Lecture notes for Advanced Data Analysis 1 (ADA1) Stat 427/527 University of New Mexico

Erik B. Erhardt
Edward J. Bedrick
Ronald M. Schrader

Fall 2017

Contents

Part I

Syllabus and Software

Chapter 0

Introduction to R, Rstudio, and ggplot

Contents

0.1 R building blocks

R as calculator In the following sections, look at the commands and output and understand how the command was interpretted to give the output.

```
#### Calculator
# Arithmetic
2 * 10
```

```
## [1] 20
```

```
1 + 2
```

```
## [1] 3
```

```
# Order of operations is preserved
1 + 5 * 10
```

```
## [1] 51
```

```
(1 + 5) * 10
```

```
## [1] 60
```

```
# Exponents use the ^ symbol
2^5
```

```
## [1] 32
```

```
9^(1/2)
```

```
## [1] 3
```

Vectors A vector is a set of numbers like a column of a spreadsheet. Below these sets of numbers are not technically "vectors", since they are not in a row/column structure, though they are ordered and can be referred to by index.

```
#### Vectors
# Create a vector with the c (short for combine) function
c(1, 4, 6, 7)
```

```
## [1] 1 4 6 7
```

```
c(1:5, 10)
```

```
## [1]  1  2  3  4  5 10
```

```
# or use a function
# (seq is short for sequence)
seq(1, 10, by = 2)
```

```
## [1] 1 3 5 7 9
```

```
seq(0, 50, length = 11)
```

```
##  [1]  0  5 10 15 20 25 30 35 40 45 50
```

```
seq(1, 50, length = 11)
```

```
##  [1]  1.0  5.9 10.8 15.7 20.6 25.5 30.4 35.3 40.2 45.1 50.0
```

```
1:10 # short for seq(1, 10, by = 1), or just
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
seq(1, 10)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
5:1
```

```
## [1] 5 4 3 2 1
```

```
# non-integer sequences
#   (Note: the [1] at the beginning of lines indicates
#          the index of the first value in that row)
seq(0, 100*pi, by = pi)
```

```
##   [1]   0.000000   3.141593   6.283185   9.424778  12.566371
##   [6]  15.707963  18.849556  21.991149  25.132741  28.274334
##  [11]  31.415927  34.557519  37.699112  40.840704  43.982297
```

```
##  [16]  47.123890  50.265482  53.407075  56.548668  59.690260
##  [21]  62.831853  65.973446  69.115038  72.256631  75.398224
##  [26]  78.539816  81.681409  84.823002  87.964594  91.106187
##  [31]  94.247780  97.389372 100.530965 103.672558 106.814150
##  [36] 109.955743 113.097336 116.238928 119.380521 122.522113
##  [41] 125.663706 128.805299 131.946891 135.088484 138.230077
##  [46] 141.371669 144.513262 147.654855 150.796447 153.938040
##  [51] 157.079633 160.221225 163.362818 166.504411 169.646003
##  [56] 172.787596 175.929189 179.070781 182.212374 185.353967
##  [61] 188.495559 191.637152 194.778745 197.920337 201.061930
##  [66] 204.203522 207.345115 210.486708 213.628300 216.769893
##  [71] 219.911486 223.053078 226.194671 229.336264 232.477856
##  [76] 235.619449 238.761042 241.902634 245.044227 248.185820
##  [81] 251.327412 254.469005 257.610598 260.752190 263.893783
##  [86] 267.035376 270.176968 273.318561 276.460154 279.601746
##  [91] 282.743339 285.884931 289.026524 292.168117 295.309709
##  [96] 298.451302 301.592895 304.734487 307.876080 311.017673
## [101] 314.159265
```

Assign variables Assigning objects to varibles.

```
#### Assign variables
# Assign a vector to a variable with <-
a <- 1:5
a
```

```
## [1] 1 2 3 4 5
```

```
b <- seq(15, 3, length = 5)
b
```

```
## [1] 15 12  9  6  3
```

```
c <- a * b
c
```

```
## [1] 15 24 27 24 15
```

Basic functions There are lots of functions available in the base package.

Type `?base` and click on `Index` at the bottom of the help page for a complete list of functions. Other functions to look at are in the `?stats` and `?datasets` packages.

```
#### Basic functions
# Lots of familiar functions work
a
```

```
## [1] 1 2 3 4 5
```

```
sum(a)
```

```
## [1] 15
prod(a)
## [1] 120
mean(a)
## [1] 3
sd(a)
## [1] 1.581139
var(a)
## [1] 2.5
min(a)
## [1] 1
median(a)
## [1] 3
max(a)
## [1] 5
range(a)
## [1] 1 5
```

Extracting subsets It will be an extremely important skill to understand the structure of your variables and pull out the information you need from them.

```
#### Extracting subsets
# Specify the indices you want in the square brackets []
a <- seq(0, 100, by = 10)
# blank = include all
a
##  [1]   0  10  20  30  40  50  60  70  80  90 100
a[]
##  [1]   0  10  20  30  40  50  60  70  80  90 100
# integer +=include, 0=include none, -=exclude
a[5]
## [1] 40
a[c(2, 4, 6, 8)]
## [1] 10 30 50 70
a[0]
## numeric(0)
a[-c(2, 4, 6, 8)]
## [1]   0  20  40  60  80  90 100
```

```
a[c(1, 1, 1, 6, 6, 9)]   # subsets can be bigger than the original set
## [1]  0  0  0 50 50 80
a[c(1,2)] <- c(333, 555) # update a subset
a
##  [1] 333 555  20  30  40  50  60  70  80  90 100
```

Boolean: True/False Subsets can be selected based on which elements meet specific conditions.

```
#### Boolean
a
##  [1] 333 555  20  30  40  50  60  70  80  90 100
(a > 50)         # TRUE/FALSE for each element
##  [1]  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
which(a > 50)    # which indicies are TRUE
## [1]  1  2  7  8  9 10 11
a[(a > 50)]
## [1] 333 555  60  70  80  90 100
!(a > 50)        # ! negates (flips) TRUE/FALSE values
##  [1] FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
a[!(a > 50)]
## [1] 20 30 40 50
```

Comparison functions All your favorite comparisons are available.

```
#### Comparison
# Here they are:  <  >  <=  >=  !=  ==  %in%
a
##  [1] 333 555  20  30  40  50  60  70  80  90 100
# equal to
a[(a == 50)]
## [1] 50
# equal to
a[(a == 55)]
## numeric(0)
# not equal to
a[(a != 50)]
##  [1] 333 555  20  30  40  60  70  80  90 100
```

```
# greater than
a[(a > 50)]
## [1] 333 555  60  70  80  90 100
# less than
a[(a < 50)]
## [1] 20 30 40
# less than or equal to
a[(a <= 50)]
## [1] 20 30 40 50
# which values on left are in the vector on right
(c(10, 14, 40, 60, 99) %in% a)
## [1] FALSE FALSE  TRUE  TRUE FALSE
```

Boolean operators compare TRUE/FALSE values and return TRUE/-FALSE values.

```
#### Boolean
# & and, | or, ! not
a
##  [1] 333 555  20  30  40  50  60  70  80  90 100
a[(a >= 50) & (a <= 90)]
## [1] 50 60 70 80 90
a[(a < 50) | (a > 100)]
## [1] 333 555  20  30  40
a[(a < 50) | !(a > 100)]
## [1]  20  30  40  50  60  70  80  90 100
a[(a >= 50) & !(a <= 90)]
## [1] 333 555 100
```

Missing values The value `NA` (not available) means the value is missing. Any calculation involving `NA` will return an `NA` by default.

```
#### Missing values
NA + 8
## [1] NA
3 * NA
## [1] NA
mean(c(1, 2, NA))
```

```
## [1] NA
# Many functions have an na.rm argument  (NA remove)
mean(c(NA, 1, 2), na.rm = TRUE)
## [1] 1.5
sum(c(NA, 1, 2))
## [1] NA
sum(c(NA, 1, 2), na.rm = TRUE)
## [1] 3
# Or you can remove them yourself
a <- c(NA, 1:5, NA)
a
## [1] NA  1  2  3  4  5 NA
is.na(a)      # which values are missing?
## [1]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
!is.na(a)     # which values are NOT missing?
## [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
a[!is.na(a)]  # return those which are NOT missing
## [1] 1 2 3 4 5
a             # note, this did not change the variable a
## [1] NA  1  2  3  4  5 NA
# To save the results of removing the NAs,
#   assign to another variable or reassign to the original variable
# Warning: if you write over variable a then the original version is gone forever!
a <- a[!is.na(a)]
a
## [1] 1 2 3 4 5
```

Your turn!

■ ClickerQs — Ch 0, R building blocks, Subset ■

■ ClickerQs — Ch 0, R building blocks, T/F selection 1 ■

■ CLICKER Qs — Ch 0, R building blocks, T/F selection 2 ■

How'd you do?

Outstanding Understanding the operations and how to put them together, without skipping steps.

Good Understanding most of the small steps, missed a couple details.

Hang in there Understanding some of the concepts but all the symbols make my eyes spin.

Reading and writing a new language takes work.

You'll get better as you practice, practice, practice[1].

Having a buddy to work with will help.

R command review This is a summary of the commands we've seen so far.

```
#### Review
<-
+ - * / ^
c()
seq() # by=, length=
sum(), prod(), mean(), sd(), var(),
min(), median(), max(), range()
a[]
(a > 1), ==, !=, >, <, >=, <=, %in%
&, |, !
NA, mean(a, na.rm = TRUE), !is.na()
```

0.2 Plotting with ggplot2

There are three primary strategies for plotting in R. The first is base graphics, using functions such as `plot()`, `points()`, and `par()`. A second is use of the package `lattice`, which creates very nice graphics quickly, though can be more difficult to create high-dimensional data displays. A third, and the one I will use throughout this course, is using package `ggplot2`, which is an implementation

[1] What's your Carnegie Hall that you're working towards?

of the "Grammar of Graphics". While creating a very simple plot requires an extra step or two, creating very complex displays are relatively straightforward as you become comfortable with the syntax.

This section is intended as an introduction to `ggplot()` and some of its capabilities (we will cover these plotting functions and others in detail in later chapters). As a basic introduction, it requires a `data.frame` object as input (a table where each row represents an observation or data point, and each column can have a different data type), and then you define plot layers that stack on top of each other, and each layer has visual/text elements that are mapped to aesthetics (colors, size, opacity). In this way, a simple set of commands can be combined to produce extremely informative displays.

In the example that follows, we consider a dataset `mpg` consisting of fuel economy data from 1999 and 2008 for 38 popular models of car.

```
#### Installing
# only needed once after installing or upgrading R
install.packages("ggplot2")
```

```
#### Library
# each time you start R
# load package ggplot2 for its functions and datasets
library(ggplot2)

# ggplot2 includes a dataset "mpg"

# ? gives help on a function or dataset
?mpg
```

Let's get a look at the dataset we're using.

```
#### mpg dataset
# head() lists the first several rows of a data.frame
head(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl      trans   drv   cty   hwy
##          <chr> <chr> <dbl> <int> <int>      <chr> <chr> <int> <int>
## 1         audi    a4   1.8  1999     4   auto(l5)     f    18    29
## 2         audi    a4   1.8  1999     4 manual(m5)     f    21    29
## 3         audi    a4   2.0  2008     4 manual(m6)     f    20    31
## 4         audi    a4   2.0  2008     4   auto(av)     f    21    30
## 5         audi    a4   2.8  1999     6   auto(l5)     f    16    26
## 6         audi    a4   2.8  1999     6 manual(m5)     f    18    26
## # ... with 2 more variables: fl <chr>, class <chr>
# str() gives the structure of the object
```

```
str(mpg)
```

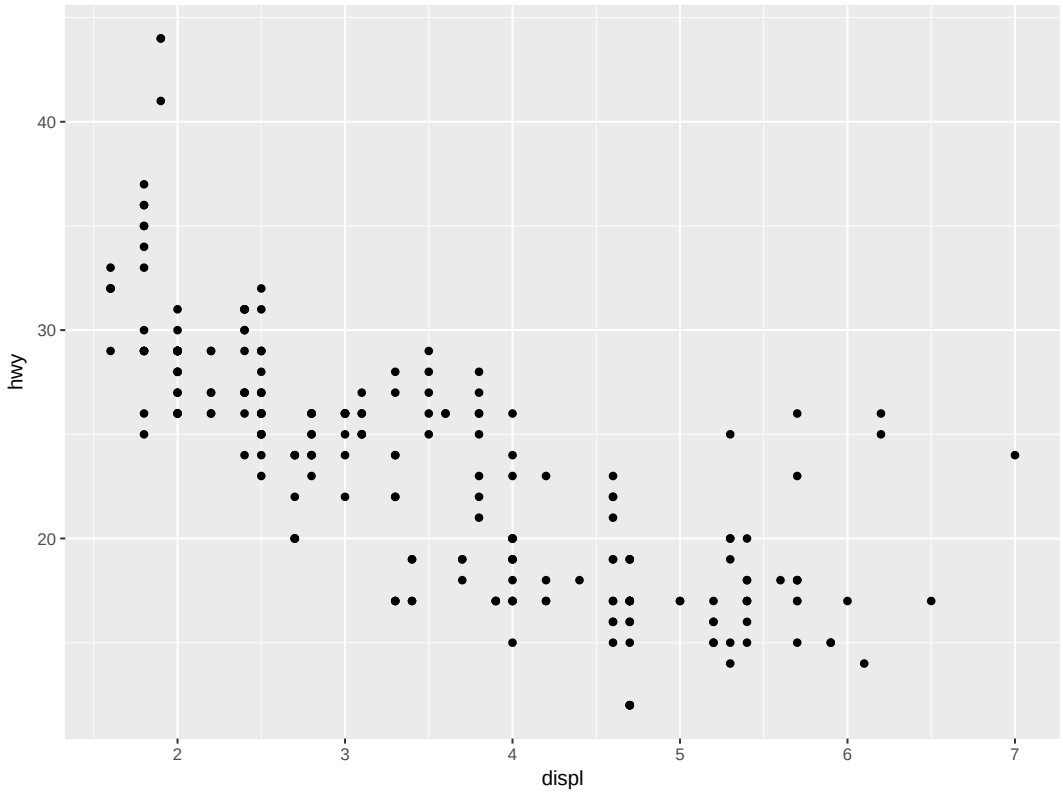
```
## Classes 'tbl_df', 'tbl' and 'data.frame': 234 obs. of  11 variables:
##  $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
##  $ model       : chr  "a4" "a4" "a4" "a4" ...
##  $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
##  $ cyl         : int  4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr  "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr  "f" "f" "f" "f" ...
##  $ cty         : int  18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int  29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr  "p" "p" "p" "p" ...
##  $ class       : chr  "compact" "compact" "compact" "compact" ...
```

```
# summary() gives frequeny tables for categorical variables
#             and mean and five-number summaries for continuous variables
summary(mpg)
```

```
##  manufacturer          model               displ            year
##  Length:234         Length:234         Min.   :1.600   Min.   :1999
##  Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999
##  Mode  :character   Mode  :character   Median :3.300   Median :2004
##                                        Mean   :3.472   Mean   :2004
##                                        3rd Qu.:4.600   3rd Qu.:2008
##                                        Max.   :7.000   Max.   :2008
##       cyl           trans               drv
##  Min.   :4.000   Length:234         Length:234
##  1st Qu.:4.000   Class :character   Class :character
##  Median :6.000   Mode  :character   Mode  :character
##  Mean   :5.889
##  3rd Qu.:8.000
##  Max.   :8.000
##       cty             hwy             fl
##  Min.   : 9.00   Min.   :12.00   Length:234
##  1st Qu.:14.00   1st Qu.:18.00   Class :character
##  Median :17.00   Median :24.00   Mode  :character
##  Mean   :16.86   Mean   :23.44
##  3rd Qu.:19.00   3rd Qu.:27.00
##  Max.   :35.00   Max.   :44.00
##     class
##  Length:234
##  Class :character
##  Mode  :character
##
##
##
```

```
#### ggplot_mpg_displ_hwy
# specify the dataset and variables
p <- ggplot(mpg, aes(x = displ, y = hwy))
```

```
p <- p + geom_point() # add a plot layer with points
print(p)
```



Geoms, aesthetics, and facets are three concepts we'll see in this section.

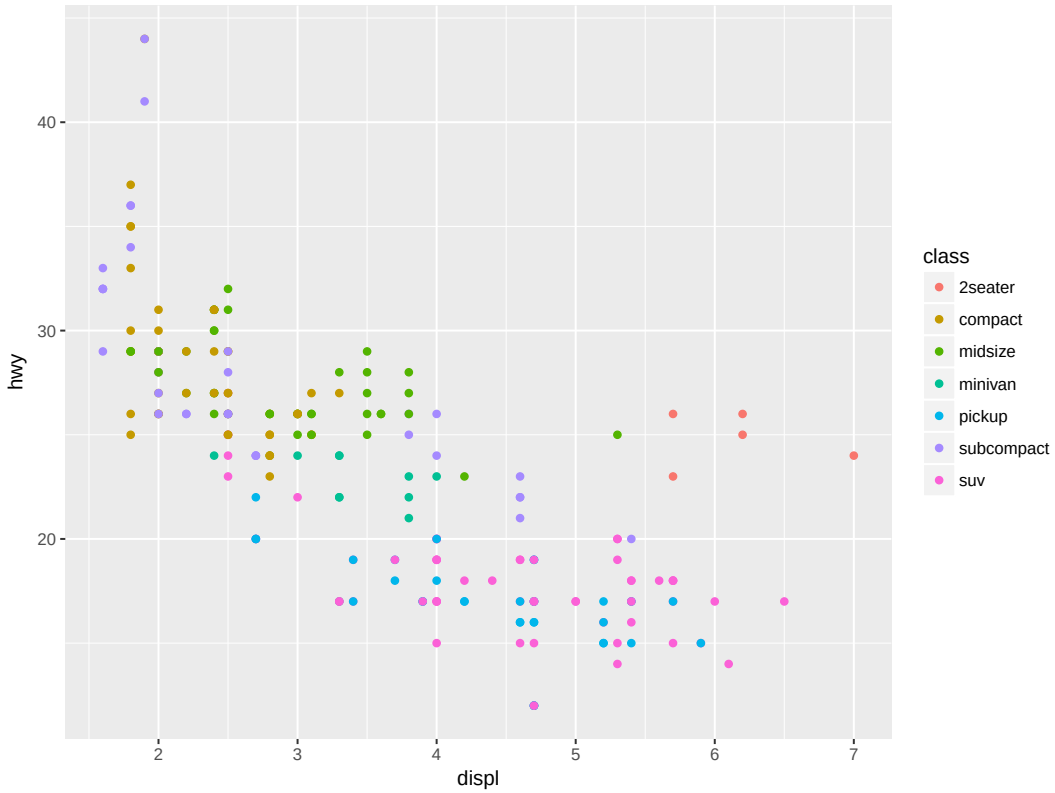
Geom: is the "type" of plot
Aesthetics: shape, colour, size, alpha
Faceting: "small multiples" displaying different subsets

Help is available. Try searching for examples, too.

- `had.co.nz/ggplot2`
- `had.co.nz/ggplot2/geom_point.html`

When certain aesthetics are defined, an appropriate legend is chosen and displayed automatically.

```
#### ggplot_mpg_displ_hwy_colour_class
p <- ggplot(mpg, aes(x = displ, y = hwy))
p <- p + geom_point(aes(colour = class))
print(p)
```



I encourage you to experiment with aesthetics!

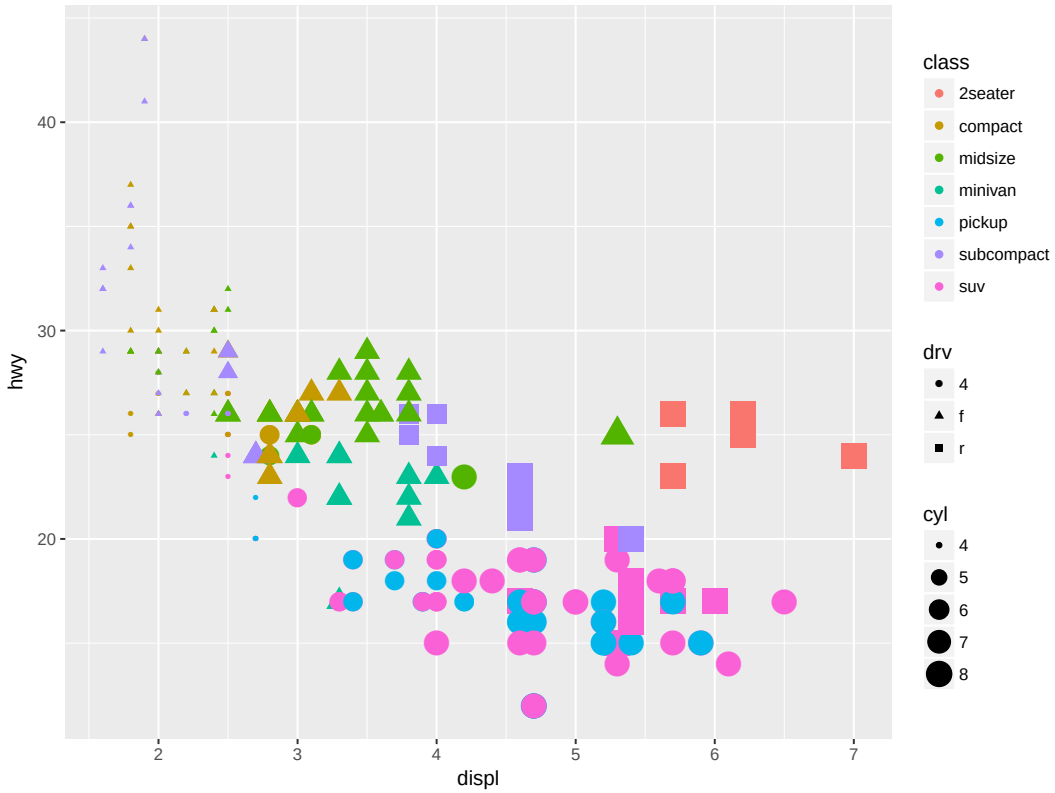
1. Assign variables to aesthetics colour, size, and shape.
2. What's the difference between discrete or continuous variables?
3. What happens when you combine multiple aesthetics?

The behavior of the aesthetics is predictable and customizable.

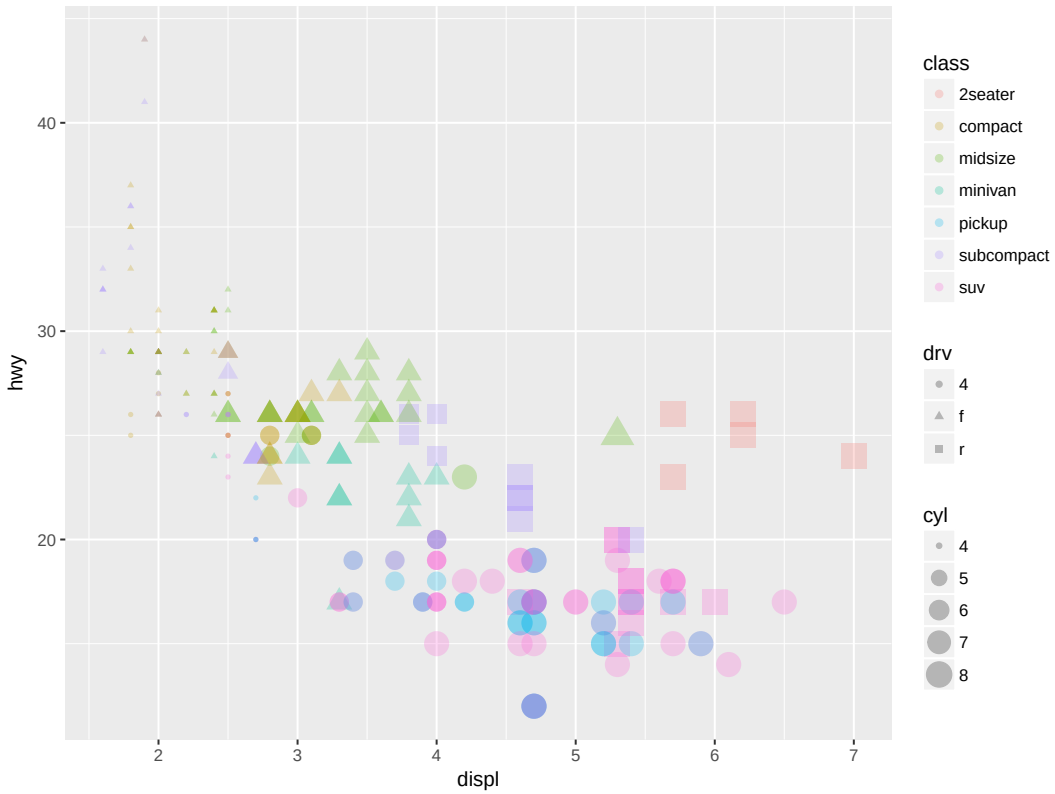
Aesthetic	Discrete	Continuous
colour	Rainbow of colors	Gradient from red to blue
size	Discrete size steps	Linear mapping between radius and value
shape	Different shape for each	Shouldn't work

Let's see a couple examples.

```
#### ggplot_mpg_displ_hwy_colour_class_size_cyl_shape_drv
p <- ggplot(mpg, aes(x = displ, y = hwy))
p <- p + geom_point(aes(colour = class, size = cyl, shape = drv))
print(p)
```




```
#### ggplot_mpg_displ_hwy_colour_class_size_cyl_shape_drv_alpha
p <- ggplot(mpg, aes(x = displ, y = hwy))
p <- p + geom_point(aes(colour = class, size = cyl, shape = drv)
                    , alpha = 1/4) # alpha is the opacity
print(p)
```



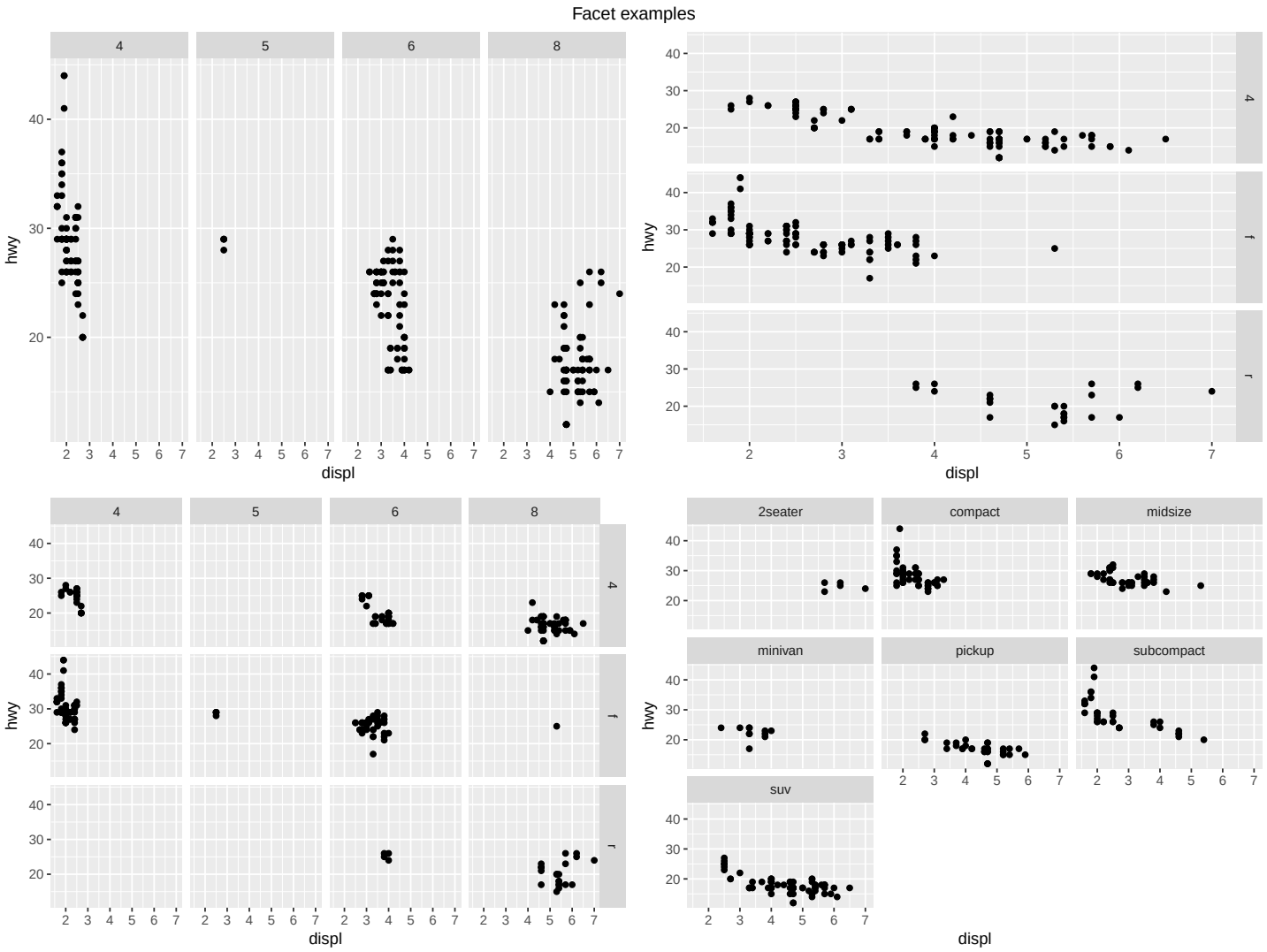
Faceting A small multiple[2] (sometimes called faceting, trellis chart, lattice chart, grid chart, or panel chart) is a series or grid of small similar graphics or charts, allowing them to be easily compared.

- Typically, small multiples will display different subsets of the data.
- Useful strategy for exploring conditional relationships, especially for large data.

Experiment with faceting of different types. What relationships would you like to see?

```
#### ggplot_mpg_displ_hwy_facet
# start by creating a basic scatterplot
p <- ggplot(mpg, aes(x = displ, y = hwy))
p <- p + geom_point()

## two methods
# facet_grid(rows ~ cols) for 2D grid, "." for no split.
# facet_wrap(~ var)       for 1D ribbon wrapped into 2D.

# examples of subsetting the scatterplot in facets
p1 <- p + facet_grid(. ~ cyl)     # columns are cyl categories
p2 <- p + facet_grid(drv ~ .)     # rows are drv categories
p3 <- p + facet_grid(drv ~ cyl)   # both
p4 <- p + facet_wrap(~ class)     # wrap plots by class category

# plot all four in one arrangement
library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4), ncol = 2, top="Facet examples")
```

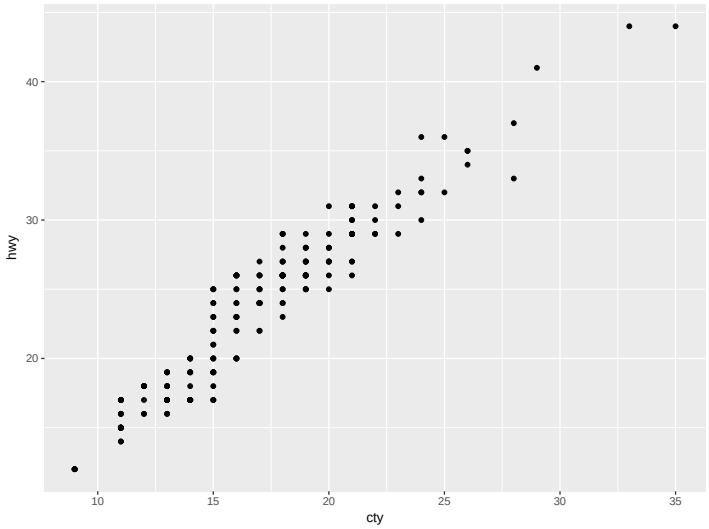
[2]According to Edward Tufte (Envisioning Information, p. 67): "At the heart of quantitative reasoning is a single question: Compared to what? Small multiple designs, multivariate and data bountiful, answer directly by visually enforcing comparisons of changes, of the differences among objects, of the scope of alternatives. For a wide range of problems in data presentation, small multiples are the best design solution."



0.2.1 Improving plots

How can this plot be improved?

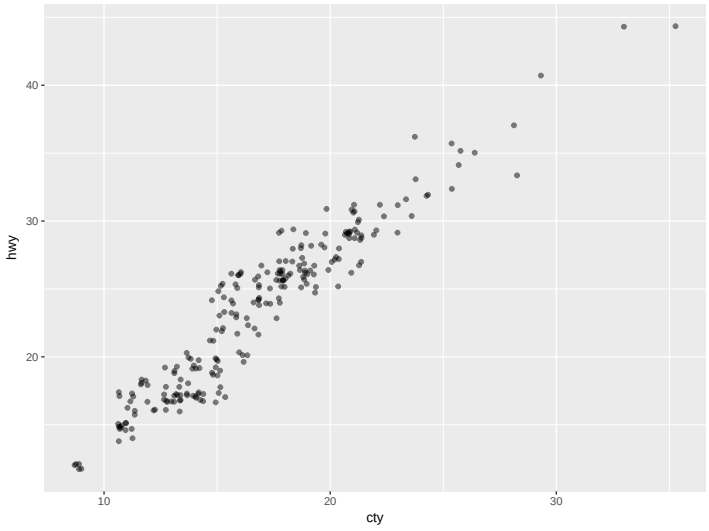
```
#### ggplot_mpg_cty_hwy
p <- ggplot(mpg, aes(x = cty, y = hwy))
p <- p + geom_point()
print(p)
```



Problem: points lie on top of each other, so it's impossible to tell how many observations each point represents.

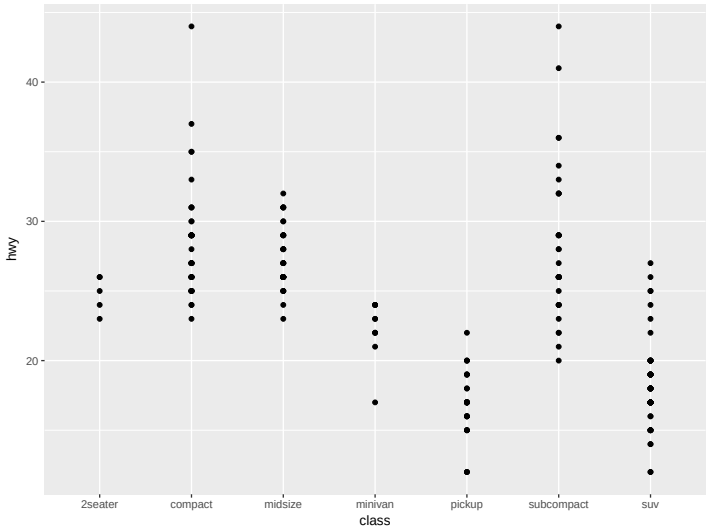
A solution: Jitter the points to reveal the individual points and reduce the opacity to 1/2 to indicate when points overlap.

```
#### ggplot_mpg_cty_hwy_jitter
p <- ggplot(mpg, aes(x = cty, y = hwy))
p <- p + geom_point(position = "jitter", alpha = 1/2)
print(p)
```



How can this plot be improved?

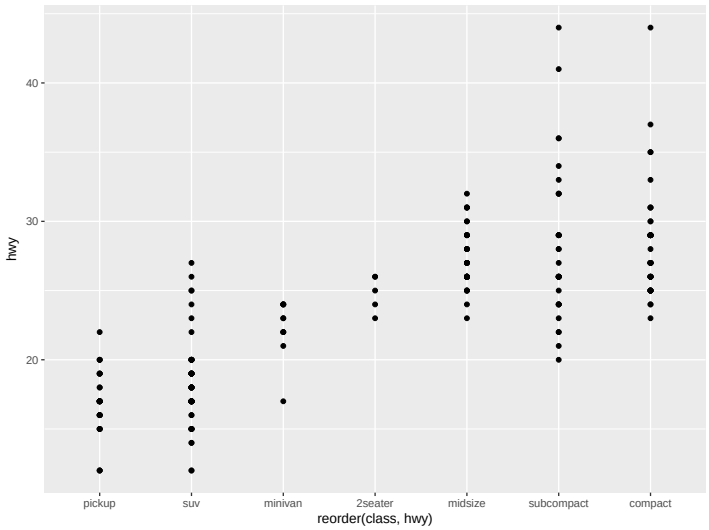
```
#### ggplot_mpg_class_hwy
p <- ggplot(mpg, aes(x = class, y = hwy))
p <- p + geom_point()
print(p)
```



Problem: The classes are in alphabetical order, which is somewhat arbitrary.

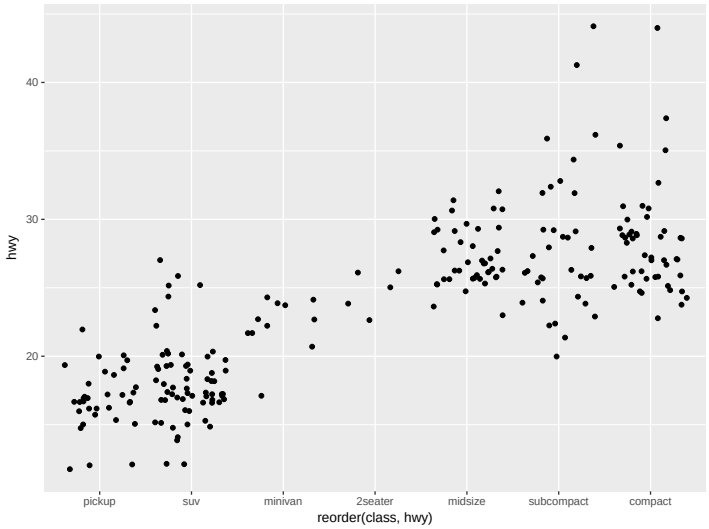
A solution: Reorder the class variable by the mean hwy for a meaningful ordering. Get help with `?reorder` to understand how this works.

```
#### ggplot_mpg_reorder_class_hwy
p <- ggplot(mpg, aes(x = reorder(class, hwy), y = hwy))
p <- p + geom_point()
print(p)
```



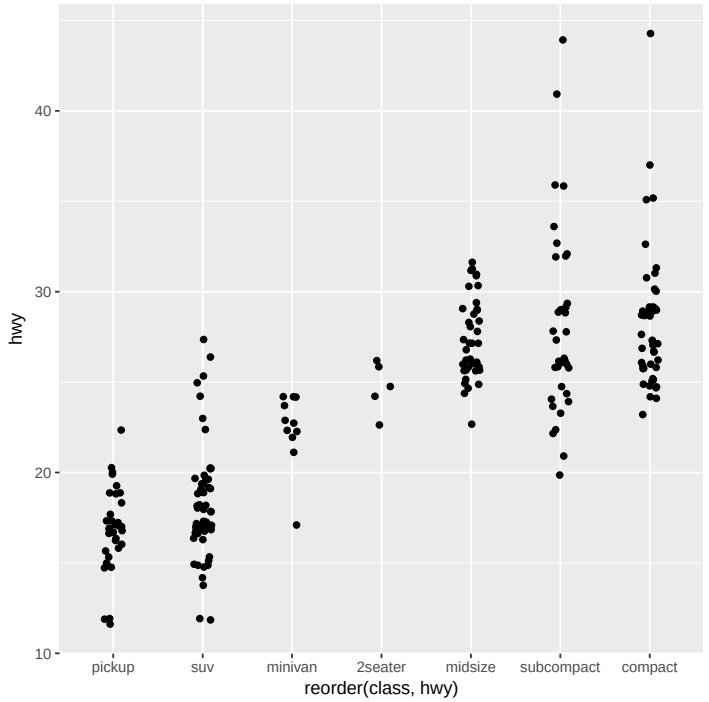
. . . add jitter

```
#### ggplot_mpg_reorder_class_hwy_jitter
p <- ggplot(mpg, aes(x = reorder(class, hwy), y = hwy))
p <- p + geom_point(position = "jitter")
print(p)
```



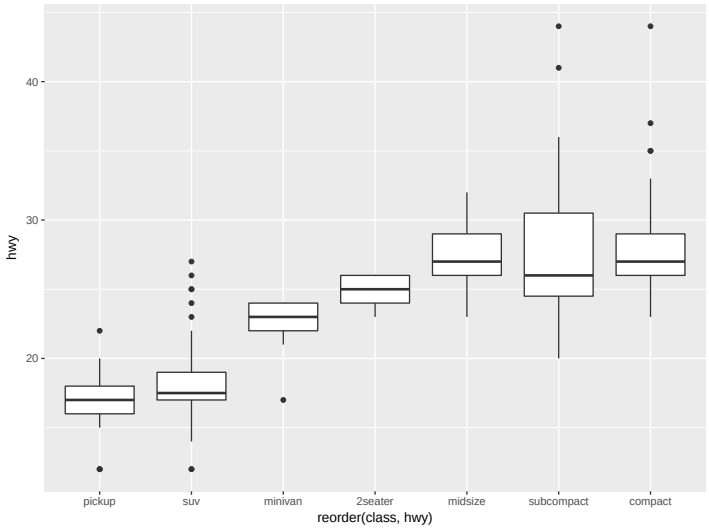
. . . a little less jitter

```
#### ggplot_mpg_reorder_class_hwy_jitter_less
p <- ggplot(mpg, aes(x = reorder(class, hwy), y = hwy))
p <- p + geom_jitter(position = position_jitter(width = 0.1))
print(p)
```



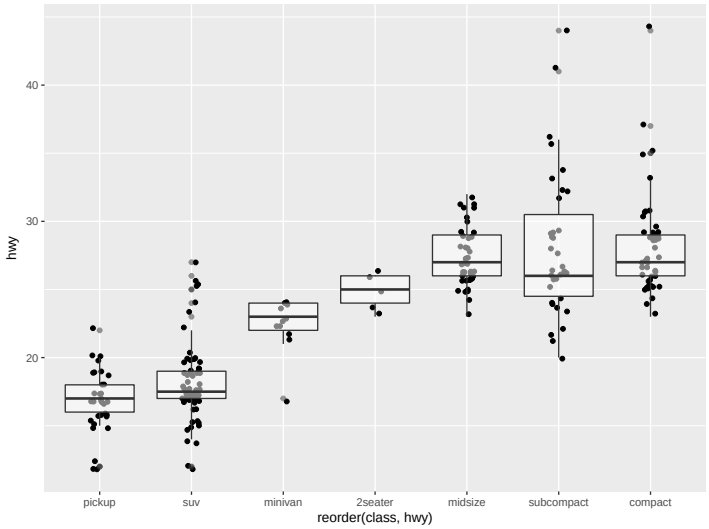
. . . or replace with boxplots

```
#### ggplot_mpg_reorder_class_hwy_boxplot
p <- ggplot(mpg, aes(x = reorder(class, hwy), y = hwy))
p <- p + geom_boxplot()
print(p)
```



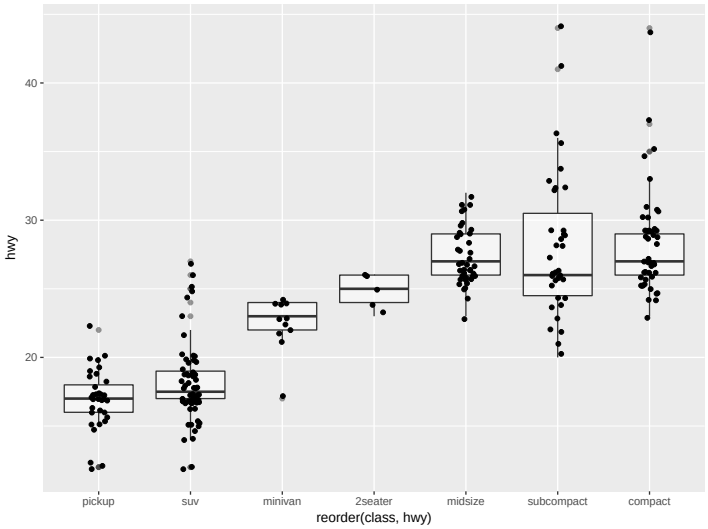
. . . or jitter those points with reduced-opacity boxplots on top

```
#### ggplot_mpg_reorder_class_hwy_jitter_boxplot
p <- ggplot(mpg, aes(x = reorder(class, hwy), y = hwy))
p <- p + geom_jitter(position = position_jitter(width = 0.1))
p <- p + geom_boxplot(alpha = 0.5)
print(p)
```



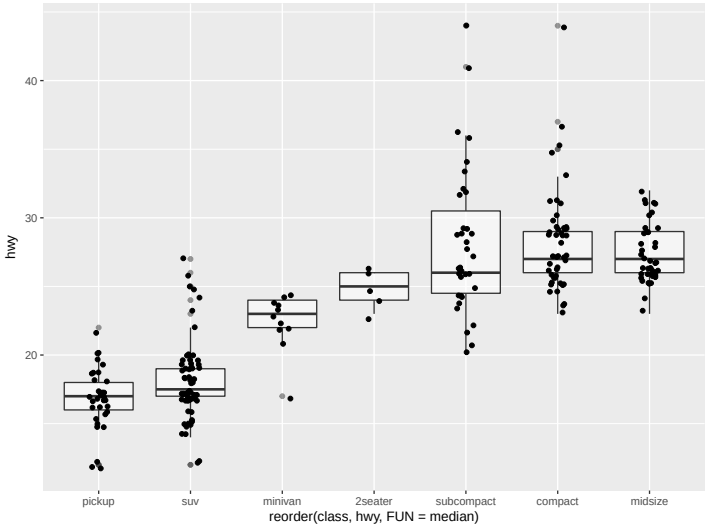
. . . even better with the boxplots with jittered points on top

```
#### ggplot_mpg_reorder_class_hwy_boxplot_jitter
p <- ggplot(mpg, aes(x = reorder(class, hwy), y = hwy))
p <- p + geom_boxplot(alpha = 0.5)
p <- p + geom_jitter(position = position_jitter(width = 0.1))
print(p)
```



. . . and can easily reorder by `median()` instead of `mean()` (mean is the default)

```
#### ggplot_mpg_reorder_class_hwy_boxplot_jitter_median
p <- ggplot(mpg, aes(x = reorder(class, hwy, FUN = median), y = hwy))
p <- p + geom_boxplot(alpha = 0.5)
p <- p + geom_jitter(position = position_jitter(width = 0.1))
print(p)
```



One-minute paper:

Muddy Any "muddy" points — anything that doesn't make sense yet?

Thumbs up Anything you really enjoyed or feel excited about?

0.3 Course Overview

See ADA2 Chapter 1 notes for a brief overview of all we'll cover in this semester of ADA1.

Part II

Summaries and displays, and one-, two-, and many-way tests of means

Chapter 1

Summarizing and Displaying Data

Contents


```
## Warning in file(filename, "r", encoding = encoding):  cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding):  cannot
open the connection
```

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.

■ CLICKER Qs — 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

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

```
#### 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

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

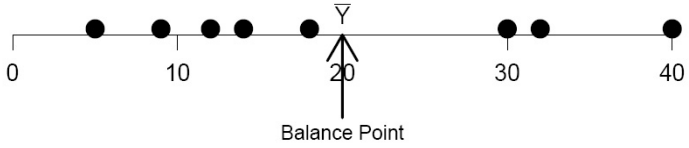
$$s = \sqrt{s^2} = 12.5.$$

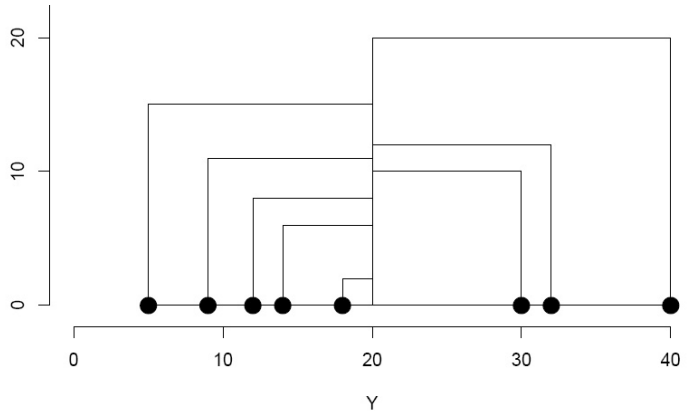
```
#### variance
var(y)
## [1] 156.2857
sd(y)
## [1] 12.50143
```

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

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

```
#### 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

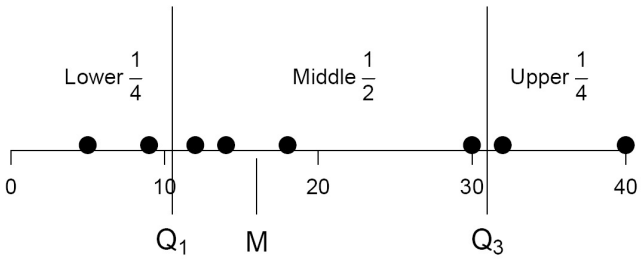
Q_3 = third quartile = median of upper half of data = 0.5(30+32) = 31 lb.

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 25$^{\text{th}}$ percentile, whereas the median and third quartiles are the 50$^{\text{th}}$ and 75$^{\text{th}}$ 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, 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
```

[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.

```
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 $\bar{Y}$.

■ CLICKER Qs — 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.

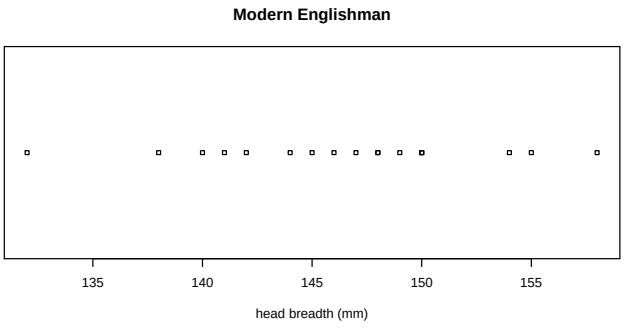
1.3.1 Dotplots

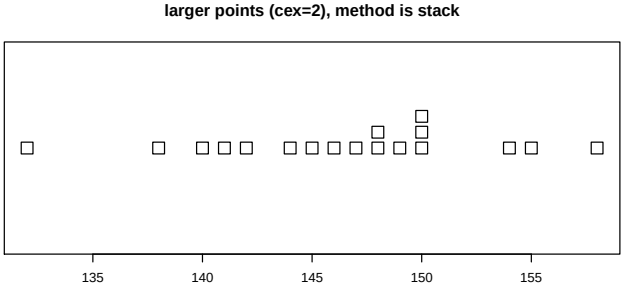
The **dotplot** breaks the range of data into many small-equal width intervals, and counts the number of observations in each interval. The interval count is superimposed on the number line at the interval midpoint as a series of dots, usually one for each observation. In the head breadth data, the intervals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

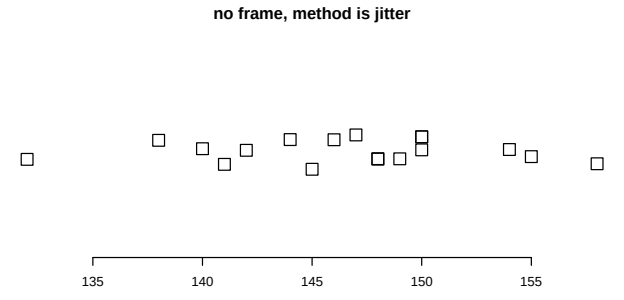
A dotplot of the head breadth data is given below. Of the examples below, the R base graphics `stripchart()` with `method="stack"` resembles the traditional dotplot.

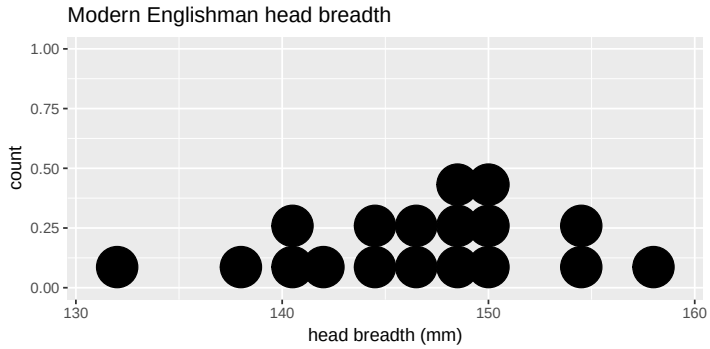
```
#### stripchart-ggplot
# stripchart (dotplot) using R base graphics
# 3 rows, 1 column
par(mfrow=c(3,1))
stripchart(hb, main="Modern Englishman", xlab="head breadth (mm)")
stripchart(hb, method="stack", cex=2
  , main="larger points (cex=2), method is stack")
stripchart(hb, method="jitter", cex=2,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

# 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)
```

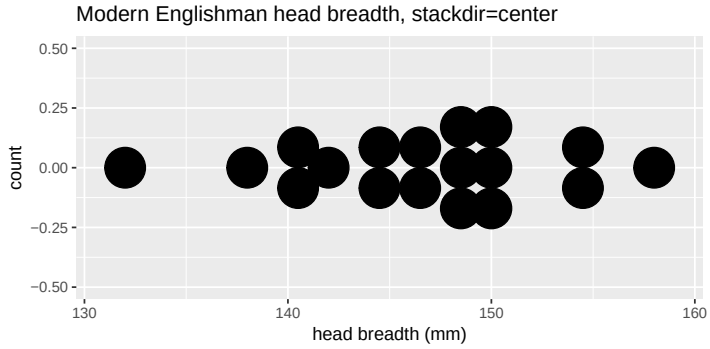
```
p1 <- p1 + labs(title = "Modern Englishman head breadth")
p1 <- p1 + xlab("head breadth (mm)")

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(grobs = list(p1, p2, p3), ncol=1)
```

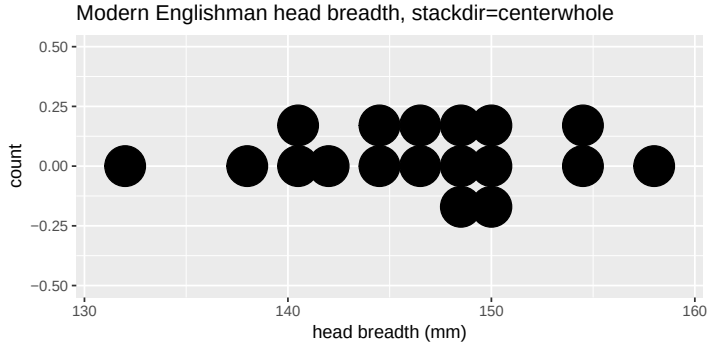








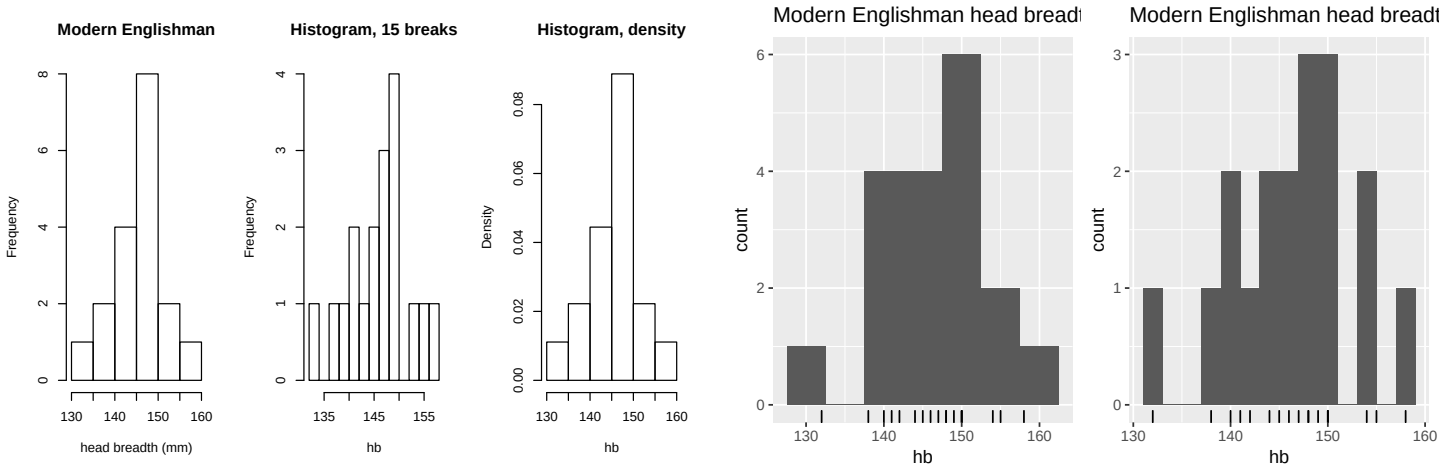




1.3.2 Histogram

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(grobs = list(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
```

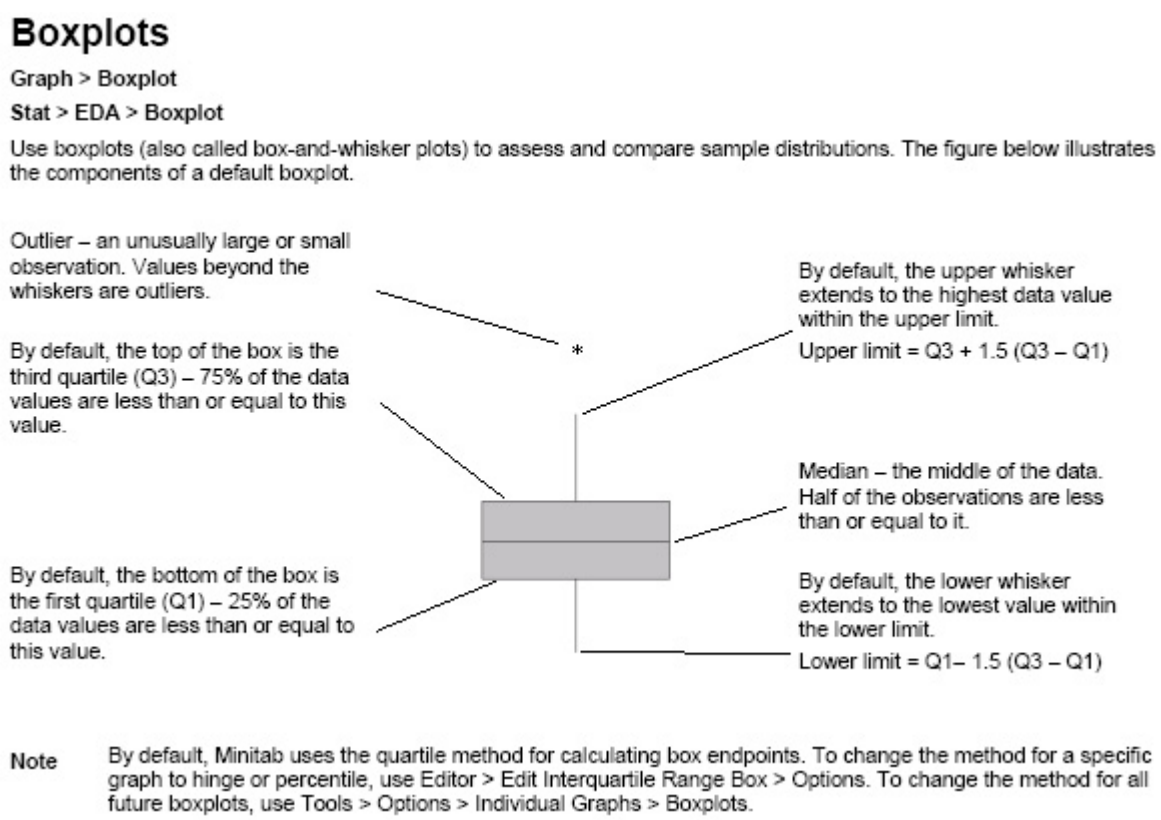
The data values are always *truncated* so that a leaf has one digit. The leaf unit (location of the decimal point) tells us the degree of round-off. This will become clearer in the next example.

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.

■ CLICKER Qs — 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.



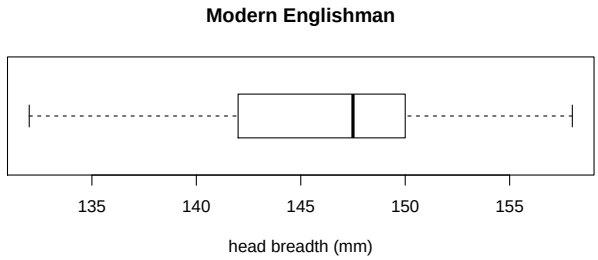
The endpoints of the box are placed at the locations of the first and third quartiles. The location of the median is identified by the line in the box. The whiskers extend to the data points closest to but not on or outside the outlier fences, which are $1.5IQR$ from the quartiles. Outliers are any values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

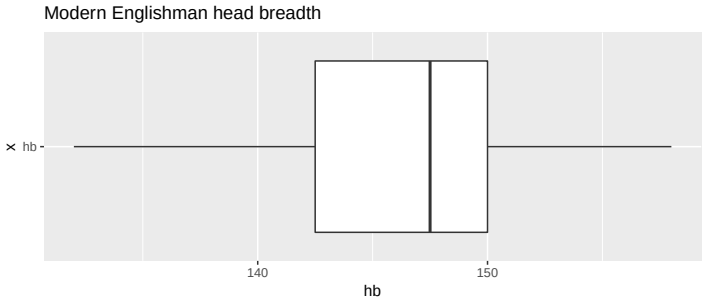
The boxplot for the head breadth data is given below. There are a lot of options that allow you to clutter the boxplot with additional information. Just use the default settings. We want to see the relative location of data (the median line), have an idea of the spread of data (IQR, the length of the 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)
```





■ CLICKER Qs — Boxplots ■

Improvements to the boxplot

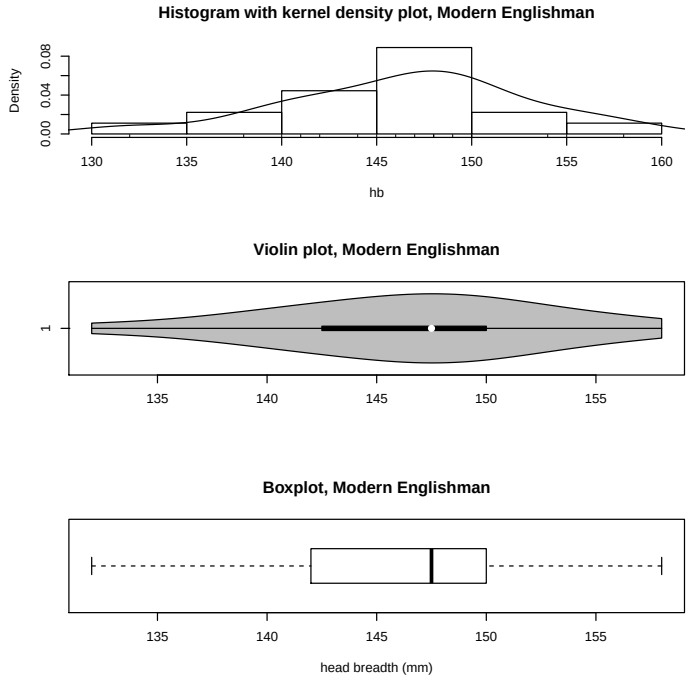
As a quick aside, a violin plot is a combination of a boxplot and a kernel density plot. They can be created using the `vioplot()` function from **vioplot** package.

```
#### 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)
```

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

```
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  100.92    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)
```

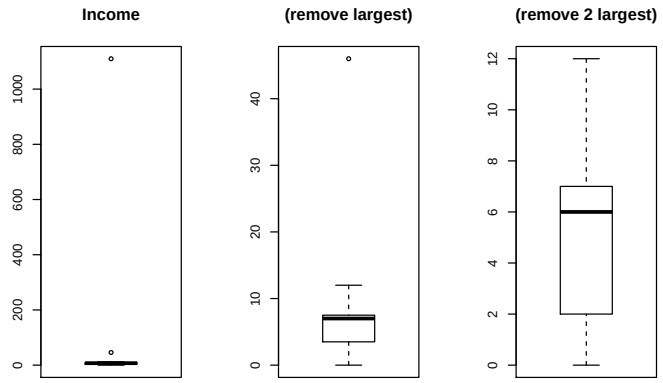
```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   0 | 022
##   0 | 557778
##   1 | 2
```

```
# 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
# 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.

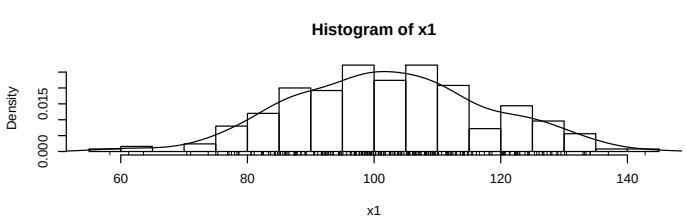
Unimodal, symmetric, bell-shaped, and no outliers The first display is **unimodal** (one peak), **symmetric**, and **bell-shaped** with no outliers. This is the prototypical normal curve. The boxplot (laid on its side for this display) shows strong evidence of symmetry: the median is about halfway between 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.

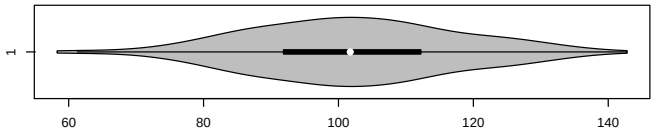
```
#### Unimodal, symmetric, bell-shaped, and no outliers (Normal distribution)
## base graphics
# sample from normal distribution
x1 <- rnorm(250, mean = 100, sd = 15)

par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
```

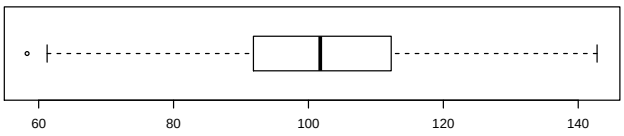
[2] "Distribution function" often refers to the "cumulative distribution function", which is a different (but one-to-one related) function than what I mean here.

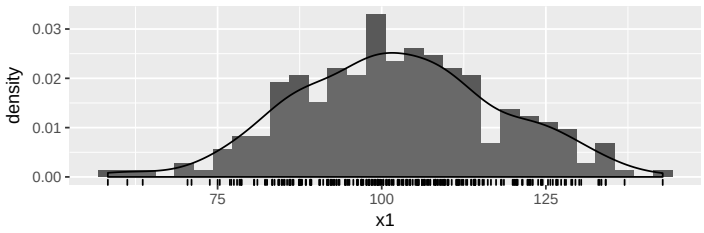
```
hist(x1, freq = FALSE, breaks = 20)
points(density(x1), type = "l")
rug(x1)

# 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(grobs = list(p1, p2, p3), ncol=1)
```

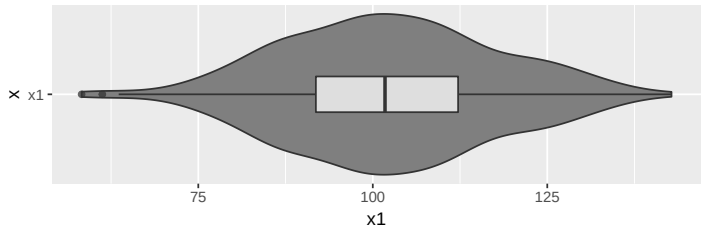
```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```

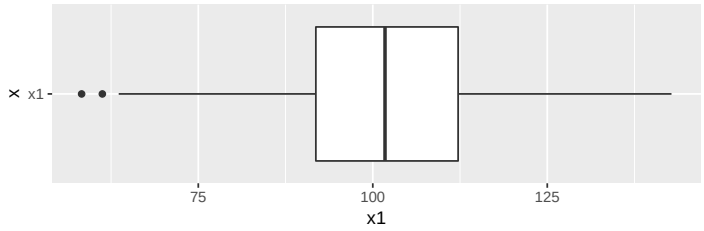













```
#### Central statistical moments
# moments package for 3rd and 4th moments: skewness() and kurtosis()
library(moments)
summary(x1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   58.28   91.84  101.75  102.37  112.23  142.80
```

```
sd(x1)
```

```
## [1] 15.29373
```

```
skewness(x1)
```

```
## [1] 0.01891139
```

```
kurtosis(x1)
```

```
## [1] 2.780282
```

```
stem(x1)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    5 | 8
##    6 | 14
##    6 |
##    7 | 014
##    7 | 5577788999
##    8 | 1112233334444
##    8 | 55556666677777788888899999
##    9 | 011122222222333333444
##    9 | 555566667777777888888888999999999
##   10 | 000000011111112222333333444444444
##   10 | 555556666666777777888889999999
```

```
##   11 | 00000001112222223344444444
##   11 | 5556677889
##   12 | 000011111223333444
##   12 | 556677778899
##   13 | 00033344
##   13 | 7
##   14 | 3
```

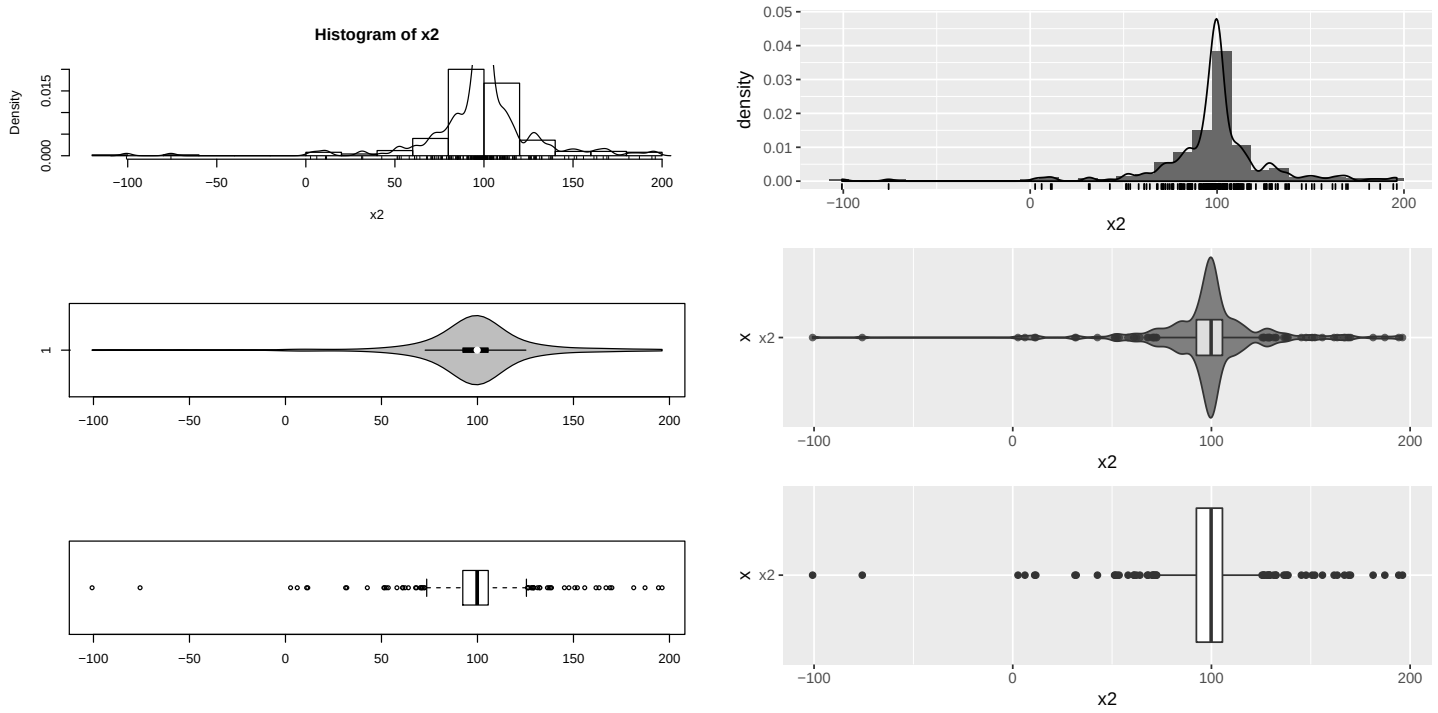
Unimodal, symmetric, heavy-tailed The boxplot is better at highlighting outliers than are other displays. The histogram and stem-and-leaf displays below appear to have the same basic shape as a normal curve (unimodal, symmetric). However, the boxplot shows that we have a dozen outliers in a sample of 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(grobs = list(p1, p2, p3), ncol=1)
```

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```




```
summary(x2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -100.72   92.42   99.84   98.47  105.54  196.16
```

```
sd(x2)
```

```
## [1] 31.12554
```

```
skewness(x2)
```

```
## [1] -1.58673
```

```
kurtosis(x2)
```

```
## [1] 14.19473
```

```
stem(x2)
```

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

```
##
##   -10 | 1
##    -8 |
##    -6 | 6
##    -4 |
##    -2 |
##    -0 |
##     0 | 3612
##     2 | 12
##     4 | 312348
##     6 | 1124880111224456679
##     8 | 000112234455555566677880111112333334444555555556666666777777777888+4
##    10 | 000000000000000000000000000011111111111111111222222233344444444555+21
##    12 | 155667889991236789
##    14 | 58126
##    16 | 23790
##    18 | 1746
```

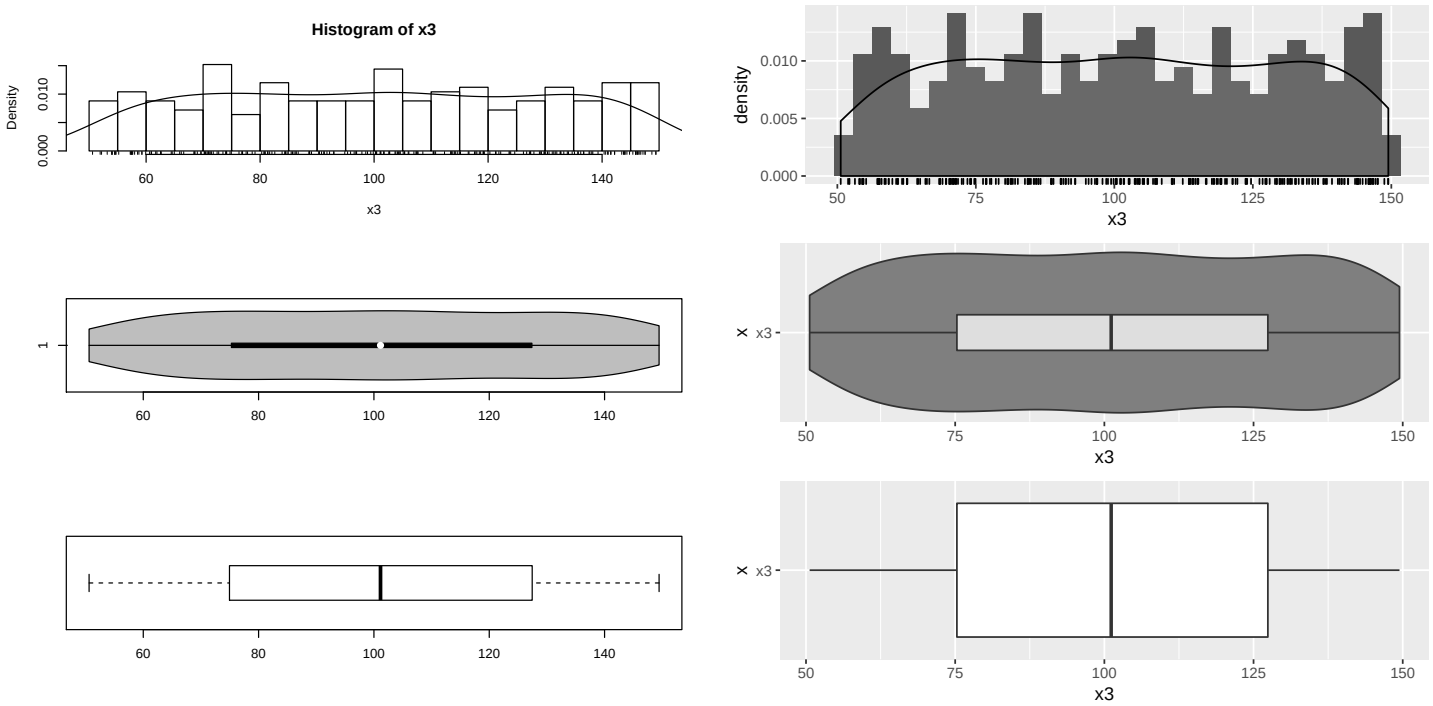
Symmetric, (uniform,) short-tailed Not all symmetric distributions are mound-shaped, as the display below suggests. The boxplot shows symmetry, but the tails of the distribution are shorter (lighter) than in the normal distribution. Note that the distance between quartiles is roughly constant here.

```
#### 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(grobs = list(p1, p2, p3), ncol=1)
```

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```




```
summary(x3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   50.61   75.29  101.14  101.06  127.40  149.46
```

```
sd(x3)
```

```
## [1] 28.90485
```

```
skewness(x3)
```

```
## [1] 0.0001547997
```

```
kurtosis(x3)
```

```
## [1] 1.783274
```

```
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 | 0111112234444
##    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 | 5566666677788999
```

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.

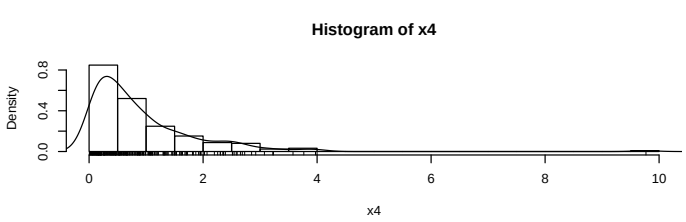
Unimodal, skewed right The distribution below is unimodal, and asymmetric or **skewed**. The distribution is said to be **skewed to the right**, or upper end, because the right tail is much longer than the left tail. The boxplot also shows the skewness – the region between the minimum observation and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

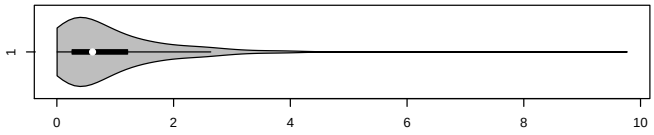
```
#### Unimodal, skewed right
# sample from exponential distribution
x4 <- rexp(250, rate = 1)
```

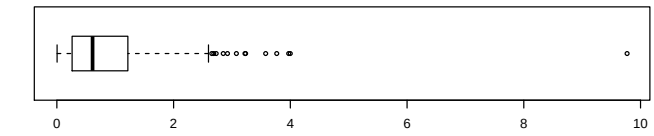
```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
```

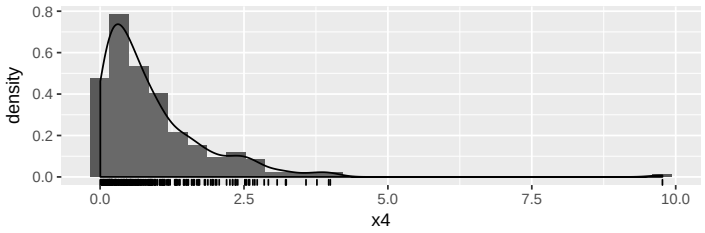
```r
hist(x4, freq = FALSE, breaks = 20)
points(density(x4), type = "l")
rug(x4)

# 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(grobs = list(p1, p2, p3), ncol=1)
```

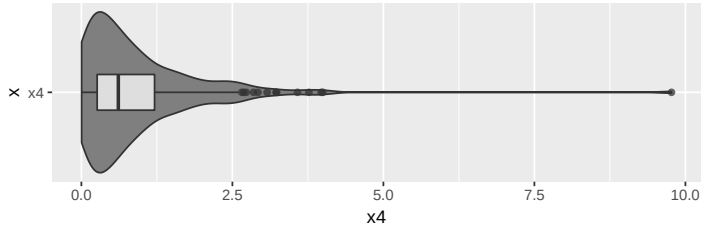
```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```

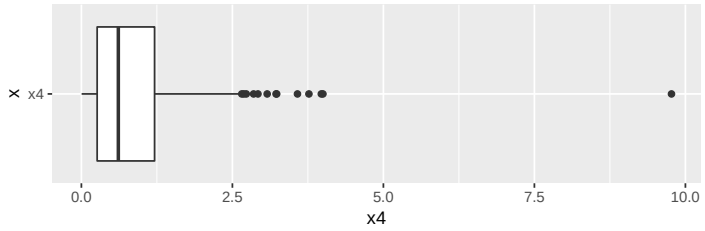













```
summary(x4)
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.003949 0.261104 0.611573 0.914406 1.211395 9.769742
sd(x4)
## [1] 1.001675
skewness(x4)
## [1] 3.538039
kurtosis(x4)
## [1] 26.81341
stem(x4)
##
##   The decimal point is at the |
##
##   0 | 000000000011111111111111111111112222222222222222222222222333333333+15
##   0 | 55555555555555555555566666666666666677777777777777888888889999999999
##   1 | 00000011111111111222233333444
##   1 | 5555555566666777788999
##   2 | 0000123333444
##   2 | 5556677799
##   3 | 122
##   3 | 68
##   4 | 00
##   4 |
##   5 |
##   5 |
##   6 |
##   6 |
```

```
##   7 |
##   7 |
##   8 |
##   8 |
##   9 |
##   9 | 8
```

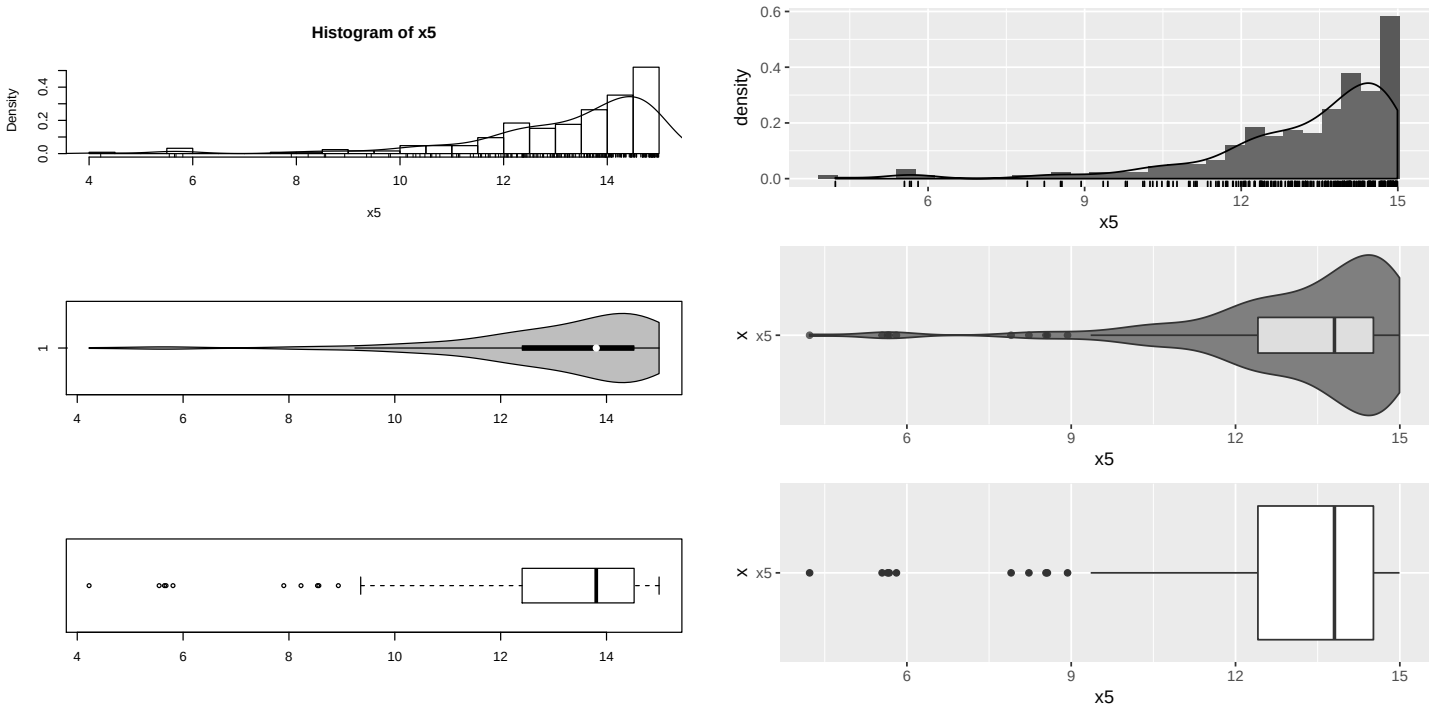
Unimodal, skewed left The distribution below is unimodal and **skewed to the left**. The two examples show that extremely skewed distributions often contain outliers in the longer tail of the distribution.

```
#### 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(grobs = list(p1, p2, p3), ncol=1)
```

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```




```
summary(x5)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.224  12.409  13.806  13.203  14.518  14.994
```

```
sd(x5)
```

```
## [1] 1.862082
```

```
skewness(x5)
```

```
## [1] -1.995207
```

```
kurtosis(x5)
```

```
## [1] 8.082068
```

```
stem(x5)
```

```
##
##   The decimal point is at the |
##
##    4 | 2
##    5 | 5678
##    6 |
##    7 | 9
##    8 | 2569
##    9 | 4488
##   10 | 11334566678
```

```
##   11 | 001124456677778899
##   12 | 000111122333444444444455666667788889999
##   13 | 00000011112222233334455566666666777777888889999999
##   14 | 000000000011111111112222222333333333344444455555555555566666777777+26
##   15 | 000000000
```

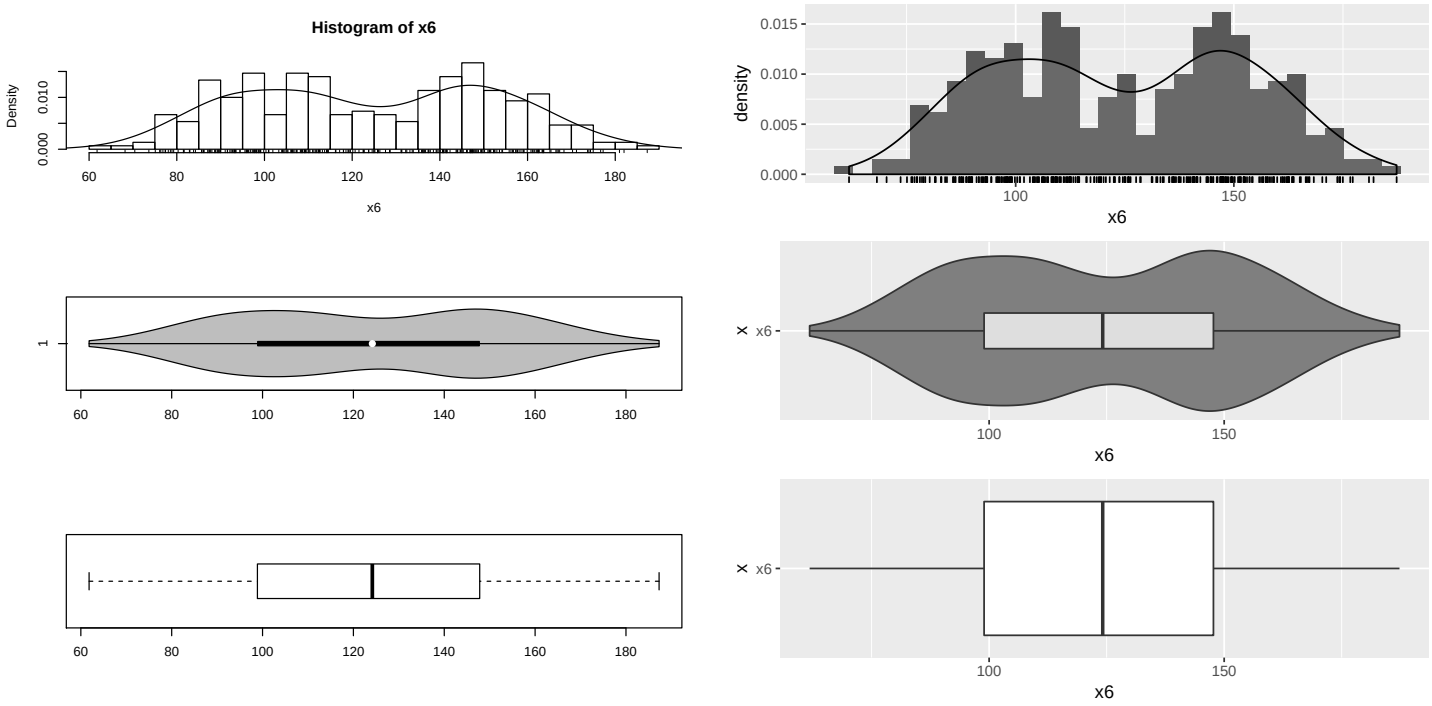
Bimodal (multi-modal) Not all distributions are unimodal. The distribution below has two modes or peaks, and is said to be **bimodal**. Distributions with three or more peaks are called **multi-modal**.

```
#### 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(grobs = list(p1, p2, p3), ncol=1)
```

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```




```
summary(x6)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   61.82   98.93  124.16  124.38  147.72  187.29
```

```
sd(x6)
```

```
## [1] 28.39398
```

```
skewness(x6)
```

```
## [1] 0.009062991
```

```
kurtosis(x6)
```

```
## [1] 1.89301
```

```
stem(x6)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    6 | 28
##    7 | 045667778999
##    8 | 022233446666778888999999
##    9 | 000001222233333334466666777888888899999999
##   10 | 00112344455556666777788899999999
##   11 | 000011222222333344444566788999
##   12 | 00012233445555666666899
##   13 | 11122344556666677778999
##   14 | 000111111111122244445555566777777777888888999999
```

```
##   15 | 000111122222334444667778889999
##   16 | 01112222233333445567778
##   17 | 014444577
##   18 | 127
```

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.

■ CLICKER Qs — 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.

Chapter 2

Estimation in One-Sample Problems

Contents

```
## Warning in file(filename, "r", encoding = encoding):  cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding):  cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

select graphical displays that meaningfully communicate properties of a sample.

assess the assumptions of the one-sample t-test visually.

decide whether the mean of a population is different from a hypothesized value.

recommend action based on a hypothesis test.

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

1. organize knowledge.

5. define parameters of interest and hypotheses in words and notation.

6. summarize data visually, numerically, and descriptively.

8. use statistical software.

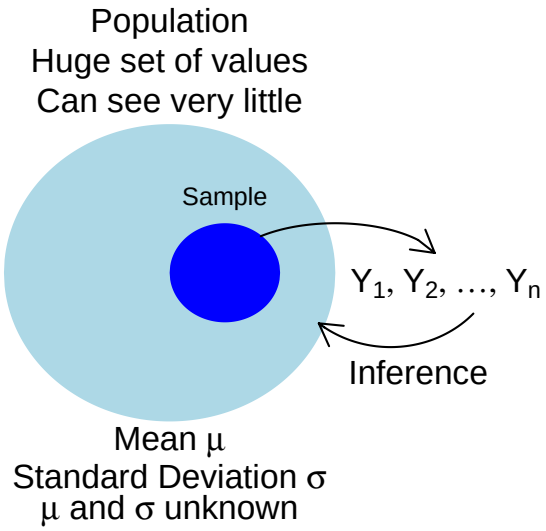
12. make evidence-based decisions.

2.1 Inference for a population mean

Suppose that you have identified a population of interest where individuals are measured on a single quantitative characteristic, say, weight, height or IQ. You select a random or representative sample from the population with the goal of estimating the (unknown) **population mean** value, identified by μ. You cannot see much of the population, but you would like to know what is typical in the population (μ). The only information you can see is that in the sample.

This is a standard problem in statistical inference, and the first inferential problem that we will tackle. For notational convenience, identify the measure-

ments on the sample as $Y_1, Y_2, \ldots, Y_n$, where n is the sample size. Given the data, our best guess, or estimate, of μ is the sample mean: $\bar{Y} = \frac{\sum_i Y_i}{n} = \frac{Y_1+Y_2+\cdots+Y_n}{n}$.



There are two main methods that are used for inferences on μ: **confidence intervals** (CI) and **hypothesis tests**. The standard CI and test procedures are based on the sample mean and the sample standard deviation, denoted by s.

■ CLICKER Qs — Inference for a population mean, 2 ■

2.1.1 Standard error, LLN, and CLT

The **standard error (SE)** is the standard deviation of the **sampling distribution** of a statistic.

The **sampling distribution** of a statistic is the distribution of that statistic, considered as a random variable, when derived from a random sample of size n.

The **standard error of the mean (SEM)** is the standard deviation of the sample-mean's estimate of a population mean. (It can also be viewed as the standard deviation of the error in the sample mean relative to the true mean, since the sample mean is an unbiased estimator.) SEM is usually estimated by the sample estimate of the population standard deviation (sample standard deviation) divided by the square root of the sample size (assuming statistical independence of the values in the sample):

$$SE_{\bar{Y}} = s/\sqrt{n}$$

where s is the sample standard deviation (i.e., the sample-based estimate of the standard deviation of the population), and n is the size (number of observations) of the sample.

In probability theory, the **law of large numbers (LLN)** is a theorem that describes the result of performing the same experiment a large number of times. According to the law, the average of the results obtained from a large number of trials (the sample mean, $\bar{Y}$) should be close to the expected value (the population mean, μ), and will tend to become closer as more trials are performed.

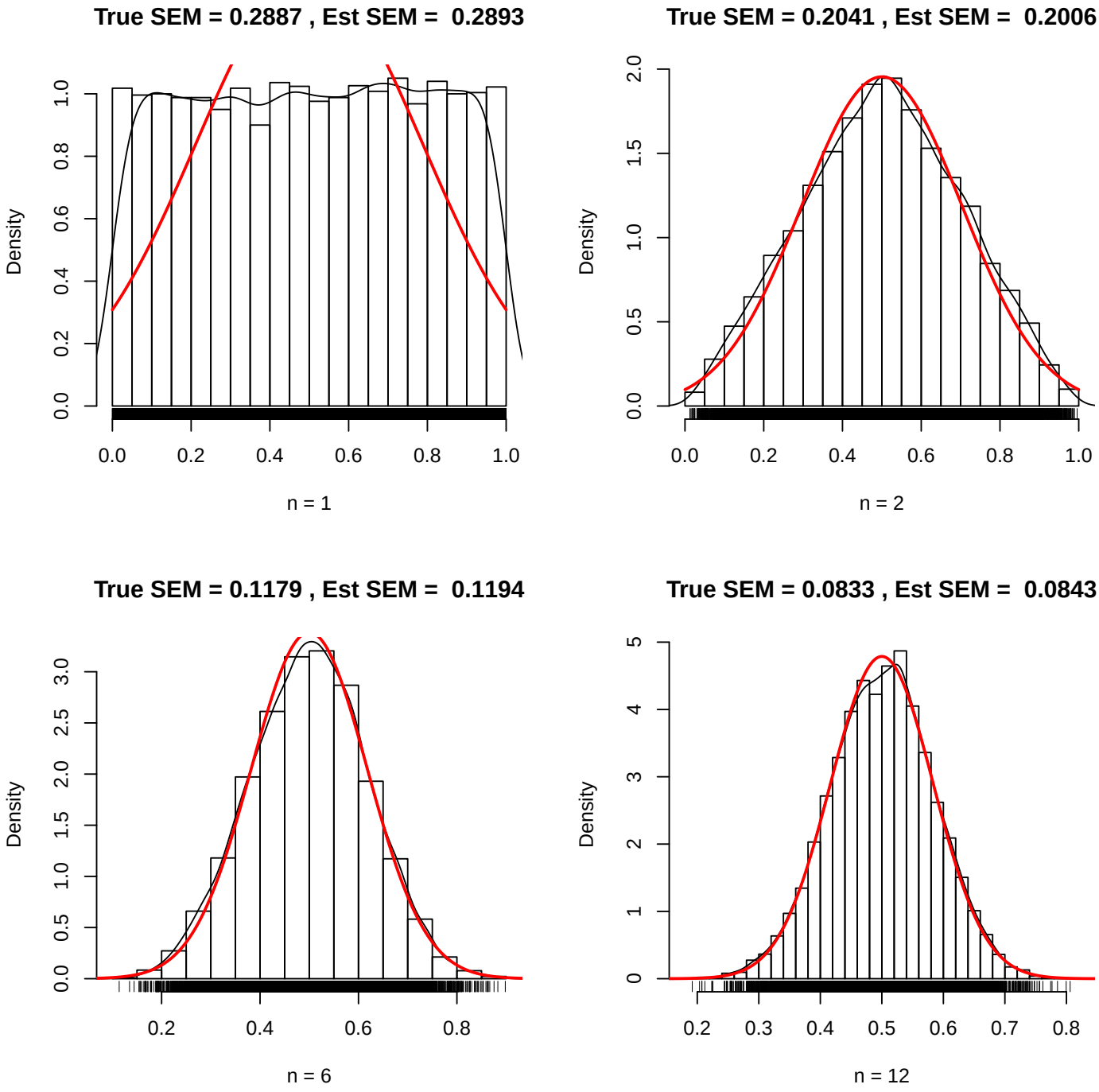
In probability theory, the **central limit theorem (CLT)** states that, given certain conditions, the mean of a sufficiently large number of independent random variables, each with finite mean and variance, will be approximately normally distributed[1].

As a joint illustration of these concepts, consider drawing random variables following a Uniform(0,1) distribution, that is, any value in the interval $[0, 1]$ is equally likely. By definition, the mean of this distribution is $\mu = 1/2$ and the variance is $\sigma^2 = 1/12$ (so the standard deviation is $\sigma = \sqrt{1/12} = 0.289$). Therefore, if we draw a sample of size n, then the standard error of the mean will be $\sigma/\sqrt{n}$, and as n gets larger the distribution of the mean will increasingly follow a normal distribution. We illustrate this by drawing $N = 10000$ samples

[1]The central limit theorem has a number of variants. In its common form, the random variables must be identically distributed. In variants, convergence of the mean to the normal distribution also occurs for non-identical distributions, given that they comply with certain conditions.

of size n and plot those N means, computing the expected and observed SEM and how well the histogram of sampled means follows a normal distribution, Notice, indeed, that even with samples as small as 2 and 6 that the properties of the SEM and the distribution are as predicted.

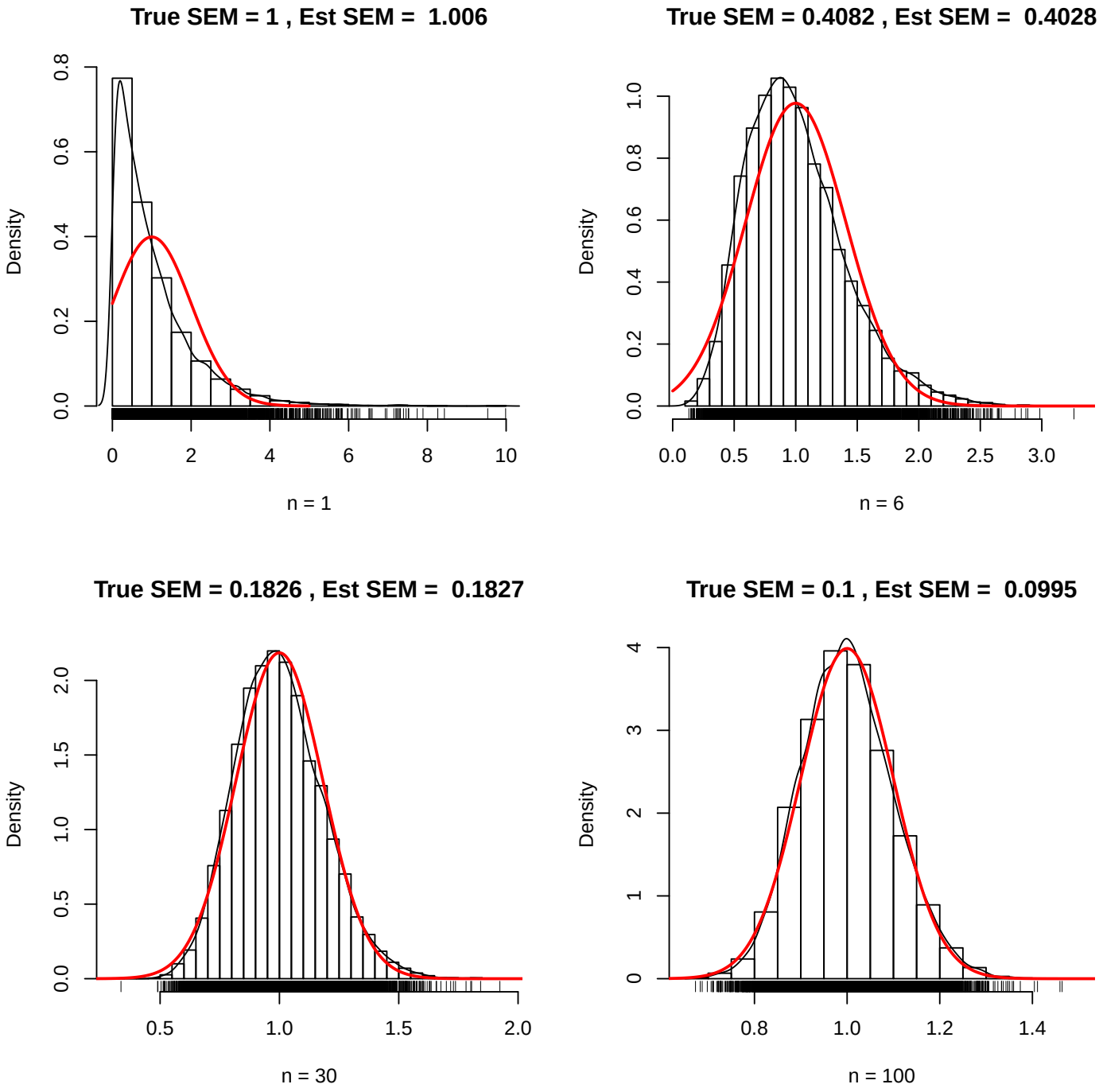
```
#### Illustration of Central Limit Theorem, Uniform distribution
# demo.clt.unif(N, n)
#   draws N samples of size n from Uniform(0,1)
#   and plots the N means with a normal distribution overlay
demo.clt.unif <- function(N, n) {
  # draw sample in a matrix with N columns and n rows
  sam <- matrix(runif(N*n, 0, 1), ncol=N);
  # calculate the mean of each column
  sam.mean <- colMeans(sam)
  # the sd of the mean is the SEM
  sam.se <- sd(sam.mean)
  # calculate the true SEM given the sample size n
  true.se <- sqrt((1/12)/n)
  # draw a histogram of the means
  hist(sam.mean, freq = FALSE, breaks = 25
      , main = paste("True SEM =", round(true.se, 4)
                  , ", Est SEM = ", round( sam.se, 4))
      , xlab = paste("n =", n))
  # overlay a density curve for the sample means
  points(density(sam.mean), type = "l")
  # overlay a normal distribution, bold and red
  x <- seq(0, 1, length = 1000)
  points(x, dnorm(x, mean = 0.5, sd = true.se), type = "l", lwd = 2, col = "red")
  # place a rug of points under the plot
  rug(sam.mean)
}

par(mfrow=c(2,2));
demo.clt.unif(10000,  1);
demo.clt.unif(10000,  2);
demo.clt.unif(10000,  6);
demo.clt.unif(10000, 12);
```



In a more extreme example, we draw samples from an Exponential(1) distribution ($\mu = 1$ and $\sigma = 1$), which is strongly skewed to the right. Notice that the normality promised by the CLT requires larger samples sizes, about $n \geq 30$, than for the previous Uniform(0,1) example, which required about $n \geq 6$.

```
#### Illustration of Central Limit Theorem, Exponential distribution
# demo.clt.exp(N, n) draws N samples of size n from Exponential(1)
# and plots the N means with a normal distribution overlay
demo.clt.exp <- function(N, n) {
```

```
  # draw sample in a matrix with N columns and n rows
  sam <- matrix(rexp(N*n, 1), ncol=N);
  # calculate the mean of each column
  sam.mean <- colMeans(sam)
  # the sd of the mean is the SEM
  sam.se <- sd(sam.mean)
  # calculate the true SEM given the sample size n
  true.se <- sqrt(1/n)
  # draw a histogram of the means
  hist(sam.mean, freq = FALSE, breaks = 25
       , main = paste("True SEM =", round(true.se, 4), ", Est SEM = ", round(sam.se, 4))
       , xlab = paste("n =", n))
  # overlay a density curve for the sample means
  points(density(sam.mean), type = "l")
  # overlay a normal distribution, bold and red
  x <- seq(0, 5, length = 1000)
  points(x, dnorm(x, mean = 1, sd = true.se), type = "l", lwd = 2, col = "red")
  # place a rug of points under the plot
  rug(sam.mean)
}

par(mfrow=c(2,2));
demo.clt.exp(10000,   1);
demo.clt.exp(10000,   6);
demo.clt.exp(10000,  30);
demo.clt.exp(10000, 100);
```



Note well that the further the population distribution is from being normal, the larger the sample size is required to be for the sampling distribution of the sample mean to be normal. If the population distribution is normal, what's the minimum sample size for the sampling distribution of the mean to be normal?

For more examples, try:

```
#### More examples for Central Limit Theorem can be illustrated with this code
# install.packages("TeachingDemos")
library(TeachingDemos)
# look at examples at bottom of the help page
?clt.examp
```

2.1.2 z-score

Given a distribution with mean $\bar{x}$ and standard deviation s, a location-scale transformation known as a z-score will shift the distribution to have mean 0 and scale the spread to have standard deviation 1:

$$z = \frac{x - \bar{x}}{s}.$$

Below, the original variable x has a normal distribution with mean 100 and standard deviation 15, Normal$(100, 15^2)$, and z has a Normal$(0, 1)$ distribution.

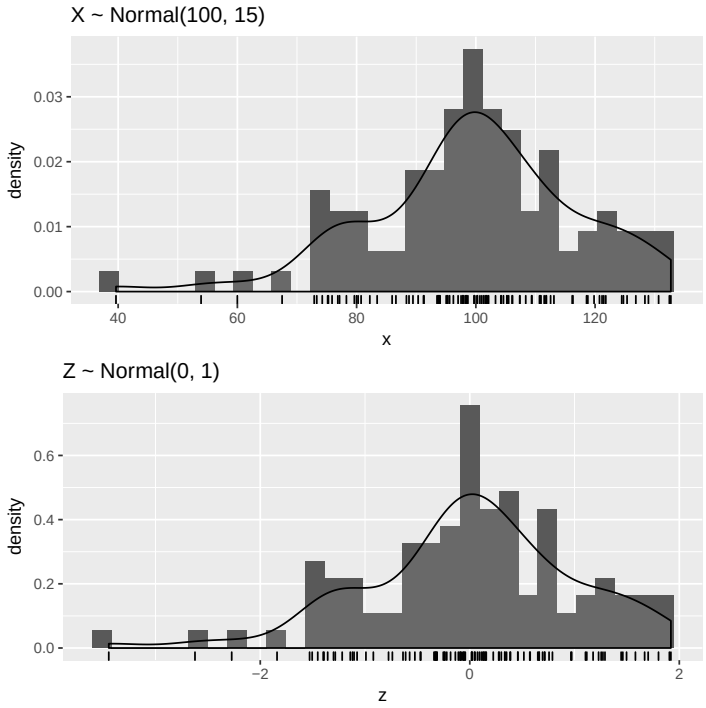
```
# sample from normal distribution
df <- data.frame(x = rnorm(100, mean = 100, sd = 15))
df$z <- scale(df$x)  # by default, this performs a z-score transformation

summary(df)
```

```
##        x                 z.V1
##  Min.   : 39.64   Min.   :-3.446123
##  1st Qu.: 90.99   1st Qu.:-0.485300
##  Median :100.00   Median : 0.033925
##  Mean   : 99.41   Mean   : 0.000000
##  3rd Qu.:110.72   3rd Qu.: 0.652006
##  Max.   :132.70   Max.   : 1.919736
```

```
## ggplot
library(ggplot2)
p1 <- ggplot(df, aes(x = x))
  # 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()
p1 <- p1 + labs(title = "X ~ Normal(100, 15)")

p2 <- ggplot(df, aes(x = z))
  # Histogram with density instead of count on y-axis
p2 <- p2 + geom_histogram(aes(y=..density..))
p2 <- p2 + geom_density(alpha=0.1, fill="white")
```

```
p2 <- p2 + geom_rug()
p2 <- p2 + labs(title = "Z ~ Normal(0, 1)")

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

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```

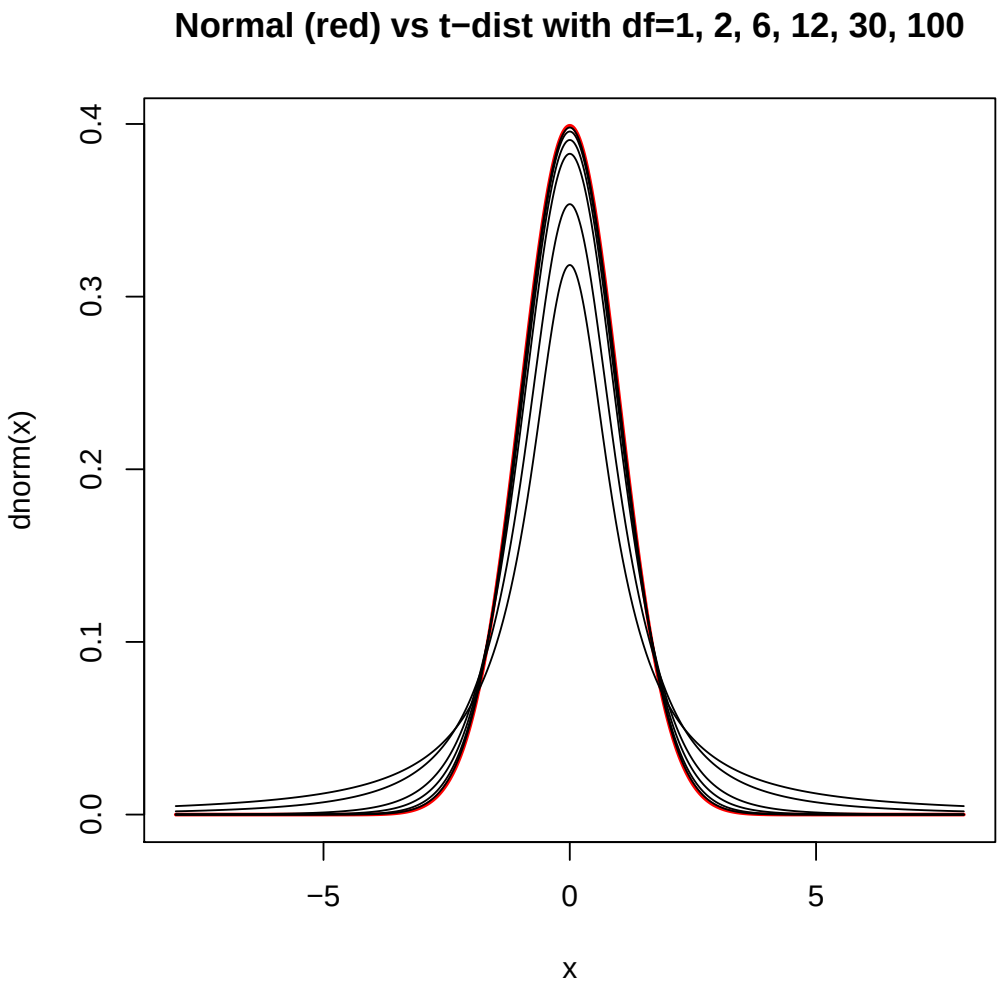


2.1.3 t-distribution

The Student's **t-distribution** is a family of continuous probability distributions that arises when estimating the mean of a normally distributed population in situations where the *sample size is small* and population *standard deviation is unknown*. The t-distribution is symmetric and bell-shaped, like the normal distribution, but has heavier tails, meaning that it is more prone to producing values that fall far from its mean. Effectively, the t-distribution is wider than the normal distribution because in addition to estimating the mean μ with $\bar{Y}$, we *also* have to estimate σ^2 with s^2, so there's some additional uncertainty. The degrees-of-freedom (df) parameter of the t-distribution is the sample size n minus the number of variance parameters estimated. Thus, $df = n - 1$ when we have one sample and $df = n - 2$ when we have two samples. As n increases,

the t-distribution becomes close to the normal distribution, and when $n = \infty$ the distributions are equivalent.

```
#### Normal vs t-distributions with a range of degrees-of-freedom
x <- seq(-8, 8, length = 1000)
par(mfrow=c(1,1))
plot(x, dnorm(x), type = "l", lwd = 2, col = "red"
     , main = "Normal (red) vs t-dist with df=1, 2, 6, 12, 30, 100")
points(x, dt(x,  1), type = "l")
points(x, dt(x,  2), type = "l")
points(x, dt(x,  6), type = "l")
points(x, dt(x, 12), type = "l")
points(x, dt(x, 30), type = "l")
points(x, dt(x,100), type = "l")
```



2.2 CI for μ

Statistical inference provides methods for drawing conclusions about a population from sample data. In this chapter, we want to make a claim about population mean μ given sample statistics $\bar{Y}$ and s.

A CI for μ is a range of plausible values for the unknown population mean μ, based on the observed data, of the form "Best Guess $\pm$ Reasonable Error of the Guess". To compute a CI for μ:

1. Define the **population parameter**, "Let μ = mean [characteristic] for population of interest".
2. Specify the **confidence coefficient**, which is a number between 0 and 100%, in the form $100(1-\alpha)\%$. Solve for α. (For example, 95% has $\alpha = 0.05$.)
3. Compute the t-critical value: $t_{\text{crit}} = t_{0.5\alpha}$ such that the area under the t-curve ($df = n-1$) to the right of t_{crit} is 0.5α. See appendix or internet for a t-table.
4. Report the CI in the form $\bar{Y} \pm t_{\text{crit}} SE_{\bar{Y}}$ or as an interval (L, U). The desired CI has lower and upper endpoints given by $L = \bar{Y} - t_{\text{crit}} SE_{\bar{Y}}$ and $U = \bar{Y} + t_{\text{crit}} SE_{\bar{Y}}$, respectively, where $SE_{\bar{Y}} = s/\sqrt{n}$ is the standard error of the sample mean.
5. Assess method assumptions (see below).

In practice, the confidence coefficient is large, say 95% or 99%, which correspond to $\alpha = 0.05$ and 0.01, respectively. The value of α expressed as a percent is known as the **error rate** of the CI.

The CI is determined once the confidence coefficient is specified and the data are collected. Prior to collecting the data, the interval is unknown and is viewed as random because it will depend on the actual sample selected. Different samples give different CIs. The **"confidence"** in, say, the 95% CI (which has a 5% error rate) can be interpreted as follows. *If you repeatedly sample the population and construct 95% CIs for μ, then 95% of the intervals will contain μ, whereas 5% will not.* The interval you construct from your data will either cover μ, or it will not.

