Chapter 1

Summarizing and Displaying Data

Learning objectives

After completing this topic, you should be able to:

use R's functions to get help and numerically summarize data.

apply R's base graphics and ggplot to visually summarize data in several ways.

explain what each plotting option does.

describe the characteristics of a data distribution.

Achieving these goals contributes to mastery in these course learning outcomes:

1. organize knowledge.

6. summarize data visually, numerically, and descriptively.

8. use statistical software.

1.1 Random variables

A **random variable** is a variable whose value is subject to variations due to chance. Random variables fall into two broad categories: qualitative and quantitative.

Qualitative data includes categorical outcomes:

Nominal Outcome is one of several categories.
Ex: Blood group, hair color.

Ordinal Outcome is one of several *ordered* categories.
Ex: Likert data such as (strongly agree, agree, neutral, disagree, strongly disagree).

Quantitative data includes numeric outcomes:

Discrete Outcome is one of a fixed set of numerical values.
Ex: Number of children.

Continuous Outcome is any numerical value.
Ex: Birthweight.

It may not always be perfectly clear which type data belong to, and may sometimes be classified based on the question being asked of the data. Distinction between nominal and ordinal variables can be subjective. For example, for vertebral fracture types: (Wedge, Concavity, Biconcavity, Crush), one could argue that a crush is worse than a biconcavity which is worse than a concavity . . . , but this is not self-evident. Distinction between ordinal and discrete variables can be subjective. For example, cancer staging (I, II, III, IV) sounds discrete, but better treated as ordinal because the "distance" between stages may be hard to define and unlikely to be equal. Continuous variables generally measured to a fixed level of precision, which makes them discrete. This "discreteness" of continuous variables is not a problem, providing there are enough levels.

■ ClickerQs — Random variables ■

1.2 Numerical summaries

Suppose we have a collection of n individuals, and we measure each individual's response on one quantitative characteristic, say height, weight, or systolic blood pressure. For notational simplicity, the collected measurements are denoted by $Y_1, Y_2, \ldots, Y_n$, where n is the **sample size**. The order in which the measurements are assigned to the place-holders $(Y_1, Y_2, \ldots, Y_n)$ is irrelevant.

Among the numerical summary measures we're interested in are the **sample mean** $\bar{Y}$ and the **sample standard deviation** s. The sample mean is a measure of **central location**, or a measure of a "typical value" for the data set. The standard deviation is a measure of **spread** in the data set. These summary statistics should be familiar to you. Let us consider a simple example to refresh your memory on how to compute them.

Suppose we have a sample of $n = 8$ children with weights (in pounds): 5, 9, 12, 30, 14, 18, 32, 40. Then

$$\begin{aligned}
\bar{Y} &= \frac{\sum_i Y_i}{n} = \frac{Y_1 + Y_2 + \cdots + Y_n}{n} \\
&= \frac{5 + 9 + 12 + 30 + 14 + 18 + 32 + 40}{8} = \frac{160}{8} = 20.
\end{aligned}$$

```
#### Numerical summaries
#### mean
y <- c(5, 9, 12, 30, 14, 18, 32, 40)
mean(y)
## [1] 20
```

The sample standard deviation is the square root of the sample variance

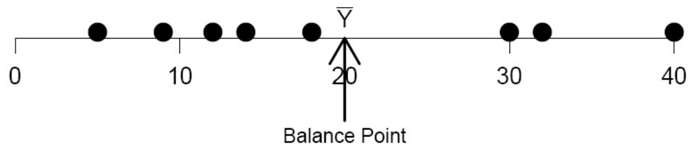
$$\begin{aligned}
s^2 &= \frac{\sum_i (Y_i - \bar{Y})^2}{n-1} = \frac{(Y_1 - \bar{Y})^2 + (Y_2 - \bar{Y})^2 + \cdots + (Y_k - \bar{Y})^2}{n-1} \\
&= \frac{(5-20)^2 + (9-20)^2 + \cdots + (40-20)^2}{7} = 156.3, \\
s &= \sqrt{s^2} = 12.5.
\end{aligned}$$

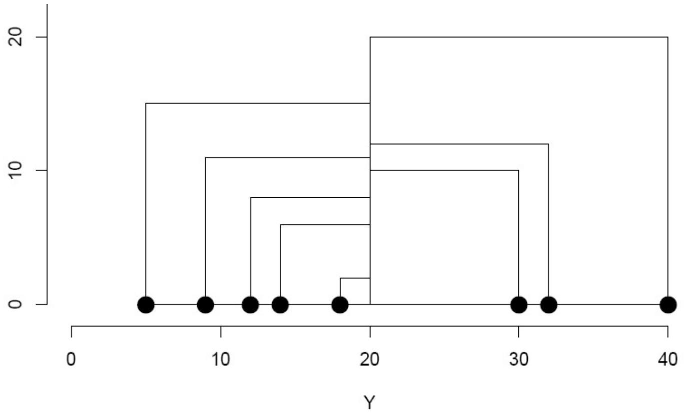
```
#### variance
var(y)
## [1] 156.2857
sd(y)
## [1] 12.50143
```

Summary statistics have well-defined units of measurement, for example, $\bar{Y} = 20$ lb, $s^2 = 156.3$ lb^2, and $s = 12.5$ lb. The standard deviation is often used instead of s^2 as a measure of spread because s is measured in the same units as the data.

Remark If the divisor for s^2 was n instead of $n-1$, then the variance would be the average squared deviation observations are from the center of the data as measured by the mean.

The following graphs should help you to see some physical meaning of the sample mean and variance. If the data values were placed on a "massless" ruler, the balance point would be the mean (20). The variance is basically the "average" (remember $n-1$ instead of n) of the total areas of all the squares obtained when squares are formed by joining each value to the mean. In both cases think about the implication of unusual values (**outliers**). What happens to the balance point if the 40 were a 400 instead of a 40? What happens to the squares?





The **sample median** M is an alternative measure of central location. The measure of spread reported along with M is the **interquartile range**, $IQR = Q_3 - Q_1$, where Q_1 and Q_3 are the first and third quartiles of the data set, respectively. To calculate the median and interquartile range, order the data from lowest to highest values, all repeated values included. The ordered weights are

$$5 \;\; 9 \;\; 12 \;\; 14 \;\; 18 \;\; 30 \;\; 32 \;\; 40.$$

```
#### sorting
sort(y)

## [1]  5  9 12 14 18 30 32 40
```

The median M is the value located at the half-way point of the ordered string. There is an even number of observations, so M is defined to be half-way between the two middle values, 14 and 18. That is, $M = 0.5(14 + 18) = 16$ lb. To get the quartiles, break the data into the lower half: 5 9 12 14, and the upper half: 18 30 32 and 40. Then

$Q_1 =$ first quartile $=$ median of lower half of data $= 0.5(9{+}12){=}10.5$ lb,

and

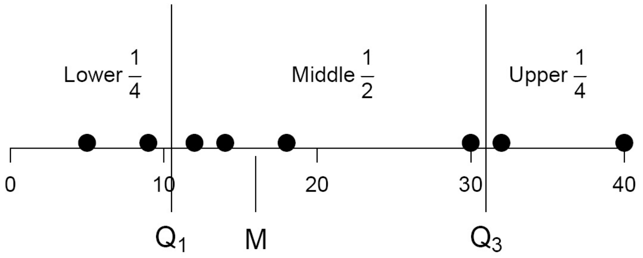
$Q_3 =$ third quartile $=$ median of upper half of data $= 0.5(30{+}32) = 31$ lb.

The interquartile range is

$$IQR = Q_3 - Q_1 = 31 - 10.5 = 20.5 \text{ lb.}$$

```
#### quartiles
median(y)

## [1] 16

fivenum(y)

## [1]  5.0 10.5 16.0 31.0 40.0

# The quantile() function can be useful, but doesn't calculate Q1 and Q3
#   as defined above, regardless of the 9 types of calculations for them!
# summary() is a combination of mean() and quantile(y, c(0, 0.25, 0.5, 0.75, 1))
summary(y)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00   11.25   16.00   20.00   30.50   40.00

# IQR
fivenum(y)[c(2,4)]

## [1] 10.5 31.0

fivenum(y)[4] - fivenum(y)[2]

## [1] 20.5

diff(fivenum(y)[c(2,4)])

## [1] 20.5
```

The quartiles, with M being the second quartile, break the data set roughly into fourths. The first quartile is also called the 25th percentile, whereas the median and third quartiles are the 50th and 75th percentiles, respectively. The IQR is the **range** for the middle half of the data.



