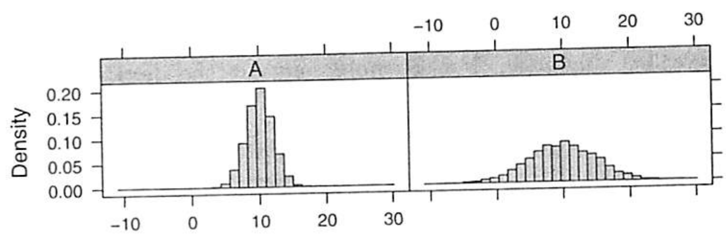


Figure 1.7. Histograms showing smaller (A) and larger (B) amounts of variation.

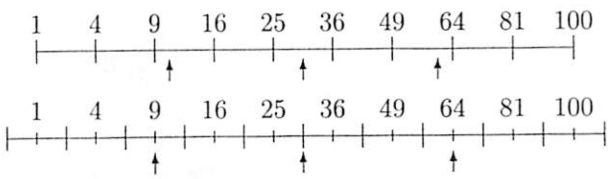


Figure 1.8. Illustrations of two methods for determining quantiles from data. Arrows indicate the locations of the 0.25-, 0.5-, and 0.75-quantiles.

intro-quantile

```
> quantile((1:10)^2)
    0%    25%    50%    75%   100%
  1.00  10.75  30.50  60.25 100.00
```

◁

A second scheme is just like the first one except that the data values are placed midway between the unit marks. In particular, this means that the 0-quantile is not the smallest value. This could be useful, for example, if we imagined we were trying to estimate the lowest value in a population from which we only had a sample. Probably the lowest value overall is less than the lowest value in our particular sample. The only remaining question is how to extrapolate in the last half unit on either side of the ruler. If we set quantiles in that range to be the minimum or maximum, the result is another type of `quantile()`.

Example 1.2.2. The method just described is what `type=5` does.

intro-quantile05a

```
> quantile((1:10)^2,type=5)
   0%   25%   50%   75%  100%
  1.0   9.0  30.5  64.0 100.0
```

Notice that quantiles below the 0.05-quantile are all equal to the minimum value.

intro-quantile05b

```
> quantile((1:10)^2,type=5,seq(0,0.10,by=0.005))
  0% 0.5%   1% 1.5%   2% 2.5%   3% 3.5%   4% 4.5%   5% 5.5%   6% 6.5%
1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.15 1.30 1.45
  7% 7.5%   8% 8.5%   9% 9.5%  10%
1.60 1.75 1.90 2.05 2.20 2.35 2.50
```

A similar thing happens with the maximum value for the larger quantiles. ◁

Other methods refine this idea in other ways, usually based on some assumptions about what the population of interest is like.

Fortunately, for large data sets, the differences between the different quantile methods are usually unimportant, so we will just let R compute quantiles for us using the `quantile()` function. For example, here are the **deciles** and **quartiles** of the Old Faithful eruption times.

faithful-quantile

```
> quantile(faithful$eruptions,(0:10)/10)
    0%    10%    20%    30%    40%    50%    60%    70%    80%    90%
1.6000 1.8517 2.0034 2.3051 3.6000 4.0000 4.1670 4.3667 4.5330 4.7000
  100%
5.1000
> quantile(faithful$eruptions,(0:4)/4)
```

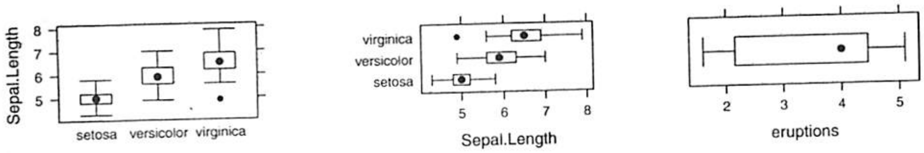


Figure 1.9. Boxplots for iris sepal length and Old Faithful eruption times.

```
    0%     25%     50%     75%    100%
1.6000 2.1627 4.0000 4.4543 5.1000
```

The latter of these provides what is commonly called the **five-number summary.** The 0-quantile and 1-quantile (at least in the default scheme) are the minimum and maximum of the data set. The 0.5-quantile gives the median, and the 0.25- and 0.75-quantiles (also called the first and third quartiles) isolate the middle 50% of the data. When these numbers are close together, then most (well, half, to be more precise) of the values are near the median. If those numbers are farther apart, then much (again, half) of the data is far from the center. The difference between the first and third quartiles is called the **interquartile range** and is abbreviated **IQR**. This is our first numerical measure of dispersion.