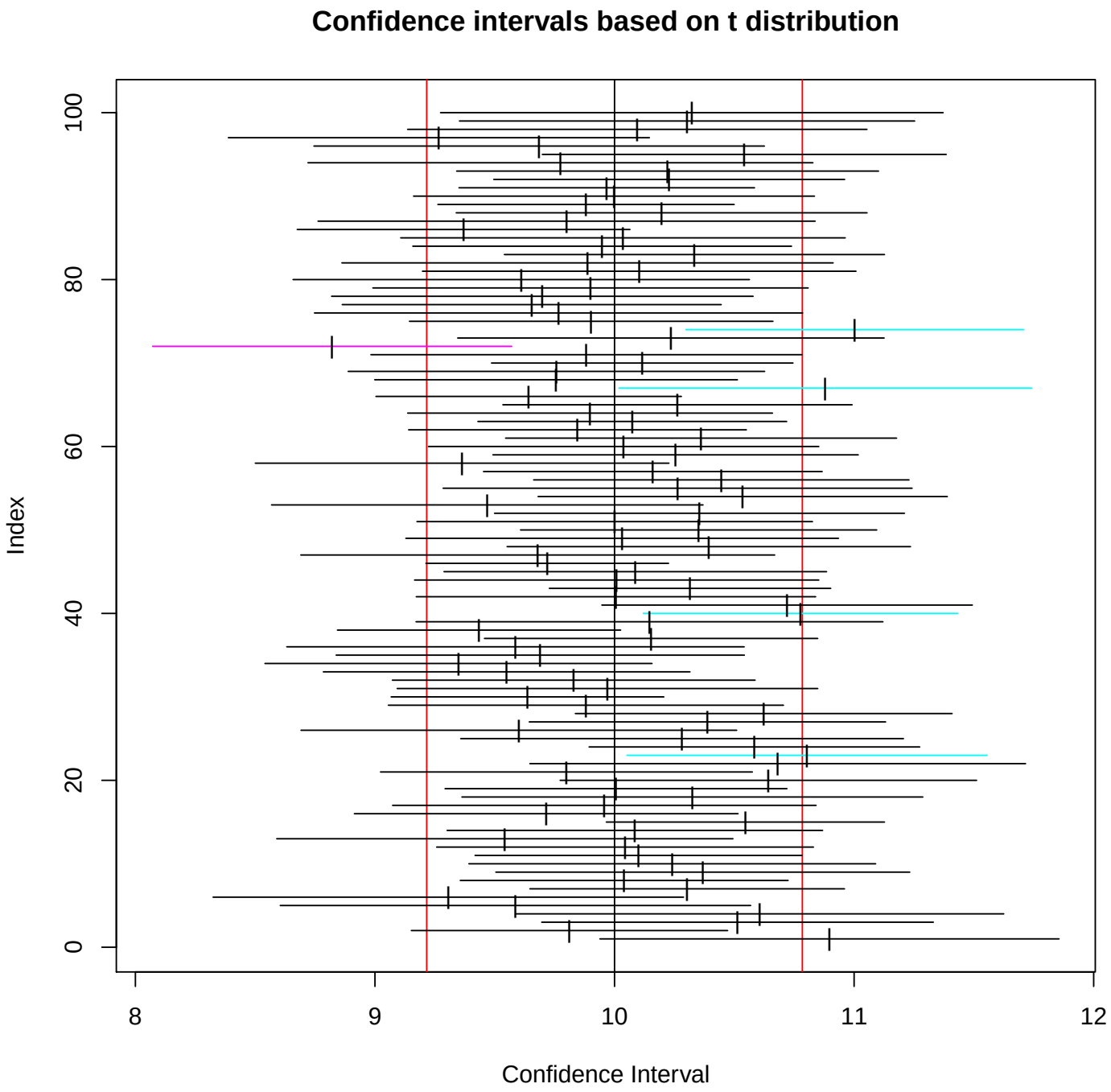
The length of the CI

$$U - L = 2t_{\text{crit}} SE_{\bar{Y}}$$

depends on the accuracy of our estimate $\bar{Y}$ of μ, as measured by the standard error of $\bar{Y}$, $SE_{\bar{Y}} = s/\sqrt{n}$. Less precise estimates of μ lead to wider intervals for a given level of confidence.

An example with 100 CIs Consider drawing a sample of 25 observations from a normally distributed population with mean 10 and sd 2. Calculate the 95% t-CI. Now do that 100 times. The plot belows reflects the variability of that process. We expect 95 of the 100 CIs to contain the true population mean of 10, that is, on average 5 times out of 100 we draw the incorrect inference that the population mean is in an interval when it does not contain the true value of 10.

```
#### Illustration of Confidence Intervals (consistent with their interpretation)
library(TeachingDemos)
ci.examp(mean.sim = 10, sd = 2, n = 25
       , reps = 100, conf.level = 0.95, method = "t")
```



■ CLICKER Qs — CI for μ, 2 ■

2.2.1 Assumptions for procedures

I described the classical CI. The procedure is based on the assumptions that the data are a **random sample** from the population of interest, and that the **population frequency curve is normal**. The population frequency curve can be viewed as a "smoothed histogram" created from the population data.

The normality assumption can never be completely verified without having the entire population data. You can assess the reasonableness of this assumption using a stem-and-leaf display or a boxplot of the sample data. The stem-and-leaf display from the data should resemble a normal curve.

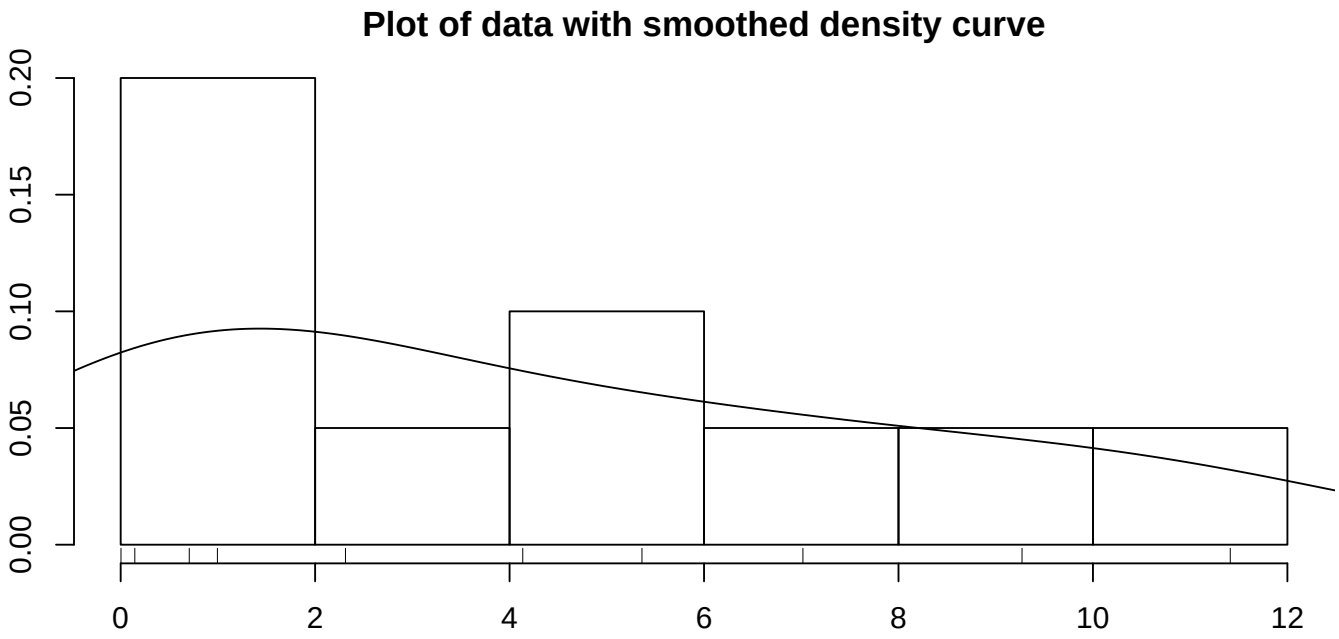
In fact, the assumptions are slightly looser than this, the population frequency curve can be anything provided the sample size is large enough that it's reasonable to assume that the **sampling distribution of the mean is normal**.

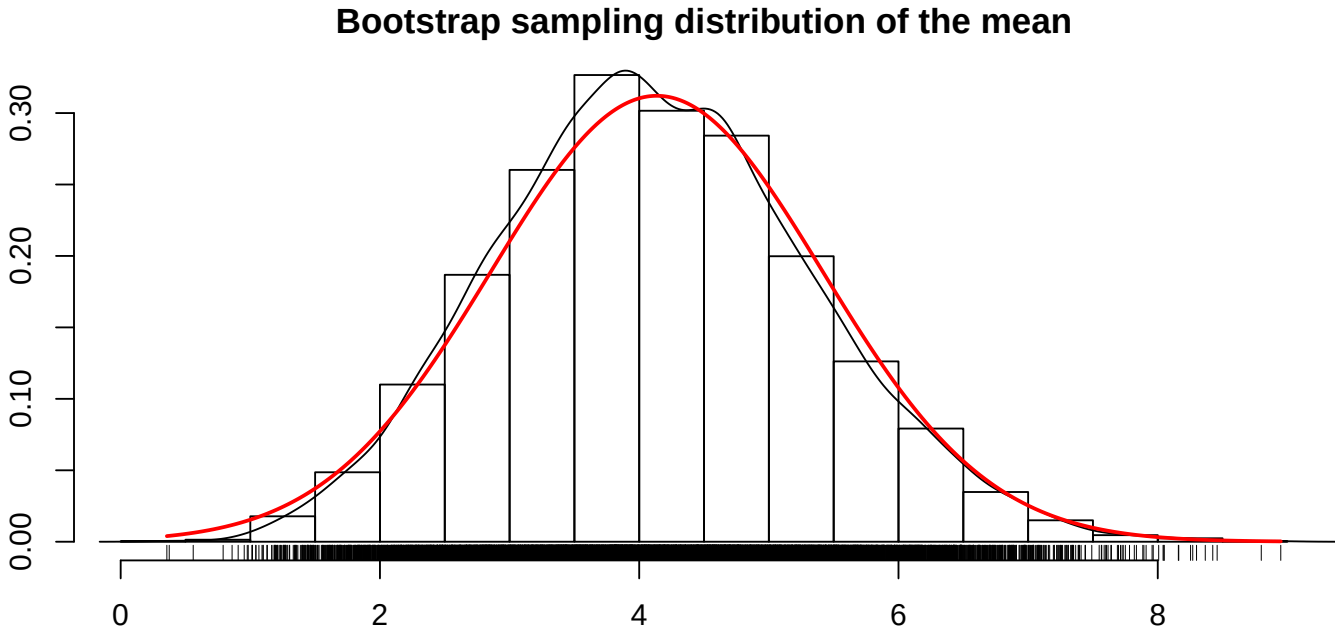
Assessing assumptions using the bootstrap

We will cover the bootstrap at the end of this course, but a brief introduction here can help us check model assumptions. Recall in Section 2.1.1 the sampling distribution examples. Assume the sample is representative of the population. Let's use our sample as a proxy for the population and repeatedly draw samples (with replacement) of size n and calculate the mean, then plot the bootstrap sampling distribution of means. If this bootstrap sampling distribution strongly deviates from normal, then that's evidence from the data that inference using the t-distribution is not appropriate. Otherwise, if roughly normal, then the t-distribution may be sensible.

```
#### Visual comparison of whether sampling distribution is close to Normal via Bootstrap
# a function to compare the bootstrap sampling distribution with
#   a normal distribution with mean and SEM estimated from the data
bs.one.samp.dist <- function(dat, N = 1e4) {
  n <- length(dat);
  # resample from data
  sam <- matrix(sample(dat, size = N * n, replace = TRUE), ncol=N);
  # draw a histogram of the means
  sam.mean <- colMeans(sam);
  # save par() settings
```

```
  old.par <- par(no.readonly = TRUE)
  # make smaller margins
  par(mfrow=c(2,1), mar=c(3,2,2,1), oma=c(1,1,1,1))
  # Histogram overlaid with kernel density curve
  hist(dat, freq = FALSE, breaks = 6
       , main = "Plot of data with smoothed density curve")
  points(density(dat), type = "l")
  rug(dat)

  hist(sam.mean, freq = FALSE, breaks = 25
       , main = "Bootstrap sampling distribution of the mean"
       , xlab = paste("Data: n =", n
                      , ", mean =", signif(mean(dat), digits = 5)
                      , ", se =", signif(sd(dat)/sqrt(n)), digits = 5))
  # overlay a density curve for the sample means
  points(density(sam.mean), type = "l")
  # overlay a normal distribution, bold and red
  x <- seq(min(sam.mean), max(sam.mean), length = 1000)
  points(x, dnorm(x, mean = mean(dat), sd = sd(dat)/sqrt(n))
         , type = "l", lwd = 2, col = "red")
  # place a rug of points under the plot
  rug(sam.mean)
  # restore par() settings
  par(old.par)
}

# example data, skewed --- try others datasets to develop your intuition
x <- rgamma(10, shape = .5, scale = 20)
bs.one.samp.dist(x)
```





Example: Age at First Heart Transplant Let us go through a hand-calculation of a CI, using R to generate summary data. I'll show you later how to generate the CI in R.

We are interested in the mean age at first heart transplant for a population of patients.

1. Define the population parameter
Let μ = mean age at the time of first heart transplant for population of patients.

2. Calculate summary statistics from sample
The ages (in years) at first transplant for a sample of 11 heart transplant patients are as follows:

54, 42, 51, 54, 49, 56, 33, 58, 54, 64, 49.

Summaries for the data are: $n = 11$, $\bar{Y} = 51.27$, and $s = 8.26$ so that $SE_{\bar{Y}} = 8.26/\sqrt{11} = 2.4904$. The degrees of freedom are $df = 11 - 1 = 10$.

3. Specify confidence level, find critical value, calculate limits
Let us calculate a 95% CI for μ. For a 95% CI $\alpha = 0.05$, so we need to find $t_{\text{crit}} = t_{0.025}$, which is 2.228. Now $t_{\text{crit}} SE_{\bar{Y}} = 2.228 \times 2.4904 = 5.55$. The lower limit on the CI is $L = 51.27 - 5.55 = 45.72$. The upper limit is $U = 51.27 + 5.55 = 56.82$.

4. Summarize in words For example, I am 95% confident that the population mean age at first transplant is 51.3 ± 5.55, that is, between 45.7 and 56.8 years (rounding off to 1 decimal place).

5. Check assumptions We will see this in several pages, sampling distribution is reasonably normal.

2.2.2 The effect of α on a two-sided CI

A two-sided $100(1-\alpha)\%$ CI for μ is given by $\bar{Y} \pm t_{\text{crit}} SE_{\bar{Y}}$. The CI is centered at $\bar{Y}$ and has length $2t_{\text{crit}} SE_{\bar{Y}}$. The confidence coefficient $100(1-\alpha)\%$ is **increased** by **decreasing** α, which increases t_{crit}. That is, increasing the confidence coefficient makes the CI wider. This is sensible: to increase your confidence that the interval captures μ you must pinpoint μ with less precision by making the CI wider. For example, a 95% CI is wider than a 90% CI.

■ CLICKERQs — CI, quickie ■

2.3 Hypothesis Testing for μ

A **hypothesis test** is used to make a **decision** about a population parameter.

Suppose you are interested in checking whether the population mean μ is equal to some prespecified value, say μ_0. This question can be formulated as a two-sided hypothesis test, where you are trying to decide which of two contradictory claims or hypotheses about μ is more reasonable given the observed data. The **null hypothesis**, or the hypothesis under test, is $H_0 : \mu = \mu_0$, whereas the **alternative hypothesis** is $H_A : \mu \neq \mu_0$.

I will explore the ideas behind hypothesis testing later. At this point, I focus on the mechanics behind the test. The steps in carrying out the test are:

1. Set up the **null and alternative hypotheses** in words and notation.
   In words: "The population mean for [what is being studied] is different from [value of μ_0]." (Note that the statement in words is in terms of the alternative hypothesis.)
   In notation: $H_0 : \mu = \mu_0$ versus $H_A : \mu \neq \mu_0$ (where μ_0 is specified by the context of the problem).
2. Choose the **size** or **significance level** of the test, denoted by α. In practice, α is set to a small value, say, 0.01 or 0.05, but theoretically can be any value between 0 and 1.
3. Compute the **test statistic**

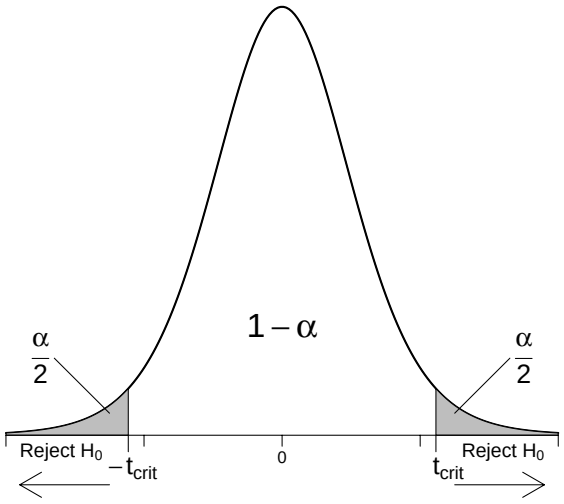
$$t_s = \frac{\bar{Y} - \mu_0}{SE_{\bar{Y}}},$$

   where $SE_{\bar{Y}} = s/\sqrt{n}$ is the standard error.
   Note: I sometimes call the test statistic t_{obs} to emphasize that the computed value depends on the observed data.
4. Compute the **critical value** $t_{\text{crit}} = t_{0.5\alpha}$ (or p-value from the test statistic) in the direction of the alternative hypothesis from the t-distribution table with degrees of freedom $df = n - 1$.
5. State the **conclusion** in terms of the problem.
   Reject H_0 in favor of H_A (i.e., decide that H_0 is false, based on the data)

if $|t_s| > t_{\text{crit}}$ or p-value $< \alpha$, that is, reject if $t_s < -t_{\text{crit}}$ or if $t_s > t_{\text{crit}}$. Otherwise, **Fail to reject** H_0.
(Note: We DO NOT *accept* H_0 — more on this later.)

6. **Check assumptions** of the test, when possible (could do earlier to save yourself some effort if they are not met).

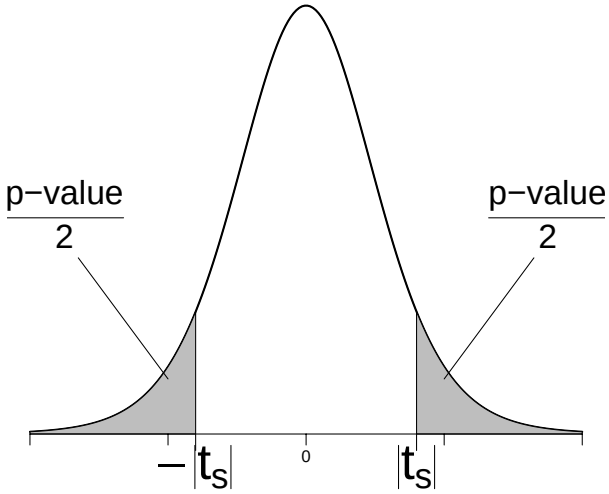
The process is represented graphically below. The area under the t-probability curve outside $\pm t_{\text{crit}}$ is the size of the test, α. One-half α is the area in each tail. You reject H_0 in favor of H_A only if the test statistic is outside $\pm t_{\text{crit}}$.



2.3.1 P-values

The **p-value**, or **observed significance level** for the test, provides a measure of plausibility for H_0. Smaller values of the p-value imply that H_0 is less plausible. To compute the p-value for a two-sided test, you

1. Compute the test statistic t_s as above.
2. Evaluate the area under the t-probability curve (with $df = n-1$) outside $\pm|t_s|$.



The p-value is the total shaded area, or twice the area in either tail. A useful **interpretation** of the p-value is that *it is the chance of obtaining data favoring H_A by this much or more* if H_0 *actually is true.* Another interpretation is that

> the **p-value** is *the probability of observing a sample mean at least as extreme as the one observed assuming μ_0 from H_0 is the true population mean.*

If the p-value is small then the sample we obtained is pretty unusual to have obtained if H_0 is true — but we actually got the sample, so probably it is not very unusual, so we would conclude H_0 is false (it would not be unusual if H_A is true).

Most, if not all, statistical packages summarize hypothesis tests with a p-value, rather than a decision (i.e., reject or not reject at a given α level). You can make a decision to reject or not reject H_0 for a size α test based on the p-value as follows — *reject H_0 if the p-value is less than α.* This decision is identical to that obtained following the formal rejection procedure given earlier. The reason for this is that the p-value can be interpreted as the smallest value you can set the size of the test and still reject H_0 given the observed data.

There are a lot of terms to keep straight here. α and t_{crit} are constants we choose (actually, one determines the other so we really only choose one, usually α) to set how rigorous evidence against H_0 needs to be. t_s and the p-value (again, one determines the other) are random variables because they are calculated from the random sample. They are the evidence against H_0.

2.3.2 Assumptions for procedures

I described the classical t-test, which assumes that the data are a random sample from the population and that the population frequency curve is normal. These are the same assumptions as for the CI.

Example: Age at First Transplant (Revisited) The ages (in years) at first transplant for a sample of 11 heart transplant patients are as follows: 54, 42, 51, 54, 49, 56, 33, 58, 54, 64, 49. Summaries for these data are: $n = 11$, $\bar{Y} = 51.27$, $s = 8.26$ and $SE_{\bar{Y}} = 2.4904$. Test the hypothesis that the mean age at first transplant is 50. Use $\alpha = 0.05$.

As in the earlier analysis, define

μ = mean age at time of first transplant for population of patients.

We are interested in testing $H_0 : \mu = 50$ against $H_A : \mu \neq 50$, so $\mu_0 = 50$.

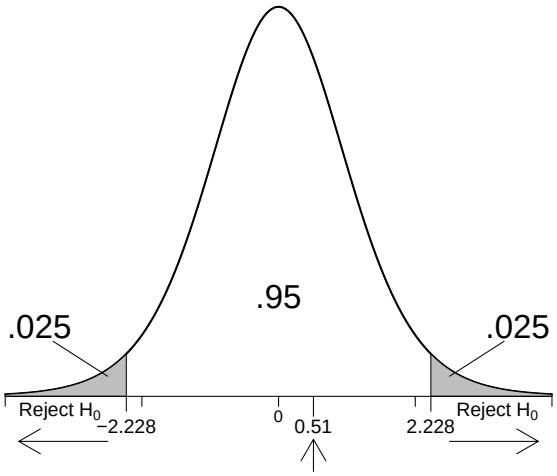
The degrees of freedom are $df = 11 - 1 = 10$. The critical value for a 5% test is $t_{\text{crit}} = t_{0.025} = 2.228$. (Note $\alpha/2 = 0.05/2 = 0.025$). The same critical value was used with the 95% CI.

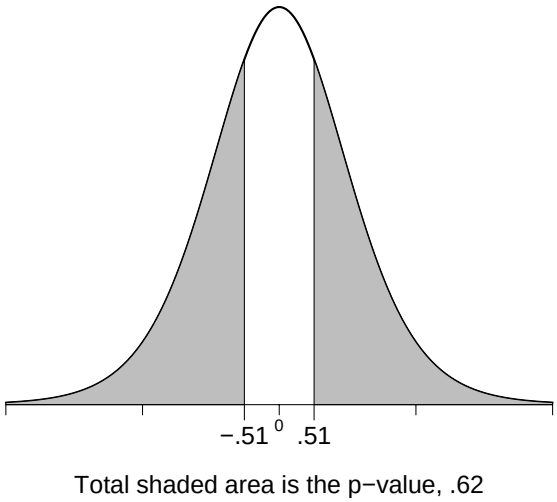
For the test,

$$t_s = \frac{\bar{Y} - \mu_0}{SE_{\bar{Y}}} = \frac{51.27 - 50}{2.4904} = 0.51.$$

Since $t_{\text{crit}} = 2.228$, we do not reject H_0 using a 5% test. Notice the placement of t_s relative to t_{crit} in the picture below. Equivalently, the p-value for the test is 0.62, thus we fail to reject H_0 because $0.62 > 0.05 = \alpha$. The results of the hypothesis test should not be surprising, since the CI tells you that 50 is a plausible value for the population mean age at transplant. Note: All you can

say is that the data *could have* come from a distribution with a mean of 50 — this is not convincing evidence that μ actually *is* 50.



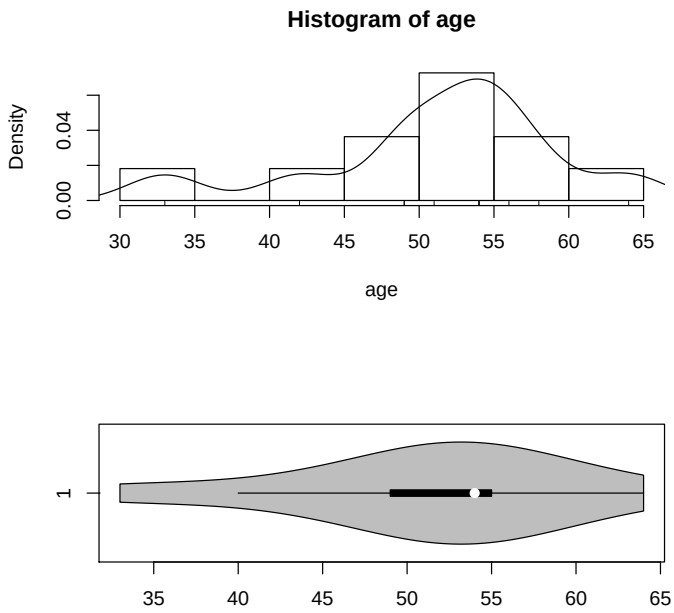


Example: Age at First Transplant R output for the heart transplant problem is given below. Let us look at the output and find all of the summaries we computed. Also, look at the graphical summaries to assess whether the t-test and CI are reasonable here.

```
#### Example: Age at First Transplant
# enter data as a vector
age <- c(54, 42, 51, 54, 49, 56, 33, 58, 54, 64, 49)
```

The age data is unimodal, skewed left, no extreme outliers.

```
par(mfrow=c(2,1))
# Histogram overlaid with kernel density curve
hist(age, freq = FALSE, breaks = 6)
points(density(age), type = "l")
rug(age)

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




```
# stem-and-leaf plot
stem(age, scale=2)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   3 | 3
##   3 |
##   4 | 2
##   4 | 99
##   5 | 1444
##   5 | 68
##   6 | 4
```

```
# t.crit
qt(1 - 0.05/2, df = length(age) - 1)
```

```
## [1] 2.228139
```

```
# look at help for t.test
?t.test
  # defaults include: alternative = "two.sided", conf.level = 0.95
t.summary <- t.test(age, mu = 50)
t.summary
```

```
##
##  One Sample t-test
##
## data:  age
## t = 0.51107, df = 10, p-value = 0.6204
## alternative hypothesis: true mean is not equal to 50
## 95 percent confidence interval:
##  45.72397 56.82149
```

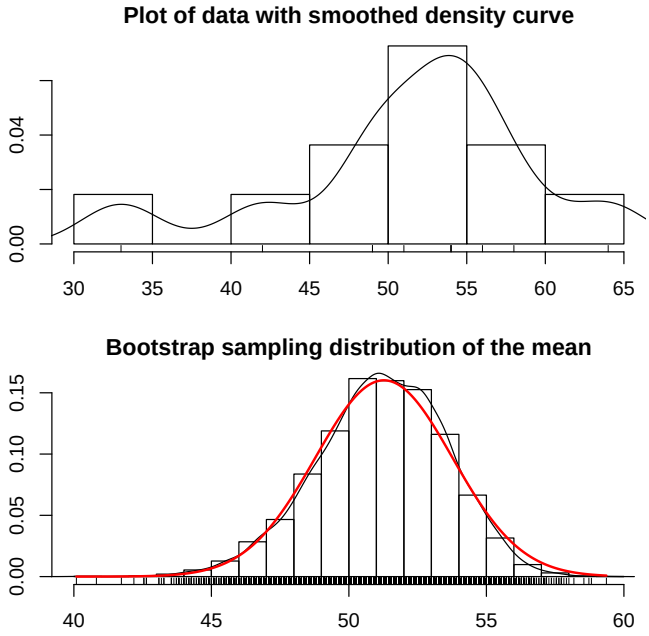
```
## sample estimates:
## mean of x
##  51.27273
```

```
summary(age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   33.00   49.00   54.00   51.27   55.00   64.00
```

The assumption of normality of the sampling distribution appears reasonablly close, using the bootstrap discussed earlier. Therefore, the results for the t-test above can be trusted.

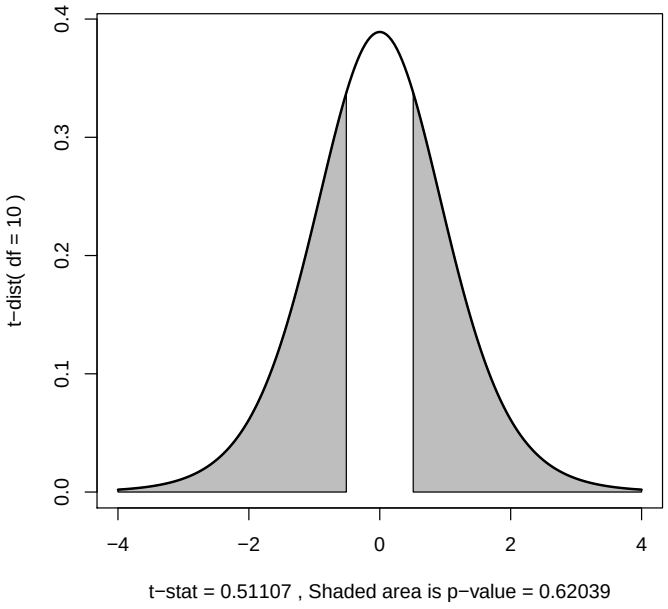
```
bs.one.samp.dist(age)
```



Aside: To print the shaded region for the p-value, you can use the result of `t.test()` with the function `t.dist.pval()` defined here.

```
# Function to plot t-distribution with shaded p-value
t.dist.pval <- function(t.summary) {
  par(mfrow=c(1,1))
  lim.extreme <- max(4, abs(t.summary$statistic) + 0.5)
  lim.lower <- -lim.extreme;
  lim.upper <-  lim.extreme;
  x.curve <- seq(lim.lower, lim.upper, length=200)
  y.curve <- dt(x.curve, df = t.summary$parameter)
  plot(x.curve, y.curve, type = "n"
       , ylab = paste("t-dist( df =", signif(t.summary$parameter, 3), ")")
       , xlab = paste("t-stat =", signif(t.summary$statistic, 5)
                      , ", Shaded area is p-value =", signif(t.summary$p.value, 5)))
  if ((t.summary$alternative == "less")
      | (t.summary$alternative == "two.sided")) {
    x.pval.l <- seq(lim.lower, -abs(t.summary$statistic), length=200)
```

```
    y.pval.l <- dt(x.pval.l, df = t.summary$parameter)
    polygon(c(lim.lower, x.pval.l, -abs(t.summary$statistic))
          , c(0, y.pval.l, 0), col="gray")
  }
  if ((t.summary$alternative == "greater")
      | (t.summary$alternative == "two.sided")) {
    x.pval.u <- seq(abs(t.summary$statistic), lim.upper, length=200)
    y.pval.u <- dt(x.pval.u, df = t.summary$parameter)
    polygon(c(abs(t.summary$statistic), x.pval.u, lim.upper)
          , c(0, y.pval.u, 0), col="gray")
  }
  points(x.curve, y.curve, type = "l", lwd = 2, col = "black")
}

# for the age example
t.dist.pval(t.summary)
```



Aside: Note that the `t.summary` object returned from `t.test()` includes a number of quantities that might be useful for additional calculations.

```
names(t.summary)
```

```
## [1] "statistic"   "parameter"   "p.value"     "conf.int"
## [5] "estimate"    "null.value"  "alternative" "method"
## [9] "data.name"
```

```
t.summary$statistic
```

```
##         t
## 0.5110715
```

```
t.summary$parameter
```

```
## df
## 10
```

```
t.summary$p.value
```

```
## [1] 0.6203942
```

```
t.summary$conf.int
```

```
## [1] 45.72397 56.82149
## attr(,"conf.level")
## [1] 0.95
```

```
t.summary$estimate
```

```
## mean of x
##  51.27273
```

```
t.summary$null.value
```

```
## mean
##   50
```

```
t.summary$alternative
```

```
## [1] "two.sided"
```

```
t.summary$method
```

```
## [1] "One Sample t-test"
```

```
t.summary$data.name
```

```
## [1] "age"
```

Example: Meteorites One theory of the formation of the solar system states that all solar system meteorites have the same evolutionary history and thus have the same cooling rates. By a delicate analysis based on measurements of phosphide crystal widths and phosphide-nickel content, the cooling rates, in degrees Celsius per million years, were determined for samples taken from meteorites named in the accompanying table after the places they were found. The Walker[2] County (Alabama, US), Uwet[3] (Cross River, Nigeria), and Tocopilla[4] (Antofagasta, Chile) meteorite cooling rate data are below.

Suppose that a hypothesis of solar evolution predicted a mean cooling rate of $\mu = 0.54$ degrees per million years for the Tocopilla meteorite. Do the observed cooling rates support this hypothesis? Test at the 5% level. The boxplot and stem-and-leaf display (given below) show good symmetry. The assumption of a normal distribution of observations basic to the t-test appears to be realistic.

Meteorite	Cooling rates
Walker County	0.69 0.23 0.10 0.03 0.56 0.10 0.01 0.02 0.04 0.22
Uwet	0.21 0.25 0.16 0.23 0.47 1.20 0.29 1.10 0.16
Tocopilla	5.60 2.70 6.20 2.90 1.50 4.00 4.30 3.00 3.60 2.40 6.70 3.80

Let

[2] http://www.lpi.usra.edu/meteor/metbull.php?code=24204

[3] http://www.lpi.usra.edu/meteor/metbull.php?code=24138

[4] http://www.lpi.usra.edu/meteor/metbull.php?code=17001

μ = mean cooling rate over all pieces of the Tocopilla meteorite.

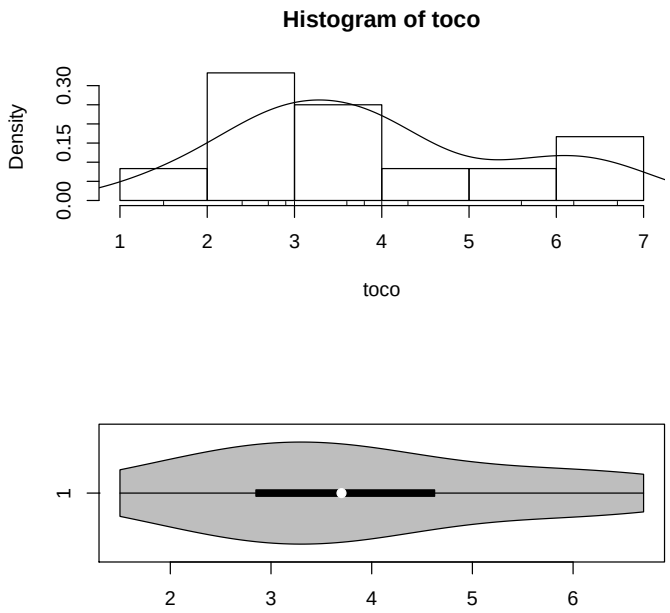
To answer the question of interest, we consider the test of $H_0 : \mu = 0.54$ against $H_A : \mu \neq 0.54$. Let us go carry out the test, compute the p-value, and calculate a 95% CI for μ. The sample summaries are $n = 12$, $\bar{Y} = 3.892$, $s = 1.583$. The standard error is $SE_{\bar{Y}} = s/\sqrt{n} = 0.457$.

R output for this problem is given below. For a 5% test (i.e., $\alpha = 0.05$), you would reject H_0 in favor of H_A because the p-value ≤ 0.05. The data strongly suggest that $\mu \neq 0.54$. The 95% CI says that you are 95% confident that the population mean cooling rate for the Tocopilla meteorite is between 2.89 and 4.90 degrees per million years. Note that the CI gives us a means to assess how different μ is from the hypothesized value of 0.54.

```
#### Example: Meteorites
# enter data as a vector
toco <- c(5.6, 2.7, 6.2, 2.9, 1.5, 4.0, 4.3, 3.0, 3.6, 2.4, 6.7, 3.8)
```

The Tocopilla data is unimodal, skewed right, no extreme outliers.

```
par(mfrow=c(2,1))
# Histogram overlaid with kernel density curve
hist(toco, freq = FALSE, breaks = 6)
points(density(toco), type = "l")
rug(toco)

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




```
# stem-and-leaf plot
stem(toco, scale=2)
```

```
##
##   The decimal point is at the |
##
##   1 | 5
##   2 | 479
##   3 | 068
##   4 | 03
##   5 | 6
##   6 | 27
```

```
# t.crit
qt(1 - 0.05/2, df = length(toco) - 1)
```

```
## [1] 2.200985
```

```
t.summary <- t.test(toco, mu = 0.54)
t.summary
```

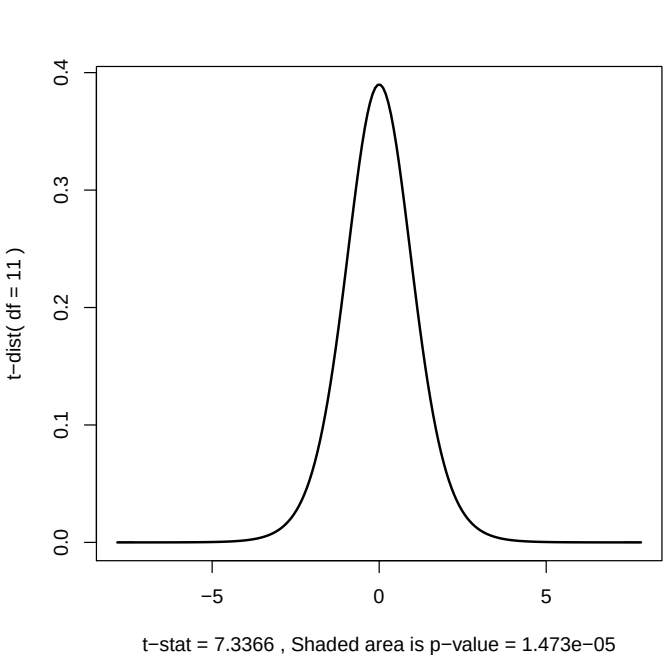
```
##
##  One Sample t-test
##
## data:  toco
## t = 7.3366, df = 11, p-value = 1.473e-05
## alternative hypothesis: true mean is not equal to 0.54
## 95 percent confidence interval:
##  2.886161 4.897172
## sample estimates:
## mean of x
##  3.891667
```

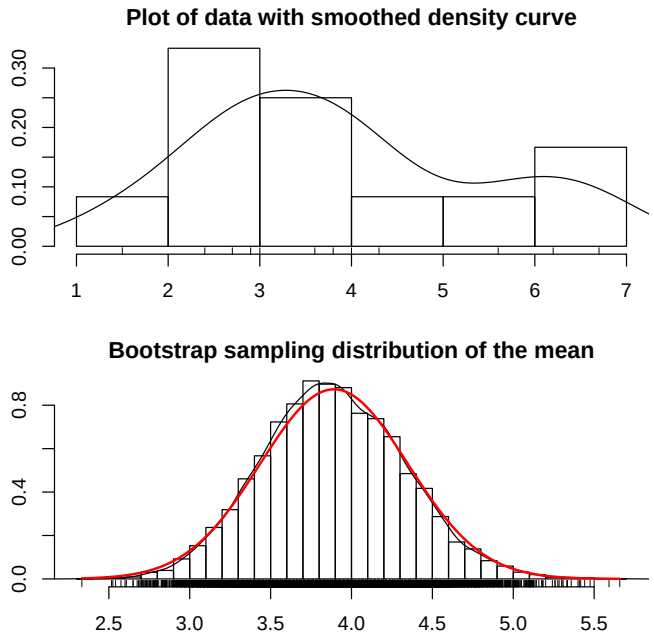
```
summary(toco)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.500   2.850   3.700   3.892   4.625   6.700
```

The assumption of normality of the sampling distribution appears reasonable. Therefore, the results for the t-test above can be trusted.

```
t.dist.pval(t.summary)
bs.one.samp.dist(toco)
```





2.3.3 The mechanics of setting up hypothesis tests

When setting up a test you should imagine you are the researcher conducting the experiment. In many studies, the researcher wishes to establish that there has been a change from the **status quo**, or that they have developed a method that produces a **change** (possibly in a specified direction) in the typical response. The researcher sets H_0 to be the **status quo** and H_A to be the **research hypothesis** — the claim the researcher wishes to make. In some studies you define the hypotheses so that H_A is the **take action** hypothesis — rejecting H_0 in favor of H_A leads one to take a radical action.

Some perspective on testing is gained by understanding the mechanics behind the tests. A hypothesis test is a decision process in the face of uncertainty. You are given data and asked which of two contradictory claims about a pop-

ulation parameter, say μ, is more reasonable. Two decisions are possible, but whether you make the correct decision depends on the true state of nature which is unknown to you.

	State of nature	
Decision	H_0 true	H_A true
Fail to reject [accept] H_0	correct decision	*Type-II error*
Reject H_0 in favor of H_A	*Type-I error*	correct decision

For a given problem, only one of these errors is possible. For example, if H_0 is true you can make a Type-I error but not a Type-II error. Any reasonable decision rule based on the data that tells us when to reject H_0 and when to not reject H_0 will have a certain probability of making a Type-I error if H_0 is true, and a corresponding probability of making a Type-II error if H_0 is false and H_A is true. For a given decision rule, define

$$\alpha = \text{Prob(Reject } H_0 \text{ given } H_0 \text{ is true)} = \text{Prob(Type-I error)}$$

and

$$\beta = \text{Prob(Fail to reject } H_0 \text{ when } H_A \text{ true)} = \text{Prob(Type-II error).}$$

The mathematics behind hypothesis tests allows you to prespecify or control α. For a given α, the tests we use (typically) have the smallest possible value of β. Given the researcher can control α, you set up the hypotheses so that committing a Type-I error is more serious than committing a Type-II error. The magnitude of α, also called the **size** or **level** of the test, should depend on the seriousness of a Type-I error in the given problem. The more serious the consequences of a Type-I error, the smaller α should be. In practice α is often set to 0.10, 0.05, or 0.01, with $\alpha = 0.05$ being the scientific standard. By setting α to be a small value, you reject H_0 in favor of H_A only if the data **convincingly indicate** that H_0 is false.

Let us piece together these ideas for the meteorite problem. Evolutionary history predicts $\mu = 0.54$. A scientist examining the validity of the theory is trying to decide whether $\mu = 0.54$ or $\mu \neq 0.54$. Good scientific practice dictates that rejecting another's claim when it is true is more serious than not being able to reject it when it is false. This is consistent with defining $H_0 : \mu = 0.54$ (the

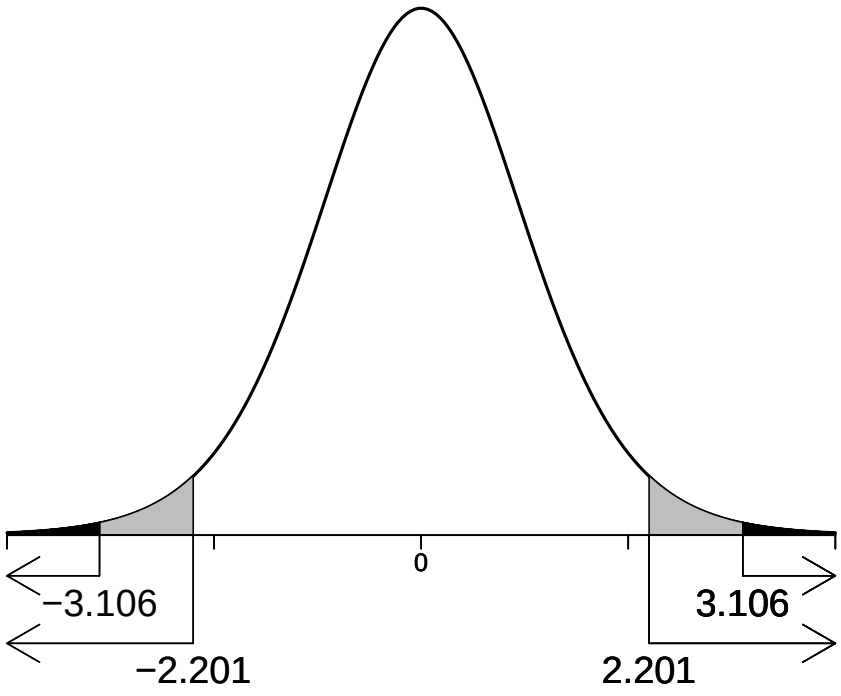
status quo) and $H_A : \mu \neq 0.54$. To convince yourself, note that the implications of a Type-I error would be to claim the evolutionary theory is false when it is true, whereas a Type-II error would correspond to not being able to refute the evolutionary theory when it is false. With this setup, the scientist will refute the theory only if the data overwhelmingly suggest that it is false.

2.3.4 The effect of α on the rejection region of a two-sided test

For a size α test, you reject $H_0 : \mu = \mu_0$ if

$$t_s = \frac{\bar{Y} - \mu_0}{SE_{\bar{Y}}}$$

satisfies $|t_s| > t_{\text{crit}}$.



Rejection Regions for .05 and .01 level tests

The critical value is computed so that the area under the t-probability curve (with $df = n - 1$) outside $\pm t_{\text{crit}}$ is α, with 0.5α in each tail. Reducing α makes t_{crit} larger. That is, reducing the size of the test makes rejecting H_0 harder because the rejection region is smaller. A pictorial representation is given above for the Tocopilla data, where $\mu_0 = 0.54$, $n = 12$, and $df = 11$. Note that $t_{\text{crit}} = 2.201$ and 3.106 for $\alpha = 0.05$ and 0.01, respectively.

The mathematics behind the test presumes that H_0 is true. Given the data, you use

$$t_s = \frac{\bar{Y} - \mu_0}{SE_{\bar{Y}}}$$

to measure how far $\bar{Y}$ is from μ_0, relative to the spread in the data given by $SE_{\bar{Y}}$. For t_s to be in the rejection region, $\bar{Y}$ must be significantly above or below μ_0, relative to the spread in the data. To see this, note that rejection

rule can be expressed as: **Reject** H_0 if

$$\bar{Y} < \mu_0 - t_{\text{crit}} SE_{\bar{Y}} \quad \text{or} \quad \bar{Y} > \mu_0 + t_{\text{crit}} SE_{\bar{Y}}.$$

The rejection rule is sensible because $\bar{Y}$ is our best guess for μ. You would reject $H_0 : \mu = \mu_0$ only if $\bar{Y}$ is so far from μ_0 that you would question the reasonableness of assuming $\mu = \mu_0$. How far $\bar{Y}$ must be from μ_0 before you reject H_0 depends on α (i.e., how willing you are to reject H_0 if it is true), and on the value of $SE_{\bar{Y}}$. For a given sample, reducing α forces $\bar{Y}$ to be further from μ_0 before you reject H_0. For a given value of α and s, increasing n allows smaller differences between $\bar{Y}$ and μ_0 to be **statistically significant** (i.e., lead to rejecting H_0). In problems where small differences between $\bar{Y}$ and μ_0 lead to rejecting H_0, you need to consider whether the observed differences are important.

In essence, the t-distribution provides an objective way to calibrate whether the observed $\bar{Y}$ is typical of what sample means look like when sampling from a normal population where H_0 is true. If all other assumptions are satisfied, and $\bar{Y}$ is inordinately far from μ_0, then our only recourse is to conclude that H_0 must be incorrect.

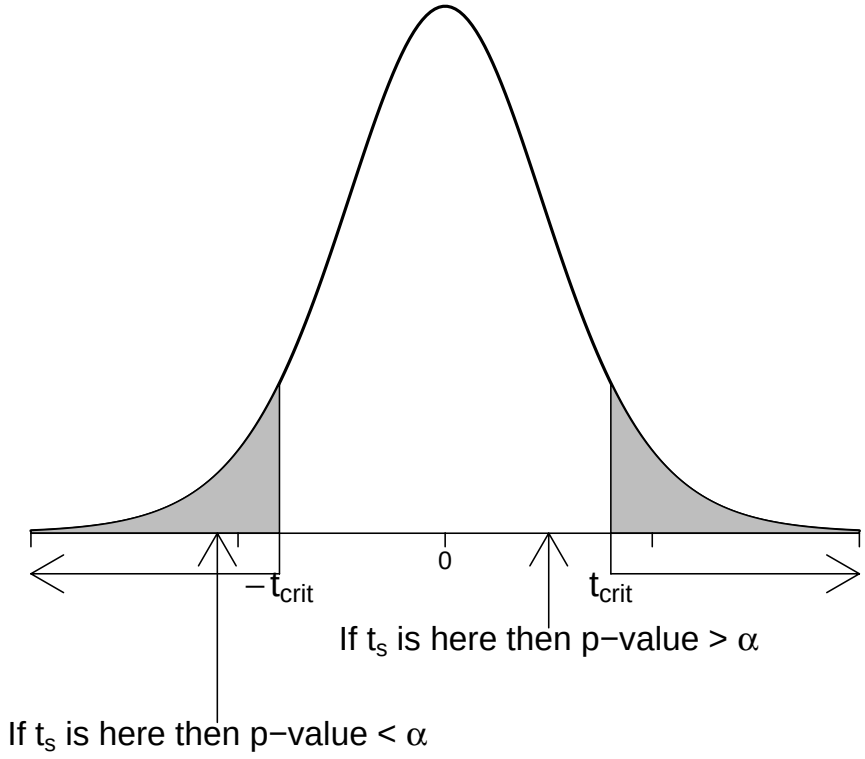
2.4 Two-sided tests, CI and p-values

An important relationship among two-sided tests of $H_0 : \mu = \mu_0$, CI, and p-values is that

$$\text{size } \alpha \text{ test rejects } H_0 \Leftrightarrow 100(1-\alpha)\% \text{ CI does not contain } \mu_0 \Leftrightarrow \text{p-value} \leq \alpha, \text{ and}$$

$$\text{size } \alpha \text{ test fails to reject } H_0 \Leftrightarrow 100(1-\alpha)\% \text{ CI contains } \mu_0 \Leftrightarrow \text{p-value} > \alpha.$$

For example, an $\alpha = 0.05$ test rejects $H_0 \Leftrightarrow$ 95% CI does not contain $\mu_0 \Leftrightarrow$ p-value ≤ 0.05. The picture below illustrates the connection between p-values and rejection regions.



Either a CI or a test can be used to decide the plausibility of the claim that $\mu = \mu_0$. Typically, you use the test to answer the question **is there a difference?** If so, you use the CI to assess **how much of a difference exists**. I believe that scientists place too much emphasis on hypothesis testing.

2.5 Statistical versus practical significance

Suppose in the Tocopilla meteorite example, you rejected $H_0 : \mu = 0.54$ at the 5% level and found a 95% two-sided CI for μ to be 0.55 to 0.58. Although you have sufficient evidence to conclude that the population mean cooling rate μ differs from that suggested by evolutionary theory, the range of plausible values for μ is small and contains only values close to 0.54. Although you have

shown statistical significance here, you need to ask yourself whether the actual difference between μ and 0.54 is large enough to be important. The answer to such questions is always problem specific.

2.6 Design issues and power

An experiment may not be sensitive enough to pick up true differences. For example, in the Tocopilla meteorite example, suppose the true mean cooling rate is $\mu = 1.00$. To have a 50% chance of correctly rejecting $H_0 : \mu = 0.54$, you would need about $n = 48$ observations. If the true mean is $\mu = 0.75$, you would need about 221 observations to have a 50% chance of correctly rejecting H_0. In general, the smaller the difference between the true and hypothesized mean (relative to the spread in the population), the more data that is needed to reject H_0. If you have prior information on the expected difference between the true and hypothesized mean, you can design an experiment appropriately by choosing the sample size required to likely reject H_0.

The **power** of a test is the probability of correctly rejecting H_0 when it is false. Equivalently,

power $= 1 -$ Prob(failing to reject H_0 when it is false) $= 1 -$ Prob(Type-II error).

For a given sample size, the tests I have discussed have maximum power (or smallest probability of a Type-II error) among all tests with fixed size α. However, the actual power may be small, so sample size calculations, as briefly highlighted above, are important prior to collecting data. See your local statistician.

```
#### Power calculations (that you can do on your own)
# install.packages("pwr")
library(pwr)
?power.t.test
power.t.test(n = NULL, delta = 1.00 - 0.54, sd = sd(toco),
             , sig.level = 0.05, power = 0.50
             , type = "one.sample", alternative = "two.sided")
power.t.test(n = NULL, delta = 0.75 - 0.54, sd = sd(toco),
             , sig.level = 0.05, power = 0.50
```

```
            , type = "one.sample", alternative = "two.sided")
```

For more examples, try:

```
# install.packages("TeachingDemos")
library(TeachingDemos)
# Demonstrate concepts of Power.
?power.examp
```

2.7 One-sided tests on μ

There are many studies where a one-sided test is appropriate. The two common scenarios are the **lower one-sided test** $H_0 : \mu = \mu_0$ (or $\mu \geq \mu_0$) versus $H_A : \mu < \mu_0$ and the **upper one-sided test** $H_0 : \mu = \mu_0$ (or $\mu \leq \mu_0$) versus $H_A : \mu > \mu_0$. Regardless of the alternative hypothesis, the tests are based on the t-statistic:

$$t_s = \frac{\bar{Y} - \mu_0}{SE_{\bar{Y}}}.$$

For the **upper one-sided test**

1. Compute the critical value t_{crit} such that the area under the t-curve to the **right** of t_{crit} is the desired size α, that is $t_{\text{crit}} = t_\alpha$.
2. Reject H_0 if and only if $t_s \geq t_{\text{crit}}$.
3. The p-value for the test is the area under the t-curve to the **right** of the test statistic t_s.

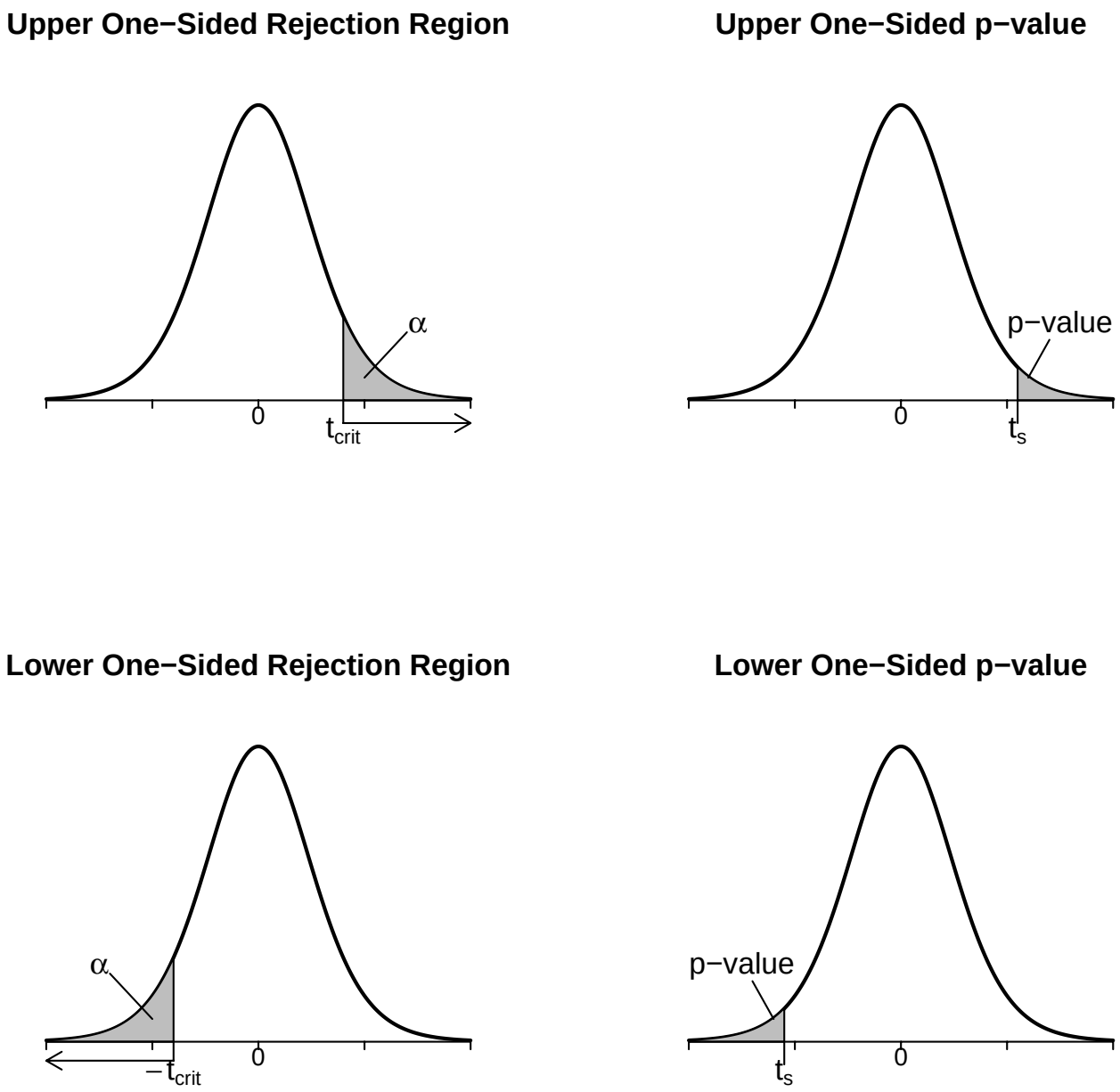
The **upper one-sided test** uses the **upper tail** of the t-distribution for a rejection region. The p-value calculation reflects the form of the rejection region. You will reject H_0 only for large positive values of t_s which require $\bar{Y}$ to be significantly greater than μ_0. Does this make sense?

For the **lower one-sided test**

1. Compute the critical value t_{crit} such that the area under the t-curve to the **right** of t_{crit} is the desired size α, that is $t_{\text{crit}} = t_\alpha$.
2. Reject H_0 if and only if $t_s \leq -t_{\text{crit}}$.
3. The p-value for the test is the area under the t-curve to the **left** of the test statistic t_s.

The **lower one-sided test** uses the **lower tail** of the t-distribution for a rejection region. The calculation of the rejection region in terms of $-t_{\text{crit}}$ is awkward but is necessary for hand calculations because many statistical tables only give upper tail percentiles. Note that here you will reject H_0 only for large negative values of t_s which require $\bar{Y}$ to be significantly less than μ_0.

As with two-sided tests, the p-value can be used to decide between rejecting or not rejecting H_0 for a test with a given size α. A picture of the rejection region and the p-value evaluation for one-sided tests is given below.



■ CLICKER Qs — One-sided tests on μ ■

Example: Weights of canned tomatoes A consumer group suspects that the average weight of canned tomatoes being produced by a large cannery

is less than the advertised weight of 20 ounces. To check their conjecture, the group purchases 14 cans of the canner's tomatoes from various grocery stores. The weights of the contents of the cans to the nearest half ounce were as follows: 20.5, 18.5, 20.0, 19.5, 19.5, 21.0, 17.5, 22.5, 20.0, 19.5, 18.5, 20.0, 18.0, 20.5. Do the data confirm the group's suspicions? Test at the 5% level.

Let $\mu =$ the population mean weight for advertised 20 ounce cans of tomatoes produced by the cannery. The company claims that $\mu = 20$, but the consumer group believes that $\mu < 20$. Hence the consumer group wishes to test $H_0 : \mu = 20$ (or $\mu \geq 20$) against $H_A : \mu < 20$. The consumer group will reject H_0 only if the data overwhelmingly suggest that H_0 is false.

You should assess the normality assumption prior to performing the t-test. The stem-and-leaf display and the boxplot suggest that the distribution might be slightly skewed to the left. However, the skewness is not severe and no outliers are present, so the normality assumption is not unreasonable.

R output for the problem is given below. Let us do a hand calculation using the summarized data. The sample size, mean, and standard deviation are 14, 19.679, and 1.295, respectively. The standard error is $SE_{\bar{Y}} = s/\sqrt{n} = 0.346$. We see that the sample mean is less than 20. But is it sufficiently less than 20 for us to be willing to publicly refute the canner's claim? Let us carry out the test, first using the rejection region approach, and then by evaluating a p-value.

The test statistic is

$$t_s = \frac{\bar{Y} - \mu_0}{SE_{\bar{Y}}} = \frac{19.679 - 20}{0.346} = -0.93.$$

The critical value for a 5% one-sided test is $t_{0.05} = 1.771$, so we reject H_0 if $t_s < -1.771$ (you can get that value from r or from the table). The test statistic is not in the rejection region. Using the t-table, the p-value is between 0.15 and 0.20. I will draw a picture to illustrate the critical region and p-value calculation. The exact p-value from R is 0.185, which exceeds 0.05.

Both approaches lead to the conclusion that we do not have sufficient evidence to reject H_0. That is, we do not have sufficient evidence to question the accuracy of the canner's claim. If you did reject H_0, is there something about

how the data were recorded that might make you uncomfortable about your conclusions?

```
#### Example: Weights of canned tomatoes
tomato <- c(20.5, 18.5, 20.0, 19.5, 19.5, 21.0, 17.5
          , 22.5, 20.0, 19.5, 18.5, 20.0, 18.0, 20.5)

par(mfrow=c(2,1))
# Histogram overlaid with kernel density curve
hist(tomato, freq = FALSE, breaks = 6)
points(density(tomato), type = "l")
rug(tomato)

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

# t.crit
qt(1 - 0.05/2, df = length(tomato) - 1)
```

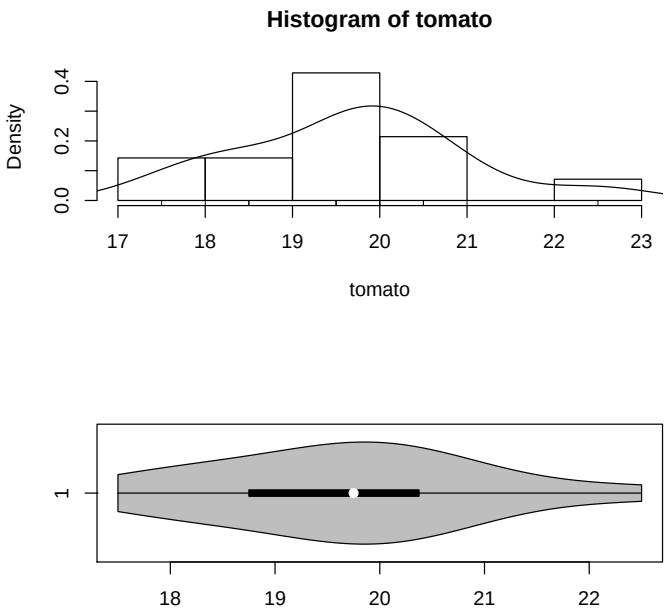
```
## [1] 2.160369
```

```
t.summary <- t.test(tomato, mu = 20, alternative = "less")
t.summary
```

```
##
##  One Sample t-test
##
## data:  tomato
## t = -0.92866, df = 13, p-value = 0.185
## alternative hypothesis: true mean is less than 20
## 95 percent confidence interval:
##      -Inf 20.29153
## sample estimates:
## mean of x
##  19.67857
```

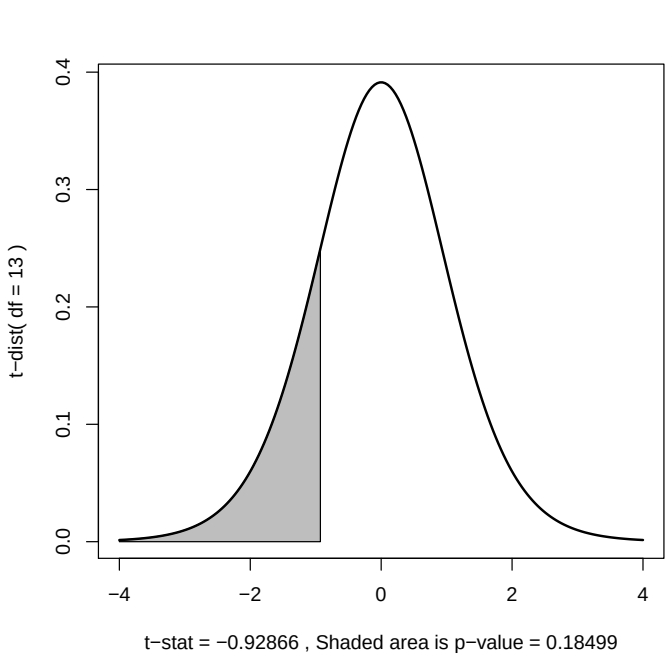
```
summary(tomato)
```

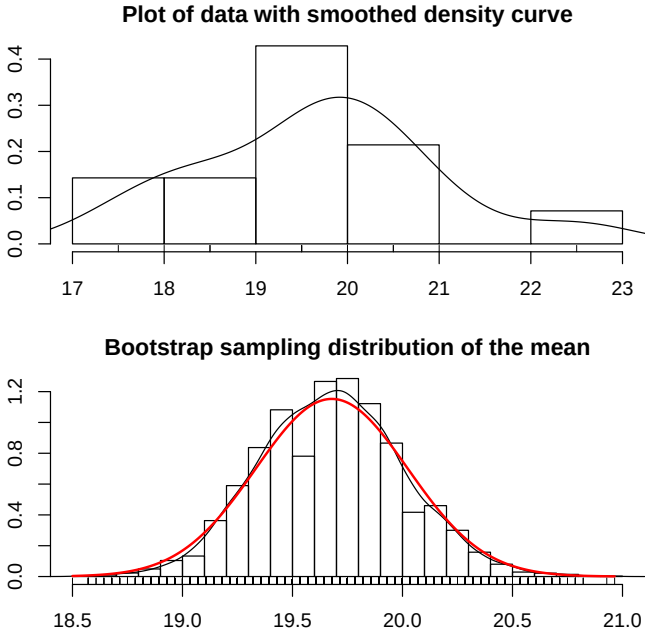
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   17.50   18.75   19.75   19.68   20.38   22.50
```



The assumption of normality of the sampling distribution appears reasonable. Therefore, the results for the t-test above can be trusted.

```
t.dist.pval(t.summary)
bs.one.samp.dist(tomato)
```





2.7.1 One-sided CIs

How should you couple a one-sided test with a CI procedure? For a **lower one-sided test**, you are interested only in an **upper bound** on μ. Similarly, with an **upper one-sided test** you are interested in a **lower bound** on μ. Computing these type of bounds maintains the consistency between tests and CI procedures. The general formulas for lower and upper $100(1-\alpha)\%$ confidence bounds on μ are given by

$$\bar{Y} - t_{\text{crit}} SE_{\bar{Y}} \qquad \text{and} \qquad \bar{Y} + t_{\text{crit}} SE_{\bar{Y}}$$

respectively, where $t_{\text{crit}} = t_\alpha$.

In the cannery problem, to get an upper 95% bound on μ, the critical value is the same as we used for the one-sided 5% test: $t_{0.05} = 1.771$. The upper bound on μ is

$$\bar{Y} + t_{0.05} SE_{\bar{Y}} = 19.679 + 1.771 \times 0.346 = 19.679 + 0.613 = 20.292.$$

Thus, you are 95% confident that the population mean weight of the canner's 20oz cans of tomatoes is less than or equal to 20.29. As expected, this interval covers 20.

If you are doing a one-sided test in R, it will generate the correct one-sided bound. That is, a lower one-sided test will generate an upper bound, whereas an upper one-sided test generates a lower bound. If you only wish to compute a one-sided bound without doing a test, you need to specify the direction of the alternative which gives the type of bound you need. An upper bound was generated by R as part of the test we performed earlier. The result agrees with the hand calculation.

Quite a few packages, do not directly compute one-sided bounds so you have to fudge a bit. In the cannery problem, to get an upper 95% bound on μ, you take the upper limit from a 90% two-sided confidence limit on μ. The rational for this is that with the 90% two-sided CI, μ will fall above the upper limit 5% of the time and fall below the lower limit 5% of the time. Thus, you are 95% confident that μ falls below the upper limit of this interval, which gives us

our one-sided bound. Here, you are 95% confident that the population mean weight of the canner's 20 oz cans of tomatoes is less than or equal to 20.29, which agrees with our hand calculation.

```
One-Sample T: Cans
Variable   N     Mean   StDev  SE Mean         90% CI
Cans      14  19.6786  1.2951   0.3461  (19.0656, 20.2915)
```

The same logic applies if you want to generalize the one-sided confidence bounds to arbitrary confidence levels and to lower one-sided bounds — always double the error rate of the desired one-sided bound to get the error rate of the required two-sided interval! For example, if you want a lower 99% bound on μ (with a 1% error rate), use the lower limit on the 98% two-sided CI for μ (which has a 2% error rate).

■ CLICKER Qs — P-value ■

Chapter 3

Two-Sample Inferences

Contents


```
## Warning in file(filename, "r", encoding = encoding):  cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding):  cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

select graphical displays that meaningfully compare independent populations.

assess the assumptions of the two-sample t-test visually.

decide whether the means between two populations are different.

recommend action based on a hypothesis test.

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

1. organize knowledge.

5. define parameters of interest and hypotheses in words and notation.

6. summarize data visually, numerically, and descriptively.

8. use statistical software.

12. make evidence-based decisions.

3.1 Comparing Two Sets of Measurements

Suppose you have collected data on one variable from two (independent) samples and you are interested in "comparing" the samples. What tools are good to use?

Example: Head Breadths In this analysis, we will compare a physical feature of modern day Englishmen with the corresponding feature of some of their ancient countrymen. The Celts were a vigorous race of people who once populated parts of England. It is not entirely clear whether they simply died out or merged with other people who were the ancestors of those who live in England today. A goal of this study might be to shed some light on possible genetic links between the two groups.

The study is based on the comparison of maximum head breadths (in millimeters) made on unearthed Celtic skulls and on a number of skulls of modern-day Englishmen. The data are given below. We have a sample of 18 Englishmen and an independent sample of 16 Celtic skulls.

```
#### Example: Head Breadths
# unstacked data as two vectors
english <- c(141, 148, 132, 138, 154, 142, 150, 146, 155, 158,
             150, 140, 147, 148, 144, 150, 149, 145)
celts   <- c(133, 138, 130, 138, 134, 127, 128, 138, 136, 131,
             126, 120, 124, 132, 132, 125)
english
```

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

```
celts
```

```
##  [1] 133 138 130 138 134 127 128 138 136 131 126 120 124 132 132 125
```

What features of these data would we likely be interested in comparing? The centers of the distributions, the spreads within each distribution, the distributional shapes, etc.

These data can be analyzed in R as either **unstacked** separate vectors or as **stacked** data where one column contains both samples, with a second column of labels or **subscripts** to distinguish the samples. It is easy to create stacked data from unstacked data and vice-versa. Many comparisons require the plots for the two groups to have the same scale, which is easiest to control when the data are stacked.

```
# stacked data as a vector of values and a vector of labels
HeadBreadth <- c(english, celts)
Group <- c(rep("English", length(english)), rep("Celts", length(celts)))
hb <- data.frame(HeadBreadth, Group)
hb
```

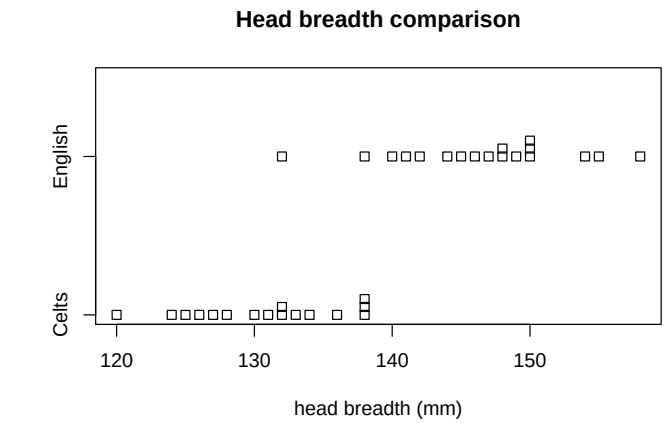
```
##    HeadBreadth   Group
## 1          141 English
## 2          148 English
## 3          132 English
## 4          138 English
## 5          154 English
## 6          142 English
## 7          150 English
## 8          146 English
## 9          155 English
## 10         158 English
## 11         150 English
## 12         140 English
## 13         147 English
## 14         148 English
## 15         144 English
```

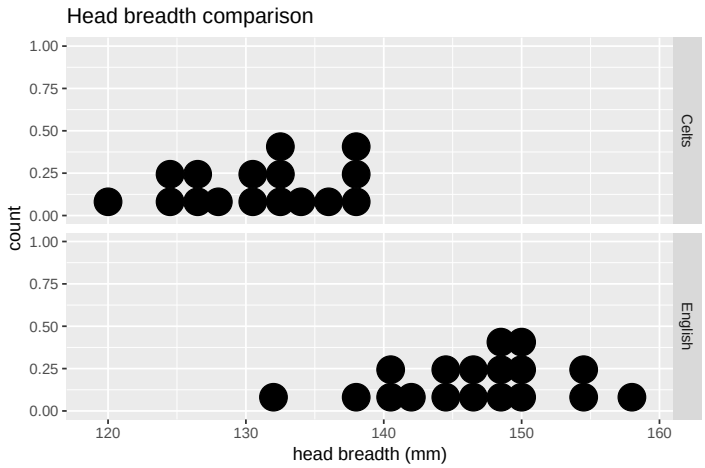
```
## 16         150 English
## 17         149 English
## 18         145 English
## 19         133   Celts
## 20         138   Celts
## 21         130   Celts
## 22         138   Celts
## 23         134   Celts
## 24         127   Celts
## 25         128   Celts
## 26         138   Celts
## 27         136   Celts
## 28         131   Celts
## 29         126   Celts
## 30         120   Celts
## 31         124   Celts
## 32         132   Celts
## 33         132   Celts
## 34         125   Celts
```

3.1.1 Plotting head breadth data:

1. A dotplot with the same scale for both samples is obtained easily from the stacked data.

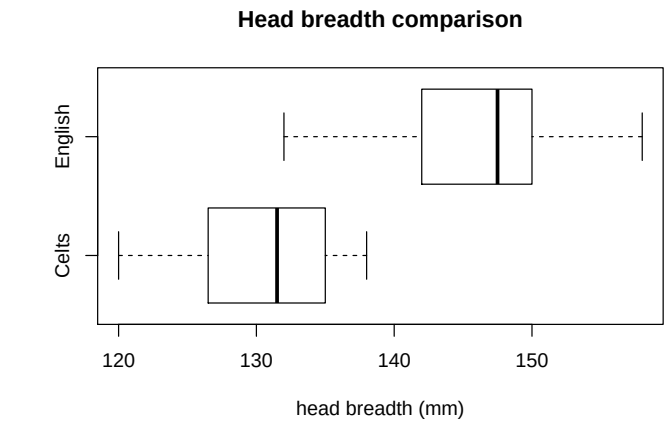
```
#### Plotting head breadth data
# stripchart (dotplot) using R base graphics
stripchart(HeadBreadth ~ Group, method = "stack", data = hb,
  main = "Head breadth comparison", xlab = "head breadth (mm)")


# stripchart (dotplot) using ggplot
library(ggplot2)
p <- ggplot(hb, aes(x = HeadBreadth))
p <- p + geom_dotplot(binwidth = 2)
p <- p + facet_grid(Group ~ .)     # rows are Group categories
p <- p + labs(title = "Head breadth comparison") + xlab("head breadth (mm)")
print(p)
```

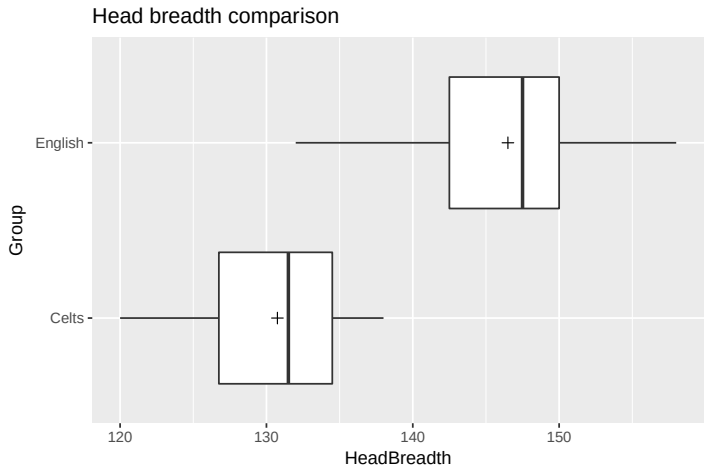




2. Boxplots for comparison are most helpful when plotted in the same axes.

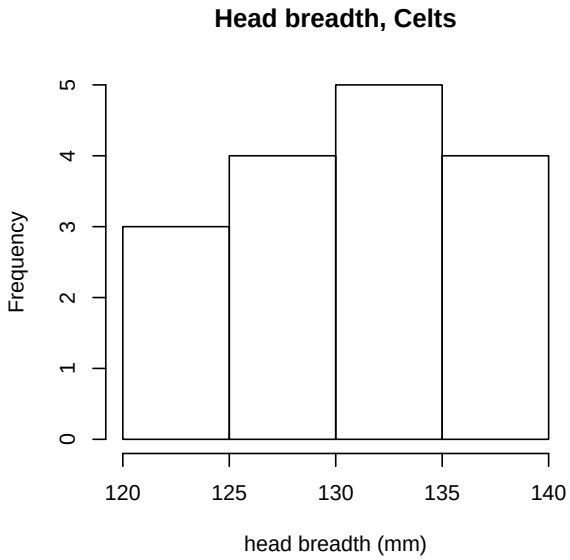
```r
# boxplot using R base graphics
boxplot(HeadBreadth ~ Group, method = "stack", data = hb,
  horizontal = TRUE,
  main = "Head breadth comparison", xlab = "head breadth (mm)")


p <- ggplot(hb, aes(x = Group, y = HeadBreadth))
p <- p + geom_boxplot()
# add a "+" at the mean
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 3, size = 2)
p <- p + coord_flip()
p <- p + labs(title = "Head breadth comparison")
print(p)
```

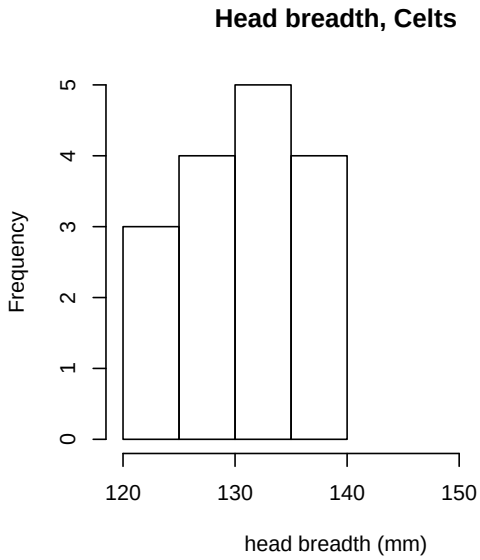


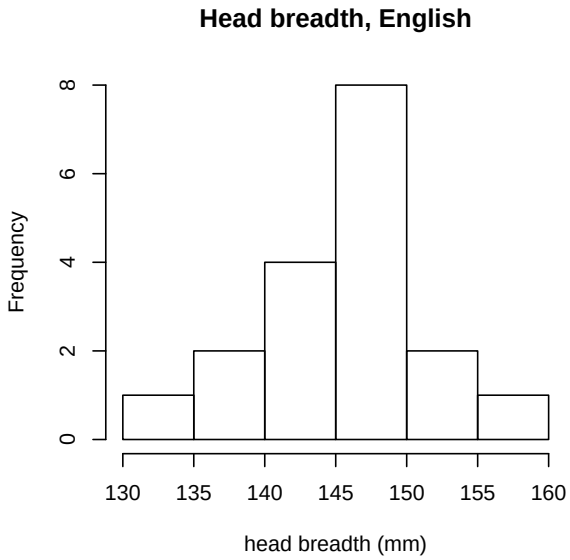


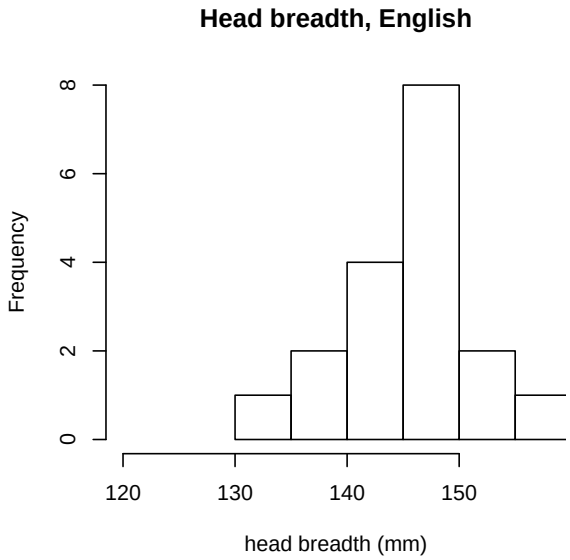
3. Histograms are hard to compare unless you make the scale and actual bins the same for both. Why is the pair on the right clearly preferable?

```r
# histogram using R base graphics
par(mfcol=c(2,2))
hist(hb$HeadBreadth[(hb$Group == "Celts")],
  main = "Head breadth, Celts", xlab = "head breadth (mm)")
hist(hb$HeadBreadth[(hb$Group == "English")],
  main = "Head breadth, English", xlab = "head breadth (mm)")

# common x-axis limits based on the range of the entire data set
hist(hb$HeadBreadth[(hb$Group == "Celts")], xlim = range(hb$HeadBreadth),
  main = "Head breadth, Celts", xlab = "head breadth (mm)")
hist(hb$HeadBreadth[(hb$Group == "English")], xlim = range(hb$HeadBreadth),
  main = "Head breadth, English", xlab = "head breadth (mm)")
```

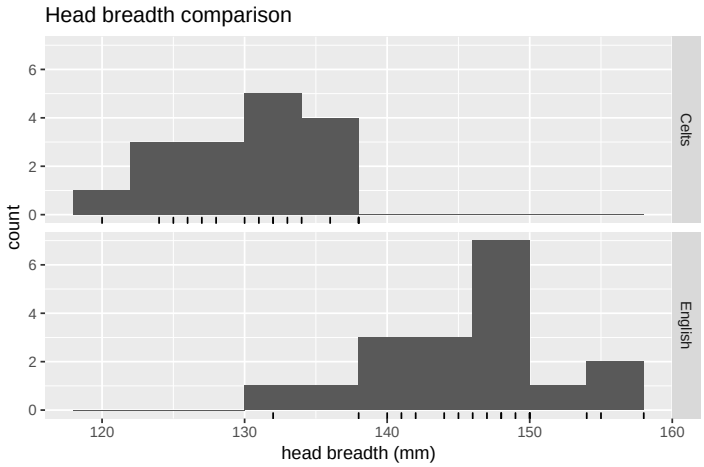




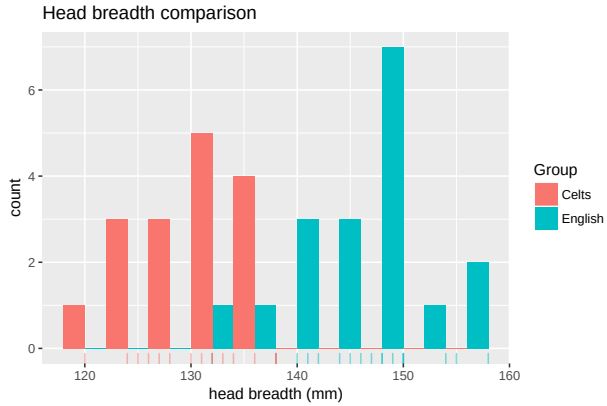





```
# histogram using ggplot
p <- ggplot(hb, aes(x = HeadBreadth))
p <- p + geom_histogram(binwidth = 4)
p <- p + geom_rug()
p <- p + facet_grid(Group ~ .)
p <- p + labs(title = "Head breadth comparison") + xlab("head breadth (mm)")
print(p)
```




```
p <- ggplot(hb, aes(x = HeadBreadth, fill=Group))
p <- p + geom_histogram(binwidth = 4, alpha = 0.5, position="identity")
p <- p + geom_rug(aes(colour=Group), alpha = 1/2)
p <- p + labs(title = "Head breadth comparison") + xlab("head breadth (mm)")
print(p)

p <- ggplot(hb, aes(x = HeadBreadth, fill=Group))
p <- p + geom_histogram(binwidth = 4, alpha = 1, position="dodge")
p <- p + geom_rug(aes(colour=Group), alpha = 1/2)
p <- p + labs(title = "Head breadth comparison") + xlab("head breadth (mm)")
print(p)
```



4. Stem-and-leaf displays for comparisons in R can be pretty useless. The stems are not forced to match (just like with histograms). It is pretty hard to make quick comparisons with the following:

```
stem(english, 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

stem(celts, scale = 2)

##
##   The decimal point is at the |
##
##   120 | 0
##   122 |
##   124 | 00
##   126 | 00
##   128 | 0
##   130 | 00
##   132 | 000
##   134 | 0
##   136 | 0
##   138 | 000
```

Using the `by()` function, you can get summaries by group.

```
#### summaries by group
# summary for separate vectors
summary(english)

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

summary(celts)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   120.0   126.8   131.5   130.8   134.5   138.0

# comparing spreads, an assumption of equal variances seems reasonable
sd(english)

## [1] 6.382421

sd(celts)

## [1] 5.434458

IQR(english)

## [1] 7.5

IQR(celts)

## [1] 7.75
```

```
# numerical summary of each column in data.frame hb by Group
by(hb, Group, summary)

## Group: Celts
##   HeadBreadth        Group
##  Min.   :120.0   Celts  :16
##  1st Qu.:126.8   English: 0
##  Median :131.5
##  Mean   :130.8
```

```
##  3rd Qu.:134.5
##  Max.   :138.0
## ------------------------------------------------
## Group: English
##   HeadBreadth        Group
##  Min.   :132.0   Celts  : 0
##  1st Qu.:142.5   English:18
##  Median :147.5
##  Mean   :146.5
##  3rd Qu.:150.0
##  Max.   :158.0
```

3.1.2 Salient Features to Notice

The dotplots, boxplots, and histograms indicate that the English and Celt samples are slightly skewed to the left. There are no outliers in either sample. It is not unreasonable to operationally assume that the population frequency curves (i.e., the histograms for the populations from which the samples were selected) for the English and Celtic head breadths are normal. Therefore, the sampling distribution of the means will be reasonably normal.

The sample means and medians are close to each other in each sample, which is not surprising given the near symmetry and the lack of outliers.

The data suggest that the typical modern English head breadth is greater than that for Celts. The data sets have comparable spreads, as measured by either the standard deviation or the IQR.

3.2 Two-Sample Methods: Paired Versus Independent Samples

Suppose you have two populations of interest, say populations 1 and 2, and you are interested in comparing their (unknown) population means, μ_1 and μ_2. Inferences on the unknown population means are based on samples from each population. In practice, most problems fall into one of two categories.

Independent samples where the sample taken from population 1 has no

effect on which observations are selected from population 2, and vice versa.

Paired or dependent samples where experimental units are paired based on factors related or unrelated to the variable measured.

The distinction between paired and independent samples may be made clear through a series of examples.

Example The English and Celt head breadth samples are independent.

Example Suppose you are interested in whether the $CaCO_3$ (calcium carbonate) level in the Atrisco well field, which is the water source for Albuquerque, is changing over time. To answer this question, the $CaCO_3$ level was recorded at each of 15 wells at two time points. These data are paired. The two samples are the observations at Times 1 and 2.

Example To compare state incomes, a random sample of New Mexico households was selected, and an independent sample of Arizona households was obtained. It is reasonable to assume independent samples.

Example Suppose you are interested in whether the husband or wife is typically the heavier smoker among couples where both adults smoke. Data are collected on households. You measure the average number of cigarettes smoked by each husband and wife within the sample of households. These data are paired, i.e., you have selected husband wife pairs as the basis for the samples. It is reasonable to believe that the responses within a pair are related, or correlated.

Although the focus here will be on comparing population means, you should recognize that in paired samples you may also be interested, as in the problems above, in how observations compare within a pair. That is, a **paired comparison** might be interested in the **difference** between the two paired samples. These goals need not agree, depending on the questions of interest. Note that with paired data, the sample sizes are equal, and equal to the number of pairs.

■ CLICKERQs — Independent or paired 1, STT.08.02.030 ■

■ CLICKERQs — Independent or paired 2, STT.08.02.040 ■

3.3 Two Independent Samples: CI and Test Using Pooled Variance

These methods assume that the populations have normal frequency curves, with equal population standard deviations, i.e., $\sigma_1 = \sigma_2$. Let $(n_1,\ \bar{Y}_1,\ s_1)$ and $(n_2,\ \bar{Y}_2,\ s_2)$ be the sample sizes, means and standard deviations from the two samples. The standard CI for $\mu_1 - \mu_2$ is given by

$$\text{CI} \ = \ (\bar{Y}_1 - \bar{Y}_2) \pm t_{\text{crit}} SE_{\bar{Y}_1 - \bar{Y}_2}$$

The t-statistic for testing $H_0 : \mu_1 - \mu_2 = 0 \quad (\mu_1 = \mu_2)$ against $H_A : \mu_1 - \mu_2 \neq 0$ $(\mu_1 \neq \mu_2)$ is given by

$$t_s = \frac{\bar{Y}_1 - \bar{Y}_2}{SE_{\bar{Y}_1 - \bar{Y}_2}}.$$

The standard error of $\bar{Y}_1 - \bar{Y}_2$ used in both the CI and the test is given by

$$SE_{\bar{Y}_1 - \bar{Y}_2} = s_{\text{pooled}} \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}.$$

Here the **pooled variance estimator**,

$$s^2_{\text{pooled}} = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2},$$

is our best estimate of the common population variance. The pooled estimator of variance is a weighted average of the two sample variances, with more weight given to the larger sample. If $n_1 = n_2$ then s^2_{pooled} is the average of s_1^2 and s_2^2.

The critical value t_{crit} for CI and tests is obtained in usual way from a t-table with $df = n_1 + n_2 - 2$. For the test, follow the one-sample procedure, with the new t_s and t_{crit}.

The pooled CI and tests are sensitive to the normality and equal standard deviation assumptions. The observed data can be used to assess the reasonableness of these assumptions. You should look at boxplots and histograms to assess normality, and should check whether $s_1 \doteq s_2$ to assess the assumption $\sigma_1 = \sigma_2$. Formal tests of these assumptions will be discussed later.

3.4 Satterthwaite's Method, unequal variances

Satterthwaite's method assumes normality, but does not require equal population standard deviations. Satterthwaite's procedures are somewhat conservative, and adjust the SE and df to account for unequal population variances. Satterthwaite's method uses the same CI and test statistic formula, with a modified standard error:

$$SE_{\bar{Y}_1 - \bar{Y}_2} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}},$$

and degrees of freedom:

$$df = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{s_1^4}{n_1^2(n_1-1)} + \frac{s_2^4}{n_2^2(n_2-1)}}.$$

Note that $df = n_1 + n_2 - 2$ when $n_1 = n_2$ and $s_1 = s_2$. The Satterthwaite and pooled variance procedures usually give similar results when $s_1 \doteq s_2$.

The df formula for Satterthwaite's method is fairly complex, so when done by hand some use a conservative df formula that uses the minimum of $n_1 - 1$ and $n_2 - 1$ instead.

3.4.1 R Implementation

R does the pooled and Satterthwaite (Welch) analyses, either on stacked or unstacked data. The output will contain a p-value for a two-sided test of equal population means and a CI for the difference in population means. If you include `var.equal = TRUE` you will get the pooled method, otherwise the output is for Satterthwaite's method.

Example: Head Breadths The English and Celts are independent samples. We looked at boxplots and histograms, which suggested that the normality assumption for the t-test is reasonable. The R output shows the English and Celt sample standard deviations and IQRs are fairly close, so the pooled and Satterthwaite results should be comparable. The pooled analysis is preferable here, but either is appropriate.

We are interested in difference in mean head breadths between Celts and English.

1. Define the population parameters and hypotheses in words and notation

Let μ_1 and μ_2 be the mean head breadth for the Celts and English, respectively.

In words: "The difference in population means between Celts and English is different from zero mm."

In notation: $H_0 : \mu_1 = \mu_2$ versus $H_A : \mu_1 \neq \mu_2$.

Alternatively: $H_0 : \mu_1 - \mu_2 = 0$ versus $H_A : \mu_1 - \mu_2 \neq 0$.

2. Calculate summary statistics from sample

Mean, standard deviation, sample size:

```
#### Calculate summary statistics
m1 <- mean(celts)
s1 <- sd(celts)
n1 <- length(celts)
m2 <- mean(english)
s2 <- sd(english)
n2 <- length(english)
c(m1, s1, n1)

## [1] 130.750000   5.434458  16.000000
```

```
c(m2, s2, n2)
```

```
## [1] 146.500000   6.382421  18.000000
```

The pooled-standard devation, standard error, and degrees-of-freedom are:

```
sdpool <- sqrt(((n1 - 1) * s1^2 + (n2 - 1) * s2^2) / (n1 + n2 - 2))
sdpool
```

```
## [1] 5.956876
```

```
SEpool <- sdpool * sqrt(1 / n1 + 1 / n2)
SEpool
```

```
## [1] 2.046736
```

```
dfpool <- n1 + n2 - 2
dfpool
```

```
## [1] 32
```

```
t_pool <- (m1 - m2) / SEpool
t_pool
```

```
## [1] -7.69518
```

The Satterthwaite SE and degrees-of-freedom are:

```
SE_Sat <- sqrt(s1^2 / n1 + s2^2 / n2)
SE_Sat
```

```
## [1] 2.027043
```

```
df_Sat <- (SE_Sat^2)^2 / (s1^4 / (n1^2 * (n1 - 1)) + s2^4 / (n2^2 * (n2 - 1)))
df_Sat
```

```
## [1] 31.9511
```

```
t_Sat <- (m1 - m2) / SE_Sat
t_Sat
```

```
## [1] -7.769937
```

3. Specify confidence level, calculate t-stat, CI limits, p-value

Let us calculate a 95% CI for $\mu_1 - \mu_2$.
Assuming equal variances, using pooled-variance procedure:

```
## Equal variances
#   var.equal = FALSE is the default

# two-sample t-test specifying two separate vectors
t.summary.eqvar <- t.test(celts, english, var.equal = TRUE)
t.summary.eqvar
```

```
##
##  Two Sample t-test
```

```
##
## data:  celts and english
## t = -7.6952, df = 32, p-value = 9.003e-09
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -19.91906 -11.58094
## sample estimates:
## mean of x mean of y
##    130.75    146.50
```

Not assuming equal variances, Satterthwaite (Welch):

```
# two-sample t-test with unequal variances (Welch = Satterthwaite)
#   specified using data.frame and a formula, HeadBreadth by Group
t.summary.uneqvar <- t.test(HeadBreadth ~ Group, data = hb, var.equal = FALSE)
t.summary.uneqvar
```

```
##
##  Welch Two Sample t-test
##
## data:  HeadBreadth by Group
## t = -7.7699, df = 31.951, p-value = 7.414e-09
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -19.8792 -11.6208
## sample estimates:
##   mean in group Celts mean in group English
##                130.75                146.50
```

The form of the output will tell you which sample corresponds to population 1 and which corresponds to population 2.

4. **Summarize in words** (Using the pooled-variance results.)

The pooled analysis strongly suggests that $H_0 : \mu_1 - \mu_2 = 0$ is false, given the large t-statistic of -7.7 and two-sided p-value of 9×10^{-9}. Because the p-value < 0.05 we reject the Null hypothesis in favor of the Alternative hypothesis concluding that the difference in population mean head breadths between the Celts and English are different.

We are 95% confident that the difference in population means, $\mu_1 - \mu_2$, is between -19.9 and -11.6 mm. That is, we are 95% confident that the population mean head breadth for Englishmen (μ_2) exceeds the population mean head breadth for Celts (μ_1) by between 11.6 and 19.9 mm.

The CI interpretation is made easier by recognizing that we concluded the population means are different, so the direction of difference must be con-

sistent with that seen in the observed data, where the sample mean head breadth for Englishmen exceeds that for the Celts. Thus, the limits on the CI for $\mu_1 - \mu_2$ tells us how much smaller the mean is for the Celts (that is, between -19.9 and -11.6 mm).

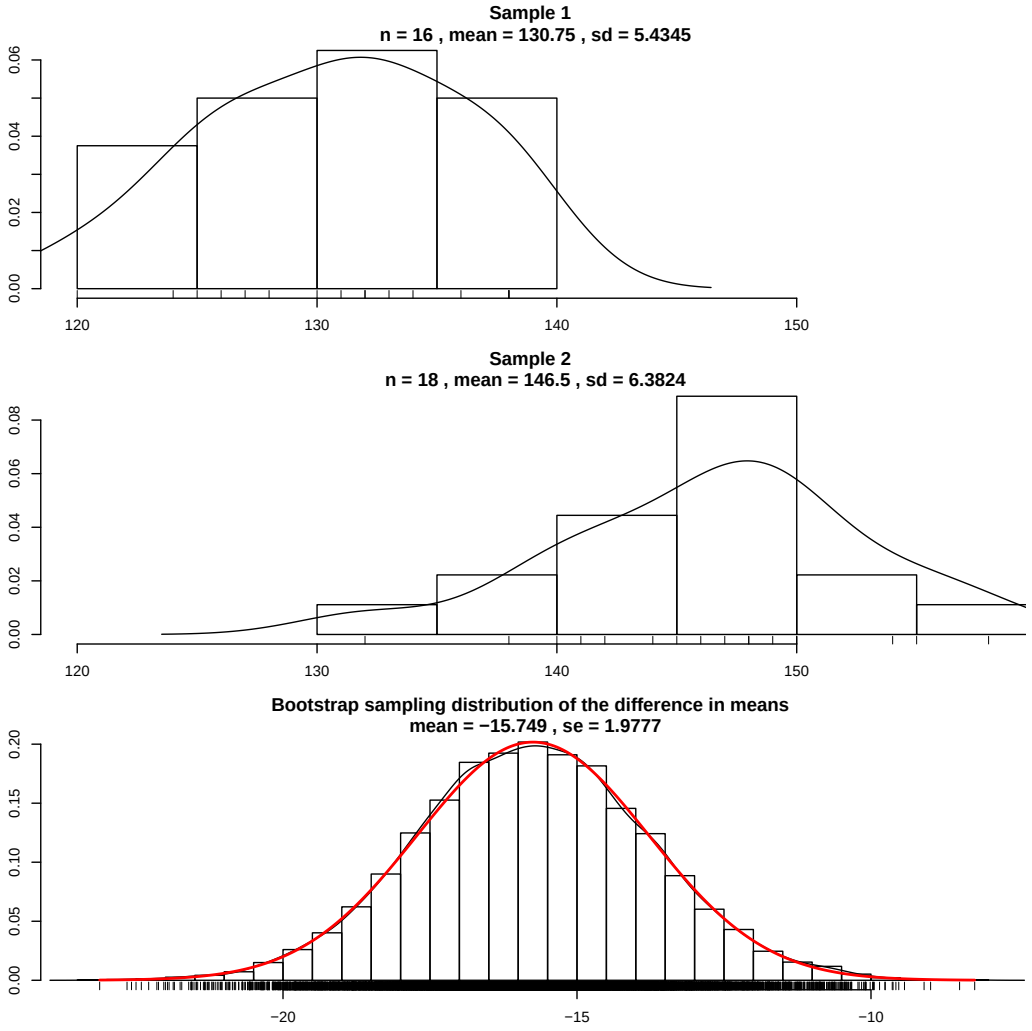
5. Check assumptions

The assumption of equal population variances will be left to a later chapter. We can test the assumption that the distribution of $\bar{Y}_1 - \bar{Y}_2$ is normal using the bootstrap in the following function.

```
#### Visual comparison of whether sampling distribution is close to Normal via Bootstrap
# a function to compare the bootstrap sampling distribution
#   of the difference of means from two samples with
#   a normal distribution with mean and SEM estimated from the data
bs.two.samp.diff.dist <- function(dat1, dat2, N = 1e4) {
  n1 <- length(dat1);
  n2 <- length(dat2);
  # resample from data
  sam1 <- matrix(sample(dat1, size = N * n1, replace = TRUE), ncol=N);
  sam2 <- matrix(sample(dat2, size = N * n2, replace = TRUE), ncol=N);
  # calculate the means and take difference between populations
  sam1.mean <- colMeans(sam1);
  sam2.mean <- colMeans(sam2);
  diff.mean <- sam1.mean - sam2.mean;
  # save par() settings
  old.par <- par(no.readonly = TRUE)
  # make smaller margins
  par(mfrow=c(3,1), mar=c(3,2,2,1), oma=c(1,1,1,1))
  # Histogram overlaid with kernel density curve
  hist(dat1, freq = FALSE, breaks = 6
      , main = paste("Sample 1", "\n"
                     , "n =", n1
                     , ", mean =", signif(mean(dat1), digits = 5)
                     , ", sd =", signif(sd(dat1), digits = 5))
      , xlim = range(c(dat1, dat2)))
  points(density(dat1), type = "l")
  rug(dat1)

  hist(dat2, freq = FALSE, breaks = 6
      , main = paste("Sample 2", "\n"
                     , "n =", n2
                     , ", mean =", signif(mean(dat2), digits = 5)
                     , ", sd =", signif(sd(dat2), digits = 5))
      , xlim = range(c(dat1, dat2)))
  points(density(dat2), type = "l")
  rug(dat2)

  hist(diff.mean, freq = FALSE, breaks = 25
      , main = paste("Bootstrap sampling distribution of the difference in means", "\n"
                   , "mean =", signif(mean(diff.mean), digits = 5)
                   , ", se =", signif(sd(diff.mean), digits = 5)))
  # overlay a density curve for the sample means
  points(density(diff.mean), type = "l")
  # overlay a normal distribution, bold and red
  x <- seq(min(diff.mean), max(diff.mean), length = 1000)
  points(x, dnorm(x, mean = mean(diff.mean), sd = sd(diff.mean))
       , type = "l", lwd = 2, col = "red")
  # place a rug of points under the plot
  rug(diff.mean)
  # restore par() settings
  par(old.par)
}
```

The distribution of difference in means in the third plot looks very close to normal.

```
bs.two.samp.diff.dist(celts, english)
```



■ CLICKER Qs — t-interval, STT.08.02.010 ■

Example: Androstenedione levels in diabetics The data consist of independent samples of diabetic men and women. For each individual, the scientist recorded their androstenedione level (ng/dL) (a hormone, and Mark McGwire's favorite dietary supplement). Let μ_1 = mean androstenedione level for the population of diabetic men, and μ_2 = mean androstenedione level for the population of diabetic women. We are interested in comparing the population means given the observed data.

The raw data and R output are given below. The boxplots suggest that the distributions are reasonably symmetric. However, the normality assumption for the women is unreasonable due to the presence of outliers. The equal population standard deviation assumption also appears unreasonable. The sample standard deviation for men is noticeably larger than the women's standard deviation, even with outliers in the women's sample.

```
#### Example: Androstenedione levels in diabetics
# Data and numerical summaries
men   <- c(217, 123,  80, 140, 115, 135,  59, 126,  70,  63,
           147, 122, 108,  70)
women <- c( 84,  87,  77,  84,  73,  66,  70,  35,  77,  73,
            56, 112,  56,  84,  80, 101,  66,  84)
level <- c(men, women)
sex <- c(rep("men", length(men)), rep("women", length(women)))
andro <- data.frame(level, sex)
andro
```

```
##    level   sex
## 1    217   men
## 2    123   men
## 3     80   men
## 4    140   men
## 5    115   men
## 6    135   men
## 7     59   men
## 8    126   men
## 9     70   men
## 10    63   men
## 11   147   men
## 12   122   men
## 13   108   men
## 14    70   men
## 15    84 women
## 16    87 women
## 17    77 women
## 18    84 women
## 19    73 women
## 20    66 women
## 21    70 women
## 22    35 women
## 23    77 women
## 24    73 women
## 25    56 women
## 26   112 women
## 27    56 women
## 28    84 women
```

```
## 29    80 women
## 30   101 women
## 31    66 women
## 32    84 women
```

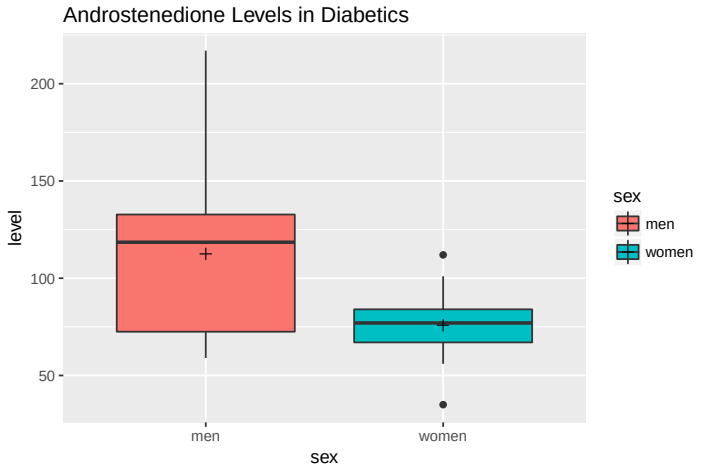
```
# numerical summaries
by(andro, sex, summary)
```

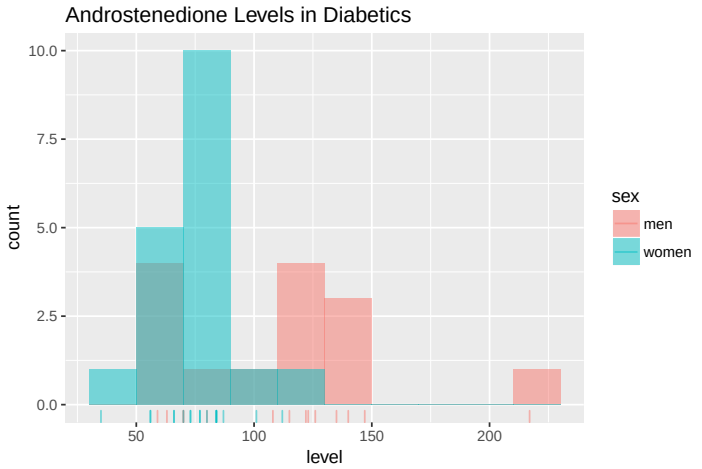
```
## sex: men
##      level          sex
##  Min.   : 59.0   men  :14
##  1st Qu.: 72.5   women: 0
##  Median :118.5
##  Mean   :112.5
##  3rd Qu.:132.8
##  Max.   :217.0
## ------------------------------------------------------
## sex: women
##      level           sex
##  Min.   : 35.00   men  : 0
##  1st Qu.: 67.00   women:18
##  Median : 77.00
##  Mean   : 75.83
##  3rd Qu.: 84.00
##  Max.   :112.00
```

```
c(sd(men), sd(women), IQR(men), IQR(women), length(men), length(women))
```

```
## [1] 42.75467 17.23625 60.25000 17.00000 14.00000 18.00000
```

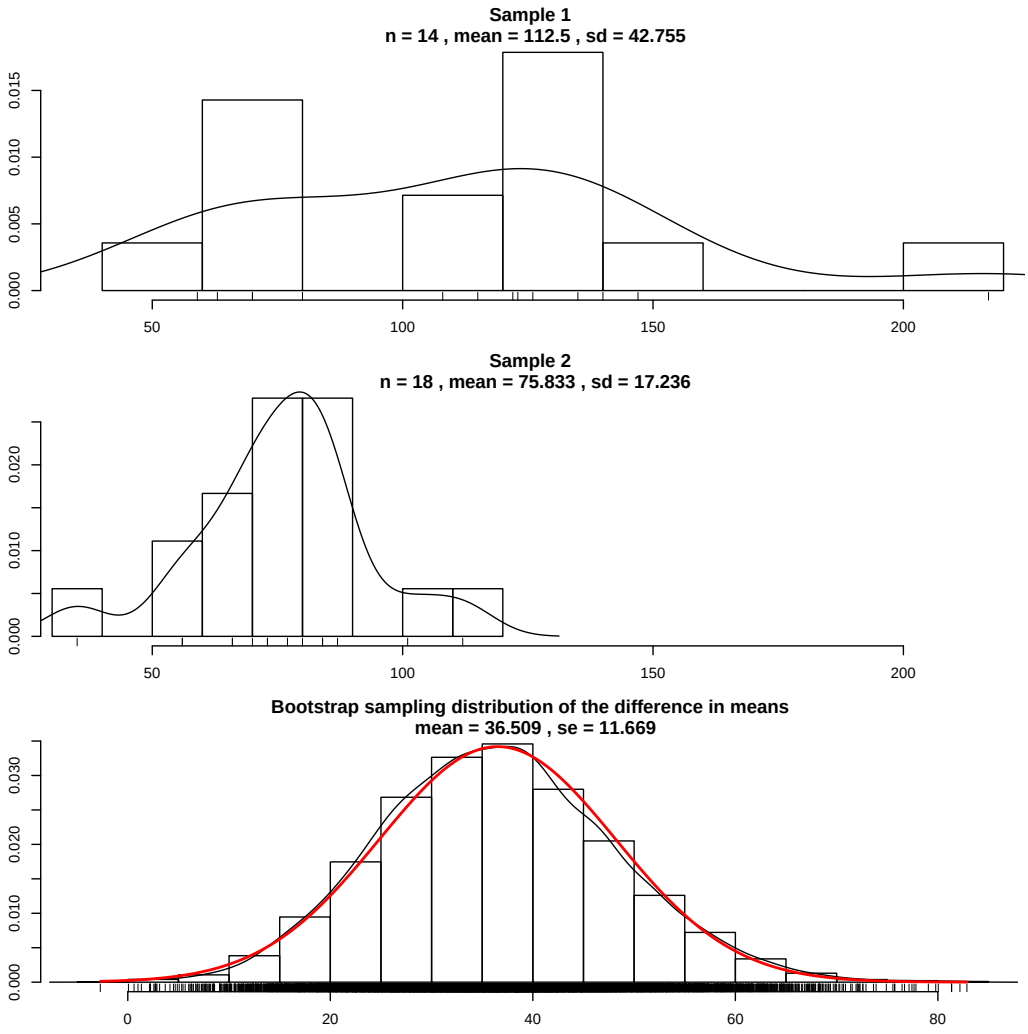
```
p <- ggplot(andro, aes(x = sex, y = level, fill=sex))
p <- p + geom_boxplot()
# add a "+" at the mean
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 3, size = 2)
#p <- p + coord_flip()
p <- p + labs(title = "Androstenedione Levels in Diabetics")
print(p)

p <- ggplot(andro, aes(x = level, fill=sex))
p <- p + geom_histogram(binwidth = 20, alpha = 0.5, position="identity")
p <- p + geom_rug(aes(colour=sex), alpha = 1/2)
p <- p + labs(title = "Androstenedione Levels in Diabetics")
print(p)
```





Because of the large difference in variances, I will be more comfortable with the Satterthwaite analysis here than the pooled variance analysis. The normality assumption of the difference in means appears to be met using the bootstrap assessment. The distribution of difference in means in the third plot looks very close to normal.

```
bs.two.samp.diff.dist(men, women)
```



1. Define the population parameters and hypotheses in words and notation

Let μ_1 and μ_2 be the mean androstenedione level for diabetic men and women, respectively.

In words: "The difference in population mean androstenedione levels between diabetic men and women is different from zero."

In notation: $H_0 : \mu_1 - \mu_2 = 0$ versus $H_A : \mu_1 - \mu_2 \neq 0$.

2. Calculate summary statistics from sample

(see above)

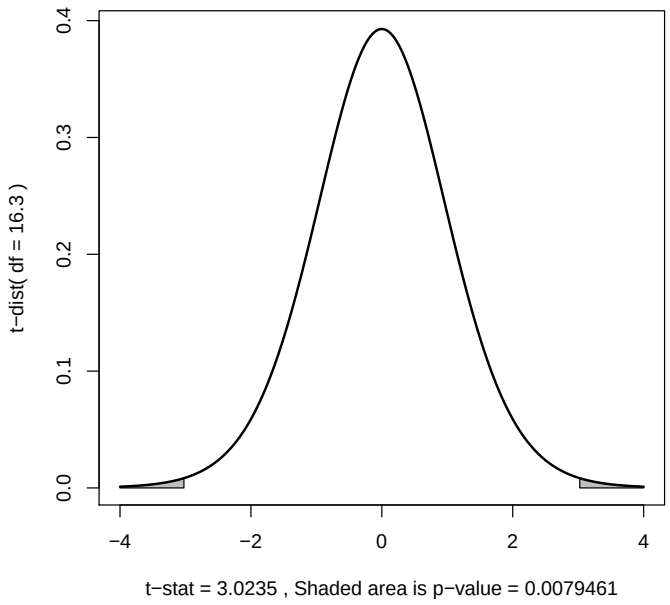
3. Specify confidence level, calculate t-stat, CI limits, p-value

Not assuming equal variances, Satterthwaite (Welch):

```
# two-sample t-test with unequal variances (Welch = Satterthwaite)
#   specified using data.frame and a formula, level by sex
t.summary <- t.test(level ~ sex, data = andro, var.equal = FALSE)
t.summary
```

```
##
##  Welch Two Sample t-test
##
## data:  level by sex
## t = 3.0235, df = 16.295, p-value = 0.007946
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  10.99555 62.33778
## sample estimates:
##   mean in group men mean in group women
##           112.50000            75.83333
```

```
# plot t-distribution with shaded p-value
t.dist.pval(t.summary)
```



4. Summarize in words The unequal-variance analysis suggests that $H_0 : \mu_1 - \mu_2 = 0$ is false, given the large t-statistic of 3.02 and two-sided p-value of 0.00795. Because the p-value < 0.05 we reject the Null hypothesis in favor of the Alternative hypothesis concluding that the difference in population mean androstenedione levels between diabetic men and women are different.