Suppose we omit the largest observation from the weight data:

$$5 \;\; 9 \;\; 12 \;\; 14 \;\; 18 \;\; 30 \;\; 32.$$

How do M and IQR change? With an odd number of observations, there is a unique middle observation in the ordered string which is M. Here $M = 14$ lb. It is unclear which half the median should fall into, so M is placed into both the lower and upper halves of the data. The lower half is 5 9 12 14, and the upper half is 14 18 30 32. With this convention, $Q_1 = 0.5(9 + 12) = 10.5$ and $Q_3 = 0.5(18 + 30) = 24$, giving $IQR = 24 - 10.5 = 13.5$ (lb).

```
#### remove largest
# remove the largest observation by removing the last of the sorted values
y2 <- sort(y)[-length(y)]
y2

## [1]  5  9 12 14 18 30 32

median(y2)

## [1] 14

fivenum(y2)

## [1]  5.0 10.5 14.0 24.0 32.0

diff(fivenum(y2)[c(2,4)])

## [1] 13.5
```

If you look at the data set with all eight observations, there actually are many numbers that split the data set in half, so the median is not uniquely defined[1], although "everybody" agrees to use the average of the two middle values. With quartiles there is the same ambiguity but no such universal agreement on what to do about it, however, so R will give slightly different values for Q_1 and Q_3 when using `summary()` and some other commands than we just calculated,

[1] The technical definition of the median for an even set of values includes the entire range between the two center values. Thus, selecting any single value in this center range is convenient and the center of this center range is one sensible choice for the median, M.

and other packages will report even different values. This has no practical implication (all the values are "correct") but it can appear confusing.

Example The data given below are the head breadths in mm for a sample of 18 modern Englishmen, with numerical summaries generated by R.

```
#### Englishmen
hb <- c(141, 148, 132, 138, 154, 142, 150, 146, 155
      , 158, 150, 140, 147, 148, 144, 150, 149, 145)

# see sorted values
sort(hb)

##  [1] 132 138 140 141 142 144 145 146 147 148 148 149 150 150 150 154
## [17] 155 158

# number of observations is the length of the vector (when no missing values)
length(hb)

## [1] 18

# default quartiles
summary(hb)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   132.0   142.5   147.5   146.5   150.0   158.0

# standard quartiles
fivenum(hb)

## [1] 132.0 142.0 147.5 150.0 158.0

# range() gives the min and max values
range(hb)

## [1] 132 158

# the range of the data is the (max - min), calculated using diff()
diff(range(hb))

## [1] 26

mean(hb)

## [1] 146.5

# standard deviation
sd(hb)

## [1] 6.382421

# standard error of the mean
se <- sd(hb)/sqrt(length(hb))
```

Note that `se` is the standard error of the sample mean, $SE_{\bar{Y}} = s/\sqrt{n}$, and is a measure of the precision of the sample mean $\bar{Y}$.

■ ClickerQs — Numerical summaries ■

1.3 Graphical summaries for one quantitative sample

There are four graphical summaries of primary interest: the **dotplot**, the **histogram**, the **stem-and-leaf** display, and the **boxplot**. There are many more possible, but these will often be useful. The plots can be customized. Make liberal use of the help for learning how to customize them. Plots can also be generated along with many statistical analyses, a point that we will return to repeatedly.

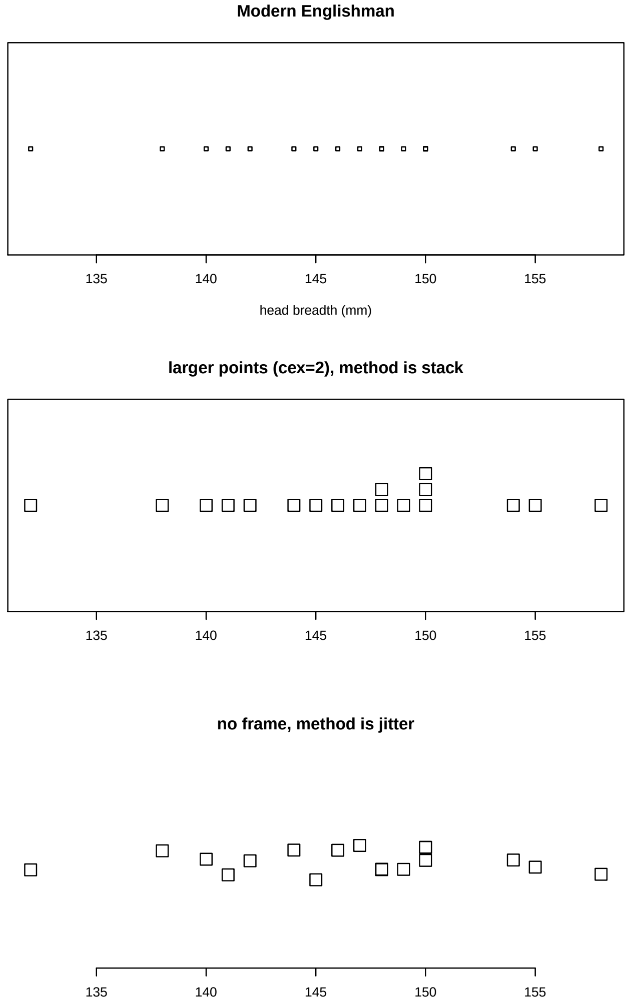
1.3.1 Dotplots

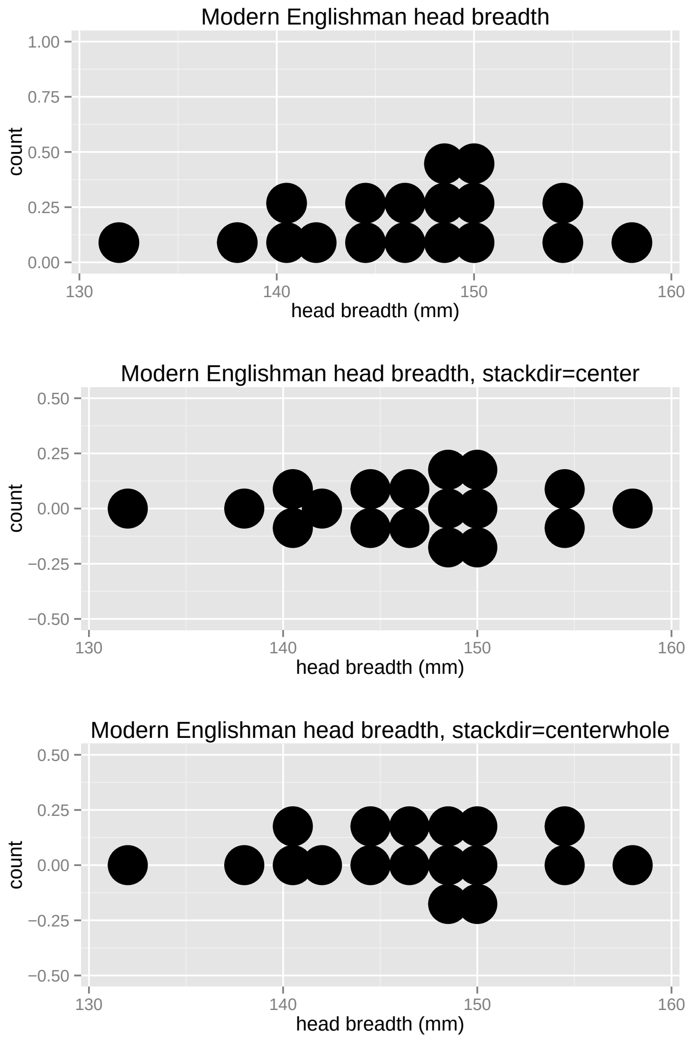
The **dotplot** breaks the range of data into many small-equal width intervals, and counts the number of observations in each interval. The interval count is superimposed on the number line at the interval midpoint as a series of dots, usually one for each observation. In the head breadth data, the intervals are centered at integer values, so the display gives the number of observations at each distinct observed head breadth.

A dotplot of the head breadth data is given below. Of the examples below, the R base graphics `stripchart()` with `method="stack"` resembles the traditional dotplot.

```
#### stripchart-ggplot
# stripchart (dotplot) using R base graphics
# 3 rows, 1 column
par(mfrow=c(3,1))
stripchart(hb, main="Modern Englishman", xlab="head breadth (mm)")
stripchart(hb, method="stack", cex=2
  , main="larger points (cex=2), method is stack")
stripchart(hb, method="jitter", cex=2, frame.plot=FALSE
  , main="no frame, method is jitter")
```

```r
# dotplot using ggplot
library(ggplot2)
# first put hb vector into a data.frame
hb_df <- data.frame(hb)
p1 <- ggplot(hb_df, aes(x = hb))
p1 <- p1 + geom_dotplot(binwidth = 2)
p1 <- p1 + labs(title = "Modern Englishman head breadth")
p1 <- p1 + xlab("head breadth (mm)")

p2 <- ggplot(hb_df, aes(x = hb))
p2 <- p2 + geom_dotplot(binwidth = 2, stackdir = "center")
p2 <- p2 + labs(title = "Modern Englishman head breadth, stackdir=center")
p2 <- p2 + xlab("head breadth (mm)")

p3 <- ggplot(hb_df, aes(x = hb))
p3 <- p3 + geom_dotplot(binwidth = 2, stackdir = "centerwhole")
p3 <- p3 + labs(title = "Modern Englishman head breadth, stackdir=centerwhole")
p3 <- p3 + xlab("head breadth (mm)")

library(gridExtra)
grid.arrange(p1, p2, p3, ncol=1)
```



