Lecture notes for
Advanced Data Analysis
ADA1 Stat 427/527 & ADA2 Stat 428/528

Department of Mathematics and Statistics
University of New Mexico

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

Fall 2016 – Spring 2017

Contents

Preface

UNM Stat 427/527: Advanced Data Analysis I (ADA1)

The course website (`https://statacumen.com/teaching/ada1`) includes these lecture notes and R code, video lectures, helper videos, quizzes, in-class assignments, homework assignments, datasets, and more course description including student learning outcomes, rubrics, and other helpful course information.

These notes include clicker questions which I used to use when I lectured from these notes. I found these, along with the think-pair-share strategy, very effective in assessing comprehension during the lecture.

Description Statistical tools for scientific research, including parametric and nonparametric methods for ANOVA and group comparisons, simple linear and multiple linear regression and basic ideas of experimental design and analysis. Emphasis placed on the use of statistical packages such as R. Course cannot be counted in the hours needed for graduate degrees in Mathematics and Statistics.

Goal Learn to produce beautiful (markdown) and reproducible (knitr) reports with informative plots (ggplot2) and tables (xtable) by writing code (R, Rstudio) to answer questions using fundamental statistical methods (all one- and two-variable methods), which youll be proud to present (poster).

Course structure The course semester structure enables students to develop the statistical, programming, visualization, and research skills to give a poster presentation. This includes conducting a literature review, developing a dataset codebook, cleaning data, performing univariate and bivariate statistal summaries, tests, and visualizations, and presenting results and evaluating peer presentations as part of a poster session. This course structure follows the GAISE recommendations.

Each week, the course structure is the following. Students prepare for class by reading these notes, watching video lectures of the notes, and taking a quiz online before class. In class, students download an Rmarkdown file and work through a real-data problem or two in teams to reinforce the content from the reading; sometimes, this includes taking data in class into a google spreadsheet and analyzing it immediately afterwards. Finally, students work on a homework assignment outside of class that is due the following week.

UNM Stat 428/528: Advanced Data Analysis II (ADA2)

The course website (`https://statacumen.com/teaching/ada2`) includes these lecture notes and R code, video lectures, helper videos, quizzes, in-class assignments, homework assignments, datasets, and more course description including student learning outcomes, rubrics, and other helpful course information.

Description A continuation of 427/527 that focuses on methods for analyzing multivariate data and categorical data. Topics include MANOVA, principal components, discriminant analysis, classification, factor analysis, analysis of contingency tables including log-linear models for multidimensional tables and logistic regression.

Goal Learn to produce beautiful (markdown) and reproducible (knitr) reports with informative plots (ggplot2) and tables (xtable) by writing code (R, Rstudio) to answer questions using fundamental statistical methods (analysis of covariance, logistic regression, and multivariate methods), which youll be proud to present (poster).

Course structure The course semester structure builds on ADA1 and enables students to develop the statistical, programming, visualization, and research skills to give a poster presentation. This includes multiple regression (with an emphasis on assessing model assumptions), logistic regression, multivariate methods, classification, and combining these methods. Visualization remains an important topic. The semester culminates in a poster session with many students using their own data. This course structure follows the GAISE recommendations.

Each week, the course structure is the following. Students prepare for class by reading these notes, watching video lectures of the notes, and taking a quiz online before class. In class, students download an Rmarkdown file and work through a real-data problem or two in teams to reinforce the content from the reading. We have used student-suggested or student-collected datasets when possible. Homework is started in class since the realistic data analysis problems can be quite involved and we want to resolve most student questions in class so that they don't get stuck on a small detail. In my experience, the students in the second semester have become quite mature in their attitudes toward data analysis, coding, and visualization; they are eager to be challenged and make well-reasoned decisions in their work.

Part I

ADA1: 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!

■	CLICKER Qs — Ch 0, R building blocks, Subset	■

■	CLICKER Qs — 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 in 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()
```

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

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 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
```