We are 95% confident confident that the difference in population means, $\mu_1 - \mu_2$, is between 11 and 62.3 ng/dL.

5. Check assumptions

As checked before, while the assumption of equal population variances is not met, the assumption that the distribution of $\bar{Y}_1 - \bar{Y}_2$ is normal using the bootstrap appeared reasonable.

As a comparison, let us examine the output for the pooled procedure (which is inappropriate since variances are unequal). The p-value for the pooled t-test is 0.002, whereas the 95% confidence limits are 14.1 and 59.2. That is, we are 95% confident that the population mean andro level for men exceeds that for women by at least 14.1 but by no more than 59.2. These results are qualitatively similar to the Satterthwaite conclusions.

3.5 One-Sided Tests

One-sided tests for two-sample problems are where the null hypothesis is $H_0 : \mu_1 - \mu_2 = 0$ but the alternative is directional, either $H_A : \mu_1 - \mu_2 < 0$ (i.e., $\mu_1 < \mu_2$) or $H_A : \mu_1 - \mu_2 > 0$ (i.e., $\mu_1 > \mu_2$). Once you understand the general form of rejection regions and p-values for one-sample tests, the one-sided two-sample tests do not pose any new problems. Use the t-statistic, with the appropriate tail of the t-distribution to define critical values and p-values. One-sided two-sample tests are directly implemented in R, by specifying the type of test with `alternative = "less"` or `alternative = "greater"`. One-sided confidence bounds are given with the one-sided tests.

3.6 Paired Analysis

With paired data, inferences on $\mu_1 - \mu_2$ are based on the sample of differences within pairs. By taking differences within pairs, two dependent samples are transformed into one sample, which contains the relevant information for inferences on $\mu_d = \mu_1 - \mu_2$. To see this, suppose the observations within a pair are Y_1 and Y_2. Then within each pair, compute the difference $d = Y_1 - Y_2$:

$$\begin{aligned} d_1 &= Y_{11} - Y_{21} \\ d_2 &= Y_{12} - Y_{22} \\ &\vdots \\ d_n &= Y_{1n} - Y_{2n} \end{aligned}$$

If the Y_1 data are from a population with mean μ_1 and the Y_2 data are from a population with mean μ_2, then the ds are a sample from a population with mean $\mu_d = \mu_1 - \mu_2$. Furthermore, if the sample of differences comes from a normal population, then we can use standard one-sample techniques on $d_1, \ldots, d_n$ to test $\mu_d = 0$ (that is, $\mu_1 = \mu_2$), and to get a CI for $\mu_d = \mu_1 - \mu_2$.

Let $\bar{d} = n^{-1} \sum_i d_i = \bar{Y}_1 - \bar{Y}_2$ be the sample mean of the differences (which is also the mean difference), and let s_d be the sample standard deviation of the

differences. The standard error of $\bar{d}$ is $SE_{\bar{d}} = s_d/\sqrt{n}$, where n is the number of pairs. The paired t-test (two-sided) CI for μ_d is given by $\bar{d} \pm t_{\text{crit}} SE_{\bar{d}}$. To test $H_0 : \mu_d = 0$ ($\mu_1 = \mu_2$) against $H_A : \mu_d \neq 0$ ($\mu_1 \neq \mu_2$), use

$$t_s = \frac{\bar{d} - 0}{SE_{\bar{d}}}$$

to compute a p-value as in a two-sided one-sample test. One-sided tests are evaluated in the usual way for one-sample tests on means.

A graphical analysis of paired data focuses on the **sample of differences**, and not on the original samples. In particular, the normality assumption is assessed on the sample of differences.

3.6.1 R Analysis

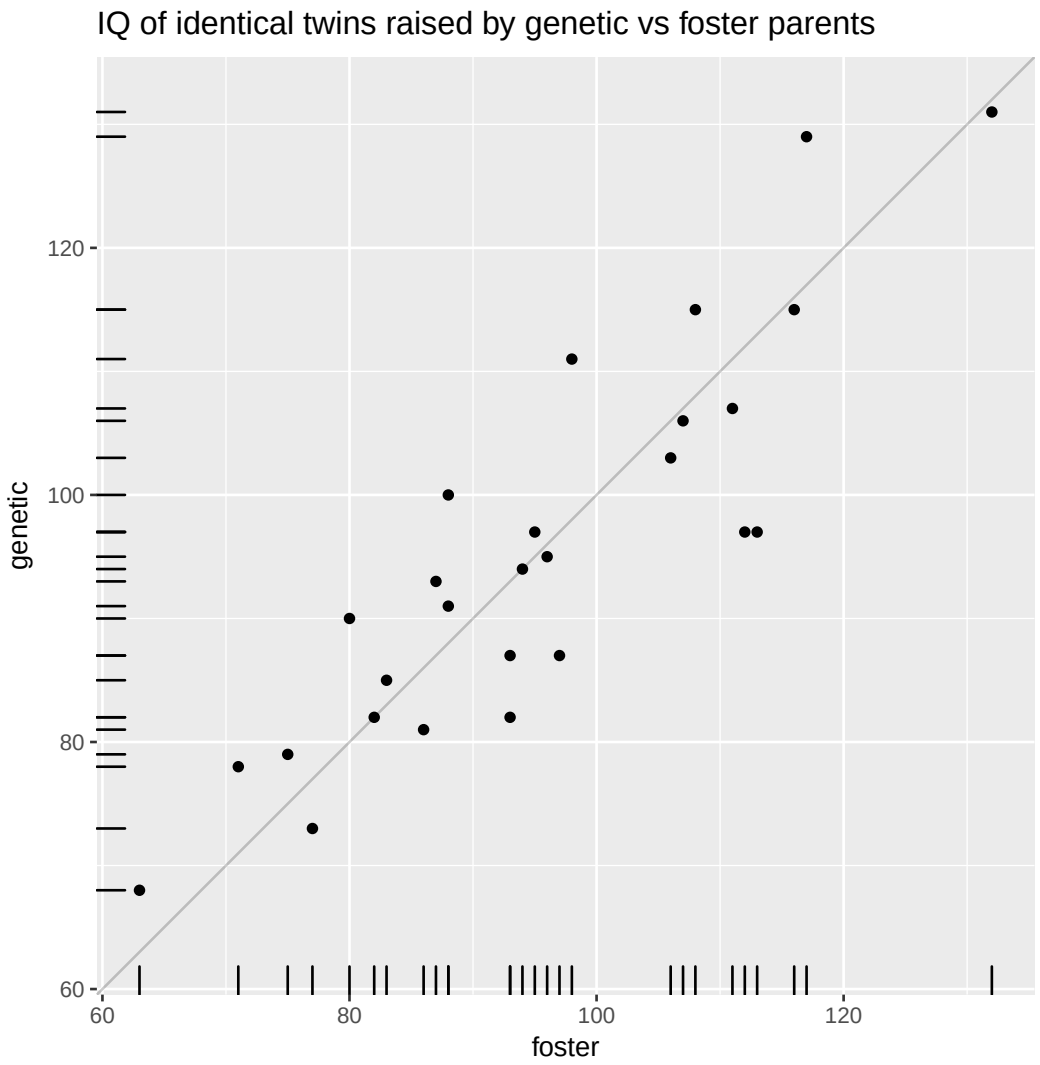
The most natural way to enter paired data is as two columns, one for each treatment group. You can then create a new column of differences, and do the usual one-sample graphical and inferential analysis on this column of differences, or you can do the paired analysis directly without this intermediate step.

Example: Paired Analysis of Data on Twins Burt (1966) presented data on IQ scores for identical twins that were raised apart, one by foster parents and one by the genetic parents. Assuming the data are a random sample of twin pairs, consider comparing the population mean IQs for twins raised at home to those raised by foster parents. Let μ_{f}=population mean IQ for twin raised by foster parents, and μ_{g}=population mean IQ for twin raised by genetic parents.

I created the data in the worksheet (c1=foster; c2=genetic), and computed the differences between the IQ scores for the children raised by the genetic and foster parents (c3=diff=genetic-foster). I also made a scatter plot of the genetic versus foster IQ scores.

```
#### Example: Paired Analysis of Data on Twins
# Data and numerical summaries
foster  <- c(82,  80,  88, 108, 116, 117, 132,  71,  75,  93,
             95,  88, 111,  63,  77,  86,  83,  93,  97,  87,
```

```
             94,  96, 112, 113, 106, 107,  98)
genetic <- c(82,  90,  91, 115, 115, 129, 131,  78,  79,  82,
             97, 100, 107,  68,  73,  81,  85,  87,  87,  93,
             94,  95,  97,  97, 103, 106, 111)
diff <- genetic - foster

axis.lim <- range(c(foster, genetic))

iq <- data.frame(foster, genetic, diff)

# scatterplot of foster and genetic IQs, with 1:1 line
p <- ggplot(iq, aes(x = foster, y = genetic))
# draw a 1:1 line, dots above line indicate "genetic > foster"
p <- p + geom_abline(intercept=0, slope=1, alpha=0.2)
p <- p + geom_point()
p <- p + geom_rug()
# make the axes square so it's a fair visual comparison
p <- p + coord_equal()
p <- p + scale_x_continuous(limits=axis.lim)
p <- p + scale_y_continuous(limits=axis.lim)
p <- p + labs(title = "IQ of identical twins raised by genetic vs foster parents")
print(p)
```

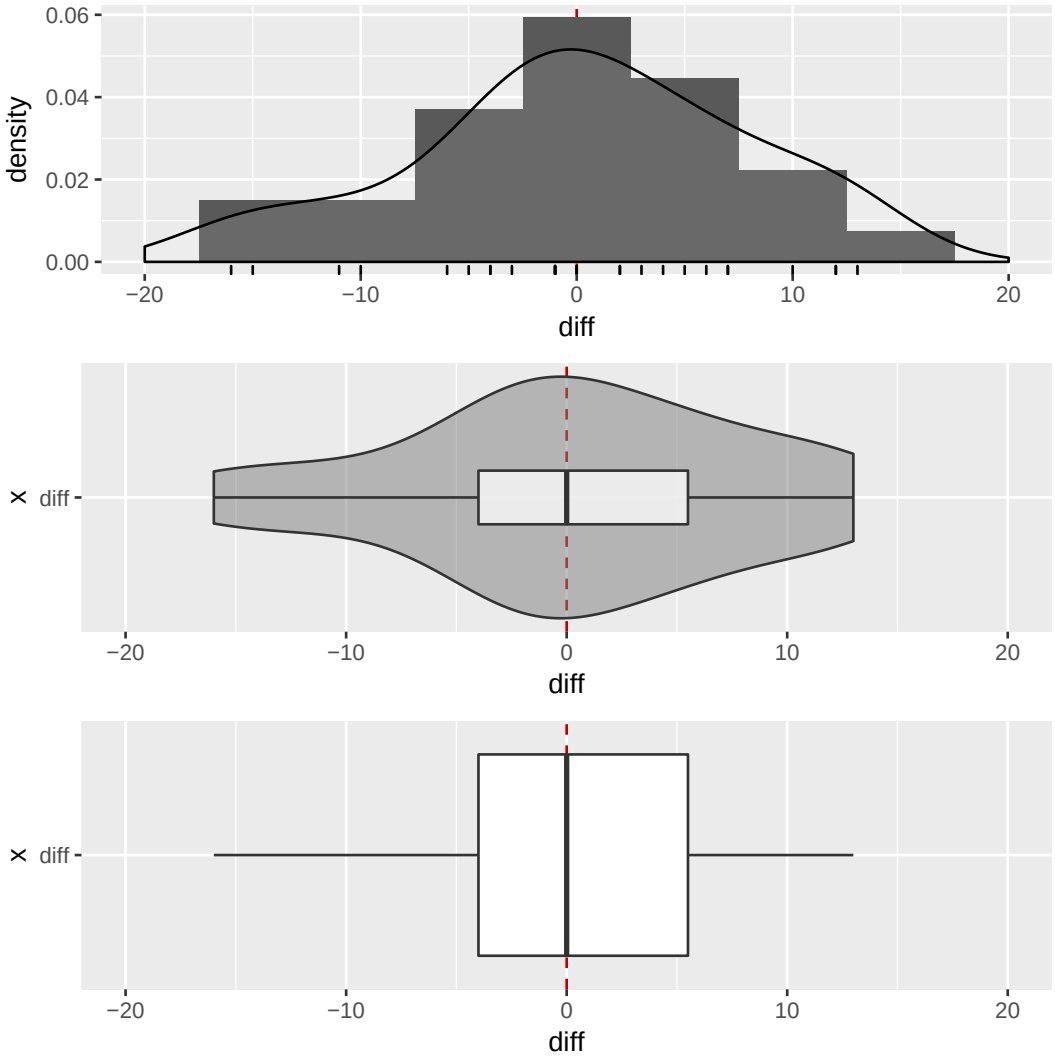


This plot of IQ scores shows that scores are related within pairs of twins. This is consistent with the need for a paired analysis.

Given the sample of differences, I created a boxplot and a stem and leaf display, neither which showed marked deviation from normality. The boxplot is centered at zero, so one would not be too surprised if the test result is insignificant.

```
p1 <- ggplot(iq, aes(x = diff))
p1 <- p1 + scale_x_continuous(limits=c(-20,+20))
# vertical line at 0
p1 <- p1 + geom_vline(xintercept=0, colour="#BB0000", linetype="dashed")
p1 <- p1 + geom_histogram(aes(y=..density..), binwidth=5)
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

# violin plot
p2 <- ggplot(iq, aes(x = "diff", y = diff))
```

```
p2 <- p2 + scale_y_continuous(limits=c(-20,+20))
p2 <- p2 + geom_hline(yintercept=0, colour="#BB0000", linetype="dashed")
p2 <- p2 + geom_violin(fill = "gray50", alpha=1/2)
p2 <- p2 + geom_boxplot(width = 0.2, alpha = 3/4)
p2 <- p2 + coord_flip()


# boxplot
p3 <- ggplot(iq, aes(x = "diff", y = diff))
p3 <- p3 + scale_y_continuous(limits=c(-20,+20))
p3 <- p3 + geom_hline(yintercept=0, colour="#BB0000", linetype="dashed")
p3 <- p3 + geom_boxplot()
p3 <- p3 + coord_flip()

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



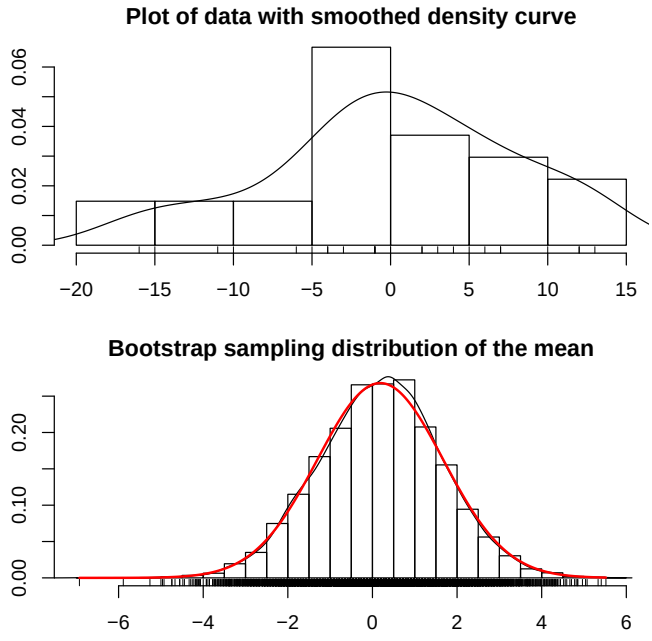
The normality assumption of the sample mean for a one-sample test is satisfied (below, left).

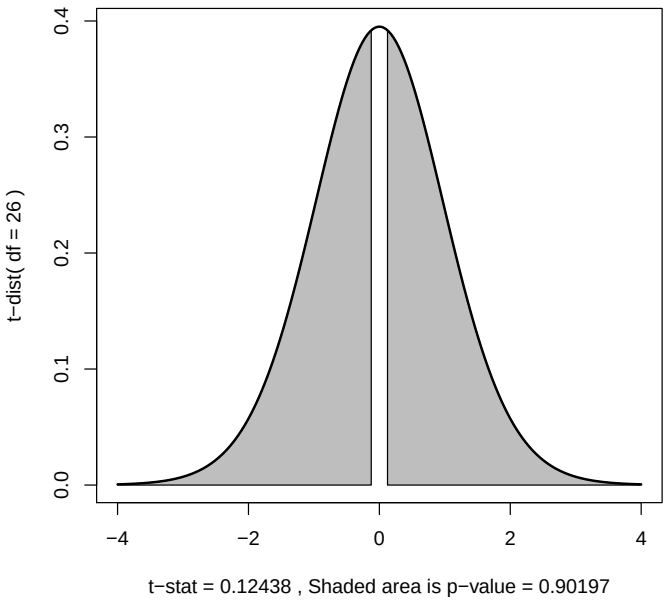
Given the sample of differences, I generated a one-sample CI and test. The hypothesis under test is $\mu_d = \mu_g - \mu_f = 0$. The p-value for this test is large. We do not have sufficient evidence to claim that the population mean IQs for twins raised apart are different. This is consistent with the CI for μ_d given below, which covers zero.

```
bs.one.samp.dist(iq$diff)

# one-sample t-test of differences (paired t-test)
t.summary <- t.test(iq$diff)
t.summary
```

```
##
##  One Sample t-test
##
## data:  iq$diff
## t = 0.12438, df = 26, p-value = 0.902
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -2.875159  3.245529
## sample estimates:
## mean of x
## 0.1851852
```

```
# plot t-distribution with shaded p-value
t.dist.pval(t.summary)
```





Alternatively, I can generate the test and CI directly from the raw data in two columns, specifying `paired=TRUE`. This gives the following output, which

leads to identical conclusions to the earlier analysis.

```
# two-sample paired t-test
t.summary <- t.test(iq$genetic, iq$foster, paired=TRUE)
t.summary
```

```
##
##  Paired t-test
##
## data:  iq$genetic and iq$foster
## t = 0.12438, df = 26, p-value = 0.902
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.875159  3.245529
## sample estimates:
## mean of the differences
##               0.1851852
```

You might ask why I tortured you by doing the first analysis, which required creating and analyzing the sample of differences, when the alternative and equivalent second analysis is so much easier. (A later topic deals with non-parametric analyses of paired data for which the differences must be first computed.)

Remark: I could have defined the difference to be the foster IQ score minus the genetic IQ score. How would this change the conclusions?

Example: Paired Comparisons of Two Sleep Remedies The following data give the amount of sleep gained in hours from two sleep remedies, A and B, applied to 10 individuals who have trouble sleeping an adequate amount. Negative values imply sleep loss. In 9 of the 10 individuals, the sleep gain on B exceeded that on A.

Let μ_A = population mean sleep gain (among troubled sleepers) on remedy A, and μ_B = population mean sleep gain (among troubled sleepers) on remedy B. Consider testing $H_0 : \mu_B - \mu_A = 0$ or equivalently $\mu_d = 0$, where $\mu_d = \mu_B - \mu_A$.

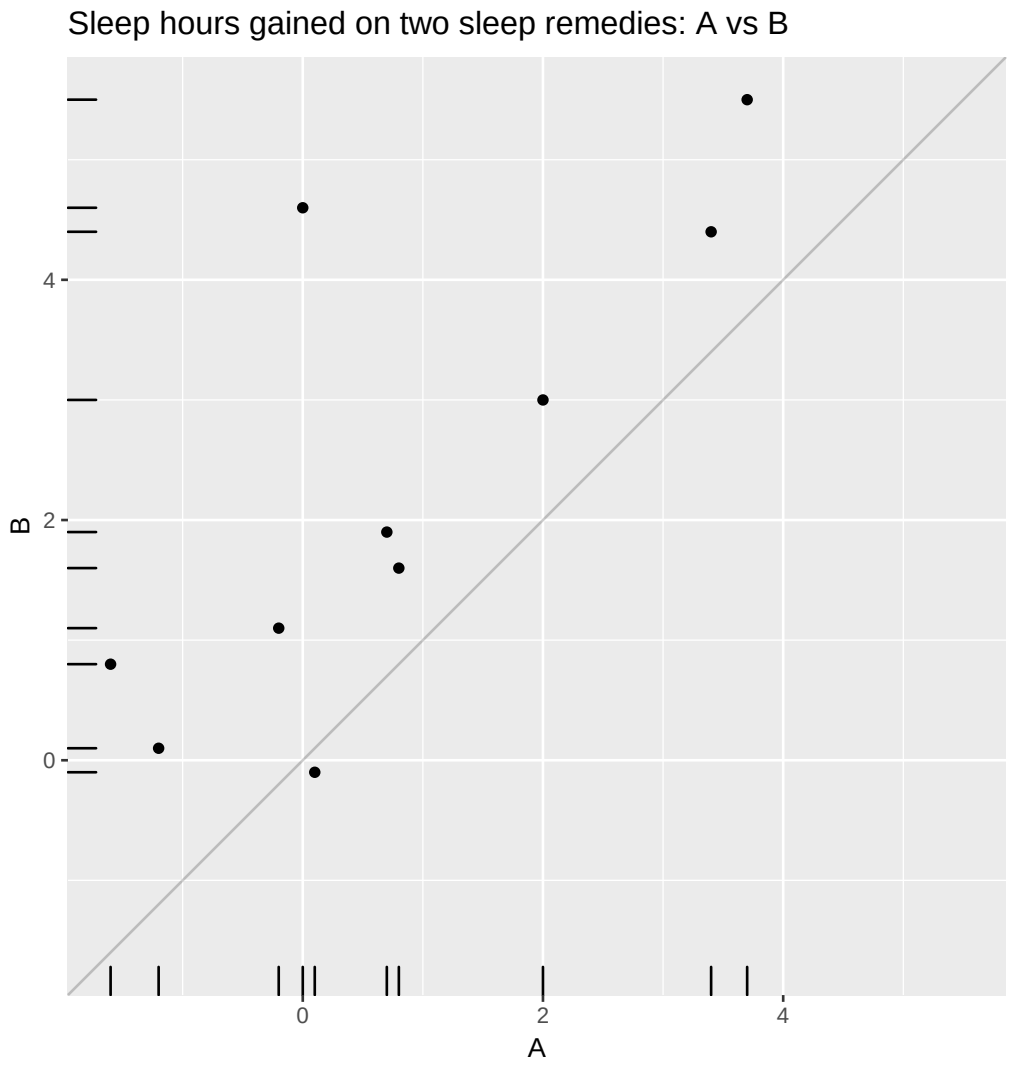
The observed distribution of differences between B and A is slightly skewed to the right, with a single outlier in the upper tail. The normality assumption of the standard one-sample t-test and CI are suspect here. I will continue with

the analysis, anyways.

```
# Data
sleep <- read.table(text="
   A    B
 0.7  1.9
-1.6  0.8
-0.2  1.1
-1.2  0.1
 0.1 -0.1
 3.4  4.4
 3.7  5.5
 0.8  1.6
 0.0  4.6
 2.0  3.0
", header = TRUE)

# calculate paired difference bewteen two remedies, D = B - A
sleep$D <- sleep$B - sleep$A
sleep
```

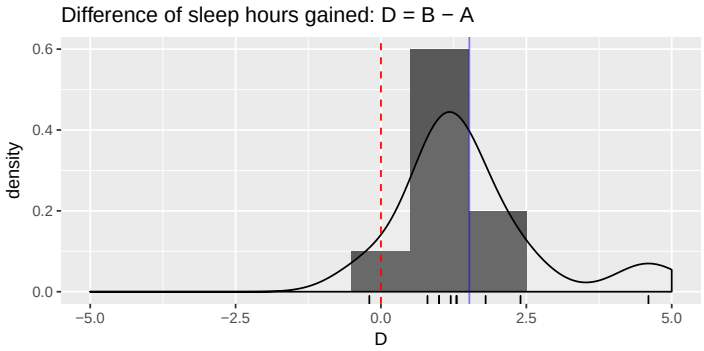
```
##       A    B    D
## 1   0.7  1.9  1.2
## 2  -1.6  0.8  2.4
## 3  -0.2  1.1  1.3
## 4  -1.2  0.1  1.3
## 5   0.1 -0.1 -0.2
## 6   3.4  4.4  1.0
## 7   3.7  5.5  1.8
## 8   0.8  1.6  0.8
## 9   0.0  4.6  4.6
## 10  2.0  3.0  1.0
```

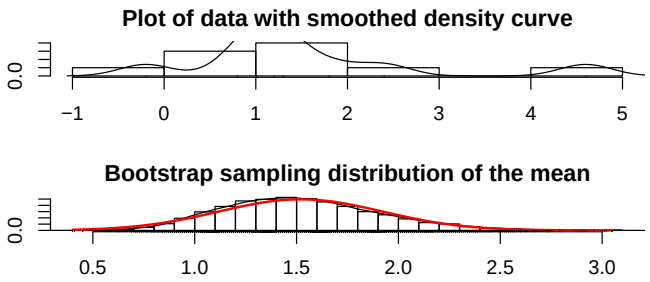
```
# determine range of each axis to use most extreme of each for square plot below
axis.lim <- range(c(sleep$A, sleep$B))

library(ggplot2)
# scatterplot of A and B sleep times, with 1:1 line
p <- ggplot(sleep, aes(x = A, y = B))
# draw a 1:1 line, dots above line indicate "B > A"
p <- p + geom_abline(intercept=0, slope=1, alpha=0.2)
p <- p + geom_point()
p <- p + geom_rug()
# make the axes square so it's a fair visual comparison
p <- p + coord_equal()
p <- p + scale_x_continuous(limits=axis.lim)
p <- p + scale_y_continuous(limits=axis.lim)
p <- p + labs(title = "Sleep hours gained on two sleep remedies: A vs B")
print(p)
```



There is evidence here against the normality assumption of the sample mean. We'll continue anyway (in practice we'd use a nonparametric method, instead, in a later chapter).

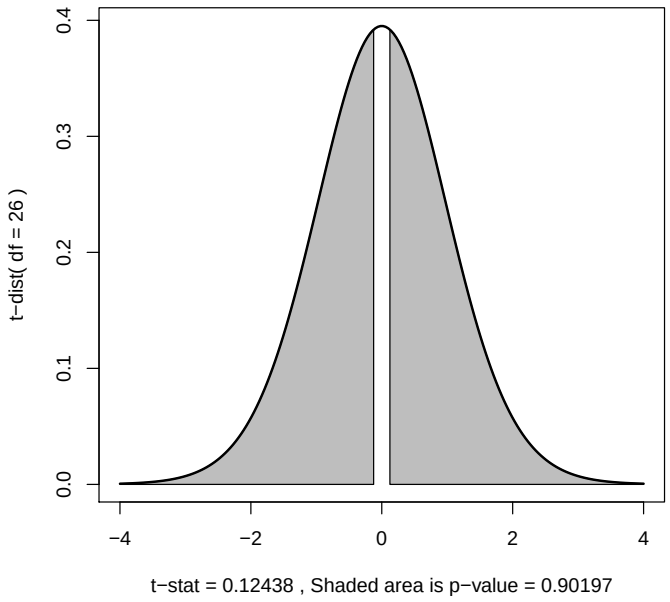
```
library(ggplot2)
p1 <- ggplot(sleep, aes(x = D))
p1 <- p1 + scale_x_continuous(limits=c(-5,+5))
p1 <- p1 + geom_histogram(aes(y=..density..), binwidth = 1)
p1 <- p1 + geom_density(alpha=0.1, fill="white", adjust = 2)
# vertical reference line at 0
p1 <- p1 + geom_vline(xintercept = 0, colour="red", linetype="dashed")
p1 <- p1 + geom_vline(xintercept = mean(sleep$D), colour="blue", alpha = 0.5)
p1 <- p1 + geom_rug()
p1 <- p1 + labs(title = "Difference of sleep hours gained: D = B - A")
print(p1)

bs.one.samp.dist(sleep$D)
```





The p-value for testing H_0 is 0.004. We'd reject H_0 at the 5% or 1% level, and conclude that the population mean sleep gains on the remedies are different. We are 95% confident that μ_B exceeds μ_A by between 0.61 and 2.43 hours. Again, these results must be reported with caution, because the normality assumption is unreasonable. However, the presence of outliers tends to make the t-test and CI conservative, so we'd expect to find similar conclusions if we used the nonparametric methods discussed later in the semester.

```
# one-sample t-test of differences (paired t-test)
t.summary <- t.test(sleep$d)
## Warning in is.na(x):  is.na() applied to non-(list or vector) of type 'NULL'
## Warning in mean.default(x):  argument is not numeric or logical:  returning NA
## Error in var(x):  'x' is NULL
t.summary
##
##  Paired t-test
##
## data:  iq$genetic and iq$foster
## t = 0.12438, df = 26, p-value = 0.902
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.875159  3.245529
## sample estimates:
## mean of the differences
##               0.1851852
# plot t-distribution with shaded p-value
t.dist.pval(t.summary)
```



Question: In what order should the remedies be given to the patients?

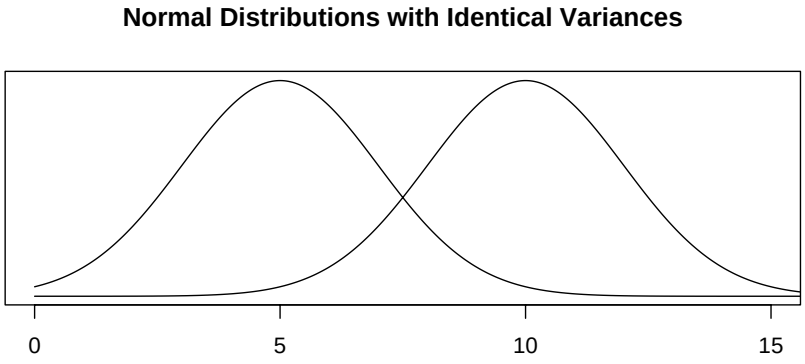
■ CLICKER Qs — Reporting results, STT.06.03.010 ■

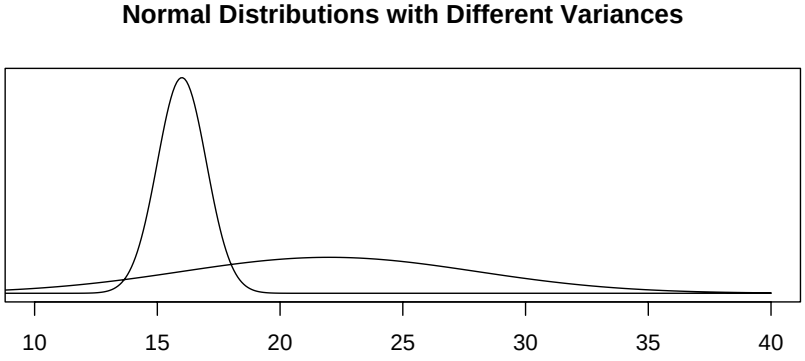
3.7 Should You Compare Means?

The mean is the most common feature on which two distributions are compared. You should not, however, blindly apply the two-sample tests (paired or unpaired) without asking yourself whether the means are the relevant feature to compare. This issue is not a big concern when, as highlighted in the first graph below, the two (normal) populations have equal spreads or standard deviations. In such cases the difference between the two population means is equal to the difference between any fixed percentile for the two distributions, so the mean difference is a natural measure of difference.

Consider instead the hypothetical scenario depicted in the bottom pane below, where the population mean lifetimes using two distinct drugs for a fatal

disease are $\mu_1 = 16$ months from time of diagnosis and $\mu_2 = 22$ months from time of diagnosis, respectively. The standard deviations under the two drugs are $\sigma_1 = 1$ and $\sigma_2 = 6$, respectively. The second drug has the higher mean lifetime, but at the expense of greater risk. For example, the first drug gives you a 97.7% chance of living at least 14 months, whereas the second drug only gives you a 90.8% chance of living at least 14 months. Which drug is best? It depends on what is important to you, a higher expected lifetime or a lower risk of dying early.





Chapter 4

Checking Assumptions

Contents


```
## Warning in file(filename, "r", encoding = encoding): cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding): cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

assess the assumptions visually and via formal tests.

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

10. Model assumptions.

4.1 Introduction

Almost all statistical methods make assumptions about the data collection process and the shape of the population distribution. If you reject the null hypothesis in a test, then a reasonable conclusion is that the null hypothesis is false, provided all the distributional assumptions made by the test are satisfied. If the assumptions are not satisfied then that alone might be the cause of rejecting H_0. Additionally, if you fail to reject H_0, that could be caused solely by failure to satisfy assumptions also. Hence, you should always check assumptions to the best of your abilities.

Two assumptions that underly the tests and CI procedures that I have discussed are that the data are a random sample, and that the population frequency curve is normal. For the pooled variance two-sample test the population variances are also required to be equal.

The random sample assumption can often be assessed from an understanding of the data collection process. Unfortunately, there are few general tests for checking this assumption. I have described exploratory (mostly visual) methods to assess the normality and equal variance assumption. I will now discuss formal methods to assess these assumptions.

4.2 Testing Normality

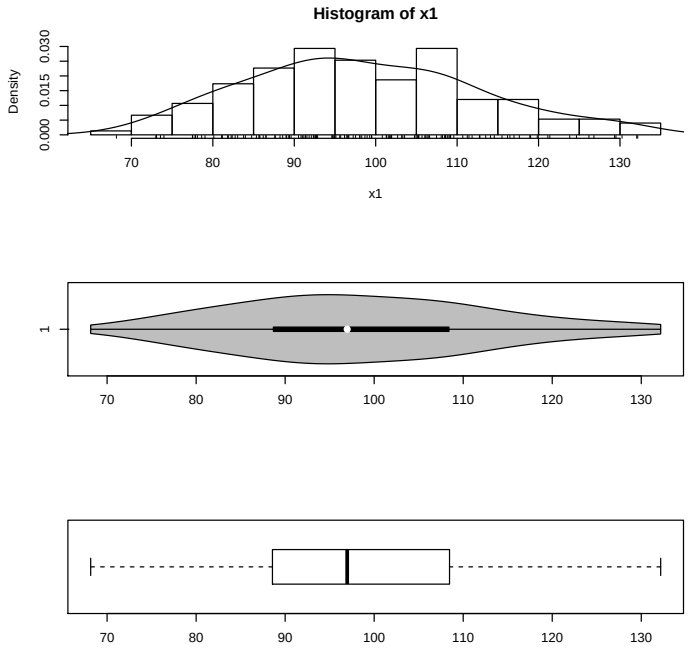
An informal test of normality can be based on a **normal scores plot**, sometimes called a **rankit plot** or a **normal probability plot** or a **normal QQ plot** (QQ = quantile-quantile). You plot the quantiles of the data against the quantiles of the normal distribution, or **expected normal order statistics** (in a standard normal distribution) for a sample with the given number of observations. The normality assumption is plausible if the plot is fairly linear. I give below several plots often seen with real data, and what they indicate about the underlying distribution.

There are multiple ways to produce QQ plots in R. The shape can depend

upon whether you plot the normal scores on the x-axis or the y-axis. It is conventional to plot the data on the y-axis and the normal scores on the x-axis.

Let's start with some data from a normal distribution.

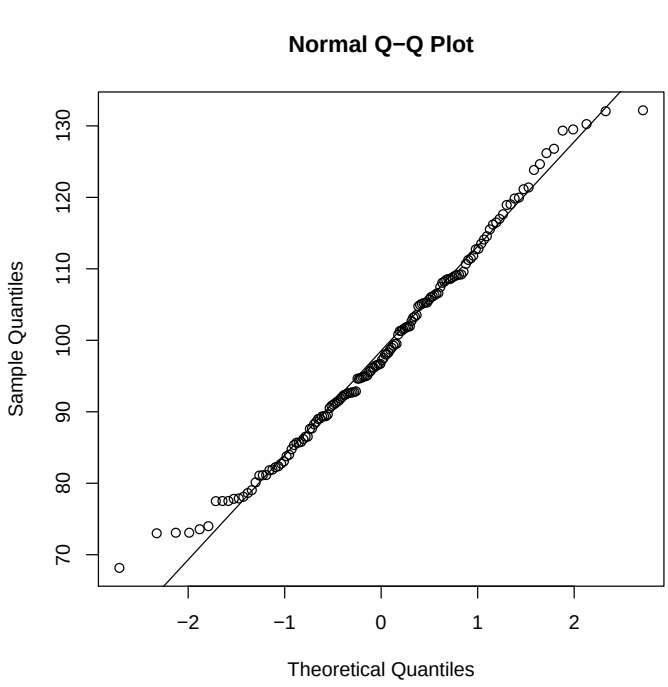
```
#### sample from normal distribution
x1 <- rnorm(150, mean = 100, sd = 15)

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)
```

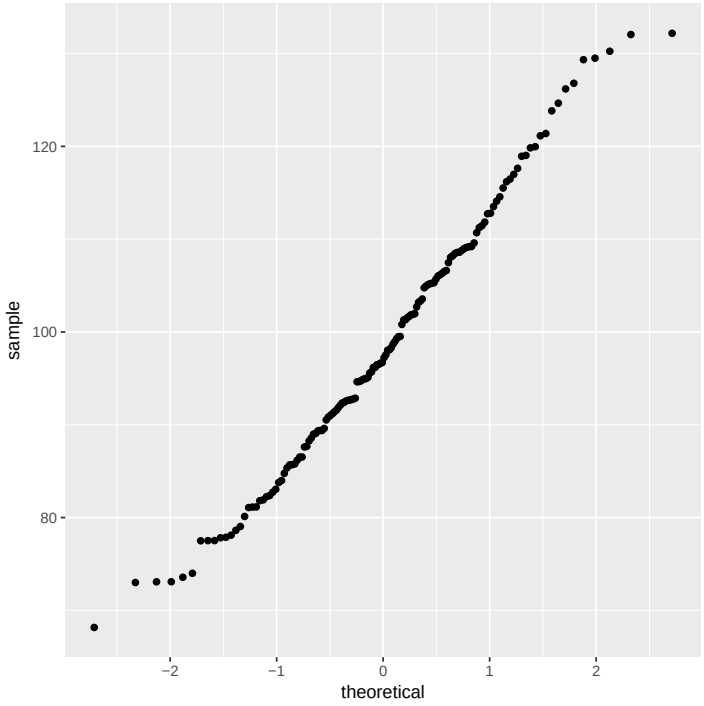


There are many ways to get adequate QQ plots. Consider how outliers shows up in the QQ plot. There may be isolated points on ends of the QQ plot, but only on the right side is there an outlier. How could you have identified that the right tail looks longer than the left tail from the QQ plot?

```
#### QQ plots
# R base graphics
par(mfrow=c(1,1))
```

```
# plots the data vs their normal scores
qqnorm(x1)
# plots the reference line
qqline(x1)


# ggplot2 graphics
library(ggplot2)
# http://had.co.nz/ggplot2/stat_qq.html
df <- data.frame(x1)
# stat_qq() below requires "sample" to be assigned a data.frame column
p <- ggplot(df, aes(sample = x1))
# plots the data vs their normal scores
p <- p + stat_qq()
print(p)
```





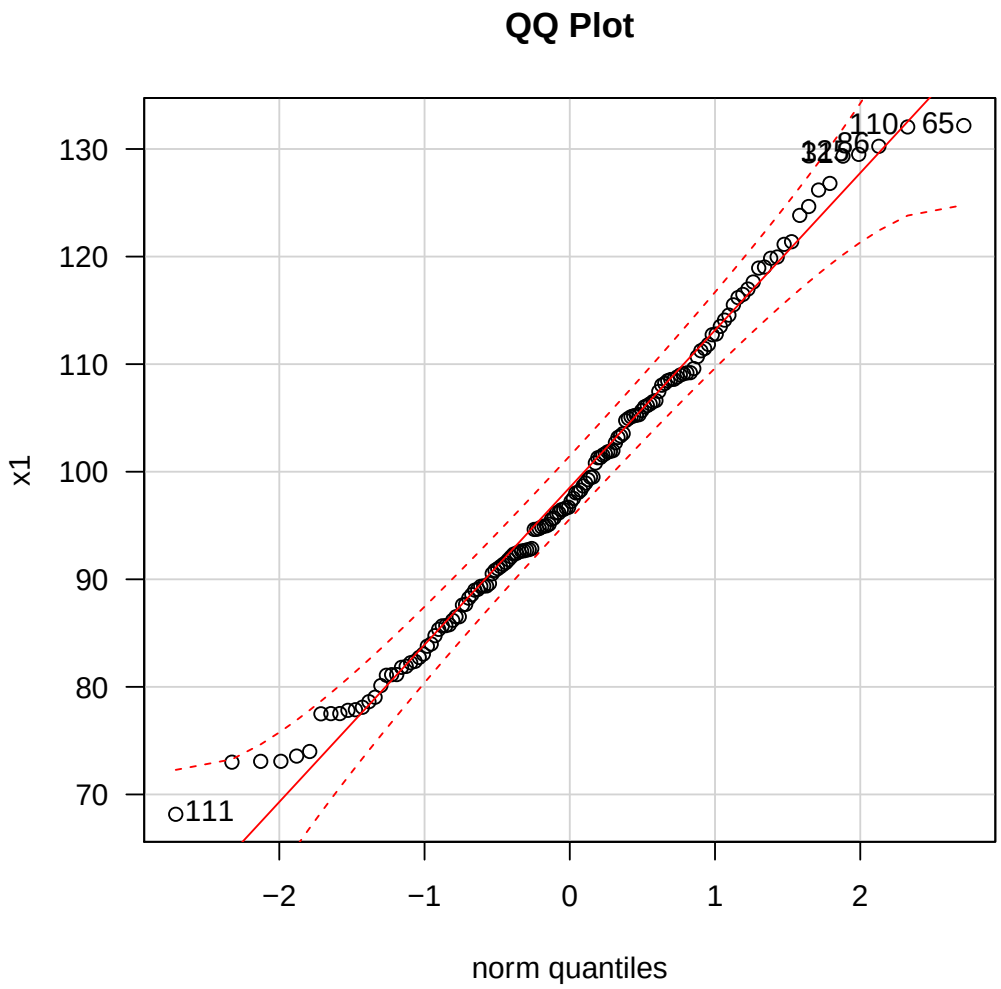
If you lay a straightedge along the bulk of the plot (putting in a regression line is not the right way to do it, even if it is easy), you see that the most extreme point on the right is a little below the line, and the last few points on the left a little above the line. What does this mean? The point on the right corresponds to a data value *more extreme* than expected from a normal distribution (the straight line is where expected and actual coincide). Extreme points on the right are above the line. What about the left? Extreme points there should be *above* the line — since the deviations from the line are above it on the left, those points are also *more extreme* than expected.

Even more useful is to add confidence intervals (point-wise, not family-wise — you will learn the meaning of those terms in the ANOVA section). You don't expect a sample from a normally distributed population to have a normal scores plot that falls exactly on the line, and the amount of deviation depends upon the sample size.

The best QQ plot I could find is available in the `car` package called `qqPlot`. Note that with the `dist=` option you can use this technique to see if the data appear from lots of possible distributions, not just normal.

```
par(mfrow=c(1,1))
# Normality of Residuals
library(car)
# qq plot for studentized resid
#   las    =  1 : turns labels on y-axis to read horizontally
#   id.n   =  n : labels n most extreme observations, and outputs to console
#   id.cex =  1 : is the size of those labels
#   lwd    =  1 : line width
qqPlot(x1, las = 1, id.n = 6, id.cex = 1, lwd = 1, main="QQ Plot")
```

```
##  65 110  86 125  31 111
## 150 149 148 147 146   1
```



In this case the x-axis is labelled "norm quantiles". You only see a couple of data values outside the limits (in the tails, where it usually happens). You expect around 5% outside the limits, so there is no indication of non-normality here. I *did* sample from a normal population.

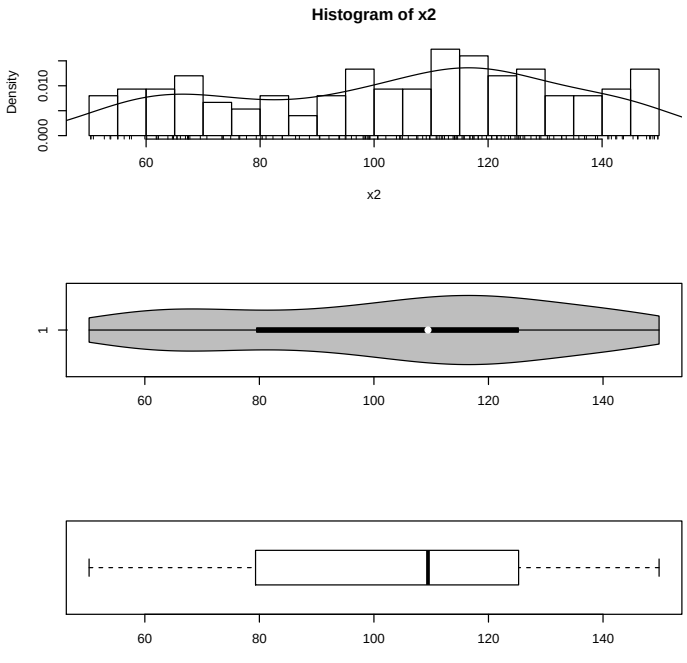
4.2.1 Normality tests on non-normal data

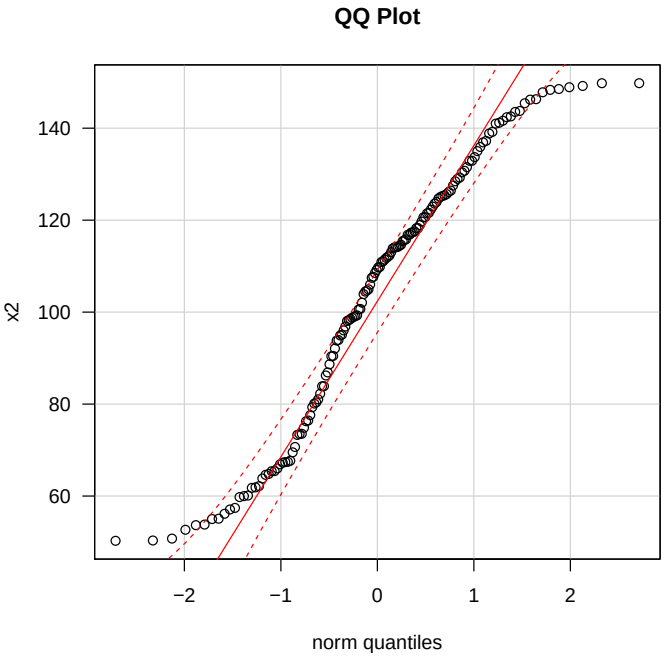
Let's turn to examples of sampling from other, non-normal distributions to see how the normal QQ plot identifies important features.

Light-tailed symmetric (Uniform)

```
#### Light-tailed symmetric (Uniform)
# sample from uniform distribution
```

```r
x2 <- runif(150, min = 50, max = 150)

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)


par(mfrow=c(1,1))
qqPlot(x2, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot")
```

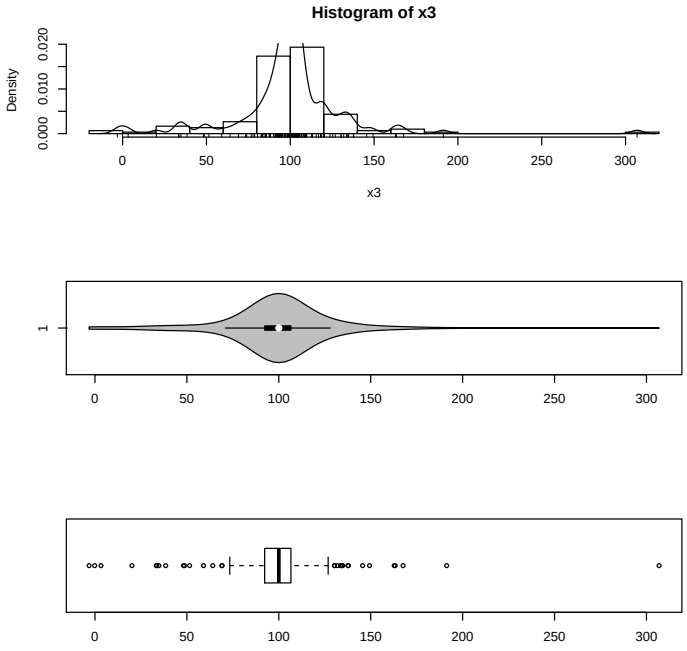


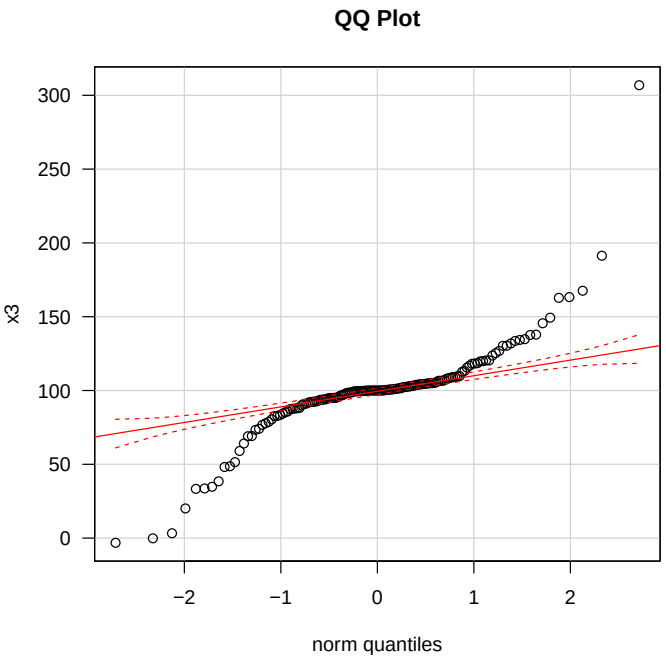


Heavy-tailed (fairly) symmetric (Normal-squared)

```r
#### Heavy-tailed (fairly) symmetric (Normal-squared)
# sample from normal distribution
x3.temp <- rnorm(150, mean = 0, sd = 1)
x3 <- sign(x3.temp)*x3.temp^2 * 15 + 100

par(mfrow=c(3,1))
```

```r
# 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)


par(mfrow=c(1,1))
qqPlot(x3, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot")
```

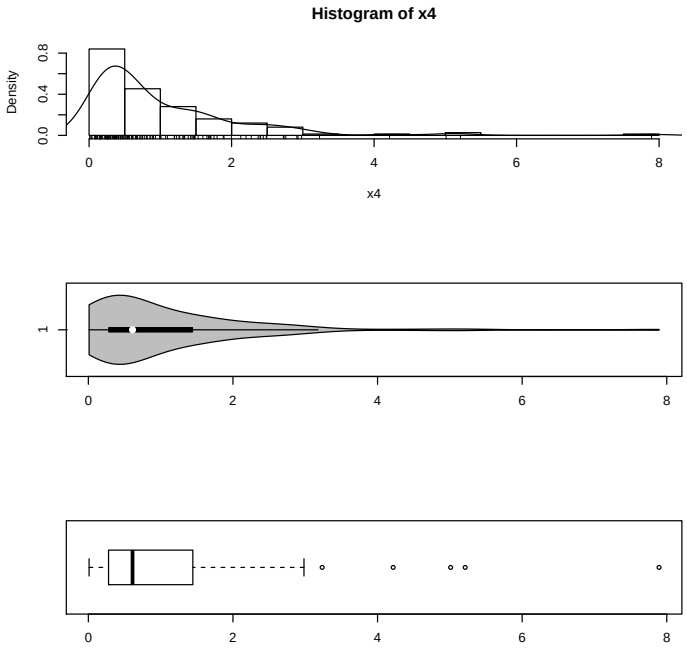


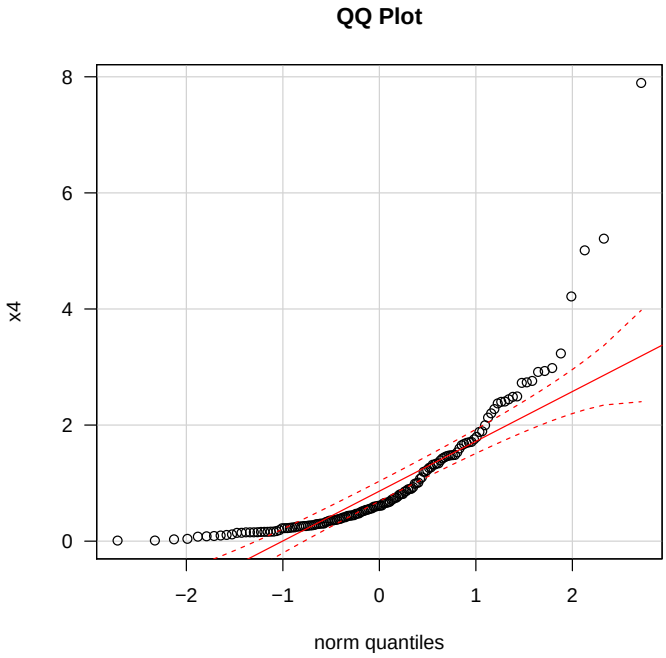


Right-skewed (Exponential)

```r
#### Right-skewed (Exponential)
# sample from exponential distribution
x4 <- rexp(150, 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)
```

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

# boxplot
boxplot(x4, horizontal=TRUE)


par(mfrow=c(1,1))
qqPlot(x4, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot")
```

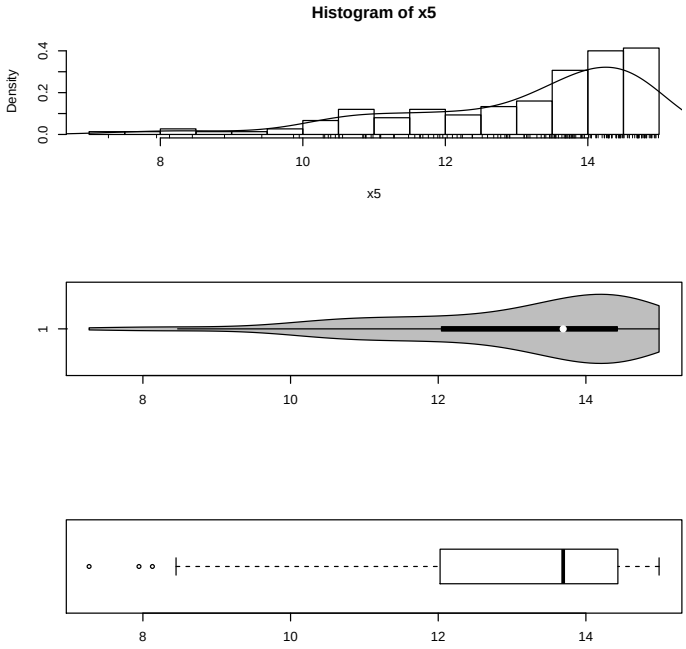


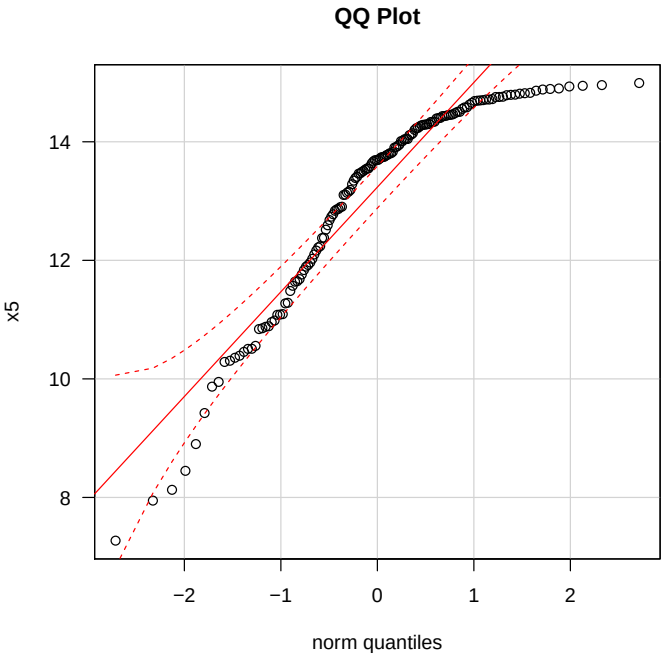


Left-skewed (Exponential, reversed)

```r
#### Left-skewed (Exponential, reversed)
# sample from exponential distribution
x5 <- 15 - rexp(150, 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")
```

```r
# boxplot
boxplot(x5, horizontal=TRUE)


par(mfrow=c(1,1))
qqPlot(x5, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot")
```





Notice how striking is the lack of linearity in the QQ plot for all the non-normal distributions, particularly the symmetric light-tailed distribution where the boxplot looks fairly good. The QQ plot is a sensitive measure of normality. Let us summarize the patterns we see regarding tails in the plots:

	Tail	
Tail Weight	Left	Right
Light	Left side of plot points left	Right side of plot points right
Heavy	Left side of plot points down	Right side of plot points up

4.3 Formal Tests of Normality

A formal test of normality is based on the **correlation** between the data and the normal scores. The correlation is a measure of the strength of a linear relationship, with the sign of the correlation indicating the direction of the

relationship (that is, + for increasing relationship, and − for decreasing). The correlation varies from −1 to +1. In a normal scores plot, you are looking for a correlation close to +1. Normality is rejected if the correlation is too small. Critical values for the correlation test of normality, which is commonly called the **Shapiro-Wilk** test, can be found in many texts.

R has several tests of normality. The **Shapiro-Wilk** test `shapiro.test()` is a base function. The R package `nortest` has five others: the Anderson-Darling test `ad.test()` is useful, related to the **Kolmogorov-Smirnov** (Lilliefors) test `lillie.test()` which is commonly used in many scientific disciplines but is essentially useless, the Cramer-von Mises test `cvm.test()`, and two more. Some packages also have the **Ryan-Joiner** test (closely related to the Shapiro-Wilk test).

Extreme outliers and skewness have the biggest effects on standard methods based on normality. The Shapiro-Wilk (SW) test is better at picking up these problems than the Kolmogorov-Smirnov (KS) test. The KS test tends to highlight deviations from normality in the center of the distribution. These types of deviations are rarely important because they do not have a noticeable effect on the operating characteristics of the standard methods. The AD and RJ tests are modifications designed to handle some of these objections.

Tests for normality may have low power in small to moderate sized samples. Visual assessment of normality is often more valuable than a formal test. The tests for the distributions of data above are below and in Figure 4.1.

Normal distribution

```
#### Formal Tests of Normality
shapiro.test(x1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x1
## W = 0.98584, p-value = 0.1289
```

```
library(nortest)
ad.test(x1)
```

```
##
##  Anderson-Darling normality test
##
```

```
## data:  x1
## A = 0.40732, p-value = 0.3446
# lillie.test(x1)
cvm.test(x1)
##
##  Cramer-von Mises normality test
##
## data:  x1
## W = 0.05669, p-value = 0.4159
```

Light-tailed symmetric

```
shapiro.test(x2)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x2
## W = 0.95252, p-value = 5.336e-05
```

```
library(nortest)
ad.test(x2)
```

```
##
##  Anderson-Darling normality test
##
## data:  x2
## A = 1.9426, p-value = 5.644e-05
```

```
# lillie.test(x2)
cvm.test(x2)
```

```
##
##  Cramer-von Mises normality test
##
## data:  x2
## W = 0.29567, p-value = 0.0003642
```

Heavy-tailed (fairly) symmetric

```
shapiro.test(x3)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x3
## W = 0.79633, p-value = 3.587e-13
```

```
library(nortest)
ad.test(x3)
```

```
##
##  Anderson-Darling normality test
##
## data:  x3
## A = 9.1433, p-value < 2.2e-16
```

```
# lillie.test(x3)
cvm.test(x3)
```

```
## Warning in cvm.test(x3):  p-value is smaller
than 7.37e-10, cannot be computed more accurately
##
##  Cramer-von Mises normality test
##
## data:  x3
## W = 1.8248, p-value = 7.37e-10
```

Right-skewed

```
shapiro.test(x4)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x4
## W = 0.74125, p-value = 5.872e-15
```

```
library(nortest)
ad.test(x4)
```

```
##
##  Anderson-Darling normality test
##
## data:  x4
## A = 9.3715, p-value < 2.2e-16
```

```
# lillie.test(x4)
cvm.test(x4)
```

```
## Warning in cvm.test(x4):  p-value is smaller
than 7.37e-10, cannot be computed more accurately
##
##  Cramer-von Mises normality test
##
## data:  x4
## W = 1.6537, p-value = 7.37e-10
```

Left-skewed

```
shapiro.test(x5)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x5
## W = 0.8743, p-value = 5.933e-10
```

```
library(nortest)
ad.test(x5)
```

```
##
##  Anderson-Darling normality test
##
## data:  x5
## A = 6.0016, p-value = 7.938e-15
```

```
# lillie.test(x5)
cvm.test(x5)
```

```
##
##  Cramer-von Mises normality test
##
## data:  x5
## W = 1.0553, p-value = 9.648e-10
```

Figure 4.1: Normality tests for non-normal distributions

Example: Paired Differences on Sleep Remedies The following boxplot and normal scores plots suggest that the underlying distribution of differences (for the paired sleep data taken from the previous chapter) is reasonably symmetric, but heavy tailed. The p-value for the SW test of normality is 0.042, and for the AD test is 0.029, both of which call into question a normality assumption. A non-parametric test comparing the sleep remedies (one that does not assume normality) is probably more appropriate here. We will return to these data later.

```
# Normality tests
shapiro.test(sleep$d)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sleep$d
## W = 0.83798, p-value = 0.04173
```

```
library(nortest)
ad.test(sleep$d)
```

```
##
##  Anderson-Darling normality test
##
## data:  sleep$d
## A = 0.77378, p-value = 0.02898
```

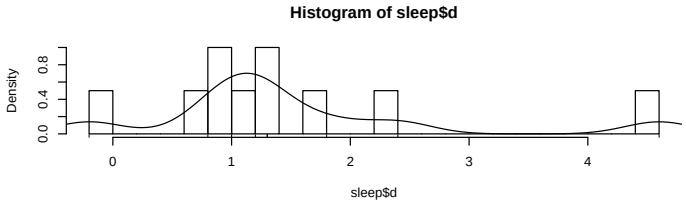
```
# lillie.test(sleep£d)
cvm.test(sleep$d)
```

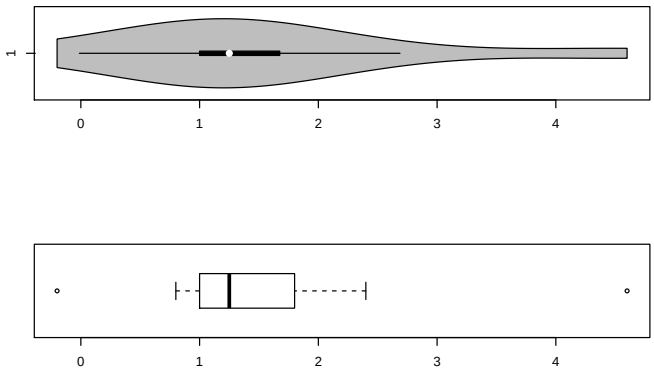
```
##
##  Cramer-von Mises normality test
##
## data:  sleep$d
## W = 0.13817, p-value = 0.02769
```

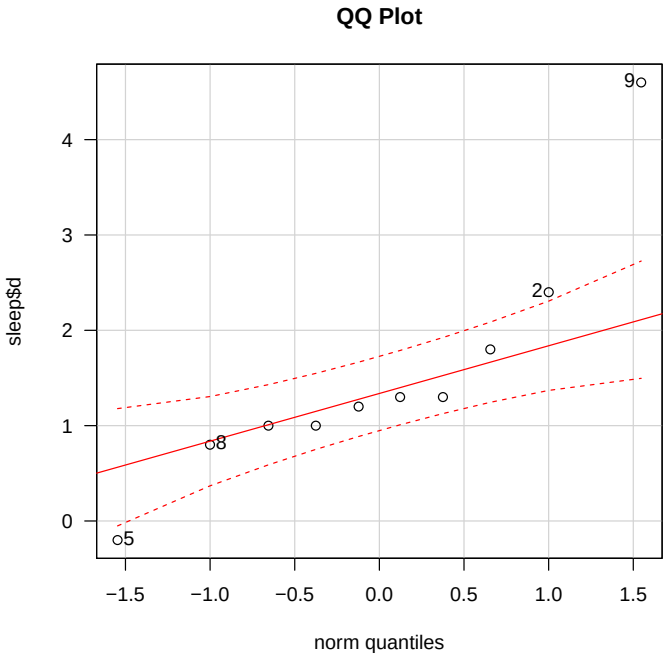
```
# plot of data
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(sleep$d, freq = FALSE, breaks = 20)
points(density(sleep$d), type = "l")
rug(sleep$d)

# violin plot
library(vioplot)
vioplot(sleep$d, horizontal=TRUE, col="gray")

# boxplot
boxplot(sleep$d, horizontal=TRUE)
```

```
# QQ plot
par(mfrow=c(1,1))
qqPlot(sleep$d, las = 1, id.n = 4, id.cex = 1, lwd = 1, main="QQ Plot")
```

```
##  9  5  2  8
## 10  1  9  2
```







Example: Androstenedione Levels This is an independent two-sample problem, so you must look at normal scores plots for males and females.

The AD test p-value and the SW test p-value for testing normality exceeds 0.10 in each sample. Thus, given the sample sizes (14 for men, 18 for women), we have insufficient evidence (at $\alpha = 0.05$) to reject normality in either population.

The women's boxplot contains two mild outliers, which is highly unusual when sampling from a normal distribution. The tests are possibly not powerful enough to pick up this type of deviation from normality in such a small sample. In practice, this may not be a big concern. The two mild outliers probably have a small effect on inferences in the sense that non-parametric methods would probably lead to similar conclusions here.

```
library(ggplot2)
p1 <- ggplot(andro, aes(x = sex, y = level, fill=sex))
p1 <- p1 + geom_boxplot()
```

Men

```
shapiro.test(men)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  men
## W = 0.90595, p-value = 0.1376
```

```
library(nortest)
ad.test(men)
```

```
##
##  Anderson-Darling normality test
##
## data:  men
## A = 0.4718, p-value = 0.2058
```

```
# lillie.test(men)
cvm.test(men)
```

```
##
##  Cramer-von Mises normality test
##
## data:  men
## W = 0.063063, p-value = 0.3221
```

Women

```
shapiro.test(women)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  women
## W = 0.95975, p-value = 0.5969
```

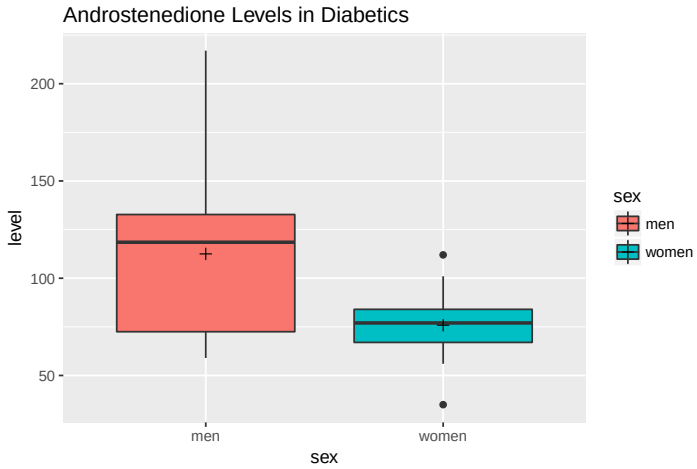
```
library(nortest)
ad.test(women)
```

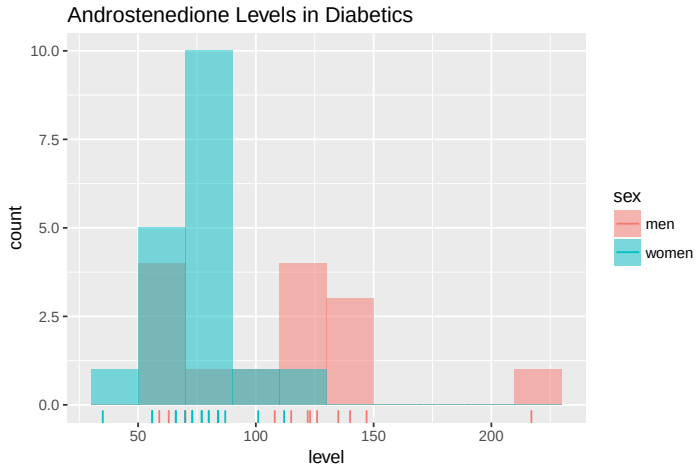
```
##
##  Anderson-Darling normality test
##
## data:  women
## A = 0.39468, p-value = 0.3364
```

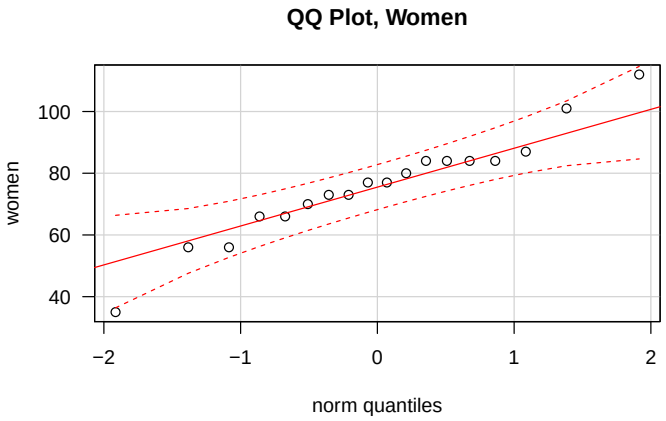
```
# lillie.test(women)
cvm.test(women)
```

```
##
##  Cramer-von Mises normality test
##
## data:  women
## W = 0.065242, p-value = 0.3057
```

```
# add a "+" at the mean
p1 <- p1 + stat_summary(fun.y = mean, geom = "point", shape = 3, size = 2)
#p1 <- p1 + coord_flip()
p1 <- p1 + labs(title = "Androstenedione Levels in Diabetics")
#print(p1)

p2 <- ggplot(andro, aes(x = level, fill=sex))
p2 <- p2 + geom_histogram(binwidth = 20, alpha = 0.5, position="identity")
p2 <- p2 + geom_rug(aes(colour=sex))
p2 <- p2 + labs(title = "Androstenedione Levels in Diabetics")
#print(p2)

library(gridExtra)
grid.arrange(grobs = list(p1, p2), ncol=1)


# QQ plot
par(mfrow=c(2,1))
qqPlot(men, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Men")
qqPlot(women, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Women")
```



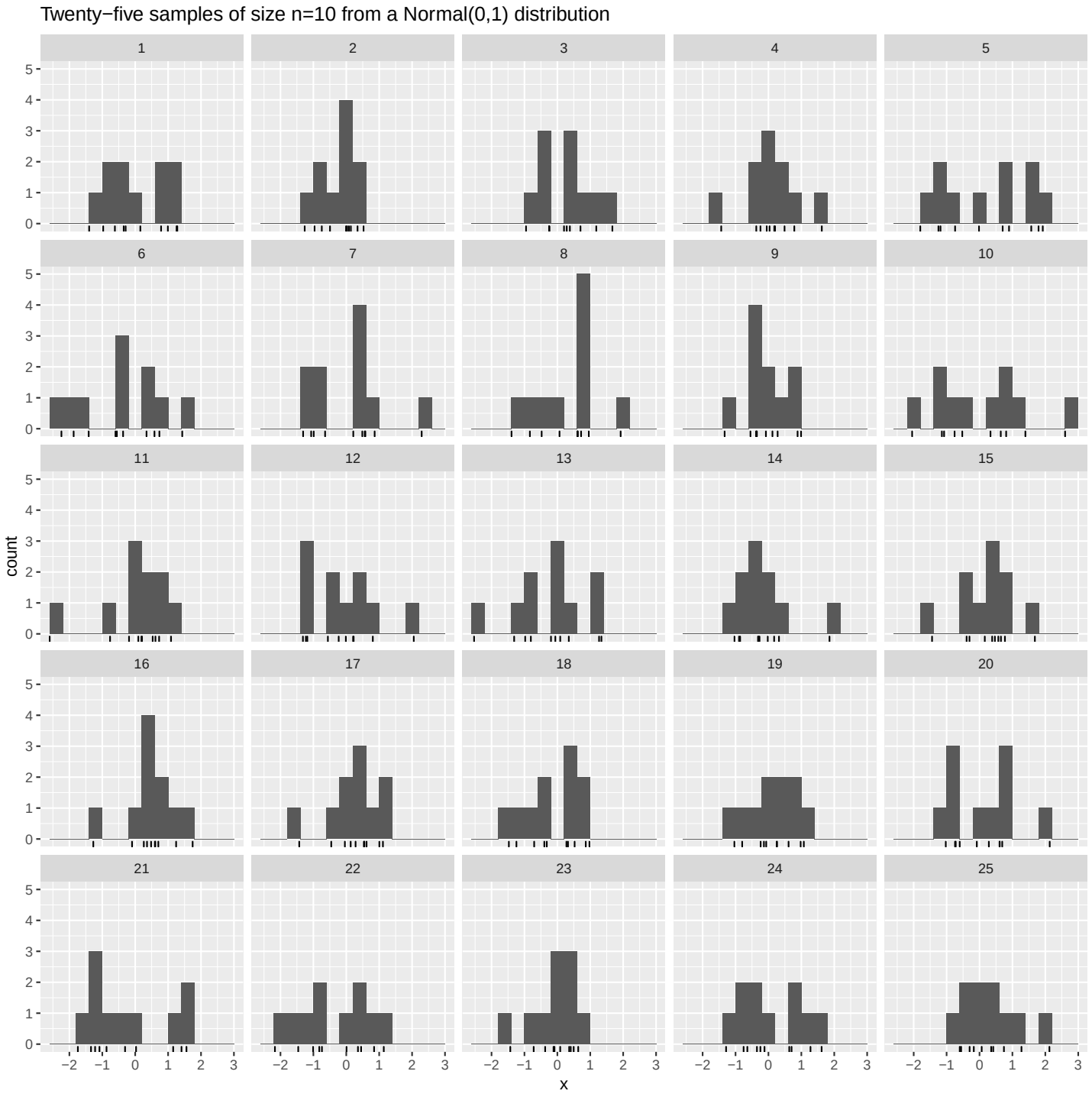




Most statisticians use graphical methods (boxplot, normal scores plot) to assess normality, and do not carry out formal tests.

You may be surprised at how variable 10 observations from a Normal(0,1) distribution looks like; here are 25 samples.

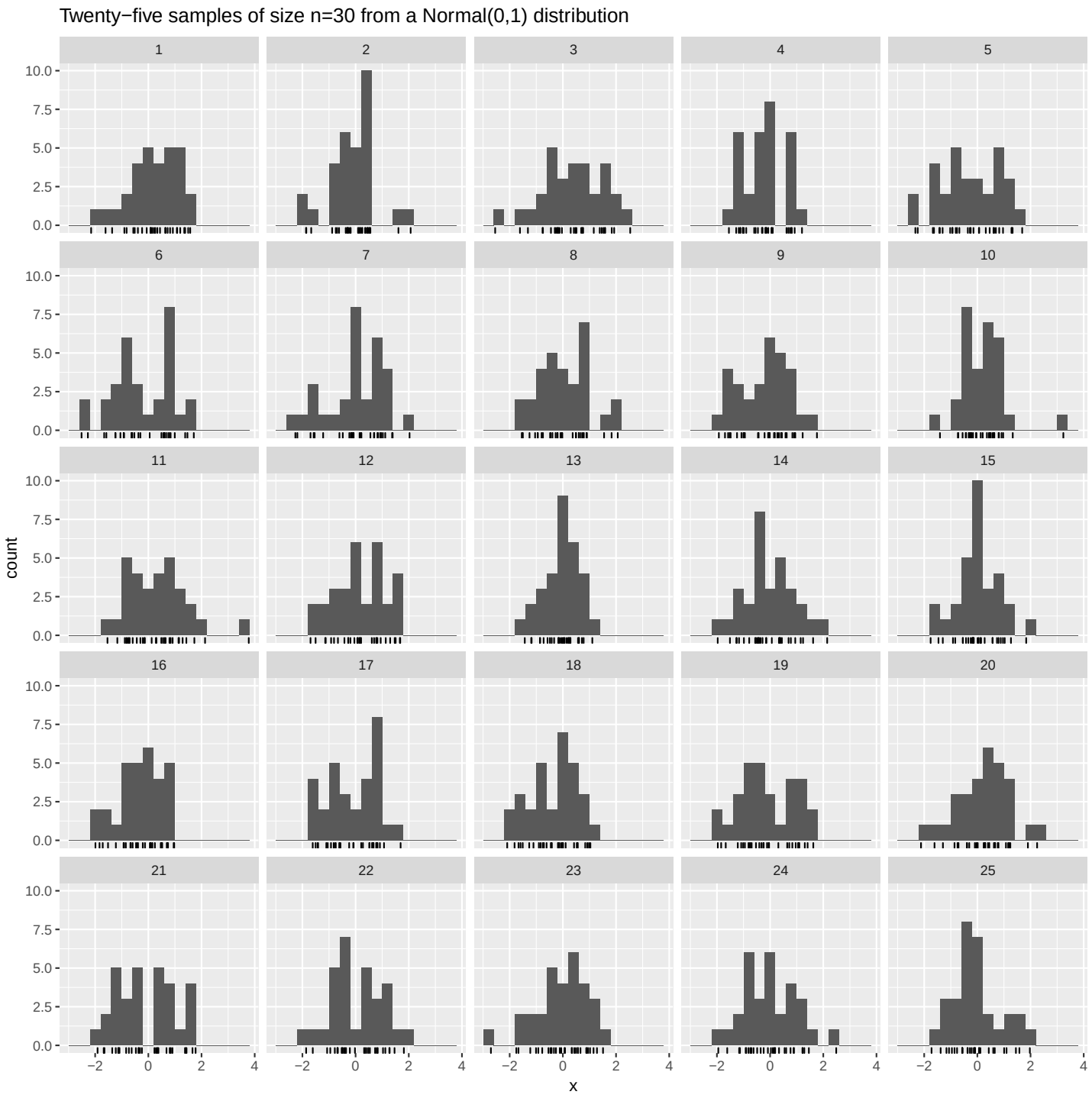
```
n = 10
r = 5
norm.many <- data.frame(id = rep(seq(1:r^2), n)
                        , x = rnorm(r^2 * n)
                         )
library(ggplot2)
p <- ggplot(norm.many, aes(x = x))
p <- p + geom_histogram(binwidth = 0.4)
p <- p + geom_rug()
p <- p + facet_wrap(~ id, ncol = r)
p <- p + labs(title = "Twenty-five samples of size n=10 from a Normal(0,1) distribution")
print(p)
```



. . . and here are samples of size $n = 30$.

```
n = 30
r = 5
norm.many <- data.frame(id = rep(seq(1:r^2), n)
                        , x = rnorm(r^2 * n)
                         )
library(ggplot2)
p <- ggplot(norm.many, aes(x = x))
p <- p + geom_histogram(binwidth = 0.4)
p <- p + geom_rug()
```

```
p <- p + facet_wrap(~ id, ncol = r)
p <- p + labs(title = "Twenty-five samples of size n=30 from a Normal(0,1) distribution")
print(p)
```



By viewing many versions of this of varying samples sizes you'll develop your intuition about what a normal sample looks like.

4.4 Testing Equal Population Variances

In the independent two sample t-test, some researchers test $H_0 : \sigma_1^2 = \sigma_2^2$ as a means to decide between using the pooled-variance procedure or Satterthwaite's methods. They suggest the pooled t-test and CI if H_0 is not rejected, and Satterthwaite's methods otherwise.

There are a number of well-known tests for equal population variances, of which Bartlett's test and Levene's test are probably the best known. Bartlett's test assumes the population distributions are normal, and is the best test when this is true. In practice, unequal variances and non-normality often go hand-in-hand, so you should check normality prior to using Bartlett's test. It is sensitive to data which is not non-normally distributed, thus it is more likely to return a "false positive" (reject H_0 of equal variances) when the data is non-normal. Levene's test is more robust to departures from normality than Bartlett's test; it is in the `car` package. Fligner-Killeen test is a non-parametric test which is very robust against departures from normality.

I will now define **Bartlett's test**, which assumes normally distributed data. As above, let $n^* = n_1 + n_2 + \cdots + n_k$, where the n_is are the sample sizes from the k groups, and define

$$v = 1 + \frac{1}{3(k-1)} \left(\sum_{i=1}^{k} \frac{1}{n_i - 1} - \frac{1}{n^* - k} \right).$$

Bartlett's statistic for testing $H_0 : \sigma_1^2 = \cdots = \sigma_k^2$ is given by

$$B_{obs} = \frac{2.303}{v} \left\{ (n-k) \log(s^2_{\text{pooled}}) - \sum_{i=1}^{k} (n_i - 1) \log(s_i^2) \right\},$$

where s^2_{pooled} is the pooled estimator of variance and s_i^2 is the estimated variance based on the i^{th} sample.

Large values of B_{obs} suggest that the population variances are unequal. For a size α test, we reject H_0 if $B_{\text{obs}} \geq \chi^2_{k-1,\text{crit}}$, where $\chi^2_{k-1,\text{crit}}$ is the upper-α percentile for the χ^2_{k-1} (chi-squared) probability distribution with $k-1$ degrees

of freedom. A generic plot of the χ^2 distribution is given below. A p-value for the test is given by the area under the chi-squared curve to the right of B_{obs}.

Example: Androstenedione Levels The sample standard deviations and samples sizes are: $s_1 = 42.8$ and $n_1 = 14$ for men and $s_2 = 17.2$ and $n_2 = 18$ for women. The sample standard deviations appear to be very different, so I would not be surprised if the test of equal population variances is highly significant. The output below confirms this: the p-values for Bartlett's F-test, Levene's Test, and Fligner-Killeen test are all much smaller than 0.05. An implication is that the standard pooled-CI and test on the population means is inappropriate.

```
#### Testing Equal Population Variances
# numerical summaries
c(mean(men), mean(women), sd(men), sd(women))
## [1] 112.50000  75.83333  42.75467  17.23625
c(IQR(men), IQR(women), length(men), length(women))
## [1] 60.25 17.00 14.00 18.00
## Test equal variance
# assumes populations are normal
bartlett.test(level ~ sex, data = andro)
##
##  Bartlett test of homogeneity of variances
##
## data:  level by sex
## Bartlett's K-squared = 11.199, df = 1, p-value = 0.0008183
# does not assume normality, requires car package
library(car)
leveneTest(level ~ sex, data = andro)
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value  Pr(>F)
## group  1  7.2015 0.01174 *
##       30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# nonparametric test
fligner.test(level ~ sex, data = andro)
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  level by sex
```

```
## Fligner-Killeen:med chi-squared = 5.8917, df = 1, p-value =
## 0.01521
```

4.5 Small sample sizes, a comment

In Daniel Kahneman's "Thinking, Fast and Slow" (Ch 10), he discusses "The Law of Small Numbers" (in contrast to the Law of Large Numbers). As an example from statisticians Howard Wainer and Harris Zwerling, he makes this observation about the incidence of kidney cancer in the 3,141 counties of the United States. "The counties in which the incidence of kidney cancer is *lowest* are mostly rural, sparsely populated, and located in traditionally Republican states in the Midwest, the South, and the West. What do you make of this?" The statistians comment: "It is both easy and tempting to infer that their low cancer rates are directly due to the clean living of the rural lifestyle — no air pollution, no water pollution, access to fresh food without additives." This makes perfect sense.

"Now consider the counties in which the incidence of kidney cancer is highest. These ailing counties tend to be mostly rural, sparsely populated, and located in traditionally Republican states in the Midwest, the South, and the West." Tongue-in-cheek, Wainer and Zwerling comment: "It is easy to infer that their high cancer rates might be directly due to the poverty of the rural lifestyle — no access to good medical care, a high-fat diet, and too much alcohol, too much tobacco." Something is wrong, of course. The rural lifestyle cannot explain both very high and very low incidence of kidney cancer.

The key factor is not that the counties were rural or predominantly Republican. It is that rural counties have small populations. The law of large numbers says that as sample sizes increase that the sample statistic converges to the population proportion, that is, large samples are more precise than small samples. What Kahneman is calling the law of small numbers warns that small samples yield extreme results more often than large samples do.

Chapter 5

One-Way Analysis of Variance

Contents


```
## Warning in file(filename, "r", encoding = encoding): cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding): cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

select graphical displays that meaningfully compare independent populations.

assess the assumptions of the ANOVA visually and by formal tests.

decide whether the means between populations are different, and how.

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

1. organize knowledge.

5. define parameters of interest and hypotheses in words and notation.

6. summarize data visually, numerically, and descriptively.

8. use statistical software.

12. make evidence-based decisions.

5.1 ANOVA

The one-way analysis of variance (**ANOVA**) is a generalization of the two sample t-test to $k \geq 2$ groups. Assume that the populations of interest have the following (unknown) population means and standard deviations:

	population 1	population 2	$\cdots$	population k
mean	μ_1	μ_2	$\cdots$	μ_k
std dev	σ_1	σ_2	$\cdots$	σ_k

A usual interest in ANOVA is whether $\mu_1 = \mu_2 = \cdots = \mu_k$. If not, then we wish to know which means differ, and by how much. To answer these questions we select samples from each of the k populations, leading to the following data summary:

	sample 1	sample 2	$\cdots$	sample k
size	n_1	n_2	$\cdots$	n_k
mean	$\bar{Y}_1$	$\bar{Y}_2$	$\cdots$	$\bar{Y}_k$
std dev	s_1	s_2	$\cdots$	s_k

A little more notation is needed for the discussion. Let Y_{ij} denote the j^{th} observation in the i^{th} sample and define the total sample size $n^* = n_1 + n_2 + \cdots + n_k$. Finally, let $\bar{\bar{Y}}$ be the average response over all samples (combined),

that is

$$\bar{\bar{Y}} = \frac{\sum_{ij} Y_{ij}}{n^*} = \frac{\sum_i n_i \bar{Y}_i}{n^*}.$$

Note that $\bar{\bar{Y}}$ is *not* the average of the sample means, unless the sample sizes n_i are equal.

An F-statistic is used to test $H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$ against $H_A :$ not H_0 (that is, at least two means are different). The assumptions needed for the standard ANOVA F-test are analogous to the independent pooled two-sample t-test assumptions: (1) Independent random samples from each population. (2) The population frequency curves are normal. (3) The populations have equal standard deviations, $\sigma_1 = \sigma_2 = \cdots = \sigma_k$.

The F-test is computed from the ANOVA table, which breaks the spread in the combined data set into two components, or **Sums of Squares** (SS). The **Within SS**, often called the **Residual SS** or the **Error SS**, is the portion of the total spread due to variability *within* samples:

$$\text{SS(Within)} = (n_1 - 1)s_1^2 + (n_2 - 1)s_2^2 + \cdots + (n_k - 1)s_k^2 = \sum_{ij}(Y_{ij} - \bar{Y}_i)^2.$$

The **Between SS**, often called the Model SS, measures the spread between the sample means

$$\text{SS(Between)} =$$
$$n_1(\bar{Y}_1 - \bar{\bar{Y}})^2 + n_2(\bar{Y}_2 - \bar{\bar{Y}})^2 + \cdots + n_k(\bar{Y}_k - \bar{\bar{Y}})^2 = \sum_i n_i(\bar{Y}_i - \bar{\bar{Y}})^2,$$

weighted by the sample sizes. These two SS add to give

$$\text{SS(Total)} = \text{SS(Between)} + \text{SS(Within)} = \sum_{ij}(Y_{ij} - \bar{\bar{Y}})^2.$$

Each SS has its own degrees of freedom (df). The df(Between) is the number of groups minus one, $k-1$. The df(Within) is the total number of observations minus the number of groups: $(n_1 - 1) + (n_2 - 1) + \cdots + (n_k - 1) = n^* - k$. These two df add to give df(Total) $= (k-1) + (n^* - k) = n^* - 1$.

The Sums of Squares and df are neatly arranged in a table, called the ANOVA table:

Source	df	SS	MS	F
Between Groups (Model)	$dfM = k - 1$	$SSM = \sum_i n_i(\bar{Y}_i - \bar{\bar{Y}})^2$	$MSM = SSM/dfM$	MSM/MSE
Within Groups (Error)	$dfE = n^* - k$	$SSE = \sum_i (n_i - 1)s_i^2$	$MSE = SSE/dfE$	
Total	$dfT = n^* - 1$	$SST = \sum_{ij}(Y_{ij} - \bar{\bar{Y}})^2$	$MST = SST/dfT$	

The Mean Square for each source of variation is the corresponding SS divided by its *df*. The Mean Squares can be easily interpreted.

The MS(Within)

$$\text{MS(Within)} = \frac{(n_1-1)s_1^2 + (n_2-1)s_2^2 + \cdots + (n_k-1)s_k^2}{n^* - k} = s^2_{\text{pooled}}$$

is a weighted average of the sample variances. The MS(Within) is known as the pooled estimator of variance, and estimates the assumed common population variance. If all the sample sizes are equal, the MS(Within) is the average sample variance. The MS(Within) is identical to the **pooled variance estimator** in a two-sample problem when $k = 2$.

The MS(Between)

$$\text{MS(Between)} = \frac{\sum_i n_i(\bar{Y}_i - \bar{\bar{Y}})^2}{k-1}$$

is a measure of variability among the sample means. This MS is a multiple of the sample variance of $\bar{Y}_1,\ \bar{Y}_2, \ldots, \bar{Y}_k$ when all the sample sizes are equal.

The MS(Total)

$$\text{MS(Total)} = \frac{\sum_{ij}(Y_{ij} - \bar{\bar{Y}})^2}{n^* - 1}$$

is the variance in the combined data set.

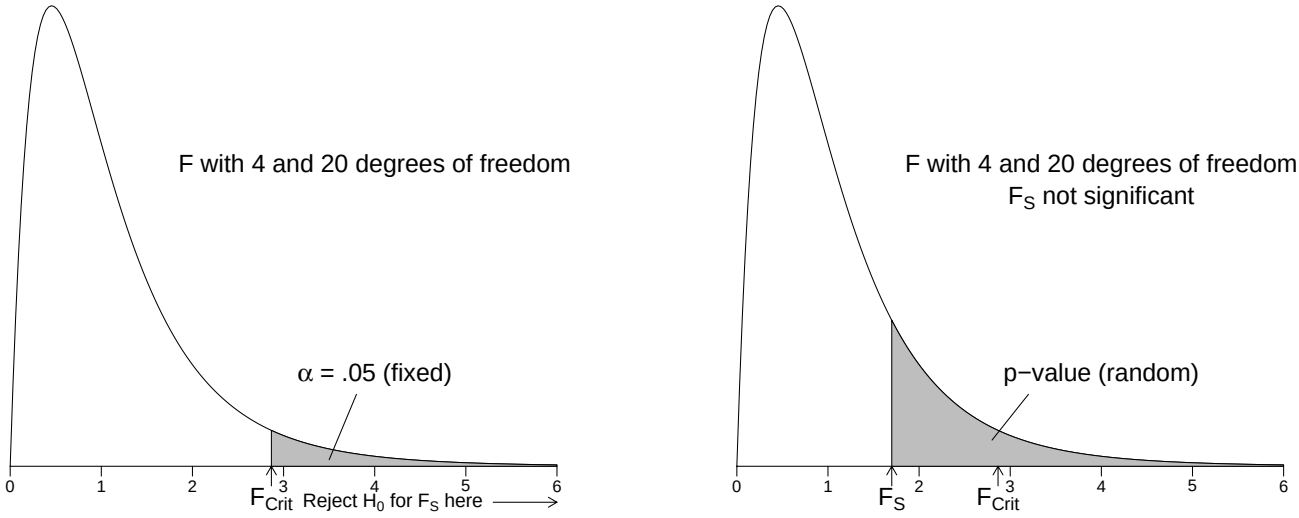
The decision on whether to reject $H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$ is based on the ratio of the MS(Between) and the MS(Within):

$$F_s = \frac{\text{MS(Between)}}{\text{MS(Within)}}.$$

Large values of F_s indicate large variability among the sample means $\bar{Y}_1,\ \bar{Y}_2, \ldots, \bar{Y}_k$ relative to the spread of the data within samples. That is, large values of F_s suggest that H_0 is false.

Formally, for a size α test, reject H_0 if $F_s \geq F_{crit}$, where F_{crit} is the upper-α percentile from an F distribution with numerator degrees of freedom $k-1$ and

denominator degrees of freedom $n^* - k$ (i.e., the *df* for the numerators and denominators in the F-ratio). The p-value for the test is the area under the F-probability curve to the right of F_s:



For $k = 2$ the ANOVA F-test is equivalent to the pooled two-sample t-test.

The specification of a one-way analysis of variance is analogous to a regression analysis (discussed later, though we've seen the specification in some plotting functions). The only difference is that the descriptive (x) variable needs to be a factor and not a numeric variable. We calculate a model object using `lm()` and extract the analysis of variance table with `anova()`.

Example: Comparison of Fats During cooking, doughnuts absorb fat in various amounts. A scientist wished to learn whether the amount absorbed depends on the type of fat. For each of 4 fats, 6 batches of 24 doughnuts were prepared. The data are grams of fat absorbed per batch.

Let

μ_i = pop mean grams of fat i absorbed per batch of 24 doughnuts (-100).

The scientist wishes to test $H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4$ against $H_A :$ not H_0. There is no strong evidence against normality here. Furthermore the sample

standard deviations (see output below) are close. The standard ANOVA appears to be appropriate here.

Row	fat1	fat2	fat3	fat4
1	164	178	175	155
2	172	191	186	166
3	168	197	178	149
4	177	182	171	164
5	190	185	163	170
6	176	177	176	168

Let's take a short detour to read the wide table and convert it into long format.

R skills: wide to long table format Many functions in R expect data to be in a long format rather than a wide format. Let's use the fat data as an example of how to read a table as text, convert the wide format to long, and then back to wide format.

```
#### Example: Comparison of Fats
fat <- read.table(text="
Row  fat1  fat2  fat3  fat4
  1   164   178   175   155
  2   172   191   186   166
  3   168   197   178   149
  4   177   182   171   164
  5   190   185   163   170
  6   176   177   176   168
", header=TRUE)
fat
```

```
##   Row fat1 fat2 fat3 fat4
## 1   1  164  178  175  155
## 2   2  172  191  186  166
## 3   3  168  197  178  149
## 4   4  177  182  171  164
## 5   5  190  185  163  170
## 6   6  176  177  176  168
```

From wide to long: Use `melt()` from the `reshape2` package.

```
#### From wide to long format
library(reshape2)
fat.long <- melt(fat,
              # id.vars: ID variables
              #   all variables to keep but not split apart on
              id.vars=c("Row"),
              # measure.vars: The source columns
              #   (if unspecified then all other variables are measure.vars)
              measure.vars = c("fat1", "fat2", "fat3", "fat4"),
```

```
              # variable.name: Name of the destination column identifying each
              #   original column that the measurement came from
              variable.name = "type",
              # value.name: column name for values in table
              value.name = "amount"
            )
## naming variables manually, the variable.name and value.name not working 11/2012
#names(fat.long) <- c("Row", "type", "amount")
fat.long
```

```
##    Row type amount
## 1    1 fat1    164
## 2    2 fat1    172
## 3    3 fat1    168
## 4    4 fat1    177
## 5    5 fat1    190
## 6    6 fat1    176
## 7    1 fat2    178
## 8    2 fat2    191
## 9    3 fat2    197
## 10   4 fat2    182
## 11   5 fat2    185
## 12   6 fat2    177
## 13   1 fat3    175
## 14   2 fat3    186
## 15   3 fat3    178
## 16   4 fat3    171
## 17   5 fat3    163
## 18   6 fat3    176
## 19   1 fat4    155
## 20   2 fat4    166
## 21   3 fat4    149
## 22   4 fat4    164
## 23   5 fat4    170
## 24   6 fat4    168
```

```
# or as simple as:
# melt(fat, "Row")
```

If you don't specify variable.name, it will name that column "variable", and if you leave out value.name, it will name that column "value".

From long to wide: Use `dcast()` from the `reshape2` package.

```
#### From long to wide format
fat.wide <- dcast(fat.long, Row ~ type, value.var = "amount")
fat.wide
```

```
##   Row fat1 fat2 fat3 fat4
## 1   1  164  178  175  155
## 2   2  172  191  186  166
```

```
## 3   3  168  197  178  149
## 4   4  177  182  171  164
## 5   5  190  185  163  170
## 6   6  176  177  176  168
```

Now that we've got our data in the long format, let's return to the ANOVA.

Back to ANOVA: Let's look at the numerical summaries. We've seen other ways of computing these so I'll show you another way.

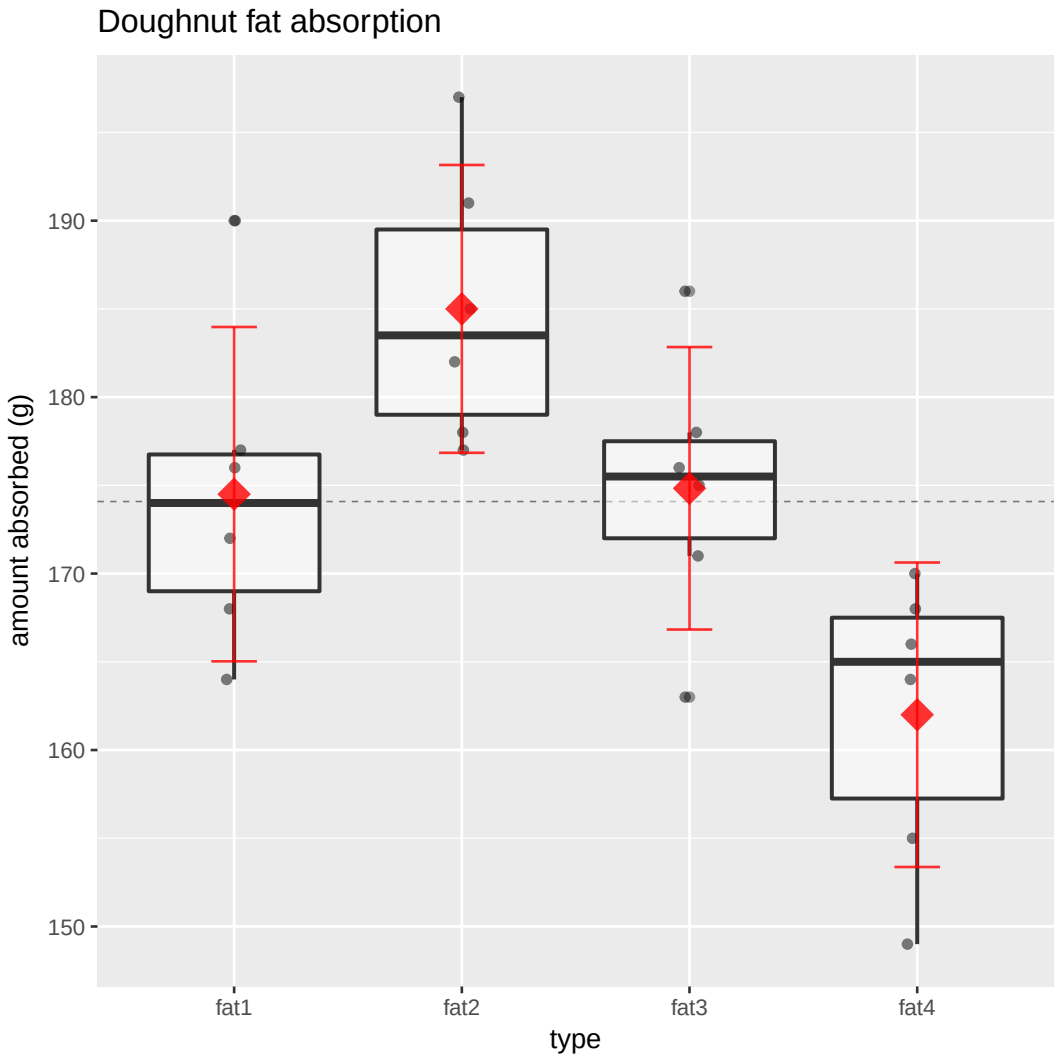
```
#### Back to ANOVA
# Calculate the mean, sd, n, and se for the four fats

# The plyr package is an advanced way to apply a function to subsets of data
#   "Tools for splitting, applying and combining data"
library(plyr)
# ddply "dd" means the input and output are both data.frames
fat.summary <- ddply(fat.long,
                     "type",
                     function(X) {
                       data.frame( m = mean(X$amount),
                                   s = sd(X$amount),
                                   n = length(X$amount)
                                 )
                     }
                    )
# standard errors
fat.summary$se   <- fat.summary$s/sqrt(fat.summary$n)
# individual confidence limits
fat.summary$ci.l <- fat.summary$m - qt(1-.05/2, df=fat.summary$n-1) * fat.summary$se
fat.summary$ci.u <- fat.summary$m + qt(1-.05/2, df=fat.summary$n-1) * fat.summary$se
fat.summary
```

```
##   type        m        s n       se     ci.l     ci.u
## 1 fat1 174.5000 9.027735 6 3.685557 165.0260 183.9740
## 2 fat2 185.0000 7.771744 6 3.172801 176.8441 193.1559
## 3 fat3 174.8333 7.626707 6 3.113590 166.8296 182.8371
## 4 fat4 162.0000 8.221922 6 3.356586 153.3716 170.6284
```

Let's plot the data with boxplots, individual points, mean, and CI by fat type.

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(fat.long, aes(x = type, y = amount))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(yintercept = mean(fat.long$amount),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
```

```
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Doughnut fat absorption") + ylab("amount absorbed (g)")
print(p)
```



The p-value for the F-test is 0.001. The scientist would reject H_0 at any of the usual test levels (such as, 0.05 or 0.01). The data suggest that the population mean absorption rates differ across fats *in some way*. The F-test does not say *how* they differ. The pooled standard deviation $s_{\text{pooled}} = 8.18$ is the "Residual standard error". We'll ignore the rest of this output for now.

```
fit.f <- aov(amount ~ type, data = fat.long)
summary(fit.f)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
```

```
## type         3   1596   531.8   7.948 0.0011 **
## Residuals   20   1338    66.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fit.f
```

```
## Call:
##    aov(formula = amount ~ type, data = fat.long)
##
## Terms:
##                      type Residuals
## Sum of Squares  1595.500  1338.333
## Deg. of Freedom        3        20
##
## Residual standard error: 8.180261
## Estimated effects may be unbalanced
```


■ CLICKERQs — ANOVA, Fat 1/2 ■

5.2 Multiple Comparison Methods: Fisher's Method

The ANOVA F-test checks whether all the population means are equal. **Multiple comparisons** are often used as a follow-up to a significant ANOVA F-test to determine which population means are different. I will discuss Fisher's, Bonferroni's, and Tukey's methods for comparing all pairs of means.

Fisher's least significant difference method **(LSD or FSD)** is a two-step process:

1. Carry out the ANOVA F-test of $H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$ at the α level. If H_0 is not rejected, stop and conclude that there is insufficient evidence to claim differences among population means. If H_0 is rejected, go to step

2.

2. Compare each pair of means using a pooled two sample t-test at the α level. Use s_{pooled} from the ANOVA table and $df = dfE$ (Residual).

To see where the name LSD originated, consider the t-test of $H_0 : \mu_i = \mu_j$ (i.e., populations i and j have same mean). The t-statistic is

$$t_s = \frac{\bar{Y}_i - \bar{Y}_j}{s_{\text{pooled}}\sqrt{\frac{1}{n_i} + \frac{1}{n_j}}}.$$

You reject H_0 if $|t_s| \geq t_{\text{crit}}$, or equivalently, if

$$|\bar{Y}_i - \bar{Y}_j| \geq t_{\text{crit}} s_{\text{pooled}} \sqrt{\frac{1}{n_i} + \frac{1}{n_j}}.$$

The minimum absolute difference between $\bar{Y}_i$ and $\bar{Y}_j$ needed to reject H_0 is the LSD, the quantity on the right hand side of this inequality. If all the sample sizes are equal $n_1 = n_2 = \cdots = n_k$ then the LSD is the same for each comparison:

$$LSD = t_{\text{crit}} s_{\text{pooled}} \sqrt{\frac{2}{n_1}},$$

where n_1 is the common sample size.

I will illustrate Fisher's method on the doughnut data, using $\alpha = 0.05$. At the first step, you reject the hypothesis that the population mean absorptions are equal because p-value= 0.001. At the second step, compare all pairs of fats at the 5% level. Here, $s_{\text{pooled}} = 8.18$ and $t_{\text{crit}} = 2.086$ for a two-sided test based on 20 df (the dfE for Residual SS). Each sample has six observations, so the LSD for each comparison is

$$LSD = 2.086 \times 8.18 \times \sqrt{\frac{2}{6}} = 9.85.$$

Any two sample means that differ by at least 9.85 in magnitude are **significantly different** at the 5% level.

An easy way to compare all pairs of fats is to order the samples by their sample means. The samples can then be grouped easily, noting that two fats are in the same group if the absolute difference between their sample means is smaller than the LSD.

Fats	Sample Mean
2	185.00
3	174.83
1	174.50
4	162.00

There are six comparisons of two fats. From this table, you can visually assess which sample means differ by at least the LSD=9.85, and which ones do not. For completeness, the table below summarizes each comparison:

Comparison	Absolute difference in means	Exceeds LSD?
Fats 2 and 3	10.17	Yes
2 and 1	10.50	Yes
2 and 4	23.00	Yes
Fats 3 and 1	0.33	No
3 and 4	12.83	Yes
Fats 1 and 4	12.50	Yes

The end product of the multiple comparisons is usually presented as a collection of **groups**, where a group is defined to be a set of populations with sample means that are not significantly different from each other. Overlap among groups is common, and occurs when one or more populations appears in two or more groups. Any overlap requires a more careful interpretation of the analysis.

There are three groups for the doughnut data, with no overlap. Fat 2 is in a group by itself, and so is Fat 4. Fats 3 and 1 are in a group together. This information can be summarized by ordering the samples from lowest to highest average, and then connecting the fats in the same group using an underscore:

```
FAT 4    FAT 1    FAT 3     FAT 2
-----    ----------------   -----
```

The results of a multiple comparisons must be interpreted carefully. At the 5% level, you have sufficient evidence to conclude that the population mean

absorption for Fat 2 and Fat 4 are each different than the other population means. However, there is insufficient evidence to conclude that the population mean absorptions for Fats 1 and 3 differ.

Be Careful with Interpreting Groups in Multiple Comparisons!

To see why you must be careful when interpreting groupings, suppose you obtain two groups in a three sample problem. One group has samples 1 and 3. The other group has samples 3 and 2:

```
  1     3     2
-----------
      -----------
```

This occurs, for example, when $|\bar{Y}_1 - \bar{Y}_2| \geq LSD$, but both $|\bar{Y}_1 - \bar{Y}_3|$ and $|\bar{Y}_3 - \bar{Y}_2|$ are less than the LSD. There is a tendency to conclude, and please try to avoid this line of attack, that populations 1 and 3 have the same mean, populations 2 and 3 have the same mean, but populations 1 and 2 have different means. This conclusion is illogical. The groupings imply that we have sufficient evidence to conclude that population means 1 and 2 are different, but insufficient evidence to conclude that population mean 3 differs from either of the other population means.

5.2.1 FSD Multiple Comparisons in R

One way to get Fisher comparisons in R uses `pairwise.t.test()` with `p.adjust.method` The resulting summary of the multiple comparisons is in terms of p-values for all pairwise two-sample t-tests using the pooled standard deviation from the ANOVA using `pool.sd = TRUE`. This output can be used to generate groupings. A summary of the p-values is given below. Let us see that we can recover the groups from this output.

```
#### Multiple Comparisons
# all pairwise comparisons among levels of fat
# Fisher's LSD (FSD) uses "none"
pairwise.t.test(fat.long$amount, fat.long$type,
                pool.sd = TRUE, p.adjust.method = "none")

##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  fat.long$amount and fat.long$type
##
```

```
##      fat1  fat2    fat3
## fat2 0.038 -       -
## fat3 0.944 0.044   -
## fat4 0.015 9.3e-05 0.013
##
## P value adjustment method: none
```

Discussion of the FSD Method: family error rate

There are $c = k(k-1)/2$ pairs of means to compare in the second step of the FSD method. Each comparison is done at the α level, where for a generic comparison of the i^{th} and j^{th} populations

$$\alpha = \text{probability of rejecting } H_0 : \mu_i = \mu_j \text{ when } H_0 \text{ is true.}$$

The individual error rate is not the only error rate that is important in multiple comparisons. The **family error rate** (FER), or the **experimentwise error rate**, is defined to be *the probability of at least one false rejection of a true hypothesis* $H_0 : \mu_i = \mu_j$ *over all comparisons.* When many comparisons are made, you *may* have a large probability of making one or more false rejections of true null hypotheses. In particular, when all c comparisons of two population means are performed, each at the α level, then

$$\alpha < FER < c\alpha.$$

For example, in the doughnut problem where $k = 4$, there are $c = 4(3)/2 = 6$ possible comparisons of pairs of fats. If each comparison is carried out at the 5% level, then $0.05 < FER < 0.30$. At the second step of the FSD method, you could have up to a 30% chance of claiming one or more pairs of population means are different if no differences existed between population means. Most statistical packages do not evaluate the FER, so the upper bound is used.

The first step of the FSD method is the ANOVA "screening" test. The multiple comparisons are carried out only if the F-test suggests that not all population means are equal. This screening test tends to deflate the FER for the two-step FSD procedure. However, the FSD method is commonly criticized for being extremely liberal (too many false rejections of true null hypotheses)

when some, but not many, differences exist — especially when the number of comparisons is large. This conclusion is fairly intuitive. When you do a large number of tests, each, say, at the 5% level, then sampling variation alone will suggest differences in 5% of the comparisons where the H_0 is true. The number of false rejections could be enormous with a large number of comparisons. For example, chance variation alone would account for an average of 50 significant differences in 1000 comparisons (about 45 populations) each at the 5% level.

5.2.2 Bonferroni Comparisons

The Bonferroni method controls the FER by reducing the individual comparison error rate. The FER is guaranteed to be no larger than a prespecified amount, say α, by setting the individual error rate for each of the c comparisons of interest to α/c. Therefore, $\alpha/c < FER < c\alpha/c = \alpha$, thus the upper bound for FER is α. Larger differences in the sample means are needed before declaring statistical significance using the Bonferroni adjustment than when using the FSD method at the α level.

■ ClickerQs — ANOVA, Bonferroni ■

Assuming all comparisons are of interest, you can implement the Bonferroni adjustment in R by specifying `p.adjust.method = "bonf"` A by-product of the Bonferroni adjustment is that we have at least $100(1-\alpha)\%$ confidence that all pairwise t-test statements hold simultaneously!

```
# Bonferroni 95% Individual p-values
# All Pairwise Comparisons among Levels of fat
pairwise.t.test(fat.long$amount, fat.long$type,
                pool.sd = TRUE, p.adjust.method = "bonf")

##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  fat.long$amount and fat.long$type
##
##      fat1    fat2    fat3
## fat2 0.22733 -       -
```

```
## fat3 1.00000 0.26241 -
## fat4 0.09286 0.00056 0.07960
##
## P value adjustment method: bonferroni
```

Looking at the output, can you create the groups? You should get the groups given below, which implies you have sufficient evidence to conclude that the population mean absorption for Fat 2 is different than that for Fat 4.

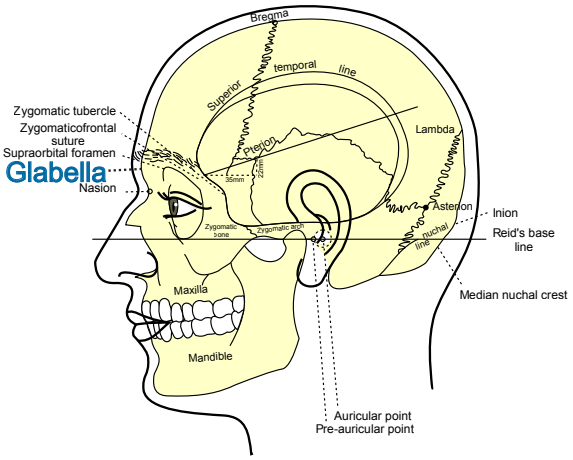
```
FAT 4    FAT 1    FAT 3    FAT 2
----------------------
         ----------------------
```

The Bonferroni method tends to produce "coarser" groups than the FSD method, because the individual comparisons are conducted at a lower (alpha/error) level. Equivalently, the minimum significant difference is inflated for the Bonferroni method. For example, in the doughnut problem with $FER \leq 0.05$, the critical value for the individual comparisons at the 0.0083 level is $t_{\text{crit}} = 2.929$ with $df = 20$. The minimum significant difference for the Bonferroni comparisons is

$$LSD = 2.929 \times 8.18 \times \sqrt{\frac{2}{6}} = 13.824$$

versus an LSD=9.85 for the FSD method. Referring back to our table of sample means on page 172, we see that the sole comparison where the absolute difference between sample means exceeds 13.824 involves Fats 2 and 4.

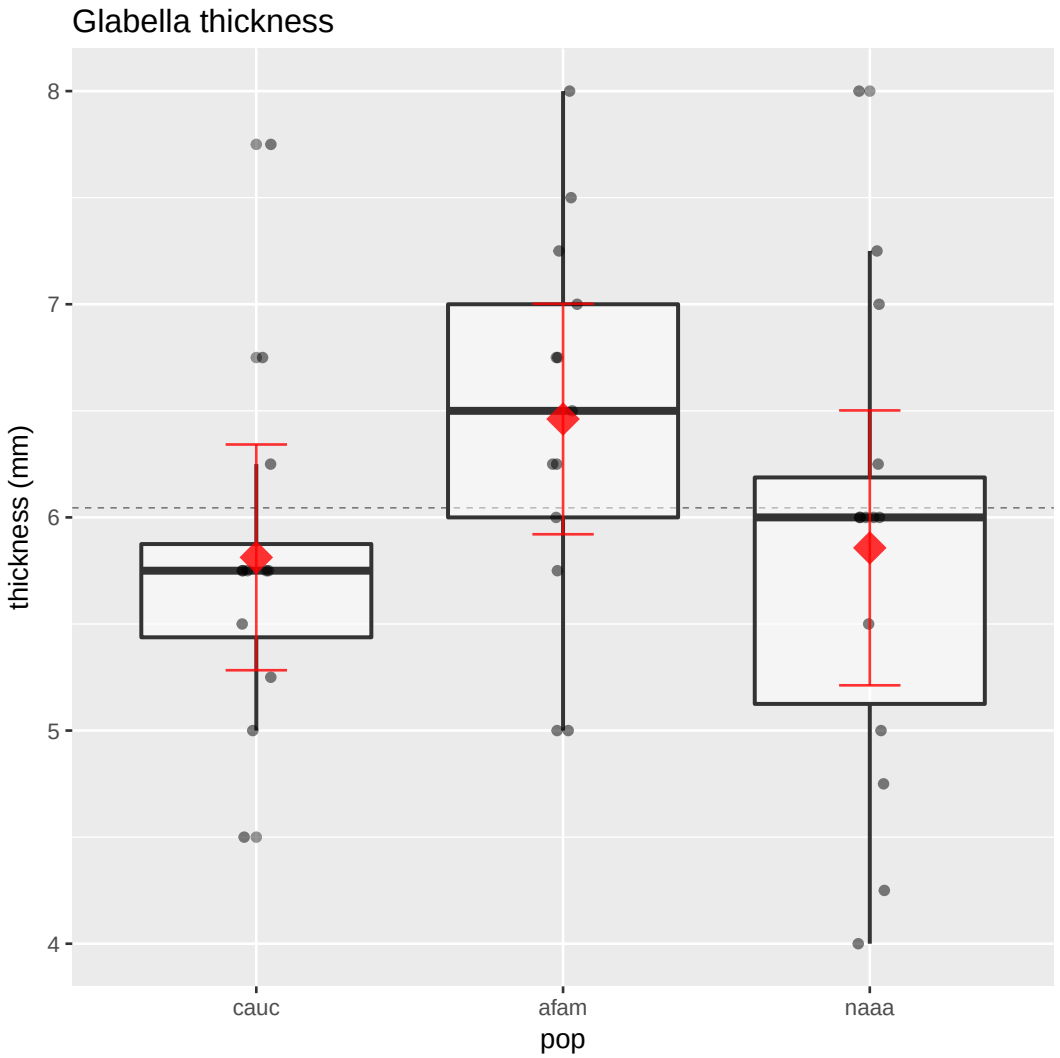
Example from Koopmans: glabella facial tissue thickness In an anthropological study of facial tissue thickness for different racial groups, data were taken during autopsy at several points on the faces of deceased individuals. The Glabella measurements taken at the bony ridge for samples of individuals from three racial groups (cauc = Caucasian, afam = African American, and naaa = Native American and Asian) follow. The data values are in mm.