The five-number summary can also be presented graphically using a **boxplot** (also called box-and-whisker plot) as in Figure 1.9. These plots were generated using

iris-bwplot

```
> bwplot(Sepal.Length~Species,data=iris)
> bwplot(Species~Sepal.Length,data=iris)
> bwplot(~eruptions,faithful)
```

The size of the box reflects the IQR. If the box is small, then the middle 50% of the data are near the median, which is indicated by a dot in these plots. (Some boxplots, including those made by the `boxplot()` use a vertical line to indicate the median.) Outliers (values that seem unusually large or small) can be indicated by a special symbol. The whiskers are then drawn from the box to the largest and smallest non-outliers. One common rule for automating outlier detection for boxplots is the 1.5 **IQR rule**. Under this rule, any value that is more than 1.5 IQR away from the box is marked as an outlier. Indicating outliers in this way is useful since it allows us to see if the whisker is long only because of one extreme value.

Variance and standard deviation

Another important way to measure the dispersion of a distribution is by comparing each value with the mean of the distribution. If the distribution is spread out, these differences will tend to be large; otherwise these differences will be small. To get a single number, we could simply add up all of the **deviation from the mean**:

$$\text{total deviation from the mean} = \sum (x - \overline{x}) \,.$$

The trouble with this is that the total deviation from the mean is always 0 because the negative deviations and the positive deviations always exactly cancel out. (See Exercise 1.10).

To fix this problem, we might consider taking the absolute value of the deviations from the mean:

$$\text{total absolute deviation from the mean} = \sum |x - \overline{x}| \; .$$

This number will only be 0 if all of the data values are equal to the mean. Even better would be to divide by the number of data values:

$$\text{mean absolute deviation} = \frac{1}{n} \sum |x - \overline{x}| \; .$$

Otherwise large data sets will have large sums even if the values are all close to the mean. The mean absolute deviation is a reasonable measure of the dispersion in a distribution, but we will not use it very often. There is another measure that is much more common, namely the **variance**, which is defined by

$$\text{variance} = \operatorname{Var}(x) = \frac{1}{n-1} \sum (x - \overline{x})^2 \; .$$

You will notice two differences from the mean absolute deviation. First, instead of using an absolute value to make things positive, we square the deviations from the mean. The chief advantage of squaring over the absolute value is that it is much easier to do calculus with a polynomial than with functions involving absolute values. Because the squaring changes the units of this measure, the square root of the variance, called the **standard deviation**, is commonly used in place of the variance.

The second difference is that we divide by $n - 1$ instead of by n. There is a very good reason for this, even though dividing by n probably would have felt much more natural to you at this point. We'll get to that very good reason later in the course (in Section 4.6). For now, we'll settle for a less good reason. If you know the mean and all but one of the values of a variable, then you can determine the remaining value, since the sum of all the values must be the product of the number of values and the mean. So once the mean is known, there are only $n - 1$ independent pieces of information remaining. That is not a particularly satisfying explanation, but it should help you remember to divide by the correct quantity.

All of these quantities are easy to compute in R.

intro-dispersion02

```
> x=c(1,3,5,5,6,8,9,14,14,20)
>
> mean(x)
[1] 8.5
> x - mean(x)
 [1] -7.5 -5.5 -3.5 -3.5 -2.5 -0.5  0.5  5.5  5.5 11.5
> sum(x - mean(x))
[1] 0
> abs(x - mean(x))
 [1]  7.5  5.5  3.5  3.5  2.5  0.5  0.5  5.5  5.5 11.5
> sum(abs(x - mean(x)))
[1] 46
```

```
> (x - mean(x))^2
 [1]  56.25  30.25  12.25  12.25   6.25   0.25   0.25  30.25  30.25
[10] 132.25
> sum((x - mean(x))^2)
[1] 310.5
> n= length(x)
> 1/(n-1) * sum((x - mean(x))^2)
[1] 34.5
> var(x)
[1] 34.5
> sd(x)
[1] 5.8737
> sd(x)^2
[1] 34.5
```

1.3. Graphical and Numerical Summaries of Multivariate Data

1.3.1. Side-by-Side Comparisons

Often it is useful to consider two or more variables together. In fact, we have already done some of this. For example, we looked at iris sepal length separated by species. This sort of side-by-side comparison – in graphical or tabular form – is especially useful when one variable is quantitative and the other categorical. Graphical or numerical summaries of the quantitative variable can be made separately for each group defined by the categorical variable (or by shingles of a second quantitative variable). See Appendix A for more examples.