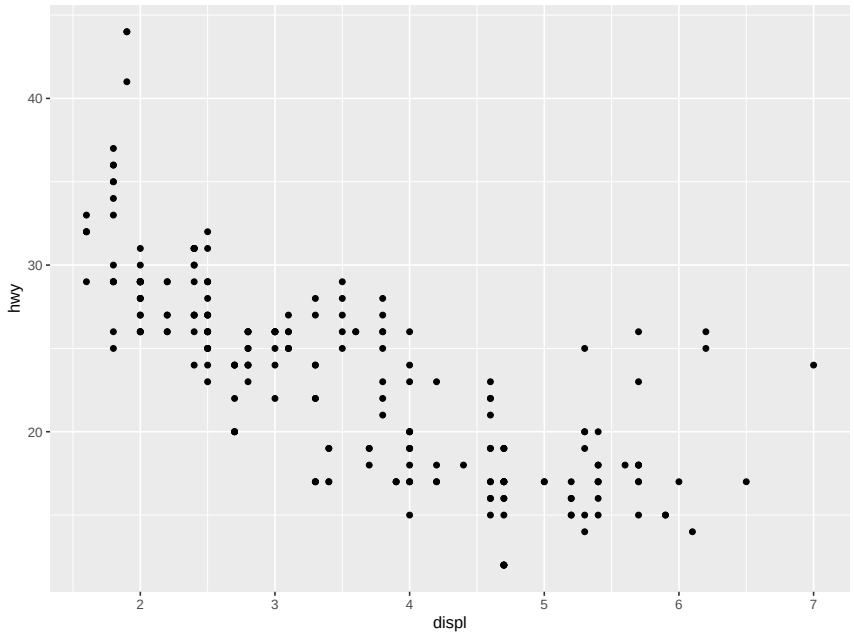
Let's get a look at the dataset we're using.

```
#### 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 fl
##   <chr>        <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr>
## 1 audi         a4      1.8  1999     4 auto~ f        18    29 p
## 2 audi         a4      1.8  1999     4 manu~ f        21    29 p
## 3 audi         a4      2    2008     4 manu~ f        20    31 p
## 4 audi         a4      2    2008     4 auto~ f        21    30 p
## 5 audi         a4      2.8  1999     6 auto~ f        16    26 p
```

```
## 6 audi         a4      2.8  1999     6 manu~ f        18    26 p
## # ... with 1 more variable: 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.

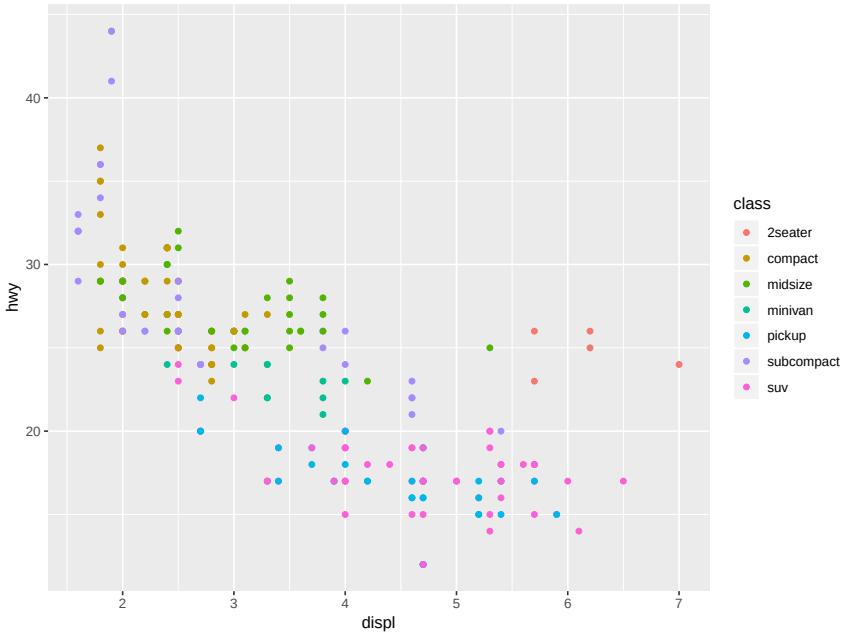
Geom: is the "type" of plot
Aesthetics: shape, colour, size, alpha
Faceting: "small multiples" displaying different subsets

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!

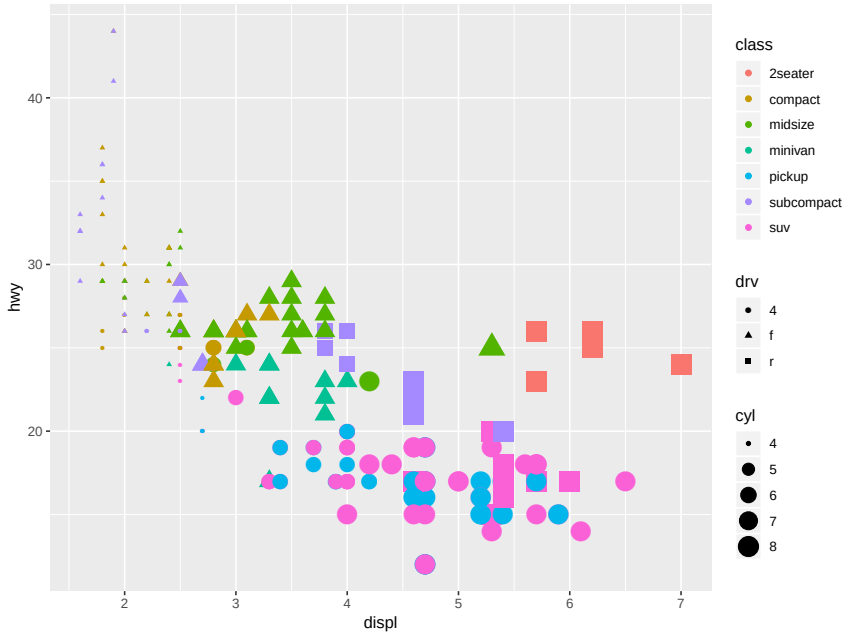
1. Assign variables to aesthetics colour, size, and shape.
2. What's the difference between discrete or continuous variables?
3. What happens when you combine multiple aesthetics?

The behavior of the aesthetics is predictable and customizable.

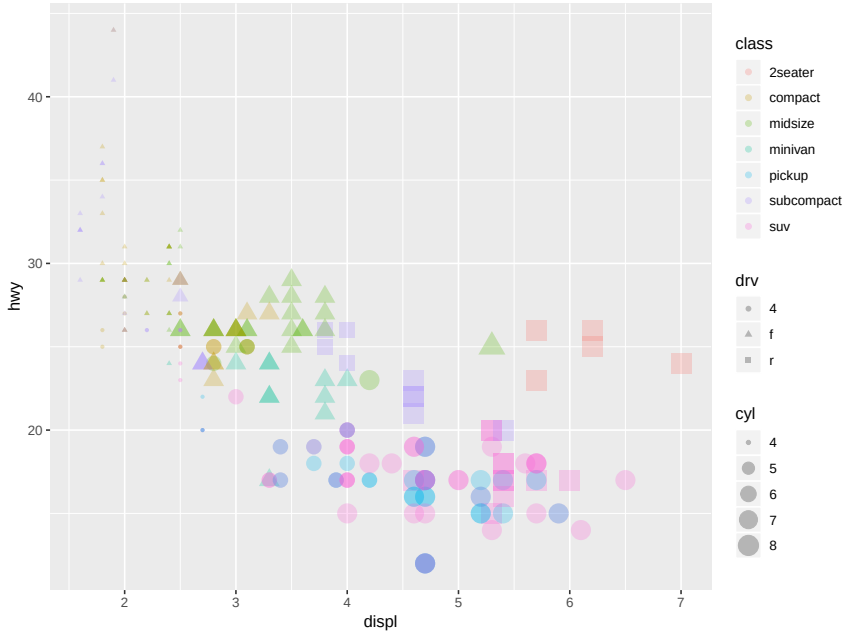
Aesthetic	Discrete	Continuous
colour	Rainbow of colors	Gradient from red to blue
size	Discrete size steps	Linear mapping between radius and value
shape	Different shape for each	Shouldn't work

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)
```