```
#### Example from Koopmans: glabella facial tissue thickness
glabella <- read.table(text="
Row  cauc  afam  naaa
  1  5.75  6.00  8.00
  2  5.50  6.25  7.00
  3  6.75  6.75  6.00
  4  5.75  7.00  6.25
  5  5.00  7.25  5.50
  6  5.75  6.75  4.00
  7  5.75  8.00  5.00
  8  7.75  6.50  6.00
  9  5.75  7.50  7.25
 10  5.25  6.25  6.00
 11  4.50  5.00  6.00
 12  6.25  5.75  4.25
 13    NA  5.00  4.75
 14    NA    NA  6.00
", header=TRUE)

glabella.long <- melt(glabella,
              id.vars=c("Row"),
              variable.name = "pop",
              value.name = "thickness",
              # remove NAs
              na.rm = TRUE
            )
# naming variables manually, the variable.name and value.name not working 11/2012
names(glabella.long) <- c("Row", "pop", "thickness")
# another way to remove NAs:
#glabella.long <- subset(glabella.long, !is.na(thickness))
```

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(glabella.long, aes(x = pop, y = thickness))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(yintercept = mean(glabella.long$thickness),
```

```
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Glabella thickness") + ylab("thickness (mm)")
print(p)
```



There are 3 groups, so there are 3 possible pairwise comparisons. If you want a Bonferroni analysis with FER of no greater than 0.05, you should do the individual comparisons at the $0.05/3 = 0.0167$ level. Except for the mild outlier in the Caucasian sample, the observed distributions are fairly symmetric,

with similar spreads. I would expect the standard ANOVA to perform well here.

Let μ_c = population mean Glabella measurement for Caucasians, μ_a = population mean Glabella measurement for African Americans, and μ_n = population mean Glabella measurement for Native Americans and Asians.

```
glabella.summary <- ddply(glabella.long, "pop",
     function(X) { data.frame( m = mean(X$thickness),
                               s = sd(X$thickness),
                               n = length(X$thickness) ) } )
glabella.summary
```

```
##    pop        m         s  n
## 1 cauc 5.812500 0.8334280 12
## 2 afam 6.461538 0.8946959 13
## 3 naaa 5.857143 1.1168047 14
```

```
fit.g <- aov(thickness ~ pop, data = glabella.long)
summary(fit.g)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## pop          2   3.40  1.6991   1.828  0.175
## Residuals   36  33.46  0.9295
```

```
fit.g
```

```
## Call:
##    aov(formula = thickness ~ pop, data = glabella.long)
##
## Terms:
##                       pop Residuals
## Sum of Squares    3.39829  33.46068
## Deg. of Freedom         2        36
##
## Residual standard error: 0.9640868
## Estimated effects may be unbalanced
```

At the 5% level, you would not reject the hypothesis that the population mean Glabella measurements are identical. That is, you do not have sufficient evidence to conclude that these racial groups differ with respect to their average Glabella measurement. **This is the end of the analysis!**

The Bonferroni intervals reinforce this conclusion, all the p-values are greater than 0.05. If you were to calculate CIs for the difference in population means, each would contain zero. You can think of the Bonferroni intervals as simultaneous CI. We're (at least) 95% confident that all of the following statements hold simultaneously: $-1.62 \leq \mu_c - \mu_a \leq 0.32$, $-0.91 \leq \mu_n - \mu_c \leq 1.00$, and $-1.54 \leq \mu_n - \mu_a \leq 0.33$. The individual CIs have level $100(1 - 0.0167)\% =$

98.33%.

```
# Bonferroni 95% Individual p-values
# All Pairwise Comparisons among Levels of glabella
pairwise.t.test(glabella.long$thickness, glabella.long$pop,
                pool.sd = TRUE, p.adjust.method = "bonf")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  glabella.long$thickness and glabella.long$pop
##
##      cauc afam
## afam 0.30 -
## naaa 1.00 0.34
##
## P value adjustment method: bonferroni
```

5.3 Further Discussion of Multiple Comparisons

The FSD and Bonferroni methods comprise the ends of the spectrum of multiple comparisons methods. Among multiple comparisons procedures, the FSD method is most likely to find differences, whether real or due to sampling variation, whereas Bonferroni is often the most conservative method. You can be reasonably sure that differences suggested by the Bonferroni method will be suggested by almost all other methods, whereas differences not significant under FSD will not be picked up using other approaches.

The Bonferroni method is conservative, but tends to work well when the number of comparisons is small, say 4 or less. A smart way to use the Bonferroni adjustment is to focus attention only on the comparisons of interest (generated independently of looking at the data!), and ignore the rest. I will return to this point later.

A commonly-used alternative is **Tukey's** honest significant difference method (HSD). John Tukey's honest significant difference method is to reject the equality of a pair of means based, not on the t-distribution, but the studentized range distribution. To implement Tukey's method with a FER of α, reject

$H_0 : \mu_i = \mu_j$ when

$$|\bar{Y}_i - \bar{Y}_j| \geq \frac{q_{crit}}{\sqrt{2}} s_{\text{pooled}} \sqrt{\frac{1}{n_i} + \frac{1}{n_j}},$$

where q_{crit} is the α level critical value of the studentized range distribution. For the doughnut fats, the groupings based on Tukey and Bonferroni comparisons are identical.

```
#### Tukey's honest significant difference method (HSD)
## Fat
# Tukey 95% Individual p-values
# All Pairwise Comparisons among Levels of fat
TukeyHSD(fit.f)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = amount ~ type, data = fat.long)
##
## $type
##                   diff        lwr        upr     p adj
## fat2-fat1  10.5000000  -2.719028 23.7190277 0.1510591
## fat3-fat1   0.3333333 -12.885694 13.5523611 0.9998693
## fat4-fat1 -12.5000000 -25.719028  0.7190277 0.0679493
## fat3-fat2 -10.1666667 -23.385694  3.0523611 0.1709831
## fat4-fat2 -23.0000000 -36.219028 -9.7809723 0.0004978
## fat4-fat3 -12.8333333 -26.052361  0.3856944 0.0590077
```

```
## Glabella
# Tukey 95% Individual p-values
# All Pairwise Comparisons among Levels of pop
TukeyHSD(fit.g)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = thickness ~ pop, data = glabella.long)
##
## $pop
##                   diff        lwr       upr     p adj
## afam-cauc  0.64903846 -0.2943223 1.5923993 0.2259806
## naaa-cauc  0.04464286 -0.8824050 0.9716907 0.9923923
## naaa-afam -0.60439560 -1.5120412 0.3032500 0.2472838
```

Another popular method controls the **false discovery rate** (FDR) instead of the FER. The FDR is the expected proportion of false discoveries amongst the rejected hypotheses. The false discovery rate is a less stringent condition

than the family-wise error rate, so these methods are more powerful than the others, though with a higher FER. I encourage you to learn more about the methods by Benjamini, Hochberg, and Yekutieli.

```
#### false discovery rate (FDR)
## Fat
# FDR
pairwise.t.test(fat.long$amount, fat.long$type,
                pool.sd = TRUE, p.adjust.method = "BH")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  fat.long$amount and fat.long$type
##
##      fat1    fat2    fat3
## fat2 0.05248 -       -
## fat3 0.94443 0.05248 -
## fat4 0.03095 0.00056 0.03095
##
## P value adjustment method: BH
```

```
## Glabella
# FDR
pairwise.t.test(glabella.long$thickness, glabella.long$pop,
                pool.sd = TRUE, p.adjust.method = "BH")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  glabella.long$thickness and glabella.long$pop
##
##      cauc afam
## afam 0.17 -
## naaa 0.91 0.17
##
## P value adjustment method: BH
```

■ ClickerQs — ANOVA, multiple comparisons ■

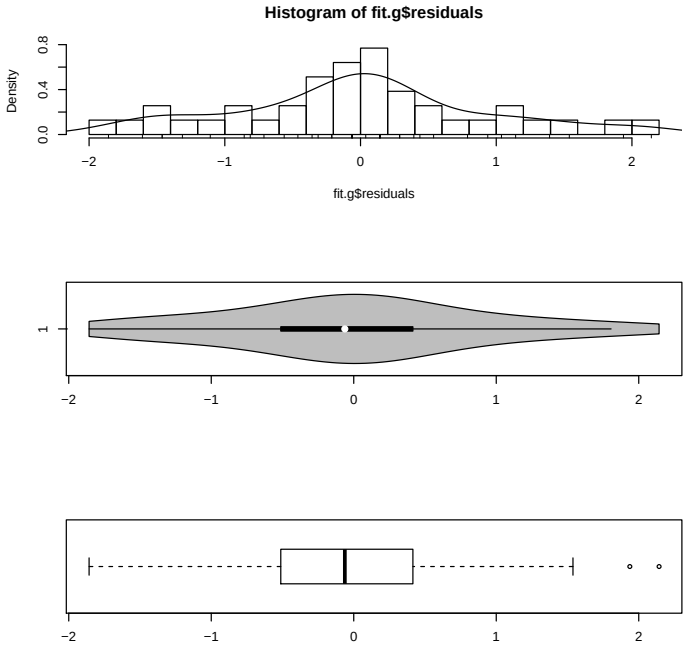
5.4 Checking Assumptions in ANOVA Problems

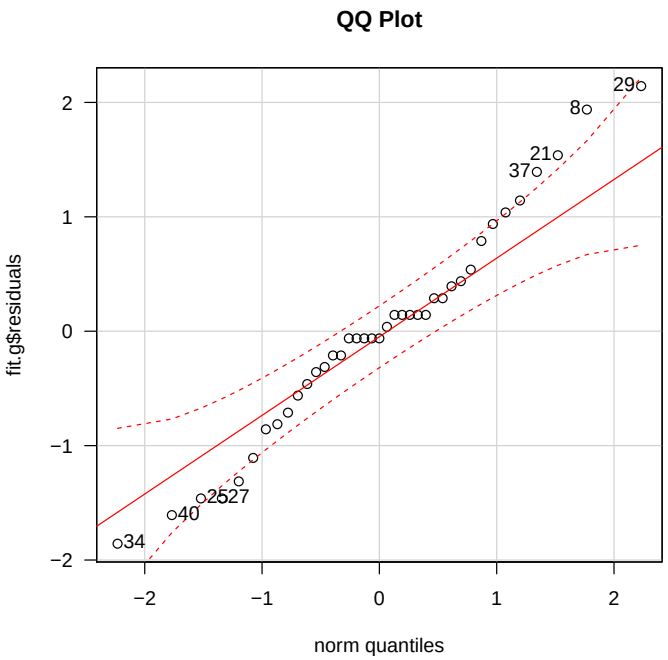
The classical ANOVA assumes that the populations have normal frequency curves and the populations have equal variances (or spreads). You can test the normality assumption using multiple normal QQ-plots and normal scores tests, which we discussed in Chapter 4. An alternative approach that is useful with three or more samples is to make a single normal scores plot for the entire data set. The samples must be *centered* at the same location for this to be meaningful. (WHY?) This is done by subtracting the sample mean from each observation in the sample, giving the so-called **residuals**. A normal scores plot or histogram of the residuals should resemble a sample from a normal population. These two plots can be generated with output in `$residuals` from the `anova()` procedure.

For the glabella residuals, there are a few observations outside the confidence bands, but the formal normality tests each have p-values > 0.2, so there's weak but unconvincing evidence of nonnormality.

```
#### Checking Assumptions in ANOVA Problems
# plot of data
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(fit.g$residuals, freq = FALSE, breaks = 20)
points(density(fit.g$residuals), type = "l")
rug(fit.g$residuals)

# violin plot
library(vioplot)
vioplot(fit.g$residuals, horizontal=TRUE, col="gray")

# boxplot
boxplot(fit.g$residuals, horizontal=TRUE)


# QQ plot
par(mfrow=c(1,1))
library(car)
qqPlot(fit.g$residuals, las = 1, id.n = 8, id.cex = 1, lwd = 1, main="QQ Plot")
```

```
## 29  8 34 40 21 25 27 37
## 39 38  1  2 37  3  4 36
```






```
shapiro.test(fit.g$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  fit.g$residuals
## W = 0.97693, p-value = 0.5927
```

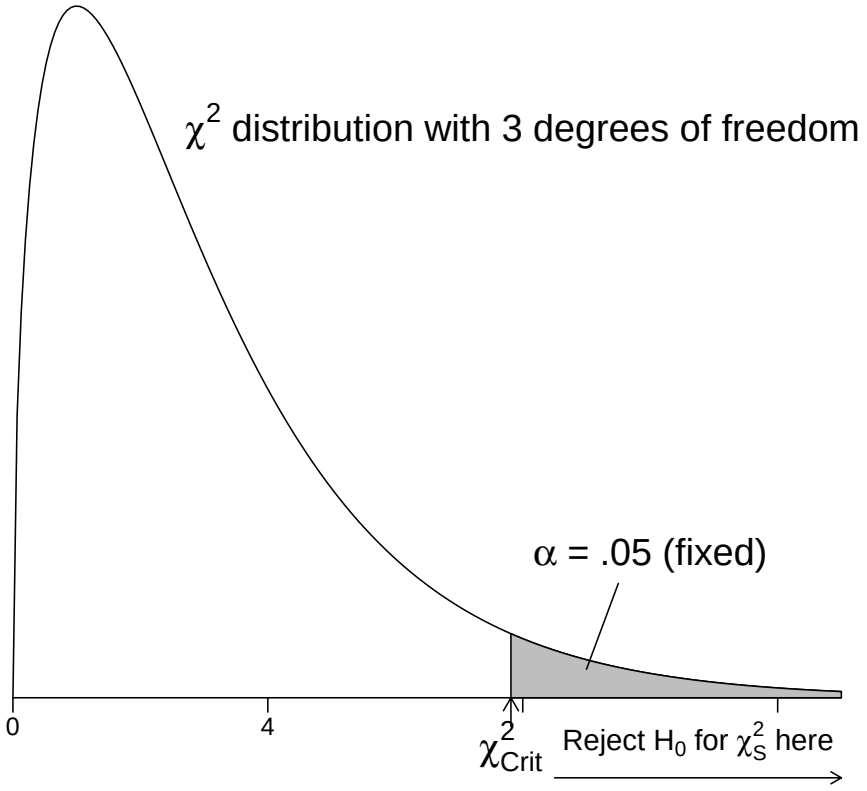
```
library(nortest)
ad.test(fit.g$residuals)
```

```
##
##  Anderson-Darling normality test
##
## data:  fit.g$residuals
## A = 0.37731, p-value = 0.3926
```

```
# lillie.test(fit.g£residuals)
cvm.test(fit.g$residuals)
```

```
##
##  Cramer-von Mises normality test
##
## data:  fit.g$residuals
## W = 0.070918, p-value = 0.2648
```

In Chapter 4, I illustrated the use of Bartlett's test and Levene's test for equal population variances, and showed how to evaluate these tests in R.



R does the calculation for us, as illustrated below. Because the p-value > 0.5, we fail to reject the null hypothesis that the population variances are equal. This result is not surprising given how close the sample variances are to each other.

```
## Test equal variance
# Barlett assumes populations are normal
bartlett.test(thickness ~ pop, data = glabella.long)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  thickness by pop
## Bartlett's K-squared = 1.1314, df = 2, p-value = 0.568
```

Levene's and Flinger's tests are consistent with Bartlett's.

```
# Levene does not assume normality, requires car package
library(car)
leveneTest(thickness ~ pop, data = glabella.long)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
```

```
## group  2  0.5286 0.5939
##       36
```

```
# Fligner is a nonparametric test
fligner.test(thickness ~ pop, data = glabella.long)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  thickness by pop
## Fligner-Killeen:med chi-squared = 1.0311, df = 2, p-value =
## 0.5972
```

5.5 Example from the Child Health and Development Study (CHDS)

We consider data from the birth records of 680 live-born white male infants. The infants were born to mothers who reported for pre-natal care to three clinics of the Kaiser hospitals in northern California. As an initial analysis, we will examine whether maternal smoking has an effect on the birth weights of these children. To answer this question, we define 3 groups based on mother's smoking history: (1) mother does not currently smoke or never smoked, (2) mother smoked less than one pack of cigarettes a day during pregnancy, and (3) mother smoked at least one pack of cigarettes a day during pregnancy.

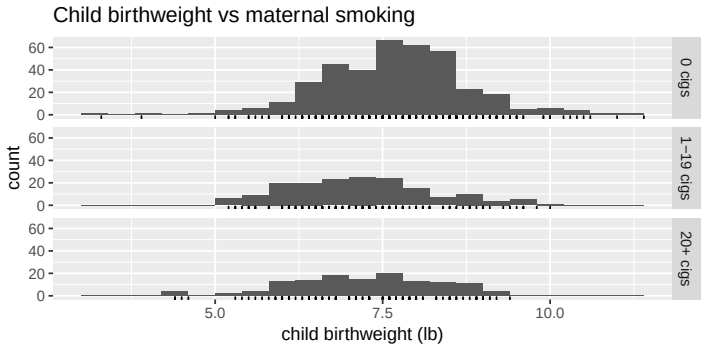
Let μ_i = pop mean birth weight (lb) for children in group i, $(i = 1, 2, 3)$. We wish to test $H_0 : \mu_1 = \mu_2 = \mu_3$ against H_A : not H_0.

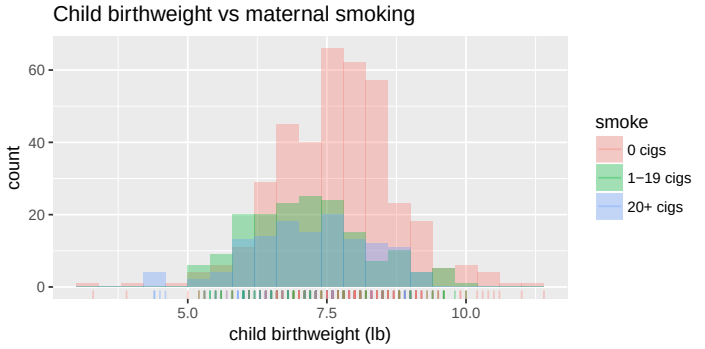
We read in the data, create a `smoke` factor variable, and plot the data by smoking group.

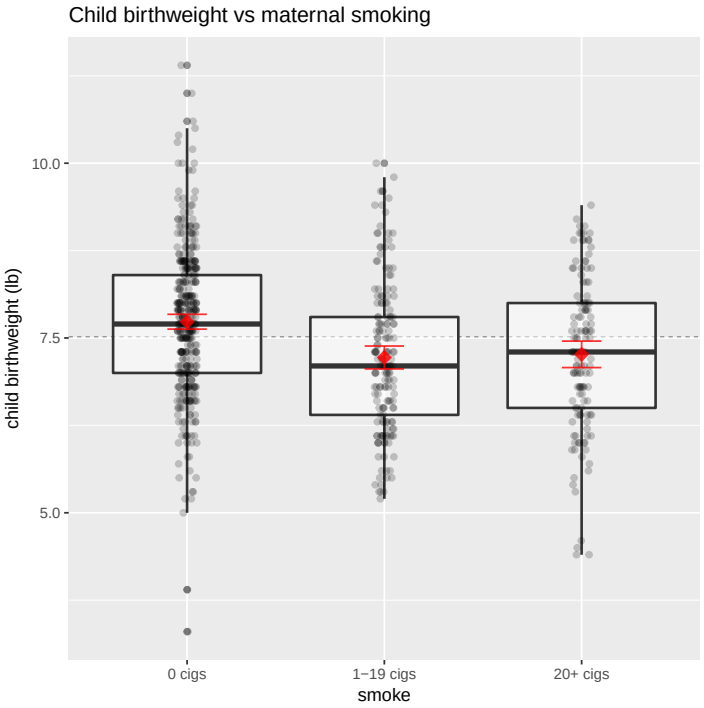
```
#### Example from the Child Health and Development Study (CHDS)
# description at http://statacumen.com/teach/ADA1/ADA1_notes_05-CHDS_desc.txt
# read data from website
chds <- read.csv("http://statacumen.com/teach/ADA1/ADA1_notes_05-CHDS.csv")

chds$smoke <- rep(NA, nrow(chds));
# no cigs
chds[(chds$m_smok == 0), "smoke"] <- "0 cigs";
# less than 1 pack (20 cigs = 1 pack)
chds[(chds$m_smok > 0) & (chds$m_smok < 20),"smoke"] <- "1-19 cigs";
# at least 1 pack (20 cigs = 1 pack)
chds[(chds$m_smok >= 20),"smoke"] <- "20+ cigs";
```

```
chds$smoke <- factor(chds$smoke)
```

```
# histogram using ggplot
p1 <- ggplot(chds, aes(x = c_bwt))
p1 <- p1 + geom_histogram(binwidth = .4)
p1 <- p1 + geom_rug()
p1 <- p1 + facet_grid(smoke ~ .)
p1 <- p1 + labs(title = "Child birthweight vs maternal smoking") +
          xlab("child birthweight (lb)")
#print(p1)

p2 <- ggplot(chds, aes(x = c_bwt, fill=smoke))
p2 <- p2 + geom_histogram(binwidth = .4, alpha = 1/3, position="identity")
p2 <- p2 + geom_rug(aes(colour = smoke), alpha = 1/3)
p2 <- p2 + labs(title = "Child birthweight vs maternal smoking") +
          xlab("child birthweight (lb)")
#print(p2)

library(gridExtra)
grid.arrange(grobs = list(p1, p2), ncol=1)


# Plot the data using ggplot
library(ggplot2)
p <- ggplot(chds, aes(x = smoke, y = c_bwt))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(yintercept = mean(chds$c_bwt),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.2)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 4,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Child birthweight vs maternal smoking") +
          ylab("child birthweight (lb)")
print(p)
```

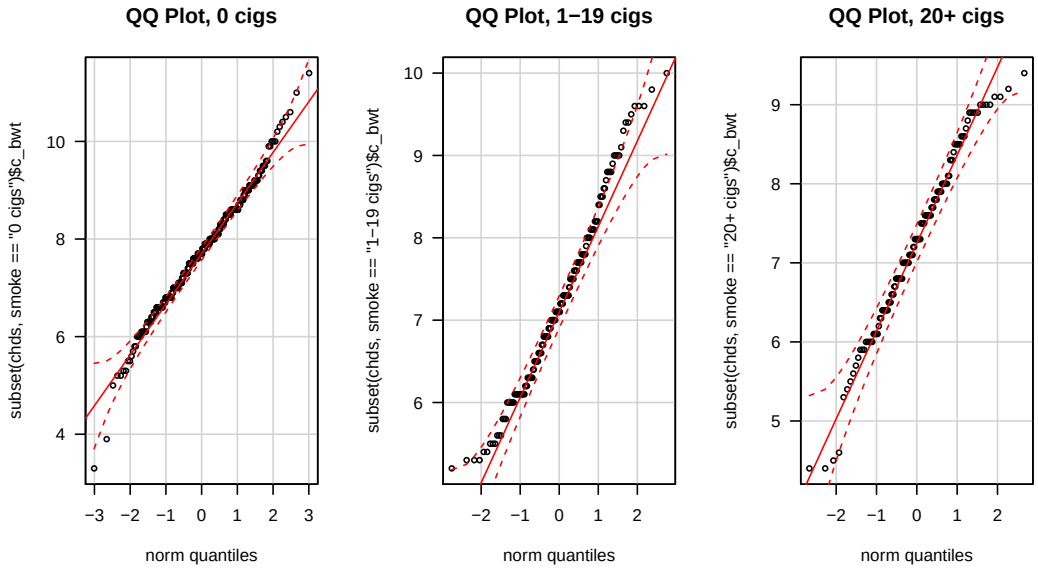






Looking at the boxplots, there is some evidence of non-normality here. Although there are outliers in the no smoking group, we need to recognize that the sample size for this group is fairly large (381). Given that boxplots are calibrated in such a way that 7 outliers per 1000 observations are expected when sampling from a normal population, 5 outliers (only 4 are visible!) out of 381 seems a bit excessive. A formal test rejects the hypothesis of normality in the no and low smoker groups. The normal probability plot and the histogram of the residuals also suggests that the population distributions are heavy tailed.

```
library(car)
par(mfrow=c(1,3))
qqPlot(subset(chds, smoke == "0 cigs"   )$c_bwt, las = 1, id.n = 0,
  id.cex = 1, lwd = 1, main="QQ Plot, 0 cigs")
qqPlot(subset(chds, smoke == "1-19 cigs")$c_bwt, las = 1, id.n = 0,
  id.cex = 1, lwd = 1, main="QQ Plot, 1-19 cigs")
qqPlot(subset(chds, smoke == "20+ cigs" )$c_bwt, las = 1, id.n = 0,
  id.cex = 1, lwd = 1, main="QQ Plot, 20+ cigs")
```




```
library(nortest)
# 0 cigs    --------------------
shapiro.test(subset(chds, smoke == "0 cigs"   )$c_bwt)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  subset(chds, smoke == "0 cigs")$c_bwt
## W = 0.98724, p-value = 0.00199
```

```
ad.test(     subset(chds, smoke == "0 cigs"   )$c_bwt)
```

```
##
##  Anderson-Darling normality test
##
## data:  subset(chds, smoke == "0 cigs")$c_bwt
## A = 0.92825, p-value = 0.01831
```

```
cvm.test(    subset(chds, smoke == "0 cigs"   )$c_bwt)
```

```
##
##  Cramer-von Mises normality test
##
## data:  subset(chds, smoke == "0 cigs")$c_bwt
## W = 0.13844, p-value = 0.03374
```

```
# 1-19 cigs --------------------
shapiro.test(subset(chds, smoke == "1-19 cigs")$c_bwt)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  subset(chds, smoke == "1-19 cigs")$c_bwt
## W = 0.97847, p-value = 0.009926
```

```
ad.test(     subset(chds, smoke == "1-19 cigs")$c_bwt)
```

```
##
```

```
##  Anderson-Darling normality test
##
## data:  subset(chds, smoke == "1-19 cigs")$c_bwt
## A = 0.83085, p-value = 0.03149
```

```
cvm.test(    subset(chds, smoke == "1-19 cigs")$c_bwt)
```

```
##
##  Cramer-von Mises normality test
##
## data:  subset(chds, smoke == "1-19 cigs")$c_bwt
## W = 0.11332, p-value = 0.07317
```

```
# 20+ cigs  --------------------
shapiro.test(subset(chds, smoke == "20+ cigs" )$c_bwt)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  subset(chds, smoke == "20+ cigs")$c_bwt
## W = 0.98127, p-value = 0.06962
```

```
ad.test(     subset(chds, smoke == "20+ cigs" )$c_bwt)
```

```
##
##  Anderson-Darling normality test
##
## data:  subset(chds, smoke == "20+ cigs")$c_bwt
## A = 0.40008, p-value = 0.3578
```

```
cvm.test(    subset(chds, smoke == "20+ cigs" )$c_bwt)
```

```
##
##  Cramer-von Mises normality test
##
## data:  subset(chds, smoke == "20+ cigs")$c_bwt
## W = 0.040522, p-value = 0.6694
```

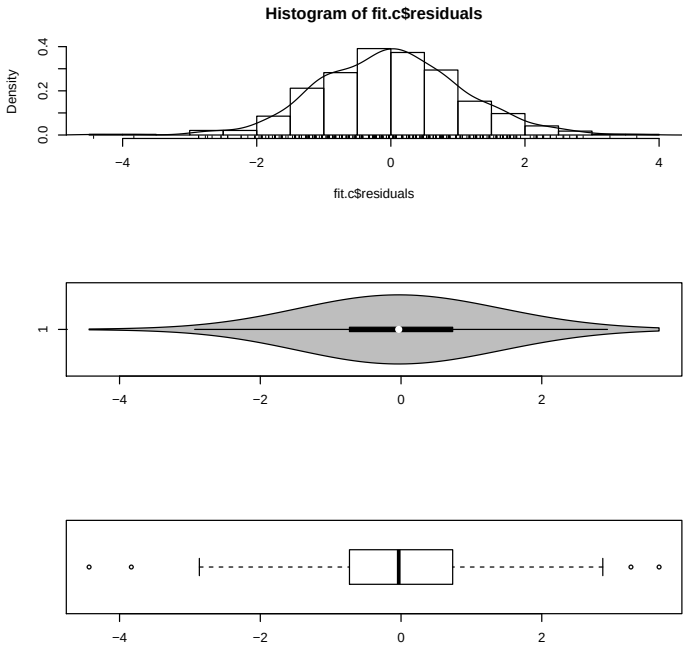
Fit the ANOVA (we'll look at the table below).

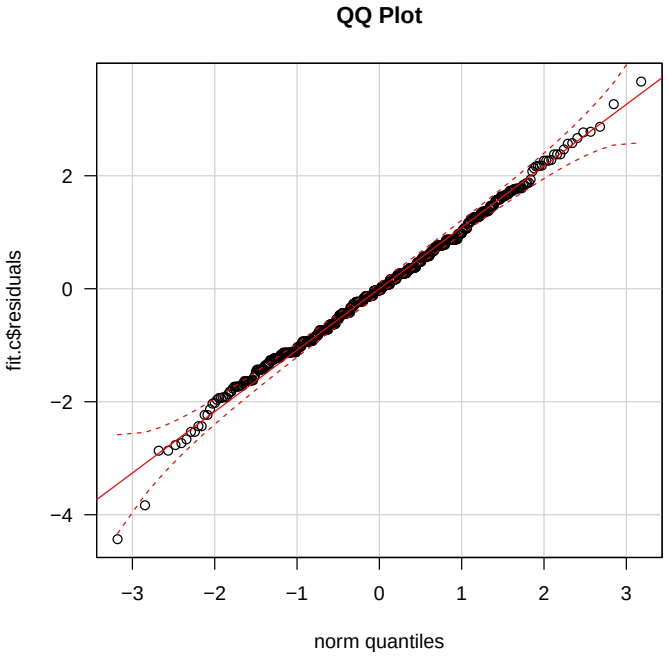
```
fit.c <- aov(c_bwt ~ smoke, data = chds)
```

A formal test of normality on the residuals of the combined sample is marginally significant (SW p-value= 0.047, others > 0.10). However, I am not overly concerned about this for the following reasons: in large samples, small deviations from normality are often statistically significant and in my experience, the small deviations we are seeing here are not likely to impact our conclusions, in the sense that non-parametric methods that do not require normality will lead to the same conclusions.

```
# plot of data
par(mfrow=c(3,1))
```

```r
# Histogram overlaid with kernel density curve
hist(fit.c$residuals, freq = FALSE, breaks = 20)
points(density(fit.c$residuals), type = "l")
rug(fit.c$residuals)

# violin plot
library(vioplot)
vioplot(fit.c$residuals, horizontal=TRUE, col="gray")

# boxplot
boxplot(fit.c$residuals, horizontal=TRUE)


# QQ plot
par(mfrow=c(1,1))
library(car)
qqPlot(fit.c$residuals, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot")
```






```r
shapiro.test(fit.c$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  fit.c$residuals
## W = 0.99553, p-value = 0.04758
```

```r
library(nortest)
ad.test(fit.c$residuals)
```

```
##
##  Anderson-Darling normality test
##
```

```
## data:  fit.c$residuals
## A = 0.62184, p-value = 0.1051
```

```
cvm.test(fit.c$residuals)
```

```
##
##  Cramer-von Mises normality test
##
## data:  fit.c$residuals
## W = 0.091963, p-value = 0.1449
```

Looking at the summaries, we see that the sample standard deviations are close. Formal tests of equal population variances are far from significant. The p-values for Bartlett's test and Levene's test are greater than 0.4. Thus, the standard ANOVA appears to be appropriate here.

```
# calculate summaries
chds.summary <- ddply(chds, "smoke",
     function(X) { data.frame( m = mean(X$c_bwt),
                               s = sd(X$c_bwt),
                               n = length(X$c_bwt) ) } )
chds.summary
```

```
##       smoke        m        s   n
## 1    0 cigs 7.732808 1.052341 381
## 2 1-19 cigs 7.221302 1.077760 169
## 3  20+ cigs 7.266154 1.090946 130
```

```
## Test equal variance
# assumes populations are normal
bartlett.test(c_bwt ~ smoke, data = chds)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  c_bwt by smoke
## Bartlett's K-squared = 0.3055, df = 2, p-value = 0.8583
```

```
# does not assume normality, requires car package
library(car)
leveneTest(c_bwt ~ smoke, data = chds)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   2  0.7591 0.4685
##       677
```

```
# nonparametric test
fligner.test(c_bwt ~ smoke, data = chds)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  c_bwt by smoke
## Fligner-Killeen:med chi-squared = 2.0927, df = 2, p-value =
## 0.3512
```

The p-value for the F-test is less than 0.0001. We would reject H_0 at any of the usual test levels (such as 0.05 or 0.01). The data suggest that the population mean birth weights differ across smoking status groups.

```
summary(fit.c)
```

```
##              Df Sum Sq Mean Sq F value   Pr(>F)
## smoke         2   40.7  20.351    17.9 2.65e-08 ***
## Residuals   677  769.5   1.137
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fit.c
```

```
## Call:
##    aov(formula = c_bwt ~ smoke, data = chds)
##
## Terms:
##                     smoke Residuals
## Sum of Squares   40.7012  769.4943
## Deg. of Freedom        2       677
##
## Residual standard error: 1.066126
## Estimated effects may be unbalanced
```

The Tukey multiple comparisons suggest that the mean birth weights are different (higher) for children born to mothers that did not smoke during pregnancy.

```
## CHDS
# Tukey 95% Individual p-values
TukeyHSD(fit.c)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = c_bwt ~ smoke, data = chds)
##
## $smoke
##                           diff        lwr        upr     p adj
## 1-19 cigs-0 cigs   -0.51150662 -0.7429495 -0.2800637 0.0000008
## 20+ cigs-0 cigs    -0.46665455 -0.7210121 -0.2122970 0.0000558
## 20+ cigs-1-19 cigs  0.04485207 -0.2472865  0.3369907 0.9308357
```

Part III

Nonparametric, categorical, and regression methods

Chapter 6

Nonparametric Methods

Contents


```
## Warning in file(filename, "r", encoding = encoding): cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding): cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

select the appropriate procedure based on assumptions.

explain reason for using one procedure over another.

decide whether the medians between multiple populations are different.

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

3. select correct statistical procedure.

5. define parameters of interest and hypotheses in words and notation.

6. summarize data visually, numerically, and descriptively.

8. use statistical software.

10. identify and explain statistical methods, assumptions, and limitations.

12. make evidence-based decisions.

6.1 Introduction

Nonparametric methods do not require the normality assumption of classical techniques. When the normality assumption is met, the ANOVA and t-test are most powerful, in that if the alternative is true these methods will make the correct decision with highest probability. However, if the normality assumption is not met, results from the ANOVA and t-test can be misleading and too liberal. I will describe and illustrate selected **non-parametric methods**, and compare them with classical methods. Some motivation and discussion of the strengths and weaknesses of non-parametric methods is given.

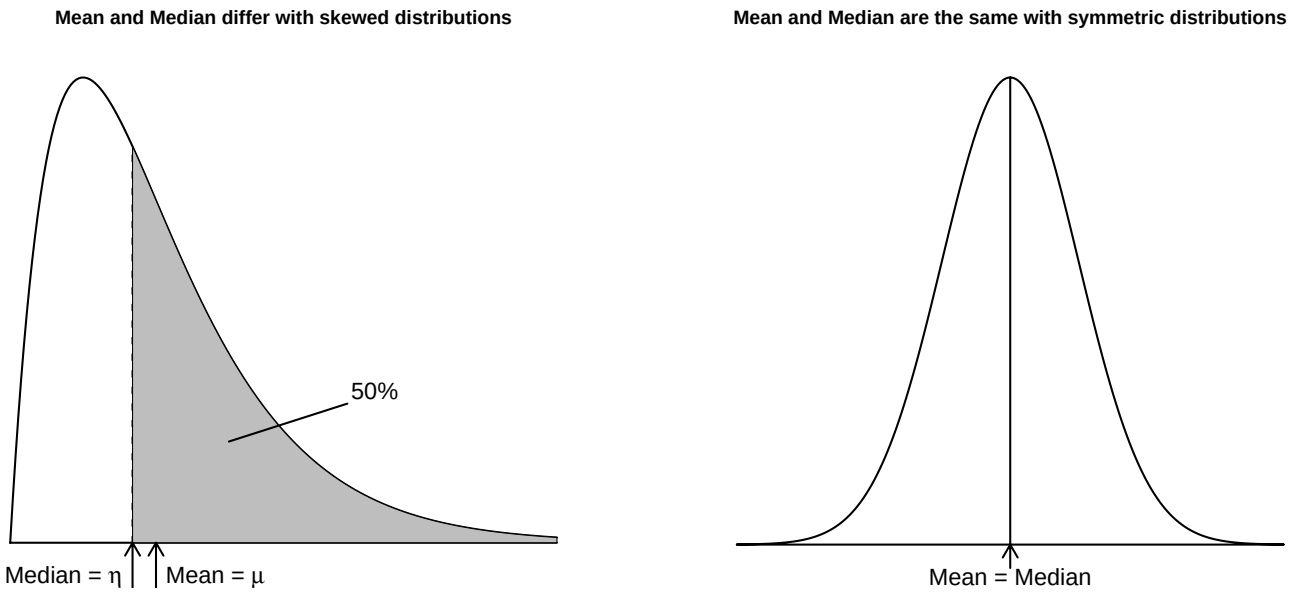
6.2 The Sign Test and CI for a Population Median

The **sign test** assumes that you have a random sample from a population, but makes **no assumption** about the population shape. The standard t-test

provides inferences on a population mean. The sign test, in contrast, provides inferences about a **population median**.

If the population frequency curve is symmetric (see below), then the population median, identified by η, and the population mean μ are identical. In this case the sign procedures provide inferences for the population mean, though less powerfully than the t-test.

The idea behind the sign test is straightforward. Suppose you have a sample of size m from the population, and you wish to test $H_0 : \eta = \eta_0$ (a given value). Let S be the number of sampled observations *above* η_0. If H_0 is true, you expect S to be approximately one-half the sample size, $0.5m$. If S is much greater than $0.5m$, the data suggests that $\eta > \eta_0$. If S is much less than $0.5m$, the data suggests that $\eta < \eta_0$.



S has a **Binomial distribution** when H_0 is true. The Binomial distribution is used to construct a test with size α (approximately). For a two-sided alternative $H_A : \eta \neq \eta_0$, the test rejects H_0 when S is significantly different from $0.5m$, as determined from the reference Binomial distribution. One-sided tests use the corresponding lower or upper tail of the distribution. To generate a CI for η, you can exploit the duality between CI and tests. A $100(1-\alpha)\%$ CI for η consists of all values η_0 not rejected by a two-sided size α test of

$H_0 : \eta = \eta_0$.

Not all test sizes and confidence levels are possible because the test statistic S is discrete valued. R's `SIGN.test()` in the `BSDA` package gives an exact p-value for the test, and approximates the desired confidence level using a linear interpolation algorithm.

Example: Income Data Recall that the income distribution is extremely skewed, with two extreme outliers at 46 and 1110.

```
#### Example: Income Data
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)
```

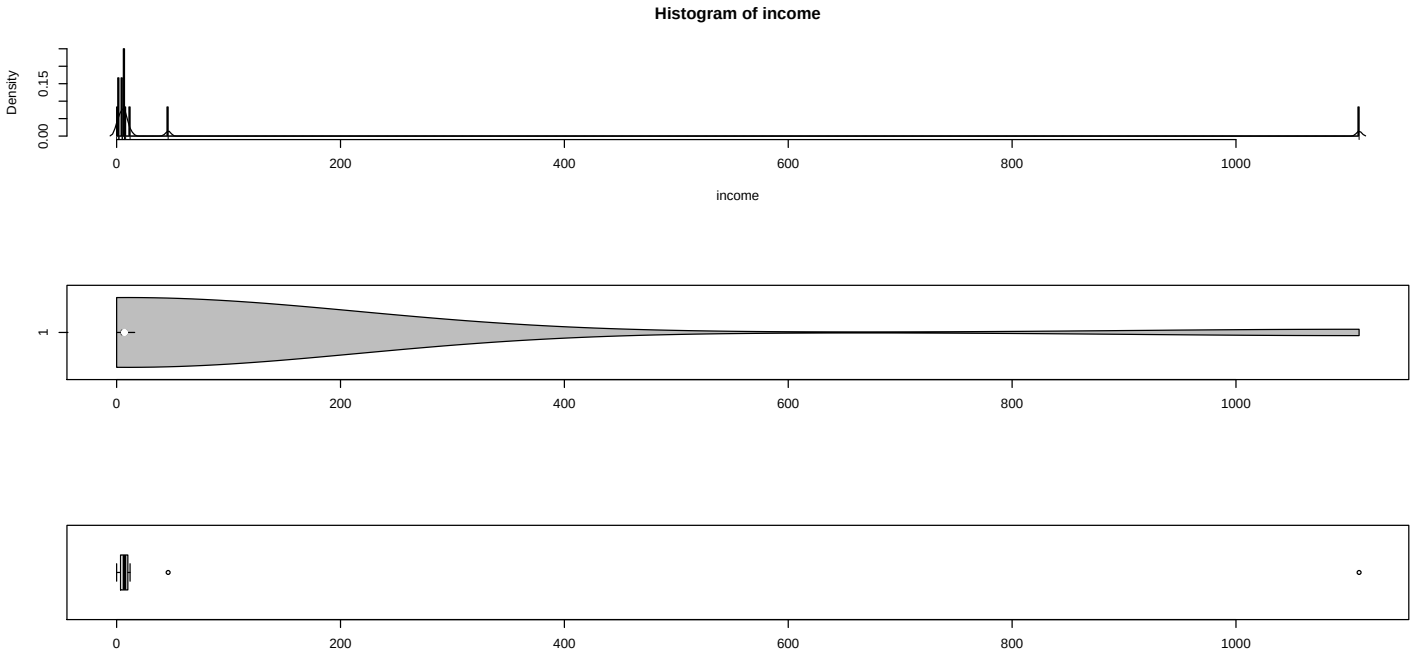
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    4.25    7.00  100.92    9.00 1110.00
```

```
sd(income)
```

```
## [1] 318.0078
```

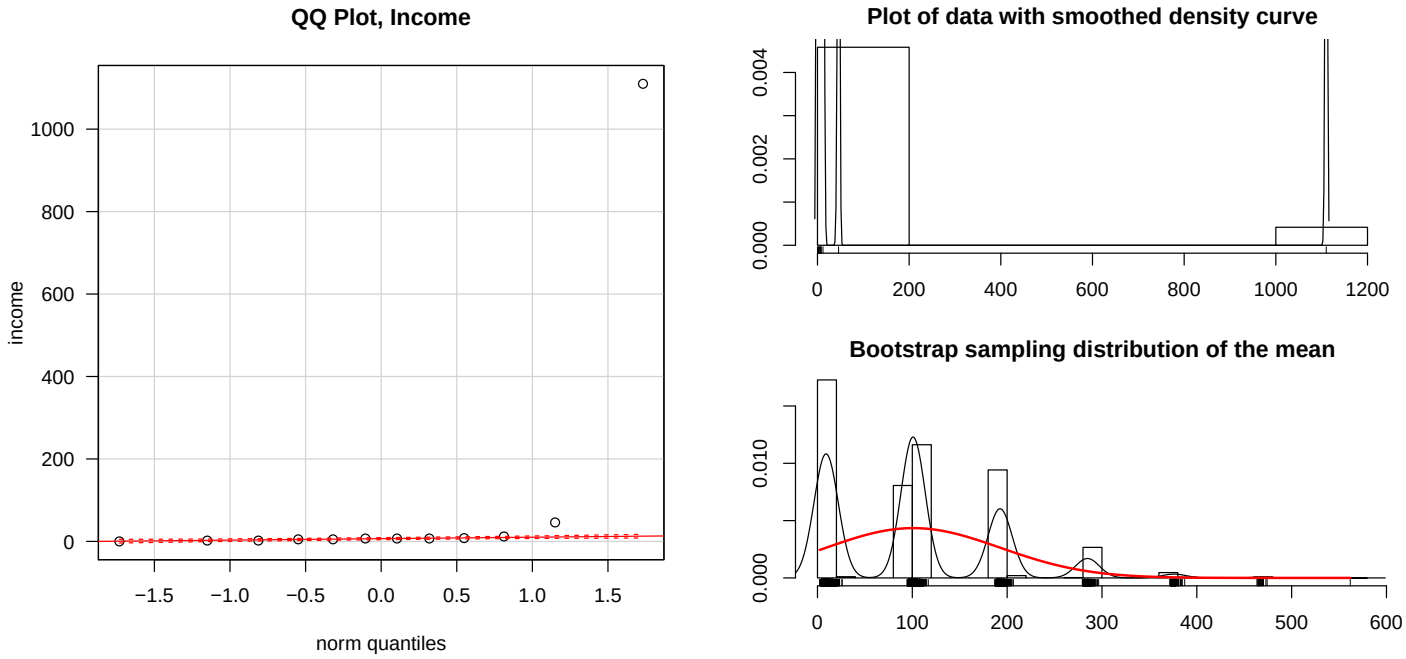
The income data is unimodal, skewed right, with two extreme outliers.

```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(income, freq = FALSE, breaks = 1000)
points(density(income), type = "l")
rug(income)

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

# boxplot
boxplot(income, horizontal=TRUE)
```



The normal QQ-plot of the sample data indicates strong deviation from normality, and the CLT can't save us: even the bootstrap sampling distribution of the mean indicates strong deviation from normality.

```
library(car)
qqPlot(income, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Income")

bs.one.samp.dist(income)
```



The presence of the outliers has a dramatic effect on the 95% CI for the population mean income μ, which goes from -101 to 303 (in 1000 dollar units).

This t-CI is suspect because the normality assumption is unreasonable. A CI for the population median income η is more sensible because the median is likely to be a more reasonable measure of typical value. Using the sign procedure, you are 95% confident that the population median income is between 2.32 and 11.57 (times \$1000).

```
library(BSDA)
```

```
## Loading required package:  lattice
##
## Attaching package:  'BSDA'
## The following objects are masked from 'package:car':
##
##    Vocab, Wool
## The following object is masked from 'package:TeachingDemos':
##
##    z.test
## The following object is masked from 'package:datasets':
##
##    Orange
```

```
t.test(income)
```

```
##
##  One Sample t-test
##
## data:  income
## t = 1.0993, df = 11, p-value = 0.2951
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -101.1359  302.9692
## sample estimates:
## mean of x
##  100.9167
```

```
SIGN.test(income)
```

```
##
##  One-sample Sign-Test
##
## data:  income
## s = 11, p-value = 0.0009766
## alternative hypothesis: true median is not equal to 0
## 95 percent confidence interval:
##   2.319091 11.574545
## sample estimates:
## median of x
##           7
##
## Achieved and Interpolated Confidence Intervals:
##
##                   Conf.Level L.E.pt  U.E.pt
```

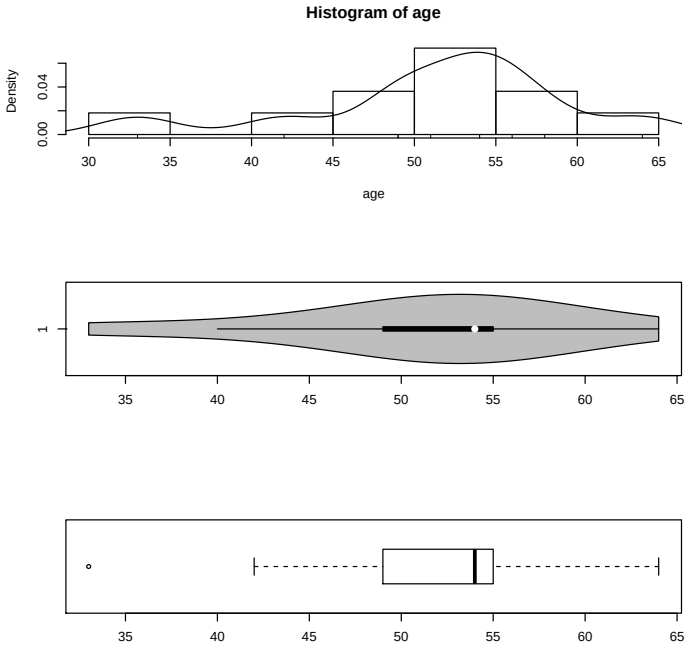
```
## Lower Achieved CI     0.8540 5.0000  8.0000
## Interpolated CI       0.9500 2.3191 11.5745
## Upper Achieved CI     0.9614 2.0000 12.0000
```

Example: Age at First Heart Transplant Recall that the distribution of ages is skewed to the left with a lower outlier. A question of interest is whether the "typical age" at first transplant is 50. This can be formulated as a test about the population median η or as a test about the population mean μ, depending on the interpretation.

```
#### Example: Age at First Heart Transplant
age <- c(54, 42, 51, 54, 49, 56, 33, 58, 54, 64, 49)
# sort in decreasing order
age <- sort(age, decreasing = TRUE)
age
##  [1] 64 58 56 54 54 54 51 49 49 42 33
summary(age)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   33.00   49.00   54.00   51.27   55.00   64.00
sd(age)
## [1] 8.25943
```

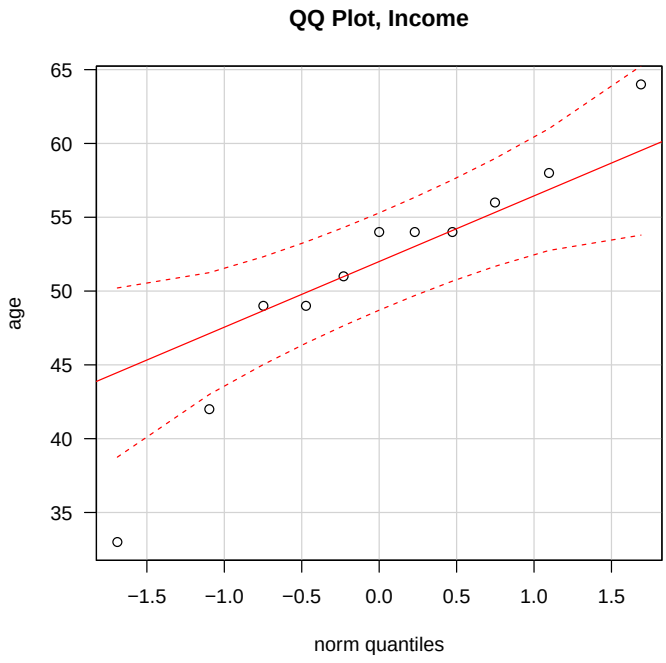
The age data is unimodal, skewed left, no extreme outliers.

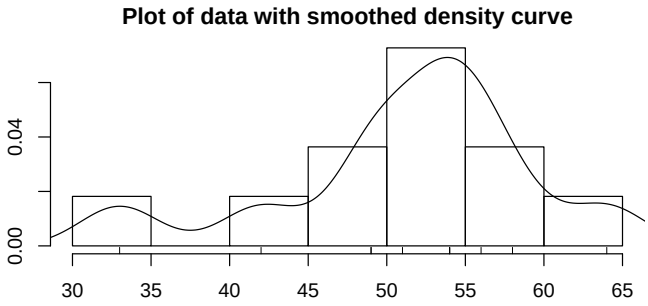
```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(age, freq = FALSE, breaks = 10)
points(density(age), type = "l")
rug(age)

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

# boxplot
boxplot(age, horizontal=TRUE)
```

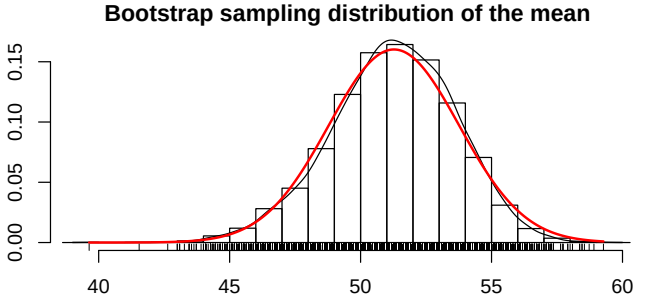


The normal QQ-plot of the sample data indicates mild deviation from normality in the left tail (2 points of 11 outside the bands), and the bootstrap sampling distribution of the mean indicates weak deviation from normality. It is good practice in this case to use the nonparametric test as a double-check of the t-test, with the nonparametric test being the more conservative test.

```
library(car)
qqPlot(age, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Income")

bs.one.samp.dist(age)
```







The sign test for $H_0 : \eta = 50$ against $H_A : \eta \neq 50$ has a p-value of 0.549, which is not sufficient to reject H_0. A 95% CI for η is 47.0 to 56.6 years, which includes the hypothesized median age of 50. Similar conclusions are reached with the t-CI and the test on μ, but you should have less confidence in these results because the normality assumption is tenuous.

```
library(BSDA)
t.test(age, mu=50)
```

```
##
##  One Sample t-test
##
## data:  age
## t = 0.51107, df = 10, p-value = 0.6204
## alternative hypothesis: true mean is not equal to 50
## 95 percent confidence interval:
##  45.72397 56.82149
## sample estimates:
## mean of x
##  51.27273
```

```
SIGN.test(age, md=50)
```

```
##
##  One-sample Sign-Test
##
## data:  age
## s = 7, p-value = 0.5488
## alternative hypothesis: true median is not equal to 50
## 95 percent confidence interval:
##  46.98909 56.57455
## sample estimates:
## median of x
##          54
##
## Achieved and Interpolated Confidence Intervals:
##
##                   Conf.Level  L.E.pt  U.E.pt
## Lower Achieved CI     0.9346 49.0000 56.0000
## Interpolated CI       0.9500 46.9891 56.5745
## Upper Achieved CI     0.9883 42.0000 58.0000
```

6.3 Wilcoxon Signed-Rank Procedures

The **Wilcoxon** procedure assumes you have a random sample from a population with a **symmetric** frequency curve. The curve need not be normal. The

test and CI can be viewed as procedures for either the population median or mean.

To illustrate the computation of the Wilcoxon statistic W, suppose you wish to test $H_0 : \mu = \mu_0 = 10$ on the made-up data below. The test statistic requires us to compute the **signs** of $X_i - \mu_0$ and the **ranks** of $|X_i - \mu_0|$. Ties in $|X_i - \mu_0|$ get the average rank and observations at μ_0 (here 10) are always discarded. The Wilcoxon statistic is the **sum of the signed ranks for observations above** $\mu_0 = 10$. For us

$$W \;=\; 6 + 4.5 + 8 + 2 + 4.5 + 7 \;=\; 32.$$

X_i	$X_i - 10$	sign	$\lvert X_i - 10\rvert$	rank	sign$\times$ rank
20	10	+	10	6	6
18	8	+	8	4.5	4.5
23	13	+	13	8	8
5	−5	−	5	3	−3
14	4	+	4	2	2
8	−2	−	2	1	−1
18	8	+	8	4.5	4.5
22	12	+	12	7	7

The sum of all ranks is always $0.5m(m+1)$, where m is the sample size. If H_0 is true, you expect W to be approximately $0.5 \times 0.5m(m+1) = 0.25m(m+1)$. Why? Recall that W adds up the ranks for observations above μ_0. If H_0 is true, you expect $1/2$ of all observations to be above μ_0, assuming the population distribution is symmetric. The ranks of observations above μ_0 should add to approximately $1/2$ times the sum of all ranks. You reject H_0 in favor of $H_A : \mu \neq \mu_0$ if W is much larger than, or much smaller than $0.25m(m+1)$. One sided tests can also be constructed. The Wilcoxon CI for μ is computed in a manner analogous to that described for the sign CI.

Here, $m = 8$ so the sum of all ranks is $0.5 \times 8 \times 9 = 36$ (check yourself). The expected value of W is $0.5 \times 0.5 \times 8 \times 9 = 18$. Is the observed value of $W = 32$ **far from** the expected value of 18? To formally answer this question,

we need to use the Wilcoxon procedures, which are implemented in R with `wilcox.test()`.

Example: Made-up Data The boxplot indicates that the distribution is fairly symmetric, so the Wilcoxon method is reasonable (so is a t-CI and test).

```
#### Example: Made-up Data
dat <- c(20, 18, 23,  5, 14,  8, 18, 22)
# sort in decreasing order
dat <- sort(dat, decreasing = TRUE)
dat
```

```
## [1] 23 22 20 18 18 14  8  5
```

```
summary(dat)
```

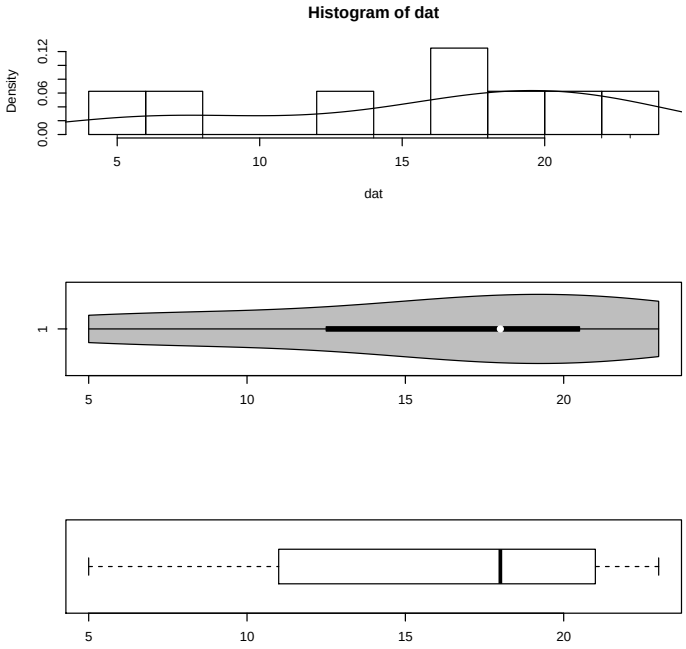
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0    12.5    18.0    16.0    20.5    23.0
```

```
sd(dat)
```

```
## [1] 6.524678
```

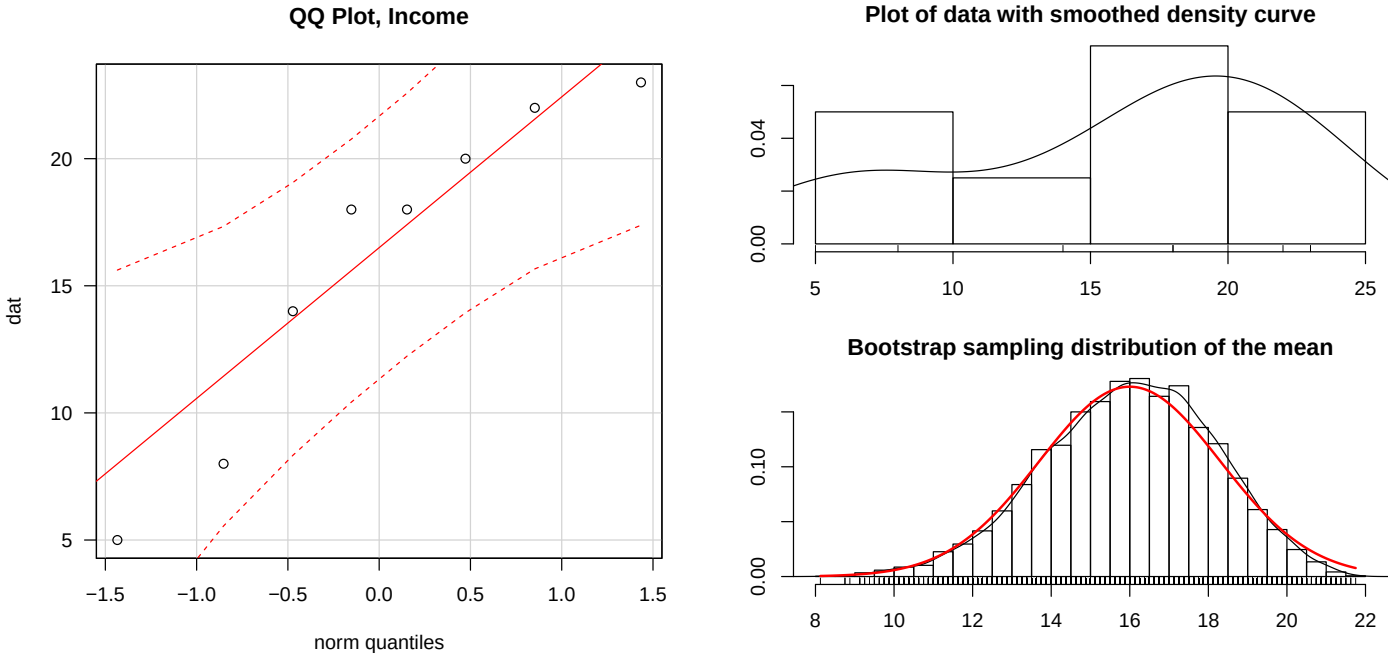
The dat data is unimodal, skewed left, no extreme outliers.

```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(dat, freq = FALSE, breaks = 10)
points(density(dat), type = "l")
rug(dat)

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

# boxplot
boxplot(dat, horizontal=TRUE)
```



The normal QQ-plot of the sample data indicates insufficient evidence of deviation from normality though both the QQ-plot and the bootstrap sampling distribution of the mean indicates weak left-skewness. Either the Wilcoxon or t-test are appropriate.

```
par(mfrow=c(1,1))
library(car)
qqPlot(dat, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Income")

bs.one.samp.dist(dat)
```



The Wilcoxon p-value with continuity correction for testing $H_0 : \mu = 10$ against a two-sided alternative is 0.058. This would not lead to rejecting H_0 at the 5% level.

```
t.test(dat, mu=10)
```

```
##
##  One Sample t-test
##
## data:  dat
## t = 2.601, df = 7, p-value = 0.03537
## alternative hypothesis: true mean is not equal to 10
## 95 percent confidence interval:
##  10.54523 21.45477
## sample estimates:
## mean of x
##        16
```

```
# with continuity correction in the normal approximation for the p-value
wilcox.test(dat, mu=10, conf.int=TRUE)
```

```
## Warning in wilcox.test.default(dat, mu = 10, conf.int = TRUE): cannot compute exact p-value
with ties
## Warning in wilcox.test.default(dat, mu = 10, conf.int = TRUE): cannot compute exact confide
interval with ties
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  dat
## V = 32, p-value = 0.0584
## alternative hypothesis: true location is not equal to 10
## 95 percent confidence interval:
##   9.500002 21.499942
## sample estimates:
## (pseudo)median
##        16.0056
```

```
# without continuity correction
wilcox.test(dat, mu=10, conf.int=TRUE, correct=FALSE)
```

```
## Warning in wilcox.test.default(dat, mu = 10, conf.int = TRUE, correct = FALSE): cannot
compute exact p-value with ties
## Warning in wilcox.test.default(dat, mu = 10, conf.int = TRUE, correct = FALSE): cannot
compute exact confidence interval with ties
##
##  Wilcoxon signed rank test
##
## data:  dat
## V = 32, p-value = 0.04967
## alternative hypothesis: true location is not equal to 10
## 95 percent confidence interval:
##  10.99996 21.00005
```

```
## sample estimates:
## (pseudo)median
##        16.0056
```

6.3.1 Nonparametric Analyses of Paired Data

Nonparametric methods for single samples can be used to analyze paired data because the difference between responses within pairs is the unit of analysis.

Example: Sleep Remedies I will illustrate Wilcoxon methods on the paired comparison of two remedies A and B for insomnia. The number of hours of sleep gained on each method was recorded.

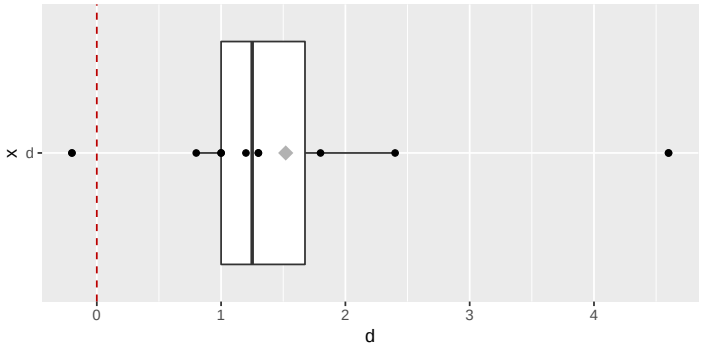
```
#### Example: Sleep Remedies
# Data and numerical summaries
a <- c( 0.7, -1.6, -0.2, -1.2,  0.1,  3.4,  3.7,  0.8,  0.0,  2.0)
b <- c( 1.9,  0.8,  1.1,  0.1, -0.1,  4.4,  5.5,  1.6,  4.6,  3.0)
d <- b - a;
sleep <- data.frame(a, b, d)
summary(sleep$d)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -0.200   1.000   1.250   1.520   1.675   4.600
```

```
shapiro.test(sleep$d)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sleep$d
## W = 0.83798, p-value = 0.04173
```

```
# boxplot
library(ggplot2)
p3 <- ggplot(sleep, aes(x = "d", y = d))
p3 <- p3 + geom_hline(yintercept=0, colour="#BB0000", linetype="dashed")
p3 <- p3 + geom_boxplot()
p3 <- p3 + geom_point()
p3 <- p3 + stat_summary(fun.y = mean, geom = "point", shape = 18,
                        size = 4, alpha = 0.3)
p3 <- p3 + coord_flip()
print(p3)
```



The boxplot shows that distribution of differences is reasonably symmetric but not normal. Recall that the Shapiro-Wilk test of normality was significant at the 5% level (p-value=0.042). It is sensible to use the Wilcoxon procedure on the differences. Let μ_B be the population mean sleep gain on remedy B, and μ_A be the population mean sleep gain on remedy A. You are 95% confident that $\mu_B - \mu_A$ is between 0.8 and 2.8 hours. Putting this another way, you are 95% confident that μ_B exceeds μ_A by between 0.8 and 2.8 hours. The p-value for testing $H_0 : \mu_B - \mu_A = 0$ against a two-sided alternative is 0.008, which strongly suggests that $\mu_B \neq \mu_A$. This agrees with the CI. Note that the t-CI and test give qualitatively similar conclusions as the Wilcoxon methods, but the t-test p-value is about half as large.

If you are uncomfortable with the symmetry assumption, you could use the sign CI for the population median difference between B and A. I will note that a 95% CI for the median difference goes from 0.86 to 2.2 hours.

```
t.test(sleep$d, mu=0)
```

```
##
##  One Sample t-test
##
## data:  sleep$d
## t = 3.7796, df = 9, p-value = 0.004352
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.610249 2.429751
## sample estimates:
## mean of x
##      1.52
```

```
# with continuity correction in the normal approximation for the p-value
wilcox.test(sleep$d, mu=0, conf.int=TRUE)
```

```
## Warning in wilcox.test.default(sleep$d, mu = 0, conf.int = TRUE): cannot compute exact
p-value with ties
```

```
## Warning in wilcox.test.default(sleep$d, mu = 0, conf.int = TRUE): cannot compute exact
confidence interval with ties
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  sleep$d
## V = 54, p-value = 0.008004
## alternative hypothesis: true location is not equal to 0
## 95 percent confidence interval:
##  0.7999339 2.7999620
## sample estimates:
## (pseudo)median
##       1.299983
```

```
# can use the paired= option
#wilcox.test(sleep£b, sleep£a, paired=TRUE, mu=0, conf.int=TRUE)
# if don't assume symmetry, can use sign test
#SIGN.test(sleep£d)
```

6.3.2 Comments on One-Sample Nonparametric Methods

For this discussion, I will assume that the underlying population distribution is (approximately) symmetric, which implies that population means and medians are equal (approximately). For symmetric distributions the t, sign, and Wilcoxon procedures are all appropriate.

If the underlying population distribution is extremely skewed, you can use the sign procedure to get a CI for the population median. Alternatively, as illustrated on HW 2, you can transform the data to a scale where the underlying distribution is nearly normal, and then use the classical t-methods. Moderate degrees of skewness will not likely have a big impact on the standard t-test and CI.

The one-sample t-test and CI are optimal when the underlying population frequency curve is normal. Essentially this means that the t-CI is, on average, narrowest among all CI procedures with given level, or that the t-test has the highest power among all tests with a given size. The width of a CI provides a measure of the sensitivity of the estimation method. For a given level CI, the narrower CI better pinpoints the unknown population mean.

With heavy-tailed symmetric distributions, the t-test and CI tend to be conservative. Thus, for example, a nominal 95% t-CI has actual coverage rates higher than 95%, and the nominal 5% t-test has an actual size smaller than 5%. The t-test and CI possess a property that is commonly called **robustness of validity**. However, data from heavy-tailed distributions can have a profound effect on the **sensitivity** of the t-test and CI. Outliers can dramatically inflate the standard error of the mean, causing the CI to be needlessly wide, and tests to have diminished power (outliers typically inflate p-values for the t-test). The sign and Wilcoxon procedures downweight the influence of outliers by looking at sign or signed-ranks instead of the actual data values. These two nonparametric methods are somewhat less efficient than the t-methods when the population is normal (efficiency is about 0.64 and 0.96 for the sign and Wilcoxon methods relative to the normal t-methods, where efficiency is the ratio of sample sizes needed for equal power), but can be infinitely more efficient with heavier than normal tailed distributions. In essence, the t-methods do not have a **robustness of sensitivity**.

Nonparametric methods have gained widespread acceptance in many scientific disciplines, but not all. Scientists in some disciplines continue to use classical t-methods because they believe that the methods are robust to non-normality. As noted above, this is a robustness of validity, not sensitivity. This misconception is unfortunate, and results in the routine use of methods that are less powerful than the non-parametric techniques. **Scientists need to be flexible and adapt their tools to the problem at hand, rather than use the same tool indiscriminately!** I have run into suspicion that use of nonparametric methods was an attempt to "cheat" in some way — properly applied, they are excellent tools that *should* be used.

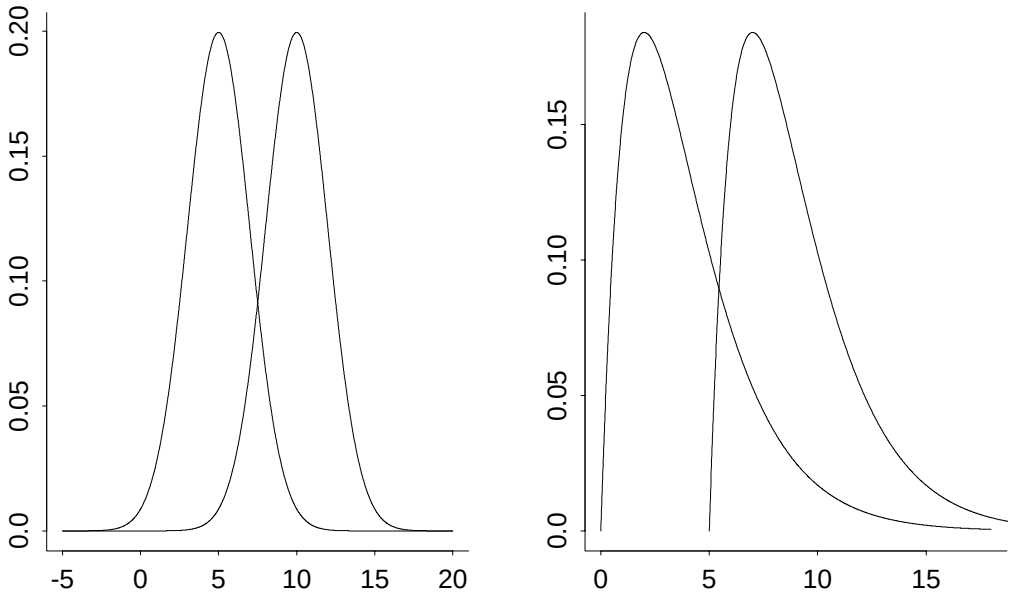
A minor weakness of nonparametric methods is that they do not easily generalize to complex modelling problems. A great deal of progress has been made in this area, but most software packages have not included the more advanced techniques (R is among the forerunners).

Nonparametric statistics used to refer almost exclusively to the set of meth-

ods such as we have been discussing that provided analogs like tests and CIs to the normal theory methods without requiring the assumption of sampling from normal distributions. There is now a large area of statistics also called nonparametric methods not focused on these goals at all. In our department we (used to) have a course titled "Nonparametric Curve Estimation & Image Reconstruction", where the focus is much more general than relaxing an assumption of normality. In that sense, what we are covering in this course could be considered "classical" nonparametrics.

6.4 (Wilcoxon-)Mann-Whitney Two-Sample Procedure

The WMW procedure assumes you have independent random samples from the two populations, and assumes that the populations have the **same shapes and spreads** (the frequency curves for the two populations are "shifted" versions of each other — see below). The frequency curves are not required to be symmetric. The WMW procedures give a CI and tests on the difference $\eta_1 - \eta_2$ between the two population medians. If the populations are symmetric, then the methods apply to $\mu_1 - \mu_2$.



The R help on `?wilcox.test` gives references to how the exact WMW procedure is actually calculated; here is a good approximation to the exact method that is easier to understand. The WMW procedure is based on ranks. The two samples are combined, ranked from smallest to largest (1=smallest) and separated back into the original samples. If the two populations have equal medians, you expect the average rank in the two samples to be roughly equal. The WMW test computes a classical two sample t-test using the pooled variance on the ranks to assess whether the sample mean ranks are significantly different.

Example: Comparison of Cooling Rates of Uwet and Walker Co. Meteorites The Uwet[1] (Cross River, Nigeria) and Walker[2] County (Alabama, US) meteorite cooling rate data are below. A primary interest is comparing the population "typical" cooling rate measurements.

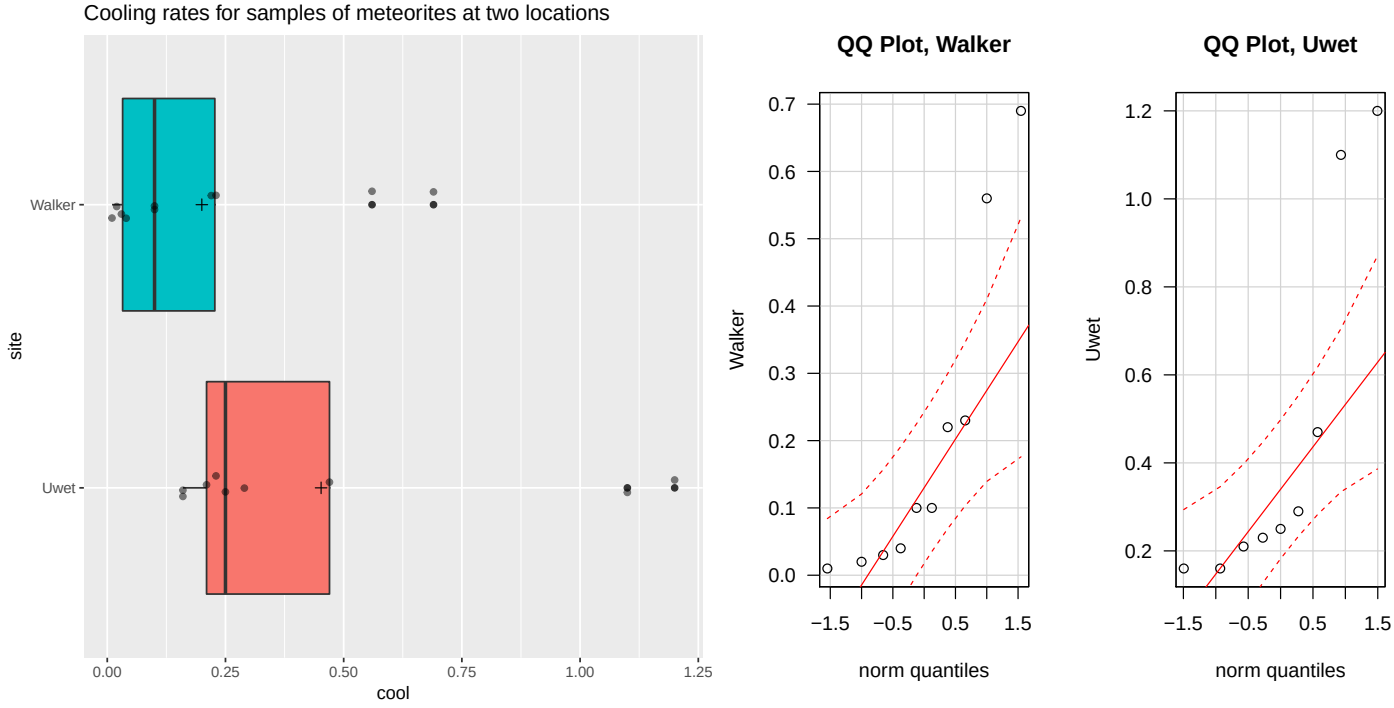
[1] `http://www.lpi.usra.edu/meteor/metbull.php?code=24138`

[2] `http://www.lpi.usra.edu/meteor/metbull.php?code=24204`

```
#### Example: Comparison of Cooling Rates of Uwet and Walker Co. Meteorites
Uwet   <- c(0.21, 0.25, 0.16, 0.23, 0.47, 1.20, 0.29, 1.10, 0.16)
Walker <- c(0.69, 0.23, 0.10, 0.03, 0.56, 0.10, 0.01, 0.02, 0.04, 0.22)
```

The boxplots and normal QQ-plots show that the distributions are rather skewed to the right. The AD test of normality indicate that a normality assumption is unreasonable for each population.

```
met <- data.frame(Uwet=c(Uwet,NA), Walker)
library(reshape2)
met.long <- melt(met, variable.name = "site", value.name = "cool", na.rm=TRUE)

## No id variables; using all as measure variables
# naming variables manually, the variable.name and value.name not working 11/2012
names(met.long) <- c("site", "cool")
library(ggplot2)
p <- ggplot(met.long, aes(x = site, y = cool, fill=site))
p <- p + geom_boxplot()
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 3, size = 2)
p <- p + coord_flip()
p <- p + labs(title = "Cooling rates for samples of meteorites at two locations")
p <- p + theme(legend.position="none")
print(p)

par(mfrow=c(1,2))
library(car)
qqPlot(Walker, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Walker")
qqPlot(Uwet, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Uwet")
```



I carried out the standard two-sample procedures to see what happens. The

pooled-variance and Satterthwaithe results are comparable, which is expected because the sample standard deviations and sample sizes are roughly equal. Both tests indicate that the mean cooling rates for Uwet and Walker Co. meteorites are not significantly different at the 10% level. You are 95% confident that the mean cooling rate for Uwet is at most 0.1 less, and no more than 0.6 greater than that for Walker Co. (in degrees per million years).

```
# numerical summaries
summary(Uwet)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1600  0.2100  0.2500  0.4522  0.4700  1.2000
```

```
c(sd(Uwet), IQR(Uwet), length(Uwet))
```

```
## [1] 0.4069944 0.2600000 9.0000000
```

```
summary(Walker)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0100  0.0325  0.1000  0.2000  0.2275  0.6900
```

```
c(sd(Walker), IQR(Walker), length(Walker))
```

```
## [1]  0.2389793  0.1950000 10.0000000
```

```
t.test(Uwet, Walker, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  Uwet and Walker
## t = 1.6689, df = 17, p-value = 0.1134
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.0666266  0.5710710
## sample estimates:
## mean of x mean of y
## 0.4522222 0.2000000
```

```
t.test(Uwet, Walker)
```

```
##
##  Welch Two Sample t-test
##
## data:  Uwet and Walker
## t = 1.6242, df = 12.652, p-value = 0.129
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.08420858  0.58865302
## sample estimates:
## mean of x mean of y
## 0.4522222 0.2000000
```

Given the marked skewness, a nonparametric procedure is more appropriate. The Wilcoxon-Mann-Whitney comparison of population medians is reasonable. Why? The WMW test of equal population medians is significant (barely) at the 5% level. You are 95% confident that median cooling rate for Uwet exceeds that for Walker by between 0+ and 0.45 degrees per million years.

```
wilcox.test(Uwet, Walker, conf.int = TRUE)
```

```
## Warning in wilcox.test.default(Uwet, Walker, conf.int = TRUE): cannot compute exact p-value
with ties
## Warning in wilcox.test.default(Uwet, Walker, conf.int = TRUE): cannot compute exact confide
intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Uwet and Walker
## W = 69.5, p-value = 0.04974
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  0.0000449737 0.4499654518
## sample estimates:
## difference in location
##              0.1702657
```

The difference between the WMW and t-test p-values and CI lengths (i.e. the WMW CI is narrower and the p-value smaller) reflects the effect of the outliers on the sensitivity of the standard tests and CI.

I conducted a pooled-variance two-sample t-test on ranks to show you that the p-value is close to the WMW p-value, as expected.

```
rank(met.long$cool)
```

```
##  [1]  9.0 13.0  7.5 11.5 15.0 19.0 14.0 18.0  7.5 17.0 11.5  5.5  3.0
## [14] 16.0  5.5  1.0  2.0  4.0 10.0
```

```
by(rank(met.long$cool), met.long$site, summary)
```

```
## met.long$site: Uwet
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.50    9.00   13.00   12.72   15.00   19.00
## ------------------------------------------------------------
## met.long$site: Walker
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    3.25    5.50    7.55   11.12   17.00
```

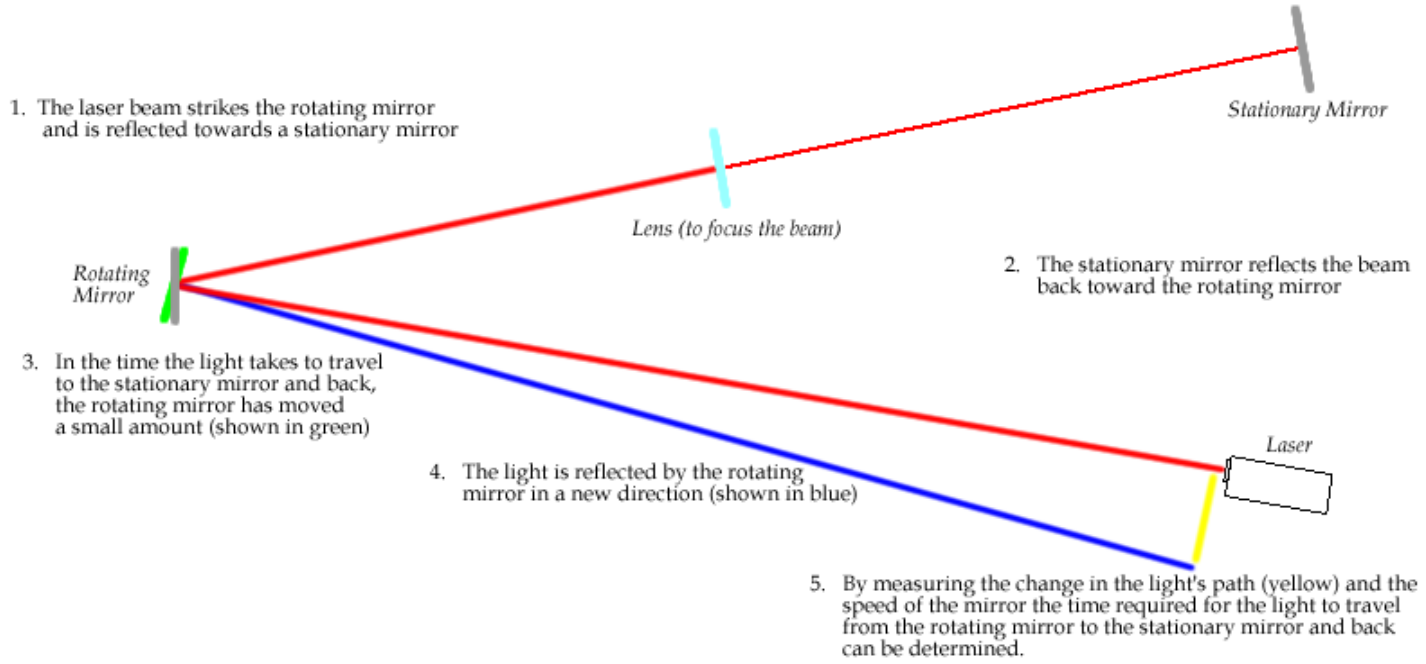
```
# note: the CI for ranks is not interpretable
t.test(rank(met.long$cool) ~ met.long$site, var.equal = TRUE)
```

```
##
##  Two Sample t-test
```

```
##
## data:  rank(met.long$cool) by met.long$site
## t = 2.2082, df = 17, p-value = 0.04125
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   0.2304938 10.1139507
## sample estimates:
##   mean in group Uwet mean in group Walker
##             12.72222              7.55000
```

Example: Newcombe's Data Experiments of historical importance were performed beginning in the eighteenth century to determine physical constants, such as the mean density of the earth, the distance from the earth to the sun, and the velocity of light. An interesting series of experiments to determine the velocity of light was begun in 1875. The first method used, and reused with refinements several times thereafter, was the rotating mirror method[3]. In this method a beam of light is reflected off a rapidly rotating mirror to a fixed mirror at a carefully measured distance from the source. The returning light is re-reflected from the rotating mirror at a different angle, because the mirror has turned slightly during the passage of the corresponding light pulses. From the speed of rotation of the mirror and from careful measurements of the angular difference between the outward-bound and returning light beams, the passage time of light can be calculated for the given distance. After averaging several calculations and applying various corrections, the experimenter can combine mean passage time and distance for a determination of the velocity of light. Simon Newcombe, a distinguished American scientist, used this method during the year 1882 to generate the passage time measurements given below, in microseconds. The travel path for this experiment was 3721 meters in length, extending from Ft. Meyer, on the west bank of the Potomac River in Washington, D.C., to a fixed mirror at the base of the Washington Monument.

[3] http://en.wikipedia.org/wiki/File:Speed_of_light_(foucault).PNG



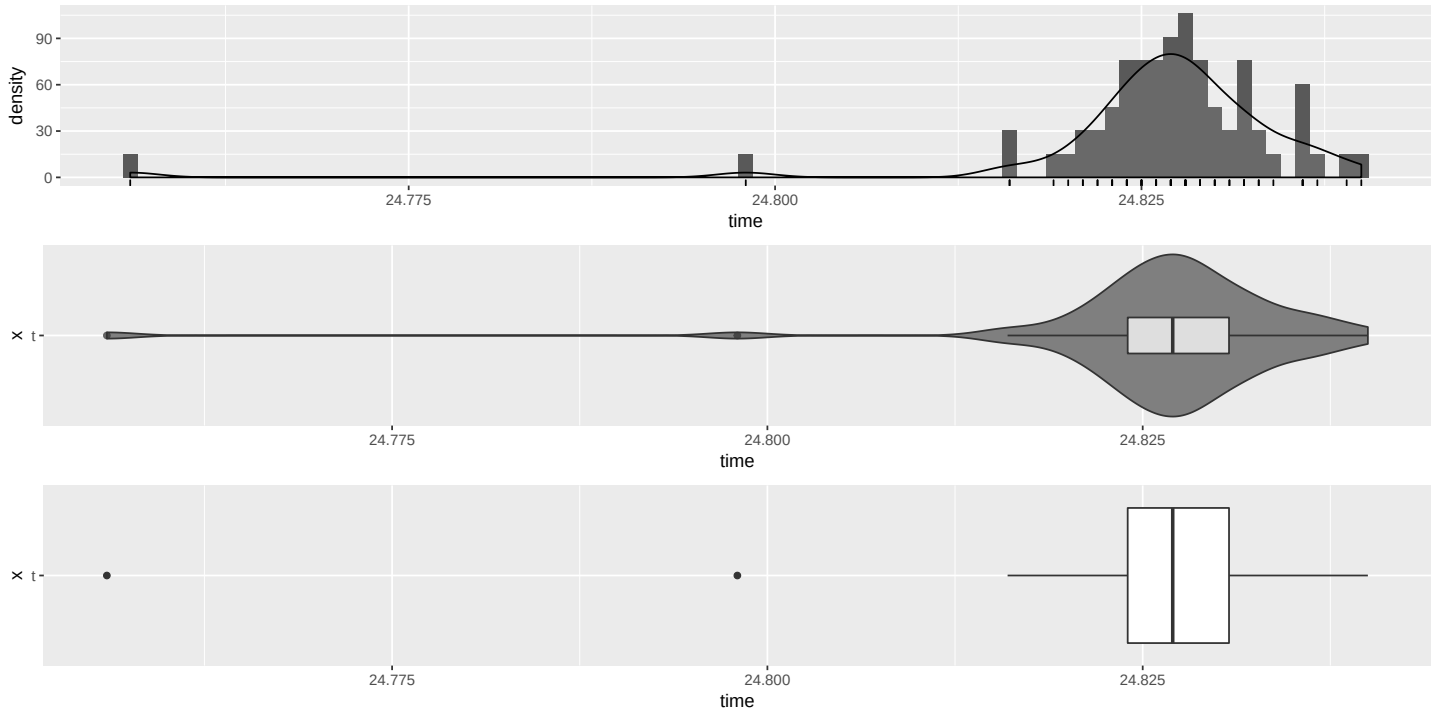
The problem is to determine a 95% CI for the "true" passage time, which is taken to be the typical time (mean or median) of the population of measurements that were or could have been taken by this experiment.

```
#### Example: Newcombe's Data
time <- c(24.828, 24.833, 24.834, 24.826, 24.824, 24.756
        , 24.827, 24.840, 24.829, 24.816, 24.798, 24.822
        , 24.824, 24.825, 24.823, 24.821, 24.830, 24.829
        , 24.831, 24.824, 24.836, 24.819, 24.820, 24.832
        , 24.836, 24.825, 24.828, 24.828, 24.821, 24.829
        , 24.837, 24.828, 24.830, 24.825, 24.826, 24.832
        , 24.836, 24.830, 24.836, 24.826, 24.822, 24.823
        , 24.827, 24.828, 24.831, 24.827, 24.827, 24.827
        , 24.826, 24.826, 24.832, 24.833, 24.832, 24.824
        , 24.839, 24.824, 24.832, 24.828, 24.825, 24.825
        , 24.829, 24.828, 24.816, 24.827, 24.829, 24.823)
library(nortest)
ad.test(time)
```

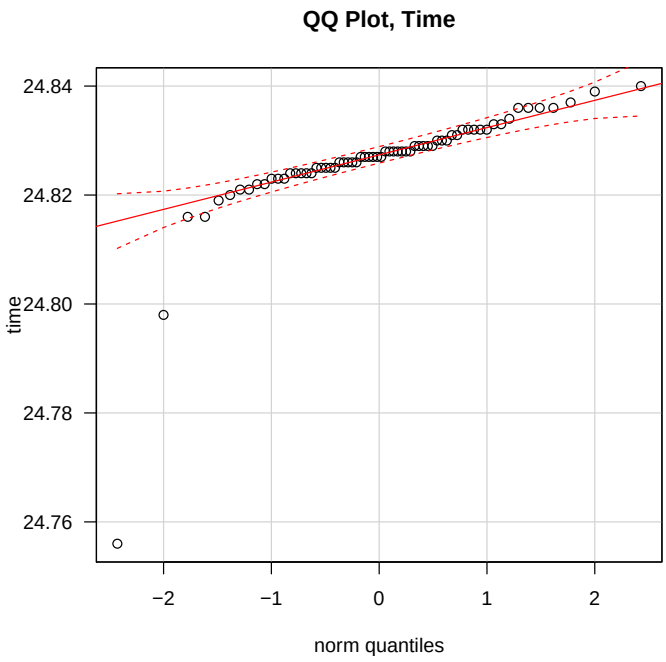
```
##
##  Anderson-Darling normality test
##
## data:  time
## A = 5.8843, p-value = 1.217e-14
```

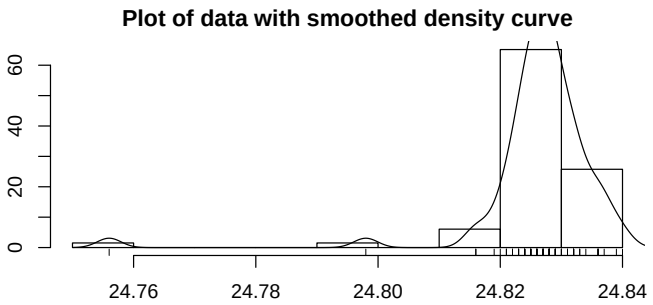
```
# Histogram overlaid with kernel density curve
Passage_df <- data.frame(time)
p1 <- ggplot(Passage_df, aes(x = time))
  # Histogram with density instead of count on y-axis
```

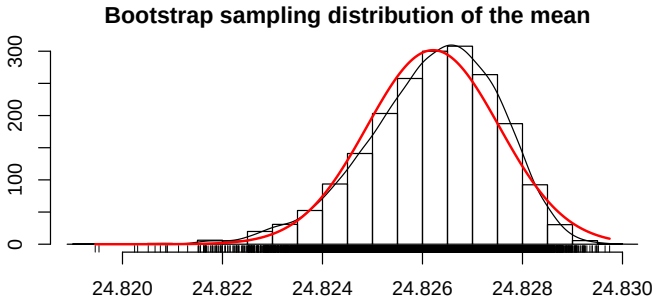
```
p1 <- p1 + geom_histogram(aes(y=..density..), binwidth=0.001)
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

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

# boxplot
p3 <- ggplot(Passage_df, aes(x = "t", y = time))
p3 <- p3 + geom_boxplot()
p3 <- p3 + coord_flip()

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




```
par(mfrow=c(1,1))
library(car)
qqPlot(time, las = 1, id.n = 0, id.cex = 1, lwd = 1, main="QQ Plot, Time")

bs.one.samp.dist(time)
```







The data set is skewed to the left, due to the presence of two extreme outliers that could potentially be misrecorded observations. Without additional information I would be hesitant to apply normal theory methods (the t-test), even though the sample size is "large" (bootstrap sampling distribution is still left-skewed). Furthermore, the t-test still suffers from a lack of robustness of sensitivity, even in large samples. A formal QQ-plot and normal test rejects, at the 0.01 level, the normality assumption needed for the standard methods.

The table below gives 95% t, sign, and Wilcoxon CIs. I am more comfortable with the sign CI for the population median than the Wilcoxon method, which assumes symmetry.

```
t.sum <- t.test(time)
t.sum$conf.int
## [1] 24.82357 24.82885
## attr(,"conf.level")
## [1] 0.95
diff(t.test(time)$conf.int)
## [1] 0.005283061
s.sum <- SIGN.test(time)
s.sum[2,c(2,3)]
## Error in s.sum[2, c(2, 3)]:  incorrect number of dimensions
diff(s.sum[2,c(2,3)])
## Error in s.sum[2, c(2, 3)]:  incorrect number of dimensions
w.sum <- wilcox.test(time, conf.int=TRUE)
w.sum$conf.int
```

```
## [1] 24.82604 24.82853
## attr(,"conf.level")
## [1] 0.95
diff(w.sum$conf.int)
## [1] 0.002487969
```

parameter	Method	CI Limits	Width
mean	t	(24.8236, 24.8289)	0.0053
median	sign	(24.8260, 24.8285)	0.0025
median	Wilcoxon	(24.8260, 24.8285)	0.0025

Note the big difference between the nonparametric and the t-CI. The nonparametric CIs are about 1/2 as wide as the t-CI. This reflects the impact that outliers have on the standard deviation, which directly influences the CI width.

6.5 Alternatives for ANOVA and Planned Comparisons

The classical ANOVA assumes that the populations have normal frequency curves and the populations have equal variances (or spreads). You learned formal tests for these assumptions in Chapter 5. When the assumptions do not hold, you can try one of the following two approaches. Before describing alternative methods, I will note that deviations from normality in one or more samples might be expected in a comparison involving many samples. You should downplay small deviations from normality in problems involving many samples.

6.5.1 Kruskal-Wallis ANOVA

The **Kruskal-Wallis** (KW) test is a non-parametric method for testing the hypothesis of equal population medians against the alternative that not all population medians are equal. The procedure assumes you have independent random samples from populations with frequency curves having **identical shapes and spreads**. The KW ANOVA is essentially the standard ANOVA based

on ranked data. That is, we combine the samples, rank the observations from smallest to largest, and then return the ranks to the original samples and do the standard ANOVA using the ranks. The KW ANOVA is a multiple sample analog of the Wilcoxon-Mann-Whitney two sample procedure. Hence, multiple comparisons for a KW analysis, be they FSD or Bonferroni comparisons, are based on the two sample WMW procedure.

6.5.2 Transforming Data

The distributions in many data sets are skewed to the right with outliers. If the sample spreads, say s and IQR, increase with an increasing mean or median, you can often **transform data** to a scale where the normality and the constant spread assumption are more nearly satisfied. The transformed data are analyzed using the standard ANOVA. The two most commonly used transforms for this problem are the square root and natural logarithm, provided the data are non-negative[4].

If the original distributions are nearly symmetric, but heavy-tailed, non-linear transformations will tend to destroy the symmetry. Many statisticians recommend methods based on trimmed means for such data. These methods are not commonly used by other researchers.

Example: Hydrocarbon (HC) Emissions Data These data are the HC emissions at idling speed, in ppm, for automobiles of different years of manufacture. The data are a random sample of all automobiles tested at an

[4]The aim behind the choice of a **variance-stabilizing transformation** is to find a simple function f to apply to values y in a data set to create new values $y' = f(y)$ such that the variability of the values y' is not related to their mean value. For example, suppose that the values y are realizations from a Poisson distribution. Because for the Poisson distribution the variance is identical to the mean, the variance varies with the mean. However, if the simple variance-stabilizing transformation $y' = \sqrt{y}$ is applied, the sampling variance will be independent of the mean. A few distributional examples are provided in the table below.

Distribution	Variance=g(mean)	Transformation $y' = f(y)$
Poisson	$\sigma^2 = \mu$	$y' = \sqrt{y}$
binomial	$\sigma^2 = \mu(1-\mu)$	$y' = \arcsin(\sqrt{(}y))$
lognormal	$\sigma^2 = \mu^2$	$y' = \log(y)$

Albuquerque shopping center. (It looks like we need to find some newer cars!)

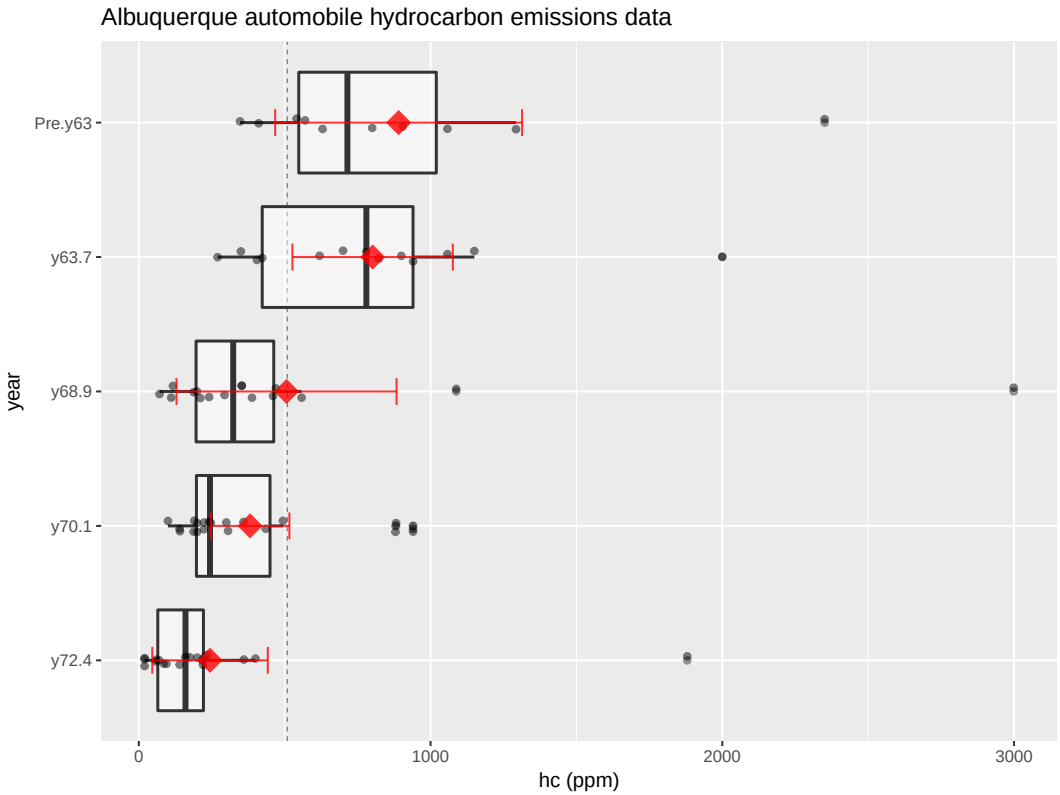
```
#### Example: Hydrocarbon (HC) Emissions Data
emis <- read.table(text="
 Pre-y63 y63-7 y68-9 y70-1 y72-4
    2351   620  1088   141   140
    1293   940   388   359   160
     541   350   111   247    20
    1058   700   558   940    20
     411  1150   294   882   223
     570  2000   211   494    60
     800   823   460   306    20
     630  1058   470   200    95
     905   423   353   100   360
     347   900    71   300    70
      NA   405   241   223   220
      NA   780  2999   190   400
      NA   270   199   140   217
      NA    NA   188   880    58
      NA    NA   353   200   235
      NA    NA   117   223  1880
      NA    NA    NA   188   200
      NA    NA    NA   435   175
      NA    NA    NA   940    85
      NA    NA    NA   241    NA
", header=TRUE)
#emis

# convert to long format
emis.long <- melt(emis,
              variable.name = "year",
              value.name = "hc",
              na.rm = TRUE
            )
## No id variables; using all as measure variables
# naming variables manually, the variable.name and value.name not working 11/2012
names(emis.long) <- c("year", "hc")
# summary of each year
by(emis.long$hc, emis.long$year, summary)
## emis.long$year: Pre.y63
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   347.0   548.2   715.0   890.6  1019.8  2351.0
## ------------------------------------------------------------
## emis.long$year: y63.7
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   270.0   423.0   780.0   801.5   940.0  2000.0
## ------------------------------------------------------------
## emis.long$year: y68.9
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##    71.0   196.2   323.5   506.3   462.5  2999.0
## ------------------------------------------------------
## emis.long$year: y70.1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   100.0   197.5   244.0   381.4   449.8   940.0
## ------------------------------------------------------
## emis.long$year: y72.4
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    20.0    65.0   160.0   244.1   221.5  1880.0
```

```
# IQR and sd of each year
by(emis.long$hc, emis.long$year, function(X) { c(IQR(X), sd(X), length(X)) })
```

```
## emis.long$year: Pre.y63
## [1] 471.5000 591.5673  10.0000
## ------------------------------------------------------
## emis.long$year: y63.7
## [1] 517.0000 454.9285  13.0000
## ------------------------------------------------------
## emis.long$year: y68.9
## [1] 266.2500 707.8026  16.0000
## ------------------------------------------------------
## emis.long$year: y70.1
## [1] 252.2500 287.8864  20.0000
## ------------------------------------------------------
## emis.long$year: y72.4
## [1] 156.5000 410.7866  19.0000
```

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(emis.long, aes(x = year, y = hc))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(yintercept = mean(emis.long$hc),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Albuquerque automobile hydrocarbon emissions data") + ylab("hc (ppm)")
# to reverse order that years print, so oldest is first on top
p <- p + scale_x_discrete(limits = rev(levels(emis.long$year)) )
p <- p + coord_flip()
p <- p + theme(legend.position="none")
print(p)
```



The standard ANOVA shows significant differences among the mean HC emissions. However, the standard ANOVA is inappropriate because the distributions are extremely skewed to the right due to presence of outliers in each sample.

```
fit.e <- aov(hc ~ year, data = emis.long)
summary(fit.e)
```

```
##             Df   Sum Sq Mean Sq F value  Pr(>F)
## year         4  4226834 1056709   4.343 0.00331 **
## Residuals   73 17759968  243287
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fit.e
```

```
## Call:
##    aov(formula = hc ~ year, data = emis.long)
##
## Terms:
##                     year Residuals
## Sum of Squares   4226834  17759968
## Deg. of Freedom        4        73
##
## Residual standard error: 493.2416
## Estimated effects may be unbalanced
```

The boxplots show that the typical HC emissions appear to decrease as

the age of car increases (the simplest description). Although the spread in the samples, as measured by the IQR, also decreases as age increases, I am more comfortable with the KW ANOVA, in part because the KW analysis is not too sensitive to differences in spreads among samples. This point is elaborated upon later. As described earlier, the KW ANOVA is essentially an ANOVA based on the ranks. I give below the ANOVA based on ranks and the output from the KW procedure. They give similar p-values, and lead to the conclusion that there are significant differences among the population median HC emissions. A simple description is that the population median emission tends to decrease with the age of the car. You should follow up this analysis with Mann-Whitney multiple comparisons.

```
# ANOVA of rank, for illustration that this is similar to what KW is doing
fit.er <- aov(rank(hc) ~ year, data = emis.long)
summary(fit.er)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## year         4  16329    4082   12.85 5.74e-08 ***
## Residuals   73  23200     318
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fit.er
```

```
## Call:
##    aov(formula = rank(hc) ~ year, data = emis.long)
##
## Terms:
##                     year Residuals
## Sum of Squares  16329.32  23199.68
## Deg. of Freedom        4        73
##
## Residual standard error: 17.82705
## Estimated effects may be unbalanced
```

```
# KW ANOVA
fit.ek <- kruskal.test(hc ~ year, data = emis.long)
fit.ek
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  hc by year
## Kruskal-Wallis chi-squared = 31.808, df = 4, p-value =
## 2.093e-06
```

It is common to transform the data to a log scale when the spread increases as the median or mean increases.

```
# log scale
emis.long$loghc <- log(emis.long$hc)
# summary of each year
by(emis.long$loghc, emis.long$year, summary)
```

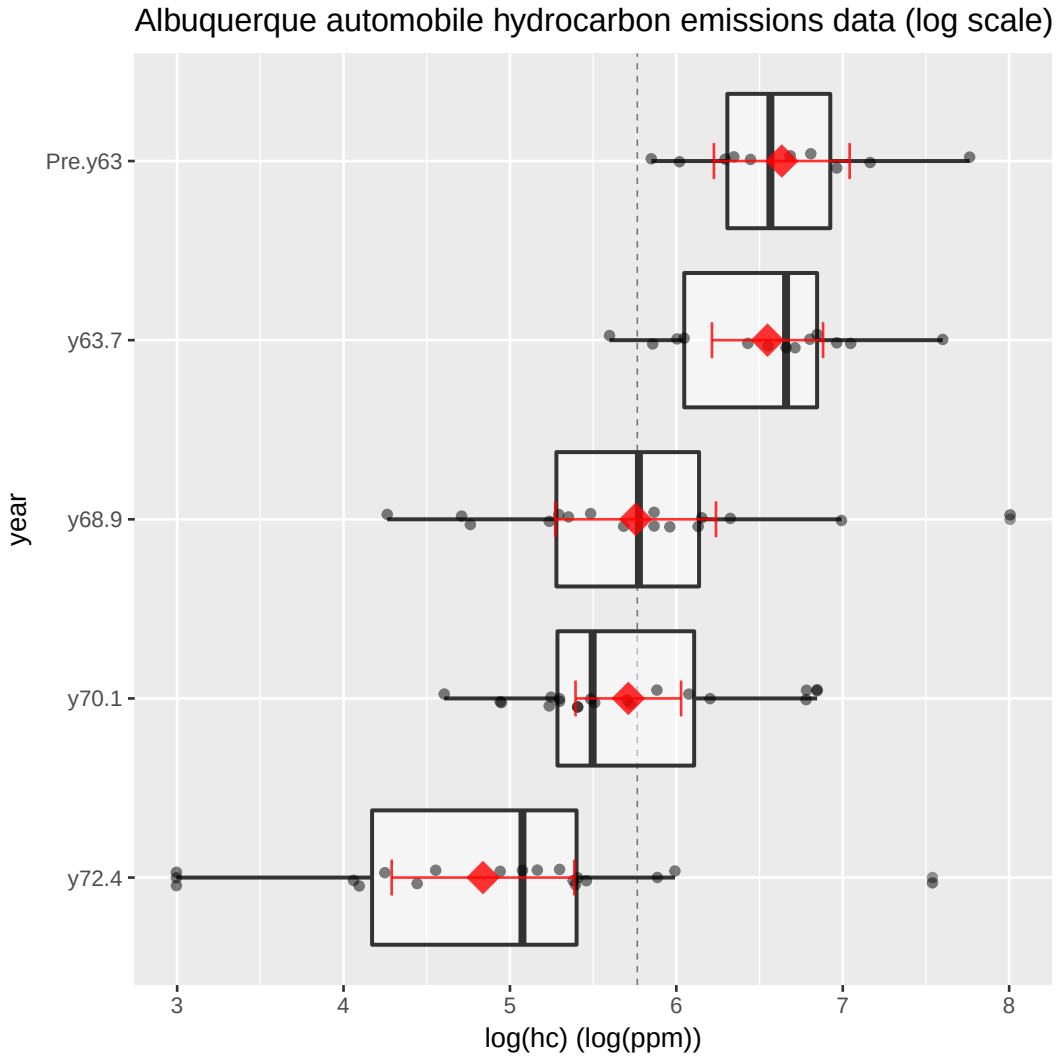
```
## emis.long$year: Pre.y63
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.849   6.306   6.565   6.634   6.925   7.763
## ------------------------------------------------------
## emis.long$year: y63.7
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.598   6.047   6.659   6.548   6.846   7.601
## ------------------------------------------------------
## emis.long$year: y68.9
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.263   5.279   5.775   5.755   6.137   8.006
## ------------------------------------------------------
## emis.long$year: y70.1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.605   5.285   5.497   5.711   6.107   6.846
## ------------------------------------------------------
## emis.long$year: y72.4
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.996   4.171   5.075   4.838   5.400   7.539
```

```
# IQR and sd of each year
by(emis.long$loghc, emis.long$year, function(X) { c(IQR(X), sd(X), length(X)) })
```

```
## emis.long$year: Pre.y63
## [1]  0.6186119  0.5702081 10.0000000
## ------------------------------------------------------
## emis.long$year: y63.7
## [1]  0.7985077  0.5524878 13.0000000
## ------------------------------------------------------
## emis.long$year: y68.9
## [1]  0.8575139  0.9061709 16.0000000
## ------------------------------------------------------
## emis.long$year: y70.1
## [1]  0.8216494  0.6775933 20.0000000
## ------------------------------------------------------
## emis.long$year: y72.4
## [1]  1.228980  1.138882 19.000000
```

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(emis.long, aes(x = year, y = loghc))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(yintercept = mean(emis.long$loghc),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
```

```
# points for observed data
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Albuquerque automobile hydrocarbon emissions data (log scale)")
p <- p + ylab("log(hc) (log(ppm))")
# to reverse order that years print, so oldest is first on top
p <- p + scale_x_discrete(limits = rev(levels(emis.long$year)) )
p <- p + coord_flip()
p <- p + theme(legend.position="none")
print(p)
```



After transformation, the samples have roughly the same spread (IQR and s) and shape. The transformation does not completely eliminate the outliers. However, I am more comfortable with a standard ANOVA on this scale than

with the original data. A difficulty here is that the ANOVA is comparing population mean log HC emission (so interpretations are on the log ppm scale, instead of the natural ppm scale). Summaries for the ANOVA on the log hydrocarbon emissions levels are given below.

```
# ANOVA of rank, for illustration that this is similar to what KW is doing
fit.le <- aov(loghc ~ year, data = emis.long)
summary(fit.le)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## year         4  31.90   7.974   11.42 2.98e-07 ***
## Residuals   73  50.98   0.698
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fit.le
```

```
## Call:
##    aov(formula = loghc ~ year, data = emis.long)
##
## Terms:
##                     year Residuals
## Sum of Squares  31.89510  50.97679
## Deg. of Freedom        4        73
##
## Residual standard error: 0.8356508
## Estimated effects may be unbalanced
```

```
# KW ANOVA -- same conclusions as original scale, since based on ranks
fit.lek <- kruskal.test(loghc ~ year, data = emis.long)
fit.lek
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  loghc by year
## Kruskal-Wallis chi-squared = 31.808, df = 4, p-value =
## 2.093e-06
```

The boxplot of the log-transformed data reinforces the reasonableness of the original KW analysis. Why? The log-transformed distributions have fairly similar shapes and spreads, so a KW analysis on these data is sensible. The ranks for the original and log-transformed data are identical, so the KW analyses on the log-transformed data and the original data must lead to the same conclusions. This suggests that the KW ANOVA is not overly sensitive to differences in spreads among the samples.