1.3.2. Scatterplots

There is another plot that is useful for looking at the relationship between two quantitative variables. A **scatterplot** (or scattergram) is essentially the familiar Cartesian coordinate plot you learned about in school. Since each observation in a bivariate data set has two values, we can plot points on a rectangular grid representing both values simultaneously. The `lattice` function for making a scatterplot is `xyplot()`.

The scatterplot in Figure 1.10 becomes even more informative if we separate the dots of the three species. Figure 1.11 shows two ways this can be done. The first uses a conditioning variable, as we have seen before, to make separate panels for each species. The second uses the `groups` argument to plot the data in the same panel but with different symbols for each species. Each of these clearly indicates that, in general, plants with wider sepals also have longer sepals but that the typical values of and the relationship between width and length differ by species.

1.3.3. Two-Way Tables and Mosaic Plots

A 1981 paper [**Rad81**] investigating racial biases in the application of the death penalty reported on 326 cases in which the defendant was convicted of murder.

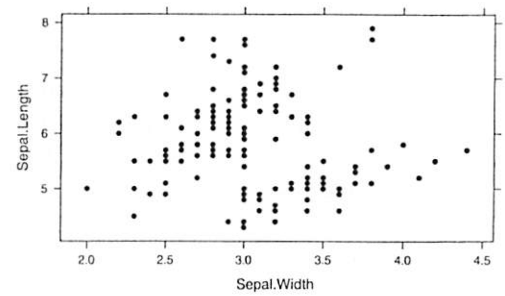


Figure 1.10. A scatterplot made with `xyplot(Sepal.Length~Sepal.Width,iris)`.

For each case they noted the race of the defendant and whether or not the death penalty was imposed. We can use R to cross tabulate this data for us:

intro-deathPenalty01

```
> xtabs(~Penalty+Victim,data=deathPenalty)
       Victim
Penalty Black White
  Death     6    30
  Not     106   184
```

Perhaps you are surprised that white defendants are more likely to receive the death penalty. It turns out that there is more to the story. The researchers also recorded the race of the victim. If we make a new table that includes this information, we see something interesting.

intro-deathPenalty02

```
> xtabs(~Penalty+Defendant+Victim,
+          data=deathPenalty)
, , Victim = Black

       Defendant
Penalty Black White
  Death     6     0
  Not      97     9
```

```
, , Victim = White

       Defendant
Penalty Black White
  Death    11    19
  Not      52   132
```

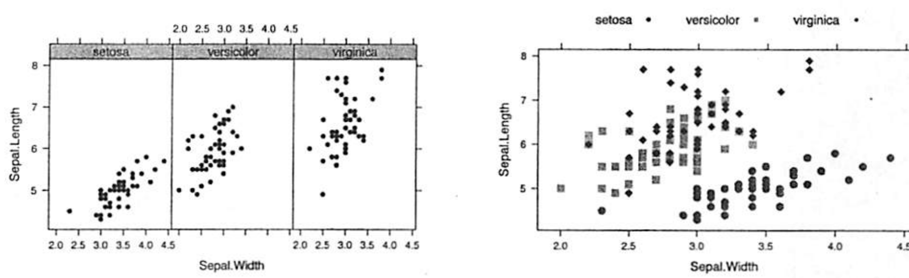


Figure 1.11. The output of `xyplot(Sepal.Length~Sepal.Width|Species,iris)` and `xyplot(Sepal.Length~Sepal.Width,groups=Species,iris,auto.key=TRUE)`.

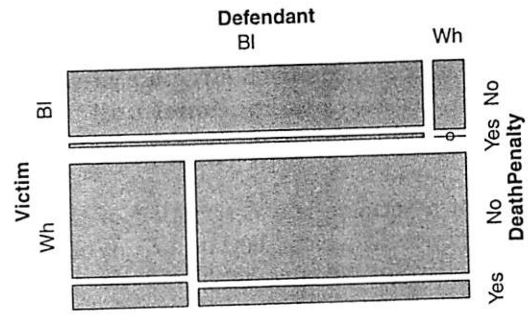


Figure 1.12. A mosaic plot of death penalty by race of defendant and victim.