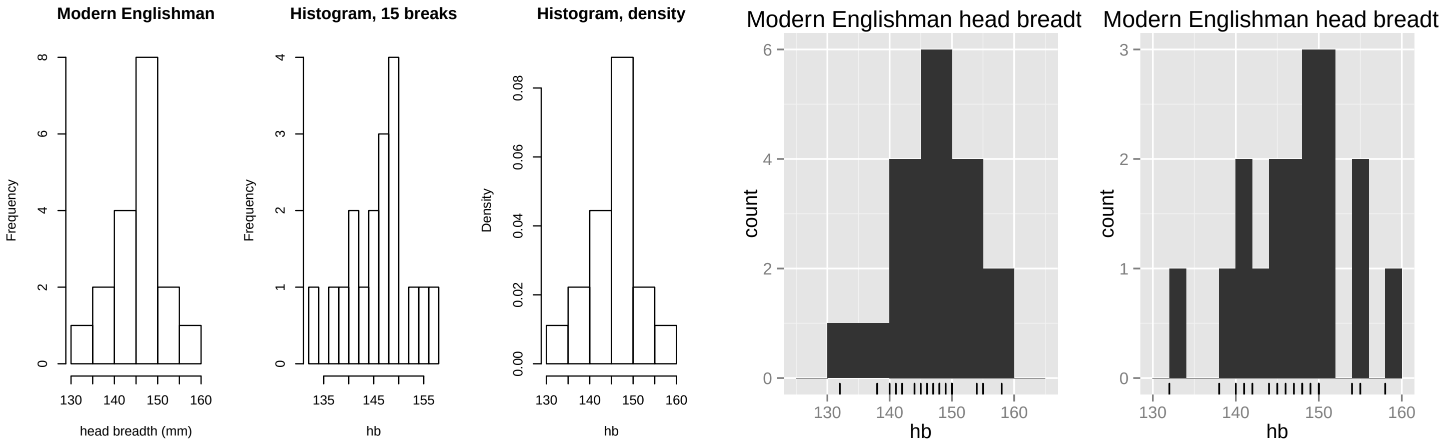


1.3.2 Histogram

The **histogram** and **stem-and-leaf** displays are similar, breaking the range of data into a smaller number of equal-width intervals. This produces graphical information about the observed distribution by highlighting where data values cluster. The histogram can use arbitrary intervals, whereas the intervals for the stem-and-leaf display use the base 10 number system. There is more arbitrariness to histograms than to stem-and-leaf displays, so histograms can sometimes be regarded a bit suspiciously.

```
#### hist
# histogram using R base graphics
  # par() gives graphical options
  #   mfrow = "multifigure by row" with 1 row and 3 columns
par(mfrow=c(1,3))
```

```
  # main is the title, xlab is x-axis label (ylab also available)
hist(hb, main="Modern Englishman", xlab="head breadth (mm)")
  # breaks are how many bins-1 to use
hist(hb, breaks = 15, main="Histogram, 15 breaks")
  # freq=FALSE changes the vertical axis to density,
  #   so the total area of the bars is now equal to 1
hist(hb, breaks = 8, freq = FALSE, main="Histogram, density")

# histogram using ggplot
library(ggplot2)
# first put hb vector into a data.frame
hb_df <- data.frame(hb)
p1 <- ggplot(hb_df, aes(x = hb))
  # always specify a binwidth for the histogram (default is range/30)
  #   try several binwidths
p1 <- p1 + geom_histogram(binwidth = 5)
p1 <- p1 + geom_rug()
p1 <- p1 + labs(title = "Modern Englishman head breadth")

p2 <- ggplot(hb_df, aes(x = hb))
  # always specify a binwidth for the histogram (default is range/30)
  #   try several binwidths
p2 <- p2 + geom_histogram(binwidth = 2)
p2 <- p2 + geom_rug()
p2 <- p2 + labs(title = "Modern Englishman head breadth")

library(gridExtra)
grid.arrange(p1, p2, nrow=1)
```



R allows you to modify the graphical display. For example, with the histogram you might wish to use different midpoints or interval widths. I will let you explore the possibilities.

1.3.3 Stem-and-leaf plot

A **stem-and-leaf** plot is a character display histogram defining intervals for a grouped frequency distribution using the base 10 number system. Intervals are generated by selecting an appropriate number of lead digits for the data values to be the stem. The remaining digits comprise the leaf. It is useful for small samples.

Character plots use typewriter characters to make graphs, and can be convenient for some simple displays, but require use of fixed fonts (like Courier) when copied to a word processing program or they get distorted.

The display almost looks upside down, since larger numbers are on the bottom rather than the top. It is done this way so that if rotated 90 degrees counter-clockwise it *is* a histogram.

The default stem-and-leaf display for the head breadth data is given below. The two columns give the stems and leaves. The data have three digits. The first two comprise the stem. The last digit is the leaf. Thus, a head breadth of 154 has a stem of 15 and leaf of 4. The possible stems are 13, 14, and 15, whereas the possible leaves are the integers from 0 to 9. In the first plot, each stem occurs once, while in the second each stem occurs twice. In the second instance, the first (top) occurrence of a stem value only holds leaves 0 through 4. The second occurrence holds leaves 5 through 9. The display is generated by placing the leaf value for each observation on the appropriate stem line. For example, the top 14 stem holds data values between 140 and 144.99. The stems on this line in the display tell us that four observations fall in this range: 140, 141, 142 and 144. Note that this stem-and-leaf display is an elaborate histogram with intervals of width 5. An advantage of the stem-and-leaf display over the histogram is that the original data values can essentially be recovered from the display.

```
#### stem-and-leaf
# stem-and-leaf plot
stem(hb)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
```

```
##   13 | 28
##   14 | 0124567889
##   15 | 000458
# scale=2 makes plot roughly twice as wide
stem(hb, scale=2)
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   13 | 2
##   13 | 8
##   14 | 0124
##   14 | 567889
##   15 | 0004
##   15 | 58
# scale=5 makes plot roughly five times as wide
stem(hb, scale=5)
##
##   The decimal point is at the |
##
##   132 | 0
##   134 |
##   136 |
##   138 | 0
##   140 | 00
##   142 | 0
##   144 | 00
##   146 | 00
##   148 | 000
##   150 | 000
##   152 |
##   154 | 00
##   156 |
##   158 | 0
```

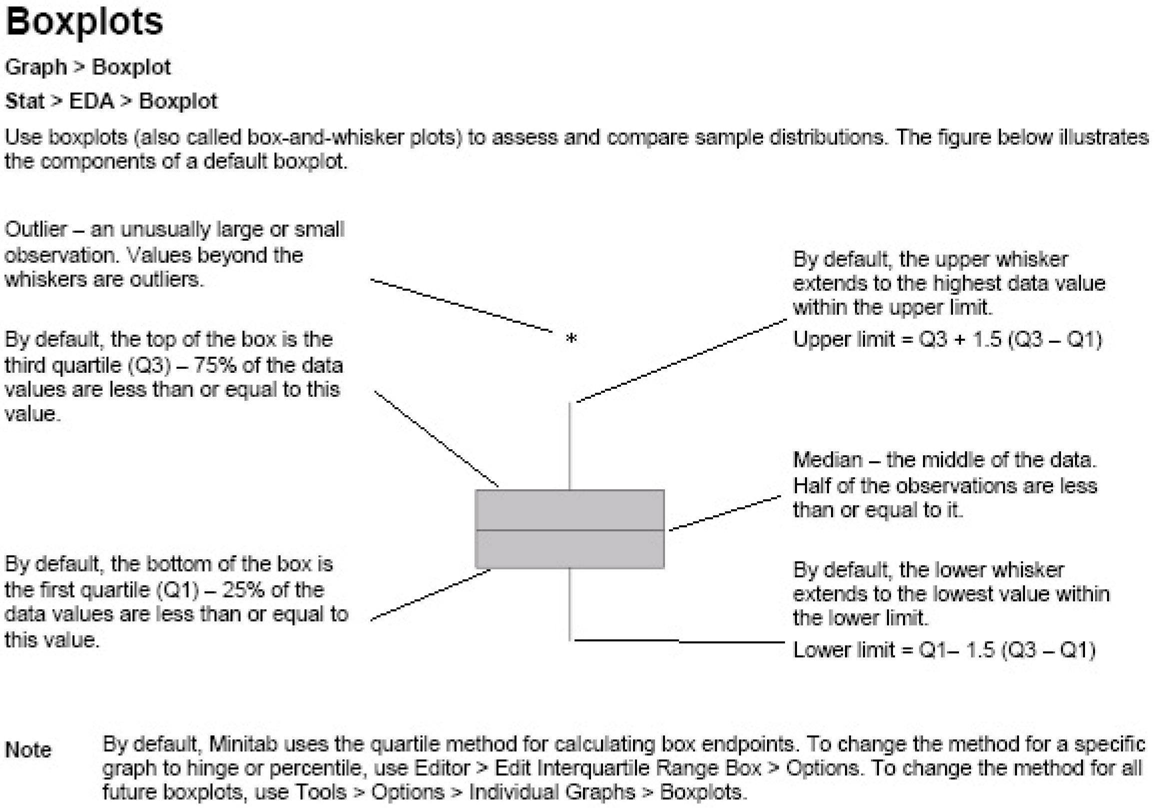
The data values are always *truncated* so that a leaf has one digit. The leaf unit (location of the decimal point) tells us the degree of round-off. This will become clearer in the next example.

Of the three displays, which is the most informative? I think the middle option is best to see the clustering and shape of distributions of numbers.