There are two reasonable analyses here: the standard ANOVA using log HC

emissions, and the KW analysis of the original data. The first analysis gives a comparison of mean log HC emissions. The second involves a comparison of median HC emissions. A statistician would present both analyses to the scientist who collected the data to make a decision on which was more meaningful (independently of the results[5]!). Multiple comparisons would be performed relative to the selected analysis (t-tests for ANOVA or WMW-tests for KW ANOVA).

Example: Hodgkin's Disease Study Plasma bradykininogen levels were measured in normal subjects, in patients with active Hodgkin's disease, and in patients with inactive Hodgkin's disease. The globulin bradykininogen is the precursor substance for bradykinin, which is thought to be a chemical mediator of inflammation. The data (in micrograms of bradykininogen per milliliter of plasma) are displayed below. The three samples are denoted by **nc** for normal controls, **ahd** for active Hodgkin's disease patients, and **ihd** for inactive Hodgkin's disease patients.

The medical investigators wanted to know if the three samples differed in their bradykininogen levels. Carry out the statistical analysis you consider to be most appropriate, and state your conclusions to this question.

Read in the data, look at summaries on the original scale, and create a plot. Also, look at summaries on the log scale and create a plot.

```
#### Example: Hodgkin's Disease Study
hd <- read.table(text="
    nc   ahd    ihd
  5.37  3.96   5.37
  5.80  3.04  10.60
  4.70  5.28   5.02
  5.70  3.40  14.30
  3.40  4.10   9.90
  8.60  3.61   4.27
  7.48  6.16   5.75
  5.77  3.22   5.03
  7.15  7.48   5.74
  6.49  3.87   7.85
  4.09  4.27   6.82
  5.94  4.05   7.90
  6.38  2.40   8.36
```

[5]It is unethical to choose a method based on the results it gives.

```
  9.24  5.81   5.72
  5.66  4.29   6.00
  4.53  2.77   4.75
  6.51  4.40   5.83
  7.00    NA   7.30
  6.20    NA   7.52
  7.04    NA   5.32
  4.82    NA   6.05
  6.73    NA   5.68
  5.26    NA   7.57
    NA    NA   5.68
    NA    NA   8.91
    NA    NA   5.39
    NA    NA   4.40
    NA    NA   7.13
", header=TRUE)
#hd

# convert to long format
hd.long <- melt(hd,
              variable.name = "patient",
              value.name = "level",
              na.rm = TRUE
            )
## No id variables; using all as measure variables
# naming variables manually, the variable.name and value.name not working 11/2012
names(hd.long) <- c("patient", "level")
# summary of each patient
by(hd.long$level, hd.long$patient, summary)









# IQR and sd of each patient
by(hd.long$level, hd.long$patient, function(X) { c(IQR(X), sd(X), length(X)) })
```

```
## ------------------------------------------------------
## hd.long$patient: ihd
## [1]  2.25500  2.17647 28.00000
```

```
# log scale
hd.long$loglevel <- log(hd.long$level)
# summary of each patient
by(hd.long$loglevel, hd.long$patient, summary)
```

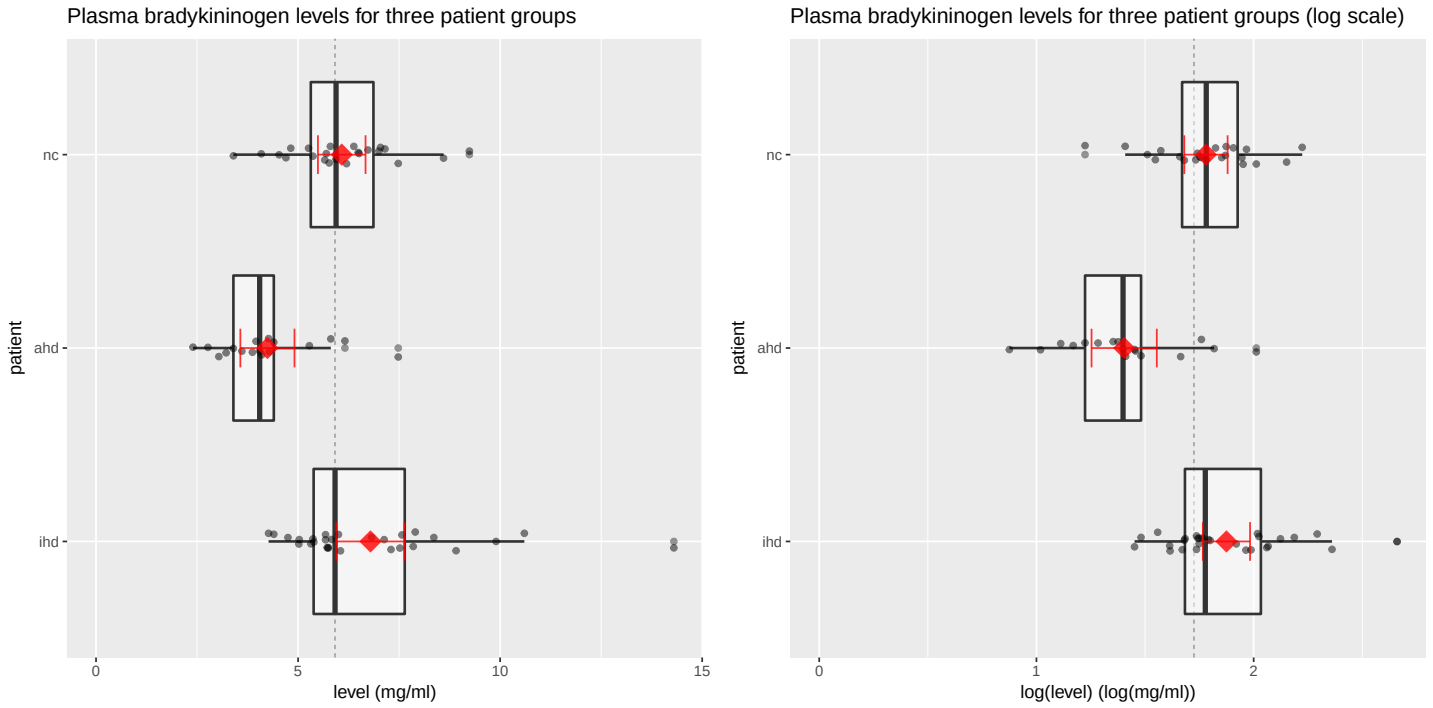
```
## hd.long$patient: nc
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.224   1.670   1.782   1.780   1.926   2.224
## ------------------------------------------------------
## hd.long$patient: ahd
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.8755  1.2238  1.3987  1.4039  1.4816  2.0122
## ------------------------------------------------------
## hd.long$patient: ihd
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.452   1.684   1.777   1.875   2.033   2.660
```

```
# IQR and sd of each patient
by(hd.long$loglevel, hd.long$patient, function(X) { c(IQR(X), sd(X), length(X)) })
```

```
## hd.long$patient: nc
## [1]  0.2557632  0.2303249 23.0000000
## ------------------------------------------------------
## hd.long$patient: ahd
## [1]  0.2578291  0.2920705 17.0000000
## ------------------------------------------------------
## hd.long$patient: ihd
## [1]  0.3496572  0.2802656 28.0000000
```

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(hd.long, aes(x = patient, y = level))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(yintercept = mean(hd.long$level),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Plasma bradykininogen levels for three patient groups")
p <- p + ylab("level (mg/ml)")
# to reverse order that years print, so oldest is first on top
p <- p + scale_x_discrete(limits = rev(levels(hd.long$patient)) )
p <- p + ylim(c(0,max(hd.long$level)))
p <- p + coord_flip()
p <- p + theme(legend.position="none")
print(p)
```

```
## log scale
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(hd.long, aes(x = patient, y = loglevel))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(yintercept = mean(hd.long$loglevel),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p <- p + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p <- p + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      colour = "red", alpha = 0.8)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, colour = "red", alpha = 0.8)
p <- p + labs(title = "Plasma bradykininogen levels for three patient groups (log scale)")
p <- p + ylab("log(level) (log(mg/ml))")
# to reverse order that years print, so oldest is first on top
p <- p + scale_x_discrete(limits = rev(levels(hd.long$patient)) )
p <- p + ylim(c(0,max(hd.long$loglevel)))
p <- p + coord_flip()
p <- p + theme(legend.position="none")
print(p)
```



Although the spread (IQR, s) in the *ihd* sample is somewhat greater than the spread in the other samples, the presence of skewness and outliers in the

boxplots is a greater concern regarding the use of the classical ANOVA. The shapes and spreads in the three samples are roughly identical, so a Kruskal-Wallis nonparametric ANOVA appears ideal. As a sidelight, I transformed plasma levels to a log scale to reduce the skewness and eliminate the outliers. The boxplots of the transformed data show reasonable symmetry across groups, but outliers are still present. I will stick with the Kruskal-Wallis ANOVA (although it would not be much of a problem to use the classical ANOVA on transformed data).

Let η_{nc} = population median plasma level for normal controls, η_{ahd} = population median plasma level for active Hodgkin's disease patients, and η_{ihd} = population median plasma level for inactive Hodgkin's disease patients. The KW test of $H_0 : \eta_{nc} = \eta_{ahd} = \eta_{ihd}$ versus H_A : not H_0 is highly significant (p-value= 0.00003), suggesting differences among the population median plasma levels. The Kruskal-Wallis ANOVA summary is given below.

```
# KW ANOVA
fit.h <- kruskal.test(level ~ patient, data = hd.long)
fit.h
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  level by patient
## Kruskal-Wallis chi-squared = 20.566, df = 2, p-value =
## 3.421e-05
```

I followed up the KW ANOVA with Bonferroni comparisons of the samples, using the Mann-Whitney two sample procedure. There are three comparisons, so an overall FER of 0.05 is achieved by doing the individual tests at the 0.05/3=0.0167 level. Alternatively, you can use 98.33% CI for differences in population medians.

```
# with continuity correction in the normal approximation for the p-value
wilcox.test(hd$nc , hd$ahd, conf.int=TRUE, conf.level = 0.9833)
```

```
## Warning in wilcox.test.default(hd$nc, hd$ahd, conf.int = TRUE, conf.level = 0.9833):  canno
compute exact p-value with ties
## Warning in wilcox.test.default(hd$nc, hd$ahd, conf.int = TRUE, conf.level = 0.9833):  canno
compute exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
```

```
## data:  hd$nc and hd$ahd
## W = 329, p-value = 0.0002735
## alternative hypothesis: true location shift is not equal to 0
## 98.33 percent confidence interval:
##  0.8599458 2.9000789
## sample estimates:
## difference in location
##               1.910067
```

```
wilcox.test(hd$nc , hd$ihd, conf.int=TRUE, conf.level = 0.9833)
```

```
## Warning in wilcox.test.default(hd$nc, hd$ihd, conf.int = TRUE, conf.level = 0.9833):  canno
compute exact p-value with ties
## Warning in wilcox.test.default(hd$nc, hd$ihd, conf.int = TRUE, conf.level = 0.9833):  canno
compute exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  hd$nc and hd$ihd
## W = 276.5, p-value = 0.3943
## alternative hypothesis: true location shift is not equal to 0
## 98.33 percent confidence interval:
##  -1.5600478  0.6800262
## sample estimates:
## difference in location
##             -0.3413932
```

```
wilcox.test(hd$ahd, hd$ihd, conf.int=TRUE, conf.level = 0.9833)
```

```
## Warning in wilcox.test.default(hd$ahd, hd$ihd, conf.int = TRUE, conf.level = 0.9833):
cannot compute exact p-value with ties
## Warning in wilcox.test.default(hd$ahd, hd$ihd, conf.int = TRUE, conf.level = 0.9833):
cannot compute exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  hd$ahd and hd$ihd
## W = 56, p-value = 2.143e-05
## alternative hypothesis: true location shift is not equal to 0
## 98.33 percent confidence interval:
##  -3.500059 -1.319957
## sample estimates:
## difference in location
##              -2.146666
```

The only comparison with a p-value greater than 0.0167 involved the **nc** and **ihd** samples. The comparison leads to two groups, and is consistent with what we see in the boxplots.

ahd **nc** **ihd**

You have sufficient evidence to conclude that the plasma bradykininogen levels

for active Hodgkin's disease patients (ahd) is lower than the population median levels for normal controls (nc) and for patients with inactive Hodgkin's disease (ihd). You do not have sufficient evidence to conclude that the population median levels for normal controls (nc) and for patients with inactive Hodgkin's disease (ihd) are different. The CIs give an indication of size of differences in the population medians.

6.5.3 Planned Comparisons

Bonferroni multiple comparisons are generally preferred to Fisher's least significant difference approach. Fisher's method does not control the familywise error rate and produces too many spurious significant differences (claims of significant differences that are due solely to chance variation and not to actual differences in population means). However, Bonferroni's method is usually very conservative when a large number of comparisons is performed — large differences in sample means are needed to claim significance. A way to reduce this conservatism is to avoid doing all possible comparisons. Instead, one should, when possible, decide *a priori* (before looking at the data) which comparisons are of primary interest, and then perform only those comparisons.

For example, suppose a medical study compares five new treatments with a control (a six group problem). The medical investigator may not be interested in all 15 possible comparisons, but only in which of the five treatments differ on average from the control. Rather than performing the 15 comparisons, each at the say $0.05/15 = 0.0033$ level, she could examine the five comparisons of interest at the $0.05/5 = 0.01$ level. By deciding beforehand which comparisons are of interest, she can justify using a 0.01 level for the comparisons, instead of the more conservative 0.0033 level needed when doing all possible comparisons.

To illustrate this idea, consider the KW analysis of HC emissions. We saw that there are significant differences among the population median HC emissions. Given that the samples have a natural ordering

Sample	Year of manufacture
1	Pre-1963
2	63 – 67
3	68 – 69
4	70 – 71
5	72 – 74

you may primarily be interested in whether the population medians for cars manufactured in consecutive samples are identical. That is, you may be primarily interested in the following 4 comparisons:

Pre-1963	vs	63 – 67
63 – 67	vs	68 – 69
68 – 69	vs	70 – 71
70 – 71	vs	72 – 74

A Bonferroni analysis would carry out each comparison at the $0.05/4 = 0.0125$ level versus the $0.05/10 = 0.005$ level when all comparisons are done.

The following output was obtained for doing these four comparisons, based on Wilcoxon-Mann-Whitney two-sample tests (why?[6]). Two-year groups are claimed to be different if the p-value is 0.0125 or below, or equivalently, if a 98.75% CI for the difference in population medians does not contain zero.

```
#### Planned Comparisons
# with continuity correction in the normal approximation for the p-value
wilcox.test(emis$y63.7, emis$Pre.y63, conf.int=TRUE, conf.level = 0.9875)
```

```
## Warning in wilcox.test.default(emis$y63.7, emis$Pre.y63, conf.int = TRUE, :  cannot compute
exact p-value with ties
## Warning in wilcox.test.default(emis$y63.7, emis$Pre.y63, conf.int = TRUE, :  cannot compute
exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  emis$y63.7 and emis$Pre.y63
## W = 61.5, p-value = 0.8524
## alternative hypothesis: true location shift is not equal to 0
## 98.75 percent confidence interval:
##  -530.0001  428.0000
```

[6]The ANOVA is the multi-sample analog to the two-sample t-test for the mean, and the KW ANOVA is the multi-sample analog to the WMW two-sample test for the median. Thus, we follow up a KW ANOVA with WMW two-sample tests at the chosen multiple comparison adjusted error rate.

```
## sample estimates:
## difference in location
##               -15.4763
```

```
wilcox.test(emis$y68.9, emis$y63.7  , conf.int=TRUE, conf.level = 0.9875)
```

```
## Warning in wilcox.test.default(emis$y68.9, emis$y63.7, conf.int = TRUE, :  cannot compute
exact p-value with ties
## Warning in wilcox.test.default(emis$y68.9, emis$y63.7, conf.int = TRUE, :  cannot compute
exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  emis$y68.9 and emis$y63.7
## W = 43, p-value = 0.007968
## alternative hypothesis: true location shift is not equal to 0
## 98.75 percent confidence interval:
##  -708.99999  -51.99998
## sample estimates:
## difference in location
##              -397.4227
```

```
wilcox.test(emis$y70.1, emis$y68.9  , conf.int=TRUE, conf.level = 0.9875)
```

```
## Warning in wilcox.test.default(emis$y70.1, emis$y68.9, conf.int = TRUE, :  cannot compute
exact p-value with ties
## Warning in wilcox.test.default(emis$y70.1, emis$y68.9, conf.int = TRUE, :  cannot compute
exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  emis$y70.1 and emis$y68.9
## W = 156, p-value = 0.9112
## alternative hypothesis: true location shift is not equal to 0
## 98.75 percent confidence interval:
##  -206.0001  171.0000
## sample estimates:
## difference in location
##              -10.99997
```

```
wilcox.test(emis$y72.4, emis$y70.1  , conf.int=TRUE, conf.level = 0.9875)
```

```
## Warning in wilcox.test.default(emis$y72.4, emis$y70.1, conf.int = TRUE, :  cannot compute
exact p-value with ties
## Warning in wilcox.test.default(emis$y72.4, emis$y70.1, conf.int = TRUE, :  cannot compute
exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  emis$y72.4 and emis$y70.1
## W = 92.5, p-value = 0.006384
## alternative hypothesis: true location shift is not equal to 0
## 98.75 percent confidence interval:
```

```
##  -285.999962   -6.000058
## sample estimates:
## difference in location
##                   -130
```

There are significant differences between the 1963-67 and 1968-69 samples, and between the 1970-71 and 1972-74 samples. You are 98.75% confident that the population median HC emissions for 1963-67 year cars is between 52 and 708.8 ppm greater than the population median for 1968-69 cars. Similarly, you are 98.75% confident that the population median HC emissions for 1970-71 year cars is between 6.1 and 285.9 ppm greater than the population median for 1972-74 cars. Overall, you are 95% confident among the four pairwise comparisons that you have not declared a difference significant when it isn't.

6.5.4 Two final ANOVA comments

It is not uncommon for researchers to combine data from groups not found to be significantly different. This is not, in general, a good practice. Just because you do not have sufficient evidence to show differences does not imply that you should treat the groups as if they are the same!

If the data distributions do not substantially deviate from normality, but the spreads are different across samples, you might consider the standard ANOVA followed with multiple comparisons using two-sample tests based on Satterthwaite's approximation.

6.6 Permutation tests

Permutation tests[7] are a subset of non-parametric statistics. The basic premise is to use only the assumption that it is possible that all of the treatment groups are equivalent, and that every member of them is the same before sampling began (i.e., the position in the group to which they belong is not differentiable from other position before the positions are filled). From this, one can calculate

[7] http://en.wikipedia.org/wiki/Resampling_(statistics)

a statistic and then see to what extent this statistic is special by seeing how likely it would be if the group assignments had been jumbled.

A permutation test (also called a randomization test, re-randomization test, or an exact test) is a type of statistical significance test in which the distribution of the test statistic under the null hypothesis is obtained by calculating all possible values of the test statistic under **rearrangements of the labels** on the observed data points. In other words, the method by which treatments are allocated to subjects in an experimental design is mirrored in the analysis of that design. If the labels are exchangeable under the null hypothesis, then the resulting tests yield exact significance levels. Confidence intervals can then be derived from the tests. The theory has evolved from the works of R.A. Fisher and E.J.G. Pitman in the 1930s.

Let's illustrate the basic idea of a permutation test using the Meteorites example. Suppose we have two groups Uwet and Walker whose sample means are $\bar{Y}_{\mathrm{U}}$ and $\bar{Y}_{\mathrm{W}}$, and that we want to test, at 5% significance level, whether they come from the same distribution. Let $n_{\mathrm{U}} = 9$ and $n_{\mathrm{W}} = 10$ be the sample size corresponding to each group. The permutation test is designed to determine whether the observed difference between the sample means is large enough to reject the null hypothesis $H_0 : \mu_{\mathrm{U}} = \mu_{\mathrm{W}}$, that the two groups have identical means.

The test proceeds as follows. First, the difference in means between the two samples is calculated: this is the observed value of the test statistic, $T_{(\mathrm{obs})}$. Then the observations of groups Uwet and Walker are pooled.

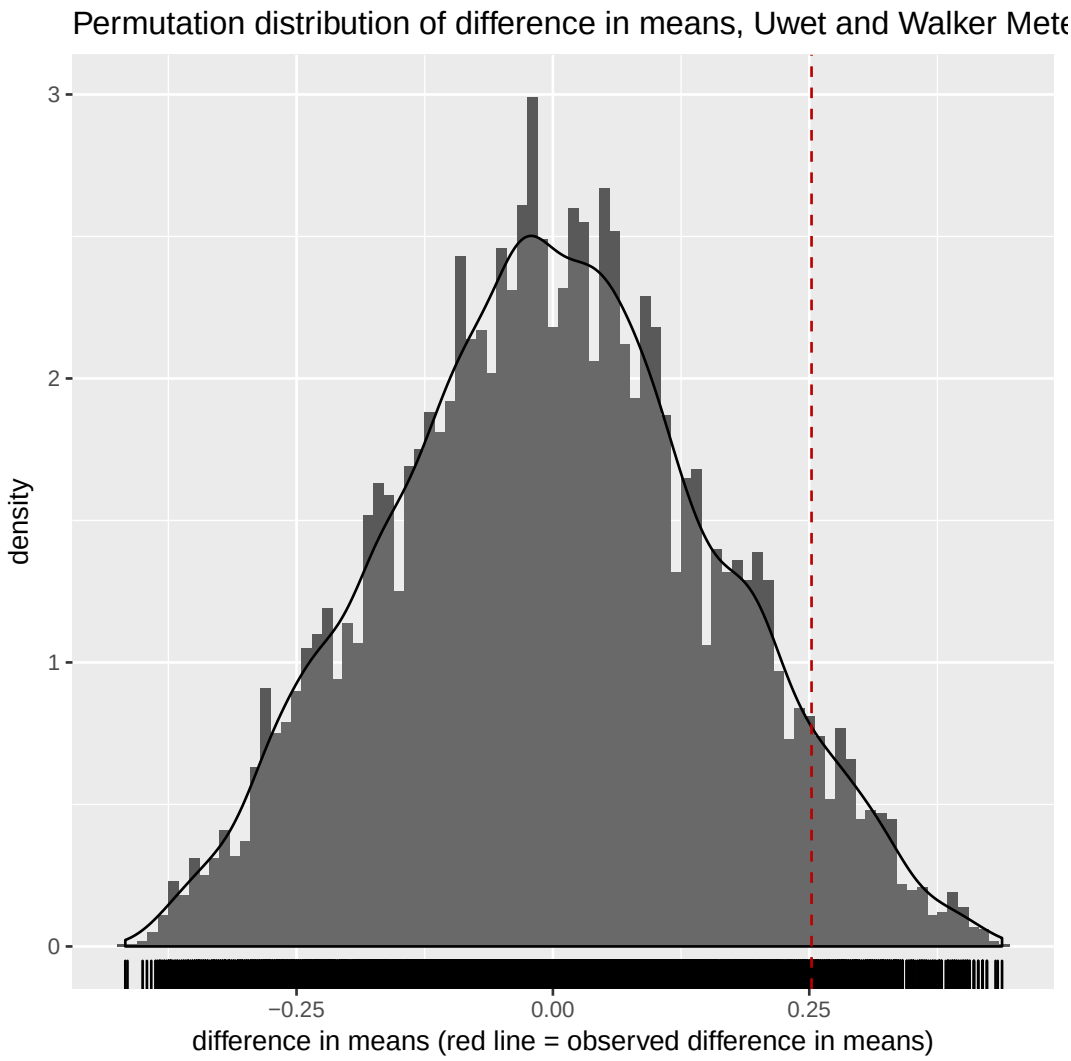
```
#### Permutation tests
# Calculated the observed difference in means
        # met.long includes both Uwet and Walker groups
Tobs <- mean(met.long[(met.long$site == "Uwet"  ), 2]) -
        mean(met.long[(met.long$site == "Walker"), 2])
Tobs
```

```
## [1] 0.2522222
```

Next, the difference in sample means is calculated and recorded for every possible way of dividing these pooled values into two groups of size $n_{\mathrm{U}} = 9$ and $n_{\mathrm{W}} = 10$ (i.e., for every permutation of the group labels Uwet and Walker). The set of these calculated differences is the exact distribution of possible differences under the null hypothesis that group label does not matter.

This exact distribution can be approximated by drawing a large number of random permutations.

```
# Plan:
#   Initialize a vector in which to store the R number of difference of means.
#   Calculate R differences in means for R permutations, storing the results.
#     Note that there are prod(1:19) = 10^17 total permutations,
#       but the R repetitions will serve as a good approximation.
#   Plot the permutation null distribution with an indication of the Tobs.

# R = a large number of repetitions
R <- 1e4
# initialize the vector of difference of means from the permutations
Tperm <- rep(NA, R)
# For each of R repetitions, permute the Uwet and Walker labels,
#   calculate the difference of means with the permuted labels,
#   and store the result in the i.R'th position of Tperm.
for (i.R in 1:R) {
  # permutation of 19 = 9+10 integers 1, 2, ..., 19
  ind.perm <- sample.int(nrow(met.long))
  # identify as "TRUE" numbers 1, ..., 9 (the number of Uwet labels)
  lab.U <- (ind.perm <= sum(met.long$site == "Uwet"))                    #£
  # identify as "TRUE" numbers 10, ..., 19 (the number of Walker labels)
  #   that is, all the non-Uwet labels
  lab.W <- !lab.U

  # calculate the difference in means and store in Tperm at index i.R
  Tperm[i.R] <- mean(met.long[lab.U, 2]) - mean(met.long[lab.W, 2])
}

# Plot the permutation null distribution with an indication of the Tobs.
dat <- data.frame(Tperm)

library(ggplot2)
p <- ggplot(dat, aes(x = Tperm))
#p <- p + scale_x_continuous(limits=c(-20,+20))
p <- p + geom_histogram(aes(y=..density..), binwidth=0.01)
p <- p + geom_density(alpha=0.1, fill="white")
p <- p + geom_rug()
# vertical line at Tobs
p <- p + geom_vline(aes(xintercept=Tobs), colour="#BB0000", linetype="dashed")
p <- p + labs(title = "Permutation distribution of difference in means, Uwet and Walker Meteor
p <- p + xlab("difference in means (red line = observed difference in means)")
print(p)
```



Notice the contrast in this permutation distribution of the difference in means from a normal distribution.

The one-sided p-value of the test is calculated as the proportion of sampled permutations where the difference in means was at least as extreme as $T_{(\text{obs})}$. The two-sided p-value of the test is calculated as the proportion of sampled permutations where the absolute difference was at least as extreme as $|T_{(\text{obs})}|$.

```
# Calculate a two-sided p-value.
p.upper <- sum((Tperm >= abs(Tobs))) / R
p.upper
## [1] 0.0623
p.lower <- sum((Tperm <= -abs(Tobs))) / R
p.lower
## [1] 0.0604
p.twosided <- p.lower + p.upper
p.twosided
```

```
## [1] 0.1227
```

Note that the two-sided p-value of 0.1227 is consistent, in this case, with the two-sample t-test p-values of 0.1134 (pooled) and 0.1290 (Satterthwaite), but different from 0.0497 (WMW). The permutation is a comparison of means **without** the normality assumption, though requires that the observations are exchangable between populations under H_0.

If the only purpose of the test is reject or not reject the null hypothesis, we can as an alternative sort the recorded differences, and then observe if $T_{(\text{obs})}$ is contained within the middle 95% of them. If it is not, we reject the hypothesis of equal means at the 5% significance level.

6.6.1 Linear model permutation tests in R

The `coin` package provides an implementation of a general framework for conditional inference procedures commonly known as permutation tests. In the help on `?"coin-package"` search for `location` to find tests for the means or medians of populations (such as `oneway_test()`). Other packages of note include `perm` and `exactRankTests` (`lmPerm` is defunct).

Below I calculate the standard t-test for the Meteorite data using `t.test()` and `lm()`, then compare that with `oneway_test()` and what we calculated using our calculation of the permutation test.

```
# standard two-sample t-test with equal variances
t.summary <- t.test(cool ~ site, data = met.long, var.equal = TRUE)
t.summary
```

```
##
##  Two Sample t-test
##
## data:  cool by site
## t = 1.6689, df = 17, p-value = 0.1134
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.0666266  0.5710710
## sample estimates:
##   mean in group Uwet mean in group Walker
##            0.4522222            0.2000000
```

```
# linear model form of t-test, "siteWalker" has estimate, se, t-stat, and p-value
lm.summary <- lm(cool ~ site, data = met.long)
```

```r
summary(lm.summary)
```

```
##
## Call:
## lm(formula = cool ~ site, data = met.long)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.2922 -0.1961 -0.1600  0.0250  0.7478
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.4522     0.1096   4.125 0.000708 ***
## siteWalker   -0.2522     0.1511  -1.669 0.113438
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3289 on 17 degrees of freedom
## Multiple R-squared:  0.1408,Adjusted R-squared:  0.09024
## F-statistic: 2.785 on 1 and 17 DF,  p-value: 0.1134
```

```r
# permutation test version
library(coin)
```

```
## Loading required package:  survival
```

```r
# Fisher-Pitman permutation test
oneway.summary <- oneway_test(cool ~ site, data = met.long)
oneway.summary
```

```
##
##  Asymptotic Two-Sample Fisher-Pitman Permutation Test
##
## data:  cool by site (Uwet, Walker)
## Z = 1.5919, p-value = 0.1114
## alternative hypothesis: true mu is not equal to 0
```

```r
  # # examples of extracting values from coins S4 class objects
  # expectation(oneway.summary)
  # covariance(oneway.summary)
  # pvalue(oneway.summary)
  # confint(oneway.summary)
pvalue(oneway.summary)
```

```
## [1] 0.1114144
```

The permutation test gives a p-value of 0.1114 which is close to our manually calculated permuatation p-value of 0.1227.

For the emisions data, below we compare the ANOVA results (assuming normality) with a permutation test without distributional assumptions.

```r
fit.e <- aov(hc ~ year, data = emis.long)
summary(fit.e)
```

```
##             Df   Sum Sq Mean Sq F value  Pr(>F)
## year         4  4226834 1056709   4.343 0.00331 **
## Residuals   73 17759968  243287
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
library(coin)
# Fisher-Pitman permutation test
oneway.summary <- oneway_test(hc ~ year, data = emis.long)
oneway.summary
```

```
##
##  Asymptotic K-Sample Fisher-Pitman Permutation Test
##
## data:  hc by
##   year (Pre.y63, y63.7, y68.9, y70.1, y72.4)
## chi-squared = 14.803, df = 4, p-value = 0.005128
```

Thus the permutation test of the ANOVA hypothesis on means rejects the null hypothesis of all equal means. A followup set of pairwise tests can be done by looping over pairs of factors.

First we list the factor levels ordered by their medians, the ordering by medians is helpful at the end when the results of the pairwise comparisons are given.

```
# these are the levels of the factor, ordered by their medians
fac.lev <- levels(reorder(levels(emis.long$year)
                          , -as.numeric(by(emis.long$loghc, emis.long$year, median)))
                  )
fac.lev
```

```
## [1] "y63.7"   "Pre.y63" "y68.9"   "y70.1"   "y72.4"
```

Create a matrix to store pairwise comparison p-values, then loop over all pairs of groups and perform a two-sample permutation test. Store the p-value for each test in the matrix.

```
# create a matrix to store pairwise comparison p-values
mc.pval <- matrix(NA
                  , nrow = length(fac.lev)
                  , ncol = length(fac.lev)
                  , dimnames = list(fac.lev, fac.lev))
# diag is always 1, no group differs from itself
diag(mc.pval) <- 1
mc.pval
```

```
##         y63.7 Pre.y63 y68.9 y70.1 y72.4
## y63.7       1      NA    NA    NA    NA
## Pre.y63    NA       1    NA    NA    NA
## y68.9      NA      NA     1    NA    NA
```

```
## y70.1      NA      NA    NA     1    NA
## y72.4      NA      NA    NA    NA     1
```

```
# loop over all pairs of factor levels, perform two-sample test,
#   and store p-value in matrix
for (i1 in 1:(length(fac.lev) - 1)) {
  for (i2 in (i1 + 1):length(fac.lev)) {
    ## DEBUG - to make sure the indexing is working, you can print them:
    # print(cat(i1, i2))

    library(coin)
    # Fisher-Pitman permutation test
    oneway.summary <- oneway_test(hc ~ year, data = subset(emis.long, (year == fac.lev[i1] | y

    # put p-value in matrix
    mc.pval[i1, i2] <- pvalue(oneway.summary)
    mc.pval[i2, i1] <- mc.pval[i1, i2]
  }
}

# p-values
mc.pval
```

```
##               y63.7     Pre.y63     y68.9       y70.1       y72.4
## y63.7   1.000000000 0.676572596 0.1993877 0.004273746 0.002185513
## Pre.y63 0.676572596 1.000000000 0.1611790 0.005319987 0.003379156
## y68.9   0.199387725 0.161179041 1.0000000 0.468455149 0.177187250
## y70.1   0.004273746 0.005319987 0.4684551 1.000000000 0.227517382
## y72.4   0.002185513 0.003379156 0.1771873 0.227517382 1.000000000
```

Summarize the results of the pairwise comparisons. Groups with a common letter are not statistically different.

```
# summary of pairwise comparisons
#   threshold is Bonferroni-corrected alpha=0.05 / 10
library(multcompView)
multcompLetters( mc.pval
                , compare = "<"
                , threshold = 0.05 / choose(length(fac.lev), 2)
                , Letters = letters
                , reversed = FALSE)
```

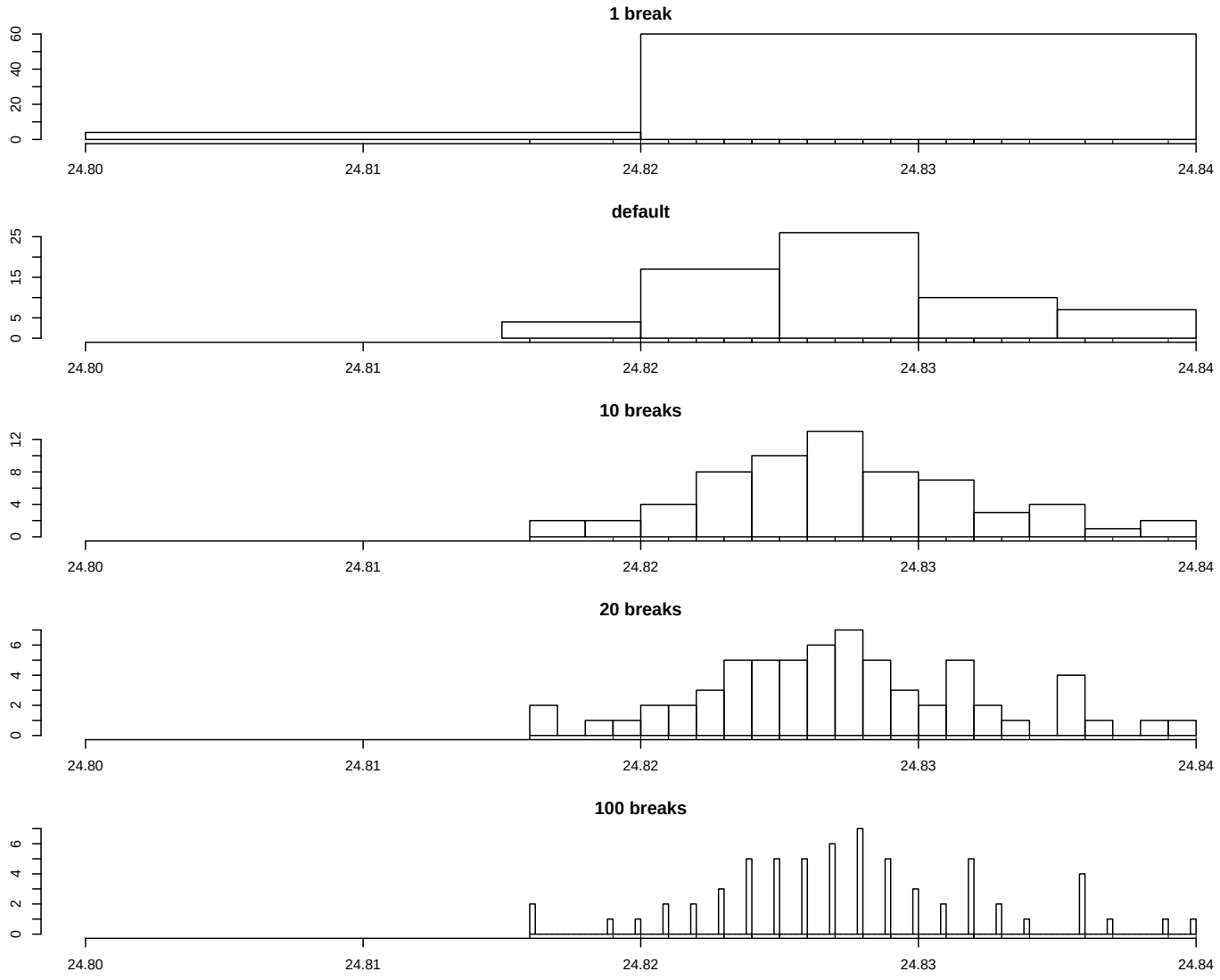
```
##   y63.7 Pre.y63   y68.9   y70.1   y72.4
##     "a"    "ab"   "abc"    "bc"     "c"
```

6.7 Density estimation

Density estimation is like a histogram: It is a method for visualizing the shape of a univariate distribution (there are methods for doing multivariate density estimation as well, but we will ignore those for the time being). In fact, I snuck in density estimation in the first chapter and have been using it all along! Let's experiment with Newcombe's speed-of-light data (excluding the two outliers).

Consider the shape of the histogram for different numbers of bins.

```
#### Density estimation
# include time ranks 3 and above, that is, remove the lowest two values
time2 <- time[(rank(time) >= 3)]

  old.par <- par(no.readonly = TRUE)
  # make smaller margins
  par(mfrow=c(5,1), mar=c(3,2,2,1), oma=c(1,1,1,1))

hist(time2, breaks=1    , main="1 break"   , xlim=c(24.80,24.84), xlab=""); rug(time2)
hist(time2,               main="default"   , xlim=c(24.80,24.84), xlab=""); rug(time2)
hist(time2, breaks=10   , main="10 breaks" , xlim=c(24.80,24.84), xlab=""); rug(time2)
hist(time2, breaks=20   , main="20 breaks" , xlim=c(24.80,24.84), xlab=""); rug(time2)
hist(time2, breaks=100  , main="100 breaks", xlim=c(24.80,24.84), xlab=""); rug(time2)

  # restore par() settings
  par(old.par)
```



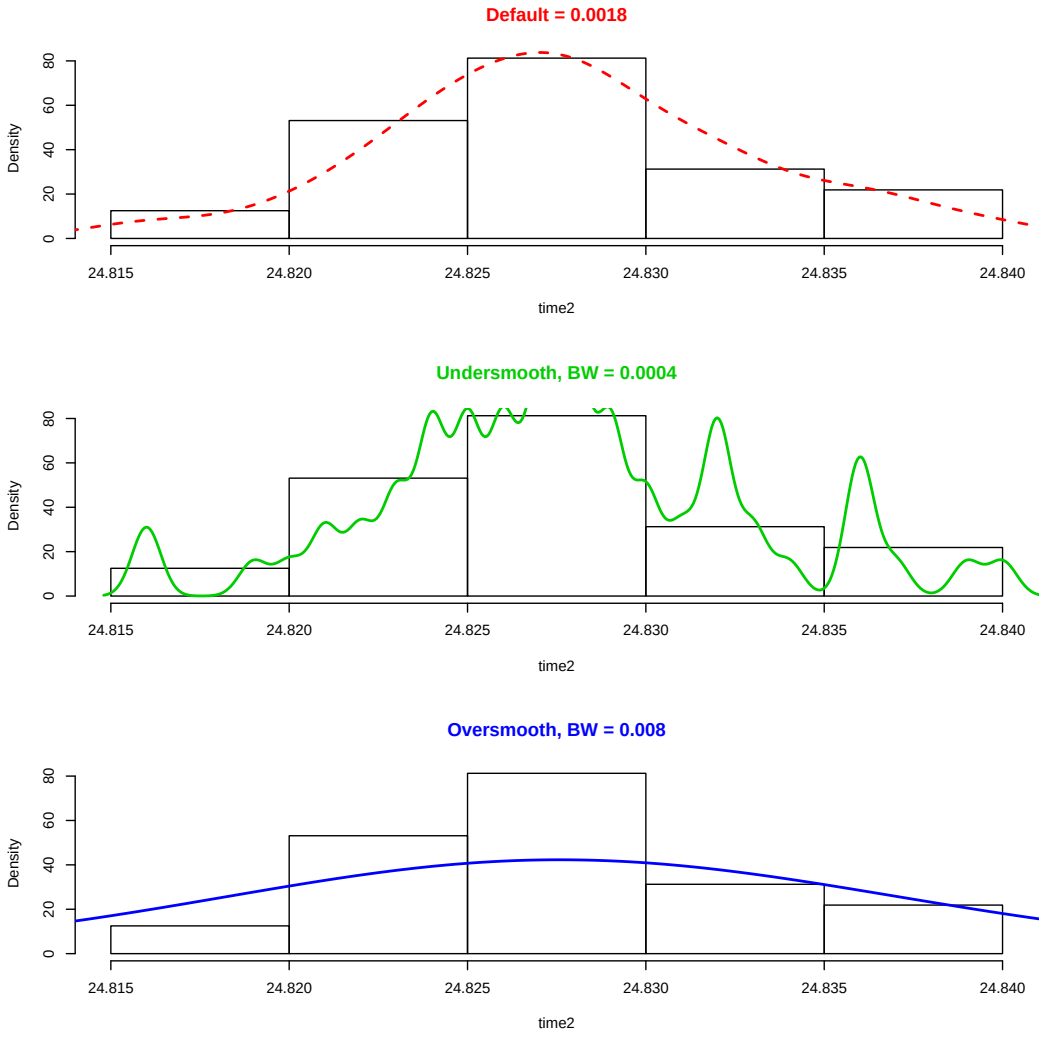
Notice that we are starting to see more and more bins that include only a single observation (or multiple observations at the precision of measurement). Taken to its extreme, this type of exercise gives in some sense a "perfect" fit to the data but is useless as an estimator of shape.

On the other hand, it is obvious that a single bin would also be completely useless. So we try in some sense to find a middle ground between these two extremes: "Oversmoothing" by using only one bin and "undersmooting" by using too many. This same paradigm occurs for density estimation, in which the amount of smoothing is determined by a quantity called the bandwidth. By default, R uses an optimal (in some sense) choice of bandwidth.

We've already used the `density()` function to provide a smooth curve to our histograms. So far, we've taken the default "bandwidth". Let's see what

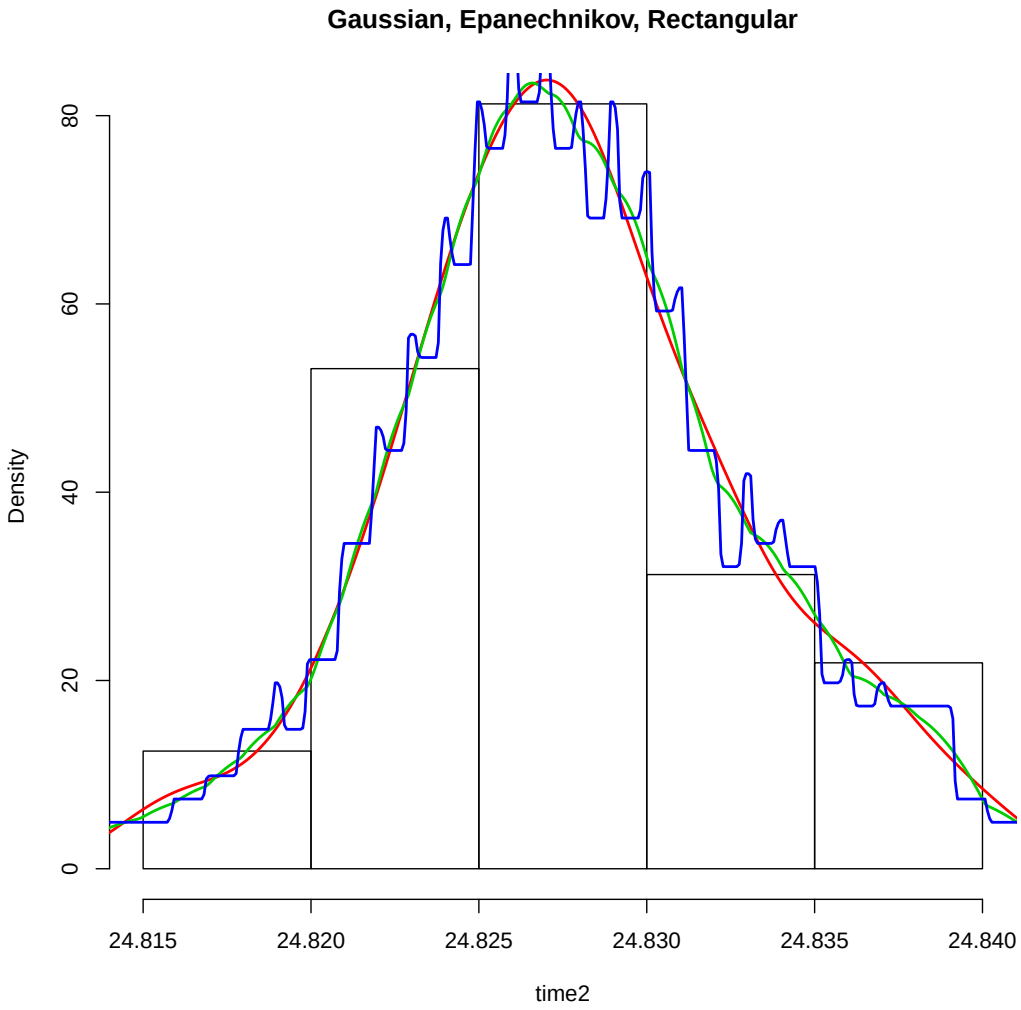
happens when we use different bandwidths.

```
par(mfrow=c(3,1))

# prob=TRUE scales the y-axis like a density function, total area = 1
hist(time2, prob=TRUE, main="")
# apply a density function, store the result
den = density(time2)
# plot density line over histogram
lines(den, col=2, lty=2, lwd=2)
# extract the bandwidth (bw) from the density line
b = round(den$bw, 4)
title(main=paste("Default =", b), col.main=2)

# undersmooth
hist(time2, prob=TRUE, main="")
lines(density(time2, bw=0.0004), col=3, lwd=2)
text(17.5, .35, "", col=3, cex=1.4)
title(main=paste("Undersmooth, BW = 0.0004"), col.main=3)

# oversmooth
hist(time2, prob=TRUE, main="")
lines(density(time2, bw=0.008), col=4, lwd=2)
title(main=paste("Oversmooth, BW = 0.008"), col.main=4)
```



The other determining factor is the kernel, which is the shape each individual point takes before all the shapes are added up for a final density line. While the choice of bandwidth is very important, the choice of kernel is not. Choosing a kernel with hard edges (such as "rect") will result in jagged artifacts, so smoother kernels are often preferred.

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

hist(time2, prob=TRUE, main="")

# default kernel is Gaussian ("Normal")
lines(density(time2)            , col=2, lty=1, lwd=2)
lines(density(time2, ker="epan"), col=3, lty=1, lwd=2)
lines(density(time2, ker="rect"), col=4, lty=1, lwd=2)
title(main="Gaussian, Epanechnikov, Rectangular")

# other kernels include: "triangular", "biweight", "cosine", "optcosine"
```


Gaussian, Epanechnikov, Rectangular
Density
0
20
40
60
80
24.815
24.820
24.825
24.830
24.835
24.840
time2

Chapter 7

Categorical Data Analysis

Contents

```
## Warning in file(filename, "r", encoding = encoding):  cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding):  cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

select the appropriate statistical method to compare summaries from categorical variables.

assess the assumptions of one-way and two-way tests of proportions and independence.

decide whether the proportions between populations are different, including in stratified and cross-sectional studies.

recommend action based on a hypothesis test.

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

1. organize knowledge.

5. define parameters of interest and hypotheses in words and notation.

6. summarize data visually, numerically, and descriptively.

8. use statistical software.

12. make evidence-based decisions.

7.1 Categorical data

When the response variable is categorical, the interesting questions are often about the probability of one possible outcome versus another, and whether these probabilities depend on other variables (continuous or categorical).

Example: Titanic The sinking of the Titanic is a famous event, and new books are still being published about it. Many well-known facts — from the proportions of first-class passengers to the "women and children first" policy, and the fact that policy was not entirely successful in saving the women and children in the third class — are reflected in the survival rates for various classes of passenger. The source provides a data set recording class, sex, age, and survival status for each person on board of the Titanic, and is based on data originally collected by the British Board of Trade[1].

```
# The Titanic dataset is a 4-dimensional table: Class, Sex, Age, Survived
library(datasets)
data(Titanic)

Titanic
```

```
## , , Age = Child, Survived = No
##
##       Sex
## Class  Male Female
##   1st     0      0
##   2nd     0      0
##   3rd    35     17
##   Crew    0      0
##
## , , Age = Adult, Survived = No
##
##       Sex
## Class  Male Female
##   1st   118      4
##   2nd   154     13
##   3rd   387     89
##   Crew  670      3
##
```

[1]British Board of Trade (1990), Report on the Loss of the "Titanic" (S.S.). British Board of Trade Inquiry Report (reprint). Gloucester, UK: Allan Sutton Publishing. Note that there is not complete agreement among primary sources as to the exact numbers on board, rescued, or lost.

```
## , , Age = Child, Survived = Yes
##
##       Sex
## Class  Male Female
##   1st     5      1
##   2nd    11     13
##   3rd    13     14
##   Crew    0      0
##
## , , Age = Adult, Survived = Yes
##
##       Sex
## Class  Male Female
##   1st    57    140
##   2nd    14     80
##   3rd    75     76
##   Crew  192     20
```

```
# reshape into long data.frame
library(reshape2)
df.titanic <- melt(Titanic, value.name = "Freq")
df.titanic
```

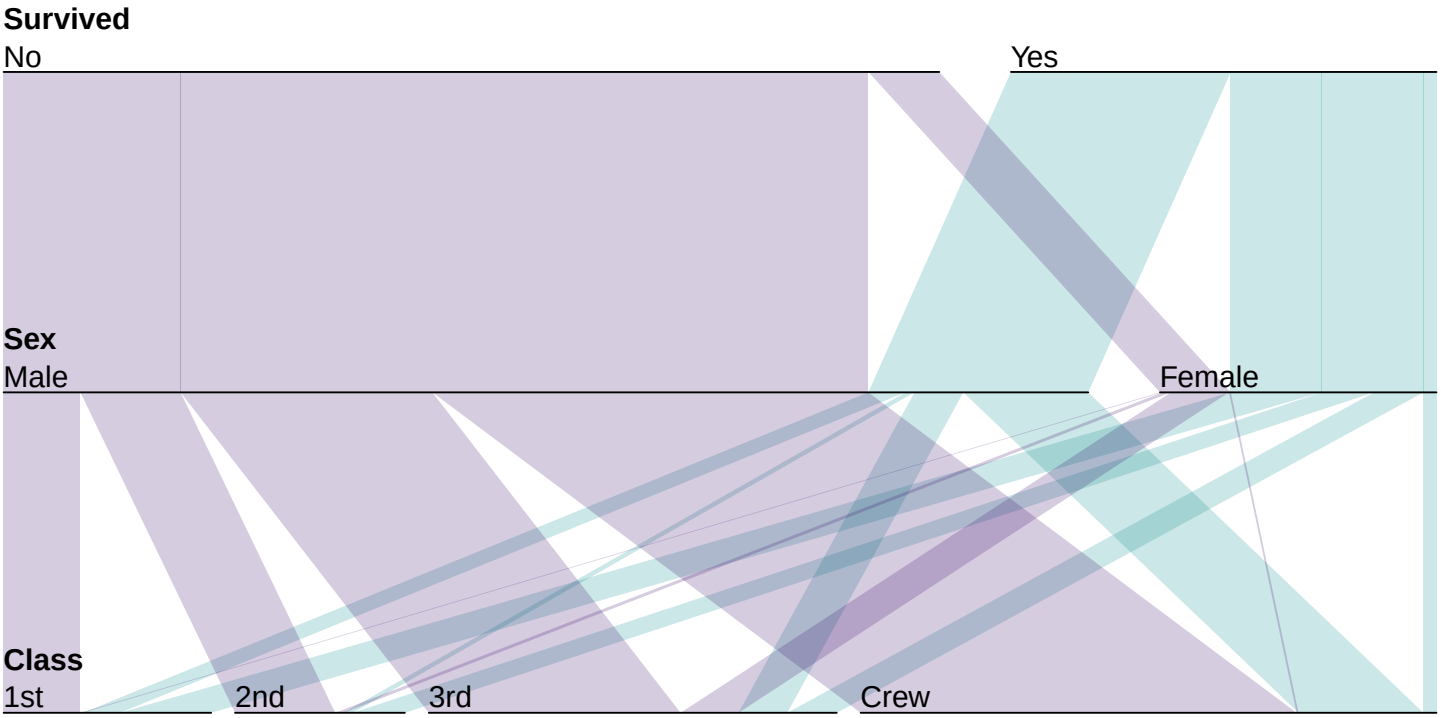
```
##    Class    Sex   Age Survived Freq
## 1    1st   Male Child       No    0
## 2    2nd   Male Child       No    0
## 3    3rd   Male Child       No   35
## 4   Crew   Male Child       No    0
## 5    1st Female Child       No    0
## 6    2nd Female Child       No    0
## 7    3rd Female Child       No   17
## 8   Crew Female Child       No    0
## 9    1st   Male Adult       No  118
## 10   2nd   Male Adult       No  154
## 11   3rd   Male Adult       No  387
## 12  Crew   Male Adult       No  670
## 13   1st Female Adult       No    4
## 14   2nd Female Adult       No   13
## 15   3rd Female Adult       No   89
## 16  Crew Female Adult       No    3
## 17   1st   Male Child      Yes    5
## 18   2nd   Male Child      Yes   11
## 19   3rd   Male Child      Yes   13
## 20  Crew   Male Child      Yes    0
## 21   1st Female Child      Yes    1
## 22   2nd Female Child      Yes   13
## 23   3rd Female Child      Yes   14
## 24  Crew Female Child      Yes    0
## 25   1st   Male Adult      Yes   57
## 26   2nd   Male Adult      Yes   14
```

```
## 27   3rd   Male Adult      Yes   75
## 28  Crew   Male Adult      Yes  192
## 29   1st Female Adult      Yes  140
## 30   2nd Female Adult      Yes   80
## 31   3rd Female Adult      Yes   76
## 32  Crew Female Adult      Yes   20
```

```
# Total number of people
sum(df.titanic$Freq)
```

```
## [1] 2201
```

```
# create colors based on survival
df.titanic$Color <- ifelse(df.titanic$Survived == "Yes", "#008888", "#330066")

# subset only the adults (since there were so few children)
df.titanic.adult <- subset(df.titanic, Age == "Adult")

# see R code on website for function parallelset()
# see help for with(), it allows temporary direct reference to columns in a data.frame
#   otherwise, we'd need to specify df.titanic.adult$Survived, ...
with(df.titanic.adult
   , parallelset(Survived, Sex, Class, freq = Freq, col = Color, alpha=0.2)
    )
```



There are many questions that can be asked of this dataset. How likely were people to survive such a ship sinking in cold water? Is the survival proportion dependent on sex, class, or age, or a combination of these? How different is the survival proportions for 1st class females versus 3rd class males?

7.2 Single Proportion Problems

Assume that you are interested in estimating the proportion p of individuals in a population with a certain characteristic or attribute based on a random or representative sample of size n from the population. The **sample proportion** $\hat{p}$ =(# with attribute in the sample)/n is the best guess for p based on the data.

This is the simplest **categorical data** problem. Each response falls into one of two exclusive and exhaustive categories, called "success" and "failure". Individuals with the attribute of interest are in the success category. The rest fall into the failure category. Knowledge of the **population proportion** p of successes characterizes the distribution across both categories because the population proportion of failures is $1 - p$.

As an aside, note that the probability that a randomly selected individual has the attribute of interest is the population proportion p with the attribute, so the terms population proportion and probability can be used interchangeably with random sampling.

7.2.1 A CI for p

A two-sided CI for p is a range of plausible values for the unknown population proportion p, based on the observed data. To compute a two-sided CI for p:

1. Specify the confidence level as the percent $100(1-\alpha)\%$ and solve for the error rate α of the CI.
2. Compute $z_{\text{crit}} = z_{0.5\alpha}$ (i.e., area under the standard normal curve to the left and to the right of z_{crit} are $1 - 0.5\alpha$ and 0.5α, respectively). `qnorm(1-0.05/2)=1.96`.
3. The $100(1-\alpha)\%$ CI for p has endpoints $L = \hat{p} - z_{\text{crit}}SE$ and $U = \hat{p} + z_{\text{crit}}SE$, respectively, where the "CI standard error" is

$$SE = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}.$$

The CI is often written as $\hat{p} \pm z_{\text{crit}}SE$.

Reminder of CI interpretation. The CI is determined once the confidence level is specified and the data are collected. Prior to collecting data, the CI is unknown and can be viewed as random because it will depend on the actual sample selected. Different samples give different CIs. The "confidence" in, say, the 95% CI (which has a 0.05 or 5% error rate) can be interpreted as follows. If you repeatedly sample the population and construct 95% CIs for p, then 95% of the intervals will contain p, whereas 5% (the error rate) will not. The CI you get from your data either covers p, or it does not.

The length of the CI

$$U - L = 2z_{\text{crit}}SE$$

depends on the accuracy of the estimate $\hat{p}$, as measured by the standard error SE. For a given $\hat{p}$, this standard error decreases as the sample size n increases, yielding a narrower CI. For a fixed sample size, this standard error is maximized at $\hat{p} = 0.5$, and decreases as $\hat{p}$ moves towards either 0 or 1. In essence, sample proportions near 0 or 1 give narrower CIs for p. However, the normal approximation used in the CI construction is less reliable for extreme values of $\hat{p}$.

■ CLICKER Qs — CI for proportions STT.08.01.010 ■

Example: Tamper resistant packaging The 1983 Tylenol poisoning episode highlighted the desirability of using tamper-resistant packaging. The article "Tamper Resistant Packaging: Is it Really?" (Packaging Engineering, June 1983) reported the results of a survey on consumer attitudes towards tamper-resistant packaging. A sample of 270 consumers was asked the question: "Would you be willing to pay extra for tamper resistant packaging?" The number of yes respondents was 189. Construct a 95% CI for the proportion p of all consumers who were willing in 1983 to pay extra for such packaging.

Here $n = 270$ and $\hat{p} = 189/270 = 0.700$. The critical value for a 95% CI for

p is $z_{0.025} = 1.96$. The CI standard error is given by

$$SE = \sqrt{\frac{0.7 \times 0.3}{270}} = 0.028,$$

so $z_{\text{crit}} SE = 1.96 \times 0.028 = 0.055$. The 95% CI for p is 0.700 ± 0.055. You are 95% confident that the proportion of consumers willing to pay extra for better packaging is between 0.645 and 0.755. (Willing to pay how much extra?)

Appropriateness of the CI

The standard CI is based on a **large-sample** standard normal approximation to

$$z = \frac{\hat{p} - p}{SE}.$$

A simple rule of thumb requires $n\hat{p} \geq 5$ and $n(1 - \hat{p}) \geq 5$ for the method to be suitable. Given that $n\hat{p}$ and $n(1-\hat{p})$ are the observed numbers of successes and failures, you should have at least 5 of each to apply the large-sample CI.

In the packaging example, $n\hat{p} = 270 \times (0.700) = 189$ (the number who support the new packaging) and $n(1 - \hat{p}) = 270 \times (0.300) = 81$ (the number who oppose) both exceed 5. The normal approximation is appropriate here.

7.2.2 Hypothesis Tests on Proportions

The following example is typical of questions that can be answered using a hypothesis test for a population proportion.

Example Environmental problems associated with leaded gasolines are well-known. Many motorists have tampered with the emission control devices on their cars to save money by purchasing leaded rather than unleaded gasoline. A *Los Angeles Times* article on March 17, 1984 reported that 15% of all California motorists have engaged in emissions tampering. A random sample of 200 cars from L.A. county was obtained, and the emissions devices on 21 are

found to be tampered with. Does this suggest that the proportion of cars in L.A. county with tampered devices differs from the statewide proportion?

Two-Sided Hypothesis Test for p

Suppose you are interested in whether the population proportion p is equal to a prespecified value, say p_0. This question can be formulated as a two-sided test. To carry out the test:

1. Define the null hypothesis $H_0 : p = p_0$ and alternative hypothesis $H_A : p \neq p_0$.
2. Choose the size or significance level of the test, denoted by α.
3. Using the standard normal probability table, find the critical value z_{crit} such that the areas under the normal curve to the left and right of z_{crit} are $1 - 0.5\alpha$ and 0.5α, respectively. That is, $z_{\text{crit}} = z_{0.5\alpha}$.
4. Compute the test statistic (often to be labelled z_{obs})

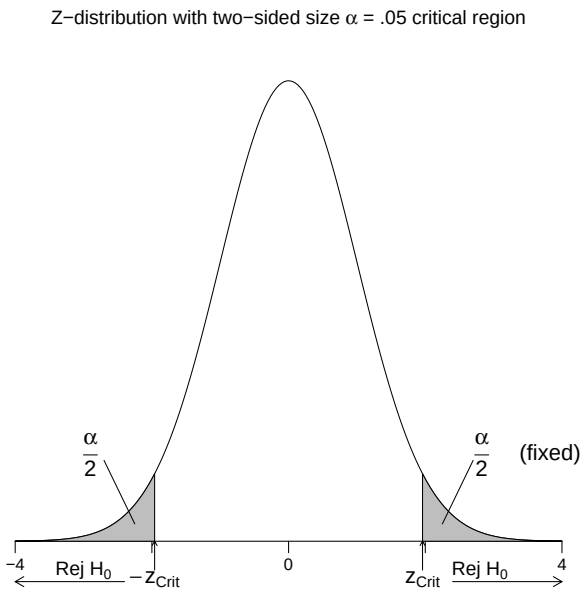
$$z_s = \frac{\hat{p} - p_0}{SE},$$

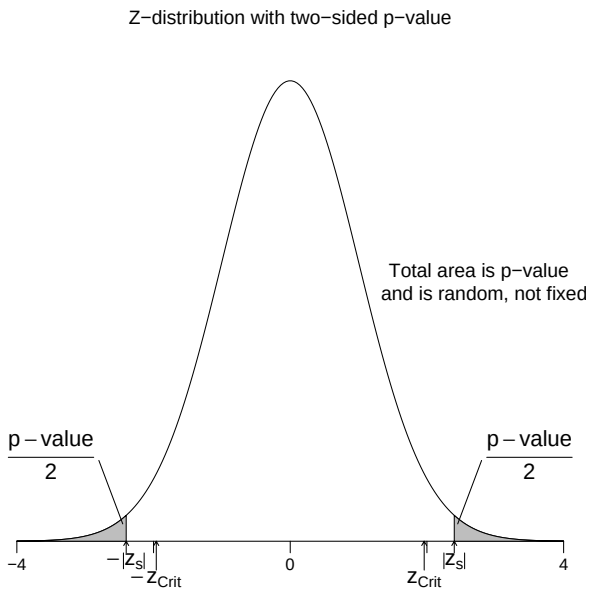
where the "test standard error" (based on the hypothesized value) is

$$SE = \sqrt{\frac{p_0(1 - p_0)}{n}}.$$

5. Reject H_0 in favor of H_A if $|z_{\text{obs}}| \geq z_{\text{crit}}$. Otherwise, do not reject H_0.

The rejection rule is easily understood visually. The area under the normal curve outside $\pm z_{\text{crit}}$ is the size α of the test. One-half of α is the area in each tail. You reject H_0 in favor of H_A if the test statistic exceeds $\pm z_{\text{crit}}$. This occurs when $\hat{p}$ is significantly different from p_0, as measured by the standardized distance z_{obs} between $\hat{p}$ and p_0.





■ CLICKER Qs — Test statistic STT.07.01.057 ■

7.2.3 The p-value for a two-sided test

To compute the p-value (not to be confused with the value of the proportion p) for a two-sided test:

1. Compute the test statistic $z_s = z_{\text{obs}}$.
2. Evaluate the area under the normal probability curve outside $\pm|z_s|$.

Recall that the null hypothesis for a size α test is rejected if and only if the p-value is less than or equal to α.

Example: Emissions data Each car in the target population (L.A. county) either has been tampered with (a success) or has not been tampered with (a failure). Let p = the proportion of cars in L.A. county with tampered emissions control devices. You want to test $H_0 : p = 0.15$ against $H_A : p \neq 0.15$ (here $p_0 = 0.15$). The critical value for a two-sided test of size $\alpha = 0.05$ is $z_{\text{crit}} = 1.96$.

The data are a sample of $n = 200$ cars. The sample proportion of cars that have been tampered with is $\hat{p} = 21/200 = 0.105$. The test statistic is

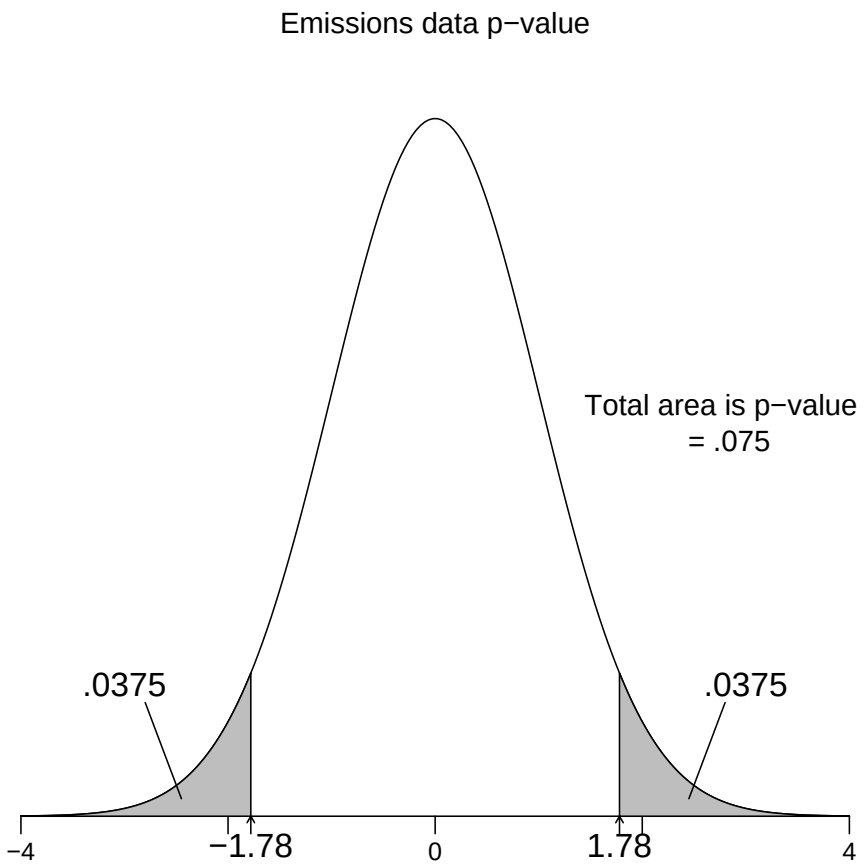
$$z_s = \frac{0.105 - 0.15}{0.02525} = -1.78,$$

where the test standard error satisfies

$$SE = \sqrt{\frac{0.15 \times 0.85}{200}} = 0.02525.$$

Given that $|z_s| = 1.78 < 1.96$, you have insufficient evidence to reject H_0 at the 5% level. That is, you have insufficient evidence to conclude that the proportion of cars in L.A. county that have been tampered with differs from the statewide proportion.

This decision is reinforced by the p-value calculation. The p-value is the area under the standard normal curve outside ± 1.78. This is $2 \times 0.0375 = 0.075$, which exceeds the test size of 0.05.



Remark The SE used in the test and CI are different. This implies that a hypothesis test and CI could potentially lead to different decisions. That is, a 95% CI for a population proportion might cover p_0 when the p-value for testing $H_0 : p = p_0$ is smaller than 0.05. This will happen, typically, only in cases where the decision is "borderline."

7.2.4 Appropriateness of Test

The z-test is based on a large-sample normal approximation, which works better for a given sample size when p_0 is closer to 0.5. The sample size needed for an accurate approximation increases dramatically the closer p_0 gets to 0 or to 1. A simple rule of thumb is that the test is appropriate when (the expected number

of successes) $np_0 \geq 5$ and (the expected number of failures) $n(1-p_0) \geq 5$.

In the emissions example, $np_0 = 200 \times (0.15) = 30$ and $n(1-p_0) = 200 \times (0.85) = 170$ exceed 5, so the normal approximation is appropriate.

7.2.5 R Implementation

```
#### Single Proportion Problems
# Approximate normal test for proportion, without Yates' continuity correction
prop.test(21, 200, p = 0.15, correct = FALSE)
```

```
##
##  1-sample proportions test without continuity correction
##
## data:  21 out of 200, null probability 0.15
## X-squared = 3.1765, df = 1, p-value = 0.07471
## alternative hypothesis: true p is not equal to 0.15
## 95 percent confidence interval:
##  0.06970749 0.15518032
## sample estimates:
##     p
## 0.105
```

```
# Approximate normal test for proportion, with Yates' continuity correction
#prop.test(21, 200, p = 0.15)
```

■ CLICKER Qs — Parachute null hypothesis STT.08.01.040 ■

■ CLICKER Qs — Parachute conclusion STT.08.01.050 ■

■ CLICKER Qs — Parachute p-value STT.08.01.060 ■

7.2.6 One-Sided Tests and One-Sided Confidence Bounds

The mechanics of tests on proportions are similar to tests on means, except we use a different test statistic and a different probability distribution for critical

values. This applies to one-sided and two-sided procedures. The example below illustrates a one-sided test and bound.

Example: brain hemispheres An article in the April 6, 1983 edition of *The Los Angeles Times* reported on a study of 53 learning-impaired youngsters at the Massachusetts General Hospital. The right side of the brain was found to be larger than the left side in 22 of the children. The proportion of the general population with brains having larger right sides is known to be 0.25. Does the data provide strong evidence for concluding, as the article claims, that the proportion of learning impaired youngsters with brains having larger right sides exceeds the proportion in the general population?

I will answer this question by computing a p-value for a one-sided test. Let p be the population proportion of learning disabled children with brains having larger right sides. I am interested in testing $H_0 : p = 0.25$ against $H_A : p > 0.25$ (here $p_0 = 0.25$).

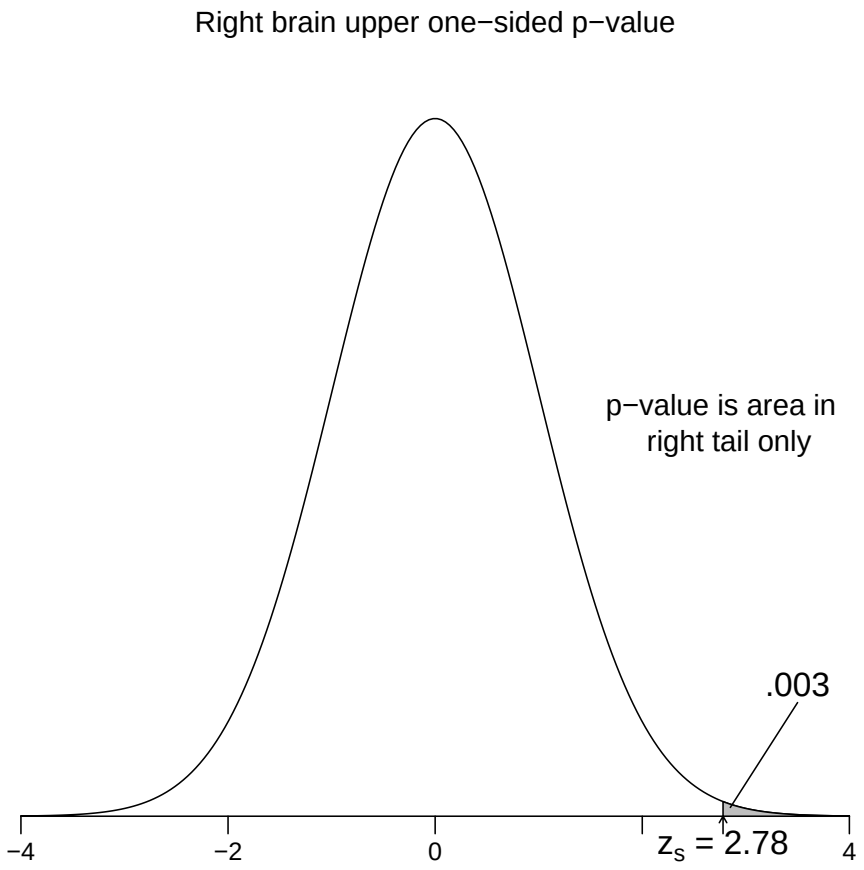
The proportion of children sampled with brains having larger right sides is $\hat{p} = 22/53 = 0.415$. The test statistic is

$$z_s = \frac{0.415 - 0.25}{0.0595} = 2.78,$$

where the test standard error satisfies

$$SE = \sqrt{\frac{0.25 \times 0.75}{53}} = 0.0595.$$

The p-value for an upper one-sided test is the area under the standard normal curve to the right of 2.78, which is approximately .003; see the picture below. I would reject H_0 in favor of H_A using any of the standard test levels, say 0.05 or 0.01. The newspaper's claim is reasonable.



A sensible next step in the analysis would be to compute a **lower confidence bound** $\hat{p} - z_{\text{crit}} SE$ for p. For illustration, consider a 95% bound. The CI standard error is

$$SE = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} = \sqrt{\frac{0.415 \times 0.585}{53}} = 0.0677.$$

The critical value for a one-sided 5% test is $z_{\text{crit}} = 1.645$, so a lower 95% bound on p is $0.415 - 1.645 \times 0.0677 = 0.304$. I am 95% confident that the population proportion of learning disabled children with brains having larger right sides is at least 0.304. Values of p smaller than 0.304 are not supported by the data.

You should verify that the sample size is sufficiently large to use the approximate methods in this example.

```r
#### Example: brain hemispheres
# Approximate normal test for proportion, without Yates' continuity correction
prop.test(22, 53, p = 0.25, alternative = "greater", correct = FALSE)
```

```
##
##  1-sample proportions test without continuity correction
##
## data:  22 out of 53, null probability 0.25
## X-squared = 7.7044, df = 1, p-value = 0.002754
## alternative hypothesis: true p is greater than 0.25
## 95 percent confidence interval:
##  0.3105487 1.0000000
## sample estimates:
##         p
## 0.4150943
```

7.2.7 Small Sample Procedures

Large sample tests and CIs for p should be interpreted with caution in small sized samples because the true error rate usually exceeds the assumed (nominal) value. For example, an assumed 95% CI, with a nominal error rate of 5%, may be only an 80% CI, with a 20% error rate. The large-sample CIs are usually overly optimistic (i.e., too narrow) when the sample size is too small to use the normal approximation.

Alan Agresti suggests the following method for a 95% CI. The standard method computes the sample proportion as $\hat{p} = x/n$ where x is the number of successes in the sample and n is the sample size. Agresti suggested using the estimated proportion $\tilde{p} = (x+2)/(n+4)$ with the standard error

$$SE = \sqrt{\frac{\tilde{p}(1-\tilde{p})}{n+4}},$$

in the "usual 95% interval" formula: $\tilde{p} \pm 1.96SE$. This appears odd, but amounts to adding two successes and two failures to the observed data, and then computing the standard CI.

This adjustment has little effect when n is large and $\hat{p}$ is not near either 0 or 1, as in the Tylenol example.

Example: swimming segregation This example is based on a case heard before the U.S. Supreme Court. A racially segregated swimming club was ordered to admit minority members. However, it is unclear whether the club has been following the spirit of the mandate. Historically, 85% of the white applicants were approved. Since the mandate, only 1 of 6 minority applicants has been approved. Is there evidence of continued discrimination?

I examine this issue by constructing a CI and a test for the probability p (or population proportion) that a minority applicant is approved. Before examining the question of primary interest, let me show that the two approximate CIs are very different, due to the small sample size. One minority applicant ($x = 1$) was approved out of $n = 6$ candidates, giving $\hat{p} = 1/6 = 0.167$.

A 95% large-sample CI for p is $(-0.14, 0.46)$. Since a negative proportion is not possible, the CI should be reported as $(0.00, 0.46)$. Agresti's 95% CI (based on 3 successes and 7 failures) is (0.02, 0.58). The big difference between the two intervals coupled with the negative lower endpoint on the standard CI suggests that the normal approximation used with the standard method is inappropriate. This view is reinforced by the rule-of-thumb calculation for using the standard interval. Agresti's CI is wider, which is consistent with my comment that the standard CI is too narrow in small samples. As a comparison, the exact 95% CI is $(0.004, 0.64)$, which agrees more closely with Agresti's interval.

I should emphasize that the exact CI is best to use, but is not available in all statistical packages, so methods based on approximations may be required, and if so, then Agresti's method is clearly better than the standard normal approximation in small sized samples.

Recall that the results of the asymptotic 95% CIs may disagree with the hypothesis test results. Exact methods will not contradict each other this way (neither do these asymptotic methods, usually).

```
#### Example: swimming segregation
## The prop.test() does an additional adjustment, so does not match precisely
##   the results in the above paragraphs

# Approximate normal test for proportion, without Yates' continuity correction
prop.test(1, 6, p = 0.85, correct = FALSE)$conf.int
```

```
## Warning in prop.test(1, 6, p = 0.85, correct = FALSE): Chi-squared approximation may be
incorrect
## [1] 0.03005337 0.56350282
## attr(,"conf.level")
## [1] 0.95
# Agresti's method
prop.test(1+2, 6+4, p = 0.85, correct = FALSE)$conf.int
## Warning in prop.test(1 + 2, 6 + 4, p = 0.85, correct = FALSE): Chi-squared approximation
may be incorrect
## [1] 0.1077913 0.6032219
## attr(,"conf.level")
## [1] 0.95
# Exact binomial test for proportion
binom.test(1, 6, p = 0.85)$conf.int
## [1] 0.004210745 0.641234579
## attr(,"conf.level")
## [1] 0.95
```

Returning to the problem, you might check for discrimination by testing $H_0 : p = 0.85$ against $H_A : p < 0.85$ using an **exact** test. The exact test p-value is 0.000 to three decimal places, and an exact upper bound for p is 0.582. What does this suggest to you?

```
# Exact binomial test for proportion
binom.test(1, 6, alternative = "less", p = 0.85)
##
##  Exact binomial test
##
## data:  1 and 6
## number of successes = 1, number of trials = 6, p-value =
## 0.0003987
## alternative hypothesis: true probability of success is less than 0.85
## 95 percent confidence interval:
##  0.0000000 0.5818034
## sample estimates:
## probability of success
##              0.1666667
```

7.3 Analyzing Raw Data

In most studies, your data will be stored in a spreadsheet with one observation per case or individual. For example, the data below give the individual responses to the applicants of the swim club.

```
#### Example: swimming segregation, raw data
## read.table options
# sep = default is any white space, but our strings contain a space,
#       so I changed this to a comma
# header = there are no column headers
# stringsAsFActors = default converts strings to factors, but I want them
#       to just be the plain character text
swim <- read.table(text="
not approved
not approved
not approved
approved
not approved
not approved
", sep = ",", header=FALSE, stringsAsFactors=FALSE)

# name the column
names(swim) <- c("application")
# show the structure of the data.frame
str(swim)
```

```
## 'data.frame': 6 obs. of  1 variable:
##  $ application: chr  "not approved" "not approved" "not approved" "approved" ...
```

```
# display the data.frame
swim
```

```
##    application
## 1 not approved
## 2 not approved
## 3 not approved
## 4     approved
## 5 not approved
## 6 not approved
```

The data were entered as alphabetic strings. We can use `table()` to count frequencies of categorical variables.

```
# count the frequency of each categorical variable
table(swim)
```

```
## swim
##     approved not approved
##            1            5
```

You can compute a CI and test for a proportion using raw data, provided the data column includes only two distinct values. The levels can be numerical or alphanumeric.

```
# use the counts from table() for input in binom.text()
# the help for binom.test() says x can be a vector of length 2
#   giving the numbers of successes and failures, respectively
#   that's exactly what table(swim) gave us
```

```
binom.test(table(swim), p = 0.85, alternative = "less")
```

```
##
##  Exact binomial test
##
## data:  table(swim)
## number of successes = 1, number of trials = 6, p-value =
## 0.0003987
## alternative hypothesis: true probability of success is less than 0.85
## 95 percent confidence interval:
##  0.0000000 0.5818034
## sample estimates:
## probability of success
##              0.1666667
```

It is possible that the order (alphabetically) is the wrong order, failures and successes, in which case we'd need to reorder the input to `binom.test()`.

In Chapter 6 we looked at the binomial distribution to obtain an exact Sign Test confidence interval for the median. Examine the following to see where the exact p-value for this test comes from.

```
n <- 6
x <- 0:n
p0 <- 0.85
bincdf <- pbinom(x, n, p0)
cdf <- data.frame(x, bincdf)
cdf
```

```
##   x       bincdf
## 1 0 1.139063e-05
## 2 1 3.986719e-04
## 3 2 5.885156e-03
## 4 3 4.733859e-02
## 5 4 2.235157e-01
## 6 5 6.228505e-01
## 7 6 1.000000e+00
```

■ CLICKER Qs — Excess successes STT.05.01.030 ■

7.4 Goodness-of-Fit Tests (Multinomial)

Example: jury pool The following data set was used as evidence in a court case. The data represent a sample of 1336 individuals from the jury pool of a large municipal court district for the years 1975–1977. The fairness of the representation of various age groups on juries was being contested. The strategy for doing this was to challenge the representativeness of the pool of individuals from which the juries are drawn. This was done by comparing the age group distribution within the jury pool against the age distribution in the district as a whole, which was available from census figures.

Age group (yrs)	Obs. Counts	Obs. Prop.	Census Prop.
18-19	23	0.017	0.061
20-24	96	0.072	0.150
25-29	134	0.100	0.135
30-39	293	0.219	0.217
40-49	297	0.222	0.153
50-64	380	0.284	0.182
65-99	113	0.085	0.102
Total:	1336	1.000	1.000

A statistical question here is whether the jury pool population proportions are equal to the census proportions across the age categories. This comparison can be formulated as a **goodness-of-fit test**, which generalizes the large-sample test on a single proportion to a categorical variable (here age) with $r > 2$ levels. For $r = 2$ categories, the goodness-of-fit test and large-sample test on a single proportion are identical. Although this problem compares two populations, only one sample is involved because the census data is a population summary!

In general, suppose each individual in a population is categorized into one and only one of r levels or categories. Let p_1, p_2, ..., p_r, be the population proportions in the r categories, where each $p_i \geq 0$ and $p_1 + p_2 + \cdots + p_r = 1$. The hypotheses of interest in a goodness-of-fit problem are $H_0 : p_1 = p_{01}$,

$p_2 = p_{02}$, ..., $p_r = p_{0r}$ and H_A : not H_0, where p_{01}, p_{02}, ..., p_{0r} are given category proportions.

The plausibility of H_0 is evaluated by comparing the hypothesized category proportions to estimated (i.e., observed) category proportions $\hat{p}_1$, $\hat{p}_2$, ..., $\hat{p}_r$ from a random or representative sample of n individuals selected from the population. The discrepancy between the hypothesized and observed proportions is measured by the Pearson chi-squared statistic:

$$\chi^2_{\text{s}} = \sum_{i=1}^{r} \frac{(O_i - E_i)^2}{E_i},$$

where O_i is the **observed** number in the sample that fall into the i^{th} category ($O_i = n\hat{p}_i$), and $E_i = np_{0i}$ is the number of individuals **expected** to be in the i^{th} category when H_0 is true.

The Pearson statistic can also be computed as the sum of the squared residuals:

$$\chi^2_{\text{s}} = \sum_{i=1}^{r} Z_i^2,$$

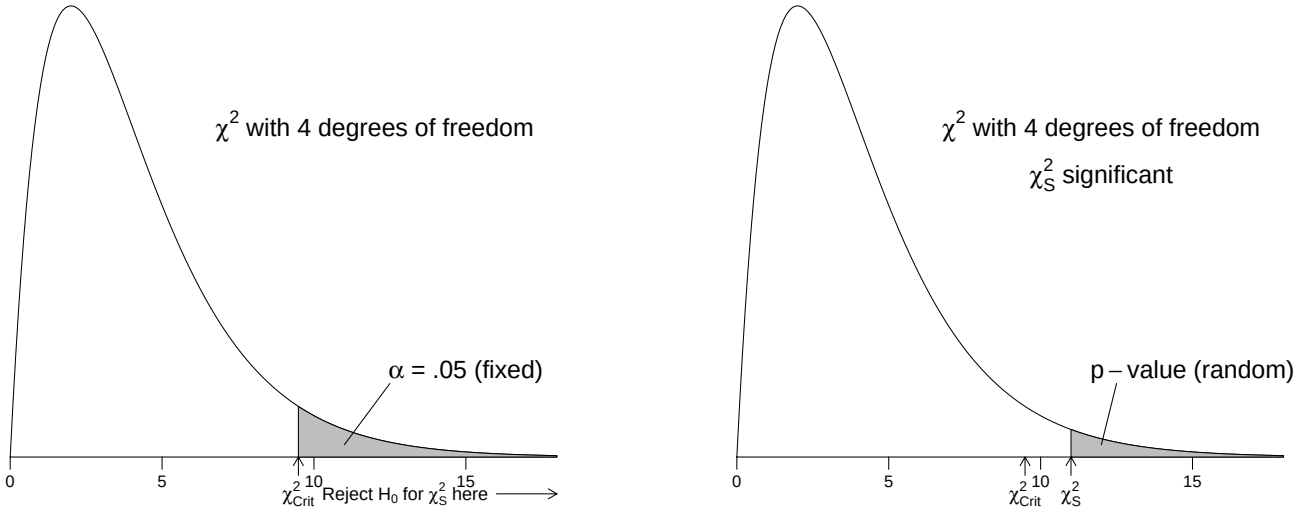
where $Z_i = (O_i - E_i)/\sqrt{E_i}$, or in terms of the observed and hypothesized category proportions

$$\chi^2_{\text{s}} = n \sum_{i=1}^{r} \frac{(\hat{p}_i - p_{0i})^2}{p_{0i}}.$$

The Pearson statistic χ^2_{s} is "small" when all of the observed counts (proportions) are close to the expected counts (proportions). The Pearson χ^2 is "large" when one or more observed counts (proportions) differs noticeably from what is expected when H_0 is true. Put another way, large values of χ^2_{s} suggest that H_0 is false.

The critical value χ^2_{crit} for the test is obtained from a chi-squared probability table with $r - 1$ degrees of freedom. The picture below shows the form of the rejection region. For example, if $r = 5$ and $\alpha = 0.05$, then you reject H_0 when $\chi^2_s \geq \chi^2_{\text{crit}} = 9.49$ (`qchisq(0.95, 5-1)`). The p-value for the test is the area

under the chi-squared curve with $df = r - 1$ to the right of the observed χ^2_s value.



Example: jury pool Let p_{18} be the proportion in the jury pool population between ages 18 and 19. Define p_{20}, p_{25}, p_{30}, p_{40}, p_{50}, and p_{65} analogously. You are interested in testing that the true jury proportions equal the census proportions, $H_0 : p_{18} = 0.061$, $p_{20} = 0.150$, $p_{25} = 0.135$, $p_{30} = 0.217$, $p_{40} = 0.153$, $p_{50} = 0.182$, and $p_{65} = 0.102$ against H_A : not H_0, using the sample of 1336 from the jury pool.

The observed counts, the expected counts, and the category residuals are given in the table below. For example, $E_{18} = 1336 \times (0.061) = 81.5$ and $Z_{18} = (23 - 81.5)/\sqrt{81.5} = -6.48$ in the 18-19 year category.

The Pearson statistic is

$$\chi^2_{\text{s}} = (-6.48)^2 + (-7.38)^2 + (-3.45)^2 + 0.18^2 + 6.48^2 + 8.78^2 + (-1.99)^2 = 231.26$$

on $r - 1 = 7 - 1 = 6$ degrees of freedom. Here $\chi^2_{\text{crit}} = 12.59$ at $\alpha = 0.05$. The p-value for the goodness-of-fit test is less than 0.001, which suggests that H_0 is false.

Age group (yrs)	Obs. Counts	Exp. Counts	Residual
18-19	23	81.5	−6.48
20-24	96	200.4	−7.38
25-29	134	180.4	−3.45
30-39	293	289.9	0.18
40-49	297	204.4	6.48
50-64	380	243.2	8.78
65-99	113	136.3	−1.99

7.4.1 Adequacy of the Goodness-of-Fit Test

The chi-squared goodness-of-fit test is a large-sample test. A conservative rule of thumb is that the test is suitable when each **expected** count is at least five. This holds in the jury pool example. There is no widely available alternative method for testing goodness-of-fit with smaller sample sizes. There is evidence, however, that the chi-squared test is **slightly conservative** (the p-values are too large, on average) when the expected counts are smaller. Some statisticians recommend that the chi-squared approximation be used when the minimum expected count is at least one, provided the expected counts are not too variable.

7.4.2 R Implementation

```
#### Example: jury pool
jury <- read.table(text="
Age    Count  CensusProp
18-19     23       0.061
20-24     96       0.150
25-29    134       0.135
30-39    293       0.217
40-49    297       0.153
50-64    380       0.182
65-99    113       0.102
", header=TRUE)

# show the structure of the data.frame
str(jury)
## 'data.frame': 7 obs. of  3 variables:
```

```
##  $ Age       : Factor w/ 7 levels "18-19","20-24",..: 1 2 3 4 5 6 7
##  $ Count     : int  23 96 134 293 297 380 113
##  $ CensusProp: num  0.061 0.15 0.135 0.217 0.153 0.182 0.102
```

```
# display the data.frame
jury
```

```
##     Age Count CensusProp
## 1 18-19    23      0.061
## 2 20-24    96      0.150
## 3 25-29   134      0.135
## 4 30-39   293      0.217
## 5 40-49   297      0.153
## 6 50-64   380      0.182
## 7 65-99   113      0.102
```

```
# calculate chi-square goodness-of-fit test
x.summary <- chisq.test(jury$Count, correct = FALSE, p = jury$CensusProp)
# print result of test
x.summary
```

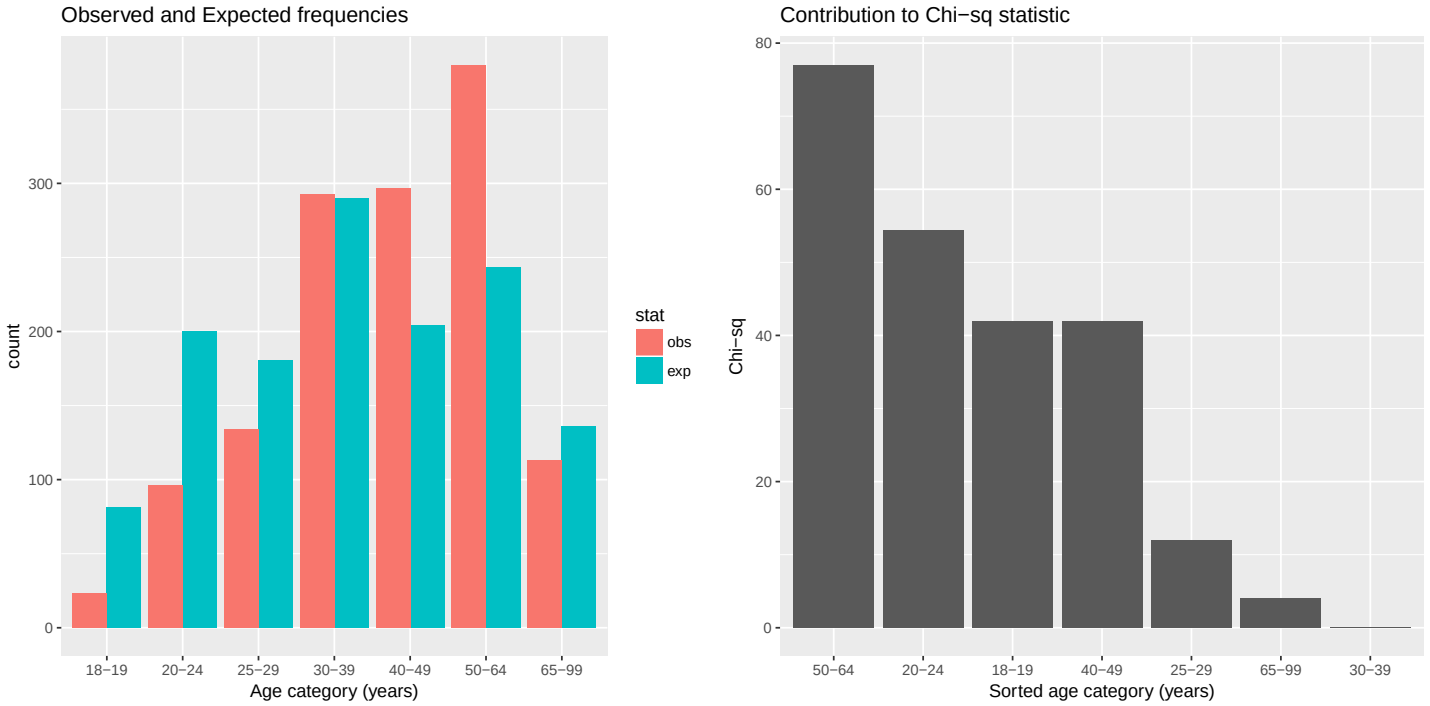
```
##
##  Chi-squared test for given probabilities
##
## data:  jury$Count
## X-squared = 231.26, df = 6, p-value < 2.2e-16
```

```
# use output in x.summary and create table
x.table <- data.frame(age = jury$Age
                      , obs = x.summary$observed
                      , exp = x.summary$expected
                      , res = x.summary$residuals
                      , chisq = x.summary$residuals^2
                      , stdres = x.summary$stdres)
x.table
```

```
##     age obs     exp        res       chisq     stdres
## 1 18-19  23  81.496 -6.4797466 41.98711613 -6.6869061
## 2 20-24  96 200.400 -7.3748237 54.38802395 -7.9991194
## 3 25-29 134 180.360 -3.4520201 11.91644267 -3.7116350
## 4 30-39 293 289.912  0.1813611  0.03289186  0.2049573
## 5 40-49 297 204.408  6.4762636 41.94199084  7.0369233
## 6 50-64 380 243.152  8.7760589 77.01921063  9.7033764
## 7 65-99 113 136.272 -1.9935650  3.97430128 -2.1037408
```

Plot observed vs expected values to help identify age groups that deviate the most. Plot contribution to chi-square values to help identify age groups that deviate the most. The term "Contribution to Chi-Square" (`chisq`) refers to the values of $\frac{(O-E)^2}{E}$ for each category. χ^2_s is the sum of those contributions.

```
library(reshape2)
x.table.obsexp <- melt(x.table,
              # id.vars: ID variables
              #   all variables to keep but not split apart on
              id.vars = c("age"),
              # measure.vars: The source columns
              #   (if unspecified then all other variables are measure.vars)
              measure.vars = c("obs", "exp"),
              # variable.name: Name of the destination column identifying each
              #   original column that the measurement came from
              variable.name = "stat",
              # value.name: column name for values in table
              value.name = "value"
            )

# naming variables manually, the variable.name and value.name not working 11/2012
names(x.table.obsexp) <- c("age", "stat", "value")

# Observed vs Expected counts
library(ggplot2)
p <- ggplot(x.table.obsexp, aes(x = age, fill = stat, weight=value))
p <- p + geom_bar(position="dodge")
p <- p + labs(title = "Observed and Expected frequencies")
p <- p + xlab("Age category (years)")
print(p)


# Contribution to chi-sq
# pull out only the age and chisq columns
x.table.chisq <- x.table[, c("age","chisq")]
# reorder the age categories to be descending relative to the chisq statistic
x.table.chisq$age <- with(x.table, reorder(age, -chisq))

p <- ggplot(x.table.chisq, aes(x = age, weight = chisq))
p <- p + geom_bar()
p <- p + labs(title = "Contribution to Chi-sq statistic")
p <- p + xlab("Sorted age category (years)")
p <- p + ylab("Chi-sq")
print(p)
```



7.4.3 Multiple Comparisons in a Goodness-of-Fit Problem

The goodness-of-fit test suggests that at least one of the age category proportions for the jury pool population differs from the census figures. A reasonable next step in the analysis would be to **separately** test the seven hypotheses: $H_0 : p_{18} = 0.061$, $H_0 : p_{20} = 0.150$, $H_0 : p_{25} = 0.135$, $H_0 : p_{30} = 0.217$, $H_0 : p_{40} = 0.153$, $H_0 : p_{50} = 0.182$, and $H_0 : p_{65} = 0.102$ to see which age categories led to this conclusion.

A Bonferroni comparison with a Family Error Rate ≤ 0.05 will be considered for this multiple comparisons problem. The error rates for the seven individual tests are set to $\alpha = 0.05/7 = 0.0071$, which corresponds to 99.29% two-sided CIs for the individual jury pool proportions. The area under the standard normal curve to the right of 2.69 is $0.05/2/7 = 0.00357$, about one-half the error rate for the individual CIs, so the critical value for the CIs, or for the tests, is $z_{\text{crit}} \approx 2.69$. The next table gives individual 99.29% CIs based on the large sample approximation. You can get the individual CIs in R using the `binom.test()` or `prop.test()` function. For example, the CI for Age Group

18-19 is obtained by specifying 23 successes in 1336 trials.

Below I perform exact binomial tests of proportion for each of the seven age categories at the Bonferroni-adjusted significance level. I save the p-values and confidence intervals in a table along with the observed and census proportions in order to display the table below.

```
b.sum1 <- binom.test(jury$Count[1], sum(jury$Count), p = jury$CensusProp[1], alternative = "two.sided", conf.level = 1-0.05/7)
b.sum2 <- binom.test(jury$Count[2], sum(jury$Count), p = jury$CensusProp[2], alternative = "two.sided", conf.level = 1-0.05/7)
b.sum3 <- binom.test(jury$Count[3], sum(jury$Count), p = jury$CensusProp[3], alternative = "two.sided", conf.level = 1-0.05/7)
b.sum4 <- binom.test(jury$Count[4], sum(jury$Count), p = jury$CensusProp[4], alternative = "two.sided", conf.level = 1-0.05/7)
b.sum5 <- binom.test(jury$Count[5], sum(jury$Count), p = jury$CensusProp[5], alternative = "two.sided", conf.level = 1-0.05/7)
b.sum6 <- binom.test(jury$Count[6], sum(jury$Count), p = jury$CensusProp[6], alternative = "two.sided", conf.level = 1-0.05/7)
b.sum7 <- binom.test(jury$Count[7], sum(jury$Count), p = jury$CensusProp[7], alternative = "two.sided", conf.level = 1-0.05/7)
# put the p-value and CI into a data.frame
b.sum  <- data.frame(
            rbind( c(b.sum1$p.value, b.sum1$conf.int)
                 , c(b.sum2$p.value, b.sum2$conf.int)
                 , c(b.sum3$p.value, b.sum3$conf.int)
                 , c(b.sum4$p.value, b.sum4$conf.int)
                 , c(b.sum5$p.value, b.sum5$conf.int)
                 , c(b.sum6$p.value, b.sum6$conf.int)
                 , c(b.sum7$p.value, b.sum7$conf.int)
            )
          )
names(b.sum) <- c("p.value", "CI.lower", "CI.upper")
b.sum$Age <- jury$Age
b.sum$Observed <- x.table$obs/sum(x.table$obs)
b.sum$CensusProp <- jury$CensusProp
b.sum
```

```
##        p.value   CI.lower   CI.upper   Age   Observed CensusProp
## 1 8.814860e-15 0.00913726 0.02920184 18-19 0.01721557      0.061
## 2 2.694633e-18 0.05415977 0.09294037 20-24 0.07185629      0.150
## 3 1.394274e-04 0.07939758 0.12435272 25-29 0.10029940      0.135
## 4 8.421685e-01 0.18962122 0.25120144 30-39 0.21931138      0.217
## 5 2.383058e-11 0.19245560 0.25433144 40-49 0.22230539      0.153
## 6 5.915839e-20 0.25174398 0.31880556 50-64 0.28443114      0.182
## 7 3.742335e-02 0.06536589 0.10707682 65-99 0.08458084      0.102
```

The CIs for the 30-39 and 65-99 year categories contain the census proportions. In the other five age categories, there are significant differences between the jury pool proportions and the census proportions. In general, young adults appear to be underrepresented in the jury pool whereas older age groups are overrepresented.

```
##
## Attaching package:  'xtable'
## The following object is masked from 'package:TeachingDemos':
##
##     digits
```

	Age	p.value	CI.lower	CI.upper	Observed	CensusProp
1	18-19	0.000	0.009	0.029	0.017	0.061
2	20-24	0.000	0.054	0.093	0.072	0.150
3	25-29	0.000	0.079	0.124	0.100	0.135
4	30-39	0.842	0.190	0.251	0.219	0.217
5	40-49	0.000	0.192	0.254	0.222	0.153
6	50-64	0.000	0.252	0.319	0.284	0.182
7	65-99	0.037	0.065	0.107	0.085	0.102

The residuals also highlight significant differences because the largest residuals correspond to the categories that contribute most to the value of χ^2_{s}. Some researchers use the residuals for the multiple comparisons, treating the Z_is as standard normal variables. Following this approach, you would conclude that the jury pool proportions differ from the proportions in the general population in every age category where $|Z_i| \geq 2.70$ (using the same Bonferroni correction). This gives the same conclusion as before.

The two multiple comparison methods are similar, but not identical. The residuals

$$Z_i = \frac{O_i - E_i}{\sqrt{E_i}} = \frac{\hat{p}_i - p_{0i}}{\sqrt{\frac{p_{0i}}{n}}}$$

agree with the large-sample statistic for testing $H_0 : p_i = p_{0i}$, except that the divisor in Z_i omits a $1 - p_{0i}$ term. The Z_is are not standard normal random variables as assumed, and the value of Z_i underestimates the significance of the observed differences. Multiple comparisons using the Z_is will find, on average, fewer significant differences than the preferred method based on the large sample tests. However, the differences between the two methods are usually minor when all of the hypothesized proportions are small.

7.5 Comparing Two Proportions: Independent Samples

The New Mexico state legislature is interested in how the proportion of registered voters that support Indian gaming differs between New Mexico and Colorado. Assuming neither population proportion is known, the state's statistician might recommend that the state conduct a survey of registered voters sampled independently from the two states, followed by a comparison of the sample proportions in favor of Indian gaming.

Statistical methods for comparing two proportions using independent samples can be formulated as follows. Let p_1 and p_2 be the proportion of populations 1 and 2, respectively, with the attribute of interest. Let $\hat{p}_1$ and $\hat{p}_2$ be the corresponding sample proportions, based on independent random or representative samples of size n_1 and n_2 from the two populations.

7.5.1 Large Sample CI and Tests for $p_1 - p_2$

A large-sample CI for $p_1 - p_2$ is $(\hat{p}_1 - \hat{p}_2) \pm z_{\text{crit}} SE_{\text{CI}}(\hat{p}_1 - \hat{p}_2)$, where z_{crit} is the standard normal critical value for the desired confidence level, and

$$SE_{\text{CI}}(\hat{p}_1 - \hat{p}_2) = \sqrt{\frac{\hat{p}_1(1-\hat{p}_1)}{n_1} + \frac{\hat{p}_2(1-\hat{p}_2)}{n_2}}$$

is the CI standard error.

A large-sample p-value for a test of the null hypothesis $H_0 : p_1 - p_2 = 0$ against the two-sided alternative $H_A : p_1 - p_2 \neq 0$ is evaluated using tail areas of the standard normal distribution (identical to one sample evaluation) in conjunction with the test statistic

$$z_s = \frac{\hat{p}_1 - \hat{p}_2}{SE_{\text{test}}(\hat{p}_1 - \hat{p}_2)},$$

where

$$SE_{\text{test}}(\hat{p}_1 - \hat{p}_2) = \sqrt{\frac{\bar{p}(1-\bar{p})}{n_1} + \frac{\bar{p}(1-\bar{p})}{n_2}} = \sqrt{\bar{p}(1-\bar{p})\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}$$

is the test standard error for $\hat{p}_1 - \hat{p}_2$. The **pooled proportion**

$$\bar{p} = \frac{n_1\hat{p}_1 + n_2\hat{p}_2}{n_1 + n_2}$$

is the proportion of successes in the two samples combined. The test standard error has the same functional form as the CI standard error, with $\bar{p}$ replacing the individual sample proportions.

The pooled proportion is the best guess at the common population proportion when $H_0 : p_1 = p_2$ is true. The test standard error estimates the standard deviation of $\hat{p}_1 - \hat{p}_2$ assuming H_0 is true.

Remark: As in the one-sample proportion problem, the test and CI SE's are different. This *can* (but usually does not) lead to some contradiction between the test and CI.

Example, vitamin C Two hundred and seventy nine (279) French skiers were studied during two one-week periods in 1961. One group of 140 skiers receiving a placebo each day, and the other 139 receiving 1 gram of ascorbic acid (Vitamin C) per day. The study was double blind — neither the subjects nor the researchers knew who received which treatment. Let p_1 be the probability that a member of the ascorbic acid group contracts a cold during the study period, and p_2 be the corresponding probability for the placebo group. Linus Pauling (Chemistry and Peace Nobel prize winner) and I are interested in testing whether $H_0 : p_1 = p_2$. The data are summarized below as a two-by-two table of counts (a contingency table)

Outcome	Ascorbic Acid	Placebo
# with cold	17	31
# with no cold	122	109
Totals	139	140

The sample sizes are $n_1 = 139$ and $n_2 = 140$. The sample proportion of skiers developing colds in the placebo and treatment groups are $\hat{p}_2 = 31/140 = 0.221$ and $\hat{p}_1 = 17/139 = 0.122$, respectively. The difference is $\hat{p}_1 - \hat{p}_2 = 0.122 - 0.221 = -0.099$. The pooled proportion is the number of skiers that developed colds divided by the number of skiers in the study: $\bar{p} = 48/279 = 0.172$.

The test standard error is

$$SE_{\text{test}}(\hat{p}_1 - \hat{p}_2) = \sqrt{0.172 \times (1 - 0.172)\left(\frac{1}{139} + \frac{1}{140}\right)} = 0.0452.$$

The test statistic is

$$z_s = \frac{0.122 - 0.221}{0.0452} = -2.19.$$

The p-value for a two-sided test is twice the area under the standard normal curve to the right of 2.19 (or twice the area to the left of -2.19), which is $2 \times (0.014) = 0.028$. At the 5% level, we reject the hypothesis that the probability of contracting a cold is the same whether you are given a placebo or Vitamin C.

A CI for $p_1 - p_2$ provides a measure of the size of the treatment effect. For a 95% CI

$$\begin{aligned} z_{\text{crit}} SE_{\text{CI}}(\hat{p}_1 - \hat{p}_2) &= 1.96\sqrt{\frac{0.221 \times (1 - 0.221)}{140} + \frac{0.122 \times (1 - 0.122)}{139}} \\ &= 1.96 \times (0.04472) = 0.088. \end{aligned}$$

The 95% CI for $p_1 - p_2$ is -0.099 ± 0.088, or $(-0.187, -0.011)$. We are 95% confident that p_2 exceeds p_1 by at least 0.011 but not by more than 0.187.

On the surface, we would conclude that a daily dose of Vitamin C decreases a French skier's chance of developing a cold by between 0.011 and 0.187 (with 95% confidence). This conclusion was somewhat controversial. Several reviews

of the study felt that the experimenter's evaluations of cold symptoms were unreliable. Many other studies refute the benefit of Vitamin C as a treatment for the common cold.

```
#### Example, vitamin C
# Approximate normal test for two-proportions, without Yates' continuity correction
prop.test(c(17, 31), c(139, 140), correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity
##  correction
##
## data:  c(17, 31) out of c(139, 140)
## X-squared = 4.8114, df = 1, p-value = 0.02827
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.18685917 -0.01139366
## sample estimates:
##    prop 1    prop 2
## 0.1223022 0.2214286
```

Conditional probability

In probability theory, a conditional probability is the probability that an event will occur, when another event is known to occur or to have occurred. If the events are A and B respectively, this is said to be "the probability of A given B". It is commonly denoted by $\Pr(A|B)$. $\Pr(A|B)$ may or may not be equal to $\Pr(A)$, the probability of A. If they are equal, A and B are said to be independent. For example, if a coin is flipped twice, "the outcome of the second flip" is independent of "the outcome of the first flip".

In the Vitamin C example above, the unconditional observed probability of contracting a cold is $\Pr(\text{cold}) = (17 + 31)/(139 + 140) = 0.172$. The conditional observed probabilities are $\Pr(\text{cold}|\text{ascorbic acid}) = 17/139 = 0.1223$ and $\Pr(\text{cold}|\text{placebo}) = 31/140 = 0.2214$. The two-sample test of $H_0 : p_1 = p_2$ where $p_1 = \Pr(\text{cold}|\text{ascorbic acid})$ and $p_2 = \Pr(\text{cold}|\text{placebo})$ is effectively testing whether $\Pr(\text{cold}) = \Pr(\text{cold}|\text{ascorbic acid}) = \Pr(\text{cold}|\text{placebo})$. This tests whether contracting a cold is independent of the vitamin C treatment.

Example, cervical dysplasia A case-control study was designed to examine risk factors for cervical dysplasia[2] (Becker et al. 1994). All the women in the study were patients at UNM clinics. The 175 cases were women, aged 18-40, who had cervical dysplasia. The 308 controls were women aged 18-40 who did not have cervical dysplasia. Each woman was classified as positive or negative, depending on the presence of HPV (human papilloma virus). The data are summarized below.

HPV Outcome	Cases	Controls
Positive	164	130
Negative	11	178
Sample size	175	308

We'll take a short detour to create this table in R, calculate the column proportions, and plot the frequencies and proportions.

We first create a table with labelled rows and columns.

```
# Create the labelled table
hpv <-
matrix(c(164, 130, 11, 178),
       nrow = 2, byrow = TRUE,
       dimnames = list("HPV.Outcome" = c("Positive", "Negative"),
                       "Group" = c("Cases", "Controls")))
hpv
```

```
##             Group
## HPV.Outcome Cases Controls
##    Positive   164      130
##    Negative    11      178
```

Next, we create column proportions.

```
# calculate the column (margin = 2) proportions
hpv.col.prop <- prop.table(hpv, margin = 2)
hpv.col.prop
```

```
##             Group
## HPV.Outcome      Cases  Controls
##    Positive 0.93714286 0.4220779
##    Negative 0.06285714 0.5779221
```

Here, we reshape the data from wide format to long format. This will allow us to make plots later, and also shows how to create these tables from a dataset in long format (which is the typical format).

[2] http://www.ncbi.nlm.nih.gov/pubmedhealth/PMH0002461/

```
# OR, convert to long format and use xtabs to produce the table
library(reshape2)
hpv.long <- melt(hpv, value.name = "Frequency")
hpv.long
```

```
##   HPV.Outcome    Group Frequency
## 1    Positive    Cases       164
## 2    Negative    Cases        11
## 3    Positive Controls       130
## 4    Negative Controls       178
```

```
T1 <- xtabs(Frequency ~ HPV.Outcome + Group, data = hpv.long)
T1
```

```
##             Group
## HPV.Outcome Cases Controls
##    Positive   164      130
##    Negative    11      178
```

```
hpv.col.prop <- prop.table(T1, margin = 2)
hpv.col.prop
```

```
##             Group
## HPV.Outcome      Cases   Controls
##    Positive 0.93714286 0.42207792
##    Negative 0.06285714 0.57792208
```

Add a column of proportions to our long-formatted data to plot these proportion values.