It appears that black defendants are more likely to receive the death penalty when the victim is white and also when the victim is black, but not if we ignore the race of the victim. This sort of apparent contradiction is known as **Simpson's paradox.** In this case, it appears that the death penalty is more likely to be given for a white victim, and since most victims are the same race as their murderer, the result is that overall white defendants are more likely (in this data set) to receive the death penalty even though black defendants are more likely (again, in this data set) to receive the death penalty for each race of victim.

The fact that our understanding of the data is so dramatically influenced by whether or not our analysis includes the race of the victim is a warning to watch for **lurking variables** – variables that have an important effect but are not included in our analysis – in other settings as well. Part of the design of a good study is selecting the right things to measure.

These cross tables can be visualized graphically using a **mosaic plot**. Mosaic plots can be generated with the core R function `mosaicplot()` or with `mosaic()` from the `vcd` package. (`vcd` is short for visualization of categorical data.) The latter is somewhat more flexible and usually produces more esthetically pleasing output. A number of different formula formats can be supplied to `mosaic()`. The results of the following code are shown in Figure 1.12.

intro-deathPenalty03

```
> require(vcd)
> mosaic(~Victim+Defendant+DeathPenalty,data=deathPen)
> structable(~Victim+Defendant+DeathPenalty,data=deathPen)
                     Defendant  Bl  Wh
Victim DeathPenalty
Bl     No                       97   9
       Yes                       6   0
Wh     No                       52 132
       Yes                      11  19
```

As always, see `?mosaic` for more information. The `vcd` package also provides an alternative to `xtabs()` called `structable()`, and if you `print()` a `mosaic()`, you will get both the graph and the table.

1.4. Summary

Data can be thought of in a 2-dimensional structure in which each **variable** has a value (possibly missing) for each **observational unit**. In most statistical software, including R, columns correspond to variables and rows correspond to the observations.

The **distribution** of a variable is a description of the values obtained by a variable and the frequency with which they occur. While simply listing all the values does describe the distribution completely, it is not easy to draw conclusions from this sort of description, especially when the number of observational units is large. Instead, we will make frequent use of numerical and graphical summaries that make it easier to see what is going on and to make comparisons.

The **mean**, **median**, **standard deviation**, and **interquartile range** are among the most common numerical summaries. The mean and median give an indication of the "center" of the distribution. They are especially useful for **unimodal** distributions but may not be appropriate summaries for distributions with other shapes. When a distribution is **skewed**, the mean and median can be quite different because the extreme values of the distribution have a large effect on the mean but not on the median. A **trimmed mean** is sometimes used as a compromise between the median and the mean. Although one could imagine other measures of spread, the standard deviation is especially important because of its relationship to important theoretical results in statistics, especially the Central Limit Theorem, which we will encounter in Chapter 4.

Even as we learn formal methods of statistical analysis, we will not abandon these numerical and graphical summaries. Appendix A provides a more complete introduction to R and includes information on how to fine-tune plots. Additional examples can be found throughout the text.

1.4.1. R Commands

Here is a table of important R commands introduced in this chapter. Usage details can be found in the examples and using the R help.

`x <- c(...)`	Concatenate arguments into a single vector and store in object `x`.
`data(x)`	(Re)load the data set `x`.
`str(x)`	Print a summary of the object `x`.
`head(x,n=4)`	First four rows of the data frame `x`.
`tail(x,n=4)`	Last four rows of the data frame `x`.
`table(x)`	Table of the values in vector `x`.
`xtabs(~x+y,data)`	Cross tabulation of `x` and `y`.

`cut(x,breaks,right=TRUE)`	Divide up the range of x into intervals and code the values in x according to which interval they fall into.
`require(fastR);` `require(lattice);` `require(Hmisc)`	Load packages.
`histogram(~x|z,data,...)`	Histogram of x conditioned on z.
`bwplot(x~z,data,...)`	Boxplot of x conditioned on z.
`xyplot(y~x|z,data,...)`	Scatterplot of y by x conditioned on z.
`stem(x)`	Stemplot of x.
`sum(x); mean(x); median(x);` `var(x); sd(x); quantile(x)`	Sum, mean, median, variance, standard deviation, quantiles of x.
`summary(y~x,data,fun)`	Summarize y by computing the function `fun` on each group defined by x [Hmisc].