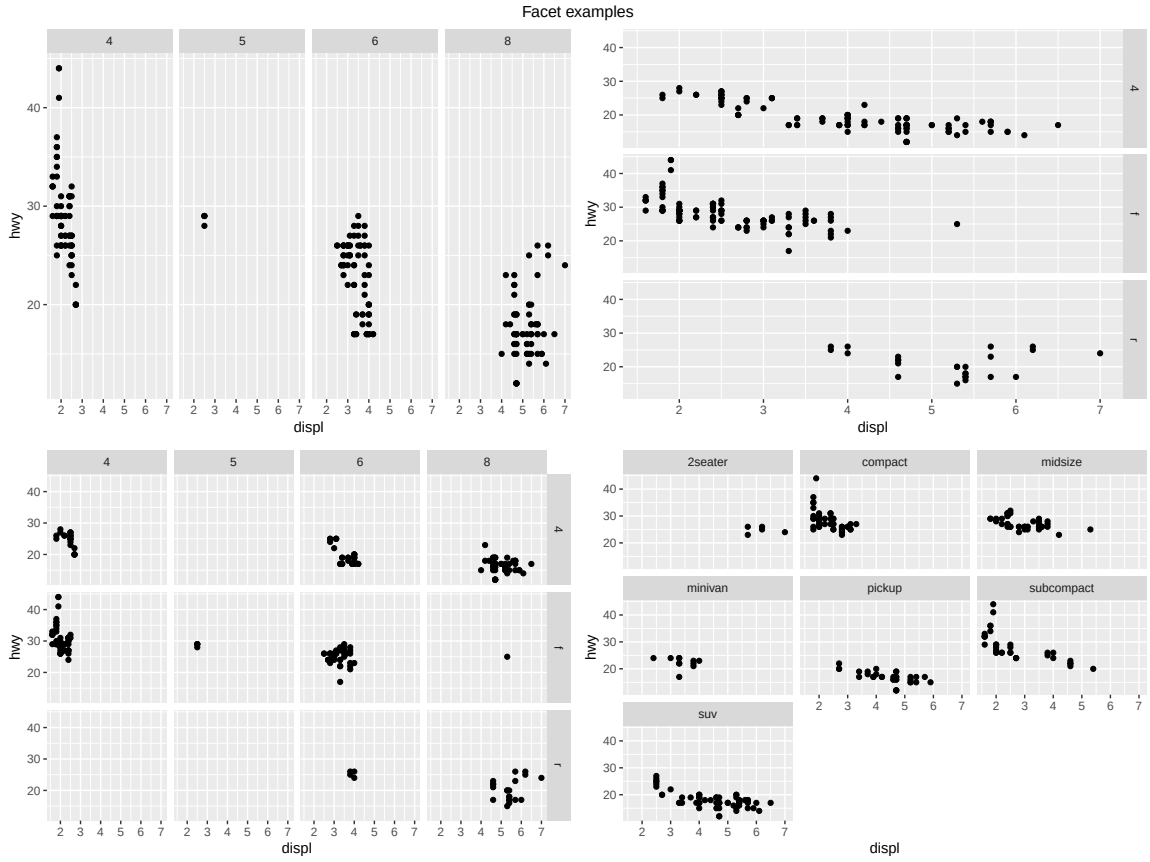
Faceting A small multiple[2] (sometimes called faceting, trellis chart, lattice chart, grid chart, or panel chart) is a series or grid of small similar graphics or charts, allowing them to be easily compared.