```
library(reshape2)
hpv.col.prop.long <- melt(hpv.col.prop, value.name = "Proportion")
hpv.col.prop.long
```

```
##   HPV.Outcome    Group Proportion
## 1    Positive    Cases 0.93714286
## 2    Negative    Cases 0.06285714
## 3    Positive Controls 0.42207792
## 4    Negative Controls 0.57792208
```

```
# join these two datasets to have both Freq and Prop columns
library(plyr)
hpv.long <- join(hpv.long, hpv.col.prop.long)
```

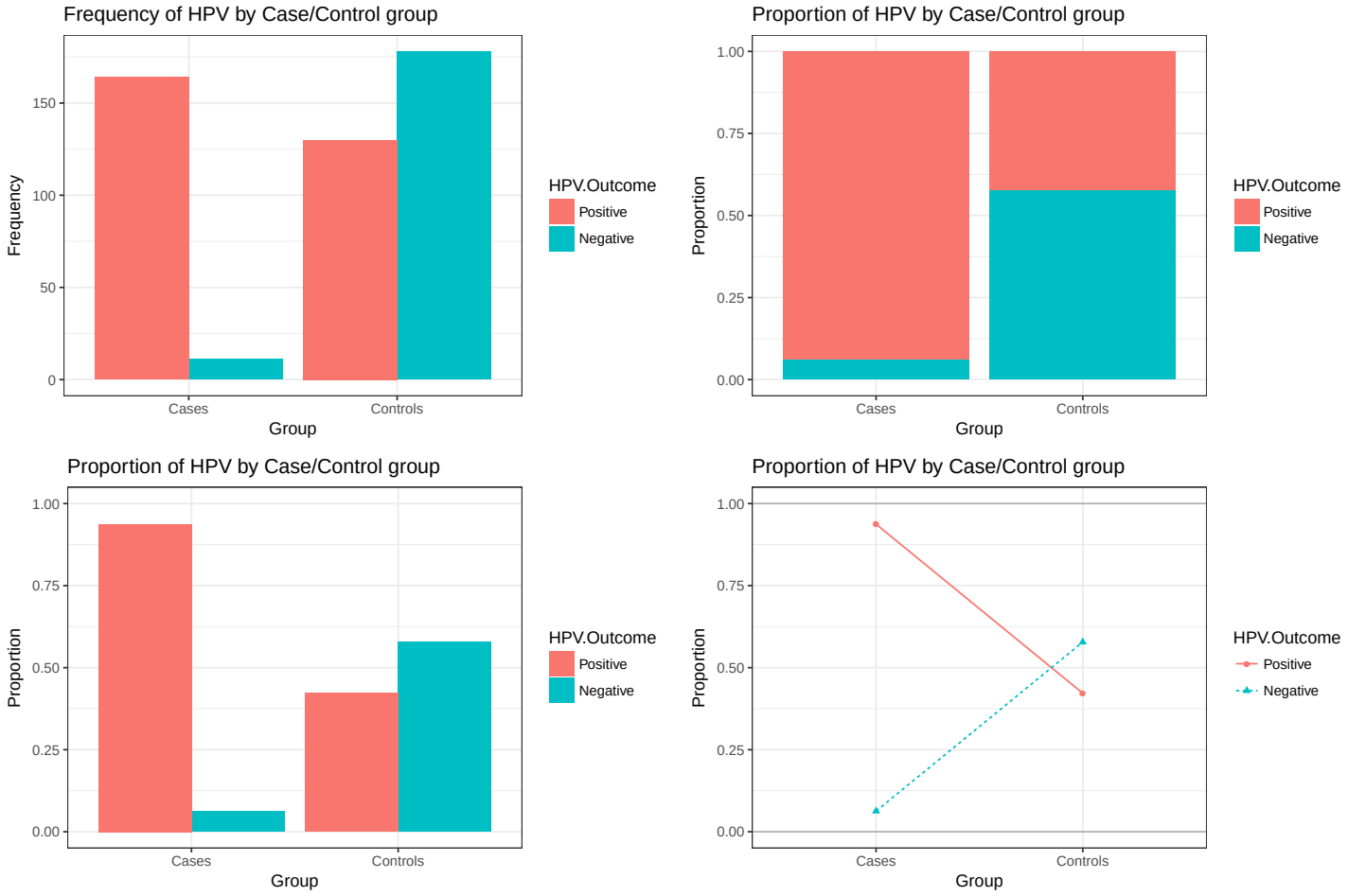
```
## Joining by:  HPV.Outcome, Group
```

```
hpv.long
```

```
##   HPV.Outcome    Group Frequency Proportion
## 1    Positive    Cases       164 0.93714286
## 2    Negative    Cases        11 0.06285714
## 3    Positive Controls       130 0.42207792
## 4    Negative Controls       178 0.57792208
```

Finally, plot the frequencies, and the proportions in three ways (the frequencies can obviously be plotted in many ways, too).

```r
# plots are easier now that data are in long format.
library(ggplot2)
p <- ggplot(data = hpv.long, aes(x = Group, y = Frequency, fill = HPV.Outcome))
p <- p + geom_bar(stat="identity", position = "dodge")
p <- p + theme_bw()
p <- p + labs(title = "Frequency of HPV by Case/Control group")
print(p)

# bars, stacked
library(ggplot2)
p <- ggplot(data = hpv.long, aes(x = Group, y = Proportion, fill = HPV.Outcome))
p <- p + geom_bar(stat="identity")
p <- p + theme_bw()
p <- p + labs(title = "Proportion of HPV by Case/Control group")
print(p)

# bars, dodged
library(ggplot2)
p <- ggplot(data = hpv.long, aes(x = Group, y = Proportion, fill = HPV.Outcome))
p <- p + geom_bar(stat="identity", position = "dodge")
p <- p + theme_bw()
p <- p + labs(title = "Proportion of HPV by Case/Control group")
p <- p + scale_y_continuous(limits = c(0, 1))
print(p)

# lines are sometimes easier, especially when many categories along the x-axis
library(ggplot2)
p <- ggplot(data = hpv.long, aes(x = Group, y = Proportion, colour = HPV.Outcome))
p <- p + geom_hline(yintercept = c(0, 1), alpha = 1/4)
p <- p + geom_point(aes(shape = HPV.Outcome))
p <- p + geom_line(aes(linetype = HPV.Outcome, group = HPV.Outcome))
p <- p + theme_bw()
p <- p + labs(title = "Proportion of HPV by Case/Control group")
p <- p + scale_y_continuous(limits = c(0, 1))
print(p)
```



Returning to the hypothesis test, let p_1 be the probability that a case is HPV positive and let p_2 be the probability that a control is HPV positive. The sample sizes are $n_1 = 175$ and $n_2 = 308$. The sample proportions of positive cases and controls are $\hat{p}_1 = 164/175 = 0.937$ and $\hat{p}_2 = 130/308 = 0.422$.

For a 95% CI

$$\begin{aligned} z_{\text{crit}} SE_{\text{CI}}(\hat{p}_1 - \hat{p}_2) &= 1.96\sqrt{\frac{0.937 \times (1-0.937)}{175} + \frac{0.422 \times (1-0.422)}{308}} \\ &= 1.96 \times (0.03336) = 0.0659. \end{aligned}$$

A 95% CI for $p_1 - p_2$ is $(0.937-0.422) \pm 0.066$, or 0.515 ± 0.066, or $(0.449, 0.581)$. I am 95% confident that p_1 exceeds p_2 by at least 0.45 but not by more than 0.58.

```
# Approximate normal test for two-proportions, without Yates' continuity correction
prop.test(c(164, 130), c(175, 308), correct = FALSE)

##
##  2-sample test for equality of proportions without continuity
```

```
##  correction
##
## data:  c(164, 130) out of c(175, 308)
## X-squared = 124.29, df = 1, p-value < 2.2e-16
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  0.4492212 0.5809087
## sample estimates:
##    prop 1    prop 2
## 0.9371429 0.4220779
```

Not surprisingly, a two-sided test at the 5% level would reject $H_0 : p_1 = p_2$. In this problem one might wish to do a one-sided test, instead of a two-sided test. Let us carry out this test, as a refresher on how to conduct one-sided tests.

```
# one-sided test, are cases more likely to be HPV positive?
prop.test(c(164, 130), c(175, 308), correct = FALSE, alternative = "greater")
```

```
##
##  2-sample test for equality of proportions without continuity
##  correction
##
## data:  c(164, 130) out of c(175, 308)
## X-squared = 124.29, df = 1, p-value < 2.2e-16
## alternative hypothesis: greater
## 95 percent confidence interval:
##  0.4598071 1.0000000
## sample estimates:
##    prop 1    prop 2
## 0.9371429 0.4220779
```

Appropriateness of Large Sample Test and CI

The standard two-sample CI and test used above are appropriate when each sample is large. A rule of thumb suggests a minimum of at least five successes (i.e., observations with the characteristic of interest) and failures (i.e., observations without the characteristic of interest) in each sample before using these methods. This condition is satisfied in our two examples.

■ ClickerQs — Comparing two proportions STT.08.02.010 ■

7.6 Effect Measures in Two-by-Two Tables

Consider a study of a particular disease, where each individual is either exposed or not-exposed to a risk factor. Let p_1 be the proportion diseased among the individuals in the exposed population, and p_2 be the proportion diseased among the non-exposed population. This population information can be summarized as a two-by-two table of population proportions:

Outcome	Exposed population	Non-Exposed population
Diseased	p_1	p_2
Non-Diseased	$1-p_1$	$1-p_2$

A standard measure of the difference between the exposed and non-exposed populations is the **absolute difference:** $p_1 - p_2$. We have discussed statistical methods for assessing this difference.

In many epidemiological and biostatistical settings, other measures of the difference between populations are considered. For example, the relative risk

$$\text{RR} = \frac{p_1}{p_2}$$

is commonly reported when the individual risks p_1 and p_2 are small. The odds ratio

$$\text{OR} = \frac{p_1/(1-p_1)}{p_2/(1-p_2)}$$

is another standard measure. Here $p_1/(1-p_1)$ is the odds of being diseased in the exposed group, whereas $p_2/(1-p_2)$ is the odds of being diseased in the non-exposed group.

I mention these measures because you may see them or hear about them. Note that each of these measures can be easily estimated from data, using the sample proportions as estimates of the unknown population proportions. For example, in the vitamin C study:

Outcome	Ascorbic Acid	Placebo
# with cold	17	31
# with no cold	122	109
Totals	139	140

the proportion with colds in the placebo group is $\hat{p}_2 = 31/140 = 0.221$. The proportion with colds in the vitamin C group is $\hat{p}_1 = 17/139 = 0.122$.

The estimated absolute difference in risk is $\hat{p}_1 - \hat{p}_2 = 0.122 - 0.221 = -0.099$. The estimated risk ratio and odds ratio are

$$\widehat{RR} = \frac{0.122}{0.221} = 0.55$$

and

$$\widehat{OR} = \frac{0.122/(1-0.122)}{0.221/(1-0.221)} = 0.49,$$

respectively.

Interpretting odds ratios, two examples Let's begin with probability[3]. Let's say that the probability of success is 0.8, thus $p = 0.8$ Then the probability of failure is $q = 1 - p = 0.2$ The odds of success are defined as odds(success) $= p/q = 0.8/0.2 = 4$, that is, the odds of success are 4 to 1. The odds of failure would be odds(failure) $= q/p = 0.2/0.8 = 0.25$, that is, the odds of failure are 1 to 4. Next, let's compute the odds ratio by OR = odds(success)/odds(failure) $= 4/0.25 = 16$ The interpretation of this odds ratio would be that the odds of success are 16 times greater than for failure. Now if we had formed the odds ratio the other way around with odds of failure in the numerator, we would have gotten something like this, OR = odds(failure)/odds(success) $= 0.25/4 = 0.0625$ Interestingly enough, the interpretation of this odds ratio is nearly the same as the one above. Here the interpretation is that the odds of failure are one-sixteenth the odds of success. In fact, if you take the reciprocal of the first odds ratio you get $1/16 = 0.0625$.

Another example This example is adapted from Pedhazur (1997). Suppose that seven out of 10 males are admitted to an engineering school while three of 10 females are admitted. The probabilities for admitting a male are,

[3]Borrowed graciously from UCLA Academic Technology Services at `http://www.ats.ucla.edu/stat/sas/faq/oratio.htm`

$p = 7/10 = 0.7$ and $q = 1 - 0.7 = 0.3$. Here are the same probabilities for females, $p = 3/10 = 0.3$ and $q = 1 - 0.3 = 0.7$. Now we can use the probabilities to compute the admission odds for both males and females, odds(male) $= 0.7/0.3 = 2.33333$ and odds(female) $= 0.3/0.7 = 0.42857$. Next, we compute the odds ratio for admission, OR $= 2.3333/0.42857 = 5.44$. Thus, the odds of a male being admitted are 5.44 times greater than for a female.

7.7 Analysis of Paired Samples: Dependent Proportions

Paired and more general **block analyses** are appropriate with longitudinal data collected over time and in medical studies where several treatments are given to the same patient over time. A key feature of these designs that invalidates the two-sample method discussed earlier is that repeated observations within a unit or individual are likely to be correlated, and not independent.

Example, President performance For example, in a random sample of $n = 1600$ voter-age Americans, 944 said that they approved of the President's performance. One month later, only 880 of the original 1600 sampled approved. The following two-by-two table gives the numbers of individuals with each of the four possible sequences of responses over time. Thus, 150 voter-age Americans approved of the President's performance when initially asked but then disapproved one month later. The row and column totals are the numbers of approvals and disapprovals for the two surveys (Agresti, 1990, p. 350).

(Obs Counts)	Second survey		
First Survey	Approve	Disapprove	Total
Approve	794	150	944
Disapprove	86	570	656
Total	880	720	1600

Let p_{AA}, p_{AD}, p_{DA}, p_{DD} be the population proportion of voter-age Americans that fall into the four categories, where the subscripts preserve the time

ordering and indicate Approval or Disapproval. For example, p_{AD} is the population proportion that approved of the President's performance initially and disapproved one month later. The population proportion that initially approved is $p_{\text{A+}} = p_{\text{AA}} + p_{\text{AD}}$. The population proportion that approved at the time of the second survey is $p_{\text{+A}} = p_{\text{AA}} + p_{\text{DA}}$. The "+" sign used as a subscript means that the replaced subscript has been summed over.

Similarly, let $\hat{p}_{\text{AA}}$, $\hat{p}_{\text{AD}}$, $\hat{p}_{\text{DA}}$, $\hat{p}_{\text{DD}}$ be the sample proportion of voter-age Americans that fall into the four categories, and let $\hat{p}_{\text{A+}} = \hat{p}_{\text{AA}} + \hat{p}_{\text{AD}}$ and $\hat{p}_{\text{+A}} = \hat{p}_{\text{AA}} + \hat{p}_{\text{DA}}$ be the sample proportion that approves the first month and the sample proportion that approves the second month, respectively. The table below summarizes the observed proportions. For example, $\hat{p}_{\text{AA}} = 794/1600 = 0.496$ and $\hat{p}_{\text{A+}} = 944/1600 = 0.496 + 0.094 = 0.590$. The sample proportion of voting-age Americans that approve of the President's performance decreased from one month to the next.

(Obs Proportions)	Second survey		
First Survey	Approve	Disapprove	Total
Approve	0.496	0.094	0.590
Disapprove	0.054	0.356	0.410
Total	0.550	0.450	1.000

The difference in the population proportions from one month to the next can be assessed by a large-sample CI for $p_{\text{A+}} - p_{\text{+A}}$, given by $(\hat{p}_{\text{A+}} - \hat{p}_{\text{+A}}) \pm z_{\text{crit}} SE_{\text{CI}}(\hat{p}_{\text{A+}} - \hat{p}_{\text{+A}})$, where the CI standard error satisfies

$$SE_{\text{CI}}(\hat{p}_{\text{A+}} - \hat{p}_{\text{+A}}) = \sqrt{\frac{\hat{p}_{\text{A+}}(1 - \hat{p}_{\text{A+}}) + \hat{p}_{\text{+A}}(1 - \hat{p}_{\text{+A}}) - 2(\hat{p}_{\text{AA}}\hat{p}_{\text{DD}} - \hat{p}_{\text{AD}}\hat{p}_{\text{DA}})}{n}}$$

One-sided bounds are constructed in the usual way.

The $-2(\cdot)$ term in the standard error accounts for the dependence between the samples at the two time points. If independent samples of size n had been selected for the two surveys, then this term would be omitted from the standard error, giving the usual two-sample CI.

For a 95% CI in the Presidential survey,

$$\begin{aligned} z_{\text{crit}} SE_{\text{CI}}(\hat{p}_{\text{A}+} - \hat{p}_{+\text{A}}) &= 1.96\sqrt{\frac{0.590 \times 0.410 + 0.550 \times 0.450 - 2(0.496 \times 0.356 - 0.094 \times 0.054)}{1600}} \\ &= 1.96 \times (0.0095) \;=\; 0.0186. \end{aligned}$$

A 95% CI for $p_{\text{A}+} - p_{+\text{A}}$ is $(0.590 - 0.550) \pm 0.019$, or $(0.021,\ 0.059)$. You are 95% confident that the population proportion of voter-age Americans that approved of the President's performance the first month was between 0.021 and 0.059 larger than the proportion that approved one month later. This gives evidence of a decrease in the President's approval rating.

A test of $H_0 : p_{\text{A}+} = p_{+\text{A}}$ can be based on the CI for $p_{\text{A}+} - p_{+\text{A}}$, or on a standard normal approximation to the test statistic

$$z_s = \frac{\hat{p}_{\text{A}+} - \hat{p}_{+\text{A}}}{SE_{\text{test}}(\hat{p}_{\text{A}+} - \hat{p}_{+\text{A}})},$$

where the test standard error is given by

$$SE_{\text{test}}(\hat{p}_{\text{A}+} - \hat{p}_{+\text{A}}) = \sqrt{\frac{\hat{p}_{\text{A}+}\hat{p}_{+\text{A}} - 2\hat{p}_{\text{AA}}}{n}}.$$

The test statistic is often written in the simplified form

$$z_s = \frac{n_{\text{AD}} - n_{\text{DA}}}{\sqrt{n_{\text{AD}} + n_{\text{DA}}}},$$

where the n_{ij}s are the observed cell counts. An equivalent form of this test, based on comparing the square of z_s to a chi-squared distribution with 1 degree of freedom, is the well-known **McNemar's test** for marginal homogeneity (or symmetry) in the two-by-two table.

For example, in the Presidential survey

$$z_s = \frac{150 - 86}{\sqrt{150 + 86}} = 4.17.$$

The p-value for a two-sided test is, as usual, the area under the standard normal curve outside ± 4.17. The p-value is less than 0.001, suggesting that H_0 is false.

R can perform this test as McNemar's test.

```
#### Example, President performance
# McNemar's test needs data as a matrix

# Presidential Approval Ratings.
#  Approval of the President's performance in office in two surveys,
#  one month apart, for a random sample of 1600 voting-age Americans.
pres <-
matrix(c(794, 150, 86, 570),
       nrow = 2, byrow = TRUE,
       dimnames = list("1st Survey" = c("Approve", "Disapprove"),
                       "2nd Survey" = c("Approve", "Disapprove")))
pres
```

```
##             2nd Survey
## 1st Survey   Approve Disapprove
##   Approve        794        150
##   Disapprove      86        570
```

```
mcnemar.test(pres, correct=FALSE)
```

```
##
##  McNemar's Chi-squared test
##
## data:  pres
## McNemar's chi-squared = 17.356, df = 1, p-value = 3.099e-05
```

```
# => significant change (in fact, drop) in approval ratings
```

7.8 Testing for Homogeneity of Proportions

Example, cancer deaths The following two-way table of counts summarizes the location of death and age at death from a study of 2989 cancer deaths (Public Health Reports, 1983).

(Obs Counts)	Location of death			
Age	Home	Acute Care	Chronic care	Row Total
15-54	94	418	23	535
55-64	116	524	34	674
65-74	156	581	109	846
75+	138	558	238	934
Col Total	504	2081	404	2989

The researchers want to compare the age distributions across locations. A one-way ANOVA would be ideal if the actual ages were given. Because the ages

are grouped, the data should be treated as categorical. Given the differences in numbers that died at the three types of facilities, a comparison of proportions or percentages in the age groups is appropriate. A comparison of counts is not.

The table below summarizes the proportion in the four age groups by location. For example, in the acute care facility $418/2081 = 0.201$ and $558/2081 = 0.268$. The **pooled proportions** are the Row Totals divided by the total sample size of 2989. The pooled summary gives the proportions in the four age categories, ignoring location of death.

The age distributions for home and for the acute care facilities are similar, but are very different from the age distribution at chronic care facilities.

To formally compare the observed proportions, one might view the data as representative sample of ages at death from the three locations. Assuming independent samples from the three locations (populations), a chi-squared statistic is used to test whether the population proportions of ages at death are identical (homogeneous) across locations. The **chi-squared test for homogeneity** of population proportions can be defined in terms of proportions, but is traditionally defined in terms of counts.

(Proportions)	Location of death			
Age	Home	Acute Care	Chronic care	Pooled
15-54	0.187	0.201	0.057	0.179
55-64	0.230	0.252	0.084	0.226
65-74	0.310	0.279	0.270	0.283
75+	0.273	0.268	0.589	0.312
Total	1.000	1.000	1.000	1.000

In general, assume that the data are independent samples from c populations (strata, groups, sub-populations), and that each individual is placed into one of r levels of a categorical variable. The raw data will be summarized as a $r \times c$ **contingency table** of counts, where the columns correspond to the samples, and the rows are the levels of the categorical variable. In the age distribution problem, $r = 4$ and $c = 3$.

To implement the test:

1. Compute the (estimated) **expected** count for each cell in the table as follows:
$$\text{E} = \frac{\text{Row Total} \times \text{Column Total}}{\text{Total Sample Size}}.$$
2. Compute the Pearson test statistic
$$\chi^2_{\text{s}} = \sum_{\text{all cells}} \frac{(O-E)^2}{E},$$
where O is the **observed** count.
3. For a size α test, reject the hypothesis of homogeneity if $\chi^2_{\text{s}} \geq \chi^2_{\text{crit}}$, where χ^2_{crit} is the upper α critical value from the chi-squared distribution with $df = (r-1)(c-1)$.

The p-value for the chi-squared test of homogeneity is equal to the area under the chi-squared curve to the right of χ^2_{s}.

For a two-by-two table of counts, the chi-squared test of homogeneity of proportions is identical to the two-sample proportion test we discussed earlier.

The (estimated) expected counts for the (15-54, Home) cell and for the (75+, Acute Care) cell in the age distribution data are $E = 535 \times 504/2989 = 90.21$ and $E = 934 \times 2081/2989 = 650.27$, respectively. The other expected counts were computed similarly, and are summarized below. The row and column sums on the tables of observed and expected counts always agree.

(Exp Counts)	Location of death			
Age	Home	Acute Care	Chronic care	Row Total
15-54	90.21	372.48	72.31	535
55-64	113.65	469.25	91.10	674
65-74	142.65	589	114.35	846
75-	157.49	650.27	126.24	934
Col Total	504	2081	404	2989

Why is a comparison of the observed and expected counts relevant for testing homogeneity? To answer this question, first note that the expected cell count

can be expressed

$$\text{E} = \text{Col Total} \times \text{Pooled proportion for category.}$$

For example, $E = 504 \times 0.179 = 90.21$ in the (15-54, Home) cell. A comparison of the observed and expected counts is a comparison of the observed category proportions in a location with the pooled proportions, taking the size of each sample into consideration. Thinking of the pooled proportion as a weighted average of the sample proportions for a category, the Pearson χ^2_s statistic is an aggregate measure of variation in the observed proportions across samples. If the category proportions are similar across samples then the category and pooled proportions are similar, resulting in a "small" value of χ^2_s. Large values of χ^2_s occur when there is substantial variation in the observed proportions across samples, in one or more categories. In this regard, the Pearson statistic is similar to the F-statistic in a one-way ANOVA (where large differences between groups result in a large F-statistic).

```
#### Example, cancer deaths
candeath <-
matrix(c(  94, 418,  23,
          116, 524,  34,
          156, 581, 109,
          138, 558, 238),
       nrow = 4, byrow = TRUE,
       dimnames = list("Age" = c("15-54", "55-64", "65-74", "75+"),
                  "Location of death" = c("Home", "Acute Care", "Chronic care")))
candeath
```

```
##        Location of death
## Age     Home Acute Care Chronic care
##   15-54   94        418           23
##   55-64  116        524           34
##   65-74  156        581          109
##   75+    138        558          238
```

```
chisq.summary <- chisq.test(candeath, correct=FALSE)
chisq.summary
```

```
##
##  Pearson's Chi-squared test
##
## data:  candeath
## X-squared = 197.62, df = 6, p-value < 2.2e-16
```

```
# The Pearson residuals
chisq.summary$residuals
```

```
##        Location of death
## Age           Home Acute Care Chronic care
##   15-54  0.3989527  2.3587229    -5.798909
##   55-64  0.2205584  2.5273526    -5.982375
##   65-74  1.1176594 -0.3297027    -0.500057
##   75+   -1.5530094 -3.6183388     9.946704
```

```
# The sum of the squared residuals is the chi-squared statistic:
chisq.summary$residuals^2
```

```
##        Location of death
## Age           Home Acute Care Chronic care
##   15-54 0.1591633  5.5635737    33.627351
##   55-64 0.0486460  6.3875111    35.788805
##   65-74 1.2491626  0.1087039     0.250057
##   75+   2.4118382 13.0923756    98.936922
```

```
sum(chisq.summary$residuals^2)
```

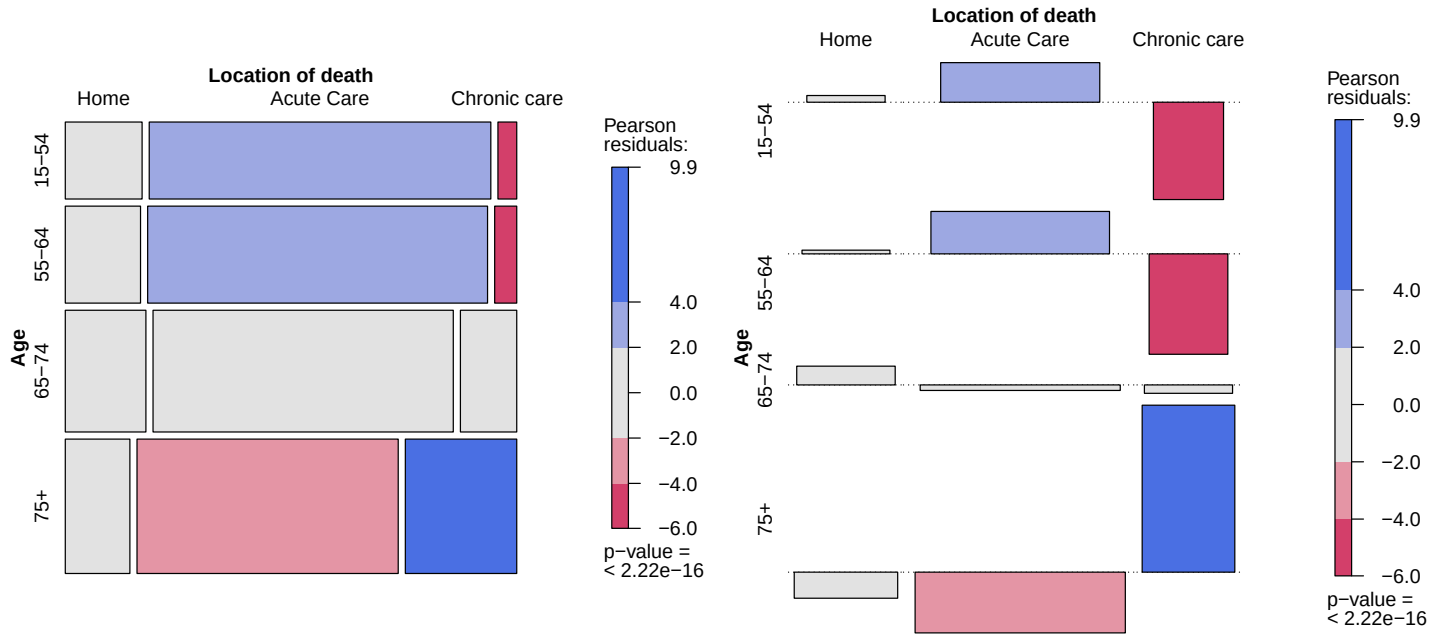
```
## [1] 197.6241
```

A visualization of the Pearson residuals is available with a `mosaic()` plot in the `vcd` package. Extended mosaic and association plots are each helpful methods of visualing complex data and evaluating deviations from a specified independence model. For extended mosaic plots, use `mosaic(x, condvar=, data=)` where `x` is a table or formula, `condvar=` is an optional conditioning variable, and `data=` specifies a data frame or a table. Include `shade=TRUE` to color the figure, and `legend=TRUE` to display a legend for the Pearson residuals.

```
# mosaic plot
library(vcd)
```

```
## Loading required package:  grid
##
## Attaching package:  'vcd'
## The following object is masked from 'package:BSDA':
##
##     Trucks
```

```
mosaic(candeath, shade=TRUE, legend=TRUE)

# association plot
library(vcd)
assoc(candeath, shade=TRUE)
```

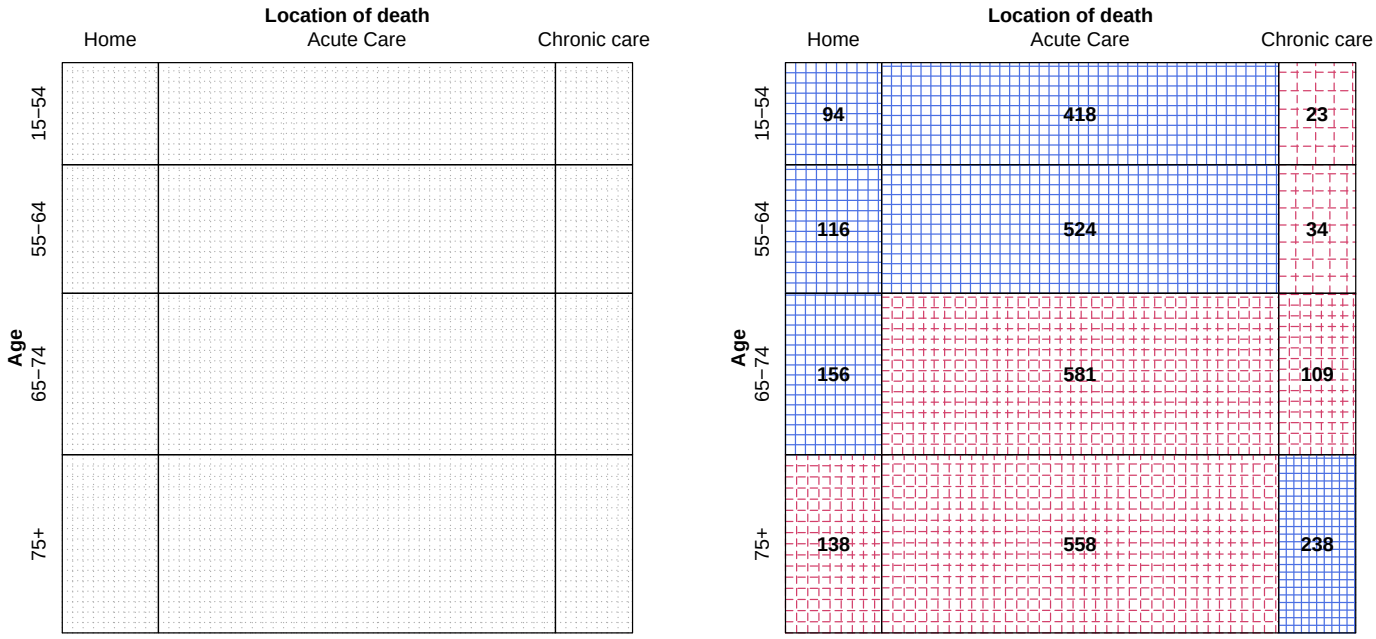


The `vcd` package provides a variety of methods for visualizing multivariate categorical data, inspired by Michael Friendly's wonderful "Visualizing Categorical Data". For more details, see The Strucplot Framework[4].

For example, a sieve plot for an n-way contingency table plots rectangles with areas proportional to the expected cell frequencies and filled with a number of squares equal to the observed frequencies. Thus, the densities visualize the deviations of the observed from the expected values.

```
# sieve plot
library(vcd)

# plot expected values
sieve(candeath, sievetype = "expected", shade = TRUE)

# plot observed table, then label cells with observed values in the cells
sieve(candeath, pop = FALSE, shade = TRUE)
labeling_cells(text = candeath, gp_text = gpar(fontface = 2))(as.table(candeath))
```

[4] http://cran.r-project.org/web/packages/vcd/vignettes/strucplot.pdf



7.8.1 Adequacy of the Chi-Square Approximation

The chi-squared tests for homogeneity and independence are large-sample tests. As with the goodness-of-fit test, a simple rule-of-thumb is that the approximation is adequate when the expected (not observed) cell counts are 5 or more. This rule is conservative, and some statisticians argue that the approximation is valid for expected counts as small as one.

In practice, the chi-squared approximation to χ^2_s tends to be a bit conservative, meaning that statistically significant results would likely retain significance had a more accurate approximation been used.

R may not print out a warning message whenever a noticeable percentage of cells have expected counts less than 5. Ideally, one would use Fisher's exact test (`fisher.test()`) for tables with small counts, and can be used for larger than 2-by-2 tables when the frequencies are not too large.

7.9 Testing for Homogeneity in Cross-Sectional and Stratified Studies

Two-way tables of counts are often collected using either **stratified sampling** or **cross-sectional** sampling.

In a **stratified design**, distinct groups, strata, or sub-populations are identified. Independent samples are selected from each group, and the sampled individuals are classified into categories. The Indian gaming example is an illustration of a stratified design (where the two strata were NM and CO voters). Stratified designs provide estimates for the strata (population) proportion in each of the categories. A test for **homogeneity of proportions** is used to compare the strata.

In a **cross-sectional design**, individuals are randomly selected from a population and classified by the levels of **two** categorical variables. With cross-sectional samples you can test homogeneity of proportions by comparing either the row proportions or by comparing the column proportions.

Example, antismoking adverts The following data (*The Journal of Advertising*, 1983, pp. 34–42) are from a cross-sectional study that involved soliciting opinions on anti-smoking advertisements. Each subject was asked whether they smoked and their reaction (on a five-point ordinal scale) to the ad. The data are summarized as a two-way table of counts, given below:

	Str. Dislike	Dislike	Neutral	Like	Str. Like	Row Tot
Smoker	8	14	35	21	19	97
Non-smoker	31	42	78	61	69	281
Col Total	39	56	113	82	88	378

The row proportions (i.e., fix a row and compute the proportions for the column categories) are

(Row Prop)	Str. Dislike	Dislike	Neutral	Like	Str. Like	Row Tot
Smoker	0.082	0.144	0.361	0.216	0.196	1.000
Non-smoker	0.110	0.149	0.278	0.217	0.245	1.000

For example, the entry for the (Smoker, Str. Dislike) cell is: $8/97 = 0.082$. Similarly, the column proportions are

(Col Prop)	Str. Dislike	Dislike	Neutral	Like	Str. Like
Smoker	0.205	0.250	0.310	0.256	0.216
Non-smoker	0.795	0.750	0.690	0.744	0.784
Total	1.000	1.000	1.000	1.000	1.000

Although it may be more natural to compare the smoker and non-smoker row proportions, the column proportions can be compared across ad responses. There is no advantage to comparing "rows" instead of "columns" in a formal test of homogeneity of proportions with cross-sectional data. The Pearson chi-squared test treats the rows and columns interchangeably, so you get the same result regardless of how you view the comparison. However, one of the two comparisons may be more natural to interpret.

Note that checking for homogeneity of proportions is meaningful in stratified studies only when the comparison is between strata! Further, if the strata correspond to columns of the table, then the column proportions or percentages are meaningful whereas the row proportions are not.

7.9.1 Testing for Independence in a Two-Way Contingency Table

The row and column classifications for a population where each individual is cross-classified by two categorical variables are said to be **independent** if each **population** cell proportion in the two-way table is the product of the proportion in a given row and the proportion in a given column. One can show that independence is equivalent to homogeneity of proportions. In particular, the two-way table of population cell proportions satisfies independence if and only if the population column proportions are homogeneous. If the population column proportions are homogeneous then so are the population row proportions.

This suggests that a test for independence or **no association** between two variables based on a cross-sectional study can be implemented using the

chi-squared test for homogeneity of proportions. This suggestion is correct. If independence is not plausible, I interpret the dependence as a deviation from homogeneity, using the classification for which the interpretation is most natural.

The Pearson chi-squared test of independence is not significant (p-value = 0.56). The observed association between smoking status and the ad reaction is not significant. This suggests, for example, that the smoker's reactions to the ad were not statistically significantly different from the non-smoker's reactions, which is consistent with the smokers and non-smokers attitudes being fairly similar.

```
#### Example, antismoking adverts
antismokead <-
matrix(c(  8, 14,  35, 21, 19,
          31, 42,  78, 61, 69),
       nrow = 2, byrow = TRUE,
       dimnames = list(
         "Status" = c("Smoker", "Non-smoker"),
         "Reaction" = c("Str. Dislike", "Dislike", "Neutral", "Like", "Str. Like")))
antismokead
```

```
##             Reaction
## Status       Str. Dislike Dislike Neutral Like Str. Like
##   Smoker                8      14      35   21         19
##   Non-smoker           31      42      78   61         69
```

```
chisq.summary <- chisq.test(antismokead, correct=FALSE)
chisq.summary
```

```
##
##  Pearson's Chi-squared test
##
## data:  antismokead
## X-squared = 2.9907, df = 4, p-value = 0.5594
```

```
# All expected frequencies are at least 5
chisq.summary$expected
```

```
##             Reaction
## Status       Str. Dislike  Dislike  Neutral     Like Str. Like
##   Smoker         10.00794 14.37037 28.99735 21.04233  22.58201
##   Non-smoker     28.99206 41.62963 84.00265 60.95767  65.41799
```

```
# Contribution to chi-squared statistic
chisq.summary$residuals^2
```

```
##             Reaction
## Status       Str. Dislike     Dislike   Neutral         Like
##   Smoker        0.4028612 0.009545628 1.2425876 8.514567e-05
```

```
##   Non-smoker    0.1390660 0.003295110 0.4289359 2.939192e-05
##             Reaction
## Status       Str. Like
##   Smoker     0.5681868
##   Non-smoker 0.1961356
```

7.9.2 Further Analyses in Two-Way Tables

The χ^2_s statistic is a **summary measure** of independence or homogeneity. A careful look at the data usually reveals the nature of the **association** or **heterogeneity** when the test is significant. There are numerous meaningful ways to explore two-way tables to identify sources of association or heterogeneity. For example, in the comparison of age distributions across locations, you might consider the 4×2 tables comparing all possible pairs of locations. Another possibility would be to compare the proportion in the 75+ age category across locations. For the second comparison you need a 2×3 table of counts, where the two rows correspond to the individuals less than 75 years old and those 75+ years old, respectively (i.e., collapse the first three rows of the original 4×2 table). The possibilities are almost limitless in large tables. Of course, theoretically generated comparisons are preferred to data dredging.

Example: drugs and nausea, testing for homogeneity A randomized double-blind experiment compared the effectiveness of several drugs in reducing postoperative nausea. All patients were anesthetized with nitrous oxide and ether. The following table shows the incidence of nausea during the first four postoperative hours of four drugs and a placebo. Compare the drugs to each other and to the placebo.

Drug	# with Nausea	# without Nausea	Sample Size
Placebo	96	70	166
Chlorpromazine	52	100	152
Dimenhydrinate	52	33	85
Pentobarbitol (100mg)	35	32	67
Pentobarbitol (150mg)	37	48	85

Let p_{PL} be the probability of developing nausea given a placebo, and define p_{CH}, p_{DI}, p_{PE100}, and p_{PE150} analogously. A simple initial analysis would be to test homogeneity of proportions: $H_0 : p_{\text{PL}} = p_{\text{CH}} = p_{\text{DI}} = p_{\text{PE100}} = p_{\text{PE150}}$ against H_A : not H_0.

The data were entered as frequencies. The output shows that the proportion of patients exhibiting nausea (see the **column** percents — the cell and row percentages are not interpretable, so they are omitted) is noticeably different across drugs. In particular, Chlorpromazine is the most effective treatment with $\hat{p}_{\text{CH}} = 0.34$ and Dimenhydrinate is the least effective with $\hat{p}_{\text{DI}} = 0.61$.

The p-value for the chi-squared test is 0.00005, which leads to rejecting H_0 at the 0.05 or 0.01 levels. The data strongly suggest there are differences in the effectiveness of the various treatments for postoperative nausea.

```
#### Example: drugs and nausea, testing for homogeneity
nausea <-
matrix(c(96,  70,   52, 100,   52,  33,   35,  32,   37,  48),
       nrow = 5, byrow = TRUE,
       dimnames = list("Drug" = c("PL", "CH", "DI", "PE100", "PE150"),
                       "Result" = c("Nausea", "No Nausea")))
nausea
##        Result
## Drug    Nausea No Nausea
##   PL        96        70
##   CH        52       100
##   DI        52        33
##   PE100     35        32
##   PE150     37        48
# Sorted proportions of nausea by drug
nausea.prop <- sort(nausea[,1]/rowSums(nausea))
nausea.prop
##        CH     PE150     PE100        PL        DI
## 0.3421053 0.4352941 0.5223881 0.5783133 0.6117647
# chi-sq test of association
chisq.summary <- chisq.test(nausea, correct=FALSE)
chisq.summary
##
##  Pearson's Chi-squared test
##
## data:  nausea
## X-squared = 24.827, df = 4, p-value = 5.451e-05
# All expected frequencies are at least 5
chisq.summary$expected
```

```
##        Result
## Drug      Nausea No Nausea
##   PL    81.35495  84.64505
##   CH    74.49369  77.50631
##   DI    41.65766  43.34234
##   PE100 32.83604  34.16396
##   PE150 41.65766  43.34234
```

A sensible follow-up analysis is to identify which treatments were responsible for the significant differences. For example, the placebo and chlorpromazine can be compared using a test of $p_{\text{PL}} = p_{\text{CH}}$ or with a CI for $p_{\text{PL}} - p_{\text{CH}}$.

In certain experiments, specific comparisons are of interest, for example a comparison of the drugs with the placebo. Alternatively, all possible comparisons might be deemed relevant. The second case is suggested here based on the problem description. I will use a Bonferroni adjustment to account for the multiple comparisons. The Bonferroni adjustment accounts for data dredging, but at a cost of less sensitive comparisons.

There are 10 possible comparisons here. The Bonferroni analysis with an overall Family Error Rate of 0.05 (or less) tests the 10 individual hypotheses at the 0.05/10=0.005 level.

```
nausea.table <- data.frame(Interval = rep(NA,10)
                         , CI.lower = rep(NA,10)
                         , CI.upper = rep(NA,10)
                         , Z        = rep(NA,10)
                         , p.value  = rep(NA,10)
                         , sig.temp = rep(NA,10)
                         , sig      = rep(NA,10))
# row names for table
nausea.table[,1] <- c("p_PL - p_CH"
                    , "p_PL - p_DI"
                    , "p_PL - p_PE100"
                    , "p_PL - p_PE150"
                    , "p_CH - p_DI"
                    , "p_CH - p_PE100"
                    , "p_CH - p_PE150"
                    , "p_DI - p_PE100"
                    , "p_DI - p_PE150"
                    , "p_PE100 - p_PE150")
# test results together in a table
i.tab <- 0
for (i in 1:4) {
  for (j in (i+1):5) {
    i.tab <- i.tab + 1
    nausea.summary <- prop.test(nausea[c(i,j),], correct = FALSE, conf.level = 1-0.05/10)
    nausea.table[i.tab, 2:6] <- c(nausea.summary$conf.int[1]
                                , nausea.summary$conf.int[2]
                                , sign(-diff(nausea.summary$estimate)) * nausea.summary$statistic^0.5
                                , nausea.summary$p.value
                                , (nausea.summary$p.value < 0.05/10))
    if (nausea.table$sig.temp[i.tab] == 1) { nausea.table$sig[i.tab] <- "*" }
                                      else { nausea.table$sig[i.tab] <- " " }
  }
}
```

```
# remove temporary sig indicator
nausea.table <- subset(nausea.table, select = -sig.temp)
#nausea.table
```

The following table gives two-sample tests of proportions with nausea and 99.5% CIs for the differences between the ten pairs of proportions. The only two p-values are less than 0.005 corresponding to $p_{\text{PL}} - p_{\text{CH}}$ and $p_{\text{CH}} - p_{\text{DI}}$. I am 99.5% confident that p_{CH} is between 0.084 and 0.389 less than p_{PL}, and I am 99.5% confident that p_{CH} is between 0.086 and 0.453 less than p_{DI}. The other differences are not significant.

	Interval	CI.lower	CI.upper	Z	p.value	sig
1	p_PL - p_CH	0.0838	0.3887	4.2182	0.0000	*
2	p_PL - p_DI	−0.2167	0.1498	−0.5099	0.6101	
3	p_PL - p_PE100	−0.1464	0.2582	0.7788	0.4361	
4	p_PL - p_PE150	−0.0424	0.3284	2.1485	0.0317	
5	p_CH - p_DI	−0.4532	−0.0861	−4.0122	0.0001	*
6	p_CH - p_PE100	−0.3828	0.0222	−2.5124	0.0120	
7	p_CH - p_PE150	−0.2788	0.0924	−1.4208	0.1554	
8	p_DI - p_PE100	−0.1372	0.3160	1.1058	0.2688	
9	p_DI - p_PE150	−0.0352	0.3881	2.3034	0.0213	
10	p_PE100 - p_PE150	−0.1412	0.3154	1.0677	0.2857	

Using ANOVA-type groupings, and arranging the treatments from most to least effective (low proportions to high), we get:

```
CH (0.34)   PE150 (0.44)   PE100 (0.52)   PL (0.58)   DI (0.61)
-----------------------------------
            ---------------------------------------------------
```

Chapter 8

Correlation and Regression

Contents

```
## Warning in file(filename, "r", encoding = encoding): cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding): cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

select graphical displays that reveal the relationship between two continuous variables.

summarize model fit.

interpret model parameters, such as slope and R^2.

assess the model assumptions visually and numerically.

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

1. organize knowledge.

5. define parameters of interest and hypotheses in words and notation.

6. summarize data visually, numerically, and descriptively.

8. use statistical software.
12. make evidence-based decisions.

8.1 Introduction

Suppose we select $n = 10$ people from the population of college seniors who plan to take the medical college admission test (MCAT) exam. Each takes the test, is coached, and then retakes the exam. Let X_i be the pre-coaching score and let Y_i be the post-coaching score for the i^{th} individual, $i = 1, 2, \ldots, n$. There are several questions of potential interest here, for example: Are Y and X related (associated), and how? Does coaching improve your MCAT score? Can we use the data to develop a mathematical model (formula) for predicting post-coaching scores from the pre-coaching scores? These questions can be addressed using **correlation** and **regression** models.

The **correlation coefficient** is a standard measure of **association** or relationship between two features Y and X. Most scientists equate Y and X being correlated to mean that Y and X are associated, related, or **dependent** upon each other. However, correlation is only a measure of the strength of a **linear relationship.** For later reference, let ρ be the correlation between Y and X in the population and let r be the sample correlation. I define r below. The population correlation is defined analogously from population data.

Suppose each of n sampled individuals is measured on two quantitative characteristics called Y and X. The data are pairs of observations (X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n), where (X_i, Y_i) is the (X, Y) pair for the i^{th} individual in the sample. The sample correlation between Y and X, also called the **Pearson product moment correlation coefficient**, is

$$r = \frac{S_{XY}}{S_X S_Y} = \frac{\sum_i (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_i (X_i - \bar{X})^2 \sum_i (Y_i - \bar{Y})^2}},$$

where

$$S_{XY} = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{n - 1}$$

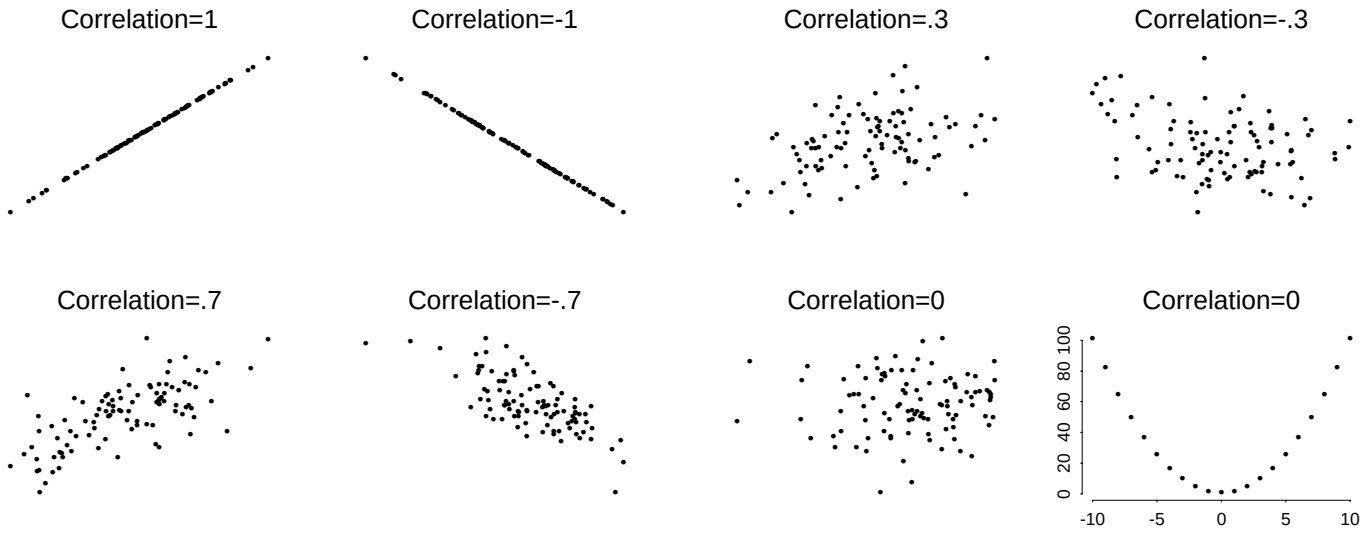
is the **sample covariance** between Y and X, and $S_Y = \sqrt{\sum_i (Y_i - \bar{Y})^2/(n-1)}$ and $S_X = \sqrt{\sum_i (X_i - \bar{X})^2/(n-1)}$ are the standard deviations for the Y and X samples.

Important properties of r:

1. $-1 \leq r \leq 1$.
2. If Y_i tends to increase linearly with X_i then $r > 0$.
3. If Y_i tends to decrease linearly with X_i then $r < 0$.
4. If there is a perfect linear relationship between Y_i and X_i with a positive slope then $r = +1$.
5. If there is a perfect linear relationship between Y_i and X_i with a negative slope then $r = -1$.
6. The closer the points (X_i, Y_i) come to forming a straight line, the closer r is to ± 1.
7. The magnitude of r is unchanged if either the X or Y sample is transformed linearly (such as feet to inches, pounds to kilograms, Celsius to Fahrenheit).
8. The correlation does not depend on which variable is called Y and which is called X.

If r is near ± 1, then there is a strong linear relationship between Y and X in the sample. This suggests we might be able to accurately predict Y from X with a linear equation (i.e., linear regression). If r is near 0, there is a weak linear relationship between Y and X, which suggests that a linear equation provides little help for predicting Y from X. The pictures below should help you develop a sense about the size of r.

Note that $r = 0$ does not imply that Y and X are not related in the sample. It only implies they are not linearly related. For example, in the last plot $r = 0$ yet $Y_i = X_i^2$, exactly.



■ CLICKER Qs — Correlation coefficients STT.02.02.010 ■

■ CLICKER Qs — Strong correlation STT.02.02.020 ■

8.2 Logarithmic transformations

Logarithms[1] are useful for understanding data, partly because they allow numbers that vary by several orders of magnitude to be viewed on a common scale, and more importantly because they allow exponential and power-law relations to be transformed into linearity.

8.2.1 Log-linear and log-log relationships: amoebas, squares, and cubes

Suppose you have an amoeba that takes one hour to divide, and then the two amoebas each divide in one more hour, and so forth. What is the equation of

[1]From: Gelman, Andrew and Deborah Nolan (2002). *Teaching statistics: A bag of tricks.* Oxford University Press.

the number of amoebas, y, as a function of time, x (in hours)? It can be written as $y = 2^x$ or, on the logarithmic scale, $\log(y) = (\log(2))x = 0.30x$.

Suppose you have the same example, but the amoeba takes three hours to divide at each step. Then the number of amoebas y after time x has the equation, $y = 2^{x/3} = (2^{1/3})^x = 1.26^x$ or, on the logarithmic scale, $\log(y) = (\log(1.26))x = 0.10x$. The slope of 0.10 is one-third the earlier slope of 0.30 because the population is growing at one-third the rate.

In the example of exponential growth of amoebas, y is logged while x remains the same. For power-law relations, it makes sense to log both x and y. How does the area of a square relate to its circumference (perimeter)? If the side of the cube has length L, then the area is L^2 and the circumference is $4L$; thus

$$\text{area} \;=\; (\text{circumference}/4)^2.$$

Taking the logarithm of both sides yields,

$$\begin{aligned} \log(\text{area}) \;&=\; 2(\log(\text{circumference}) - \log(4)) \\ \log(\text{area}) \;&=\; -1.20 + 2\log(\text{circumference}), \end{aligned}$$

a linear relation on the log-log scale.

What is the relation between the surface area and volume of a cube? In terms of the side length L, are $6L^2$ and L^3, respectively. On the original scale, this is

$$\text{surface area} \;=\; 6(\text{volume})^{2/3},$$

or, on the logarithmic scale,

$$\log(\text{surface area}) \;=\; \log(6) + (2/3)\log(\text{volume}).$$

Example: Log-linear transformation: world population Consider the world population from the year 0 to 2000. Compare the data in the table below with the plots on the original and logarithmic scales; again, the plots reveals the pattern much clearer than the table.

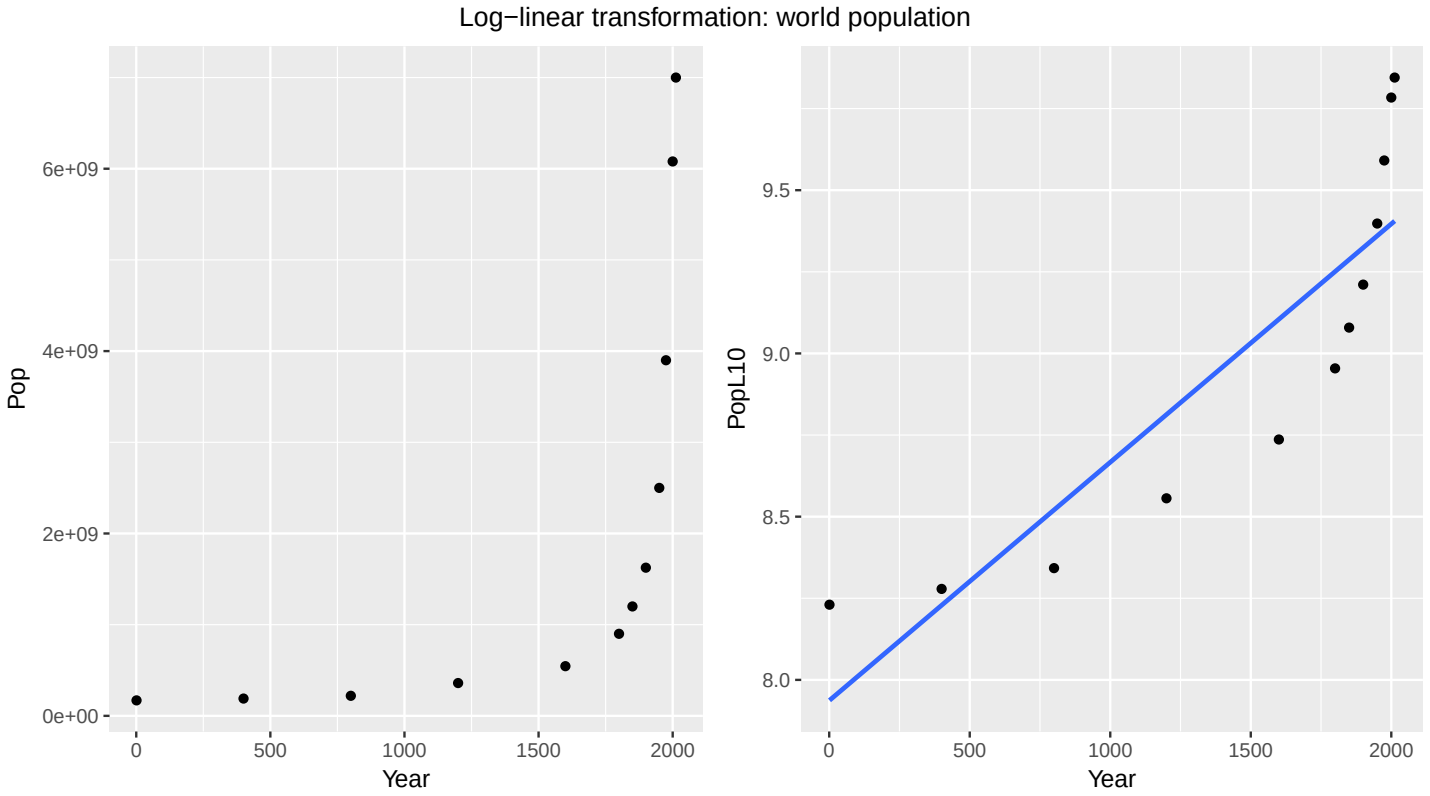
On the raw scale, all you can see is that the population has increased very fast recently. On the log scale, convex curvature is apparent — that is, the rate of increase has itself increased. On the logarithmic graph, the least-squares regression line is drawn. The estimated world population has been growing even faster than exponentially! What would you guess the population to be fore year 1400? Would you be surprised that it was 350 million? It is actually lower than the year 1200 population, because of plague and other factors. This is an illustration that even interpolation can sometimes go awry.

```
#### Example: Log-linear population growth
pop <- read.table(text="
Year Pop_M
   1   170
 400   190
 800   220
1200   360
1600   545
1800   900
1850  1200
1900  1625
1950  2500
1975  3900
2000  6080
2012  7000
", header=TRUE)
pop$Pop    <- 1e6 * pop$Pop_M  # convert to millions
pop$PopL10 <- log10(pop$Pop)

# calculate the residuals from a simple linear regression
lm.fit <- lm(PopL10 ~ Year, data = pop)
# include those residuals in the pop table
pop$Res <- residuals(lm.fit)
pop$R10 <- 10^residuals(lm.fit) # residuals on the original scale
```

	Year	Pop_M	Pop	PopL10	Res	R10
1	1	170	170000000	8.230	0.293	1.964
2	400	190	190000000	8.279	0.050	1.122
3	800	220	220000000	8.342	−0.178	0.663
4	1200	360	360000000	8.556	−0.256	0.554
5	1600	545	545000000	8.736	−0.368	0.428
6	1800	900	900000000	8.954	−0.296	0.505
7	1850	1200	1200000000	9.079	−0.208	0.619
8	1900	1625	1625000000	9.211	−0.113	0.771
9	1950	2500	2500000000	9.398	0.038	1.091
10	1975	3900	3900000000	9.591	0.213	1.631
11	2000	6080	6080000000	9.784	0.387	2.439
12	2012	7000	7000000000	9.845	0.440	2.752

```
library(ggplot2)

p1 <- ggplot(pop, aes(x = Year, y = Pop))
p1 <- p1 + geom_point()
#print(p1)
```

```
p2 <- ggplot(pop, aes(x = Year, y = PopL10))
p2 <- p2 + geom_point()
p2 <- p2 + geom_smooth(method = lm, se = FALSE)
#print(p2)

library(gridExtra)
grid.arrange(grobs = list(p1, p2), nrow=1
             , top = "Log-linear transformation: world population")
```



When using data of this nature, consider the source of the population numbers. How would you estimate the population of the world in the year 1?

Example: Log-log transformation: metabolic rates A rich source of examples when covering log-log transformations are biological scaling relations[2]. The relationship[3] between body mass (M, g) and basal metabolic rate (BMR, ml of O_2 per h (similar to Watts?)) for mammalian orders for selected data are summarized in plots below, both on the original and log-log scales. The linear regression summarizes the dark points, the mean for each of the species groups,

[2]One of the world experts in allometric scaling is Prof. Jim Brown, UNM Biology, `http://biology.unm.edu/jhbrown`

[3]White and Seymour (2003) PNAS, 10.1073/pnas.0436428100

and the colored points are individual species. The curved regression is found by inverting the linear regression onto the original scale. The third plot displays the log axes with values on the original scale.

```
# http://www.ncbi.nlm.nih.gov/pmc/articles/PMC153045/
# Supp:
# http://www.ncbi.nlm.nih.gov/pmc/articles/PMC153045/bin/pnas_0436428100_index.html
# Supporting information for White and Seymour (2003)
#   Proc. Natl. Acad. Sci. USA, 10.1073/pnas.0436428100

library(gdata)
```

```
## gdata:  read.xls support for 'XLS' (Excel 97-2004) files
## gdata:  ENABLED.
##
## gdata:  read.xls support for 'XLSX' (Excel 2007+) files
## gdata:  ENABLED.
##
## Attaching package:  'gdata'
## The following object is masked from 'package:gridExtra':
##
##    combine
## The following object is masked from 'package:stats':
##
##    nobs
## The following object is masked from 'package:utils':
##
##    object.size
## The following object is masked from 'package:base':
##
##    startsWith
```

```
fn <- "data/ADA1_notes_08_data_log-logScaling_BodyMassMetabolicRate_2003_WhiteSeymour.xlsx"
bm.bmr <- read.xls(fn, skip = 4, stringsAsFactors = TRUE)
bm.bmr$Log10BodyMass <- log10(bm.bmr$BodyMass)
bm.bmr$Log10BaseMetRate <- log10(bm.bmr$BaseMetRate)

# remove a very strange group
bm.bmr <- subset(bm.bmr, !(Group == "Artiodactyla 7"))
str(bm.bmr)
```

```
## 'data.frame': 634 obs. of  9 variables:
##  $ Group           : Factor w/ 18 levels "Artiodactyla 7",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Genus           : Factor w/ 88 levels "","Acrobatidae",..: 1 12 12 12 12 12 12 12 12 31 ...
##  $ Species         : Factor w/ 621 levels "","2n = 52","2n = 54",..: 1 22 67 68 79 206 614 615 616 8 ...
##  $ BodyMass        : num  4452 3600 10000 7720 5444 ...
##  $ T               : num  37.5 38.6 37 38 38.2 38.8 38 38.7 NA 39 ...
##  $ BaseMetRate     : num  1244 1374 2687 3860 1524 ...
##  $ Ref             : Factor w/ 239 levels "","1","10","100",..: 1 230 3 15 24 37 3 50 61 72 ...
##  $ Log10BodyMass   : num  3.65 3.56 4 3.89 3.74 ...
##  $ Log10BaseMetRate: num  3.09 3.14 3.43 3.59 3.18 ...
```

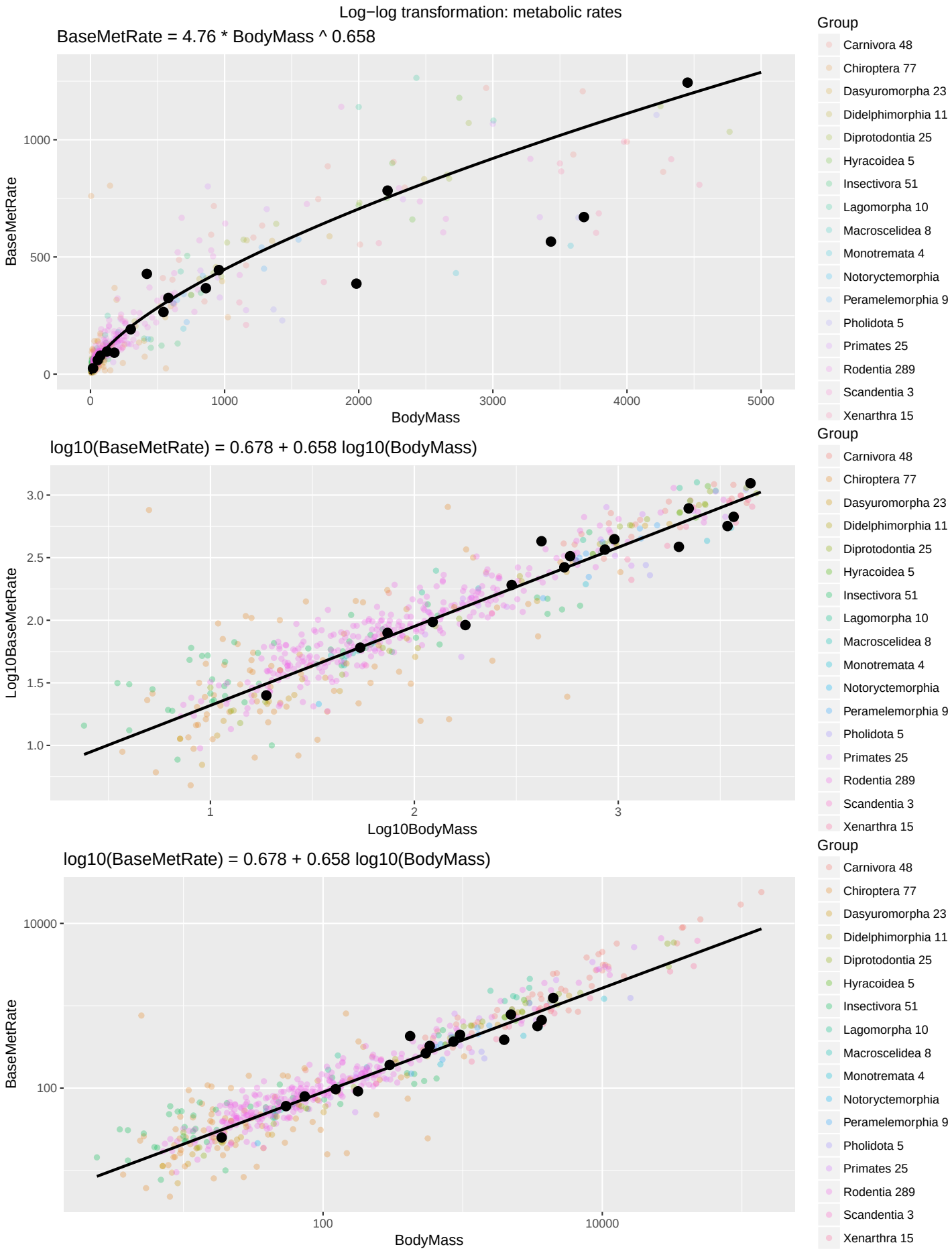
```
# log-log scale linear regression
lm.fit <- lm(Log10BaseMetRate ~ Log10BodyMass, data = bm.bmr)
# coefficients for regression line
coef(lm.fit)
```

```
##   (Intercept) Log10BodyMass
##     0.6775600     0.6575572
```

```
library(ggplot2)

p1 <- ggplot(subset(bm.bmr, (Genus == "")), aes(x = BodyMass, y = BaseMetRate))
p1 <- p1 + geom_point(data = subset(bm.bmr, !(Genus == "")), aes(colour = Group), alpha = 0.2)
p1 <- p1 + geom_point(size = 3)
  # Using a custom function
  f.org.scale <- function(BodyMass) { 10^coef(lm.fit)[1] * BodyMass ^ coef(lm.fit)[2]}
p1 <- p1 + stat_function(fun = f.org.scale, size = 1)
p1 <- p1 + labs(title = paste("BaseMetRate = ", signif(10^coef(lm.fit)[1], 3), " * ", "BodyMass ^ ", signif(coef(lm.fit)[2], 3), sep = "
p1 <- p1 + scale_y_continuous(limits=c(0, 1300))
p1 <- p1 + scale_x_continuous(limits=c(0, 5000))
#print(p1)

p2 <- ggplot(subset(bm.bmr, (Genus == "")), aes(x = Log10BodyMass, y = Log10BaseMetRate))
p2 <- p2 + geom_point(data = subset(bm.bmr, !(Genus == "")), aes(colour = Group), alpha = 0.3)
p2 <- p2 + geom_point(size = 3)
p2 <- p2 + geom_smooth(method = lm, se = FALSE, fullrange = TRUE, size = 1, colour = "black")
```

```
p2 <- p2 + labs(title = paste("log10(BaseMetRate) = ", signif(coef(lm.fit)[1], 3), " + ", signif(coef(lm.fit)[2], 3), " log10(BodyMass)"
p2 <- p2 + scale_y_continuous(limits=c(NA, log10(1300)))
p2 <- p2 + scale_x_continuous(limits=c(NA, log10(5000)))
#print(p2)

p3 <- ggplot(subset(bm.bmr, (Genus == "")), aes(x = BodyMass, y = BaseMetRate))
p3 <- p3 + geom_point(data = subset(bm.bmr, !(Genus == "")), aes(colour = Group), alpha = 0.3)
p3 <- p3 + geom_point(size = 3)
p3 <- p3 + geom_smooth(method = lm, se = FALSE, fullrange = TRUE, size = 1, colour = "black")
p3 <- p3 + labs(title = paste("log10(BaseMetRate) = ", signif(coef(lm.fit)[1], 3), " + ", signif(coef(lm.fit)[2], 3), " log10(BodyMass)"
p3 <- p3 + scale_y_log10()#limits=c(NA, log10(1300)))
p3 <- p3 + scale_x_log10()#limits=c(NA, log10(5000)))
#print(p3)

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3), ncol=1
           , top = "Log-log transformation: metabolic rates")
```

```
## Warning:  Removed 56 rows containing missing values (geom_point).
## Warning:  Removed 5 rows containing missing values (geom_point).
## Warning:  Removed 5 rows containing non-finite values (stat_smooth).
## Warning:  Removed 56 rows containing missing values (geom_point).
## Warning:  Removed 5 rows containing missing values (geom_point).
## Warning:  Removed 5 rows containing non-finite values (stat_smooth).
## Warning:  Removed 1 rows containing missing values (geom_point).
## Warning:  Removed 5 rows containing missing values (geom_point).
```


Log−log transformation: metabolic rates
BaseMetRate = 4.76 * BodyMass ^ 0.658
BaseMetRate
BodyMass
log10(BaseMetRate) = 0.678 + 0.658 log10(BodyMass)
Log10BaseMetRate
Log10BodyMass
log10(BaseMetRate) = 0.678 + 0.658 log10(BodyMass)
BaseMetRate
BodyMass
Group
Carnivora 48
Chiroptera 77
Dasyuromorpha 23
Didelphimorphia 11
Diprotodontia 25
Hyracoidea 5
Insectivora 51
Lagomorpha 10
Macroscelidea 8
Monotremata 4
Notoryctemorphia
Peramelemorphia 9
Pholidota 5
Primates 25
Rodentia 289
Scandentia 3
Xenarthra 15

The table below provides predictions over a range of scales. Note that the smallest mammel in this dataset is about 2.4 grams. A 5-gram mammal uses about 13.7 Watts, so 1000 5-gram mammals use about 13714 Watts. Whereas, one 5000-gram mammal uses 1287 Watts. Thus, larger mammals give off less heat than the equivalent weight of many smaller mammals.

```
pred.bm.bmr <- data.frame(BodyMass = 5 * c(1, 10, 100, 1000))
pred.bm.bmr$Log10BodyMass <- log10(pred.bm.bmr$BodyMass)
pred.bm.bmr$Log10BaseMetRate <- predict(lm.fit, pred.bm.bmr)
pred.bm.bmr$BaseMetRate <- 10^pred.bm.bmr$Log10BaseMetRate

tab.pred <- subset(pred.bm.bmr, select = c("BodyMass", "BaseMetRate"))
```

	BodyMass	BaseMetRate
1	5	13.71
2	50	62.33
3	500	283.33
4	5000	1287.79

We want to focus on the slope in the log-log plot, which is the exponent in original scale plot. On the log scale we have

$$\log(\text{BaseMetRate}) \;=\; 0.678 + 0.658\log(\text{BodyMass}).$$

To interpret the slope, for each unit increase in (predictor, x-variable) log(BodyMass), the expected increase in (response, y-variable) log(BaseMetRate) is 4.55. By exponentiating on both sides, the expression on the original scale is

$$\begin{aligned}\text{BaseMetRate} &= 10^{0.678} \times \text{BodyMass}^{0.658} \\ &= 4.76 \times \text{BodyMass}^{0.658}.\end{aligned}$$

For example, if you multiply body mass by 10, then you multiply metabolic rate by $10^{0.658} = 4.55$. If you multiply body mass by 100, then you multiply metabolic rate by $100^{0.658} = 20.7$, and so forth. The relation between metabolic rate and body mass is less than linear (that is, the exponent 0.658 is less than 1.0, and the line in the original-scal plot curves downward, not upward), which implies that the equivalent mass of small mammals gives off more heat, and the equivalent mass of large mammals gives off less heat.

This seems related to the general geometrical relation that surface area and volume are proportional to linear dimension to the second and third power, respectively, and thus surface area should be proportional to volume to the 2/3 power. Heat produced by a mammal is emitted from its surface, and it would thus be reasonable to suspect metabolic rate to be proportional to the 2/3 power of body mass. Biologists have considered whether the empirical slope is closer to 3/4 or 2/3; the important thing here is to think about log transformations and power laws (and have a chat with Jim Brown or someone from his lab at UNM for the contextual details). As an aside, something not seen from this plot is that males tend to be above the line and females below the line.

8.3 Testing that $\rho = 0$

Suppose you want to test $H_0 : \rho = 0$ against $H_A : \rho \neq 0$, where ρ is the population correlation between Y and X. This test is usually interpreted as a test of no association, or relationship, between Y and X in the population. Keep in mind, however, that ρ measures the strength of a **linear** relationship.

The standard test of $H_0 : \rho = 0$ is based on the magnitude of r. If we let

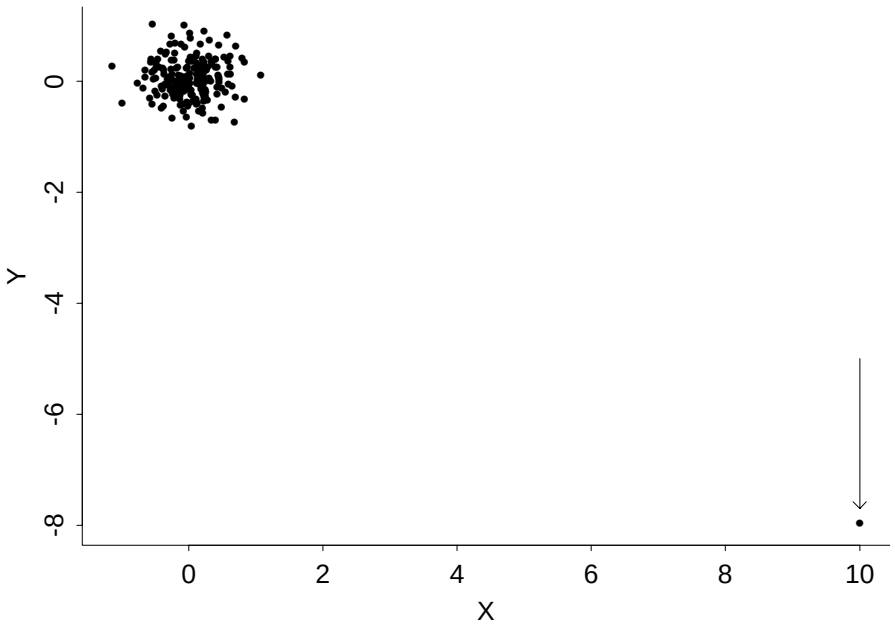
$$t_s = r\sqrt{\frac{n-2}{1-r^2}},$$

then the test rejects H_0 in favor of H_A if $|t_s| \geq t_{\text{crit}}$, where t_{crit} is the two-sided test critical value from a t-distribution with $df = n - 2$. The p-value for the test is the area under the t-curve outside $\pm t_s$ (i.e., two-tailed test p-value).

This test assumes that the data are a random sample from a **bivariate normal population** for $(X,\ Y)$. This assumption implies that all linear combinations of X and Y, say $aX + bY$, are normal. In particular, the (marginal) population frequency curves for X and Y are normal. At a minimum, you should make boxplots of the X and Y samples to check marginal normality. For large-sized samples, a plot of Y against X should be roughly an elliptical cloud, with the density of the points decreasing as the points move away from the center of the cloud.

8.3.1 The Spearman Correlation Coefficient

The Pearson correlation r can be highly influenced by outliers in one or both samples. For example, $r \approx -1$ in the plot below. If you delete the one extreme case with the largest X and smallest Y value then $r \approx 0$. The two analyses are contradictory. The first analysis (ignoring the plot) suggests a strong linear relationship, whereas the second suggests the lack of a linear relationship. I will not strongly argue that you should (must?) delete the extreme case, but I am concerned about any conclusion that depends heavily on the presence of a single observation in the data set.



Spearman's rank correlation coefficient r_S is a sensible alternative to r when normality is unreasonable or outliers are present. Most books give a computational formula for r_S. I will verbally describe how to compute r_S. First, order the X_is and assign them ranks. Then do the same for the Y_is and replace the original data pairs by the pairs of ranked values. The Spearman rank correlation is the Pearson correlation computed from the pairs of ranks.

The Spearman correlation r_S estimates the **population rank correlation coefficient**, which is a measure of the strength of linear relationship between population ranks. The Spearman correlation, as with other rank-based methods, is not sensitive to the presence of outliers in the data (or any information about the marginal distribution of X or Y). In the plot above, $r_S \approx 0$ whether the unusual point is included or excluded from the analysis. In samples without unusual observations and a linear trend, you often find that the Spearman and Pearson correlations are similar, $r_S \approx r$.

An important point to note is that the magnitude of the Spearman correlation does not change if either X or Y or both are transformed (monotonically). Thus, if r_S is noticeably greater than r, a transformation of the data might provide a stronger linear relationship.

Example: Blood loss Eight patients underwent a thyroid operation. Three variables were measured on each patient: weight in kg, time of operation in minutes, and blood loss in ml. The scientists were interested in the factors that influence blood loss.

```
#### Example: Blood loss
thyroid <- read.table(text="
weight time blood_loss
  44.3  105        503
  40.6   80        490
  69.0   86        471
  43.7  112        505
  50.3  109        482
  50.2  100        490
  35.4   96        513
  52.2  120        464
", header=TRUE)

# show the structure of the data.frame
str(thyroid)
```

```
## 'data.frame': 8 obs. of  3 variables:
##  $ weight    : num  44.3 40.6 69 43.7 50.3 50.2 35.4 52.2
##  $ time      : int  105 80 86 112 109 100 96 120
##  $ blood_loss: int  503 490 471 505 482 490 513 464
```

```
# display the data.frame
#thyroid
```

	weight	time	blood_loss
1	44.3	105	503
2	40.6	80	490
3	69.0	86	471
4	43.7	112	505
5	50.3	109	482
6	50.2	100	490
7	35.4	96	513
8	52.2	120	464

Below, we calculate the Pearson correlations between all pairs of variables (left), as well as the p-values (right) for testing whether the correlation is equal to zero.

```
p.corr <- cor(thyroid);
#p.corr

# initialize pvalue table with dim names
p.corr.pval <- p.corr;
for (i1 in 1:ncol(thyroid)) {
  for (i2 in 1:ncol(thyroid)) {
    p.corr.pval[i1,i2] <- cor.test(thyroid[, i1], thyroid[, i2])$p.value
  }
}
#p.corr.pval
```

	weight	time	blood_loss
weight	1.000	−0.066	−0.772
time	−0.066	1.000	−0.107
blood_loss	−0.772	−0.107	1.000

	weight	time	blood_loss
weight	0.0000	0.8761	0.0247
time	0.8761	0.0000	0.8003
blood_loss	0.0247	0.8003	0.0000

Similarly, we calculate the Spearman (rank) correlation table (left), as well as the p-values (right) for testing whether the correlation is equal to zero.

```
s.corr <- cor(thyroid, method = "spearman");
#s.corr

# initialize pvalue table with dim names
s.corr.pval <- p.corr;
for (i1 in 1:ncol(thyroid)) {
  for (i2 in 1:ncol(thyroid)) {
    s.corr.pval[i1,i2] <- cor.test(thyroid[, i1], thyroid[, i2],
                                   method = "spearman")$p.value
  }
}
## Warning in cor.test.default(thyroid[, i1], thyroid[, i2], method = "spearman"):  Cannot
compute exact p-value with ties
```

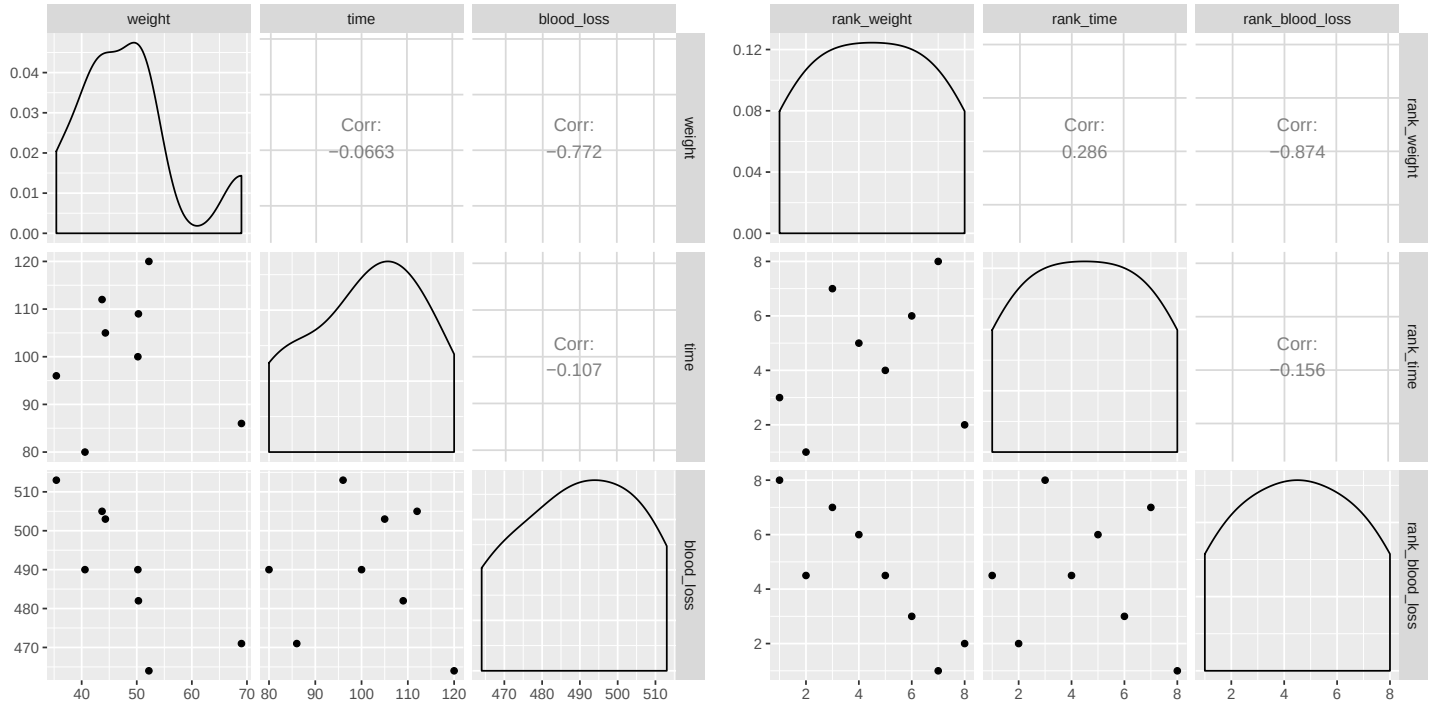
```
## Warning in cor.test.default(thyroid[, i1], thyroid[, i2], method = "spearman"):  Cannot
compute exact p-value with ties
## Warning in cor.test.default(thyroid[, i1], thyroid[, i2], method = "spearman"):  Cannot
compute exact p-value with ties
## Warning in cor.test.default(thyroid[, i1], thyroid[, i2], method = "spearman"):  Cannot
compute exact p-value with ties
## Warning in cor.test.default(thyroid[, i1], thyroid[, i2], method = "spearman"):  Cannot
compute exact p-value with ties
#s.corr.pval
```

	weight	time	blood_loss
weight	1.000	0.286	−0.874
time	0.286	1.000	−0.156
blood_loss	−0.874	−0.156	1.000

	weight	time	blood_loss
weight	0.0000	0.5008	0.0045
time	0.5008	0.0000	0.7128
blood_loss	0.0045	0.7128	0.0000

Here are scatterplots for the original data and the ranks of the data using `ggpairs()` from the `GGally` package with `ggplot2`.

```
# Plot the data using ggplot
library(ggplot2)
library(GGally)
p1 <- ggpairs(thyroid[,1:3])
print(p1)


thyroid$rank_weight     <- rank(thyroid$weight    )
thyroid$rank_time       <- rank(thyroid$time      )
thyroid$rank_blood_loss <- rank(thyroid$blood_loss)

p2 <- ggpairs(thyroid[,4:6])
print(p2)
```



Comments:

1. (Pearson correlations). Blood loss tends to decrease linearly as weight increases, so r should be negative. The output gives $r = -0.77$. There is not much of a linear relationship between blood loss and time, so r should be close to 0. The output gives $r = -0.11$. Similarly, weight and time have a weak negative correlation, $r = -0.07$.
2. The Pearson and Spearman correlations are fairly consistent here. Only the correlation between blood loss and weight is significantly different from zero at the $\alpha = 0.05$ level (the p-values are given below the correlations).
3. (Spearman p-values) R gives the correct p-values. Calculating the p-value using the Pearson correlation on the ranks is not correct, strictly speaking.

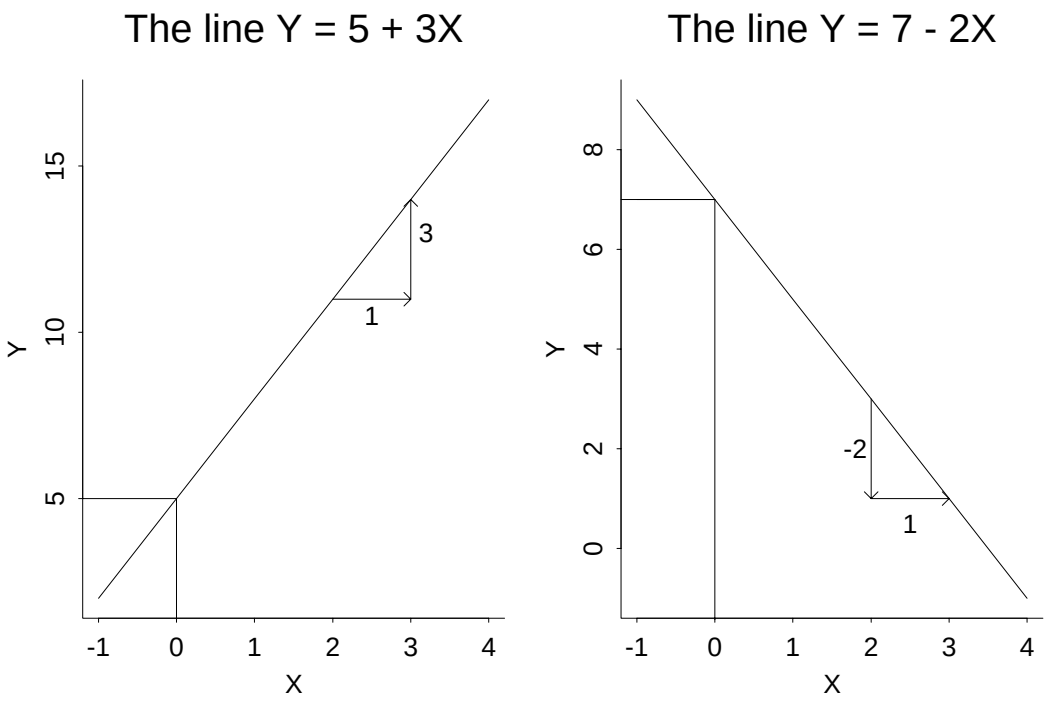
■ ClickerQs — Coney Island STT.02.02.050 ■

8.4 Simple Linear Regression

In linear regression, we are interested in developing a linear equation that best summarizes the relationship in a sample between the **response variable** Y and the **predictor variable** (or **independent variable**) X. The equation is also used to predict Y from X. The variables are not treated symmetrically in regression, but the appropriate choice for the response and predictor is usually apparent.

8.4.1 Linear Equation

If there is a perfect linear relationship between Y and X then $Y = \beta_0 + \beta_1 X$ for some β_0 and β_1, where β_0 is the Y-intercept and β_1 is the slope of the line. Two plots of linear relationships are given below. The left plot has $\beta_0 = 5$ and $\beta_1 = 3$. The slope is positive, which indicates that Y increases linearly when X increases. The right plot has $\beta_0 = 7$ and $\beta_1 = -2$. The slope is negative, which indicates that Y decreases linearly when X increases.



■ CLICKER Qs — Equation STT.02.03.010 ■

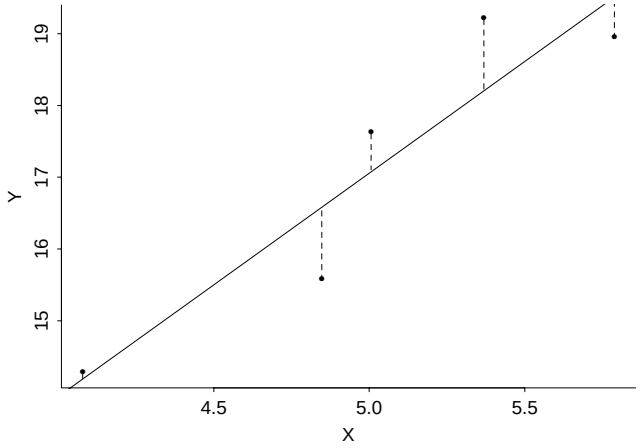
8.4.2 Least Squares

Data rarely, if ever, fall on a straight line. However, a straight line will often describe the **trend** for a set of data. Given a data set, (X_i, Y_i), $i = 1, \ldots, n$, with a **linear trend**, what linear equation "best" summarizes the observed relationship between Y and X? There is no universally accepted definition of "best", but many researchers accept the **Least Squares** line (LS line) as a reasonable summary.

Mathematically, the LS line chooses the values of β_0 and β_1 that minimize

$$\sum_{i=1}^{n} \{Y_i - (\beta_0 + \beta_1 X_i)\}^2$$

over all possible choices of β_0 and β_1. These values can be obtained using calculus. Rather than worry about this calculation, note that the LS line makes the sum of squared (vertical) deviations between the responses Y_i and the line as small as possible, over all possible lines. The LS line goes through the mean point, $(\bar{X}, \bar{Y})$, which is typically in the "the heart" of the data, and is often closely approximated by an eye-ball fit to the data.



The equation of the LS line is

$$\hat{y} = b_0 + b_1 X$$

where the intercept b_0 satisfies

$$b_0 = \bar{Y} - b_1 \bar{X}$$

and the slope is

$$b_1 = \frac{\sum_i (Y_i - \bar{Y})(X_i - \bar{X})}{\sum_i (X_i - \bar{X})^2} = r \frac{S_Y}{S_X}.$$

As before, r is the Pearson correlation between Y and X, whereas S_Y and S_X are the sample standard deviations for the Y and X samples, respectively. The **sign of the slope** and the **sign of the correlation** are **identical** (i.e., + correlation implies + slope).

Special symbols b_0 and b_1 identify the LS intercept and slope to distinguish the LS line from the generic line $Y = \beta_0 + \beta_1 X$. You should think of $\hat{Y}$ as the **fitted value** at X, or the value of the LS line at X.

Fit a regression for the equation estimates from `summary()`. Note that we'll reuse the output of `lm()` over and over again.

```
lm.blood.wt <- lm(blood_loss ~ weight, data = thyroid)
lm.blood.wt
```

```
##
## Call:
## lm(formula = blood_loss ~ weight, data = thyroid)
##
## Coefficients:
## (Intercept)       weight
##       552.4         -1.3
```

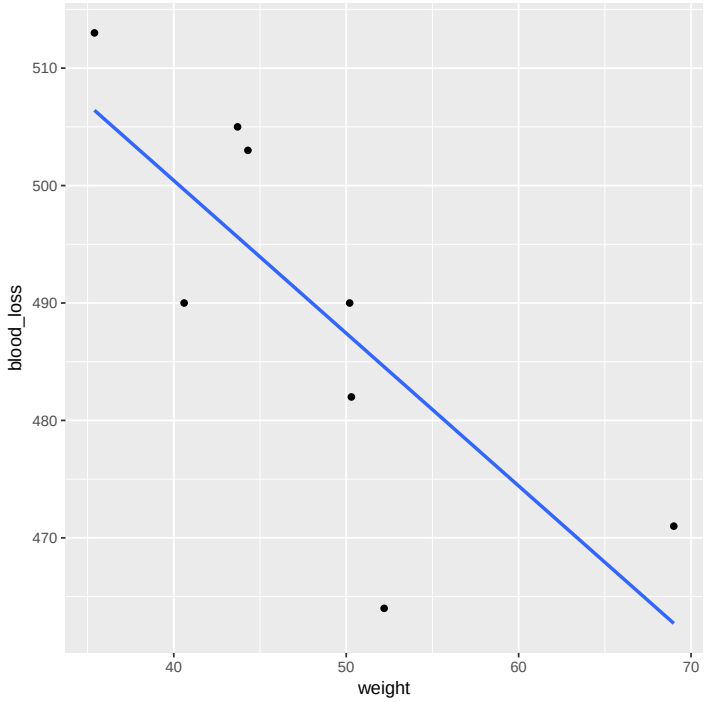
```
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm.blood.wt)
```

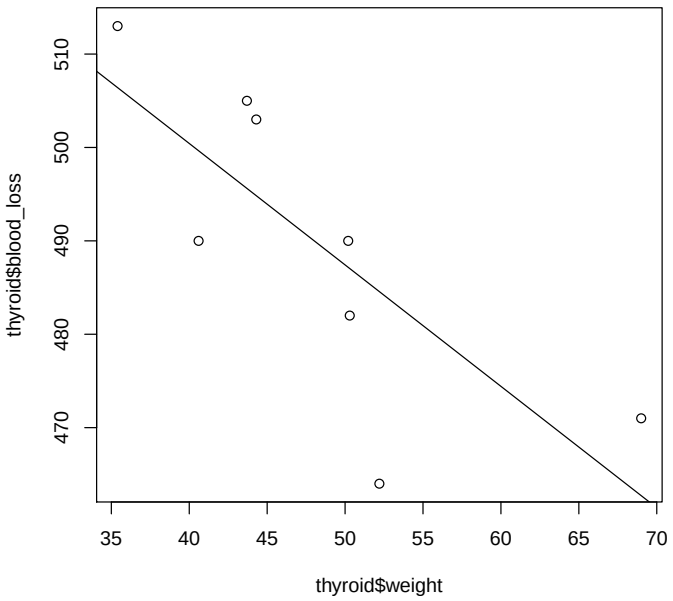
```
##
## Call:
## lm(formula = blood_loss ~ weight, data = thyroid)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.565  -6.189   4.712   8.192   9.382
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 552.4420    21.4409   25.77 2.25e-07 ***
## weight       -1.3003     0.4364   -2.98   0.0247 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.66 on 6 degrees of freedom
## Multiple R-squared:  0.5967,Adjusted R-squared:  0.5295
## F-statistic: 8.878 on 1 and 6 DF,  p-value: 0.02465
```

Create a scatterplot with regression fit.

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(thyroid, aes(x = weight, y = blood_loss))
p <- p + geom_point()
p <- p + geom_smooth(method = lm, se = FALSE)
print(p)


# Base graphics: Plot the data with linear regression fit and confidence bands
# scatterplot
plot(thyroid$weight, thyroid$blood_loss)
# regression line from lm() fit
abline(lm.blood.wt)
```





For the **thyroid operation data** with Y = Blood loss in ml and X = Weight in kg, the LS line is $\hat{Y} = 552.44 - 1.30X$, or Predicted Blood Loss = $552.44 - 1.30$ Weight. For an $86kg$ individual, the Predicted Blood Loss =

$552.44 - 1.30 \times 86 = 440.64ml$.

The LS regression coefficients for this model are interpreted as follows. The intercept b_0 is the predicted blood loss for a 0 kg individual. The intercept has no meaning here. The slope b_1 is the predicted increase in blood loss for each additional kg of weight. The slope is -1.30, so the predicted *decrease* in blood loss is 1.30 ml for each increase of 1 kg in weight.

Any fitted linear relationship holds only approximately and does not necessarily extend outside the range of the data. In particular, nonsensical predicted blood losses of less than zero are obtained at very large weights outside the range of data.

8.5 ANOVA Table for Regression

The LS line minimizes

$$\sum_{i=1}^{n}\{Y_i - (\beta_0 + \beta_1 X_i)\}^2$$

over all choices for β_0 and β_1. Inserting the LS estimates b_0 and b_1 into this expression gives

$$\text{Residual Sums of Squares} = \sum_{i=1}^{n}\{Y_i - (b_0 + b_1 X_i)\}^2.$$

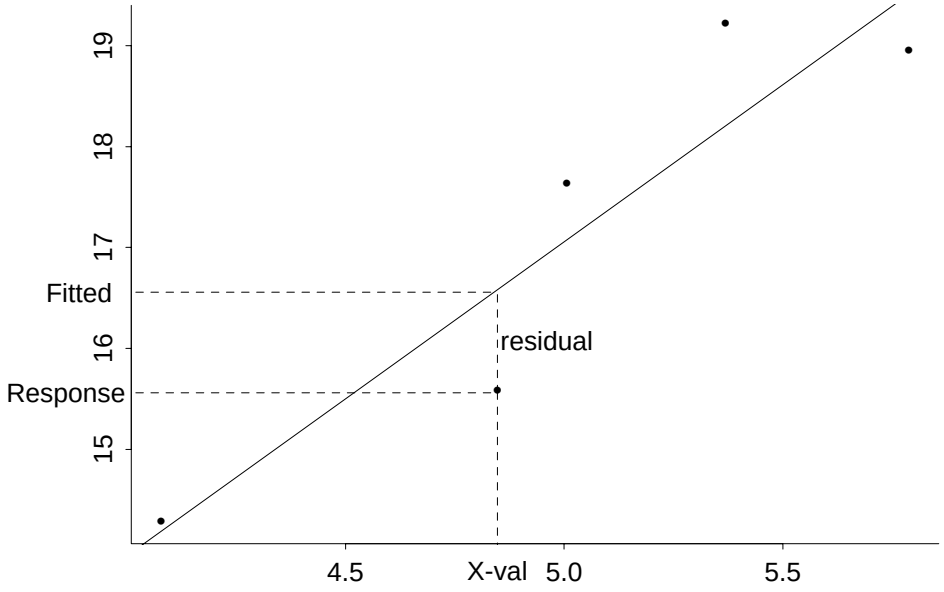
Several bits of notation are needed. Let

$$\hat{Y}_i = b_0 + b_1 X_i$$

be the **predicted** or fitted Y-value for an X-value of X_i and let $e_i = Y_i - \hat{Y}_i$. The fitted value $\hat{Y}_i$ is the value of the LS line at X_i whereas the **residual** e_i is the distance that the observed response Y_i is from the LS line. Given this notation,

$$\text{Residual Sums of Squares} = \text{Res SS } = \sum_{i=1}^{n}(Y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} e_i^2.$$

Here is a picture to clarify matters:



The Residual SS, or sum of squared residuals, is *small* if each $\hat{Y}_i$ is *close to* Y_i (i.e., the line closely fits the data). It can be shown that

$$\text{Total SS in Y} = \sum_{i=1}^{n}(Y_i - \bar{Y})^2 \geq \text{Res SS} \geq 0.$$

Also define

$$\text{Regression SS} = \text{Reg SS} = \text{Total SS} \ - \ \text{Res SS} = b_1 \sum_{i=1}^{n}(Y_i - \bar{Y})(X_i - \bar{X}).$$

The Total SS measures the variability in the Y-sample. Note that

$$0 \leq \text{Regression SS} \leq \text{Total SS}.$$

The percentage of the variability in the Y-sample that is **explained by the linear relationship** between Y and X is

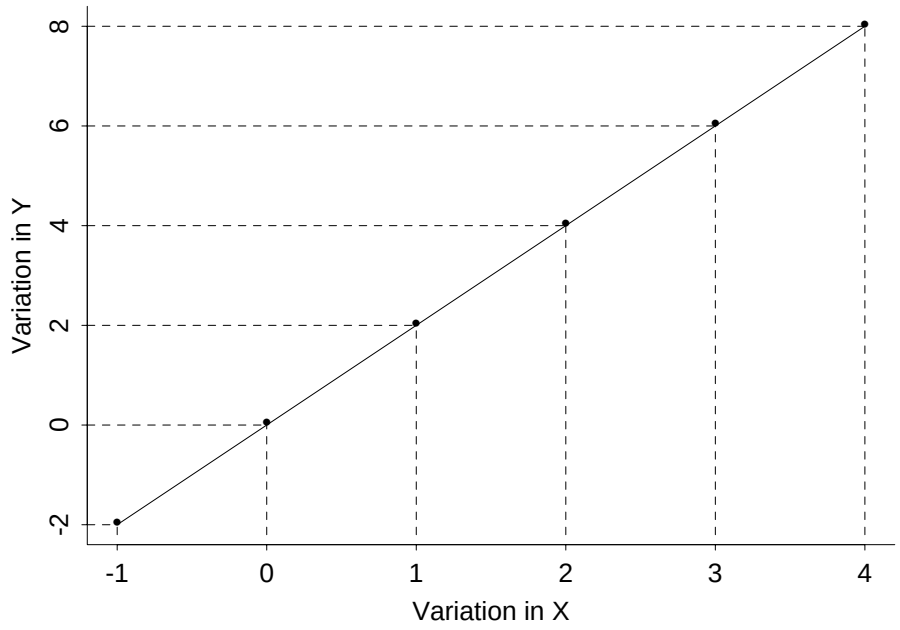
$$R^2 = \text{coefficient of determination} = \frac{\text{Reg SS}}{\text{Total SS}}.$$

Given the definitions of the Sums of Squares, we can show $0 \leq R^2 \leq 1$ and

$$R^2 = \text{square of Pearson correlation coefficient } = r^2.$$

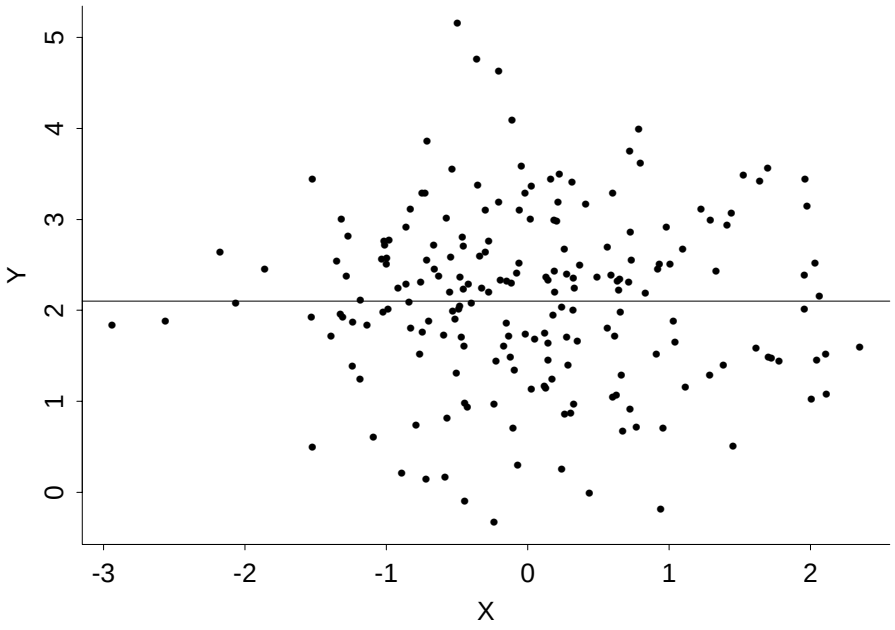
To understand the interpretation of R^2, at least in two extreme cases, note that

Reg SS = Total SS $\Leftrightarrow$ Res SS = 0
$\Leftrightarrow$ all the data points fall on a straight line
$\Leftrightarrow$ all the variability in Y is explained by the linear relationshi
(which has variation)
$\Leftrightarrow$ $R^2 = 1$. (see the picture below)



Furthermore,

$$\begin{aligned}
\text{Reg SS} = 0 &\Leftrightarrow \text{Total SS} = \text{Res SS} \\
&\Leftrightarrow b_1 = 0 \\
&\Leftrightarrow \text{LS line is } \hat{Y} = \bar{Y} \\
&\Leftrightarrow \text{none of the variability in } Y \text{ is explained by a linear relationship} \\
&\Leftrightarrow R^2 = 0.
\end{aligned}$$



Each Sum of Squares has a corresponding *df* (degrees of freedom). The Sums of Squares and *df* are arranged in an analysis of variance (ANOVA) table:

Source	*df*	SS	MS
Regression	1	SSR	MSR
Residual (Error)	$n-2$	SSE	MSE
Total	$n-1$		

The Total *df* is $n-1$. The Residual *df* is n minus the number of parameters (2) estimated by the LS line. The Regression *df* is the number of predictor

variables (1) in the model. A Mean Square is always equal to the Sum of Squares divided by the *df*. Sometime the following notation is used for the Residual MS: $s^2_{Y|X} = \text{Resid(SS)}/(n-2)$.

```
# ANOVA table of the simple linear regression fit
anova(lm.blood.wt)
```

```
## Analysis of Variance Table
##
## Response: blood_loss
##           Df  Sum Sq Mean Sq F value  Pr(>F)
## weight     1 1207.45 1207.45  8.8778 0.02465 *
## Residuals  6  816.05  136.01
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

8.5.1 Brief discussion of the output for blood loss problem

1. Identify fitted line: Blood Loss $= 552.44 - 1.30$ Weight (i.e., $b_0 = 552.44$ and $b_1 = -1.30$).
   What is the line of best fit? What is the direction of the relationship?
2. Locate Analysis of Variance Table.
   This tests the the hypothesis $H_0 : \beta_j = 0$, $j = 0, 1, \ldots, p$ (for all $p+1$ beta coefficients), against H_A : not H_0, i.e., at least one $\beta_j \neq 0$. More on this later.
3. Locate Parameter Estimates Table.
   Given that not all the betas are zero from the ANOVA, which parameter betas are different from zero, and what are their estimated values and standard errors? More on this later.
4. Note that $R^2 = 0.5967 = (-0.77247)^2 = r^2$.
   R^2 indicates the proportion of the variance explained by the regression model. This indicates the predictive ability of the model, and is *not* a indication of model fit.

■ ClickerQs — Salaries STT.02.02.060 ■

8.6 The regression model

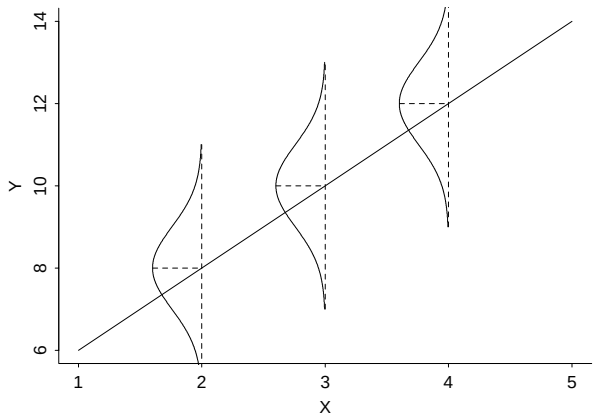
The following statistical model is assumed as a means to provide error estimates for the LS line, regression coefficients, and predictions. Assume that the data (X_i, Y_i), $i = 1, \ldots, n$, are a sample of (X, Y) values from the population of interest, and

1. The mean in the population of all responses Y at a given X value (sometimes called $\mu_{Y|X}$) falls on a straight line, $\beta_0 + \beta_1 X$, called the population regression line.
2. The variation among responses Y at a given X value is the same for each X, and is denoted by $\sigma^2_{Y|X}$.
3. The population of responses Y at a given X is normally distributed.
4. The pairs (X_i, Y_i) are a random sample from the population. Alternatively, we can think that the X_is were fixed by the experimenter, and that the Y_i are random responses at the selected predictor values.

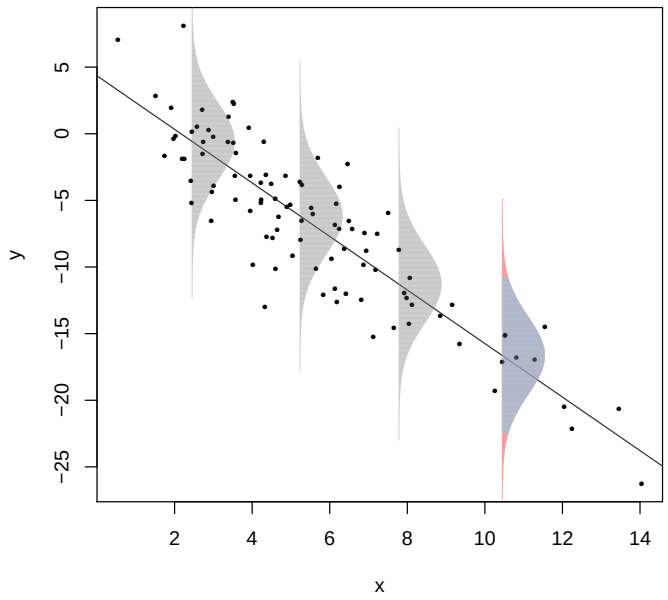
The model is usually written in the form

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

(i.e., Response = Mean Response + Residual), where the ε_is are, by virtue of assumptions 2, 3, and 4, independent normal random variables with mean 0 and variance $\sigma^2_{Y|X}$. The following picture might help see this. Note that the population regression line is unknown, and is estimated from the data using the LS line.



In the plot below, data are simulated where $y_i = 4 - 2x_i + e_i$, where $x_i \sim \text{Gamma}(3, 0.5)$ and $e_i \sim \text{Normal}(0, 3^2)$. The data are plotted and a linear regression is fit and the mean regression line is overlayed. Select normal distributions with variance estimated from the linear model fit are overlayed, one which indicates limits at two standard deviations. See the R code to create this image.



Model assumptions In decreasing order of importance, the model assumptions are

1. **Validity.** Most importantly, the data you are analyzing should map to

the research question you are trying to answer. This sounds obvious but is often overlooked or ignored because it can be inconvenient.

2. **Additivity and linearity.** The most important mathematical assumption of the regression model is that its deterministic component is a linear function of the separate predictors.

3. **Independence of errors.** This assumption depends on how the data were collected.

4. **Equal variance of errors.**

5. **Normality of errors.**

Normality and equal variance are typically minor concerns, unless you're using the model to make predictions for individual data points.

8.6.1 Back to the Data

There are three unknown population parameters in the model: β_0, β_1 and $\sigma^2_{Y|X}$. Given the data, the LS line

$$\hat{Y} = b_0 + b_1 X$$

estimates the population regression line $Y = \beta_0 + \beta_1 X$. The LS line is our best guess about the unknown population regression line. Here b_0 estimates the intercept β_0 of the population regression line and b_1 estimates the slope β_1 of the population regression line.

The i^{th} **observed residual** $e_i = Y_i - \hat{Y}_i$, where $\hat{Y}_i = b_0 + b_1 X_i$ is the i^{th} **fitted value**, estimates the **unobservable residual** ε_i (ε_i is unobservable because β_0 and β_1 are unknown). The Residual MS from the ANOVA table is used to estimate $\sigma^2_{Y|X}$:

$$s^2_{Y|X} = \text{Res MS} = \frac{\text{Res SS}}{\text{Res df}} = \frac{\sum_i (Y_i - \hat{Y}_i)^2}{n-2}.$$

The denominator $df = n - 2$ is the number of observations minus the number of beta parameters in the model, i.e., β_0 and β_1.

8.7 CI and tests for β_1

A CI for β_1 is given by $b_1 \pm t_{\text{crit}} SE_{b_1}$, where the standard error of b_1 under the model is

$$SE_{b_1} = \frac{s_{Y|X}}{\sqrt{\sum_i (X_i - \bar{X})^2}},$$

and where t_{crit} is the appropriate critical value for the desired CI level from a t-distribution with df =Res df.

To test $H_0 : \beta_1 = \beta_{10}$ (a given value) against $H_A : \beta_1 \neq \beta_{10}$, reject H_0 if $|t_s| \geq t_{\text{crit}}$, where

$$t_s = \frac{b_1 - \beta_{10}}{SE_{b_1}},$$

and t_{crit} is the t-critical value for a two-sided test, with the desired size and df =Res df. Alternatively, you can evaluate a p-value in the usual manner to make a decision about H_0.

```
# CI for beta1
sum.lm.blood.wt <- summary(lm.blood.wt)
sum.lm.blood.wt$coefficients
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 552.442023 21.4408832 25.765824 2.253105e-07
## weight       -1.300327  0.4364156 -2.979562 2.465060e-02
```

```
est.beta1 <- sum.lm.blood.wt$coefficients[2,1]
se.beta1  <- sum.lm.blood.wt$coefficients[2,2]
sum.lm.blood.wt$fstatistic
```

```
##    value    numdf    dendf
## 8.877788 1.000000 6.000000
```

```
df.beta1  <- sum.lm.blood.wt$fstatistic[3]
t.crit    <- qt(1-0.05/2, df.beta1)
t.crit
```

```
## [1] 2.446912
```

```
CI.lower  <- est.beta1 - t.crit * se.beta1
CI.upper  <- est.beta1 + t.crit * se.beta1
c(CI.lower, CI.upper)
```

```
## [1] -2.3681976 -0.2324567
```

The parameter estimates table gives the standard error, t-statistic, and p-value for testing $H_0 : \beta_1 = 0$. Analogous summaries are given for the intercept, β_0, but these are typically of less interest.

8.7.1 Testing $\beta_1 = 0$

Assuming the mean relationship is linear, consider testing $H_0 : \beta_1 = 0$ against $H_A : \beta_1 \neq 0$. This test can be conducted using a t-statistic, as outlined above, or with an ANOVA F-test, as outlined below.

For the analysis of variance (ANOVA) F-test, compute

$$F_s = \frac{\text{Reg MS}}{\text{Res MS}}$$

and reject H_0 when F_s exceeds the critical value (for the desired size test) from an F-table with numerator $df = 1$ and denominator $df = n - 2$ (see `qf()`). The hypothesis of zero slope (or no relationship) is rejected when F_s is large, which happens when a significant portion of the variation in Y is explained by the linear relationship with X.

The p-values from the t-test and the F-test are always equal. Furthermore this p-value is equal to the p-value for testing no correlation between Y and X, using the t-test described earlier. Is this important, obvious, or disconcerting?

8.8 A CI for the population regression line

I can not overemphasize the **power** of the regression model. The model allows you to estimate the mean response at any X value in the range for which the model is reasonable, even if little or no data is observed at that location.

We estimate the mean population response among individuals with $X = X_p$

$$\mu_p = \beta_0 + \beta_1 X_p,$$

with the fitted value, or the value of the least squares line at X_p:

$$\hat{Y}_p = b_0 + b_1 X_p.$$

X_p is not necessarily one of the observed X_is in the data. To get a CI for μ_p, use $\hat{Y}_p \pm t_{\text{crit}} SE(\hat{Y}_p)$, where the standard error of $\hat{Y}_p$ is

$$SE(\hat{Y}_p) = s_{Y|X}\sqrt{\frac{1}{n} + \frac{(X_p - \bar{X})^2}{\sum_i (X_i - \bar{X})^2}}.$$

The t-critical value is identical to that used in the subsection on CI for β_1.