Exercises

1.1. Read as much of Appendix A as you need to do the exercises there.

1.2. The `pulse` variable in the `littleSurvey` data set contains self-reported pulse rates.

a) Make a histogram of these values. What problem does this histogram reveal?
b) Make a decision about what values should be removed from the data and make a histogram of the remaining values. (You can use the `subset` argument of the `histogram()` function to restrict the data or you can create a new vector and make a histogram from that.)
c) Compute the mean and median of your restricted set of pulse rates.

1.3. The `pulse` variable in the `littleSurvey` data set contains self-reported pulse rates. Make a table or graph showing the distribution of the last digits of the recorded pulse rates and comment on the distribution of these digits. Any conjectures?

Note: `%%` is the modulus operator in R. So `x %% 10` gives the remainder after dividing x by 10, which is the last digit.

1.4. Some students in introductory statistics courses were asked to select a number between 1 and 30 (inclusive). The results are in the `number` variable in the `littleSurvey` data set.

a) Make a table showing the frequency with which each number was selected using `table()`.

b) Make a histogram of these values with bins centered at the integers from 1 to 30.

c) What numbers were most frequently chosen? Can you get R to find them for you?

d) What numbers were least frequently chosen? Can you get R to find them for you?

e) Make a table showing how many students selected odd versus even numbers.

1.5. The distribution of a quantitative variable is **symmetric about** m if whenever there are k observations with value $m + d$, there are also k observations with value $m - d$. Equivalently, if the values are $x_1 \le x_2 \le \cdots \le x_n$, then $x_i + x_{n+1-i} = 2m$ for all i.

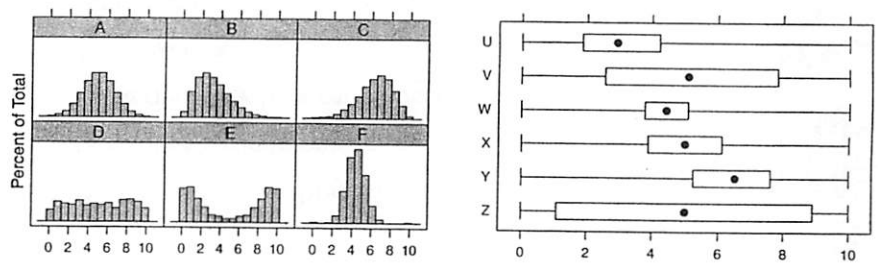
a) Show that if a distribution is symmetric about m, then m is the median. (You may need to handle separately the cases where the number of values is odd and even.)

b) Show that if a distribution is symmetric about m, then m is the mean.

c) Create a small distribution such that the mean and median are equal to m but the distribution is not symmetric about m.

1.6. Describe some situations where the mean or median is clearly a better measure of central tendency than the other.

1.7. Below are histograms and boxplots from six distributions. Match each histogram (A–F) with its corresponding boxplot (U–Z).



1.8. The function `bwplot()` does not use the `quantile()` function to compute its five-number summary. Instead it uses `fivenum()`. Technically, `fivenum()` computes the **hinges** of the data rather than quantiles. Sometimes `fivenum()` and `quantile()` agree:

fivenum-a

```
> fivenum(1:11)
[1]  1.0  3.5  6.0  8.5 11.0
> quantile(1:11)
  0%  25%  50%  75% 100%
 1.0  3.5  6.0  8.5 11.0
```

But sometimes they do not:

fivenum-b

```
> fivenum(1:10)
[1]  1.0  3.0  5.5  8.0 10.0
> quantile(1:10)
   0%   25%   50%   75%  100%
 1.00  3.25  5.50  7.75 10.00
```

Compute `fivenum()` on a number of data sets and answer the following questions:

a) When does `fivenum()` give the same values as `quantile()`?

b) What method is `fivenum()` using to compute the five numbers?

1.9. Design some data sets to test whether by default `bwplot()` uses the 1.5 IQR rule to determine whether it should indicate data as outliers.

1.10. Show that the total deviation from the mean, defined by

$$\text{total deviation from the mean} = \sum_{i=1}^{n}(x_i - \overline{x}) ,$$

is 0 for any distribution.

1.11. We could compute the mean absolute deviation from the *median* instead of from the mean. Show that the mean absolute deviation from the median is never larger than the mean absolute deviation from the mean.