- 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")
```

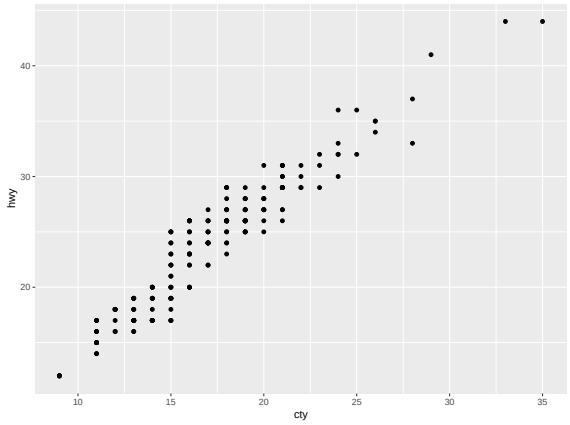
[2]According to Edward Tufte (Envisioning Information, p. 67): "At the heart of quantitative reasoning is a single question: Compared to what? Small multiple designs, multivariate and data bountiful, answer directly by visually enforcing comparisons of changes, of the differences among objects, of the 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?

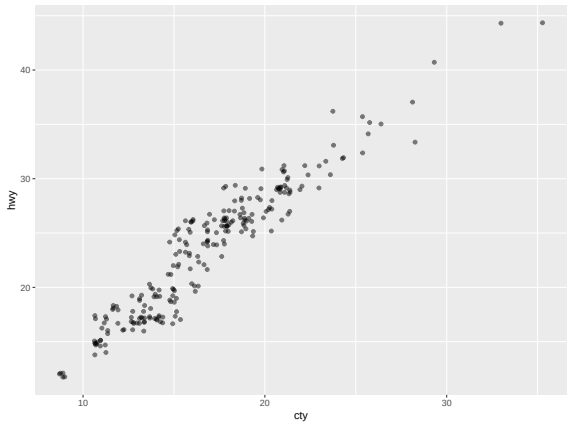
```
#### ggplot_mpg_cty_hwy
p <- ggplot(mpg, aes(x = cty, y = hwy))
p <- p + geom_point()
print(p)
```



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

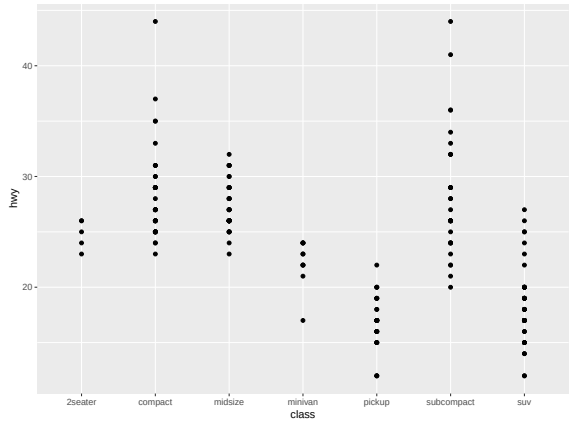
A solution: Jitter the points to reveal the individual points and reduce the opacity to 1/2 to indicate when points overlap.

```
#### 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?

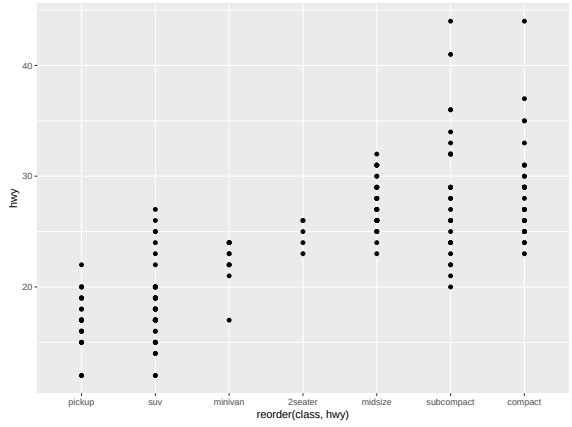
```
#### ggplot_mpg_class_hwy
p <- ggplot(mpg, aes(x = class, y = hwy))
p <- p + geom_point()
print(p)
```