■ ClickerQs — Stem-and-leaf plot ■

1.3.4 Boxplot or box-and-whiskers plot

The **boxplot** breaks up the range of data values into regions about the center of the data, measured by the median. The boxplot highlights **outliers** and provides a visual means to assess "**normality**". The following help entry outlines the construction of the boxplot, given the placement of data values on the axis.



Boxplots

Graph > Boxplot

Stat > EDA > Boxplot

Use boxplots (also called box-and-whisker plots) to assess and compare sample distributions. The figure below illustrates the components of a default boxplot.

Outlier – an unusually large or small observation. Values beyond the whiskers are outliers.

By default, the top of the box is the third quartile (Q3) – 75% of the data values are less than or equal to this value.

By default, the bottom of the box is the first quartile (Q1) – 25% of the data values are less than or equal to this value.

By default, the upper whisker extends to the highest data value within the upper limit.

Upper limit = Q3 + 1.5 (Q3 – Q1)

Median – the middle of the data. Half of the observations are less than or equal to it.

By default, the lower whisker extends to the lowest value within the lower limit.

Lower limit = Q1– 1.5 (Q3 – Q1)

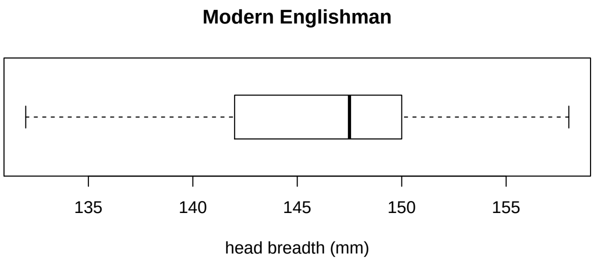
Note By default, Minitab uses the quartile method for calculating box endpoints. To change the method for a specific graph to hinge or percentile, use Editor > Edit Interquartile Range Box > Options. To change the method for all future boxplots, use Tools > Options > Individual Graphs > Boxplots.

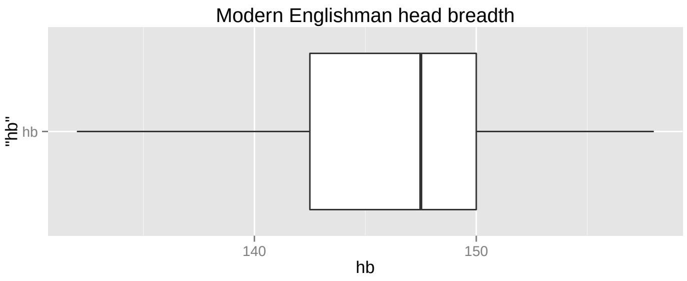
The endpoints of the box are placed at the locations of the first and third quartiles. The location of the median is identified by the line in the box. The whiskers extend to the data points closest to but not on or outside the outlier fences, which are $1.5IQR$ from the quartiles. Outliers are any values on or outside the outlier fences.

The boxplot for the head breadth data is given below. There are a lot of options that allow you to clutter the boxplot with additional information. Just use the default settings. We want to see the relative location of data (the median line), have an idea of the spread of data (IQR, the length of the box), see the shape of the data (relative distances of components from each other – to be covered later), and identify outliers (if present). The default boxplot has all these components.

Note that the boxplots below are horizontal to better fit on the page. The `horizontal=TRUE` and `coord_flip()` commands do this.

```
#### boxplot
fivenum(hb)

## [1] 132.0 142.0 147.5 150.0 158.0

# boxplot using R base graphics
par(mfrow=c(1,1))
boxplot(hb, horizontal=TRUE
      , main="Modern Englishman", xlab="head breadth (mm)")

# boxplot using ggplot
library(ggplot2)
# first put hb vector into a data.frame
hb_df <- data.frame(hb)
p <- ggplot(hb_df, aes(x = "hb", y = hb))
p <- p + geom_boxplot()
p <- p + coord_flip()
p <- p + labs(title = "Modern Englishman head breadth")
print(p)
```





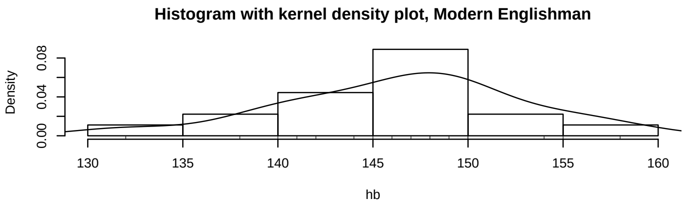
■ CLICKERQs — Boxplots ■

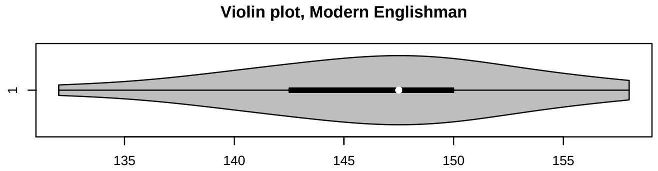
Improvements to the boxplot

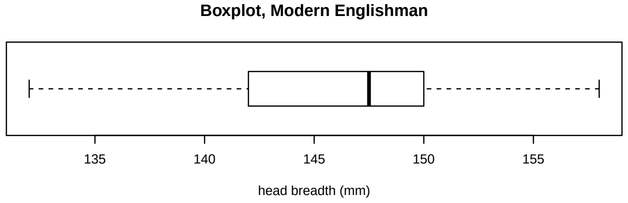
As a quick aside, a violin plot is a combination of a boxplot and a kernel density plot. They can be created using the `vioplot()` function from **vioplot** package.

```
#### vioplot
# vioplot using R base graphics
# 3 rows, 1 column
par(mfrow=c(3,1))

# histogram
```

```
hist(hb, freq = FALSE
    , main="Histogram with kernel density plot, Modern Englishman")
# Histogram overlaid with kernel density curve
points(density(hb), type = "l")
# rug of points under histogram
rug(hb)

# violin plot
library(vioplot)
vioplot(hb, horizontal=TRUE, col="gray")
title("Violin plot, Modern Englishman")

# boxplot
boxplot(hb, horizontal=TRUE
      , main="Boxplot, Modern Englishman", xlab="head breadth (mm)")
```







Example: income The data below are incomes in $1000 units for a sample of 12 retired couples. Numerical and graphical summaries are given. There are two stem-and-leaf displays provided. The first is the default display.

```
#### Income examples
income <- c(7, 1110, 7, 5, 8, 12, 0, 5, 2, 2, 46, 7)
# sort in decreasing order
income <- sort(income, decreasing = TRUE)
income
## [1] 1110   46   12    8    7    7    7    5    5    2    2    0
summary(income)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    4.25    7.00  100.90    9.00 1110.00
```

```
# stem-and-leaf plot
stem(income)
```

```
##
##   The decimal point is 3 digit(s) to the right of the |
##
##   0 | 00000000000
##   0 |
##   1 | 1
```

Because the two large outliers, I trimmed them to get a sense of the shape of the distribution where most of the observations are.

```
#### remove largest
# remove two largest values (the first two)
income2 <- income[-c(1,2)]
income2
```

```
##  [1] 12  8  7  7  7  5  5  2  2  0
```

```
summary(income2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    2.75    6.00    5.50    7.00   12.00
```

```
# stem-and-leaf plot
stem(income2)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   0 | 022
##   0 | 557778
##   1 | 2
```

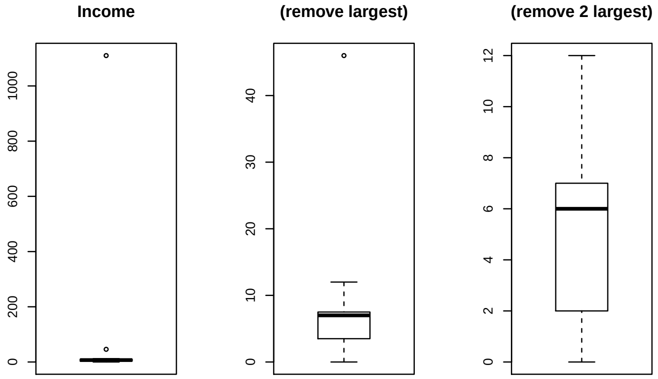
```
# scale=2 makes plot roughly twice as wide
stem(income2, scale=2)
```

```
##
##   The decimal point is at the |
##
##    0 | 0
##    2 | 00
##    4 | 00
##    6 | 000
##    8 | 0
##   10 |
##   12 | 0
```

Boxplots with full data, then incrementally removing the two largest outliers.

```
#### income-boxplot
# boxplot using R base graphics
```

```r
# 1 row, 3 columns
par(mfrow=c(1,3))
boxplot(income, main="Income")
boxplot(income[-1], main="(remove largest)")
boxplot(income2, main="(remove 2 largest)")
```



1.4 Interpretation of Graphical Displays for Numerical Data

In many studies, the data are viewed as a subset or **sample** from a larger collection of observations or individuals under study, called the **population**. A primary goal of many statistical analyses is to generalize the information in the sample to **infer** something about the population. For this generalization to be possible, the sample must reflect the basic patterns of the population. There are several ways to collect data to ensure that the sample reflects the basic properties of the population, but the simplest approach, by far, is to take a random or "representative" sample from the population. A **random sample** has the property that every possible sample of a given size has the same chance of being the sample (eventually) selected (though we often do this only once). Random sampling eliminates any systematic biases associated with the selected observations, so the information in the sample should accurately reflect features of the population. The process of sampling introduces random variation or random errors associated with summaries. Statistical tools are used to calibrate the size of the errors.