8.8.1 CI for predictions

Suppose a future individual (i.e., someone not used to compute the LS line) has $X = X_p$. The best prediction for the response Y of this individual is the value of the least squares line at X_p:

$$\hat{Y}_p = b_0 + b_1 X_p.$$

To get a CI (prediction interval) for an individual response, use $\hat{Y}_p \pm t_{\text{crit}} SE_{pred}(\hat{Y}_p)$, where

$$SE_{pred}(\hat{Y}_p) = s_{Y|X}\sqrt{1 + \frac{1}{n} + \frac{(X_p - \bar{X})^2}{\sum_i (X_i - \bar{X})^2}},$$

and t_{crit} is identical to the critical value used for a CI on β_1. The prediction variance has two parts: (1) the 1 indicates the variability associated with the data around the mean (regression line), and (2) the rest is the variability associated with estimating the mean.

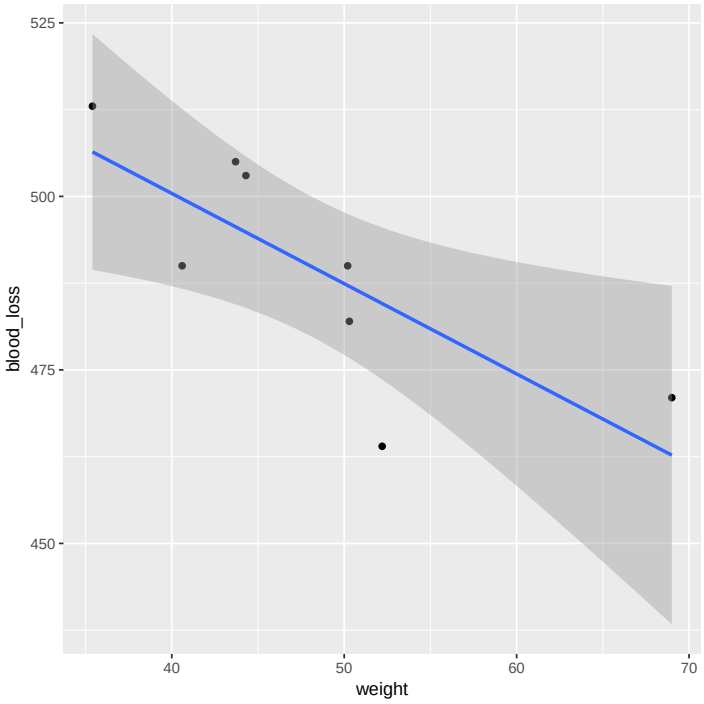
For example, in the blood loss problem you may want to estimates the blood loss for an $50kg$ individual, and to get a CI for this prediction. This problem is different from computing a CI for the mean blood loss of all $50kg$ individuals!

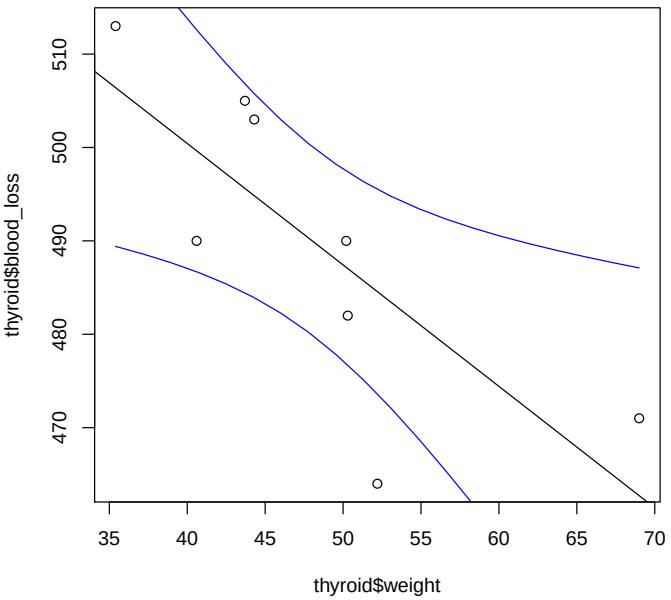
```
# CI for the mean and PI for a new observation at weight=50
predict(lm.blood.wt, data.frame(weight=50), interval = "confidence", level = 0.95)
##        fit      lwr      upr
## 1 487.4257 477.1575 497.6938
predict(lm.blood.wt, data.frame(weight=50), interval = "prediction", level = 0.95)
##        fit     lwr      upr
## 1 487.4257 457.098 517.7533
```

Comments

1. The prediction interval is wider than the CI for the mean response. This is reasonable because you are less confident in predicting an individual response than the mean response for all individuals.
2. The CI for the mean response and the prediction interval for an individual response become wider as X_p moves away from $\bar{X}$. That is, you get a more sensitive CI and prediction interval for X_ps near the center of the data.
3. In plots below include confidence and prediction bands along with the fitted LS line.

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(thyroid, aes(x = weight, y = blood_loss))
p <- p + geom_point()
p <- p + geom_smooth(method = lm, se = TRUE)
print(p)


# Base graphics: Plot the data with linear regression fit and confidence bands
# scatterplot
plot(thyroid$weight, thyroid$blood_loss)
# regression line from lm() fit
abline(lm.blood.wt)
# x values of weight for predictions of confidence bands
x.pred <- data.frame(weight = seq(min(thyroid$weight), max(thyroid$weight),
                         length = 20))
# draw upper and lower confidence bands
lines(x.pred$weight, predict(lm.blood.wt, x.pred,
                             interval = "confidence")[, "upr"], col = "blue")
lines(x.pred$weight, predict(lm.blood.wt, x.pred,
                             interval = "confidence")[, "lwr"], col = "blue")
```





8.8.2 A further look at the blood loss data

- The LS line is: Predicted Blood Loss = 552.442 - 1.30 Weight.
- The R^2 is 0.597 (i.e., 59.7%).
- The F-statistic for testing $H_0 : \beta_1 = 0$ is $F_{obs} = 8.88$ with a p-value=0.025. The Error MS is $s^2_{Y|X} = 136.0$; see ANOVA table.
- The Parameter Estimates table gives b_0 and b_1, their standard errors, and t-statistics and p-values for testing $H_0 : \beta_0 = 0$ and $H_0 : \beta_1 = 0$. The t-test and F-test p-values for testing that the slope is zero are identical. We could calculate a 95% CI for β_0 and β_1. If we did so (using the t critical value) we find we are 95% confident that the slope of the population regression line is between -2.37 and -0.23.
- Suppose we are interested in estimating the average blood loss among all 50kg individuals. The estimated mean blood loss is $552.442 - 1.30033 \times 50 = 487.43$. Reading off the plot, we are 95% confident that the mean blood loss of all 50kg individuals is between (approximately) 477 and 498 ml. A 95% prediction interval for the blood loss of a single 50 kg person is less precise (about 457 to 518 ml).

As a summary we might say that weight is important for explaining the variation in blood loss. In particular, the estimated slope of the least squares line (Predicted Blood loss = 552.442 - 1.30 Weight) is significantly different from zero (p-value = 0.0247), with weight explaining approximately 60% (59.7%) of the variation in blood loss for this sample of 8 thyroid operation patients.

8.9 Model Checking and Regression Diagnostics

8.9.1 Introduction

The simple linear regression model is usually written as

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

where the ε_is are independent normal random variables with mean 0 and variance σ^2. The model implies (1) The average Y-value at a given X-value is linearly related to X. (2) The variation in responses Y at a given X value is constant. (3) The population of responses Y at a given X is normally distributed. (4) The observed data are a random sample.

A regression analysis is never complete until these assumptions have been checked. In addition, you need to evaluate whether individual observations, or groups of observations, are unduly influencing the analysis. A first step in any analysis is to plot the data. The plot provides information on the linearity and constant variance assumption.

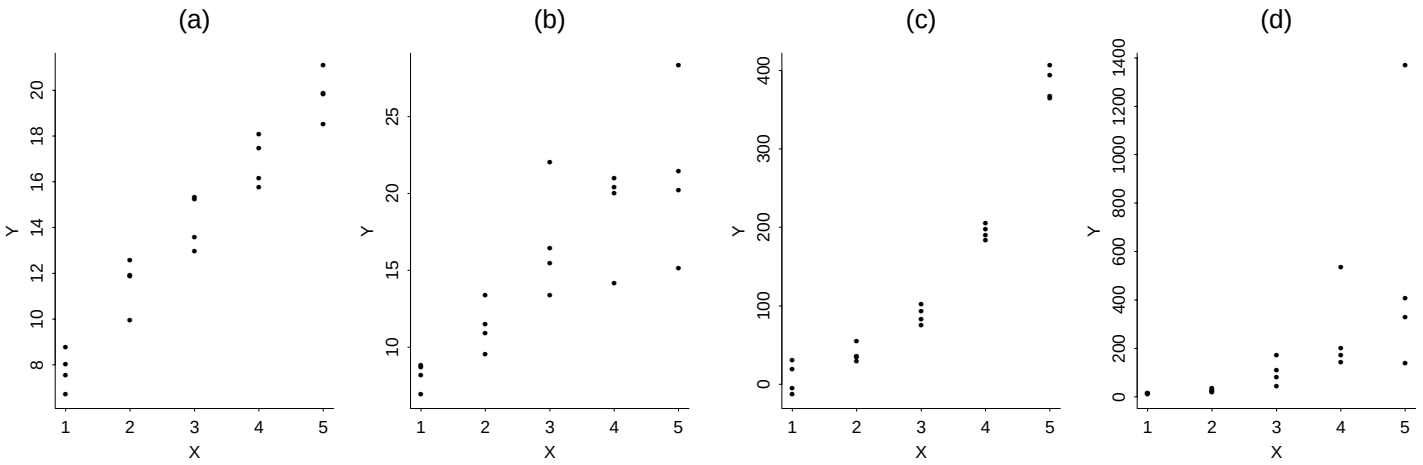


Figure (a) is the only plot that is consistent with the assumptions. The plot shows a linear relationship with constant variance. The other figures show one or more deviations. Figure (b) shows a linear relationship but the variability increases as the mean level increases. In Figure (c) we see a nonlinear relationship with constant variance, whereas (d) exhibits a nonlinear relationship with non-constant variance.

In many examples, nonlinearity or non-constant variability can be addressed by **transforming** Y or X (or both), or by fitting **polynomial models**. These issues will be addressed later.

8.9.2 Residual Analysis

A variety of methods for assessing model adequacy are based on the **observed residuals**,

$$e_i = Y_i - \hat{Y}_i \quad \text{i.e., Observed} - \text{Fitted values.}$$

The residual is the difference between the observed values and predicted or fitted values. The residual is the part of the observation that is not explained by the fitted model. You can analyze residuals to determine the adequacy of the model. A large residual identifies an observation poorly fit by the model.

The residuals are usually plotted in various ways to assess potential inadequacies. The observed residuals have different variances, depending on X_i.

Recall that the standard error of $\hat{Y}_i$ (and therefore e_i) is

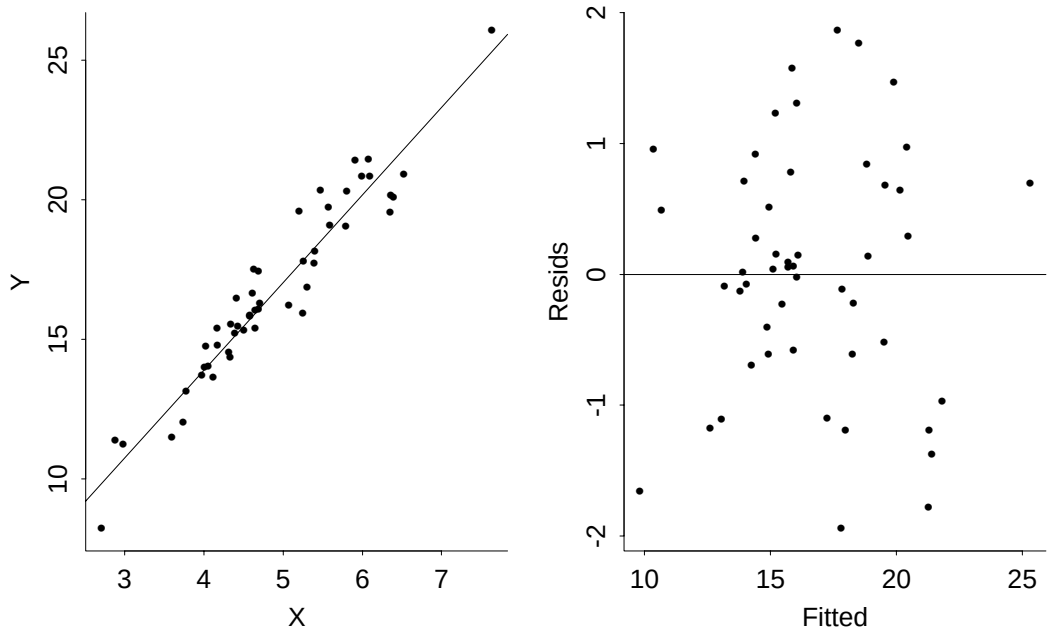
$$\text{SE}(\hat{Y}_i) \;=\; \text{SE}(e_i) \;=\; s_{Y|X}\sqrt{\frac{1}{n}+\frac{(X_i-\bar{X})^2}{\sum_j (X_j-\bar{X})^2}}.$$

So many statisticians prefer to plot the **studentized residuals** (sometimes called the standardized residuals or internally Studentized residual)

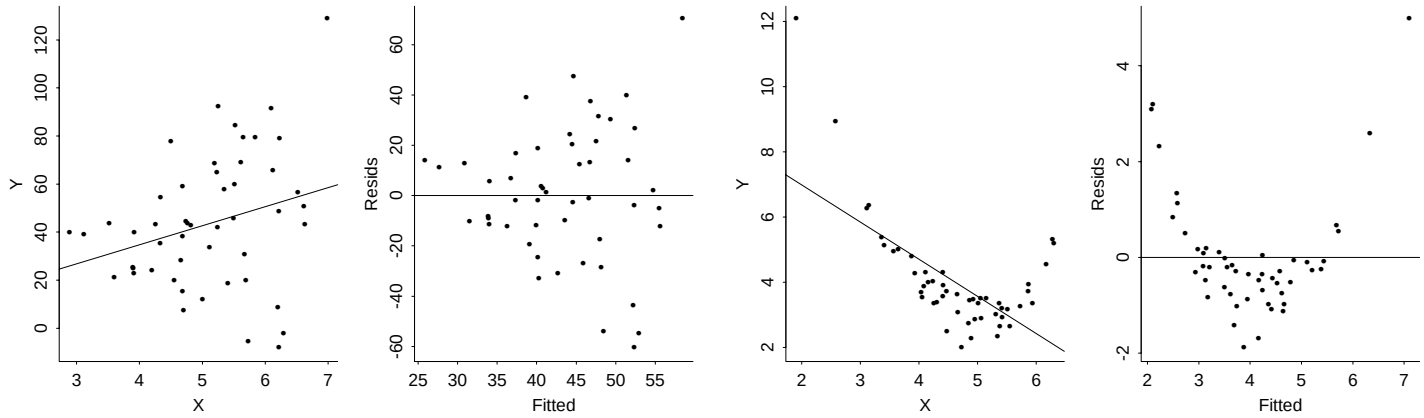
$$r_i = \frac{e_i}{SE(e_i)}.$$

The standardized residual is the residual, e_i, divided by an estimate of its standard deviation. This form of the residual takes into account that the residuals may have different variances, which can make it easier to detect outliers. The studentized residuals have a constant variance of 1 (approximately). Standardized residuals greater than 2 and less than -2 are usually considered large. I will focus on diagnostic methods using the studentized residuals.

A plot of the studentized residuals r_i against the fitted values $\hat{Y}_i$ often reveals inadequacies with the model. The real power of this plot is with multiple predictor problems (multiple regression). The information contained in this plot with simple linear regression is similar to the information contained in the original data plot, except it is scaled better and eliminates the effect of the trend on your perceptions of model adequacy. The residual plot should exhibit no systematic dependence of the sign or the magnitude of the residuals on the fitted values:

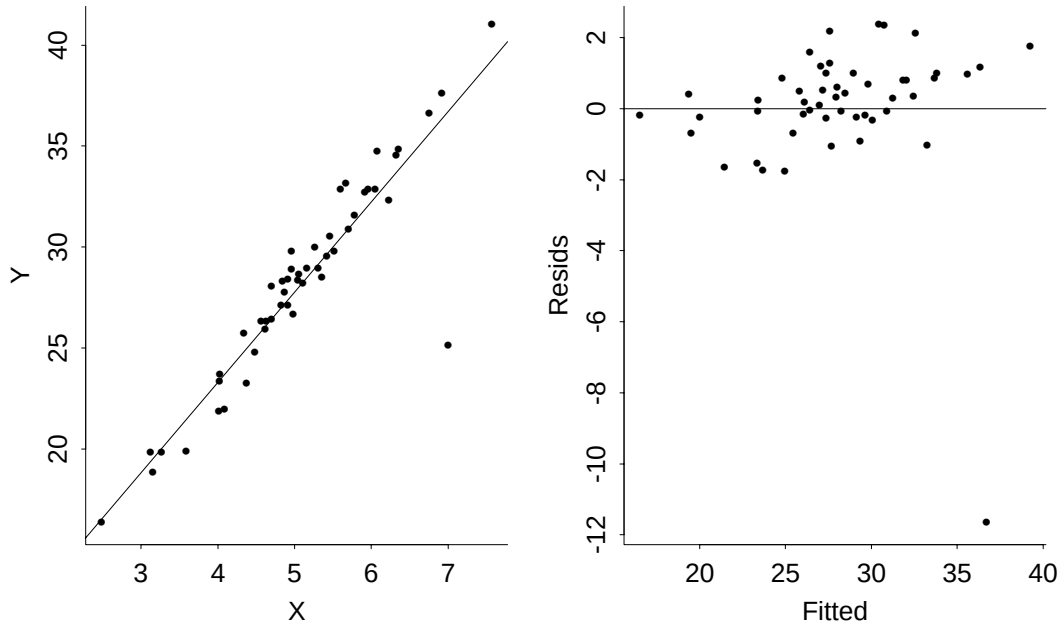


The following sequence of plots show how inadequacies in the data plot appear in a residual plot. The first plot shows a roughly linear relationship between Y and X with non-constant variance. The residual plot shows a megaphone shape rather than the ideal horizontal band. A possible remedy is a **weighted least squares** analysis to handle the non-constant variance (see end of chapter for an example), or to transform Y to stabilize the variance. Transforming the data may destroy the linearity.



The plot above shows a nonlinear relationship between Y and X. The residual plot shows a systematic dependence of the sign of the residual on the fitted value. Possible remedies were mentioned earlier.

The plot below shows an **outlier**. This case has a large residual and large studentized residual. A sensible approach here is to refit the model after holding out the case to see if any conclusions change.



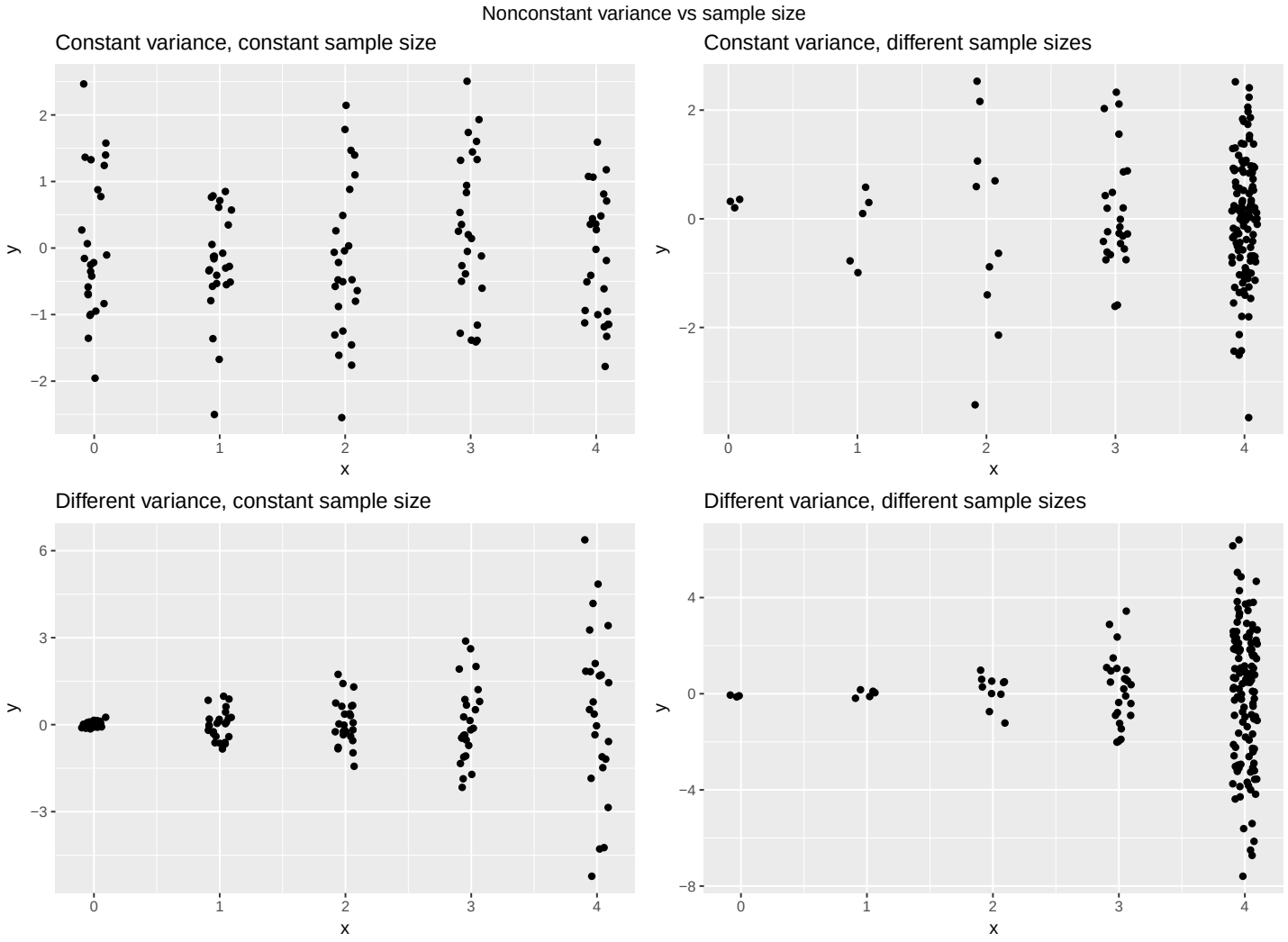
A third type of residual is called an **externally Studentized residual** (or Studentized deleted residual or deleted t-residual). The problem with residuals is that a highly influential value can force the residual to have a very small value. This measure tries to correct for that by looking at how well the model fits this observation without using this observation to construct the fit. It is quite possible for the deleted residual to be huge when the raw residual is tiny.

The studentized deleted residual for observation i^{th} is calculated by fitting the regression based on all of the cases except the i^{th} one. The residual is then divided by its estimated standard deviation. Since the Studentized deleted residual for the i^{th} observation estimates all quantities with this observation deleted from the data set, the i^{th} observation cannot influence these estimates. Therefore, unusual Y values clearly stand out. Studentized deleted residuals with large absolute values are considered large. If the regression model is appropriate, with no outlying observations, each Studentized deleted residual follows the t-distribution with $n - 1 - p$ degrees of freedom.

Nonconstant variance vs sample size Because more extreme observations are more likely to occur with larger sample sizes, sometimes when sample size depends on X it can appear as noncontant variance. Below sample sizes are either all 25 or (3, 5, 10, 25, 125), and standard deviations are either all 1 or (0.1, 0.5, 1, 1.5, 3). Note that constant variance and different sample sizes appears as though it has nonconstant variance.

```
#### Nonconstant variance vs sample size
dat.var.sam <- function(n, s) {
  dat <- data.frame(
           x = c(rep(0, n[1]),
                 rep(1, n[2]),
                 rep(2, n[3]),
                 rep(3, n[4]),
                 rep(4, n[5])),
           y = c(rnorm(n[1], mean = 0, sd = s[1]),
                 rnorm(n[2], mean = 0, sd = s[2]),
                 rnorm(n[3], mean = 0, sd = s[3]),
                 rnorm(n[4], mean = 0, sd = s[4]),
                 rnorm(n[5], mean = 0, sd = s[5]))
         )
```

```
  return(dat)
}

library(ggplot2)

n <- c(25, 25, 25, 25, 25)
s <- c(1, 1, 1, 1, 1)
dat <- dat.var.sam(n, s)
p1 <- ggplot(dat, aes(x = x, y = y))
p1 <- p1 + geom_point(position = position_jitter(width = 0.1))
p1 <- p1 + labs(title = "Constant variance, constant sample size")
#print(p1)

n <- c(3, 5, 10, 25, 125)
s <- c(1, 1, 1, 1, 1)
dat <- dat.var.sam(n, s)
p2 <- ggplot(dat, aes(x = x, y = y))
p2 <- p2 + geom_point(position = position_jitter(width = 0.1))
p2 <- p2 + labs(title = "Constant variance, different sample sizes")
#print(p2)

n <- c(25, 25, 25, 25, 25)
s <- c(0.1, 0.5, 1, 1.5, 3)
dat <- dat.var.sam(n, s)
p3 <- ggplot(dat, aes(x = x, y = y))
p3 <- p3 + geom_point(position = position_jitter(width = 0.1))
p3 <- p3 + labs(title = "Different variance, constant sample size")
#print(p3)

n <- c(3, 5, 10, 25, 125)
s <- c(0.1, 0.5, 1, 1.5, 3)
dat <- dat.var.sam(n, s)
p4 <- ggplot(dat, aes(x = x, y = y))
p4 <- p4 + geom_point(position = position_jitter(width = 0.1))
p4 <- p4 + labs(title = "Different variance, different sample sizes")
#print(p4)

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4), nrow=2, ncol=2
             , top = "Nonconstant variance vs sample size")
```



8.9.3 Checking Normality

The normality assumption for the ε_is can be evaluated visually with a boxplot or a normal probability plot (rankit plot) of the r_i, or formally with a Shapiro-Wilk test. The normal probability plot often highlights **outliers**, or poorly fitted cases. If an outlier is held out of the data and a new analysis is performed, the resulting normal scores plot may be roughly linear, but often will show a short-tailed distribution. (Why?).

You must interpret regression tests and CI with caution with non-normal data. Statisticians developed robust regression methods for non-normal data which are available in R packages.

8.9.4 Checking Independence

Diagnosing dependence among observations requires an understanding of the data collection process. There are a variety of graphical and inferential tools for checking independence for data collected over time (called a time series). The easiest check is to plot the r_i against time index and look for any suggestive patterns.

8.9.5 Outliers

Outliers are observations that are poorly fitted by the regression model. The response for an outlier is far from the fitted line, so outliers have large positive or negative values of the studentized residual r_i. Usually, $|r_i| > 2$ is considered large. Outliers are often highlighted in residual plots.

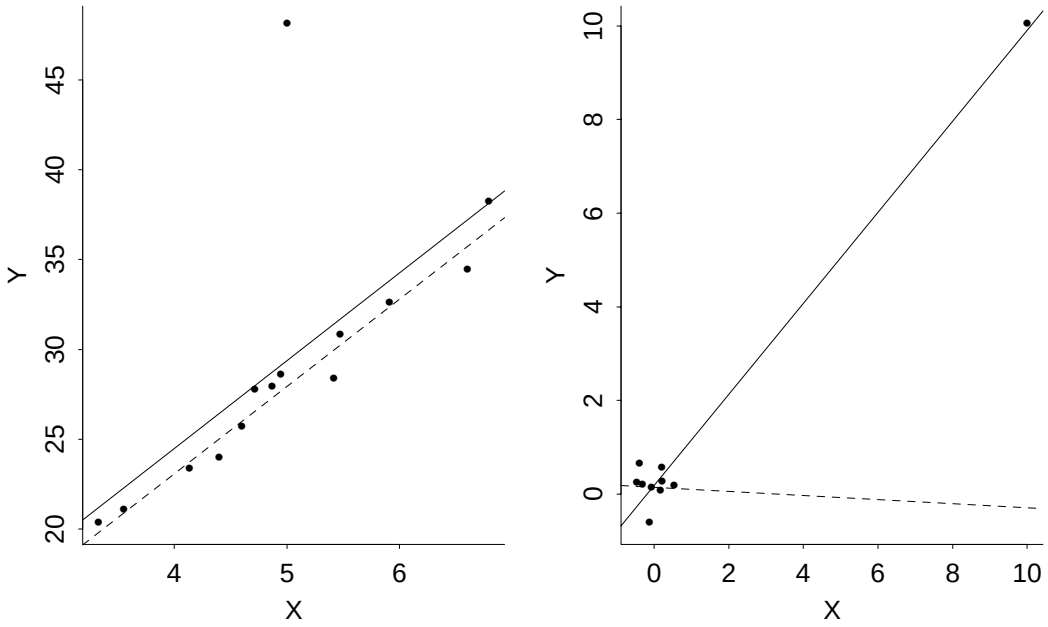
What do you do with outliers? Outliers may be due to incorrect recordings of the data or failure of the measuring device, or indications or a change in the mean or variance structure for one or more cases. Incorrect recordings should be fixed if possible, but otherwise deleted from the analysis.

Routine deletion of outliers from the analysis is not recommended. This practice can have a dramatic effect on the fit of the model and the perceived precision of parameter estimates and predictions. Analysts who routinely omit outliers without cause tend to overstate the significance of their findings and get a false sense of precision in their estimates and predictions. To assess effects of outliers, a data analyst should repeat the analysis holding out the outliers to see whether any substantive conclusions are changed. Very often the only real effect of an outlier is to inflate MSE and hence make p-values a little larger and CIs a little wider than necessary, but without substantively changing conclusions. They can completely mask underlying patterns, however.

■ ClickerQs — Outliers STT.02.04.040 ■

8.9.6 Influential Observations

Certain data points can play an important role in determining the position of the LS line. These data points may or may not be outliers. In the plots below, the solid line is the LS line from the full data set, whereas the dashed line is the LS line after omitting the unusual point. For example, the observation with $Y > 45$ in the first plot is an outlier relative to the LS fit. The extreme observation in the second plot has a very small r_i. Both points are highly **influential observations** — the LS line changes dramatically when these observations are held out.



In the second plot, the extreme value is a **high leverage** value, which is basically an outlier among the X values; Y does not enter in this calculation. This influential observation is not an outlier because its presence in the analysis determines that the LS line will essentially pass through it! These are values with the *potential* of greatly distorting the fitted model. They may or may not actually have distorted it.

The `hat` variable from the `influence()` function on the object returned from `lm()` fit will give the leverages: `influence(lm.output)$hat`. Leverage values fall between 0 and 1. Experts consider a leverage value greater than $2p/n$ or $3p/n$, where p is the number of predictors or factors plus the constant and n is the number of observations, large and suggest you examine the corresponding observation. A rule-of-thumb is to identify observations with leverage over $3p/n$ or 0.99, whichever is smaller.

Dennis Cook developed a measure of the impact that individual cases have on the placement of the LS line. His measure, called **Cook's distance** or Cook's D, provides a summary of how far the LS line changes when each individual point is held out (one at a time) from the analysis. While high leverage values indicate observations that have the *potential* of causing trouble, those with high Cook's D values actually *do* disproportionately affect the overall fit. The case with the largest D has the greatest impact on the placement of the LS line. However, the actual influence of this case may be small. In the plots above, the observations I focussed on have the largest Cook's Ds.

A simple, but not unique, expression for Cook's distance for the j^{th} case is

$$D_j \propto \sum_i (\hat{Y}_i - \hat{Y}_{i[-j]})^2,$$

where $\hat{Y}_{i[-j]}$ is the fitted value for the i^{th} case when the LS line is computed from all the data except case j. Here $\propto$ means that D_j is a multiple of $\sum_i (\hat{Y}_i - \hat{Y}_{i[-j]})^2$ where the multiplier does not depend on the case. This expression implies that D_j is also an overall measure of how much the fitted values change when case j is deleted.

Observations with large D values may be outliers. Because D is calculated using leverage values and standardized residuals, it considers whether an observation is unusual with respect to both x- and y-values. To interpret D, compare it to the F-distribution with $(p, n-p)$ degrees-of-freedom to determine the corresponding percentile. If the percentile value is less than 10% or 20%, the observation has little influence on the fitted values. If the percentile value is greater than 50%, the observation has a major influence on the fitted

values and should be examined.

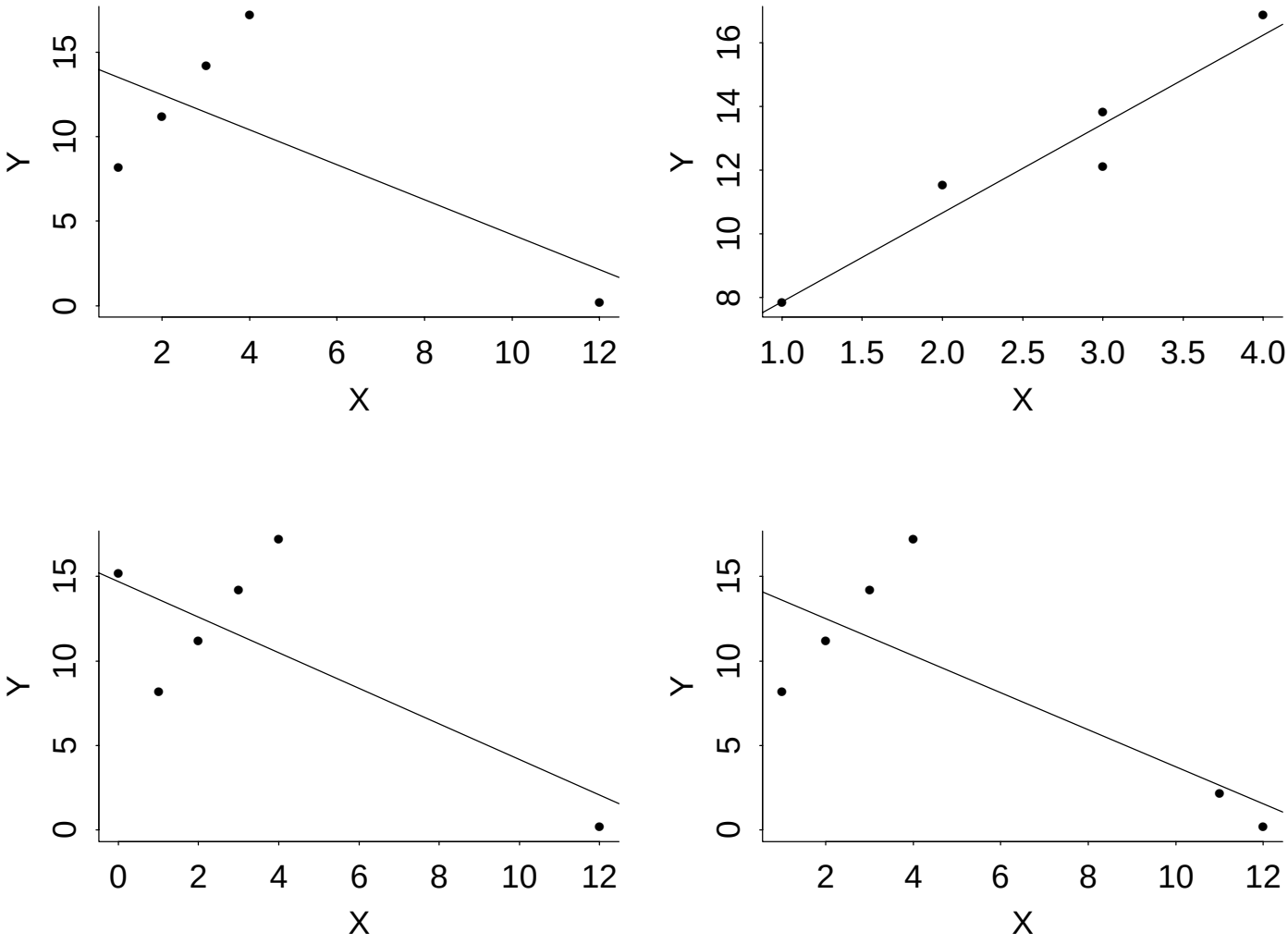
Many statisticians make it a lot simpler than this sounds and use 1 as a cutoff value for large Cook's D (when D is on the appropriate scale). Using the cutoff of 1 can simplify an analysis, since frequently one or two values will have noticeably larger D values than other observations without actually having much effect, *but it can be important to explore any observations that stand out.* Cook's distance values for each observation from a linear regression fit are given with `cooks.distance(lm.output)`.

Given a regression problem, you should locate the points with the largest D_js and see whether holding these cases out has a decisive influence on the fit of the model or the conclusions of the analysis. You can examine the relative magnitudes of the D_js across cases without paying much attention to the actual value of D_j, but there are guidelines (see below) on how large D_j needs to be before you worry about it.

It is difficult to define a good strategy for dealing with outliers and influential observations. Experience is the best guide. I will show you a few examples that highlight some standard phenomena. One difficulty you will find is that certain observations may be outliers because other observations are influential, or vice-versa. If an influential observation is held out, an outlier may remain an outlier, may become influential, or both, or neither. Observations of moderate influence may become more, or less influential, when the most influential observation is held out.

Thus, any sequential refitting of models holding out of observations should not be based on the original (full-data) summaries, but rather on the summaries that result as individual observations are omitted. I tend to focus more on influential observations than outliers.

In the plots below, which cases do you think are most influential, and which are outliers. What happens in each analysis if I delete the most influential case? Are any of the remaining cases influential or poorly fitted?



Many researchers are hesitant to delete points from an analysis. I think this view is myopic, and in certain instances, such as the Gesell example to be discussed, can not be empirically supported. Being rigid about this can lead to some silly analyses of data, but one needs a very good reason and full disclosure if any points are deleted.

8.9.7 Summary of diagnostic measures

The various measures discussed above often flag the same observations as unusual, but they certainly can flag different observations. At the very least I examine standardized residuals and Cooks's D values. They are invaluable diagnostic measures, but nothing is perfect. Observations can be unusual in

groups — a pair of unusual high leverage values close to each other will not necessarily be flagged by Cook's D since removing just one may not affect the fit very much. Any analysis takes some careful thought.

These measures and techniques really are designed for multiple regression problems where several predictor variables are used. We are using them in simple linear regression to learn what they do and see what they have to say about data, but in truth it is fairly simple with one variable to see what may be an outlier in the x-direction, to see if the data are poorly fit, etc. With more variables all that becomes quite difficult and these diagnostics are essential parts of those analyses.

8.10 Regression analysis suggestion

There are a lot options allowed in R. I will make a few suggestions here on how to start an analysis. What you find once you get started determines what more you might wish to explore.

1. **Plot the data.** With lots of variables the matrix plot is valuable as a quick screen. If you want to see better resolution on an individual scatter plot, do the individual scatter plot.
2. Do any obvious **transformations** of the data. We will discuss this in a lot more detail later. Re-plot with transformations.
3. **Fit the least squares equation**.
4. **Examine the residual plots and results.** Check for the patterns discussed earlier.
   (a) Plotting several diagnostic plots together seems convenient to me. This gives you the essential plot (residuals vs. fitted values) plus a quick check on normality and possible violations of independence and influence. If you see something that needs closer investigation, you may need to recreate one of the plots larger by itself.
   (b) Do you see curvature in the standardized residual plot? If the sign of the residuals has a distinct pattern vs. the fitted values, the linear fit

is not adequate and you need some remedy, such as transformations. Note that standardized residuals are more conventional and show you what actually happened, while deleted residuals are probably the better diagnostic tool for identifying problem cases.

(c) Does it appear $\sigma_{Y|X}$ depends upon X (we are assuming it does not)? A megaphone pattern in residuals vs. fits is the classic (not the only) pattern to look for. Weighted least squares or transformations may be called for.

(d) Do you see obvious outliers? Make sure you do not have a mis-recorded data value. It might be worth refitting the equation without the outlier to see if it affects conclusions substantially.

(e) Is the normality assumption reasonable? This can be very closely related to the preceding points.

(f) Is there a striking pattern in residuals vs. order of the data? This can be an indication that the independence assumption is not valid.

5. **Check the Cook's D values.** The Cook's distance plot and Residuals vs. Leverage (with Cook's D) plot are both helpful.
6. If you found problem observations, omit them from the analysis and see if any conclusions change substantially. There are two good ways to do this.
   (a) Subset the `data.frame` using `subset()`.
   (b) Use `lm()` with the `weights=` option with weights of 0 for the excluded observations, weights of 1 for those included.

You may need to repeat all these steps many times for a complete analysis.

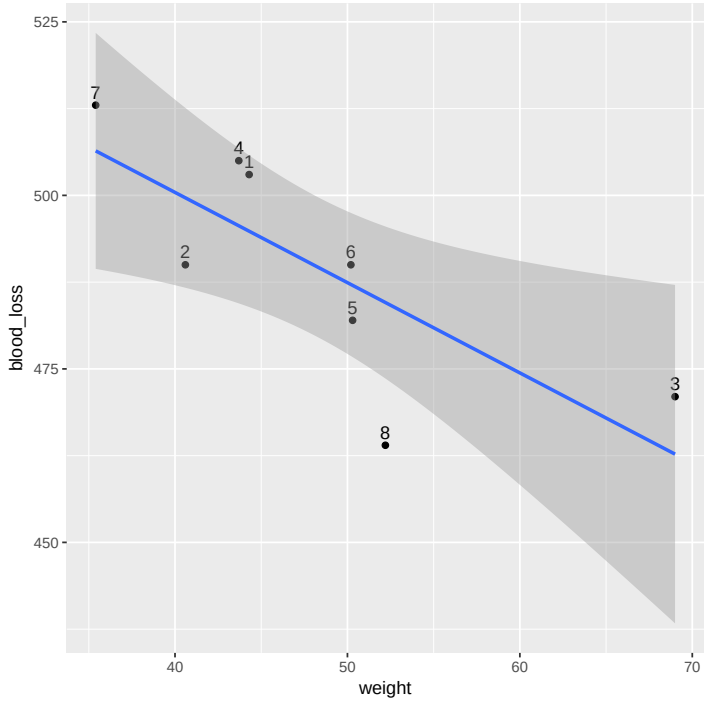
8.10.1 Residual and Diagnostic Analysis of the Blood Loss Data

We looked at much of this before, but let us go through the above steps systematically. Recall the data set (we want to predict blood loss from weight):

	weight	time	blood_loss
1	44.3	105	503
2	40.6	80	490
3	69.0	86	471
4	43.7	112	505
5	50.3	109	482
6	50.2	100	490
7	35.4	96	513
8	52.2	120	464

1. *Plot the data.* Plot blood loss vs. weight.

```
# create data ids
thyroid$id <- 1:nrow(thyroid)
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(thyroid, aes(x = weight, y = blood_loss, label = id))
p <- p + geom_point()
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5)
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
print(p)
```



Clearly the heaviest individual is an unusual value that warrants a closer look (maybe data recording error). I might be inclined to try a transformation here (such as `log(weight)`) to make that point a little less influential.

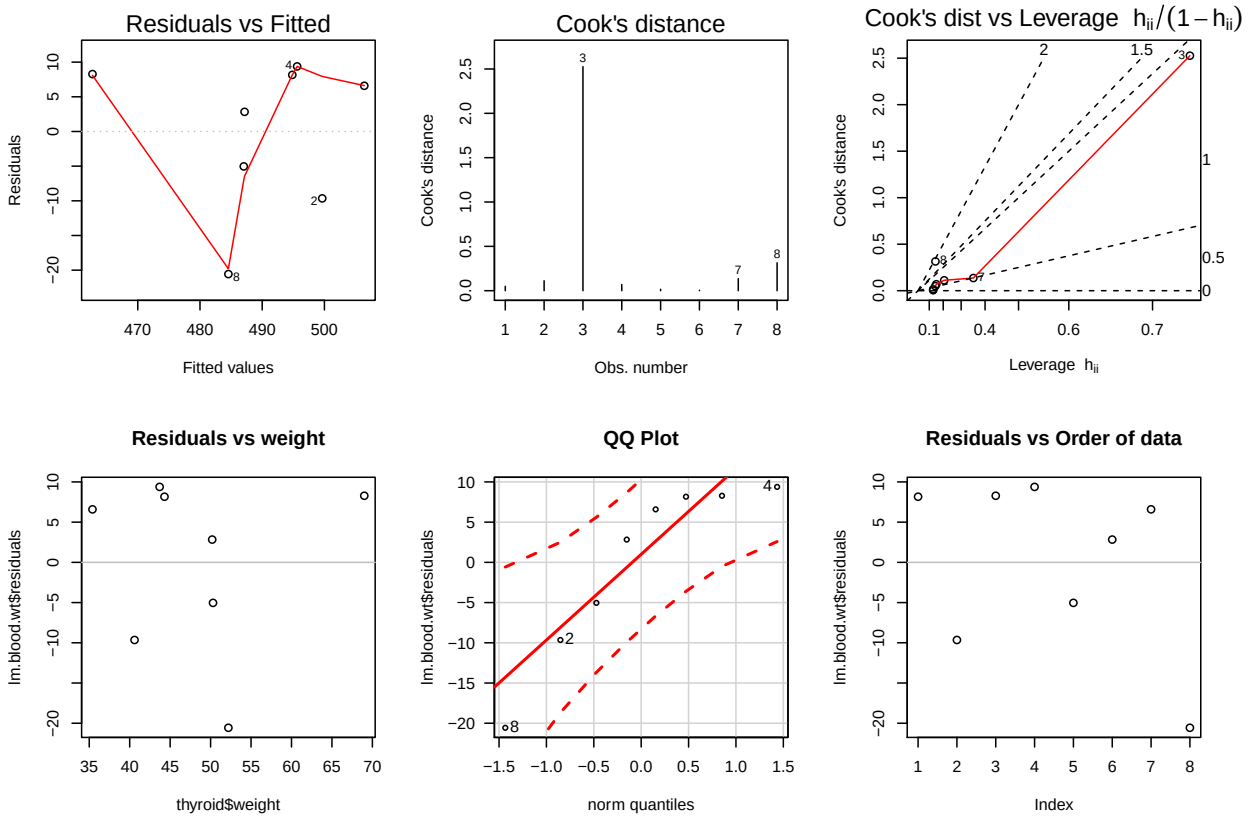
2. *Do any obvious transformations of the data.* We will look at transformations later.
3. *Fit the least squares equation.* Blood Loss appears significantly negatively associated with weight.

```
lm.blood.wt <- lm(blood_loss ~ weight, data = thyroid)
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm.blood.wt)

##
## Call:
## lm(formula = blood_loss ~ weight, data = thyroid)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.565  -6.189   4.712   8.192   9.382
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 552.4420    21.4409   25.77 2.25e-07 ***
## weight       -1.3003     0.4364   -2.98   0.0247 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.66 on 6 degrees of freedom
## Multiple R-squared:  0.5967,Adjusted R-squared:  0.5295
## F-statistic: 8.878 on 1 and 6 DF,  p-value: 0.02465
```

   (a) *Graphs: Check Standardized Residuals (or the Deleted Residuals).* The residual plots:

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.blood.wt, which = c(1,4,6))

# residuals vs weight
plot(thyroid$weight, lm.blood.wt$residuals, main="Residuals vs weight")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
# qq plot for studentized resid
#   las    =  1 : turns labels on y-axis to read horizontally
#   id.n   =  n : labels n most extreme observations, and outputs to console
#   id.cex =  1 : is the size of those labels
#   lwd    =  1 : line width
qqPlot(lm.blood.wt$residuals, las = 1, id.n = 3, main="QQ Plot")

## 8 2 4
```

```
## 1 2 8

# residuals vs order of data
plot(lm.blood.wt$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



4. *Examine the residual plots and results.*
   (a) *Do you see curvature?* There does not appear to be curvature (and it could be hard to detect with so few points).
   (b) *Does it appear $\sigma_{Y|X}$ depends upon X?* Not much evidence for this.
   (c) *Do you see obvious outliers?* Observation 3 is an outlier in the x direction, and therefore possibly a high leverage point and influential on the model fit.
   (d) *Is the normality assumption reasonable?* There appears to be some skewness, but with so few points normality may be reasonable.
   (e) *Is there a striking pattern in residuals vs. order of the data?* No striking pattern.
5. *Check the Cook's D values.* We anticipated that the 3rd observation is

affecting the fit by a lot more than any other values. The D-value is much larger than 1. Note that the residual is not large for this value.

6. *Omit problem observations from the analysis and see if any conclusions change substantially.* Let us refit the equation without observation 3 to see if anything changes drastically. I will use the weighted least squares approach discussed earlier on this example. Define a variable **wt** that is 1 for all observations except obs. 3, and make it 0 for that one.

```
# wt = 1 for all except obs 3 where wt = 0
thyroid$wt <- as.numeric(!(thyroid$id == 3))
thyroid$wt
```

```
## [1] 1 1 0 1 1 1 1 1
```

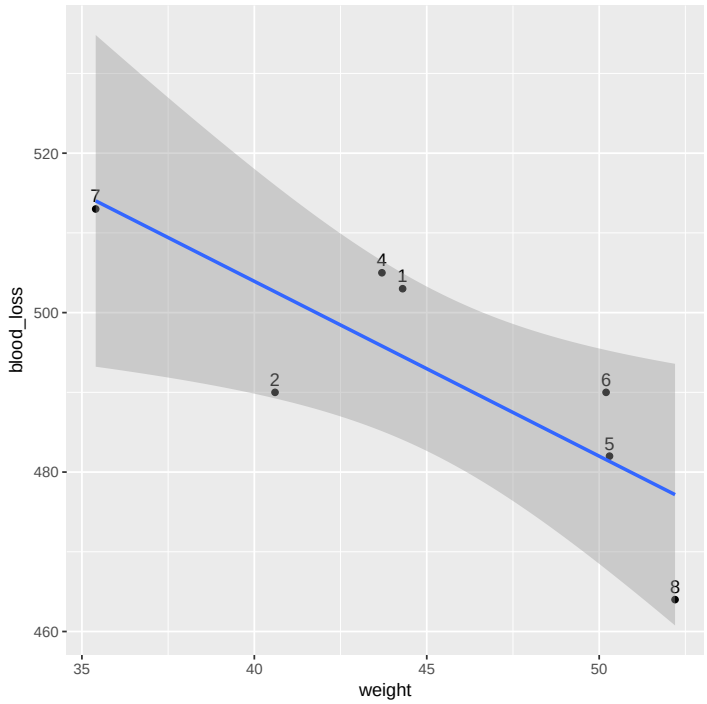
What changes by deleting case 3? The fitted line gets steeper (slope changes from -1.30 to -2.19), adjusted R^2 gets larger (up to 58% from 53%), and S changes from 11.7 to 10.6. Because the Weight values are much less spread out, $SE(\hat{\beta}_1)$ becomes quite a bit larger (to 0.714, up from 0.436) and we lose a degree of freedom for MS Error (which will penalize us on tests and CIs). Just about any quantitative statement we would want to make using CIs would be about the same either way since CIs will overlap a great deal, and our qualitative interpretations are unchanged (Blood Loss drops with Weight). Unless something shows up in the plots, I don't see any very important changes here.

```
lm.blood.wt.no3 <- lm(blood_loss ~ weight, data = thyroid, weights = wt)
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm.blood.wt.no3)
```

```
##
## Call:
## lm(formula = blood_loss ~ weight, data = thyroid, weights = wt)
##
## Weighted Residuals:
##       1       2       3       4       5       6       7
##  8.5033 -12.6126   0.0000   9.1872   0.6641   8.4448  -1.0186
##       8
## -13.1683
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 591.6677    32.5668  18.168 9.29e-06 ***
## weight       -2.1935     0.7144  -3.071   0.0278 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.6 on 5 degrees of freedom
## Multiple R-squared:  0.6535,Adjusted R-squared:  0.5842
```

```
## F-statistic: 9.428 on 1 and 5 DF,  p-value: 0.02777
```

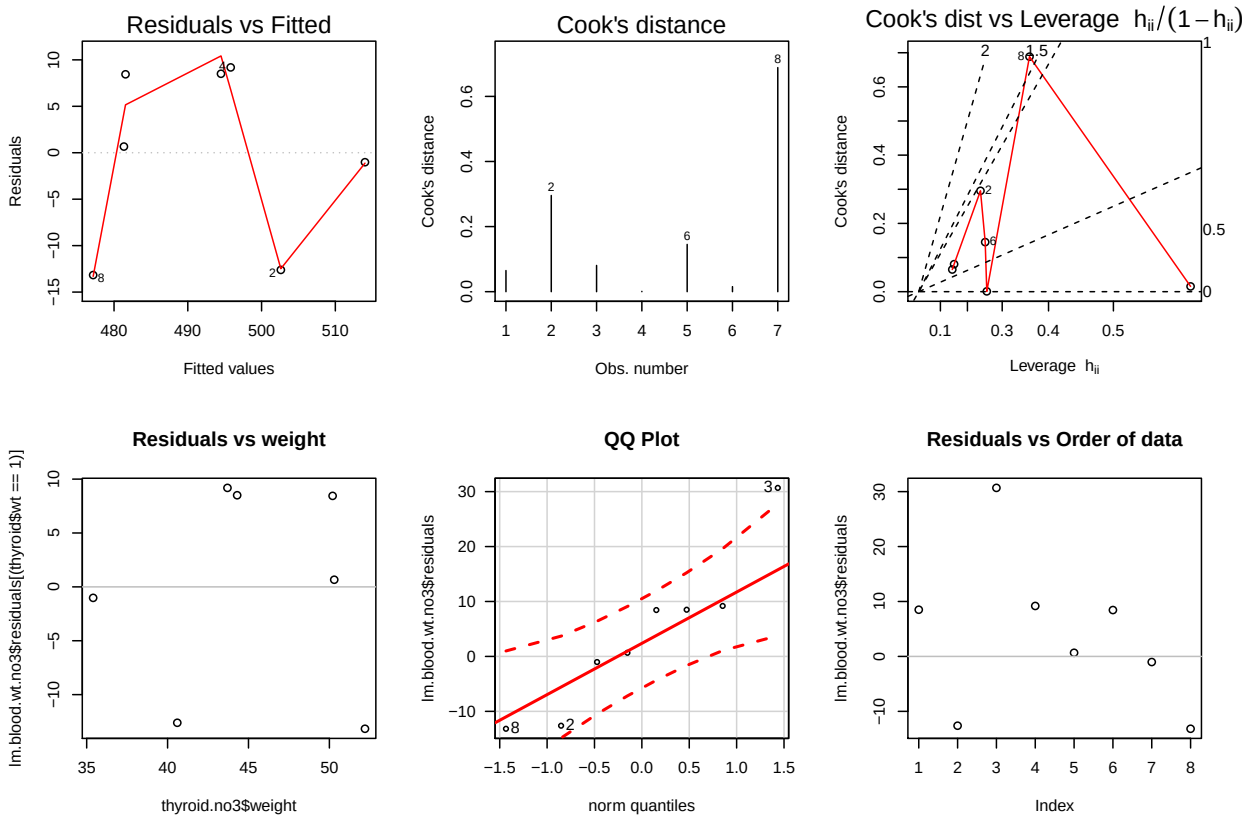
```
# exclude obs 3
thyroid.no3 <- subset(thyroid, wt == 1)
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(thyroid.no3, aes(x = weight, y = blood_loss, label = id))
p <- p + geom_point()
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5)
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
print(p)
```



Nothing very striking shows up in the residual plots, and no Cook's D values are very large among the remaining observations.

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.blood.wt.no3, which = c(1,4,6))

# residuals vs weight
plot(thyroid.no3$weight, lm.blood.wt.no3$residuals[(thyroid$wt == 1)]
   , main="Residuals vs weight")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
# qq plot for studentized resid
#   las    =  1 : turns labels on y-axis to read horizontally
```

```
#   id.n   =  n : labels n most extreme observations, and outputs to console
#   id.cex =  1 : is the size of those labels
#   lwd    =  1 : line width
qqPlot(lm.blood.wt.no3$residuals, las = 1, id.n = 3, main="QQ Plot")

## 3 8 2
## 8 1 2

# residuals vs order of data
plot(lm.blood.wt.no3$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



How much difference is there in a practical sense? Examine the 95% prediction interval for a new observation at Weight = 50kg. Previously we saw that interval based on all 8 observations was from 457.1 to 517.8 ml of Blood Loss. Based on just the 7 observations the prediction interval is 451.6 to 512.4 ml. There really is no practical difference here.

```
# CI for the mean and PI for a new observation at weight=50
predict(lm.blood.wt    , data.frame(weight=50), interval = "prediction")
##        fit     lwr      upr
## 1 487.4257 457.098 517.7533
predict(lm.blood.wt.no3, data.frame(weight=50), interval = "prediction")
```

```
## Warning in predict.lm(lm.blood.wt.no3, data.frame(weight = 50), interval = "prediction"):
Assuming constant prediction variance even though model fit is weighted
##        fit      lwr      upr
## 1 481.9939 451.5782 512.4096
```

Therefore, while obs. 3 was potentially influential, whether the value is included or not makes very little difference in the model fit or relationship between Weight and BloodLoss.

8.10.2 Gesell data

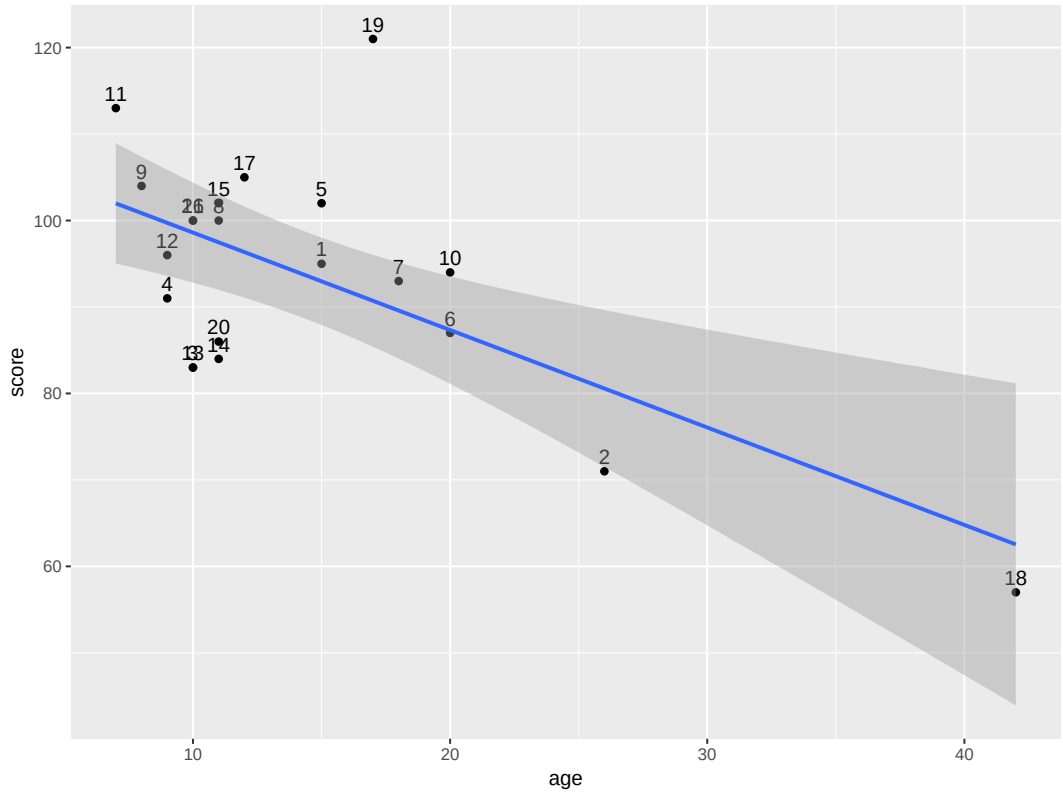
These data are from a UCLA study of cyanotic heart disease in children. The predictor is the age of the child in months at first word and the response variable is the Gesell adaptive score, for each of 21 children.

	id	age	score
1	1	15	95
2	2	26	71
3	3	10	83
4	4	9	91
5	5	15	102
6	6	20	87
7	7	18	93
8	8	11	100
9	9	8	104
10	10	20	94
11	11	7	113
12	12	9	96
13	13	10	83
14	14	11	84
15	15	11	102
16	16	10	100
17	17	12	105
18	18	42	57
19	19	17	121
20	20	11	86
21	21	10	100

Let us go through the same steps as before.

1. Plot Score versus Age. Comment on the relationship between Score and Age.

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(gesell, aes(x = age, y = score, label = id))
p <- p + geom_point()
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5)
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
print(p)
```



2. There are no obvious transformations to try here.
3. Fit a simple linear regression model. Provide an equation for the LS line. Does age at first word appear to be an "important predictor" of Gesell adaptive score? (i.e., is the estimated slope significantly different from zero?)

```
lm.score.age <- lm(score ~ age, data = gesell)
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm.score.age)
```

```
##
## Call:
## lm(formula = score ~ age, data = gesell)
##
```

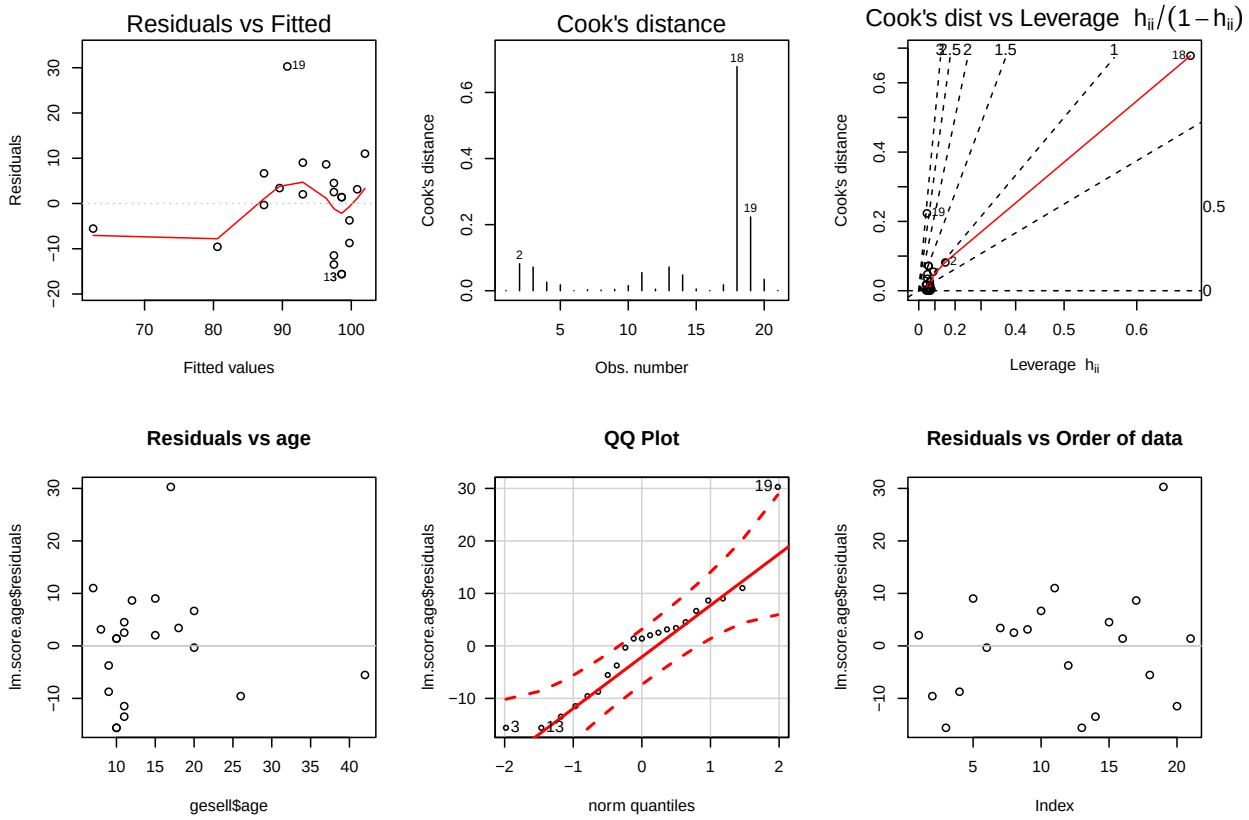
```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.604  -8.731   1.396   4.523  30.285
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 109.8738     5.0678  21.681 7.31e-15 ***
## age          -1.1270     0.3102  -3.633  0.00177 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.02 on 19 degrees of freedom
## Multiple R-squared:   0.41,Adjusted R-squared:  0.3789
## F-statistic:  13.2 on 1 and 19 DF,  p-value: 0.001769
```

4. Do these plots suggest any inadequacies with the model?

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.score.age, which = c(1,4,6))

# residuals vs weight
plot(gesell$age, lm.score.age$residuals, main="Residuals vs age")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.score.age$residuals, las = 1, id.n = 3, main="QQ Plot")
```

```
## 19  3 13
## 21  1  2
```

```
# residuals vs order of data
plot(lm.score.age$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



5. Observations 18 and 19 stand out with relatively high Cook's D. The cutoff line is only a rough guideline. Those two were flagged with high influence and standardized residual, respectively, also. Be sure to examine the scatter plot carefully to see why 18 and 19 stand out.
6. Consider doing two additional analyses: Analyze the data after omitting case 18 only and analyze the data after omitting case 19 only. Refit the regression model for each of these two scenarios. Provide a summary table such as the following, giving the relevant summary statistics for the three analyses. Discuss the impact that observations 18 and 19 have individually on the fit of the model.

   When observation 18 is omitted, the estimated slope is not significantly different from zero (p-value = 0.1489), indicating that age is not an important predictor of Gesell score. This suggests that the significance of age as a predictor in the original analysis was due solely to the presence of observation 18. Note the dramatic decrease in R^2 after deleting observation 18.

The fit of the model appears to improve when observation 19 is omitted. For example, R^2 increases noticeably and the p-value for testing the significance of the slope decreases dramatically (in a relative sense). These tendencies would be expected based on the original plot. However, this improvement is misleading. Once observation 19 is omitted, observation 18 is much more influential. Again the significance of the slope is due to the presence of observation 18.

Feature	Full data	Omit 18	Omit 19
b_0	109.87	105.63	109.30
b_1	-1.13	-0.78	-1.19
$SE(b_0)$	5.07	7.16	3.97
$SE(b_1)$	0.31	0.52	0.24
R^2	0.41	0.11	0.57
p-val for $H_0 : \beta_1 = 0$	0.002	0.149	0.000

Can you think of any reasons to justify doing the analysis without observation 18?

If you include observation 18 in the analysis, you are assuming that the mean Gesell score is linearly related to age over the entire range of observed ages. Observation 18 is far from the other observations on age (age for observation 18 is 42; the second highest age is 26; the lowest age is 7). There are no children with ages between 27 and 41, so we have no information on whether the relationship is roughly linear over a significant portion of the range of ages. I am comfortable deleting observation 18 from the analysis because it's inclusion forces me to make an assumption that I can not check using these data. I am only willing to make predictions of Gesell score for children with ages roughly between 7 and 26. However, once this point is omitted, age does not appear to be an important predictor.

A more complete analysis would delete observation 18 and 19 together. What would you expect to see if you did this?

8.11 Weighted Least Squares

Earlier I indicated that nonconstant error variance can be addressed (sometimes) with weighted least squares. The *scedastic function* is the conditional variance of Y given X. Y is said to be **heteroscedastic** if it's variance depends on the value of X (variance changes), and **homoscedastic** if it's variance does not depend on X (constant variance).

Recall the LS (OLS or ordinary LS) line chooses the values of β_0 and β_1 that minimize

$$\sum_{i=1}^{n}\{Y_i - (\beta_0 + \beta_1 X_i)\}^2$$

over all possible choices of β_0 and β_1. The weighted LS (WLS) line chooses the values of β_0 and β_1 that minimize

$$\sum_{i=1}^{n} w_i\{Y_i - (\beta_0 + \beta_1 X_i)\}^2$$

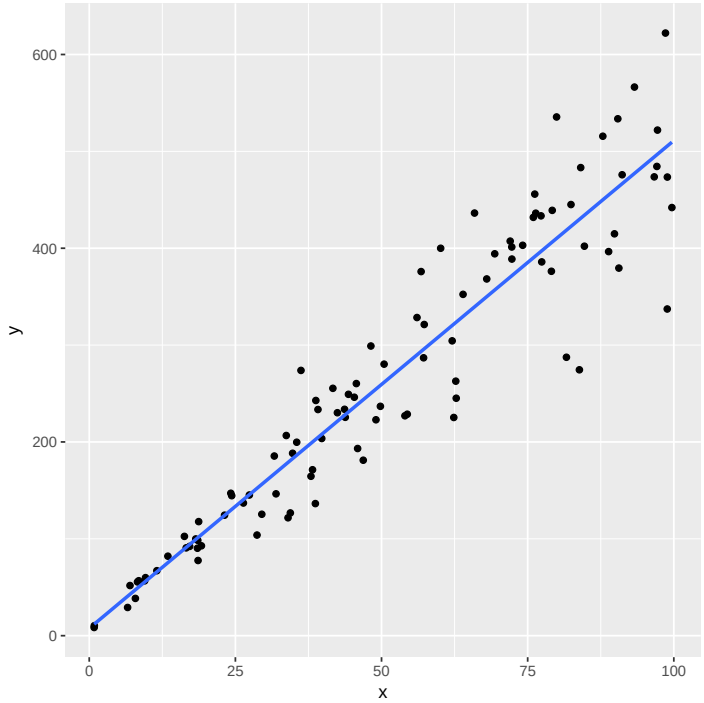
over all possible choices of β_0 and β_1. If $\sigma_{Y|X}$ depends up X, then the correct choice of weights is inversely proportional to *variance*, $w_i \propto \sigma^2_{Y|X}$.

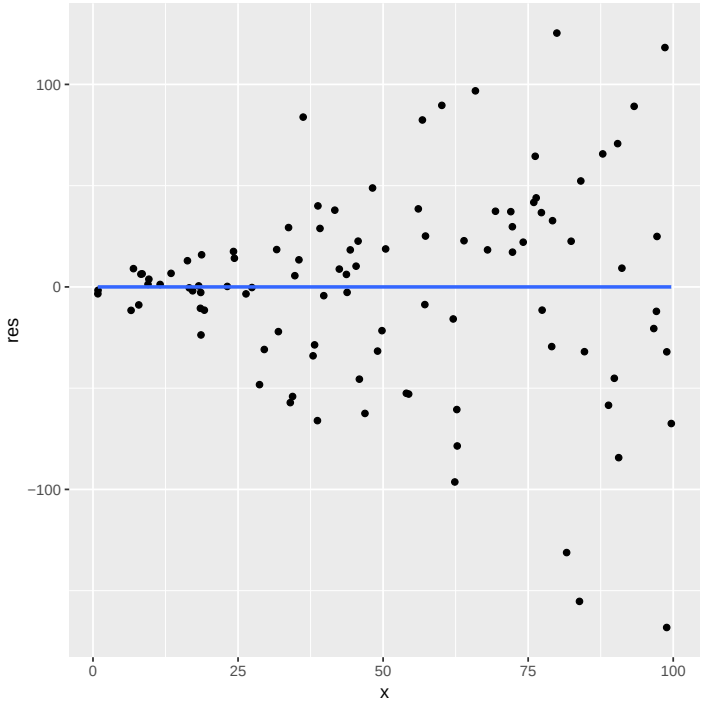
Consider the following data and plot of y vs. x and standardized OLS residuals vs x. It is very clear that variability increases with x.

```
#### Weighted Least Squares
# R code to generate data
set.seed(7)
n <- 100
# 1s, Xs uniform 0 to 100
X <- matrix(c(rep(1,n),runif(n,0,100)), ncol=2)
# intercept and slope (5, 5)
beta <- matrix(c(5,5),ncol=1)
# errors are X*norm(0,1), so variance increases with X
e <- X[,2]*rnorm(n,0,1)
# response variables
y <- X %*% beta + e

# put data into data.frame
wlsdat <- data.frame(y, x = X[,2])

# fit regression
lm.y.x <- lm(y ~ x, data = wlsdat)

# put residuals in data.frame
wlsdat$res <- lm.y.x$residuals
```

```
# ggplot: Plot the data with linear regression fit
library(ggplot2)
p <- ggplot(wlsdat, aes(x = x, y = y))
```

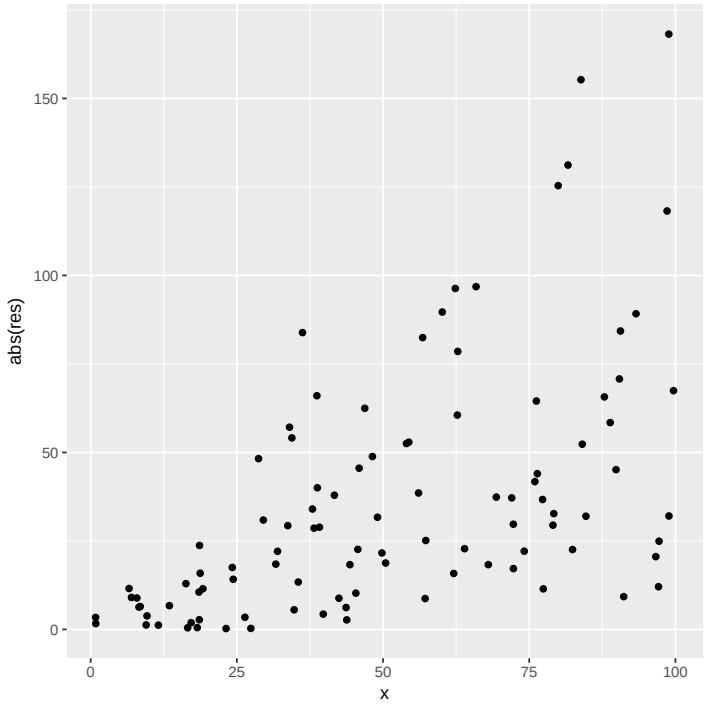
```
p <- p + geom_point()
p <- p + geom_smooth(method = lm, se = FALSE)
print(p)


# ggplot: Plot the residuals
library(ggplot2)
p <- ggplot(wlsdat, aes(x = x, y = res))
p <- p + geom_point()
p <- p + geom_smooth(method = lm, se = FALSE)
print(p)
```





In order to use WLS to solve this problem, we need some form for $\sigma^2_{Y|X}$. Finding that form is a real problem with WLS. It can be useful to plot the absolute value of the standardized residual vs. x to see if the top boundary seems to follow a general pattern.

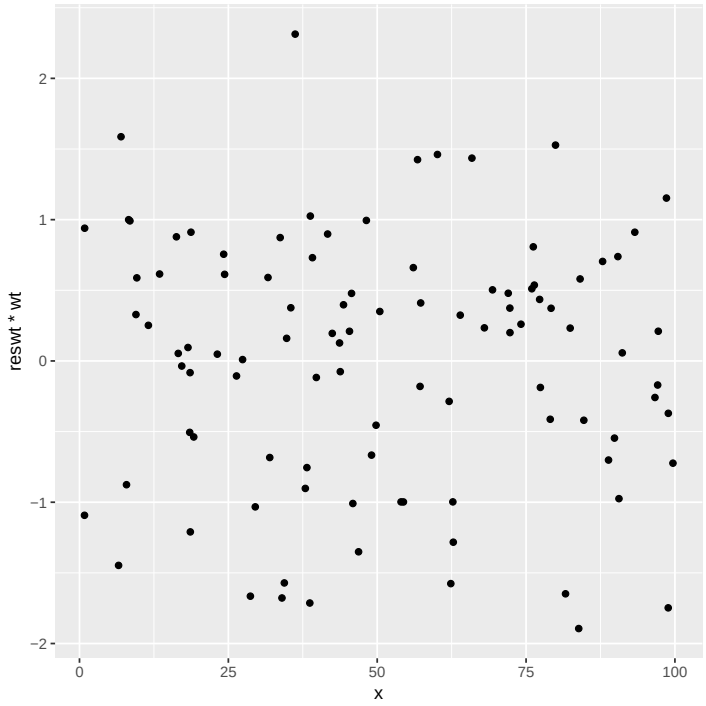
```
# ggplot: Plot the absolute value of the residuals
library(ggplot2)
p <- ggplot(wlsdat, aes(x = x, y = abs(res)))
p <- p + geom_point()
print(p)
```



It is plausible the upper boundary is linear, so let us try $w_i = \frac{1}{x^2}$. Standardized residuals from this WLS fit look very good. Note that raw (non-standardized) residuals will still have the same pattern — it is essential to use standardized residuals here.

```
# fit regression
lm.y.x.wt <- lm(y ~ x, data = wlsdat, weights = x^(-2))

# put residuals in data.frame
wlsdat$reswt <- lm.y.x.wt$residuals
wlsdat$wt <- lm.y.x.wt$weights^(1/2)
```

```
# ggplot: Plot the absolute value of the residuals
library(ggplot2)
p <- ggplot(wlsdat, aes(x = x, y = reswt*wt))
p <- p + geom_point()
print(p)
```



Compare also the OLS fitted equation:

```
summary(lm.y.x)
```

```
##
## Call:
## lm(formula = y ~ x, data = wlsdat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -168.175  -24.939    2.542   24.973  125.366
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.6308    10.4256   0.732    0.466
## x             5.0348     0.1791  28.116   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 50.53 on 98 degrees of freedom
## Multiple R-squared:  0.8897,Adjusted R-squared:  0.8886
## F-statistic: 790.5 on 1 and 98 DF,  p-value: < 2.2e-16
```

to the WLS fitted equation:

```
summary(lm.y.x.wt)
```

```
##
## Call:
## lm(formula = y ~ x, data = wlsdat, weights = x^(-2))
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -1.8939 -0.6707  0.1777  0.5963  2.3132
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.09309    0.54931   9.272 4.61e-15 ***
## x            5.10690    0.09388  54.399  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8902 on 98 degrees of freedom
## Multiple R-squared:  0.9679,Adjusted R-squared:  0.9676
## F-statistic:  2959 on 1 and 98 DF,  p-value: < 2.2e-16
```

Clearly the weighted fit looks better, although note that everything is based on the weighted SS. In practice it can be pretty difficult to determine the correct set of weights, but WLS works much better than OLS if appropriate. I actually simulated this data set using $\beta_0 = \beta_1 = 5$. Which fit actually did better?

Part IV

Additional topics

Chapter 9

Introduction to the Bootstrap

Contents


```
## Warning in file(filename, "r", encoding = encoding):  cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding):  cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

explain the bootstrap principle for hypothesis tests and inference.

decide (for simple problems) how to construct a bootstrap procedure.

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

1. organize knowledge.
5. define parameters of interest and hypotheses in words and notation.
8. use statistical software.
12. make evidence-based decisions.

9.1 Introduction

Statistical theory attempts to answer three basic questions:

1. How should I collect my data?
2. How should I analyze and summarize the data that I've collected?
3. How accurate are my data summaries?

Question 3 consitutes part of the process known as statistical inference. The bootstrap makes certain kinds of statistical inference[1]. Let's look at an example.

Example: Aspirin and heart attacks, large-sample theory Does aspirin prevent heart attacks in healthy middle-aged men? A controlled, randomized, double-blind study was conducted and gathered the following data.

	heart attacks (fatal plus non-fatal)	subjects
aspirin group:	104	11037
placebo group:	189	11034

A good experimental design, such as this one, simplifies the results! The ratio of the two rates (the risk ratio) is

$$\hat{\theta} = \frac{104/11037}{189/11034} = 0.55.$$

Because of the solid experimental design, we can believe that the aspirin-takers only have 55% as many heart attacks as the placebo-takers.

[1]Efron (1979), "Bootstrap methods: another look at the jackknife." Ann. Statist. 7, 1–26

We are not really interested in the estimated ratio $\hat{\theta}$, but the true ratio, θ. That is the ratio if we could treat all possible subjects, not just a sample of them. Large sample theory tells us that the log risk ratio has an approximate Normal distribution. The standard error of the log risk ratio is estimated simply by the square root of the sum of the reciprocals of the four frequencies:

$$\text{SE}(\log(RR)) = \sqrt{\frac{1}{104} + \frac{1}{189} + \frac{1}{11037} + \frac{1}{11034}} = 0.1228$$

The 95% CI for $\log(\theta)$ is

$$\log(\hat{\theta}) \pm 1.96 \times \text{SE}(\log(RR)), \quad (-0.839, -0.357),$$

and expontiating gives the CI on the ratio scale,

$$\exp\{\log(\hat{\theta}) \pm 1.96 \times \text{SE}(\log(RR))\}, \quad (0.432, 0.700).$$

The same data that allowed us to estimate the ratio θ with $\hat{\theta} = 0.55$ also allowed us to get an idea of the estimate's accuracy.

Example: Aspirin and strokes, large-sample theory The aspirin study tracked strokes as well as heart attacks.

	strokes	subjects
aspirin group:	119	11037
placebo group:	98	11034

The ratio of the two rates (the risk ratio) is

$$\hat{\theta} = \frac{119/11037}{98/11034} = 1.21.$$

It looks like aspirin is actually harmful, now, however the 95% interval for the true stroke ratio θ is $(0.925, 1.583)$. This includes the neutral value $\theta = 1$, at which aspirin would be no better or worse than placebo for strokes.

9.2 Bootstrap

The bootstrap is a data-based simulation method for statistical inference, which can be used to produce inferences like those in the previous slides. The term "bootstrap" comes from literature. In "The Adventures of Baron Munchausen", by Rudolph Erich Raspe, the Baron had fallen to the bottom of a deep lake, and he thought to get out by *pulling himself up by his own bootstraps*.

9.2.1 Ideal versus Bootstrap world, sampling distributions

Ideal world

1. Population of interest
2. Obtain many simple random samples (SRSs) of size n
3. For each SRS, calculate statistic of interest (θ)
4. Sampling distribution is the distribution of the calculated statistic

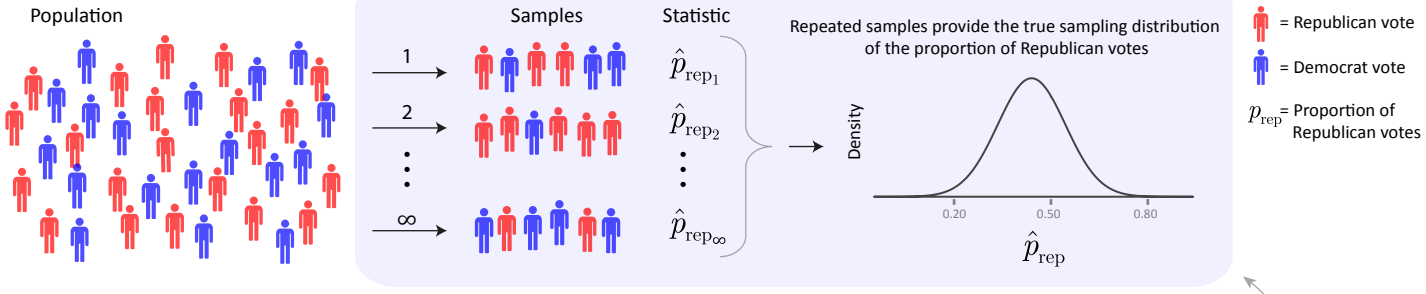
Bootstrap world

1. Population of interest; One empirical distribution based on a sample of size n
2. Obtain many bootstrap resamples of size n
3. For each resample, calculate statistic of interest (θ^*)
4. Bootstrap distribution is the distribution of the calculated statistic
5. Bootstrap distribution estimates the sampling distribution centered at the statistic (not the parameter).

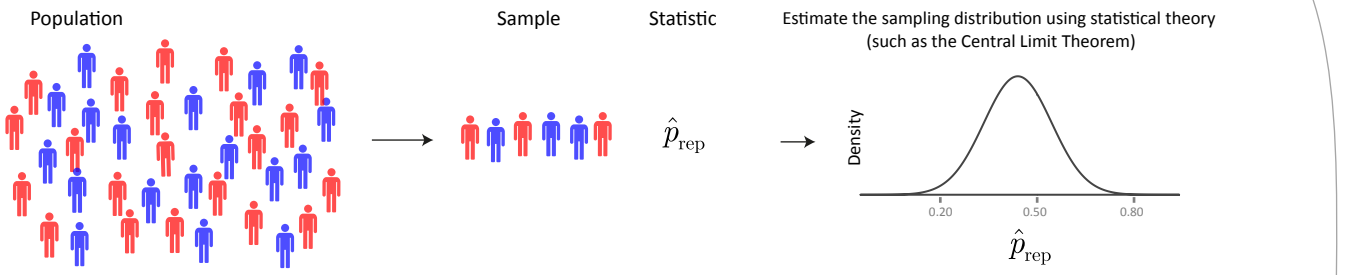
Cartoon: Estimating the proportion of Republican votes Imagine a two-candidate eleection. The following illustrates how to use exit polls to estimate (with uncertainty) the probability of a Republican win.

How can the sampling distribution of the proportion of Republican votes be estimated?

The ideal case: draw repeated (infinite) samples from the population

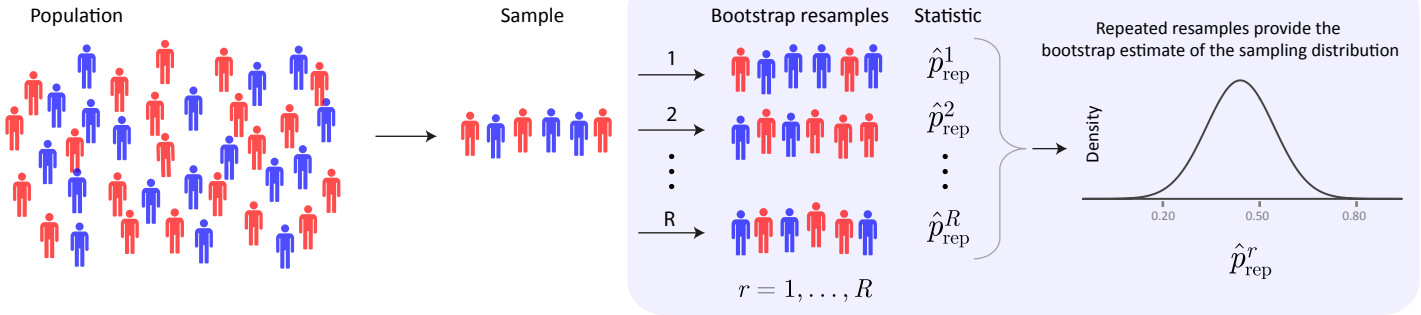


The traditional case: a single sample is observed

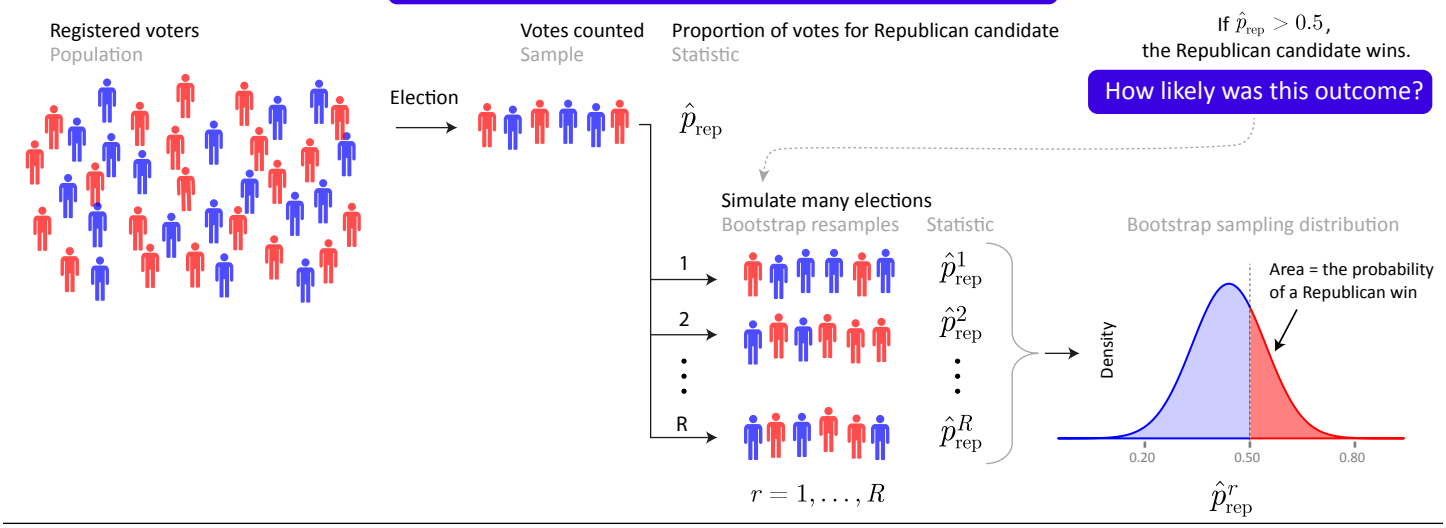


Note the similar structure between the ideal and bootstrap cases

The bootstrap case: a single sample is resampled



How is the bootstrap used in this scenario?



Example: Aspirin and strokes, bootstrap Here's how the bootstrap works in the stroke example. We create two populations:

- the first consisting of 119 ones and $11037 - 119 = 10918$ zeros,
- the second consisting of 98 ones and $11034 - 98 = 10936$ zeros.

We draw with replacement a sample of 11037 items from the first population, and a sample of 11034 items from the second population. Each is called a *bootstrap sample*. From these we derive the bootstrap replicate of $\hat{\theta}$:

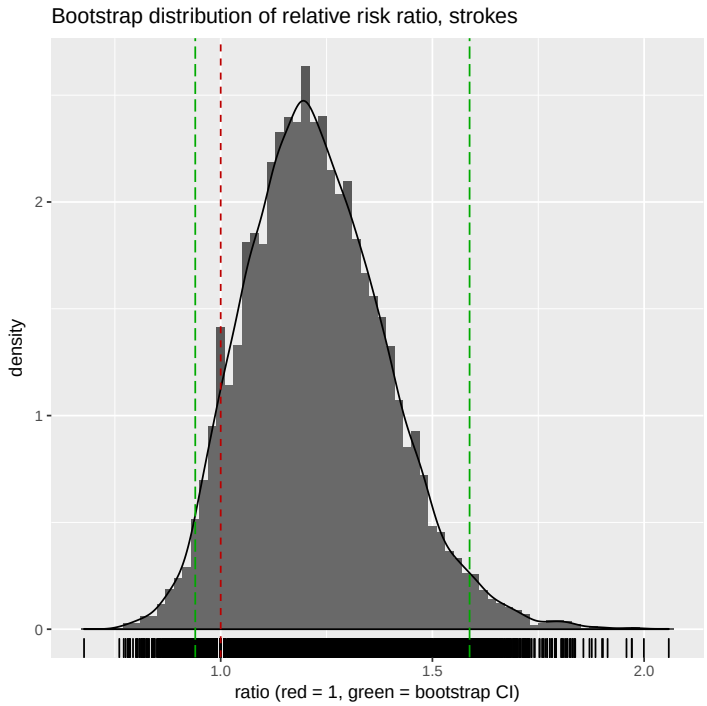
$$\hat{\theta}^* = \frac{\text{Proportion of ones in bootstrap sample 1}}{\text{Proportion of ones in bootstrap sample 2}}.$$

Repeat this process a large number of times, say 10000 times, and obtain 10000 *bootstrap replicates* $\hat{\theta}^*$. The summaries are in the code, followed by a histogram of bootstrap replicates, $\hat{\theta}^*$.

```
#### Example: Aspirin and strokes, bootstrap
# sample size (n) and successes (s) for sample 1 (aspirin) and 2 (placebo)
n <- c(11037, 11034)
s <- c(  119,    98)
# data for samples 1 and 2, where 1 = success (stroke), 0 = failure (no stroke)
dat1 <- c(rep(1, s[1]), rep(0, n[1] - s[1]))
dat2 <- c(rep(1, s[2]), rep(0, n[2] - s[2]))
# draw R bootstrap replicates
R <- 10000
# init location for bootstrap samples
bs1 <- rep(NA, R)
bs2 <- rep(NA, R)
# draw R bootstrap resamples of proportions
for (i in 1:R) {
  # proportion of successes in bootstrap samples 1 and 2
  # (as individual steps for group 1:)
  resam1 <- sample(dat1, n[1], replace = TRUE)
  success1 <- sum(resam1)
  bs1[i] <- success1 / n[1]
  # (as one line for group 2:)
  bs2[i] <- sum(sample(dat2, n[2], replace = TRUE)) / n[2]
}
# bootstrap replicates of ratio estimates
rat <- bs1 / bs2
# sort the ratio estimates to obtain bootstrap CI
rat.sorted <- sort(rat)
# 0.025th and 0.975th quantile gives equal-tail bootstrap CI
CI.bs <- c(rat.sorted[round(0.025*R)], rat.sorted[round(0.975*R+1)])
CI.bs
```

```
## [1] 0.9399154 1.5878036
```

```
## Plot the bootstrap distribution with CI
# First put data in data.frame for ggplot()
dat.rat <- data.frame(rat)

library(ggplot2)
p <- ggplot(dat.rat, aes(x = rat))
p <- p + geom_histogram(aes(y=..density..), binwidth=0.02)
p <- p + geom_density(alpha=0.1, fill="white")
p <- p + geom_rug()
# vertical line at 1 and CI
p <- p + geom_vline(xintercept=1, colour="#BB0000", linetype="dashed")
p <- p + geom_vline(xintercept=CI.bs[1], colour="#00AA00", linetype="longdash")
p <- p + geom_vline(xintercept=CI.bs[2], colour="#00AA00", linetype="longdash")
p <- p + labs(title = "Bootstrap distribution of relative risk ratio, strokes")
p <- p + xlab("ratio (red = 1, green = bootstrap CI)")
print(p)
```



In this simple case, the confidence interval derived from the bootstrap $(0.94, 1.588)$ agrees very closely with the one derived from statistical theory $(0.925, 1.583)$. Bootstrap methods are intended to simplify the calculation of inferences like those using large-sample theory, producing them in an automatic way even in situations much more complicated than the risk ratio in the aspirin example.

9.2.2 The accuracy of the sample mean

For sample means, and essentially *only* for sample means, an accuracy formula (for the standard error of the parameter) is easy to obtain (using the delta method). We'll see how to use the bootstrap for the sample mean, then for the more complicated situation of assessing the accuracy of the median.

Bootstrap Principle The **plug-in principle** is used when the underlying distribution is unknown and you substitute your best guess for what that distribution is. What to substitute?

Empirical distribution ordinary bootstrap
Smoothed distribution (kernel) smoothed bootstrap
Parametric distribution parametric bootstrap
Satisfy assumptions such as the null hypothesis

This substitution works in many cases, but not always. Keep in mind that the bootstrap distribution is centered at the statistic, not the parameter. Implemention is done by Monte Carlo sampling.

The bootstrap is commonly implemented in one of two ways, nonparametrically or parametrically. An *exact* **nonparametric bootstrap** requires n^n samples! That's one for every possible combination of each of n observation positions taking the value of each of n observations. This is sensibly approximated by using the Monte Carlo strategy of drawing a large number (1000 or 10000) of random resamples. On the other hand, a **parametric bootstrap** first assumes a distribution for the population (such as a normal distribution) and estimates the distributional parameters (such as the mean and variance) from the observed sample. Then, the Monte Carlo strategy is used to draw a large number (1000 or 10000) of samples from the estimated parametric distribution.

Example: Mouse survival, two-sample t-test, mean Sixteen mice were randomly assigned to a treatment group or a control group. Shown are their survival times, in days, following a test surgery. Did the treatment prolong survival?

Group	Data	n	Mean	SE
Control:	52, 104, 146, 10, 51, 30, 40, 27, 46	9	56.22	14.14
Treatment:	94, 197, 16, 38, 99, 141, 23	7	86.86	25.24
		Difference:	30.63	28.93

Numerical and graphical summaries of the data are below. There seems to be a slight difference in variability between the two treatment groups.

```
#### Example: Mouse survival, two-sample t-test, mean
treatment <- c(94, 197, 16, 38, 99, 141, 23)
control   <- c(52, 104, 146, 10, 51, 30, 40, 27, 46)
survive <- c(treatment, control)
group   <- c(rep("Treatment", length(treatment)), rep("Control", length(control)))
mice <- data.frame(survive, group)

library(plyr)
# ddply "dd" means the input and output are both data.frames
mice.summary <- ddply(mice,
                      "group",
                      function(X) {
                        data.frame( m = mean(X$survive),
                                    s = sd(X$survive),
                                    n = length(X$survive)
                                  )
                      }
                     )
# standard errors
mice.summary$se   <- mice.summary$s/sqrt(mice.summary$n)
# individual confidence limits
mice.summary$ci.l <- mice.summary$m - qt(1-.05/2, df=mice.summary$n-1) * mice.summary$se
mice.summary$ci.u <- mice.summary$m + qt(1-.05/2, df=mice.summary$n-1) * mice.summary$se
```

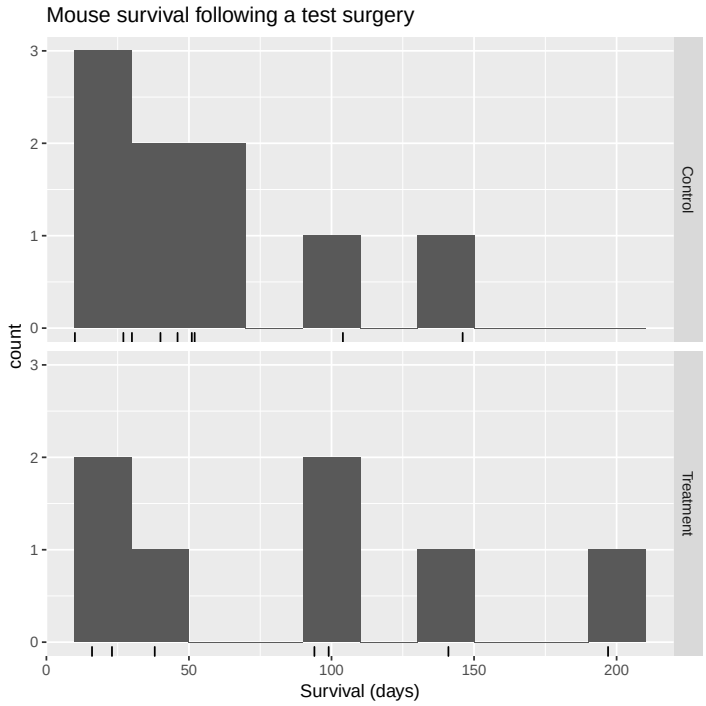
```
mice.summary
```

```
##       group        m        s n       se     ci.l      ci.u
## 1   Control 56.22222 42.47581 9 14.15860 23.57242  88.87202
## 2 Treatment 86.85714 66.76683 7 25.23549 25.10812 148.60616
```

```
diff(mice.summary$m)                                                    #£
```

```
## [1] 30.63492
```

```
# histogram using ggplot
p <- ggplot(mice, aes(x = survive))
p <- p + geom_histogram(binwidth = 20)
p <- p + geom_rug()
p <- p + facet_grid(group ~ .)
p <- p + labs(title = "Mouse survival following a test surgery") + xlab("Survival (days)")
print(p)
```



The standard error for the difference is $28.93 = \sqrt{25.24^2 + 14.14^2}$, so the observed difference of 30.63 is only 30.63/28.93=1.05 estimated standard errors greater than zero, an *insignificant* result.

The two-sample t-test of the difference in means confirms the lack of statistically significant difference between these two treatment groups with a p-value=0.3155.

```
t.test(survive ~ group, data = mice)
```

```
##
##  Welch Two Sample t-test
##
## data:  survive by group
## t = -1.0587, df = 9.6545, p-value = 0.3155
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -95.42263  34.15279
## sample estimates:
##   mean in group Control mean in group Treatment
##                56.22222                 86.85714
```

But these are small samples, and the control sample does not look normal. We could do a nonparametric two-sample test of difference of medians. Or, we could use the bootstrap to make our inference.

Example: Mouse survival, two-sample bootstrap, mean Here's how the bootstrap works in the two-sample mouse example. We draw with replacement from each sample, calculate the mean for each sample, then take the difference in means. Each is called a *bootstrap sample* of the difference in means. From these we derive the bootstrap replicate of $\hat{\mu}$:

$$\hat{\mu}^* = \bar{x}^* - \bar{y}^*.$$

Repeat this process a large number of times, say 10000 times, and obtain 10000 *bootstrap replicates* $\hat{\mu}^*$. The summaries are in the code, followed by a histogram of bootstrap replicates, $\hat{\mu}^*$.

```
#### Example: Mouse survival, two-sample bootstrap, mean
# draw R bootstrap replicates
R <- 10000
# init location for bootstrap samples
bs1 <- rep(NA, R)
bs2 <- rep(NA, R)
# draw R bootstrap resamples of means
for (i in 1:R) {
  bs2[i] <- mean(sample(control,   replace = TRUE))
  bs1[i] <- mean(sample(treatment, replace = TRUE))
}
# bootstrap replicates of difference estimates
bs.diff <- bs1 - bs2
sd(bs.diff)
```

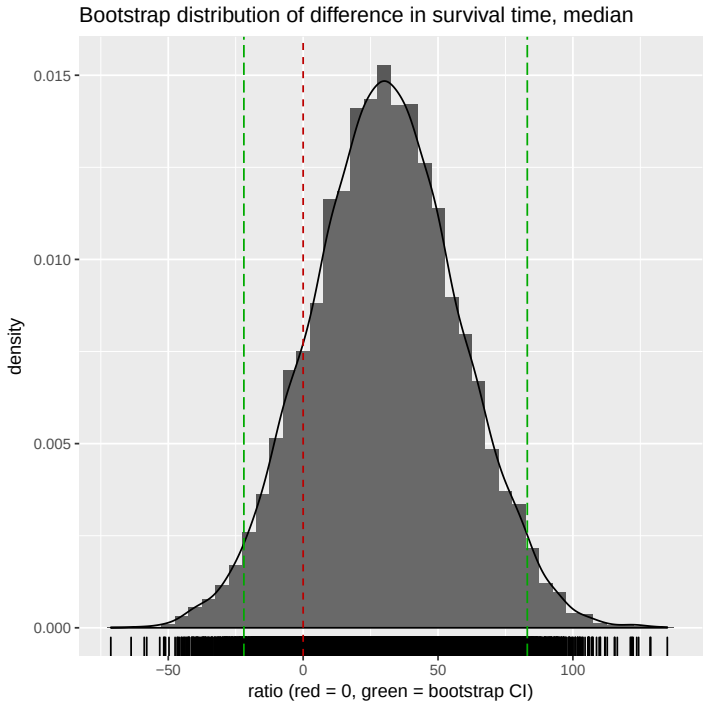
```
## [1] 27.00087
```

```
# sort the difference estimates to obtain bootstrap CI
diff.sorted <- sort(bs.diff)
# 0.025th and 0.975th quantile gives equal-tail bootstrap CI
CI.bs <- c(diff.sorted[round(0.025*R)], diff.sorted[round(0.975*R+1)])
CI.bs
```

```
## [1] -21.96825  83.09524
```

```
## Plot the bootstrap distribution with CI
# First put data in data.frame for ggplot()
dat.diff <- data.frame(bs.diff)

library(ggplot2)
p <- ggplot(dat.diff, aes(x = bs.diff))
p <- p + geom_histogram(aes(y=..density..), binwidth=5)
p <- p + geom_density(alpha=0.1, fill="white")
p <- p + geom_rug()
# vertical line at 0 and CI
p <- p + geom_vline(xintercept=0, colour="#BB0000", linetype="dashed")
```

```
p <- p + geom_vline(xintercept=CI.bs[1], colour="#00AA00", linetype="longdash")
p <- p + geom_vline(xintercept=CI.bs[2], colour="#00AA00", linetype="longdash")
p <- p + labs(title = "Bootstrap distribution of difference in survival time, median")
p <- p + xlab("ratio (red = 0, green = bootstrap CI)")
print(p)
```



Example: Mouse survival, two-sample bootstrap, median For most statistics (such as the median) we don't have a formula for the limiting value of the standard error, but in fact no formula is needed. Instead, we use the numerical output of the bootstrap program. The summaries are in the code, followed by a histogram of bootstrap replicates, $\hat{\eta}^*$.

Group	Data	(n)	Median	est. SE
Control:	52, 104, 146, 10, 51, 30, 40, 27, 46	(9)	46	?
Treatment:	94, 197, 16, 38, 99, 141, 23	(7)	94	?
		Difference:	48	?

```
#### Example: Mouse survival, two-sample bootstrap, median
sort(control)
## [1]  10  27  30  40  46  51  52 104 146
```

```r
sort(treatment)
```

```
## [1]  16  23  38  94  99 141 197
```

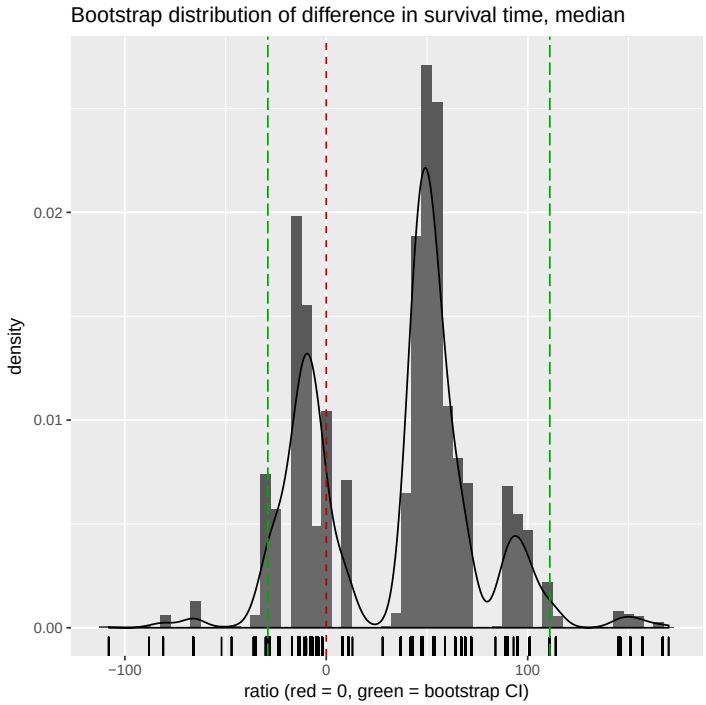
```r
# draw R bootstrap replicates
R <- 10000
# init location for bootstrap samples
bs1 <- rep(NA, R)
bs2 <- rep(NA, R)
# draw R bootstrap resamples of medians
for (i in 1:R) {
  bs2[i] <- median(sample(control,   replace = TRUE))
  bs1[i] <- median(sample(treatment, replace = TRUE))
}
# bootstrap replicates of difference estimates
bs.diff <- bs1 - bs2
sd(bs.diff)
```

```
## [1] 40.43024
```

```r
# sort the difference estimates to obtain bootstrap CI
diff.sorted <- sort(bs.diff)
# 0.025th and 0.975th quantile gives equal-tail bootstrap CI
CI.bs <- c(diff.sorted[round(0.025*R)], diff.sorted[round(0.975*R+1)])
CI.bs
```

```
## [1] -29 111
```

```r
## Plot the bootstrap distribution with CI
# First put data in data.frame for ggplot()
dat.diff <- data.frame(bs.diff)

library(ggplot2)
p <- ggplot(dat.diff, aes(x = bs.diff))
p <- p + geom_histogram(aes(y=..density..), binwidth=5)
p <- p + geom_density(alpha=0.1, fill="white")
p <- p + geom_rug()
# vertical line at 0 and CI
p <- p + geom_vline(xintercept=0, colour="#BB0000", linetype="dashed")
p <- p + geom_vline(xintercept=CI.bs[1], colour="#00AA00", linetype="longdash")
p <- p + geom_vline(xintercept=CI.bs[2], colour="#00AA00", linetype="longdash")
p <- p + labs(title = "Bootstrap distribution of difference in survival time, median")
p <- p + xlab("ratio (red = 0, green = bootstrap CI)")
print(p)
```



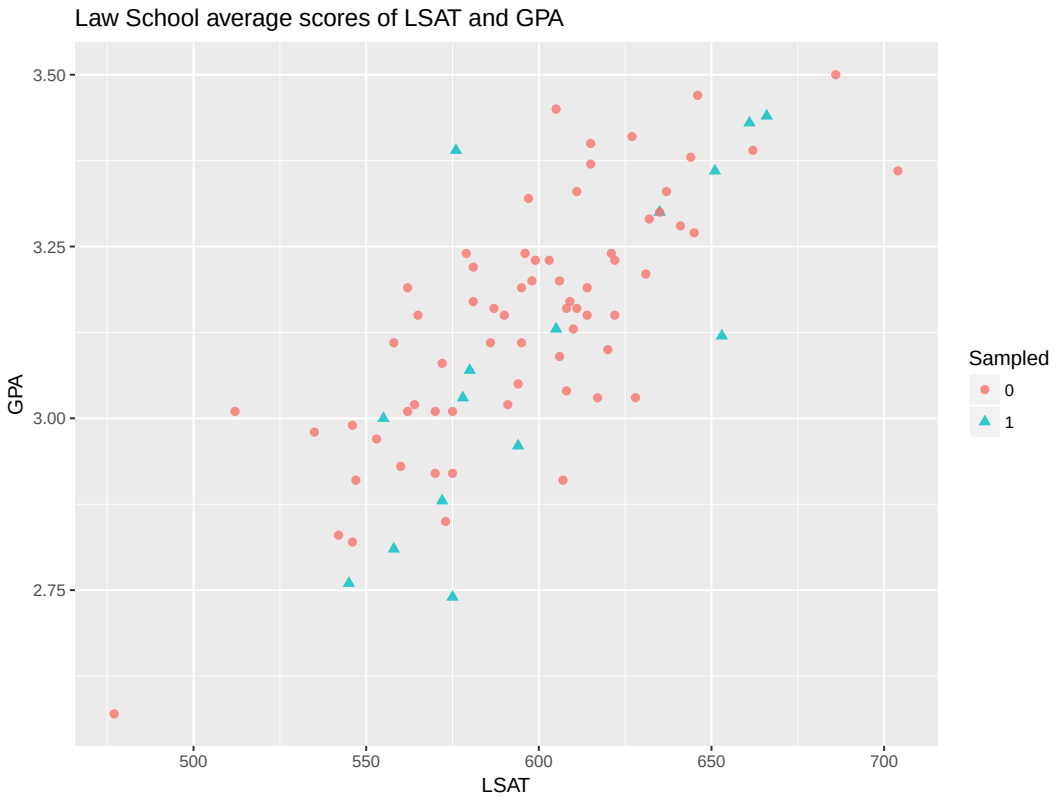
9.2.3 Comparing bootstrap sampling distribution from population and sample

Example: Law School, correlation of (LSAT, GPA) The population of average student measurements of (LSAT, GPA) for the universe of 82 law schools are in the table below. Imagine that we don't have all 82 schools worth of data. Consider taking a random sample of 15 schools, indicated by the +'s.

School	LSAT	GPA	School	LSAT	GPA	School	LSAT	GPA
1	622	3.23	28	632	3.29	56	641	3.28
2	542	2.83	29	587	3.16	57	512	3.01
3	579	3.24	30	581	3.17	58	631	3.21
4+	653	3.12	31+	605	3.13	59	597	3.32
5	606	3.09	32	704	3.36	60	621	3.24
6+	576	3.39	33	477	2.57	61	617	3.03
7	620	3.10	34	591	3.02	62	637	3.33
8	615	3.40	35+	578	3.03	62	572	3.08
9	553	2.97	36+	572	2.88	64	610	3.13
10	607	2.91	37	615	3.37	65	562	3.01
11	558	3.11	38	606	3.20	66	635	3.30
12	596	3.24	39	603	3.23	67	614	3.15
13+	635	3.30	40	535	2.98	68	546	2.82
14	581	3.22	41	595	3.11	69	598	3.20
15+	661	3.43	42	575	2.92	70+	666	3.44
16	547	2.91	43	573	2.85	71	570	3.01
17	599	3.23	44	644	3.38	72	570	2.92
18	646	3.47	45+	545	2.76	73	605	3.45
19	622	3.15	46	645	3.27	74	565	3.15
20	611	3.33	47+	651	3.36	75	686	3.50
21	546	2.99	48	562	3.19	76	608	3.16
22	614	3.19	49	609	3.17	77	595	3.19
23	628	3.03	50+	555	3.00	78	590	3.15
24	575	3.01	51	586	3.11	79+	558	2.81
25	662	3.39	52+	580	3.07	80	611	3.16
26	627	3.41	53+	594	2.96	81	564	3.02
27	608	3.04	54	594	3.05	82+	575	2.74
			55	560	2.93			

```
#### Example: Law School, correlation of (LSAT, GPA)
School  <- 1:82

LSAT    <- c(622, 542, 579, 653, 606, 576, 620, 615, 553, 607, 558, 596, 635,
             581, 661, 547, 599, 646, 622, 611, 546, 614, 628, 575, 662, 627,
             608, 632, 587, 581, 605, 704, 477, 591, 578, 572, 615, 606, 603,
             535, 595, 575, 573, 644, 545, 645, 651, 562, 609, 555, 586, 580,
             594, 594, 560, 641, 512, 631, 597, 621, 617, 637, 572, 610, 562,
             635, 614, 546, 598, 666, 570, 570, 605, 565, 686, 608, 595, 590,
             558, 611, 564, 575)

GPA     <- c(3.23, 2.83, 3.24, 3.12, 3.09, 3.39, 3.10, 3.40, 2.97, 2.91, 3.11,
             3.24, 3.30, 3.22, 3.43, 2.91, 3.23, 3.47, 3.15, 3.33, 2.99, 3.19,
```

```
             3.03, 3.01, 3.39, 3.41, 3.04, 3.29, 3.16, 3.17, 3.13, 3.36, 2.57,
             3.02, 3.03, 2.88, 3.37, 3.20, 3.23, 2.98, 3.11, 2.92, 2.85, 3.38,
             2.76, 3.27, 3.36, 3.19, 3.17, 3.00, 3.11, 3.07, 2.96, 3.05, 2.93,
             3.28, 3.01, 3.21, 3.32, 3.24, 3.03, 3.33, 3.08, 3.13, 3.01, 3.30,
             3.15, 2.82, 3.20, 3.44, 3.01, 2.92, 3.45, 3.15, 3.50, 3.16, 3.19,
             3.15, 2.81, 3.16, 3.02, 2.74)

Sampled <- c(0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
             0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
             1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
             0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1)

# law = population
law <- data.frame(School, LSAT,  GPA, Sampled)
law$Sampled <- factor(law$Sampled)
# law.sam = sample
law.sam <- subset(law, Sampled == 1)

library(ggplot2)
p <- ggplot(law, aes(x = LSAT, y = GPA))
p <- p + geom_point(aes(colour = Sampled, shape = Sampled), alpha = 0.8, size = 2)
p <- p + labs(title = "Law School average scores of LSAT and GPA")
print(p)
```



Let's bootstrap the sample of 15 observations to get the bootstrap sampling distribution of correlation (for sampling 15 from the population). From the

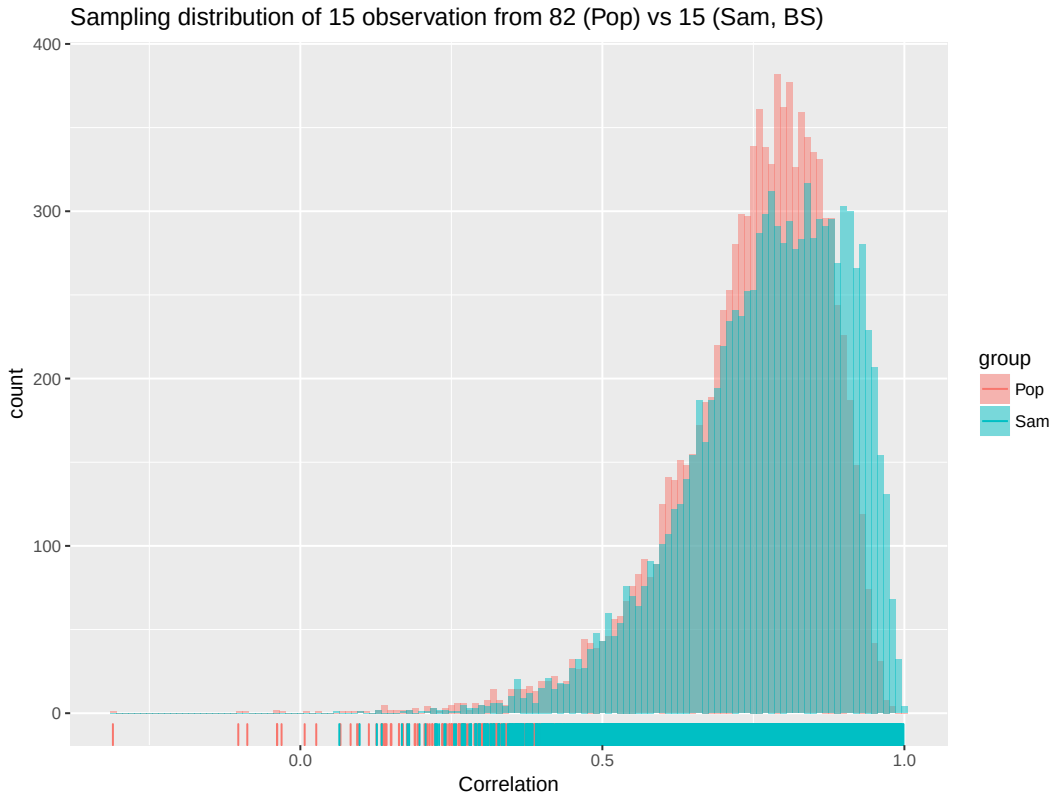
bootstrap sampling distribution we'll calculate a bootstrap confidence interval for the true population correlation, as well as a bootstrap standard deviation for the correlation. But how well does this work? Let's compare it against the *true* sampling distribution by drawing 15 random schools from the population of 82 schools and calculating the correlation. If the bootstrap works well (from our hopefully representative sample of 15), then the bootstrap sampling distribution from the 15 schools will be close to the true sampling distribution.

The code below does that, followed by two histograms. In this case, the histograms are noticeably non-normal, having a long tail toward the left. Inferences based on the normal curve are suspect when the bootstrap histogram is markedly non-normal. The histogram on the left is the nonparametric bootstrap sampling distribution using only the $n = 15$ sampled schools with 10000 bootstrap replicates of $\widehat{\text{corr}}(x^*)$. The histogram on the right is the true sampling distribution using 10000 replicates of $\widehat{\text{corr}}(x^*)$ from the population of law school data, repeatedly drawing $n = 15$ without replacement from the $N = 82$ points. Impressively, the bootstrap histogram on the left strongly resembles the population histogram on the right. Remember, in a real problem we would only have the information on the left, from which we would be trying to infer the situation on the right.