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

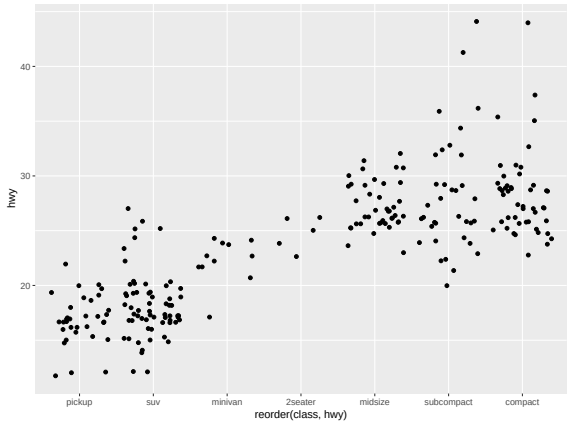
A solution: Reorder the class variable by the mean hwy for a meaningful ordering. Get help with `?reorder` to understand how this works.

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



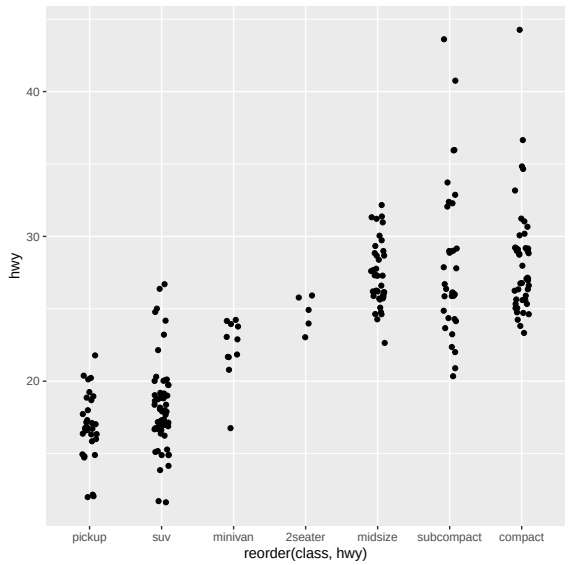
. . . add jitter

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



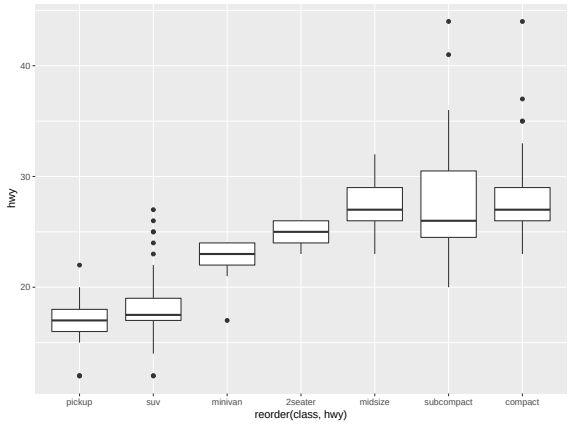
. . . a little less jitter

```
#### 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)
```



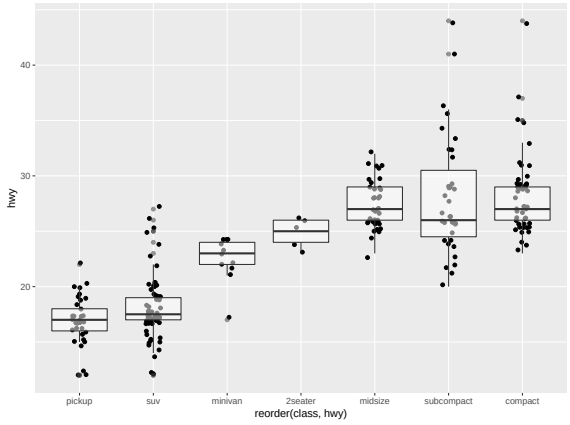
. . . or replace with boxplots

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



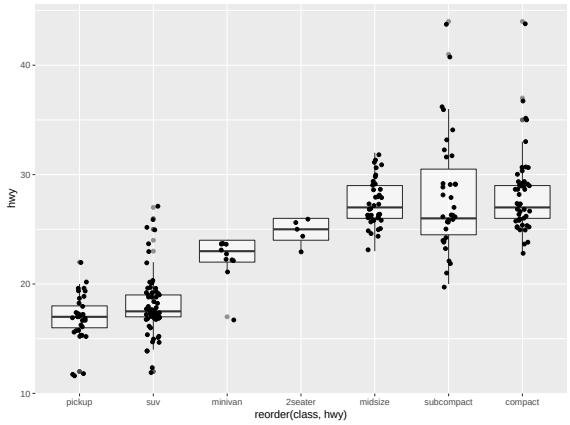
. . . or jitter those points with reduced-opacity boxplots on top

```
#### 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)
```