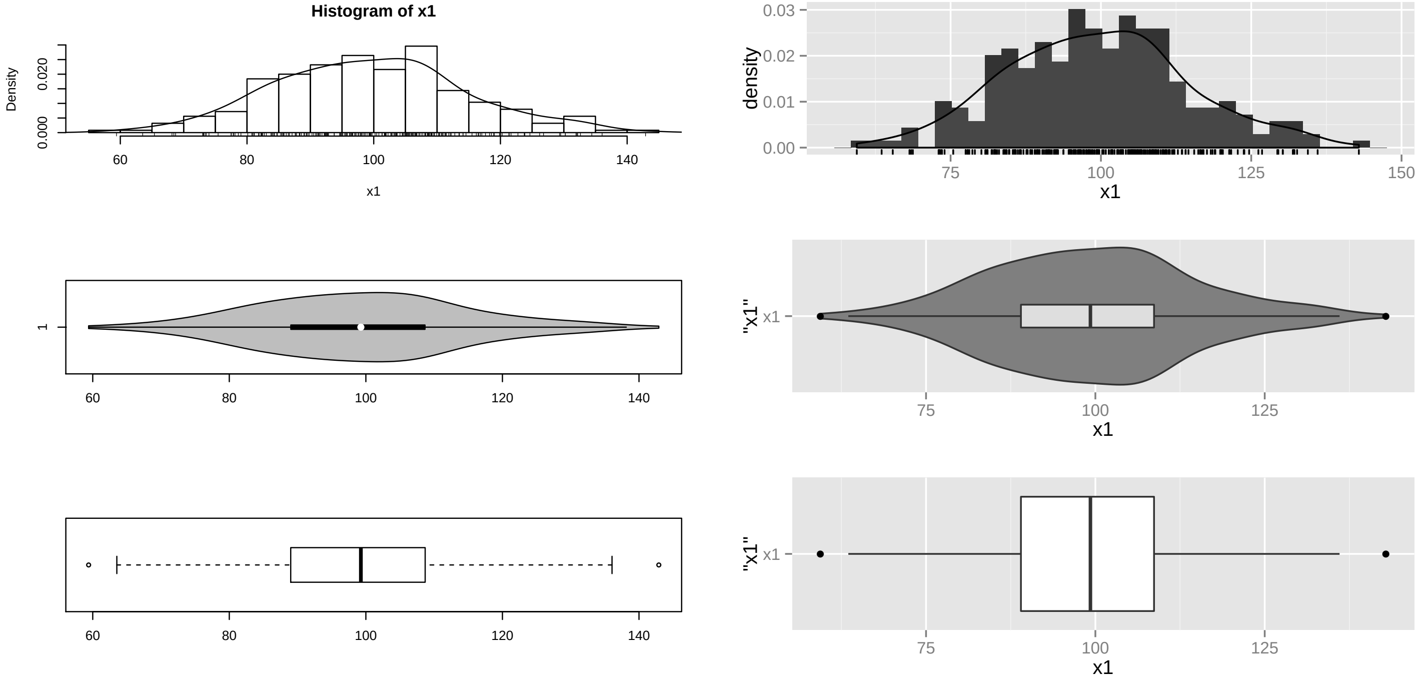
Whether we are looking at a histogram (or stem-and-leaf, or dotplot) from a sample, or are conceptualizing the histogram generated by the population data, we can imagine approximating the "envelope" around the display with a smooth curve. The smooth curve that approximates the population histogram is called the **population frequency curve** or **population probability density function** or **population distribution**[2]. Statistical methods for inference about a population usually make assumptions about the shape of the population frequency curve. A common assumption is that the population has a normal frequency curve. In practice, the observed data are used to assess the reasonableness of this assumption. In particular, a sample display should resemble a population display, provided the collected data are a random or representative sample from the population. Several common shapes for frequency distributions are given below, along with the statistical terms used to describe them.

Unimodal, symmetric, bell-shaped, and no outliers The first display is **unimodal** (one peak), **symmetric**, and **bell-shaped** with no outliers. This is the prototypical normal curve. The boxplot (laid on its side for this display) shows strong evidence of symmetry: the median is about halfway between the first and third quartiles, and the tail lengths are roughly equal. The boxplot is calibrated in such a way that 7 of every 1000 observations are outliers (more than $1.5(Q_3 - Q_1)$ from the quartiles) in samples from a population with a normal frequency curve. Only 2 out of every 1 million observations are extreme outliers (more than $3(Q_3 - Q_1)$ from the quartiles). We do not have any outliers here out of 250 observations, but we certainly could have some without indicating nonnormality. If a sample of 30 observations contains 4 outliers, two of which are extreme, would it be reasonable to assume the population from which the data were collected has a normal frequency curve? Probably not.

```
#### Unimodal, symmetric, bell-shaped, and no outliers (Normal distribution)
## base graphics
```

[2] "Distribution function" often refers to the "cumulative distribution function", which is a different (but one-to-one related) function than what I mean here.

```
# sample from normal distribution
x1 <- rnorm(250, mean = 100, sd = 15)

par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(x1, freq = FALSE, breaks = 20)
points(density(x1), type = "l")
rug(x1)

# violin plot
library(vioplot)
vioplot(x1, horizontal=TRUE, col="gray")

# boxplot
boxplot(x1, horizontal=TRUE)


## ggplot
# Histogram overlaid with kernel density curve
x1_df <- data.frame(x1)
p1 <- ggplot(x1_df, aes(x = x1))
  # Histogram with density instead of count on y-axis
p1 <- p1 + geom_histogram(aes(y=..density..))
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

# violin plot
p2 <- ggplot(x1_df, aes(x = "x1", y = x1))
p2 <- p2 + geom_violin(fill = "gray50")
p2 <- p2 + geom_boxplot(width = 0.2, alpha = 3/4)
p2 <- p2 + coord_flip()

# boxplot
p3 <- ggplot(x1_df, aes(x = "x1", y = x1))
p3 <- p3 + geom_boxplot()
p3 <- p3 + coord_flip()

library(gridExtra)
grid.arrange(p1, p2, p3, ncol=1)
## stat_bin:  binwidth defaulted to range/30.  Use ’binwidth = x’ to adjust this.
```



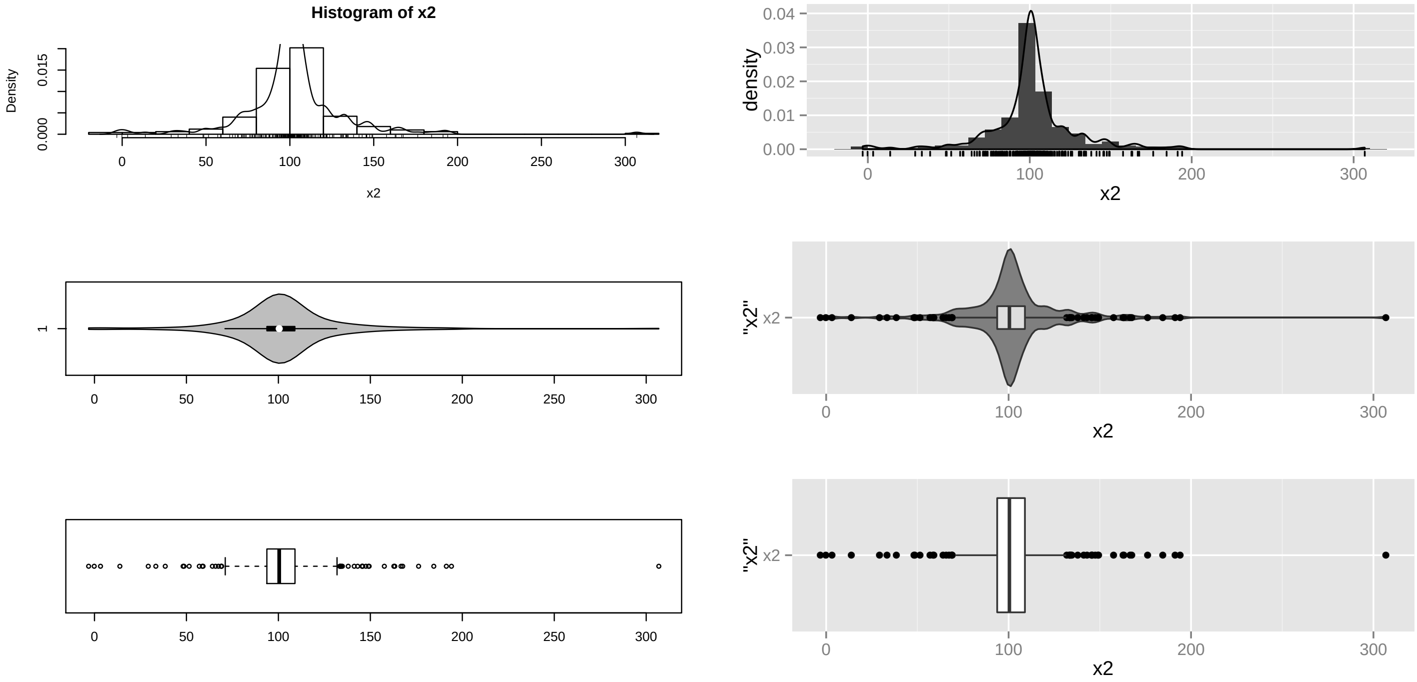
```
#### Central statistical moments
# moments package for 3rd and 4th moments: skewness() and kurtosis()
library(moments)
summary(x1)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   59.40   89.01   99.27   99.34  108.70  142.90

sd(x1)

## [1] 15.1007

skewness(x1)

## [1] 0.142431

kurtosis(x1)

## [1] -0.1399989

stem(x1)

##
##   The decimal point is 1 digit(s) to the right of the |
##
##    5 | 9
##    6 | 4
##    6 | 5889
##    7 | 3333344
##    7 | 578888899
##    8 | 01111122222223344444
##    8 | 55555666667777888889999999
##    9 | 000111111122222233333344
##    9 | 5555555556666666677777888888889999999
##   10 | 00000111222222233333344444
##   10 | 555555555666666667777777788888899999999
##   11 | 0000011111122233444
##   11 | 566677788999
##   12 | 00001123444
##   12 | 5679
##   13 | 00022234
##   13 | 6
##   14 | 3
```