```
# draw R bootstrap replicates
R <- 10000
# init location for bootstrap samples
bs.pop <- rep(NA, R)
bs.sam <- rep(NA, R)
# draw R bootstrap resamples of medians
for (i in 1:R) {
  # sample() draws indicies then bootstrap correlation of LSAT and GPA
  # population
  bs.pop[i] = cor(law    [sample(seq(1,nrow(law    )), nrow(law.sam)
                                 , replace = TRUE), 2:3])[1, 2]
  # sample
  bs.sam[i] = cor(law.sam[sample(seq(1,nrow(law.sam)), nrow(law.sam)
                                 , replace = TRUE), 2:3])[1, 2]
}

# sort the difference estimates to obtain bootstrap CI
diff.sorted <- sort(bs.pop)
# 0.025th and 0.975th quantile gives equal-tail bootstrap CI
```

```
CI.bs.pop <- c(diff.sorted[round(0.025*R)], diff.sorted[round(0.975*R+1)])
# population correlation
cor(law    [, c(2,3)])[1,2]
## [1] 0.7599979
CI.bs.pop
## [1] 0.4296745 0.9271040
sd(bs.pop)
## [1] 0.1295076
# sort the difference estimates to obtain bootstrap CI
diff.sorted <- sort(bs.sam)
# 0.025th and 0.975th quantile gives equal-tail bootstrap CI
CI.bs.sam <- c(diff.sorted[round(0.025*R)], diff.sorted[round(0.975*R+1)])
# sample correlation
cor(law.sam[, c(2,3)])[1,2]
## [1] 0.7763745
CI.bs.sam
## [1] 0.4637826 0.9637982
sd(bs.sam)
## [1] 0.1334595
```

```
law.bs.df <- data.frame(corr = c(bs.pop, bs.sam), group = c(rep("Pop",R),rep("Sam",R)))
# histogram using ggplot
library(ggplot2)
p <- ggplot(law.bs.df, aes(x = corr, fill=group))
p <- p + geom_histogram(binwidth = .01, alpha = 0.5, position="identity")
p <- p + geom_rug(aes(colour=group))
p <- p + labs(title = "Sampling distribution of 15 observation from 82 (Pop) vs 15 (Sam, BS)")
         xlab("Correlation")
print(p)
```


Sampling distribution of 15 observation from 82 (Pop) vs 15 (Sam, BS)
400
300
200
100
0
count
0.0
0.5
1.0
Correlation
group
Pop
Sam

Chapter 10

Power and Sample size

Contents


```
## Warning in file(filename, "r", encoding = encoding):  cannot
open file 'ADA1_12_RFunctions.R': No such file or directory
## Error in file(filename, "r", encoding = encoding):  cannot
open the connection
```

Learning objectives

After completing this topic, you should be able to:

assess the power of a test or

determine the required sample size for a study.

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

7. Distinguish between statistical significance and scientific relevance.

10. Identify and explain the statistical methods, assumptions, and limitations.

12. Make evidence-based decisions by constructing and deciding between testable hypotheses using appropriate data and methods.

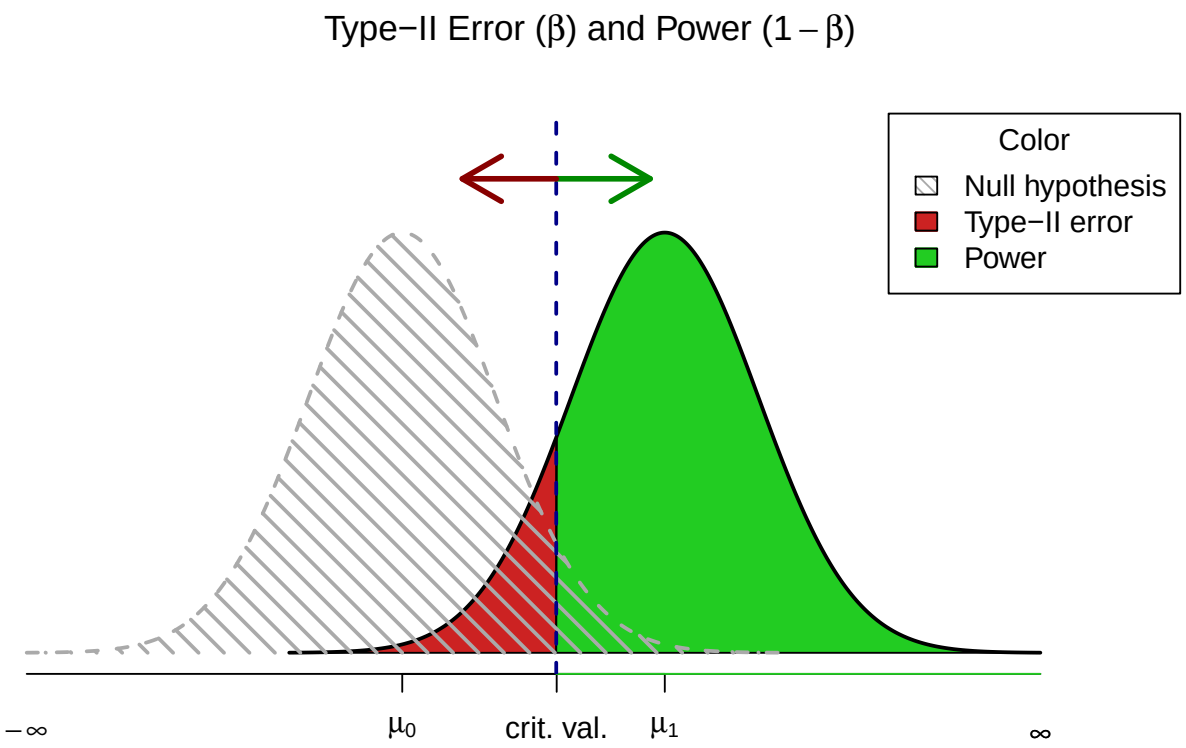
10.1 Power Analysis

The meaning of statistical power **Power** *is the probability* $(1-\beta)$ *of detecting an effect, given that the effect is really there.* In other words, it is the probability of correctly rejecting the null hypothesis when it is in fact false. For example, let's say that we have a simple study with drug A and a placebo group, and that the drug truly is effective; the power is the probability of finding a difference between the two groups. So, imagine that we had a power of $1-\beta = 0.8$ and that this simple study was conducted many times. Having power of 0.8 means that 80% of the time, we would get a statistically significant difference between the drug A and placebo groups. This also means that 20% of the times that we run this experiment, we will not obtain a statistically significant effect between the two groups, even though there really is an effect in reality. That is, the probability of a Type-II error is $\beta = 0.2$.

One-sample power figure Consider the plot below for a one-sample one-tailed greater-than t-test. If the null hypothesis, $H_0 : \mu = \mu_0$, is true, then the test statistic t is follows the null distribution indicated by the hashed area. Under a specific alternative hypothesis, $H_1 : \mu = \mu_1$, the test statistic t follows the distribution indicated by the solid area. If α is the probability of making a Type-I error (rejecting H_0 when it is true), then "crit. val." indicates the location of the t_{crit} value associated with H_0 on the scale of the data. The rejection region is the area under H_0 that is at least as far as "crit. val." is from μ_0. The power $(1-\beta)$ of the test is the green area, the area under H_1 in the rejection region. A Type-II error is made when H_1 is true, but we fail to reject H_0 in the red region. (Note, for a two-tailed test the rejection region for both tails under the H_1 curve contribute to the power.)

```r
#### One-sample power
# Power plot with two normal distributions
# http://stats.stackexchange.com/questions/14140/how-to-best-display-graphically-type-ii-bet

x <- seq(-4, 4, length=1000)
hx <- dnorm(x, mean=0, sd=1)

plot(x, hx, type="n", xlim=c(-4, 8), ylim=c(0, 0.5),
ylab = "",
xlab = "",
main= expression(paste("Type-II Error (", beta, ") and Power (", 1-beta, ")")), axes=FALSE)

#shift = qnorm(1-0.025, mean=0, sd=1)*1.7
shift = qnorm(1-0.05, mean=0, sd=1)*1.7  # one-tailed
xfit2 <- x + shift
yfit2 <- dnorm(xfit2, mean=shift, sd=1)

#axis(1, at = c(-qnorm(.025), 0, shift, -4),
#     labels = expression("p-value", 0, mu, -infinity ))
#axis(1, at = c(-qnorm(.025), 0, shift),
#     labels = expression((t[alpha/2]), mu[0], mu[1]))
axis(1, at = c(-qnorm(.05), 0, shift),
     labels = expression("crit. val.", mu[0], mu[1]))
axis(1, at = c(-4, 4+shift),
     labels = expression(-infinity, infinity ), lwd=1, lwd.tick=FALSE)


## The alternative hypothesis area
# The red - underpowered area
lb <- min(xfit2)
#ub <- round(qnorm(.975),2)
ub <- round(qnorm(.95),2)
col1 = "#CC2222"

i <- xfit2 >= lb & xfit2 <= ub
polygon(c(lb,xfit2[i],ub), c(0,yfit2[i],0), col=col1)

# The green area where the power is
col2 = "#22CC22"
i <- xfit2 >= ub
polygon(c(ub,xfit2[i],max(xfit2)), c(0,yfit2[i],0), col=col2)

# Outline the alternative hypothesis
lines(xfit2, yfit2, lwd=2)

## Print null hypothesis area
#col_null = "#DDDDDD"
#polygon(c(min(x), x,max(x)), c(0,hx,0), col=col_null)
```

```
#lines(x, hx, lwd=2)
col_null = "#AAAAAA"
polygon(c(min(x), x,max(x)), c(0,hx,0), col=col_null, lwd=2, density=c(10, 40), angle=-45, bor
lines(x, hx, lwd=2, lty="dashed", col=col_null)


axis(1, at = (c(ub, max(xfit2))), labels=c("", expression(infinity)),
    col=col2, lwd=1, lwd.tick=FALSE)

#legend("topright", inset=.05, title="Color",
#   c("Null hypoteses","Type II error", "True"), fill=c(col_null, col1, col2), horiz=FALSE)
legend("topright", inset=.015, title="Color",
   c("Null hypothesis","Type-II error", "Power"), fill=c(col_null, col1, col2),
       angle=-45,
       density=c(20, 1000, 1000), horiz=FALSE)

abline(v=ub, lwd=2, col="#000088", lty="dashed")

arrows(ub, 0.45, ub+1, 0.45, lwd=3, col="#008800")
arrows(ub, 0.45, ub-1, 0.45, lwd=3, col="#880000")
```



Example: IQ drug Imagine that we are evaluating the effect of a putative memory enhancing drug. We have randomly sampled 25 people from a population known to be normally distributed with a μ of 100 and a σ of 15. We administer the drug, wait a reasonable time for it to take effect, and then test our subjects' IQ. Assume that we were so confident in our belief that the drug would either increase IQ or have no effect that we entertained one-sided (directional) hypotheses. Our null hypothesis is that after administering the drug $\mu \leq 100$ and our alternative hypothesis is $\mu > 100$.

These hypotheses must first be converted to exact hypotheses. Converting the null is easy: it becomes $\mu = 100$. The alternative is more troublesome. If we knew that the effect of the drug was to increase IQ by 15 points, our exact alternative hypothesis would be $\mu = 115$, and we could compute power, the probability of correctly rejecting the false null hypothesis given that μ is really equal to 115 after drug treatment, not 100 (normal IQ). But if we already knew how large the effect of the drug was, we would not need to do inferential statistics...

One solution is to decide on a **minimum nontrivial effect size**. What is the smallest effect that you would consider to be nontrivial? Suppose that you decide that if the drug increases μ_{IQ} by 2 or more points, then that is a nontrivial effect, but if the mean increase is less than 2 then the effect is trivial.

Now we can test the null of $\mu = 100$ versus the alternative of $\mu = 102$. Consider the previous plot. Let the left curve represent the distribution of sample means if the null hypothesis were true, $\mu = 100$. This sampling distribution has a $\mu = 100$ and a $\sigma_{\bar{Y}} = 15/\sqrt{25} = 3$. Let the right curve represent the sampling distribution if the exact alternative hypothesis is true, $\mu = 102$. Its μ is 102 and, assuming the drug has no effect on the variance in IQ scores, also has $\sigma_{\bar{Y}} = 3$.

The green area in the upper tail of the null distribution (gray hatched curve) is α. Assume we are using a one-tailed α of 0.05. How large would a sample mean need be for us to reject the null? Since the upper 5% of a normal distribution extends from 1.645σ above the μ up to positive infinity, the sample mean

IQ would need be $100 + 1.645(3) = 104.935$ or more to reject the null. What are the chances of getting a sample mean of 104.935 or more if the alternative hypothesis is correct, if the drug increases IQ by 2 points? The area under the alternative curve from 104.935 up to positive infinity represents that probability, which is power. Assuming the alternative hypothesis is true, that $\mu = 102$, the probability of rejecting the null hypothesis is the probability of getting a sample mean of 104.935 or more in a normal distribution with $\mu = 102$, $\sigma = 3$. $Z = (104.935 - 102)/3 = 0.98$, and $P(Z > 0.98) = 0.1635$. That is, power is about 16%. If the drug really does increase IQ by an average of 2 points, we have a 16% chance of rejecting the null. If its effect is even larger, we have a greater than 16% chance.

Suppose we consider 5 (rather than 2) the minimum nontrivial effect size. This will separate the null and alternative distributions more, decreasing their overlap and increasing power. Now, $Z = (104.935 - 105)/3 = -0.02$, $P(Z > -0.02) = 0.5080$ or about 51%. **It is easier to detect large effects than small effects.**

Suppose we conduct a 2-tailed test, since the drug could actually decrease IQ; α is now split into both tails of the null distribution, 0.025 in each tail. We shall reject the null if the sample mean is 1.96 or more standard errors away from the μ of the null distribution. That is, if the mean is $100 + 1.96(3) = 105.88$ or more (or if it is $100 - 1.96(3) = 94.12$ or less) we reject the null. The probability of that happening if the alternative is correct ($\mu = 105$) is: $Z = (105.88 - 105)/3 = 0.29$, $P(Z > 0.29) = 0.3859$, and $P(Z < (94.12 - 105)/3) = P(Z < -3.63) = 0.00014$, for a total power $= (1 - \beta) = 0.3859 + 0.00014$, or about 39%. Note that our power is less than it was with a one-tailed test. **If you can correctly predict the direction of effect, a one-tailed test is more powerful than a two-tailed test.**

Consider what would happen if you increased sample size to 100. Now the $\sigma_{\bar{Y}} = 15/\sqrt{100} = 1.5$. With the null and alternative distributions are narrower, and should overlap less, increasing power. With $\sigma_{\bar{Y}} = 1.5$ the sample mean will need be $100 + (1.96)(1.5) = 102.94$ (rather than 105.88 from before)

or more to reject the null. If the drug increases IQ by 5 points, power is: $Z = (102.94 - 105)/1.5 = -1.37$, $P(Z > -1.37) = 0.9147$, or between 91 and 92%. **Anything that decreases the standard error will increase power. This may be achieved by increasing the sample size N or by reducing the σ of the dependent variable.** The σ of the dependent variable may be reduced by reducing the influence of extraneous variables upon the dependent variable (eliminating "noise" in the dependent variable makes it easier to detect the signal).

Now consider what happens if you change the significance level, α. Let us reduce α to 0.01. Now the sample mean must be 2.58 or more standard errors from the null μ before we reject the null. That is, $100 + 2.58(1.5) = 103.87$ (rather than 102.94 with $\alpha = 0.05$). Under the alternative, $Z = (103.87 - 105)/1.5 = -0.75$, $P(Z > -0.75) = 0.7734$ or about 77%, less than it was with $\alpha = 0.05$. **Reducing α reduces power.**

Please note that all of the above analyses have assumed that we have used a normally distributed test statistic, as $Z = (\bar{Y} - \mu_0)/\sigma_{\bar{Y}}$ will be if the dependent variable is normally distributed in the population or if sample size is large enough to invoke the central limit theorem (CLT). Remember that using Z also requires that you know the population σ rather than estimating it from the sample data. We more often estimate the population σ, using Student's t as the test statistic. If N is fairly large, Student's t is nearly normal, so this is no problem. For example, with a two-tailed $\alpha = 0.05$ and $N = 25$, we went out ± 1.96 standard errors to mark off the rejection region. With Student's t on $N - 1 = 24$ df we should have gone out ± 2.064 standard errors. But 1.96 versus 2.06 is a relatively trivial difference, so we should feel comfortable with the normal approximation. If, however, we had $N = 5$, $df = 4$, critical $t = \pm 2.776$, then the normal approximation would not do. A more complex analysis would be needed.

10.2 Effect size

For the one-sample test, the effect size in σ units is $d = (\mu_1 - \mu_0)/\sigma$. For our IQ problem with minimum nontrivial effect size at 5 IQ points, $d = (105 - 100)/15 = 1/3$. Cohen's[1] conventions for small, medium, and large effects for a two-sample difference test between two means is in the table below.

One- or two-sample difference of means

Size of effect	d	% variance
small	0.2	1
medium	0.5	6
large	0.8	16

Cohen has conventions for other tests (correlation, contingency tables, etc.), but they should be used with caution.

What is a small or even trivial effect in one context may be a large effect in another context. For example, Rosnow and Rosenthal (1989) discussed a 1988 biomedical research study on the effects of taking a small, daily dose of aspirin. Each participant was instructed to take one pill a day. For about half of the participants the pill was aspirin, for the others it was a placebo. The dependent variable was whether or not the participant had a heart attack during the study. In terms of a correlation coefficient, the size of the observed effect was $r = 0.034$. In terms of percentage of variance explained, that is 0.12%. In other contexts this might be considered a trivial effect, but it this context it was so large an effect that the researchers decided it was unethical to continue the study and the contacted all of the participants who were taking the placebo and told them to start taking aspirin every day.

10.3 Sample size

Before you can answer the question "how many subjects do I need,"' you will have to answer several other questions, such as:

[1]Cohen, J. (1988). Statistical power analysis for the behavior sciences. (2nd ed.). Hillsdale, NJ: Erlbaum.

- How much power do I want?
- What is the likely size (in the population) of the effect I am trying to detect, or, what is smallest effect size that I would consider of importance?
- What criterion of statistical significance will I employ?
- What test statistic will I employ?
- What is the standard deviation (in the population) of the criterion variable?
- For correlated samples designs, what is the correlation (in the population) between groups?

If one considers Type I and Type II errors equally serious, then one should have enough power to make $\alpha = \beta$. If employing the traditional 0.05 criterion of statistical significance, that would mean you should have 95% power. However, getting 95% power usually involves expenses too great – that is, too many samples.

A common convention is to try to get at least enough data to have 80% power. So, how do you figure out how many subjects you need to have the desired amount of power. There are several methods, including:

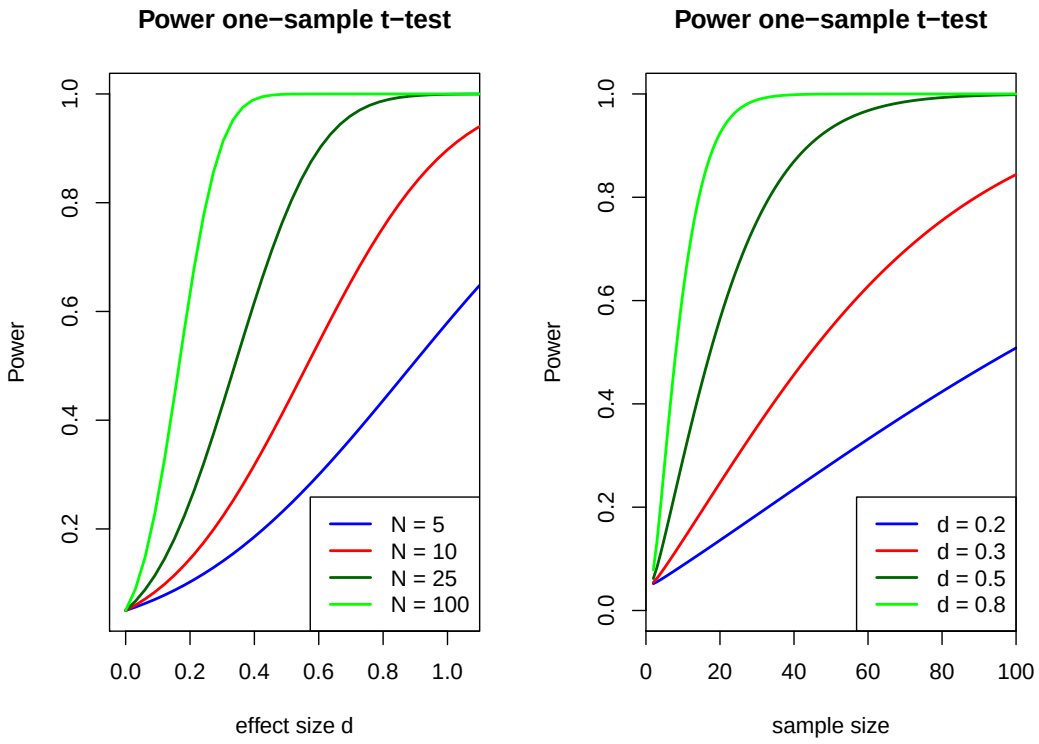
- You could buy an expensive, professional-quality software package to do the power analysis.
- You could buy an expensive, professional-quality book on power analysis and learn to do the calculations yourself and/or to use power tables and figures to estimate power.
- You could try to find an interactive web page on the Internet that will do the power analysis for you. This is probably fine, but be cautious.
- You could download and use the G Power program, which is free, not too difficult to use, and generally reliable (this is not to say that it is error free).
- You could use the simple guidelines provided in Jacob Cohen's "A Power Primer" (Psychological Bulletin, 1992, 112, 155-159).

The plots below indicate the amount of power for a given effect size and sample size for a one-sample t-test and ANOVA test. This graph makes clear

the diminishing returns you get for adding more and more subjects if you already have moderate to high power. For example, let's say we're doing a one-sample test and we an effect size of 0.2 and have only 10 subjects. We can see that we have a power of about 0.15, which is really, really low. Going to 25 subjects increases our power to about 0.25, and to 100 subjects increases our power to about 0.6. But if we had a large effect size of 0.8, 10 subjects would already give us a power of about 0.8, and using 25 or 100 subjects would both give a power at least 0.98. So each additional subject gives you less additional power. This curve also illustrates the "cost" of increasing your desired power from 0.8 to 0.98.

```
# Power curve plot for one-sample t-test with range of sample sizes
# http://stackoverflow.com/questions/4680163/power-vs-effect-size-plot/4680786#4680786

P     <- 3                              # number of groups for ANOVA
fVals <- seq(0, 1.2, length.out=100)    # effect sizes f for ANOVA
dVals <- seq(0, 3, length.out=100)      # effect sizes d for t-Test
#nn    <- seq(10, 25, by=5)              # group sizes
nn    <- c(5,10,25,100)                 # group sizes
alpha <- 0.05                           # test for level alpha

# function to calculate one-way ANOVA power for given group size
getFPow <- function(n) {
    critF <- qf(1-alpha, P-1, P*n - P) # critical F-value

    # probabilities of exceeding this F-value given the effect sizes f
    # P*n*fVals^2 is the non-centrality parameter
    1-pf(critF, P-1, P*n - P, P*n * fVals^2)
}

# function to calculate one-sample t-Test power for given group size
getTPow <- function(n) {
    critT <- qt(1-alpha, n-1)           # critical t-value

    # probabilities of exceeding this t-value given the effect sizes d
    # sqrt(n)*d is the non-centrality parameter
    1-pt(critT, n-1, sqrt(n)*dVals)
}

powsF <- sapply(nn, getFPow)      # ANOVA power  for for all group sizes
powsT <- sapply(nn, getTPow)      # t-Test power for for all group sizes

#dev.new(width=10, fig.height=5)
par(mfrow=c(1, 2))
```

```
matplot(dVals, powsT, type="l", lty=1, lwd=2, xlab="effect size d",
        ylab="Power", main="Power one-sample t-test", xaxs="i",
        xlim=c(-0.05, 1.1), col=c("blue", "red", "darkgreen", "green"))
#legend(x="bottomright", legend=paste("N =", c(5,10,25,100)), lwd=2,
#       col=c("blue", "red", "darkgreen", "green"))
legend(x="bottomright", legend=paste("N =", nn), lwd=2,
       col=c("blue", "red", "darkgreen", "green"))
#matplot(fVals, powsF, type="l", lty=1, lwd=2, xlab="effect size f",
#        ylab="Power", main=paste("Power one-way ANOVA, ", P, " groups", sep=""), xaxs="i",
#        xlim=c(-0.05, 1.1), col=c("blue", "red", "darkgreen", "green"))
##legend(x="bottomright", legend=paste("Nj =", c(10, 15, 20, 25)), lwd=2,
##       col=c("blue", "red", "darkgreen", "green"))
#legend(x="bottomright", legend=paste("Nj =", nn), lwd=2,
#       col=c("blue", "red", "darkgreen", "green"))

library(pwr)
pwrt2 <- pwr.t.test(d=.2,n=seq(2,100,1),
          sig.level=.05,type="one.sample", alternative="two.sided")
pwrt3 <- pwr.t.test(d=.3,n=seq(2,100,1),
          sig.level=.05,type="one.sample", alternative="two.sided")
pwrt5 <- pwr.t.test(d=.5,n=seq(2,100,1),
          sig.level=.05,type="one.sample", alternative="two.sided")
pwrt8 <- pwr.t.test(d=.8,n=seq(2,100,1),
          sig.level=.05,type="one.sample", alternative="two.sided")
#plot(pwrt£n, pwrt£power, type="b", xlab="sample size", ylab="power")
matplot(matrix(c(pwrt2$n,pwrt3$n,pwrt5$n,pwrt8$n),ncol=4),
        matrix(c(pwrt2$power,pwrt3$power,pwrt5$power,pwrt8$power),ncol=4),
        type="l", lty=1, lwd=2, xlab="sample size",
        ylab="Power", main="Power one-sample t-test", xaxs="i",
        xlim=c(0, 100), ylim=c(0,1), col=c("blue", "red", "darkgreen", "green"))
legend(x="bottomright", legend=paste("d =", c(0.2, 0.3, 0.5, 0.8)), lwd=2,
       col=c("blue", "red", "darkgreen", "green"))
```



Reasons to do a power analysis There are several of reasons why one might do a power analysis. (1) Perhaps the most common use is to determine the necessary number of subjects needed to detect an effect of a given size. Note that trying to find the absolute, bare minimum number of subjects needed in the study is often not a good idea. (2) Additionally, power analysis can be used to determine power, given an effect size and the number of subjects available. You might do this when you know, for example, that only 75 subjects are available (or that you only have the budget for 75 subjects), and you want to know if you will have enough power to justify actually doing the study. In most cases, there is really no point to conducting a study that is seriously underpowered. Besides the issue of the number of necessary subjects, there are other good reasons for doing a power analysis. (3) For example, a power analysis is often required as part of a grant proposal. (4) And finally, doing a power analysis is often just part of doing good research. A power analysis is a good way of making sure that you have thought through every aspect of the study and the statistical analysis before you start collecting data.

Limitations Despite these advantages of power analyses, there are some limitations. (1) One limitation is that power analyses do not typically generalize very well. If you change the methodology used to collect the data or change the statistical procedure used to analyze the data, you will most likely have to redo the power analysis. (2) In some cases, a power analysis might suggest a number of subjects that is inadequate for the statistical procedure. For example (beyond the scope of this class), a power analysis might suggest that you need 30 subjects for your logistic regression, but logistic regression, like all maximum likelihood procedures, require much larger sample sizes. (3) Perhaps the most important limitation is that a standard power analysis gives you a "best case scenario" estimate of the necessary number of subjects needed to detect the effect. In most cases, this "best case scenario" is based on assumptions and educated guesses. If any of these assumptions or guesses are incorrect, you may have less power than you need to detect the effect. (4) Finally, because power analyses are based on assumptions and educated guesses, you often get a range of the number of subjects needed, not a precise number. For example, if you do not know what the standard deviation of your outcome measure will be, you guess at this value, run the power analysis and get X number of subjects. Then you guess a slightly larger value, rerun the power analysis and get a slightly larger number of necessary subjects. You repeat this process over the plausible range of values of the standard deviation, which gives you a range of the number of subjects that you will need.

Other considerations After all of this discussion of power analyses and the necessary number of subjects, we need to stress that power is not the only consideration when determining the necessary sample size. For example, different researchers might have different reasons for conducting a regression analysis. (1) One might want to see if the regression coefficient is different from zero, (2) while the other wants to get a very precise estimate of the regression coefficient with a very small confidence interval around it. This second purpose requires a larger sample size than does merely seeing if the regression coefficient is different

from zero. (3) Another consideration when determining the necessary sample size is the assumptions of the statistical procedure that is going to be used (e.g., parametric vs nonparametric procedure). (4) The number of statistical tests that you intend to conduct will also influence your necessary sample size: the more tests that you want to run, the more subjects that you will need (multiple comparisons). (5) You will also want to consider the representativeness of the sample, which, of course, influences the generalizability of the results. Unless you have a really sophisticated sampling plan, the greater the desired generalizability, the larger the necessary sample size.

10.4 Power calculation via simulation

Using the principles of the bootstrap (to be covered later) we can estimate statistical power through simulation.

Example: IQ drug, revisited Recall that we sample $N = 25$ people from a population known to be normally distributed with a μ of 100 and a σ of 15. Consider the first one-sided alternative $H_0 : \mu = 100$ and $H_1 : \mu > 100$. Assume the *minimum nontrivial effect size* was that the drug increases μ_{IQ} by 2 or more points, so that the specific alternative to consider is $H_1 : \mu = 102$. What is the power of this test?

We already saw how to calculate this analytically. To solve this computationally, we need to simulate samples of $N = 25$ from the alternative distribution ($\mu = 102$ and $\sigma = 15$) and see what proportion of the time we correctly reject H_0.

```
#### Example: IQ drug, revisited
# R code to simulate one-sample one-sided power

# Strategy:
  # Do this R times:
  #   draw a sample of size N from the distribution specified by the alternative hypothesis
  #     That is, 25 subjects from a normal distribution with mean 102 and sigma 15
  #   Calculate the mean of our sample
  #   Calculate the associated z-statistic
```

```
  #   See whether that z-statistic has a p-value < 0.05 under H0: mu=100
  #      If we reject H0, then set reject = 1, else reject = 0.
  # Finally, the proportion of rejects we observe is the approximate power

n     <-  25;              # sample size of 25
mu0   <- 100;              # null hypothesis mean of 100
mu1   <- 102;              # alternative mean of 102
#mu1   <- 105;               # alternative mean of 105
sigma <-  15;              # standard deviation of normal population

alpha <- 0.05;             # significance level

R     <- 10000;            # Repetitions to draw sample and see whether we reject H0
                           #   The proportion of these that reject H0 is the power

reject <- rep(NA, R);      # allocate a vector of length R with missing values (NA)
                           #   to fill with 0 (fail to reject H0) or 1 (reject H0)
for (i in 1:R) {
  sam <- rnorm(n, mean=mu1, sd=sigma);  # sam is a vector with 25 values

  ybar <- mean(sam);  # Calculate the mean of our sample sam

  z <- (ybar - mu0) / (sigma / sqrt(n));  # z-statistic (assumes we know sigma)
                           # we could also have calculated the t-statistic, here

  pval <- 1-pnorm(z);   # one-sided right-tail p-value
                        #   pnorm gives the area to the left of z
                        #   therefore, the right-tail area is 1-pnorm(z)

  if (pval < 0.05) {
    reject[i] <- 1;  # 1 for correctly rejecting H0
  } else {
    reject[i] <- 0;  # 0 for incorrectly fail to reject H0
  }

}

power <- mean(reject);  # the average reject (proportion of rejects) is the power
power
```

```
## [1] 0.166
```

```
# 0.1655  for mu1=102
# 0.5082  for mu1=105
```

Our simulation (this time) with $\mu_1 = 102$ gave a power of 0.166 (exact answer is $P(Z > 0.98) = 0.1635$). Rerunning with $\mu_1 = 105$ gave a power of 0.5082 (exact answer is $P(Z > -0.02) = 0.5080$). Our simulation well-

approximates the true value, and the power can be made more precise by increasing the number of repetitions calculated. However, two to three decimal precision is quite sufficient.

Example: Head breadth Recall the head breadth example in Chapter 3 comparing maximum head breadths (in millimeters) of modern day Englishmen with ancient Celts. The data are summarized below.

```
Descriptive Statistics: ENGLISH, CELTS
Variable   N    Mean  SE Mean  StDev  Minimum      Q1  Median      Q3  Maximum
ENGLISH   18  146.50     1.50   6.38   132.00  141.75  147.50  150.00   158.00
CELTS     16  130.75     1.36   5.43   120.00  126.25  131.50  135.50   138.00
```

Imagine that we don't have the information above. Imagine we have been invited to a UK university to take skull measurements for 18 modern day Englishmen, and 16 ancient Celts. We have some information about modern day skulls to use as prior information for measurement mean and standard deviation. What is the power to observe a difference between the populations? Let's make some reasonable assumptions that allows us to be a bit conservative. Let's assume the sampled skulls from each of our populations is a random sample with common standard deviation 7mm, and let's assume we can't get the full sample but can only measure 15 skulls from each population. At a significance level of $\alpha = 0.05$, what is the power for detecting a difference of 5, 10, 15, 20, or 25 mm?

The theoretical two-sample power result is not too hard to derive (and is available in text books), but let's simply compare the power calculated exactly and by simulation.

For the exact result we use R library **pwr**. Below is the function call as well as the result. Note that we specified multiple effect sizes (diff/SD) in one call of the function.

```
# R code to compute exact two-sample two-sided power
library(pwr) # load the power calculation library

pwr.t.test(n = 15,
    d = c(5,10,15,20,25)/7,
    sig.level = 0.05,
    power = NULL,
    type = "two.sample",
    alternative = "two.sided")
```

```
##
##      Two-sample t test power calculation
##
##               n = 15
##               d = 0.7142857, 1.4285714, 2.1428571, 2.8571429, 3.5714286
##       sig.level = 0.05
##           power = 0.4717438, 0.9652339, 0.9998914, 1.0000000, 1.0000000
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

To simulate the power under the same circumstances, we follow a similar strategy as in the one-sample example.

```
# R code to simulate two-sample two-sided power

# Strategy:
  # Do this R times:
  #   draw a sample of size N from the two hypothesized distributions
  #     That is, 15 subjects from a normal distribution with specified means and sigma=7
  #   Calculate the mean of the two samples
  #   Calculate the associated z-statistic
  #   See whether that z-statistic has a p-value < 0.05 under H0: mu_diff=0
  #      If we reject H0, then set reject = 1, else reject = 0.
  # Finally, the proportion of rejects we observe is the approximate power

n     <-  15;             # sample size of 25
mu1   <- 147;             # null hypothesis English mean
mu2   <- c(142, 137, 132, 127, 122);  # Celt means
sigma <-   7;             # standard deviation of normal population

alpha <- 0.05;            # significance level

R     <- 2e4;             # Repetitions to draw sample and see whether we reject H0
                          #   The proportion of these that reject H0 is the power

power <- rep(NA,length(mu2));  # allocate a vector to store the calculated power in

for (j in 1:length(mu2)) {  # do for each value of mu2

  reject <- rep(NA, R);     # allocate a vector of length R with missing values (NA)
                            #   to fill with 0 (fail to reject H0) or 1 (reject H0)

  for (i in 1:R) {
    sam1 <- rnorm(n, mean=mu1   , sd=sigma);  # English sample
    sam2 <- rnorm(n, mean=mu2[j], sd=sigma);  # Celt sample

    ybar1 <- mean(sam1);  # Calculate the mean of our sample sam
```

```
    ybar2 <- mean(sam2);  # Calculate the mean of our sample sam

    # z-statistic (assumes we know sigma)
    # we could also have calculated the t-statistic, here
    z <- (ybar2 - ybar1) / (sigma * sqrt(1/n+1/n));

    pval.Left  <- pnorm(z);    # area under left tail
    pval.Right <- 1-pnorm(z);  # area under right tail
    # p-value is twice the smaller tail area
    pval <- 2 * min(pval.Left, pval.Right);

    if (pval < 0.05) {
      reject[i] <- 1;  # 1 for correctly rejecting H0
    } else {
      reject[i] <- 0;  # 0 for incorrectly fail to reject H0
    }

  }

  # the average reject (proportion of rejects) is the power
  power[j] <- mean(reject);
}

power
## [1] 0.49275 0.97650 1.00000 1.00000 1.00000
```

Note the similarity between power calculated using both the exact and simulation methods. If there is a power calculator for your specific problem, it is best to use that because it is faster and there is no programming. However, using the simulation method is better if we wanted to entertain different sample sizes with different standard deviations, etc. There may not be a standard calculator for our specific problem, so knowing how to simulate the power can be valuable.

Mean				Sample size		Power	
μ_E	μ_C	diff	SD	n_E	n_C	exact	simulated
147	142	5	7	15	15	0.4717	0.4928
147	137	10	7	15	15	0.9652	0.9765
147	132	15	7	15	15	0.9999	1
147	127	20	7	15	15	1.0000	1
147	122	25	7	15	15	1.0000	1

Chapter 11

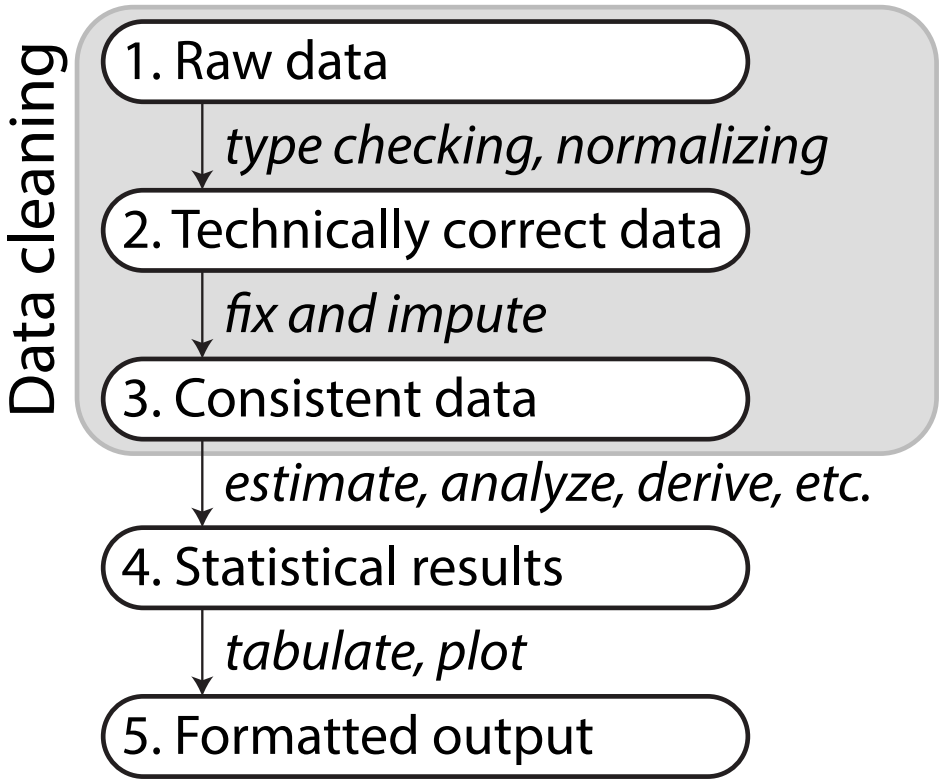
Data Cleaning

Contents

Data cleaning[1], or data preparation, is an essential part of statistical analysis. In fact, in practice it is often more time-consuming than the statistical analysis itself. Data cleaning may profoundly influence the statistical statements based on the data. Typical actions like imputation or outlier handling obviously influence the results of a statistical analyses. For this reason, data cleaning should be considered a statistical operation, to be performed in a reproducible manner. The R statistical environment provides a good environment for reproducible data cleaning since all cleaning actions can be scripted and therefore reproduced.

11.1 The five steps of statistical analysis

Statistical analysis can be viewed as the result of a number of value-increasing data processing steps.



[1]Content in this chapter is derived with permission from Statistics Netherlands at `http://cran.r-project.org/doc/contrib/de_Jonge+van_der_Loo-Introduction_to_data_cleaning_with_R.pdf`

Each box represents data in a certain state while each arrow represents the activities needed to get from one state to the other.

1. Raw Data The data "as is" may lack headers, contain wrong data types (e.g., numbers stored as strings), wrong category labels, unknown or unexpected character encoding and so on. Reading such files into an R `data.frame` directly is either difficult or impossible without some sort of preprocessing.

2. Technically correct data The data can be read into an R `data.frame`, with correct names, types and labels, without further trouble. However, that does not mean that the values are error-free or complete.
For example, an age variable may be reported negative, an under-aged person may be registered to possess a driver's license, or data may simply be missing. Such inconsistencies obviously depend on the subject matter that the data pertains to, and they should be ironed out before valid statistical inference from such data can be produced.

3. Consistent data The data is ready for statistical inference. It is the data that most statistical theories use as a starting point. Ideally, such theories can still be applied without taking previous data cleaning steps into account. In practice however, data cleaning methods like imputation of missing values will influence statistical results and so must be accounted for in the following analyses or interpretation thereof.

4. Statistical results The results of the analysis have been produced and can be stored for reuse.

5. Formatted output The results in tables and figures ready to include in statistical reports or publications.

Best practice Store the input data for each stage (raw, technically correct, consistent, results, and formatted) separately for reuse. Each step between the stages may be performed by a separate R script for reproducibility.

11.2 R background review

11.2.1 Variable types

The most basic variable in R is a vector. An R vector is a sequence of values of the same type. All basic operations in R act on vectors (think of the element-wise arithmetic, for example). The basic types in R are as follows.

numeric Numeric data (approximations of the real numbers)
integer Integer data (whole numbers)
factor Categorical data (simple classifications, like gender)
ordered Ordinal data (ordered classifications, like educational level)
character Character data (strings)
raw Binary data (rarely used)

All basic operations in R work element-wise on vectors where the shortest argument is recycled if necessary. Why does the following code work the way it does?

```
# vectors have variables of _one_ type
c(1, 2, "three")
## [1] "1"     "2"     "three"
# shorter arguments are recycled
(1:3) * 2
## [1] 2 4 6
(1:4) * c(1, 2)
## [1] 1 4 3 8
# warning! (why?)
(1:4) * (1:3)
## Warning in (1:4) * (1:3): longer object length is not a multiple of shorter object length
## [1] 1 4 9 4
```

11.2.2 Special values and value-checking functions

Below are the definitions and some illustrations of the special values `NA`, `NULL`, $\pm$`Inf`, and `NaN`.

- `NA` Stands for "not available". `NA` is a placeholder for a missing value. All basic operations in R handle NA without crashing and mostly return

NA as an answer whenever one of the input arguments is NA. If you understand NA, you should be able to predict the result of the following R statements.

```
NA + 1
sum(c(NA, 1, 2))
median(c(NA, 1, 2, 3), na.rm = TRUE)
length(c(NA, 2, 3, 4))
3 == NA
NA == NA
TRUE | NA
# use is.na() to detect NAs
is.na(c(1, NA, 3))
```

- `NULL` Think of `NULL` as the empty set from mathematics; it has no class (its class is `NULL`) and has length 0 so it does not take up any space in a vector.

```
length(c(1, 2, NULL, 4))
sum(c(1, 2, NULL, 4))
x <- NULL
length(x)
c(x, 2)
# use is.null() to detect NULL variables
is.null(x)
```

- `Inf` Stands for "infinity" and only applies to vectors of class numeric (not integer). Technically, `Inf` is a valid numeric that results from calculations like division of a number by zero. Since `Inf` is a numeric, operations between `Inf` and a finite numeric are well-defined and comparison operators work as expected.

```
pi/0
2 * Inf
Inf - 1e+10
Inf + Inf
3 < -Inf
Inf == Inf
# use is.infinite() to detect Inf variables
is.infinite(-Inf)
```

- `NaN` Stands for "not a number". This is generally the result of a calculation of which the result is unknown, but it is surely not a number. In particular operations like $0/0$, `Inf - Inf` and `Inf/Inf` result in `NaN`. Technically, `NaN` is of class numeric, which may seem odd since it is used to indicate that something is not numeric. Computations involving numbers and `NaN` always result in `NaN`.

```
NaN + 1
exp(NaN)
```

```
# use is.nan() to detect NULL variables
is.nan(0/0)
```

Note that `is.finite()` checks a numeric vector for the occurrence of any non-numerical or special values.

```
is.finite(c(1, NA, 2, Inf, 3, -Inf, 4, NULL, 5, NaN, 6))
##  [1]  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE
```

11.3 From raw to technically correct data

11.3.1 Technically correct data

Limiting ourselves to "rectangular" data sets read from a text-based format, **technically correct** data in R

1. is stored in a `data.frame` with suitable columns names, and
2. each column of the `data.frame` is of the R type that adequately represents the value domain.

The second demand implies that numeric data should be stored as `numeric` or `integer`, textual data should be stored as `character` and categorical data should be stored as a `factor` or `ordered` vector, with the appropriate levels.

Best practice Whenever you need to read data from a foreign file format, like a spreadsheet or proprietary statistical software that uses undisclosed file formats, make that software responsible for exporting the data to an open format that can be read by R.

11.3.2 Reading text data into an R data.frame

In the following, we assume that the text-files we are reading contain data of at most one unit per line. The number of attributes, their format and separation symbols in lines containing data may differ over the lines. This includes files in fixed-width or csv-like format, but excludes XML-like storage formats.

Reading text

`read.table()` and similar functions below will read a text file and return a `data.frame`.

Best practice. A freshly read `data.frame` should always be inspected with functions like `head()`, `str()`, and `summary()`.

The `read.table()` function is the most flexible function to read tabular data that is stored in a textual format. The other read-functions below all eventually use `read.table()` with some fixed parameters and possibly after some preprocessing. Specifically

- `read.csv()` for comma separated values with period as decimal separator.
- `read.csv2()` for semicolon separated values with comma as decimal separator.
- `read.delim()` tab-delimited files with period as decimal separator.
- `read.delim2()` tab-delimited files with comma as decimal separator.
- `read.fwf()` data with a predetermined number of bytes per column.

Additional optional arguments include:

Argument	Description
`header`	Does the first line contain column names?
`col.names`	`character` vector with column names.
`na.string`	Which strings should be considered `NA`?
`colClasses`	`character` vector with the types of columns. Will coerce the columns to the specified types.
`stringsAsFactors`	If `TRUE`, converts all `character` vectors into `factor` vectors.
`sep`	Field separator.

Except for `read.table()` and `read.fwf()`, each of the above functions assumes by default that the first line in the text file contains column headers. The following demonstrates this on the following text file.

```
21,6.0
42,5.9
18,5.7*
21,NA
```

Read the file with defaults, then specifying necessary options.

```
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch18_unnamed.txt"
# first line is erroneously interpreted as column names
person <- read.csv(fn.data)
person
```

```
##   X21 X6.0
## 1  42  5.9
## 2  18 5.7*
## 3  21 <NA>
```

```
# instead, use header = FALSE and specify the column names
person <- read.csv(file = fn.data
                   , header = FALSE
                   , col.names = c("age", "height")
                    )
person
```

```
##   age height
## 1  21    6.0
## 2  42    5.9
## 3  18   5.7*
## 4  21   <NA>
```

If `colClasses` is not specified by the user, `read.table()` will try to determine the column types. Although this may seem convenient, it is noticeably slower for larger files (say, larger than a few MiB) and it may yield unexpected results. For example, in the above script, one of the rows contains a malformed numerical variable (`5.7*`), causing R to interpret the whole column as a text variable. Moreover, by default text variables are converted to factor, so we are now stuck with a height variable expressed as levels in a categorical variable:

```
str(person)
```

```
## 'data.frame': 4 obs. of  2 variables:
##  $ age   : int  21 42 18 21
##  $ height: Factor w/ 3 levels "5.7*","5.9","6.0": 3 2 1 NA
```

As an alternative, columns can be read in as character by setting `stringsAsFactors`. Next, one of the `as.`-functions can be applied to convert to the desired type, as shown below.

```
person <- read.csv(file = fn.data
                   , header = FALSE
                   , col.names = c("age", "height")
                   , stringsAsFactors = FALSE)
person
```

```
##   age height
## 1  21    6.0
## 2  42    5.9
## 3  18   5.7*
```

```
## 4  21   <NA>
person$height <- as.numeric(person$height)
## Warning:  NAs introduced by coercion
person
##   age height
## 1  21    6.0
## 2  42    5.9
## 3  18     NA
## 4  21     NA
```

Now, everything is read in and the `height` column is translated to `numeric`, with the exception of the row containing `5.7*`. Moreover, since we now get a warning instead of an error, a script containing this statement will continue to run, albeit with less data to analyse than it was supposed to. It is of course up to the programmer to check for these extra `NA`'s and handle them appropriately.

Reading data with `readLines`

When the rows in a data file are not uniformly formatted you can consider reading in the text line-by-line and transforming the data to a rectangular set yourself. With `readLines()` you can exercise precise control over how each line is interpreted and transformed into fields in a rectangular data set. We use the following data as an example.

```
%% Data on the Dalton Brothers
Gratt ,1861,1892
Bob,1892
1871,Emmet ,1937
% Names, birth and death dates
```

And this is the table we want.

Name	Birth	Death
Gratt	1861	1892
Bob	NA	1892
Emmet	1871	1937

The file has comments on several lines (starting with a % sign) and a missing value in the second row. Moreover, in the third row the name and birth date have been swapped. We want a general strategy so that if we had a file with 10,000 records we could process them all. The table suggests one strategy.

	Step	result
1	Read the data with readLines	`character`
2	Select lines containing data	`character`
3	Split lines into separate fields	list of character vectors
4	Standardize rows	list of equivalent vectors
5	Transform to `data.frame`	`data.frame`
6	Normalize and coerce to correct type	`data.frame`

Step 1. Reading data. The `readLines()` function accepts filename as argument and returns a character vector containing one element for each line in the file. `readLines()` detects both the end-of-line and carriage return characters so lines are detected regardless of whether the file was created under DOS, UNIX, or MAC (each OS has traditionally had different ways of marking an end-of-line). Reading in the Daltons file yields the following.

```
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch18_dalton.txt"
dalton.txt <- readLines(fn.data)
dalton.txt

## [1] "%% Data on the Dalton Brothers" "Gratt ,1861,1892"
## [3] "Bob,1892"                       "1871,Emmet ,1937"
## [5] "% Names, birth and death dates"

str(dalton.txt)

##  chr [1:5] "%% Data on the Dalton Brothers" "Gratt ,1861,1892" ...
```

The variable `dalton.txt` has 5 character elements, equal to the number of lines in the textfile.

Step 2. Selecting lines containing data. This is generally done by throwing out lines containing comments or otherwise lines that do not contain any data fields. You can use `grep()` or `grepl()` to detect such lines. Regular expressions[2], though challenging to learn, can be used to specify what you're searching for. I usually search for an example and modify it to meet my needs.

```
# detect lines starting (^) with a percentage sign (%)
ind.nodata <- grepl("^%", dalton.txt)
ind.nodata

## [1]  TRUE FALSE FALSE FALSE  TRUE

# and throw them out
```

[2] `http://en.wikipedia.org/wiki/Regular_expression`

```
!ind.nodata

## [1] FALSE  TRUE  TRUE  TRUE FALSE

dalton.dat <- dalton.txt[!ind.nodata]
dalton.dat

## [1] "Gratt ,1861,1892" "Bob,1892"         "1871,Emmet ,1937"
```

Here, the first argument of `grepl()` is a search pattern, where the caret (`^`) indicates a start-of-line. The result of `grepl()` is a logical vector that indicates which elements of `dalton.txt` contain the pattern 'start-of-line' followed by a percent-sign. The functionality of `grep()` and `grepl()` will be discussed in more detail later.

Step 3. Split lines into separate fields. This can be done with `strsplit()`. This function accepts a character vector and a split argument which tells `strsplit()` how to split a string into substrings. The result is a list of character vectors.

```
# remove whitespace by substituting nothing where spaces appear
dalton.dat2 <- gsub(" ", "", dalton.dat)
# split strings by comma
dalton.fieldList <- strsplit(dalton.dat2, split = ",")
dalton.fieldList

## [[1]]
## [1] "Gratt" "1861"  "1892"
##
## [[2]]
## [1] "Bob"  "1892"
##
## [[3]]
## [1] "1871"  "Emmet" "1937"
```

Here, `split=` is a single character or sequence of characters that are to be interpreted as field separators. By default, `split` is interpreted as a regular expression, and the meaning of a special characters can be ignored by passing `fixed=TRUE` as extra parameter.

Step 4. Standardize rows. The goal of this step is to make sure that (a) every row has the same number of fields and (b) the fields are in the right order. In `read.table()`, lines that contain fewer fields than the maximum number of fields detected are appended with `NA`. One advantage of the do-it-yourself approach shown here is that we do not have to make this assumption.

The easiest way to standardize rows is to write a function that takes a single character vector as input and assigns the values in the right order.
The function below accepts a character vector and assigns three values to an output vector of class `character`. The `grepl()` statement detects fields containing alphabetical values `a-z` or `A-Z`. To assign year of birth and year of death, we use the knowledge that all Dalton brothers were born before and died after 1890. To retrieve the fields for each row in the example, we need to apply this function to every element of `dalton.fieldList`.

```
# function to correct column order for Dalton data
f.assignFields <- function(x) {
  # create a blank character vector of length 3
  out <- character(3)
  # get name and put into first position
  ind.alpha <- grepl("[[:alpha:]]", x)
  out[1] <- x[ind.alpha]
  # get birth date (if any) and put into second position
  ind.num.birth <- which(as.numeric(x) < 1890)
    # if there are more than 0 years <1890,
    #   then return that value to second position,
    #   else return NA to second position
  out[2] <- ifelse(length(ind.num.birth) > 0, x[ind.num.birth], NA)
  # get death date (if any) and put into third position (same strategy as birth)
  ind.num.death <- which(as.numeric(x) > 1890)
  out[3] <- ifelse(length(ind.num.death) > 0, x[ind.num.death], NA)
  out
}
```

The function `lapply()` will apply the function `f.assignFields()` to each list element in `dalton.fieldList`.

```
dalton.standardFields <- lapply(dalton.fieldList, f.assignFields)
```

```
## Warning in which(as.numeric(x) < 1890): NAs introduced by coercion
## Warning in which(as.numeric(x) > 1890): NAs introduced by coercion
## Warning in which(as.numeric(x) < 1890): NAs introduced by coercion
## Warning in which(as.numeric(x) > 1890): NAs introduced by coercion
## Warning in which(as.numeric(x) < 1890): NAs introduced by coercion
## Warning in which(as.numeric(x) > 1890): NAs introduced by coercion
```

```
dalton.standardFields
```

```
## [[1]]
## [1] "Gratt" "1861"  "1892"
##
## [[2]]
## [1] "Bob"  NA     "1892"
##
## [[3]]
## [1] "Emmet" "1871"  "1937"
```

The advantage of this approach is having greater flexibility than `read.table` offers. However, since we are interpreting the value of fields here, it is unavoidable to know about the contents of the dataset which makes it hard to generalize the field assigner function. Furthermore, `f.assignFields()` function we wrote is still relatively fragile. That is, it crashes for example when the input vector contains two or more text-fields or when it contains more than one numeric value larger than 1890. Again, no one but the data analyst is probably in a better position to choose how safe and general the field assigner should be.

Step 5. Transform to data.frame. There are several ways to transform a list to a `data.frame` object. Here, first all elements are copied into a matrix which is then coerced into a `data.frame`.

```
# unlist() returns each value in a list in a single object
unlist(dalton.standardFields)
```

```
## [1] "Gratt" "1861"  "1892"  "Bob"   NA      "1892"  "Emmet" "1871"
## [9] "1937"
```

```
# there are three list elements in dalton.standardFields
length(dalton.standardFields)
```

```
## [1] 3
```

```
# fill a matrix will the character values
dalton.mat <- matrix(unlist(dalton.standardFields)
                     , nrow = length(dalton.standardFields)
                     , byrow = TRUE
                      )
dalton.mat
```

```
##      [,1]    [,2]   [,3]
## [1,] "Gratt" "1861" "1892"
## [2,] "Bob"   NA     "1892"
## [3,] "Emmet" "1871" "1937"
```

```
# name the columns
colnames(dalton.mat) <- c("name", "birth", "death")
dalton.mat
```

```
##      name    birth  death
## [1,] "Gratt" "1861" "1892"
## [2,] "Bob"   NA     "1892"
## [3,] "Emmet" "1871" "1937"
```

```
# convert to a data.frame but don't turn character variables into factors
dalton.df <- as.data.frame(dalton.mat, stringsAsFactors=FALSE)
str(dalton.df)
```

```
## 'data.frame': 3 obs. of  3 variables:
##  $ name : chr  "Gratt" "Bob" "Emmet"
##  $ birth: chr  "1861" NA "1871"
##  $ death: chr  "1892" "1892" "1937"

dalton.df

##    name birth death
## 1 Gratt  1861  1892
## 2   Bob  <NA>  1892
## 3 Emmet  1871  1937
```

The function `unlist()` concatenates all vectors in a list into one large character vector. We then use that vector to fill a matrix of class character. However, the matrix function usually fills up a matrix column by column. Here, our data is stored with rows concatenated, so we need to add the argument `byrow=TRUE`. Finally, we add column names and coerce the matrix to a `data.frame`. We use `stringsAsFactors=FALSE` since we have not started interpreting the values yet.

Step 6. Normalize and coerce to correct types. This step consists of preparing the character columns of our `data.frame` for coercion and translating numbers into numeric vectors and possibly character vectors to factor variables. String normalization and type conversion are discussed later. In this example we can suffice with the following statements.

```
dalton.df$birth <- as.numeric(dalton.df$birth)
dalton.df$death <- as.numeric(dalton.df$death)
str(dalton.df)

## 'data.frame': 3 obs. of  3 variables:
##  $ name : chr  "Gratt" "Bob" "Emmet"
##  $ birth: num  1861 NA 1871
##  $ death: num  1892 1892 1937

dalton.df

##    name birth death
## 1 Gratt  1861  1892
## 2   Bob    NA  1892
## 3 Emmet  1871  1937
```

11.4 Type conversion

Converting a variable from one type to another is called coercion. The reader is probably familiar with R's basic coercion functions, but as a reference they are listed here.

```
as.numeric
as.integer
as.character
as.logical
as.factor
as.ordered
```

Each of these functions takes an R object and tries to convert it to the class specified behind the "`as.`". By default, values that cannot be converted to the specified type will be converted to a `NA` value while a warning is issued.

```
as.numeric(c("7", "7*", "7.0", "7,0"))
## Warning:  NAs introduced by coercion
## [1]  7 NA  7 NA
```

In the remainder of this section we introduce R's typing and storage system and explain the difference between R types and classes. After that we discuss date conversion.

11.4.1 Introduction to R's typing system

Everything in R is an object. An object is a container of data endowed with a label describing the data. Objects can be created, destroyed, or overwritten on-the-fly by the user. The function class returns the class label of an R object.

```
class(c("abc", "def"))
## [1] "character"
class(1:10)
## [1] "integer"
class(c(pi, exp(1)))
## [1] "numeric"
class(factor(c("abc", "def")))
## [1] "factor"
# all columns in a data.frame
sapply(dalton.df, class)
##        name       birth       death
## "character"   "numeric"   "numeric"
```

For the user of R these class labels are usually enough to handle R objects in R scripts. Under the hood, the basic R objects are stored as C structures as C is the language in which R itself has been written. The type of C structure that is used to store a basic type can be found with the typeof function. Compare the results below with those in the previous code snippet.

```
typeof(c("abc", "def"))
## [1] "character"
typeof(1:10)
## [1] "integer"
typeof(c(pi, exp(1)))
## [1] "double"
typeof(factor(c("abc", "def")))
## [1] "integer"
```

Note that the type of an R object of class `numeric` is `double`. The term `double` refers to double precision, which is a standard way for lower-level computer languages such as C to store approximations of real numbers. Also, the type of an object of class `factor` is `integer`. The reason is that R saves memory (and computational time!) by storing factor values as integers, while a translation table between factor and integers are kept in memory. Normally, a user should not have to worry about these subtleties, but there are exceptions (the homework includes an example of the subtleties).

In short, one may regard the class of an object as the object's type from the user's point of view while the type of an object is the way R looks at the object. It is important to realize that R's coercion functions are fundamentally functions that change the underlying type of an object and that class changes are a consequence of the type changes.

11.4.2 Recoding factors

In R, the value of categorical variables is stored in factor variables. A factor is an integer vector endowed with a table specifying what integer value corresponds to what level. The values in this translation table can be requested with the levels function.

```
f <- factor(c("a", "b", "a", "a", "c"))
f
```

```
## [1] a b a a c
## Levels: a b c
levels(f)
## [1] "a" "b" "c"
as.numeric(f)
## [1] 1 2 1 1 3
```

You may need to create a translation table by hand. For example, suppose we read in a vector where `1` stands for `male`, `2` stands for `female` and `0` stands for `unknown`. Conversion to a factor variable can be done as in the example below.

```
# example:
gender <- c(2, 1, 1, 2, 0, 1, 1)
gender
## [1] 2 1 1 2 0 1 1
# recoding table, stored in a simple vector
recode <- c(male = 1, female = 2)
recode
##   male female
##      1      2
gender <- factor(gender, levels = recode, labels = names(recode))
gender
## [1] female male   male   female <NA>   male   male
## Levels: male female
```

Note that we do not explicitly need to set NA as a label. Every integer value that is encountered in the first argument, but not in the levels argument will be regarded missing.

Levels in a factor variable have no natural ordering. However in multivariate (regression) analyses it can be beneficial to fix one of the levels as the reference level. R's standard multivariate routines (lm, glm) use the first level as reference level. The relevel function allows you to determine which level comes first.

```
gender <- relevel(gender, ref = "female")
gender
## [1] female male   male   female <NA>   male   male
## Levels: female male
```

Levels can also be reordered, depending on the mean value of another variable, for example:

```
age <- c(27, 52, 65, 34, 89, 45, 68)
gender <- reorder(gender, age)
gender
## [1] female male   male   female <NA>   male   male
## attr(,"scores")
```

```
## female   male
##   30.5   57.5
## Levels: female male
```

Here, the means are added as a named vector attribute to gender. It can be removed by setting that attribute to NULL.

```
attr(gender, "scores") <- NULL
gender
```

```
## [1] female male   male   female <NA>   male   male
## Levels: female male
```

11.4.3 Converting dates

The base R installation has three types of objects to store a time instance: `Date`, `POSIXlt`, and `POSIXct`. The `Date` object can only be used to store dates, the other two store date and/or time. Here, we focus on converting text to `POSIXct` objects since this is the most portable way to store such information.

Under the hood, a `POSIXct` object stores the number of seconds that have passed since January 1, 1970 00:00. Such a storage format facilitates the calculation of durations by subtraction of two `POSIXct` objects.

When a `POSIXct` object is printed, R shows it in a human-readable calender format. For example, the command `Sys.time()` returns the system time provided by the operating system in `POSIXct` format.

```
current_time <- Sys.time()
class(current_time)
```

```
## [1] "POSIXct" "POSIXt"
```

```
current_time
```

```
## [1] "2017-08-17 17:58:34 MDT"
```

Here, `Sys.time()` uses the time zone that is stored in the locale settings of the machine running R.

Converting from a calender time to POSIXct and back is not entirely trivial, since there are many idiosyncrasies to handle in calender systems. These include leap days, leap seconds, daylight saving times, time zones and so on. Converting from text to `POSIXct` is further complicated by the many textual conventions of time/date denotation. For example, both 28 September 1976 and 1976/09/28 indicate the same day of the same year. Moreover, the name of the month (or

weekday) is language-dependent, where the language is again defined in the operating system's locale settings.

The `lubridate` package contains a number of functions facilitating the conversion of text to `POSIXct` dates. As an example, consider the following code.

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
## The following object is masked from 'package:plyr':
##
##     here
## The following object is masked from 'package:base':
##
##     date
```

```
dates <- c("15/02/2013"
         , "15 Feb 13"
         , "It happened on 15 02 '13")
dmy(dates)
```

```
## [1] "2013-02-15" "2013-02-15" "2013-02-15"
```

Here, the function `dmy` assumes that dates are denoted in the order day-month-year and tries to extract valid dates. Note that the code above will only work properly in locale settings where the name of the second month is abbreviated to Feb. This holds for English or Dutch locales, but fails for example in a French locale (Fevrier).

There are similar functions for all permutations of `d`, `m`, and `y`. Explicitly, all of the following functions exist.

```
dmy()
dym()
mdy()
myd()
ydm()
ymd()
```

So once it is known in what order days, months and years are denoted, extraction is very easy.

Note It is not uncommon to indicate years with two numbers, leaving out the indication of century. Recently in R, 00-69 was interpreted as 2000-2069 and 70-99 as 1970-1999; this behaviour is according to the 2008 POSIX standard, but one should expect that this interpretation changes over time. Currently all are now 2000-2099.

```
dmy("01 01 68")
## [1] "2068-01-01"
dmy("01 01 69")
## [1] "1969-01-01"
dmy("01 01 90")
## [1] "1990-01-01"
dmy("01 01 00")
## [1] "2000-01-01"
```

It should be noted that `lubridate` (as well as R's base functionality) is only capable of converting certain standard notations. For example, the following notation does not convert.

```
dmy("15 Febr. 2013")
## Warning:  All formats failed to parse.  No formats found.
## [1] NA
```

The standard notations that can be recognized by R, either using `lubridate` or R's built-in functionality are shown below. The complete list can be found by typing `?strptime` in the R console. These are the day, month, and year formats recognized by R.

Code	Description	Example
`%a`	Abbreviated weekday name in the current locale.	`Mon`
`%A`	Full weekday name in the current locale.	`Monday`
`%b`	Abbreviated month name in the current locale.	`Sep`
`%B`	Full month name in the current locale.	`September`
`%m`	Month number (01-12)	`09`
`%d`	Day of the month as decimal number (01-31).	`28`
`%y`	Year without century (00-99)	`13`
`%Y`	Year including century.	`2013`

Here, the names of (abbreviated) week or month names that are sought for in the text depend on the locale settings of the machine that is running R.

If you know the textual format that is used to describe a date in the input, you may want to use R's core functionality to convert from text to `POSIXct`. This can be done with the `as.POSIXct` function. It takes as arguments a character

vector with time/date strings and a string describing the format.

```
dates <- c("15-9-2009", "16-07-2008", "17 12-2007", "29-02-2011")
as.POSIXct(dates, format = "%d-%m-%Y")
```

```
## [1] "2009-09-15 MDT" "2008-07-16 MDT" NA
## [4] NA
```

In the format string, date and time fields are indicated by a letter preceded by a percent sign (`%`). Basically, such a `%`-code tells R to look for a range of substrings. For example, the `%d` indicator makes R look for numbers 1-31 where precursor zeros are allowed, so 01, 02, ..., 31 are recognized as well. Strings that are not in the exact format specified by the format argument (like the third string in the above example) will not be converted by `as.POSIXct`. Impossible dates, such as the leap day in the fourth date above are also not converted.

Finally, to convert dates from `POSIXct` back to `character`, one may use the format function that comes with base R. It accepts a `POSIXct` date/time object and an output format string.

```
mybirth <- dmy("28 Sep 1976")
format(mybirth, format = "I was born on %B %d, %Y")
```

```
## [1] "I was born on September 28, 1976"
```

11.5 Character-type manipulation

Because of the many ways people can write the same things down, character data can be difficult to process. For example, consider the following excerpt of a data set with a gender variable.

```
gender
M
male
Female
fem.
```

If this would be treated as a factor variable without any preprocessing, obviously four, not two classes would be stored. The job at hand is therefore to automatically recognize from the above data whether each element pertains to male or female. In statistical contexts, classifying such "messy" text strings into a number of fixed categories is often referred to as *coding*.

Below we discuss two complementary approaches to string coding: *string normalization* and *approximate text matching*. In particular, the following topics are discussed.

- Remove prepending or trailing white spaces.
- Pad strings to a certain width.
- Transform to upper/lower case.
- Search for strings containing simple patterns (substrings).
- Approximate matching procedures based on string distances.

11.5.1 String normalization

String normalization techniques are aimed at transforming a variety of strings to a smaller set of string values which are more easily processed. By default, R comes with extensive string manipulation functionality that is based on the two basic string operations: *finding* a pattern in a string and *replacing* one pattern with another. We will deal with R's generic functions below but start by pointing out some common string cleaning operations.

The `stringr` package offers a number of functions that make some some string manipulation tasks a lot easier than they would be with R's base functions. For example, extra white spaces at the beginning or end of a string can be removed using `str_trim()`.

```
library(stringr)
str_trim(" hello world ")
## [1] "hello world"
str_trim(" hello world ", side = "left")
## [1] "hello world "
str_trim(" hello world ", side = "right")
## [1] " hello world"
```

Conversely, strings can be padded with spaces or other characters with `str_pad()` to a certain width. For example, numerical codes are often represented with prepending zeros.

```
str_pad(112, width = 6, side = "left", pad = 0)
## [1] "000112"
```

Both `str_trim()` and `str_pad()` accept a side argument to indicate whether trimming or padding should occur at the beginning (`left`), end (`right`), or both

sides of the string.

Converting strings to complete upper or lower case can be done with R's built-in `toupper()` and `tolower()` functions.

```
toupper("Hello world")
## [1] "HELLO WORLD"
tolower("Hello World")
## [1] "hello world"
```

11.5.2 Approximate string matching

There are two forms of string matching. The first consists of determining whether a (range of) substring(s) occurs within another string. In this case one needs to specify a range of substrings (called a *pattern*) to search for in another string. In the second form one defines a distance metric between strings that measures how "different" two strings are. Below we will give a short introduction to pattern matching and string distances with R.

There are several pattern matching functions that come with base R. The most used are probably `grep()` and `grepl()`. Both functions take a pattern and a character vector as input. The output only differs in that `grepl()` returns a logical index, indicating which element of the input character vector contains the pattern, while `grep()` returns a numerical index. You may think of `grep(...)` as `which(grepl(...))`.

In the most simple case, the pattern to look for is a simple substring. For example, from the previous example, we get the following.

```
gender <- c("M", "male ", "Female", "fem.")
grepl("m", gender)
## [1] FALSE  TRUE  TRUE  TRUE
grep("m", gender)
## [1] 2 3 4
```

Note that the result is case sensitive: the capital `M` in the first element of gender does not match the lower case `m`. There are several ways to circumvent this case sensitivity. Either by case normalization or by the optional argument `ignore.case`.

```
grepl("m", gender, ignore.case = TRUE)
## [1] TRUE TRUE TRUE TRUE
```

```
grepl("m", tolower(gender))
## [1] TRUE TRUE TRUE TRUE
```

Obviously, looking for the occurrence of `m` or `M` in the gender vector does not allow us to determine which strings pertain to male and which not. Preferably we would like to search for strings that start with an `m` or `M`. Fortunately, the search patterns that `grep()` accepts allow for such searches. The beginning of a string is indicated with a caret (`^`).

```
grepl("^m", gender, ignore.case = TRUE)
## [1]  TRUE  TRUE FALSE FALSE
```

Indeed, the `grepl()` function now finds only the first two elements of gender. The caret is an example of a so-called meta-character. That is, it does not indicate the caret itself but something else, namely the beginning of a string. The search patterns that `grep()`, `grepl()` (and `sub()` and `gsub()`) understand have more of these meta-characters, namely:

```
. \ | ( ) [ { ^ $ * + ?
```

If you need to search a string for any of these characters, you can use the option `fixed=TRUE`.

```
grepl("^", gender, fixed = TRUE)
## [1] FALSE FALSE FALSE FALSE
```

This will make `grepl()` or `grep()` ignore any meta-characters in the search string (and thereby search for the "^" character).

Search patterns using meta-characters are called *regular expressions*. Regular expressions[3] offer powerful and flexible ways to search (and alter) text. A concise description of regular expressions allowed by R's built-in string processing functions can be found by typing `?regex` at the R command line. If you frequently have to deal with "messy" text variables, learning to work with regular expressions is a worthwhile investment. Moreover, since many popular programming languages support some dialect of regexps, it is an investment that could pay off several times.

We now turn our attention to the second method of approximate matching, namely string distances. A string distance is an algorithm or equation that indicates how much two strings differ from each other. An important distance measure is implemented by the R's native `adist()` function. This function counts how many basic operations are needed to turn one string into another. These operations include insertion, deletion, or substitution of a single charac-

[3] `http://en.wikipedia.org/wiki/Regular_expression`

ter. For example

```
adist("abc", "bac")
```

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

The result equals two since turning "abc" into "bac" involves two character substitutions: abc $\rightarrow$ bbc $\rightarrow$ bac.

Using `adist()`, we can compare fuzzy text strings to a list of known codes. For example:

```
codes <- c("male", "female")
# calculate pairwise distances between the gender strings and codes strings
dist.g.c <- adist(gender, codes)
# add column and row names
colnames(dist.g.c) <- codes
rownames(dist.g.c) <- gender
dist.g.c
```

```
##        male female
## M         4      6
## male      1      3
## Female    2      1
## fem.      4      3
```

Here, `adist()` returns the distance matrix between our vector of fixed codes and the input data. For readability we added row and column names accordingly. Now, to find out which code matches best with our raw data, we need to find the index of the smallest distance for each row of dist.g.c. This can be done as follows.

```
ind.min <- apply(dist.g.c, 1, which.min)
data.frame(rawtext = gender, coded = codes[ind.min])
```

```
##   rawtext  coded
## 1       M   male
## 2    male   male
## 3  Female female
## 4    fem. female
```

We use `apply()` to apply `which.min()` to every row of `dist.g.c`. Note that in the case of multiple minima, the first match will be returned. At the end of this subsection we show how this code can be simplified with the `stringdist` package.

Finally, we mention three more functions based on string distances. First, the R built-in function `agrep()` is similar to `grep()`, but it allows one to specify

a maximum Levenshtein distance[4] between the input pattern and the found substring. The `agrep()` function allows for searching for regular expression patterns, which makes it very flexible.

Secondly, the `stringdist` package offers a function called `stringdist()` which can compute a variety of string distance metrics, some of which are likely to provide results that are better than `adist()`'s. Most importantly, the distance function used by `adist()` does not allow for character transpositions, which is a common typographical error. Using the optimal string alignment distance (the default choice for `stringdist()`) we get

```
library(stringdist)
stringdist("abc", "bac")
```

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

The answer is now 1 (not 2 as with `adist()`), since the optimal string alignment distance allows for transpositions of adjacent characters: abc $\rightarrow$ bac.

Thirdly, the `stringdist` package provides a function called `amatch()`, which mimics the behaviour of R's `match()` function: it returns an index to the closest match within a maximum distance. Recall the earlier gender and code example.

```
# this yields the closest match of 'gender' in 'codes' (within a distance of 4)
ind <- amatch(gender, codes, maxDist = 4)
ind
```

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

```
# store results in a data.frame
data.frame(rawtext = gender, code = codes[ind])
```

```
##   rawtext   code
## 1       M   male
## 2    male   male
## 3  Female female
## 4    fem. female
```

[4]Informally, the Levenshtein distance between two words is the minimum number of single-character edits (i.e., insertions, deletions, or substitutions) required to change one word into the other: `https://en.wikipedia.org/wiki/Levenshtein_distance`.

11.6 From technically correct data to consistent data

Consistent data are technically correct data that are fit for statistical analysis. They are data in which missing values, special values, (obvious) errors and outliers are either removed, corrected, or imputed. The data are consistent with constraints based on real-world knowledge about the subject that the data describe.

Consistency can be understood to include **in-record consistency**, meaning that no contradictory information is stored in a single record, and **cross-record consistency**, meaning that statistical summaries of different variables do not conflict with each other. Finally, one can include cross-dataset consistency, meaning that the dataset that is currently analyzed is consistent with other datasets pertaining to the same subject matter. In this tutorial *we mainly focus on methods dealing with in-record consistency*, with the exception of outlier handling which can be considered a cross-record consistency issue.

The process towards consistent data always involves the following three steps.

- **Detection** *of an inconsistency.* That is, one establishes which constraints are violated. For example, an age variable is constrained to non-negative values.

- **Selection** *of the field or fields causing the inconsistency.* This is trivial in the case of a univariate demand as in the previous step, but may be more cumbersome when cross-variable relations are expected to hold. For example the marital status of a child must be unmarried. In the case of a violation it is not immediately clear whether age, marital status, or both are wrong.

- **Correction** *of the fields that are deemed erroneous by the selection method.* This may be done through deterministic (model-based) or stochastic methods.

For many data correction methods these steps are not necessarily neatly separated.

First, we introduce a number of techniques dedicated to the detection of errors and the selection of erroneous fields. If the field selection procedure is performed separately from the error detection procedure, it is generally referred to as **error localization**. Next, we describe techniques that implement correction methods based on "direct rules" or "deductive correction". In these techniques, erroneous values are replaced by better ones by directly deriving them from other values in the same record. Finally, we give an overview of some commonly used imputation techniques that are available in R.

11.6.1 Detection and localization of errors

This section details a number of techniques to detect univariate and multivariate constraint violations.

Missing values

A missing value, represented by `NA` in R, is a placeholder for a datum of which the type is known but its value isn't. Therefore, it is impossible to perform statistical analysis on data where one or more values in the data are missing. One may choose to either omit elements from a dataset that contain missing values or to impute a value, but missingness is something to be dealt with prior to any analysis.

In practice, analysts, but also commonly used numerical software may confuse a missing value with a default value or category. For instance, in Excel 2010, the result of adding the contents of a field containing the number 1 with an empty field results in 1. This behaviour is most definitely unwanted since Excel silently imputes "0" where it should have said something along the lines of "unable to compute". It should be up to the analyst to decide how empty values are handled, since a default imputation may yield unexpected or erroneous results for reasons that are hard to trace.

Another commonly encountered mistake is to confuse an `NA` in categorical data with the category *unknown*. If *unknown* is indeed a category, it should be added as a factor level so it can be appropriately analyzed. Consider as an example a categorical variable representing place of birth. Here, the category *unknown* means that we have no knowledge about where a person is born. In contrast, `NA` indicates that we have no information to determine whether the birth place is known or not.

The behaviour of R's core functionality is completely consistent with the idea that the analyst must decide what to do with missing data. A common choice, namely "leave out records with missing data" is supported by many base functions through the `na.rm` option.

```
age <- c(23, 16, NA)
mean(age)
## [1] NA
mean(age, na.rm = TRUE)
## [1] 19.5
```

Functions such as `sum()`, `prod()`, `quantile()`, `sd()`, and so on all have this option. Functions implementing bivariate statistics such as `cor()` and `cov()` offer options to include complete or pairwise complete values.

Besides the `is.na()` function, that was already mentioned previously, R comes with a few other functions facilitating `NA` handling. The `complete.cases()` function detects rows in a `data.frame` that do not contain any missing value. Recall the person data set example from earlier.

```
print(person)
##   age height
## 1  21    6.0
## 2  42    5.9
## 3  18     NA
## 4  21     NA
complete.cases(person)
## [1]  TRUE  TRUE FALSE FALSE
```

The resulting logical can be used to remove incomplete records from the `data.frame`. Alternatively the `na.omit()` function, does the same.

```
persons_complete <- na.omit(person)
persons_complete
```

```
##   age height
## 1  21    6.0
## 2  42    5.9
```

```
na.action(persons_complete)
```

```
## 3 4
## 3 4
## attr(,"class")
## [1] "omit"
```

The result of the `na.omit()` function is a `data.frame` where incomplete rows have been deleted. The `row.names` of the removed records are stored in an attribute called `na.action`.

Note. It may happen that a missing value in a data set means 0 or Not applicable. If that is the case, it should be explicitly imputed with that value, because it is not unknown, but was coded as empty.

Special values

As explained previously, numeric variables are endowed with several formalized special values including $\pm$`Inf`, `NA`, and `NaN`. Calculations involving special values often result in special values, and since a statistical statement about a real-world phenomenon should never include a special value, it is desirable to handle special values prior to analysis. For numeric variables, special values indicate values that are not an element of the mathematical set of real numbers. The function `is.finite()` determines which values are "regular" values.

```
is.finite(c(1, Inf, NaN, NA))
```

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

This function accepts vectorial input. With little effort we can write a function that may be used to check every numerical column in a `data.frame`.

```
f.is.special <- function(x) {
  if (is.numeric(x)) {
    return(!is.finite(x))
  } else {
```

```
    return(is.na(x))
  }
}
person
```

```
##   age height
## 1  21    6.0
## 2  42    5.9
## 3  18     NA
## 4  21     NA
```

```
sapply(person, f.is.special)
```

```
##        age height
## [1,] FALSE  FALSE
## [2,] FALSE  FALSE
## [3,] FALSE   TRUE
## [4,] FALSE   TRUE
```

Here, the `f.is.special()` function is applied to each column of person using `sapply()`. `f.is.special()` checks its input vector for numerical special values if the type is `numeric`, otherwise it only checks for `NA`.

Outliers

There is a vast body of literature on outlier detection, and several definitions of **outlier** exist. A general definition by Barnett and Lewis defines an outlier in a data set as *an observation (or set of observations) which appear to be inconsistent with that set of data.* Although more precise definitions exist (see e.g., the book by Hawkins), this definition is sufficient for the current chapter. Below we mention a few fairly common graphical and computational techniques for outlier detection in univariate numerical data. In a previous chapter, we've discussed using PCA as a graphical technique to help detect multivariate outliers.

Note. Outliers do not equal errors. They should be detected, but not necessarily removed. Their inclusion in the analysis is a statistical decision.

For more or less unimodal and symmetrically distributed data, Tukey's box-and-whisker method for outlier detection is often appropriate. In this method, an observation is an outlier when it is larger than the so-called "whiskers" of

the set of observations. The upper whisker is computed by adding 1.5 times the interquartile range to the third quartile and rounding to the nearest lower observation. The lower whisker is computed likewise. The base R installation comes with function `boxplot.stats()`, which, amongst other things, list the outliers.

```
x <- c(1:10, 20, 30)
boxplot.stats(x)
```

```
## $stats
## [1]  1.0  3.5  6.5  9.5 10.0
##
## $n
## [1] 12
##
## $conf
## [1] 3.76336 9.23664
##
## $out
## [1] 20 30
```

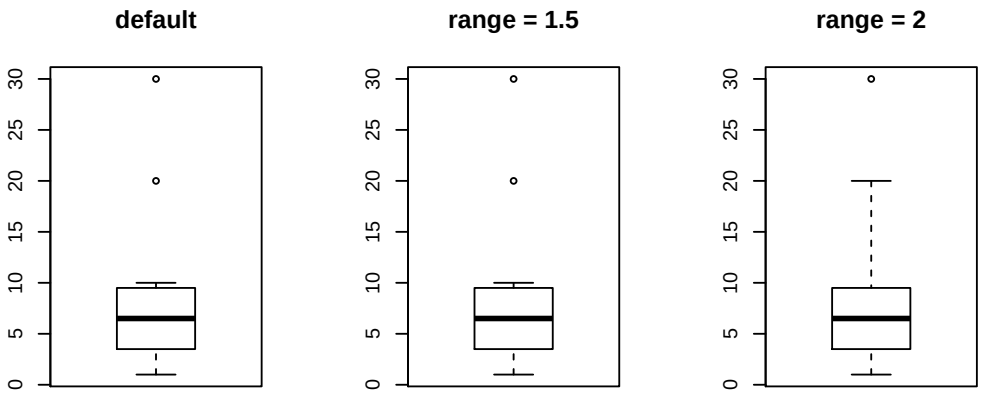
Here, `20` and `30` are detected as outliers since they are above the upper whisker of the observations in `x`. The factor 1.5 used to compute the whisker is to an extent arbitrary and it can be altered by setting the coef option of `boxplot.stats()`. A higher coefficient means a higher outlier detection limit (so for the same dataset, generally less upper or lower outliers will be detected).

```
boxplot.stats(x, coef = 2)$out
```

```
## [1] 30
```

The box-and-whisker method can be visualized with the box-and-whisker plot, where the box indicates the interquartile range and the median, the whiskers are represented at the ends of the box-and-whisker plots and outliers are indicated as separate points above or below the whiskers.

```
op <- par(no.readonly = TRUE)          # save plot settings
par(mfrow=c(1,3))
boxplot(x, main="default")
boxplot(x, range = 1.5, main="range = 1.5")
boxplot(x, range = 2,   main="range = 2")
par(op)                                # restore plot settings
```



The box-and-whisker method fails when data distribution is skewed, as in an exponential or log-normal distribution. In that case one can attempt to transform the data, for example with a logarithm or square root transformation. Another option is to use a method that takes the skewness into account.

A particularly easy-to-implement method for outlier detection with positive observations is by Hiridoglou and Berthelot. In this method, an observation is an outlier when

$$h(x) \;=\; \max\left(\frac{x}{x^*}, \frac{x^*}{x}\right) \;\geq\; r, \quad \text{and} \quad x > 0.$$

Here, r is a user-defined reference value and x^* is usually the median observation, although other measures of centrality may be chosen. Here, the score function $h(x)$ grows as $1/x$ as x approaches zero and grows linearly with x when it is larger than x^*. It is therefore appropriate for finding outliers on both sides of the distribution. Moreover, because of the different behaviour for small and large x-values, it is appropriate for skewed (long-tailed) distributions. An implementation of this method in R does not seem available but it is implemented simple enough as follows.

```
f.hb.outlier <- function(x,r) {
  x <- x[is.finite(x)]
  stopifnot(length(x) > 0 , all(x>0)) # if empty vector or non-positive values, quit
  xref <- median(x)
  if (xref <= sqrt(.Machine$double.eps)) {
    warning("Reference value close to zero: results may be inaccurate")
  }
```

```
  pmax(x/xref, xref/x) > r
}
f.hb.outlier(x, r = 4)
```

```
##  [1]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [12]  TRUE
```

The above function returns a logical vector indicating which elements of x are outliers.

11.6.2 Edit rules for detecting obvious inconsistencies

An obvious inconsistency occurs when a record contains a value or combination of values that cannot correspond to a real-world situation. For example, a person's age cannot be negative, a man cannot be pregnant and an under-aged person cannot possess a drivers license.

Such knowledge can be expressed as *rules* or constraints. In data editing literature these rules are referred to as *edit rules* or *edits*, in short. Checking for obvious inconsistencies can be done straightforwardly in R using logical indices and recycling. For example, to check which elements of `x` obey the rule '`x` must be non negative' one simply uses the following.

```
x_nonnegative <- (x >= 0)
```

However, as the number of variables increases, the number of rules may increase rapidly and it may be beneficial to manage the rules separate from the data. Moreover, since multivariate rules may be interconnected by common variables, deciding which variable or variables in a record cause an inconsistency may not be straightforward.

The `editrules` package allows one to define rules on categorical, numerical or mixed-type data sets which each record must obey. Furthermore, `editrules` can check which rules are obeyed or not and allows one to find the minimal set of variables to adapt so that all rules can be obeyed. The package also implements a number of basic rule operations allowing users to test rule sets for contradictions and certain redundancies.

As an example, we will work with a small file containing the following data.

```
age,agegroup,height,status,yearsmarried
21,adult,6.0,single,-1
2,child,3,married, 0
18,adult,5.7,married, 20
221,elderly, 5,widowed, 2
34,child, -7,married, 3
```

We read this data into a variable called people and define some restrictions on age using `editset()`.

```
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch18_people.txt"
people <- read.csv(fn.data)

people
```

```
##   age agegroup height  status yearsmarried
## 1  21    adult    6.0  single          -1
## 2   2    child    3.0 married           0
## 3  18    adult    5.7 married          20
## 4 221  elderly    5.0 widowed           2
## 5  34    child   -7.0 married           3
```

```
library(editrules)
```

```
## Loading required package:  igraph
##
## Attaching package:  'igraph'
## The following objects are masked from 'package:lubridate':
##
##    %--%, union
## The following objects are masked from 'package:stats':
##
##    decompose, spectrum
## The following object is masked from 'package:base':
##
##    union
##
## Attaching package:  'editrules'
## The following objects are masked from 'package:igraph':
##
##    blocks, normalize
```

```
E <- editset(c("age >=0", "age <= 150"))
E
```

```
##
## Edit set:
## num1 : 0 <= age
## num2 : age <= 150
```

The `editset()` function parses the textual rules and stores them in an

`editset` object. Each rule is assigned a name according to it's type (`numeric`, `categorical`, or `mixed`) and a number. The data can be checked against these rules with the `violatedEdits()` function. Record 4 contains an error according to one of the rules: an `age` of 221 is not allowed.

```
violatedEdits(E, people)
```

```
##       edit
## record  num1  num2
##      1 FALSE FALSE
##      2 FALSE FALSE
##      3 FALSE FALSE
##      4 FALSE  TRUE
##      5 FALSE FALSE
```

`violatedEdits()` returns a `logical` array indicating for each row of the data, which rules are violated. The number and type of rules applying to a data set usually quickly grow with the number of variables. With `editrules`, users may read rules, specified in a limited R-syntax, directly from a text file using the `editfile()` function. As an example consider the contents of the following text file (note, you can't include braces in your `if()` statement).

```
# numerical rules
age >= 0
height > 0
age <= 150
age > yearsmarried

# categorical rules
status %in% c("married", "single", "widowed")
agegroup %in% c("child", "adult", "elderly")
if ( status == "married" )  agegroup %in% c("adult","elderly")

# mixed rules
if ( status %in% c("married","widowed"))  age - yearsmarried >= 17
if ( age < 18 )              agegroup == "child"
if ( age >= 18 && age <65 )  agegroup == "adult"
if ( age >= 65 )             agegroup == "elderly"
```

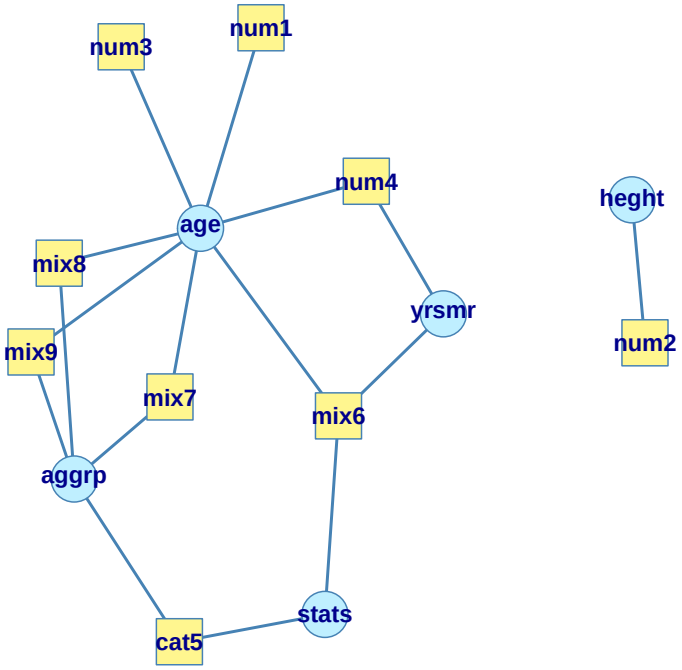
There are rules pertaining to purely numerical, purely categorical and rules pertaining to both data types. Moreover, there are univariate as well as multivariate rules. Comments are written behind the usual # character. The rule set can be read as follows.

```
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch18_edits.txt"
E <- editfile(fn.data)
E
```

```
##
## Data model:
## dat6 : agegroup %in% c('adult', 'child', 'elderly')
## dat7 : status %in% c('married', 'single', 'widowed')
##
## Edit set:
## num1 : 0 <= age
## num2 : 0 < height
## num3 : age <= 150
## num4 : yearsmarried < age
## cat5 : if( agegroup == 'child' ) status != 'married'
## mix6 : if( age < yearsmarried + 17 ) !( status %in% c('married', 'widowed') )
## mix7 : if( age < 18 ) !( agegroup %in% c('adult', 'elderly') )
## mix8 : if( 18 <= age & age < 65 ) !( agegroup %in% c('child', 'elderly') )
## mix9 : if( 65 <= age ) !( agegroup %in% c('adult', 'child') )
```

Since rules may pertain to multiple variables, and variables may occur in several rules (e.g., the `age` variable in the current example), there is a dependency between rules and variables. It can be informative to show these dependencies in a graph using the plot function. Below the graph plot shows the interconnection of restrictions. Blue circles represent variables and yellow boxes represent restrictions. The lines indicate which restrictions pertain to what variables.

```
op <- par(no.readonly = TRUE)          # save plot settings
par(mfrow=c(1,1), mar = c(0,0,0,0))
plot(E)
par(op)                                # restore plot settings
```

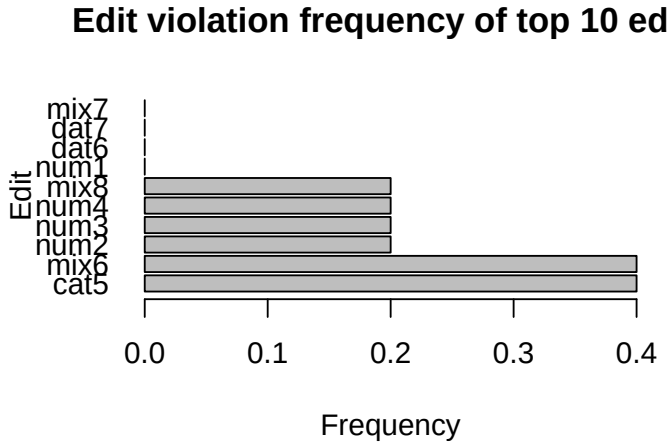


As the number of rules grows, looking at the full array produced by `violatedEdits(` becomes cumbersome. For this reason, `editrules` offers methods to summarize or visualize the result.

```
ve <- violatedEdits(E, people)
summary(ve)
```

```
## Edit violations, 5 observations, 0 completely missing (0%):
##
##  editname freq rel
##      cat5    2 40%
##      mix6    2 40%
##      num2    1 20%
##      num3    1 20%
##      num4    1 20%
##      mix8    1 20%
##
## Edit violations per record:
##
##  errors freq rel
##       0    1 20%
##       1    1 20%
##       2    2 40%
##       3    1 20%
```

```
plot(ve)
```



Here, the edit labeled `cat5` is violated by two records (20% of all records). Violated edits are sorted from most to least often violated. The plot visualizes the same information.

Error localization

The interconnectivity of edits is what makes error localization difficult. For example, the graph above shows that a record violating edit `num4` may contain an error in `age` and/or `yrsmr` (years married). Suppose that we alter `age` so that `num4` is not violated anymore. We then run the risk of violating up to *six* other edits containing `age`.

If we have no other information available but the edit violations, it makes sense to minimize the number of fields being altered. This principle, commonly

referred to as the principle of Fellegi and Holt, is based on the idea that errors occur relatively few times and when they do, they occur randomly across variables. Over the years several algorithms have been developed to solve this minimization problem of which two have been implemented in `editrules`. The `localizeErrors()` function provides access to this functionality.

As an example we take two records from the `people` dataset from the previous subsection.

```
id <- c(2, 5)
people[id, ]
```

```
##   age agegroup height  status yearsmarried
## 2   2    child      3 married            0
## 5  34    child     -7 married            3
```

```
violatedEdits(E, people[id, ])
```

```
##       edit
## record  num1  num2  num3  num4  dat6  dat7 cat5  mix6  mix7  mix8
##      2 FALSE FALSE FALSE FALSE FALSE FALSE TRUE  TRUE FALSE FALSE
##      5 FALSE  TRUE FALSE FALSE FALSE FALSE TRUE FALSE FALSE  TRUE
##       edit
## record  mix9
##      2 FALSE
##      5 FALSE
```

Record 2 violates `mix6` while record 5 violates edits `num2`, `cat5`, and `mix8`. We use `localizeErrors()`, with a mixed-integer programming (MIP) approach to find the minimal set of variables to adapt.

```
le <- localizeErrors(E, people[id, ], method = "mip")
le$adapt
```

```
##     age agegroup height status yearsmarried
## 2 FALSE    FALSE  FALSE   TRUE        FALSE
## 5 FALSE     TRUE   TRUE  FALSE        FALSE
```

Here, the `le` object contains some processing metadata and a logical array labeled `adapt` which indicates the minimal set of variables to be altered in each record. It can be used in correction and imputation procedures for filling in valid values. Such procedures are not part of `editrules`, but for demonstration purposes we will manually fill in new values showing that the solution computed by `localizeErrors()` indeed allows one to repair records to full compliance with all edit rules.

```
people[2, "status"]   <- "single"
people[5, "height"]   <- 7
people[5, "agegroup"] <- "adult"
summary(violatedEdits(E, people[id, ]))
```

```
## No violations detected, 0 checks evaluated to NA
## NULL
```

The behaviour of `localizeErrors()` can be tuned with various options. It is possible to supply a confidence weight for each variable allowing for fine grained control on which values should be adapted. It is also possible to choose a branch-and-bound based solver (instead of the MIP solver used here), which is typically slower but allows for more control.

11.6.3 Correction

Correction methods aim to fix inconsistent observations by altering invalid values in a record based on information from valid values. Depending on the method this is either a single-step procedure or a two-step procedure where first, an error localization method is used to empty certain fields, followed by an imputation step.

In some cases, the cause of errors in data can be determined with enough certainty so that the solution is almost automatically known. In recent years, several such methods have been developed and implemented in the `deducorrect` package.

For the purposes of ADA1, we will manually correct errors, either by replacing values or by excluding observations.

Simple transformation rules

In practice, data cleaning procedures involve a lot of *ad-hoc* transformations. This may lead to long scripts where one selects parts of the data, changes some variables, selects another part, changes some more variables, etc. When such scripts are neatly written and commented, they can almost be treated as a log of the actions performed by the analyst. However, as scripts get longer it is better

to store the transformation rules separately and log which rule is executed on what record. The deducorrect package offers functionality for this. Consider as an example the following (fictitious) dataset listing the body length of some brothers.

```
marx <- read.table(text = "
name    height unit
Groucho 170.00 cm
Zeppo     1.74 m
Chico    70.00 inch
Gummo   168.00 cm
Harpo     5.91 ft
", header=TRUE, stringsAsFactors = FALSE)
marx
```

```
##      name height unit
## 1 Groucho 170.00   cm
## 2   Zeppo   1.74    m
## 3   Chico  70.00 inch
## 4   Gummo 168.00   cm
## 5   Harpo   5.91   ft
```

The task here is to standardize the lengths and express all of them in meters. The obvious way would be to use indexing techniques, which would look something like this.

```
marx_m <- marx
ind <- (marx$unit == "cm")                    # indexes for cm
marx_m[ind, "height"] <- marx$height[ind] / 100
marx_m[ind, "unit"]   <- "m"
ind <- (marx$unit == "inch")                  # indexes for inch
marx_m[ind, "height"] <- marx$height[ind] / 39.37
marx_m[ind, "unit"]   <- "m"
ind <- (marx$unit == "ft")                    # indexes for ft
marx_m[ind, "height"] <- marx$height[ind] / 3.28
marx_m[ind, "unit"]   <- "m"
marx_m
```

```
##      name   height unit
## 1 Groucho 1.700000    m
## 2   Zeppo 1.740000    m
## 3   Chico 1.778004    m
## 4   Gummo 1.680000    m
## 5   Harpo 1.801829    m
```

Such operations quickly become cumbersome. Of course, in this case one could write a for-loop but that would hardly save any code. Moreover, if you want to check afterwards which values have been converted and for what reason,

there will be a significant administrative overhead. The `deducorrect` package takes all this overhead off your hands with the `correctionRules()` functionality. For example, to perform the above task, one first specifies a file with correction rules as follows.

```
# convert centimeters
if ( unit == "cm" ){
  height <- height / 100
  unit <- "m" # set all units to meter
}
# convert inches
if (unit == "inch" ){
  height <- height / 39.37
  unit <- "m" # set all units to meter
}
# convert feet
if (unit == "ft" ){
  height <- height / 3.28
  unit <- "m" # set all units to meter
}
```

With `deducorrect` we can read these rules, apply them to the data and obtain a log of all actual changes as follows.

```
library(deducorrect)
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch18_conversions.txt"
# read the conversion rules.
R <- correctionRules(fn.data)
R
```

```
## Object of class 'correctionRules'
## ##  1-------
##   if (unit == "cm") {
##       height <- height/100
##       unit <- "m"
##   }
## ##  2-------
##   if (unit == "inch") {
##       height <- height/39.37
##       unit <- "m"
##   }
## ##  3-------
##   if (unit == "ft") {
##       height <- height/3.28
##       unit <- "m"
##   }
```

`correctionRules()` has parsed the rules and stored them in a `correctionRules`

object. We may now apply them to the data.

```
cor <- correctWithRules(R, marx)
```

The returned value, `cor`, is a list containing the corrected data

```
cor$corrected
```

```
##      name   height unit
## 1 Groucho 1.700000    m
## 2   Zeppo 1.740000    m
## 3   Chico 1.778004    m
## 4   Gummo 1.680000    m
## 5   Harpo 1.801829    m
```

as well as a log of applied corrections.

```
cor$corrections[1:4]
```

```
##   row variable  old              new
## 1   1   height  170               1.7
## 2   1     unit  cm                m
## 3   3   height 70   1.77800355600711
## 4   3     unit inch m
## 5   4   height  168             1.68
## 6   4     unit  cm               m
## 7   5   height 5.91 1.80182926829268
## 8   5     unit ft   m
```

The log lists for each row, what variable was changed, what the old value was and what the new value is. Furthermore, the fifth column of `cor$corrections` shows the corrections that were applied (not shown above for formatting reasons).

```
cor$corrections[5]
```

```
##                                                          how
## 1     if (unit == "cm") { height <- height/100 unit <- "m" }
## 2     if (unit == "cm") { height <- height/100 unit <- "m" }
## 3 if (unit == "inch") { height <- height/39.37 unit <- "m" }
## 4 if (unit == "inch") { height <- height/39.37 unit <- "m" }
## 5     if (unit == "cm") { height <- height/100 unit <- "m" }
## 6     if (unit == "cm") { height <- height/100 unit <- "m" }
## 7    if (unit == "ft") { height <- height/3.28 unit <- "m" }
## 8    if (unit == "ft") { height <- height/3.28 unit <- "m" }
```

So here, with just two commands, the data is processed and all actions logged in a `data.frame` which may be stored or analyzed. The rules that may be applied with `deducorrect` are rules that can be executed record-by-record.

By design, there are some limitations to which rules can be applied with

`correctWithRules()`. The processing rules should be executable record-by-record. That is, it is not permitted to use functions like `mean()` or `sd()`. The symbols that may be used can be listed as follows.

```
getOption("allowedSymbols")
```

```
##  [1] "if"         "else"       "is.na"      "is.finite" "=="
##  [6] "<"          "<="         "="          ">="        ">"
## [11] "!="         "!"          "%in%"       "identical" "sign"
## [16] "abs"        "||"         "|"          "&&"        "&"
## [21] "("          "{"          "<-"         "="         "+"
## [26] "-"          "*"          "^"          "/"         "%%"
## [31] "%/%"
```

When the rules are read by `correctionRules()`, it checks whether any symbol occurs that is not in the list of allowed symbols and returns an error message when such a symbol is found as in the following example.

```
correctionRules(expression(x <- mean(x)))
```

```
##
## Forbidden symbols found:
## ## ERR  1 ------
## Forbidden symbols: mean
##   x <- mean(x)
```

```
## Error in correctionRules.expression(expression(x <- mean(x))):  Forbidden symbols found
```

Finally, it is currently not possible to add new variables using `correctionRules()` although such a feature will likely be added in the future.

Deductive correction

When the data you are analyzing is generated by people rather than machines or measurement devices, certain typical human-generated errors are likely to occur. Given that data has to obey certain edit rules, the occurrence of such errors can sometimes be detected from raw data with (almost) certainty. Examples of errors that can be detected are typing errors in numbers (under linear restrictions) rounding errors in numbers and sign errors or variable swaps. The `deducorrect` package has a number of functions available that can correct such errors. Below we give some examples, every time with just a single edit rule. The functions can handle larger sets of edits however.

[*I will complete this section if we need it for our Spring semester.*]

Deterministic imputation

In some cases a missing value can be determined because the observed values combined with their constraints force a unique solution.

[*I will complete this section if we need it for our Spring semester.*]

11.6.4 Imputation

Imputation is the process of estimating or deriving values for fields where data is missing. There is a vast body of literature on imputation methods and it goes beyond the scope of this chapter to discuss all of them.

There is no one single best imputation method that works in all cases. The imputation model of choice depends on what auxiliary information is available and whether there are (multivariate) edit restrictions on the data to be imputed. The availability of R software for imputation under edit restrictions is limited. However, a viable strategy for imputing numerical data is to first impute missing values without restrictions, and then minimally adjust the imputed values so that the restrictions are obeyed. Separately, these methods are available in R.

Appendix A

Custom R functions

Contents

A.1 Ch 2. Estimation in One-Sample Problems

```
# a function to compare the bootstrap sampling distribution with
#   a normal distribution with mean and SEM estimated from the data
bs.one.samp.dist <- function(dat, N = 1e4) {
  n <- length(dat);
  # resample from data
  sam <- matrix(sample(dat, size = N * n, replace = TRUE), ncol=N);
  # draw a histogram of the means
  sam.mean <- colMeans(sam);
  # save par() settings
  old.par <- par(no.readonly = TRUE)
  # make smaller margins
  par(mfrow=c(2,1), mar=c(3,2,2,1), oma=c(1,1,1,1))
  # Histogram overlaid with kernel density curve
  hist(dat, freq = FALSE, breaks = 6
       , main = "Plot of data with smoothed density curve")
  points(density(dat), type = "l")
  rug(dat)

  hist(sam.mean, freq = FALSE, breaks = 25
       , main = "Bootstrap sampling distribution of the mean"
       , xlab = paste("Data: n =", n
```

```
                    , ", mean =", signif(mean(dat), digits = 5)
                    , ", se =", signif(sd(dat)/sqrt(n)), digits = 5))
  # overlay a density curve for the sample means
  points(density(sam.mean), type = "l")
  # overlay a normal distribution, bold and red
  x <- seq(min(sam.mean), max(sam.mean), length = 1000)
  points(x, dnorm(x, mean = mean(dat), sd = sd(dat)/sqrt(n))
       , type = "l", lwd = 2, col = "red")
  # place a rug of points under the plot
  rug(sam.mean)
  # restore par() settings
  par(old.par)
}
```

```
# Function ot plot t-distribution with shaded p-value
t.dist.pval <- function(t.summary) {
  par(mfrow=c(1,1))
  lim.extreme <- max(4, abs(t.summary$statistic) + 0.5)
  lim.lower <- -lim.extreme;
  lim.upper <-  lim.extreme;
  x.curve <- seq(lim.lower, lim.upper, length=200)
  y.curve <- dt(x.curve, df = t.summary$parameter)
  plot(x.curve, y.curve, type = "n"
      , ylab = paste("t-dist( df =", signif(t.summary$parameter, 3), ")")
      , xlab = paste("t-stat =", signif(t.summary$statistic, 5)
                     , ", Shaded area is p-value =", signif(t.summary$p.value, 5)))
  if ((t.summary$alternative == "less")
      | (t.summary$alternative == "two.sided")) {
    x.pval.l <- seq(lim.lower, -abs(t.summary$statistic), length=200)
    y.pval.l <- dt(x.pval.l, df = t.summary$parameter)
    polygon(c(lim.lower, x.pval.l, -abs(t.summary$statistic))
          , c(0, y.pval.l, 0), col="gray")
  }
  if ((t.summary$alternative == "greater")
      | (t.summary$alternative == "two.sided")) {
    x.pval.u <- seq(abs(t.summary$statistic), lim.upper, length=200)
    y.pval.u <- dt(x.pval.u, df = t.summary$parameter)
    polygon(c(abs(t.summary$statistic), x.pval.u, lim.upper)
          , c(0, y.pval.u, 0), col="gray")
  }
  points(x.curve, y.curve, type = "l", lwd = 2, col = "blue")
}
```

A.2 Ch 3. Two-Sample Inferences

```r
# a function to compare the bootstrap sampling distribution
#   of the difference of means from two samples with
#   a normal distribution with mean and SEM estimated from the data
bs.two.samp.diff.dist <- function(dat1, dat2, N = 1e4) {
  n1 <- length(dat1);
  n2 <- length(dat2);
  # resample from data
  sam1 <- matrix(sample(dat1, size = N * n1, replace = TRUE), ncol=N);
  sam2 <- matrix(sample(dat2, size = N * n2, replace = TRUE), ncol=N);
  # calculate the means and take difference between populations
  sam1.mean <- colMeans(sam1);
  sam2.mean <- colMeans(sam2);
  diff.mean <- sam1.mean - sam2.mean;
  # save par() settings
  old.par <- par(no.readonly = TRUE)
  # make smaller margins
  par(mfrow=c(3,1), mar=c(3,2,2,1), oma=c(1,1,1,1))
  # Histogram overlaid with kernel density curve
  hist(dat1, freq = FALSE, breaks = 6
      , main = paste("Sample 1", "\n"
                     , "n =", n1
                     , ", mean =", signif(mean(dat1), digits = 5)
                     , ", sd =", signif(sd(dat1), digits = 5))
      , xlim = range(c(dat1, dat2)))
  points(density(dat1), type = "l")
  rug(dat1)

  hist(dat2, freq = FALSE, breaks = 6
      , main = paste("Sample 2", "\n"
                     , "n =", n2
                     , ", mean =", signif(mean(dat2), digits = 5)
                     , ", sd =", signif(sd(dat2), digits = 5))
      , xlim = range(c(dat1, dat2)))
  points(density(dat2), type = "l")
  rug(dat2)

  hist(diff.mean, freq = FALSE, breaks = 25
      , main = paste("Bootstrap sampling distribution of the difference in means", "\n"
                     , "mean =", signif(mean(diff.mean), digits = 5)
                     , ", se =", signif(sd(diff.mean), digits = 5)))
  # overlay a density curve for the sample means
  points(density(diff.mean), type = "l")
  # overlay a normal distribution, bold and red
  x <- seq(min(diff.mean), max(diff.mean), length = 1000)
  points(x, dnorm(x, mean = mean(diff.mean), sd = sd(diff.mean))
       , type = "l", lwd = 2, col = "red")
  # place a rug of points under the plot
  rug(diff.mean)
```

```
  # restore par() settings
  par(old.par)
}
```