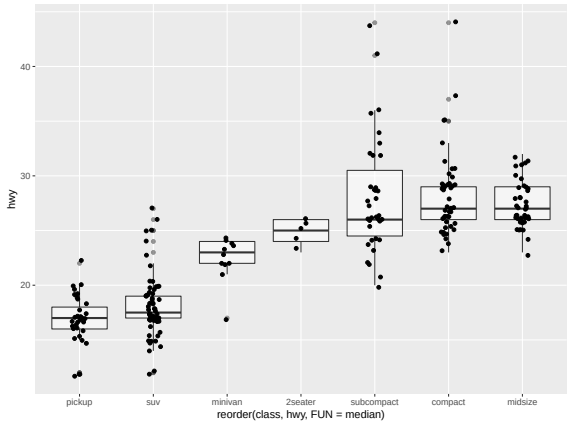
. . . even better with the boxplots with jittered points on top

```
#### 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

ADA1: Summaries and displays, and one-, two-, and many-way tests of means

Chapter 1

Summarizing and Displaying Data

Contents

Learning objectives

After completing this topic, you should be able to:

use R's functions to get help and numerically summarize data.
apply R's base graphics and ggplot to visually summarize data in several ways.
explain what each plotting option does.
describe the characteristics of a data distribution.

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

1. organize knowledge.
6. summarize data visually, numerically, and descriptively.
8. use statistical software.

1.1 Random variables

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

Qualitative data includes categorical outcomes:

Nominal Outcome is one of several categories.

Ex: Blood group, hair color.

Ordinal Outcome is one of several *ordered* categories.

Ex: Likert data such as (strongly agree, agree, neutral, disagree, strongly disagree).

Quantitative data includes numeric outcomes:

Discrete Outcome is one of a fixed set of numerical values.

Ex: Number of children.

Continuous Outcome is any numerical value.

Ex: Birthweight.

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

■ ClickerQs — Random variables ■

1.2 Numerical summaries

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

Among the numerical summary measures we're interested in are the **sample mean** $\bar{Y}$ and the **sample standard deviation** s. The sample mean is a measure of **central location**, or a measure of a "typical value" for the data set. The standard deviation is a measure of **spread** in the data set. These summary statistics

should be familiar to you. Let us consider a simple example to refresh your memory on how to compute them.

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

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

```
#### 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

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

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

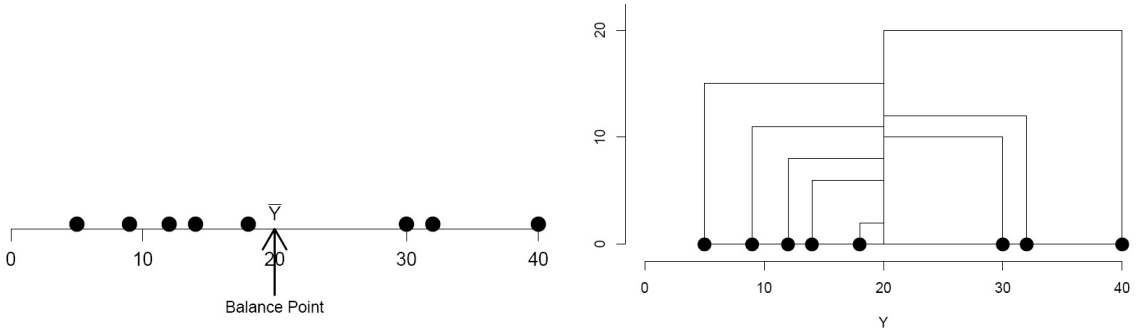
```
#### variance
var(y)
## [1] 156.2857
sd(y)
## [1] 12.50143
```

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.

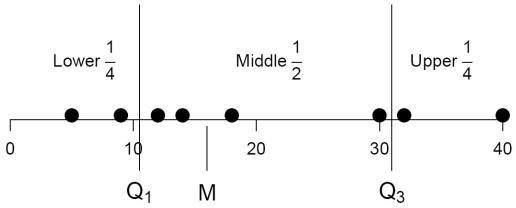
The interquartile range is

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

```
#### 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^{th} percentile, whereas the median and third quartiles are the 50^{th} and 75^{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:

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

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
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
```