Unimodal, symmetric, heavy-tailed The boxplot is better at highlighting outliers than are other displays. The histogram and stem-and-leaf displays below appear to have the same basic shape as a normal curve (unimodal, symmetric). However, the boxplot shows that we have a dozen outliers in a sample of 250 observations. We would only expect about two outliers in 250 observations when sampling from a population with a normal frequency curve. The frequency curve is best described as unimodal, symmetric, and **heavy-tailed**.

```
#### Unimodal, symmetric, heavy-tailed
# sample from normal distribution
x2.temp <- rnorm(250, mean = 0, sd = 1)
x2 <- sign(x2.temp)*x2.temp^2 * 15 + 100
```

```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(x2, freq = FALSE, breaks = 20)
points(density(x2), type = "l")
rug(x2)

# violin plot
library(vioplot)
vioplot(x2, horizontal=TRUE, col="gray")

# boxplot
boxplot(x2, horizontal=TRUE)


# Histogram overlaid with kernel density curve
x2_df <- data.frame(x2)
p1 <- ggplot(x2_df, aes(x = x2))
  # Histogram with density instead of count on y-axis
p1 <- p1 + geom_histogram(aes(y=..density..))
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

# violin plot
```

```
p2 <- ggplot(x2_df, aes(x = "x2", y = x2))
p2 <- p2 + geom_violin(fill = "gray50")
p2 <- p2 + geom_boxplot(width = 0.2, alpha = 3/4)
p2 <- p2 + coord_flip()

# boxplot
p3 <- ggplot(x2_df, aes(x = "x2", y = x2))
p3 <- p3 + geom_boxplot()
p3 <- p3 + coord_flip()

library(gridExtra)
grid.arrange(p1, p2, p3, ncol=1)
## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x' to adjust this.
```



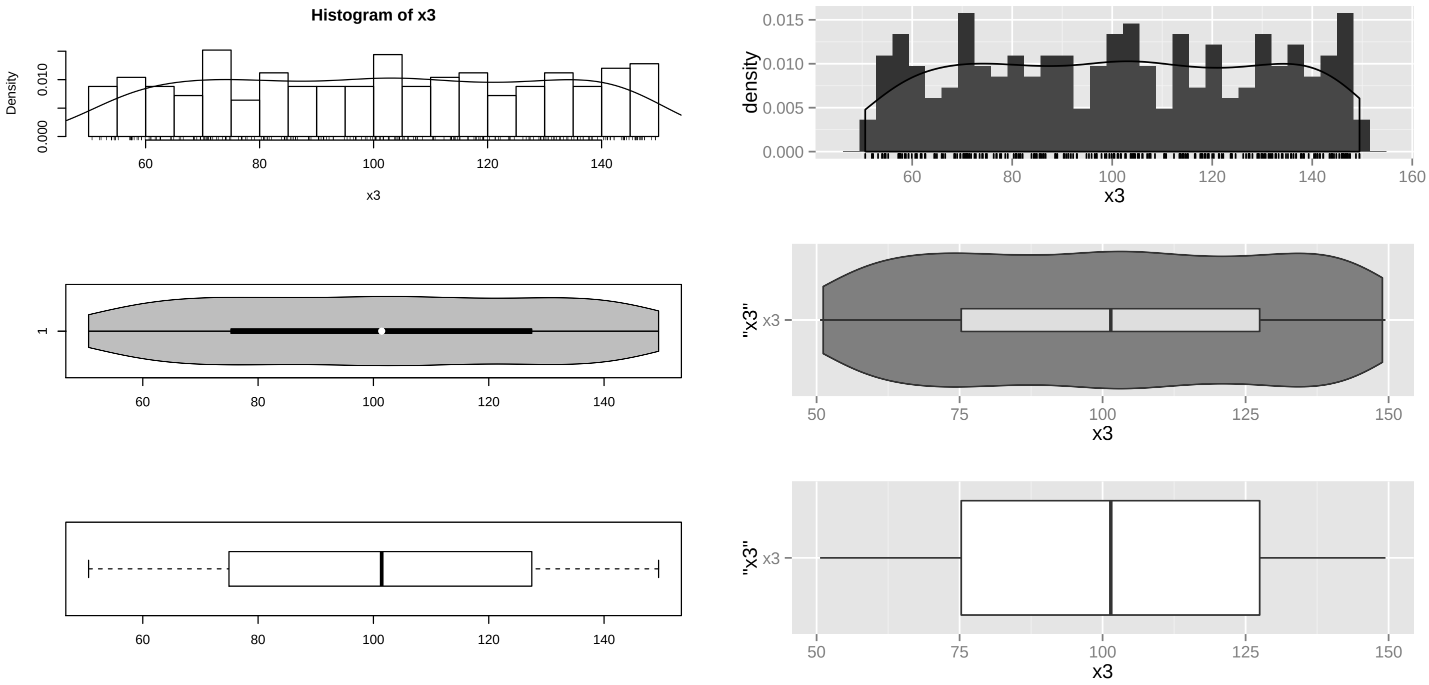
```
summary(x2)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -3.186  93.750 100.400 102.100 108.900 306.900
sd(x2)
## [1] 29.4546
skewness(x2)
## [1] 1.117841
kurtosis(x2)
## [1] 10.7752
stem(x2)
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   -0 | 30
##    0 | 34
##    2 | 938
##    4 | 891799
##    6 | 4679991123334678899
##    8 | 001233345667888890012222222333444445555666667777788888899999999
##   10 | 00000000000000000000000000000000111111111222222222222233333333444445+33
##   12 | 000001222455601122344458
##   14 | 135668998
##   16 | 33786
##   18 | 514
##   20 |
##   22 |
##   24 |
##   26 |
##   28 |
##   30 | 7
```

Symmetric, (uniform,) short-tailed Not all symmetric distributions are mound-shaped, as the display below suggests. The boxplot shows symmetry, but the tails of the distribution are shorter (lighter) than in the normal distribution. Note that the distance between quartiles is roughly constant here.

```
#### Symmetric, (uniform,) short-tailed
# sample from uniform distribution
x3 <- runif(250, min = 50, max = 150)
```

```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(x3, freq = FALSE, breaks = 20)
points(density(x3), type = "l")
rug(x3)

# violin plot
library(vioplot)
vioplot(x3, horizontal=TRUE, col="gray")

# boxplot
boxplot(x3, horizontal=TRUE)


# Histogram overlaid with kernel density curve
x3_df <- data.frame(x3)
p1 <- ggplot(x3_df, aes(x = x3))
  # Histogram with density instead of count on y-axis
```

```
p1 <- p1 + geom_histogram(aes(y=..density..))
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

# violin plot
p2 <- ggplot(x3_df, aes(x = "x3", y = x3))
p2 <- p2 + geom_violin(fill = "gray50")
p2 <- p2 + geom_boxplot(width = 0.2, alpha = 3/4)
p2 <- p2 + coord_flip()

# boxplot
p3 <- ggplot(x3_df, aes(x = "x3", y = x3))
p3 <- p3 + geom_boxplot()
p3 <- p3 + coord_flip()

library(gridExtra)
grid.arrange(p1, p2, p3, ncol=1)
## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x' to adjust this.
```



```
summary(x3)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   50.61   75.29  101.40  101.30  127.50  149.50
sd(x3)
## [1] 29.02638
skewness(x3)
## [1] -0.009479507
kurtosis(x3)
```