1.12. We could compute the mean absolute deviation from any number c (c for center). Show that the mean absolute deviation from c is always at least as large as the mean absolute deviation from the median. Thus the median is a minimizer of mean absolute deviation.

1.13. Let $SS(c) = \sum(x_i - c)^2$. (SS stands for sum of squares.) Show that the smallest value of $SS(c)$ occurs when $c = \overline{x}$. This shows that the mean is a minimizer of SS.

1.14. Find a distribution with 10 values between 0 and 10 that has as large a variance as possible.

1.15. Find a distribution with 10 values between 0 and 10 that has as small a variance as possible.

1.16. The `pitching2005` data set in the `fastR` package contains 2005 season statistics for each pitcher in the major leagues. Use graphical and numerical summaries of this data set to explore whether there are differences between the two leagues, restricting your attention to pitchers that started at least 5 games (the variable `GS` stands for 'games started'). You may select the statistics that are of interest to you.

If you are not much of a baseball fan, try using `ERA` (earned run average), which is a measure of how many runs score while a pitcher is pitching. It is measured in runs per nine innings.

1.17. Repeat the previous problem using batting statistics. The `fastR` data set `batting` contains data on major league batters over a large number of years. You may want to restrict your attention to a particular year or set of years.

1.18. Have major league batting averages changed over time? If so, in what ways? Use the data in the `batting` data set to explore this question. Use graphical and numerical summaries to make your case one way or the other.

1.19. The `faithful` data set contains two variables: the duration (`eruptions`) of the eruption and the time until the next eruption (`waiting`).

a) Make a scatterplot of these two variables and comment on any patterns you see.

b) Remove the first value of `eruptions` and the last value of `waiting`. Make a scatterplot of these two vectors.

c) Which of the two scatterplots reveals a tighter relationship? What does that say about the relationship between eruption duration and the interval between eruptions?

1.20. The results of a little survey that has been given to a number of statistics students are available in the `littleSurvey` data set. Make some conjectures about the responses and use R's graphical and numerical summaries to see if there is any (informal) evidence to support your conjectures. See `?littleSurvey` for details about the questions on the survey.

1.21. The `utilities` data set contains information from utilities bills for a personal residence over a number of years. This problem explores gas usage over time.

a) Make a scatterplot of gas usage (`ccf`) vs. time. You will need to combine `month` and `year` to get a reasonable measurement for time. Such a plot is called a **time series plot**.

b) Use the `groups` argument (and perhaps `type=c('p','l')`, too) to make the different months of the year distinguishable in your scatterplot.

c) Now make a boxplot of gas usage (`ccf`) vs. `factor(month)`. Which months are most variable? Which are most consistent?

d) What patterns do you see in the data? Does there appear to be any change in gas usage over time? Which plots help you come to your conclusion?

1.22. Note that March and May of 2000 are outliers due to a bad meter reading. Utility bills come monthly, but the number of days in a billing cycle varies from month to month. Add a new variable to the `utilities` data set using

utilities-ccfpday

```
> utilities$ccfpday <- utilities$ccf / utilities$billingDays
> plot1 <- xyplot( ccfpday ~ (year + month/12), utilities, groups=month )
> plot2 <- bwplot( ccfpday ~ factor(month), utilities )
```

Repeat the previous exercise using `ccfpday` instead of `ccf`. Are there any noticeable differences between the two analyses?

1.23. The `utilities` data set contains information from utilities bills for a personal residence over a number of years. One would expect that the gas bill would be related to the average temperature for the month.

Make a scatterplot showing the relationship between `ccf` (or, better, `ccfpday`; see Exercise 1.22) and `temp`. Describe the overall pattern. Are there any outliers?

1.24. The `utilities` data set contains information from utilities bills for a personal residence over a number of years. The variables `gasbill` and `ccf` contain the gas bill (in dollars) and usage (in 100 cubic feet) for a personal residence. Use plots to explore the cost of gas over the time period covered in the `utilities` data set. Look for both seasonal variation in price and any trends over time.

1.25. The `births78` data set contains the number of births in the United States for each day of 1978.

a) Make a histogram of the number of births. You may be surprised by the shape of the distribution. (Make a stemplot too if you like.)

b) Now make a scatterplot of births vs. day of the year. What do you notice? Can you conjecture any reasons for this?

c) Can you make a plot that will help you see if your conjecture seems correct? (Hint: Use `groups`.)