[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.

```
# 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.

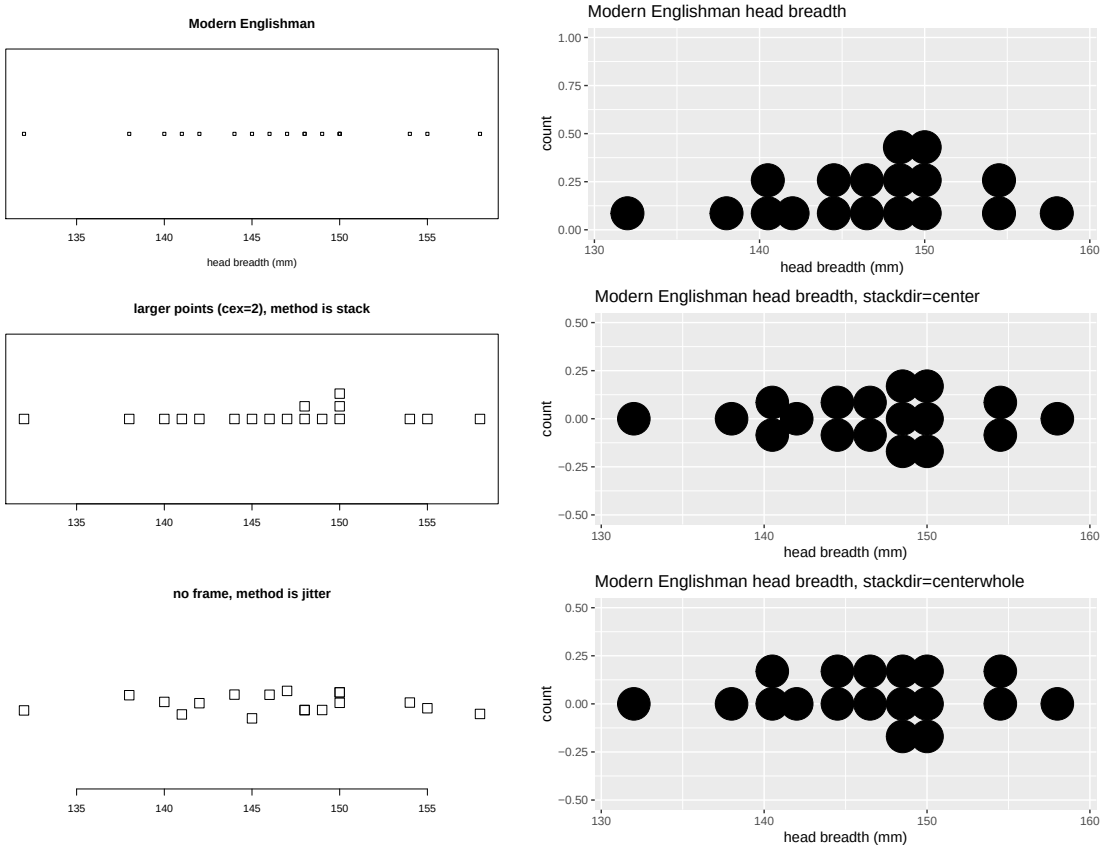
1.3.1 Dotplots

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)
```