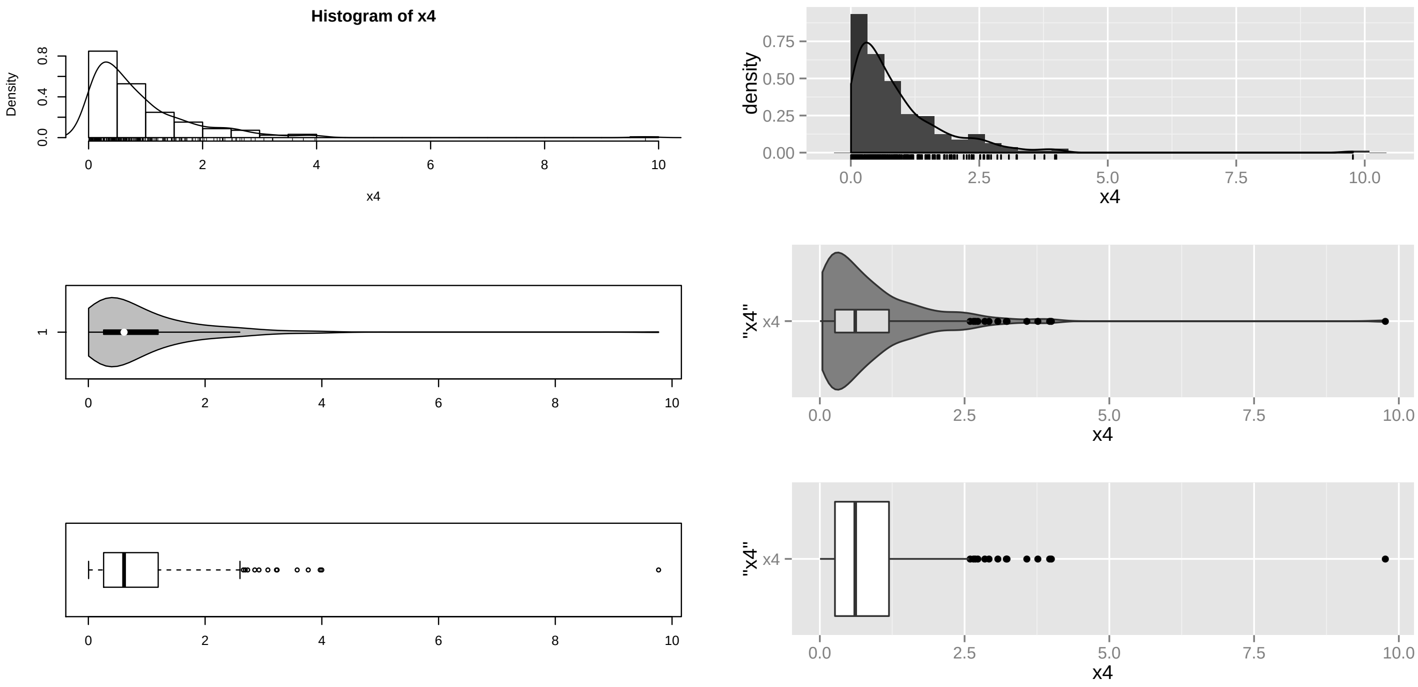
```
## [1] -1.236084
stem(x3)
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    5 | 12234444
##    5 | 555577778889999
##    6 | 0111223334
##    6 | 556678899
##    7 | 0000011111122334444
##    7 | 5567778899
##    8 | 011111224444
##    8 | 5556666799999
##    9 | 0001112233
##    9 | 55667778999
##   10 | 00000111223334444
##   10 | 5555566777889
##   11 | 001123344444
##   11 | 55577888899999
##   12 | 001122444
##   12 | 5677778999
##   13 | 000011222333344
##   13 | 556667788889
##   14 | 01111222344444
##   14 | 55666666777788999
```

The mean and median are identical in a population with a (exact) symmetric frequency curve. The histogram and stem-and-leaf displays for a sample selected from a symmetric population will tend to be fairly symmetric. Further, the sample means and medians will likely be close.

Unimodal, skewed right The distribution below is unimodal, and asymmetric or **skewed**. The distribution is said to be **skewed to the right**, or upper end, because the right tail is much longer than the left tail. The boxplot also shows the skewness – the region between the minimum observation and the median contains half the data in less than 1/5 the range of values. In addition, the upper tail contains several outliers.

```
#### Unimodal, skewed right
# sample from exponential distribution
x4 <- rexp(250, rate = 1)
```

```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(x4, freq = FALSE, breaks = 20)
points(density(x4), type = "l")
rug(x4)

# violin plot
library(vioplot)
vioplot(x4, horizontal=TRUE, col="gray")

# boxplot
boxplot(x4, horizontal=TRUE)


# Histogram overlaid with kernel density curve
x4_df <- data.frame(x4)
p1 <- ggplot(x4_df, aes(x = x4))
  # Histogram with density instead of count on y-axis
p1 <- p1 + geom_histogram(aes(y=..density..))
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

# violin plot
p2 <- ggplot(x4_df, aes(x = "x4", y = x4))
p2 <- p2 + geom_violin(fill = "gray50")
p2 <- p2 + geom_boxplot(width = 0.2, alpha = 3/4)
p2 <- p2 + coord_flip()

# boxplot
p3 <- ggplot(x4_df, aes(x = "x4", y = x4))
p3 <- p3 + geom_boxplot()
p3 <- p3 + coord_flip()

library(gridExtra)
grid.arrange(p1, p2, p3, ncol=1)
## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x' to adjust this.
```



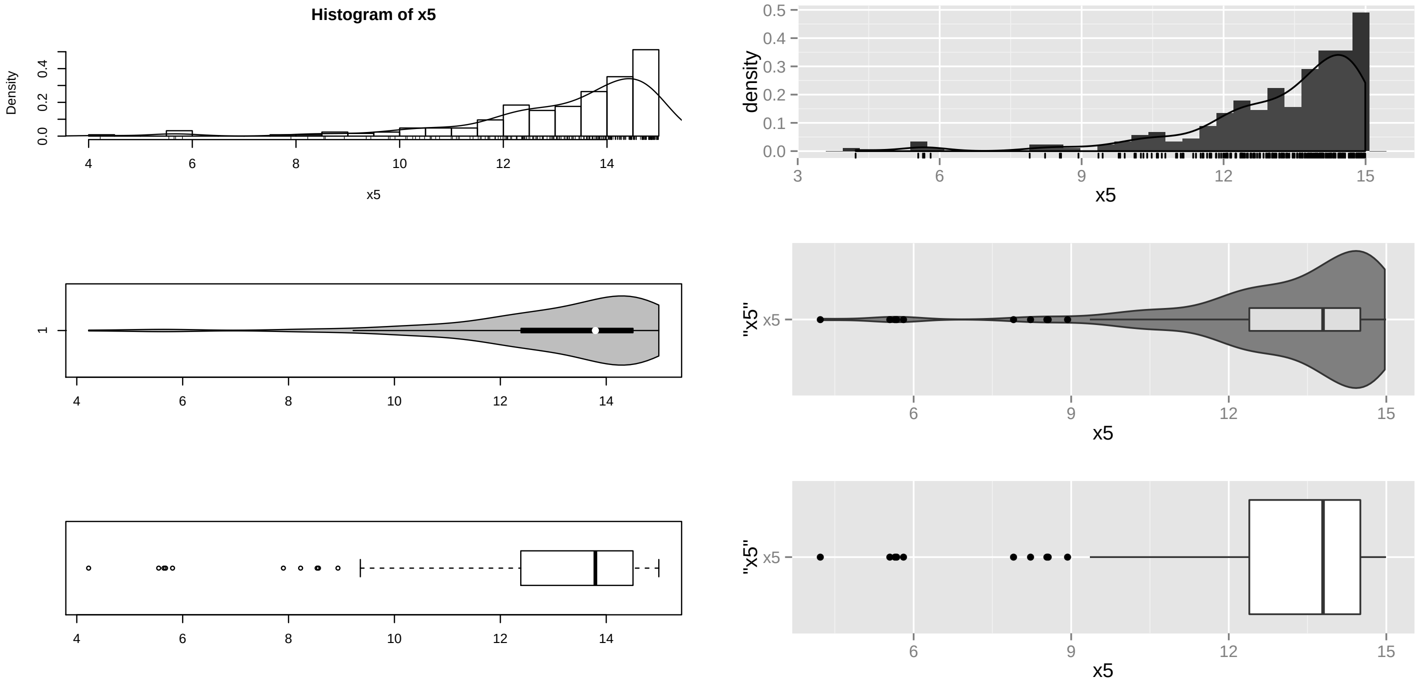
```
summary(x4)
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.003949 0.261100 0.611600 0.908200 1.194000 9.770000
sd(x4)
## [1] 0.9963259
skewness(x4)
## [1] 3.574832
kurtosis(x4)
## [1] 24.23519
stem(x4)
##
##   The decimal point is at the |
##
##   0 | 0000000000111111111111111111111122222222222222222222222222333333333+15
##   0 | 5555555555555555555556666666666666677777777777778888888889999999999
##   1 | 0000000111111111112222333333444
##   1 | 5555555566666777788999
##   2 | 0000123333444
##   2 | 556677799
##   3 | 122
##   3 | 68
##   4 | 00
##   4 |
##   5 |
##   5 |
##   6 |
##   6 |
##    7 |
##    7 |
##    8 |
##    8 |
##    9 |
##    9 | 8
```

Unimodal, skewed left The distribution below is unimodal and **skewed to the left**. The two examples show that extremely skewed distributions often contain outliers in the longer tail of the distribution.

```
#### Unimodal, skewed left
# sample from uniform distribution
x5 <- 15 - rexp(250, rate = 0.5)
```

```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(x5, freq = FALSE, breaks = 20)
points(density(x5), type = "l")
rug(x5)

# violin plot
library(vioplot)
vioplot(x5, horizontal=TRUE, col="gray")

# boxplot
boxplot(x5, horizontal=TRUE)


# Histogram overlaid with kernel density curve
x5_df <- data.frame(x5)
p1 <- ggplot(x5_df, aes(x = x5))
  # Histogram with density instead of count on y-axis
p1 <- p1 + geom_histogram(aes(y=..density..))
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

# violin plot
p2 <- ggplot(x5_df, aes(x = "x5", y = x5))
p2 <- p2 + geom_violin(fill = "gray50")
p2 <- p2 + geom_boxplot(width = 0.2, alpha = 3/4)
p2 <- p2 + coord_flip()

# boxplot
p3 <- ggplot(x5_df, aes(x = "x5", y = x5))
p3 <- p3 + geom_boxplot()
```

```
p3 <- p3 + coord_flip()

library(gridExtra)
grid.arrange(p1, p2, p3, ncol=1)
## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x' to adjust this.
```



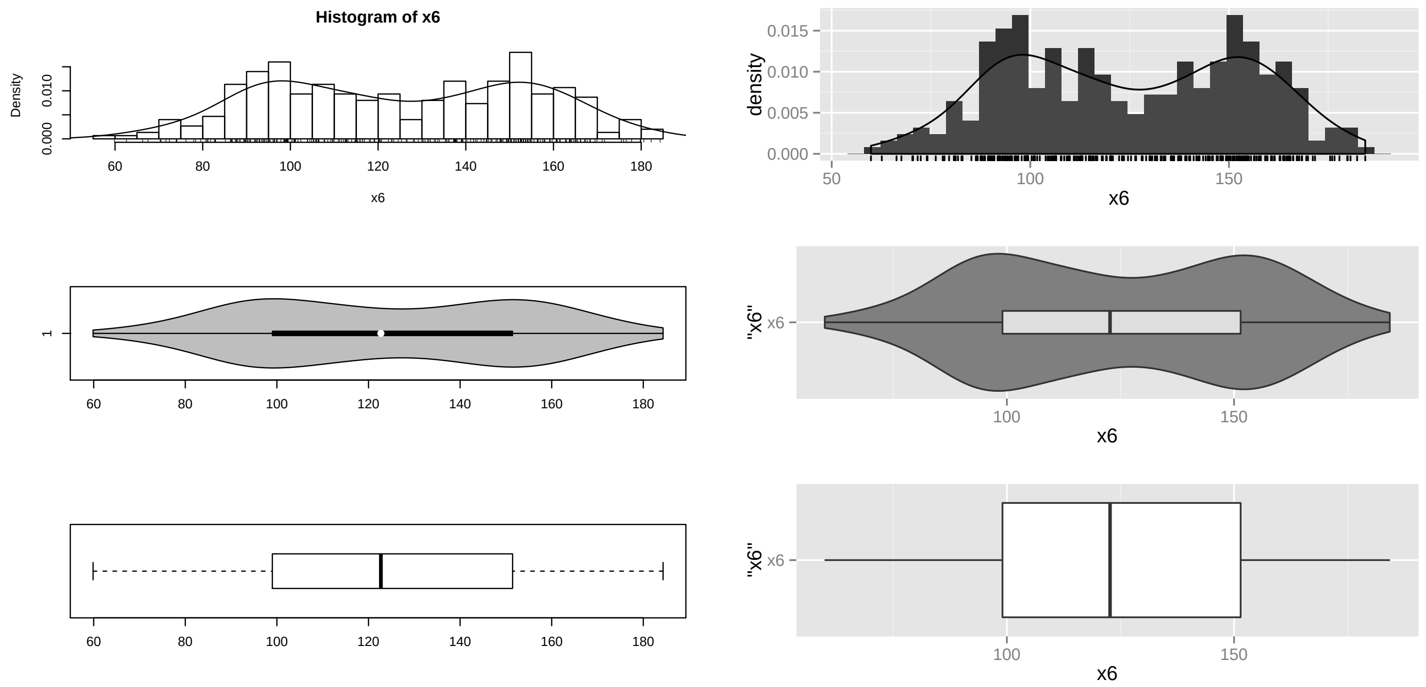
```
summary(x5)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.224  12.390  13.800  13.180  14.510  14.990
sd(x5)
## [1] 1.870509
skewness(x5)
## [1] -1.949473
kurtosis(x5)
## [1] 4.825639
stem(x5)
##
##   The decimal point is at the |
##
##    4 | 2
##    5 | 5678
##    6 |
##    7 | 9
##    8 | 2569
##    9 | 44889
##   10 | 11334566678
##   11 | 001124456677778899
##   12 | 0001111223334444444444455666667788889999
##   13 | 0000001111222222333344555666666667777778888889999999
##   14 | 00000000001111111111122222223333333333444444555555555555666667777777+25
##   15 | 000000000
```

Bimodal (multi-modal) Not all distributions are unimodal. The distribution below has two modes or peaks, and is said to be **bimodal**. Distributions with three or more peaks are called **multi-modal**.

```
#### Bimodal (multi-modal)
# sample from uniform distribution
x6 <- c(rnorm(150, mean = 100, sd = 15), rnorm(150, mean = 150, sd = 15))
```

```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(x6, freq = FALSE, breaks = 20)
points(density(x6), type = "l")
rug(x6)

# violin plot
library(vioplot)
vioplot(x6, horizontal=TRUE, col="gray")

# boxplot
boxplot(x6, horizontal=TRUE)


# Histogram overlaid with kernel density curve
x6_df <- data.frame(x6)
p1 <- ggplot(x6_df, aes(x = x6))
  # Histogram with density instead of count on y-axis
p1 <- p1 + geom_histogram(aes(y=..density..))
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

# violin plot
p2 <- ggplot(x6_df, aes(x = "x6", y = x6))
p2 <- p2 + geom_violin(fill = "gray50")
p2 <- p2 + geom_boxplot(width = 0.2, alpha = 3/4)
p2 <- p2 + coord_flip()

# boxplot
p3 <- ggplot(x6_df, aes(x = "x6", y = x6))
p3 <- p3 + geom_boxplot()
p3 <- p3 + coord_flip()
```

```
library(gridExtra)
grid.arrange(p1, p2, p3, ncol=1)
## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x' to adjust this.
```



```
summary(x6)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   59.87   99.03  122.70  124.70  151.40  184.30
sd(x6)
## [1] 29.59037
skewness(x6)
## [1] -0.005788873
kurtosis(x6)
## [1] -1.15984
stem(x6)
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    5 |
##    6 | 0368
##    7 | 002244688
##    8 | 01112333566777788889999
##    9 | 0000011122222233334444555556666777899999999999999
##   10 | 0011111122444555666666678899
##   11 | 00011222333334555555666678899
##   12 | 00000001123345567889
##   13 | 00011122334456666777788889999
##    14 | 00122233455556677788888999
##    15 | 00000001111222223333344444455555666777888999
##    16 | 01111233344444455555666778889
##    17 | 0125678
##    18 | 00124
```

The boxplot and histogram or stem-and-leaf display (or dotplot) are used **together** to describe the distribution. The boxplot does not provide information about modality – it only tells you about skewness and the presence of outliers.

As noted earlier, many statistical methods assume the population frequency curve is normal. Small deviations from normality usually do not dramatically influence the operating characteristics of these methods. We worry most when the deviations from normality are severe, such as extreme skewness or heavy tails containing multiple outliers.

■ ClickerQs — Graphical summaries ■

1.5 Interpretations for examples

The head breadth sample is slightly skewed to the left, unimodal, and has no outliers. The distribution does not deviate substantially from normality. The various measures of central location ($\bar{Y} = 146.5$, $M = 147.5$) are close, which is common with fairly symmetric distributions containing no outliers.

The income sample is extremely skewed to the right due to the presence of two extreme outliers at 46 and 1110. A normality assumption here is unrealistic.

It is important to recognize the influence that outliers can have on the values of $\bar{Y}$ and s. The median and interquartile range are more robust (less sensitive) to the presence of outliers. For the income data $\bar{Y} = 100.9$ and $s = 318$, whereas $M = 7$ and $IQR = 8.3$. If we omit the two outliers, then $\bar{Y} = 5.5$ and $s = 3.8$, whereas $M = 6$ and $IQR = 5.25$.

The mean and median often have similar values in data sets without outliers, so it does not matter much which one is used as the “typical value”. This issue is important, however, in data sets with extreme outliers. In such instances, the median is often more reasonable. For example, is $\bar{Y} = 100.9$ a reasonable measure for a typical income in this sample, given that the second largest income is only 46?

R Discussion I have included basic pointers on how to use R in these notes. I find that copying an R code example from the internet, then modifying the code to apply to my data is a productive strategy. When you’re first learning, “pretty good” is often “good enough”, especially when it comes to plots (in other words, spending 20 minutes vs 2 hours on a plot is fine). I will demonstrate most of what you need and I will be happy to answer questions. You will learn a lot more by using and experimenting with R than by watching.