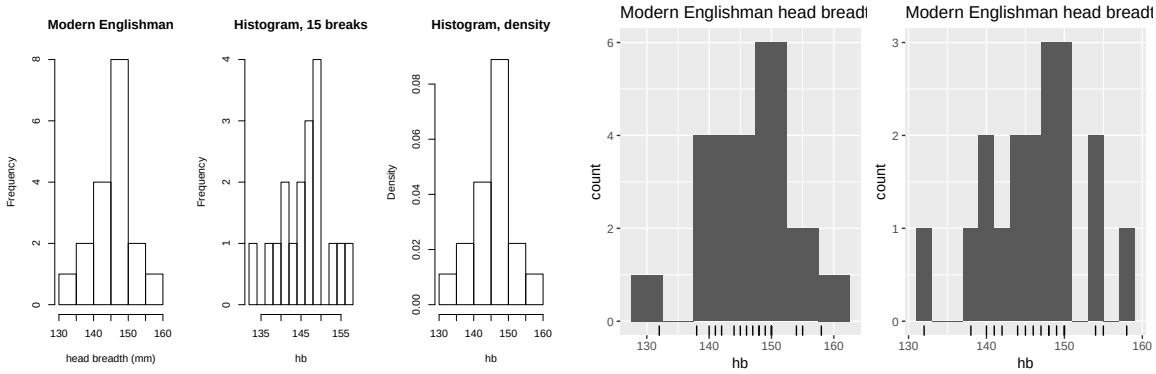
```r
# first put hb vector into a data.frame
hb_df <- data.frame(hb)
p1 <- ggplot(hb_df, aes(x = hb))
p1 <- p1 + geom_dotplot(binwidth = 2)
p1 <- p1 + labs(title = "Modern Englishman head breadth")
p1 <- p1 + xlab("head breadth (mm)")

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

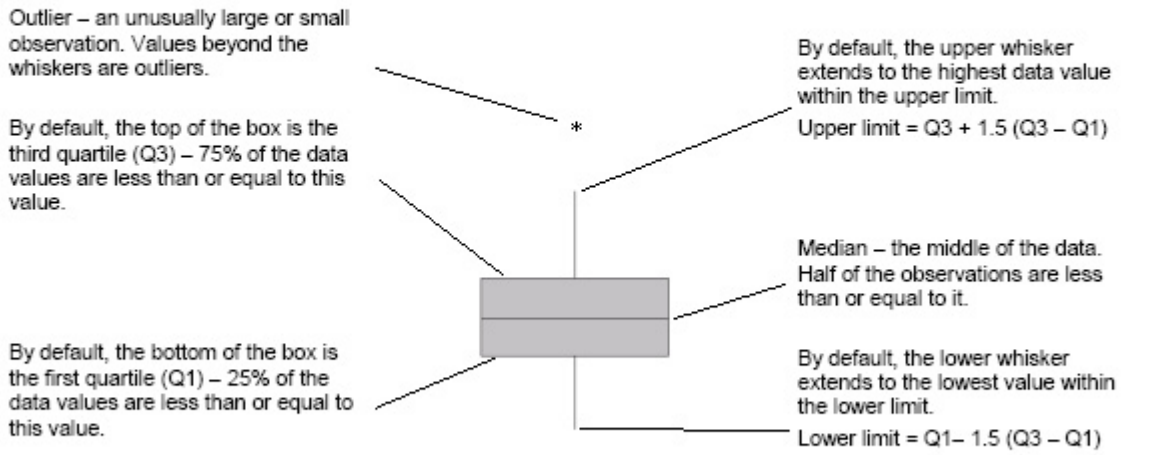
The **boxplot** breaks up the range of data values into regions about the center of the data, measured by the median. The boxplot highlights **outliers** and provides a visual means to assess "**normality**". The following help entry outlines the construction of the boxplot, given the placement of data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

Boxplots

Graph > Boxplot

Stat > EDA > Boxplot

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



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

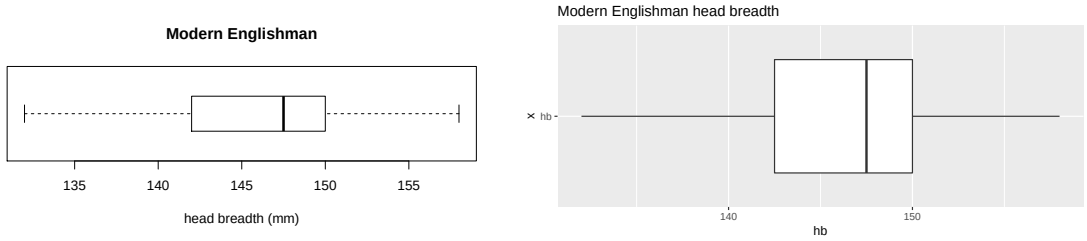
The endpoints of the box are placed at the locations of the first and third quartiles. The location of the median is identified by the line in the box. The whiskers extend to the data points closest to but not on or outside the outlier fences, which are $1.5IQR$ from the quartiles. Outliers are any values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

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

Note that the boxplots below are horizontal to better fit on the page. The

`horizontal=TRUE` and `coord_flip()` commands do this.

```
#### boxplot
fivenum(hb)
## [1] 132.0 142.0 147.5 150.0 158.0
# boxplot using R base graphics
par(mfrow=c(1,1))
boxplot(hb, horizontal=TRUE
      , main="Modern Englishman", xlab="head breadth (mm)")

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



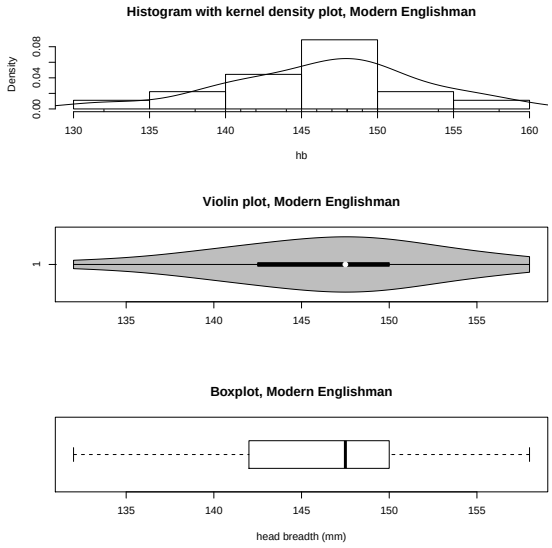
■ CLICKER Qs — Boxplots ■

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 2.2-5.5:  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)
```

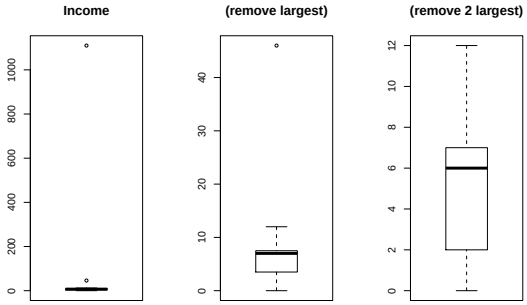
```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   0 | 022
##   0 | 557778
##   1 | 2
```

```
# 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 dis-

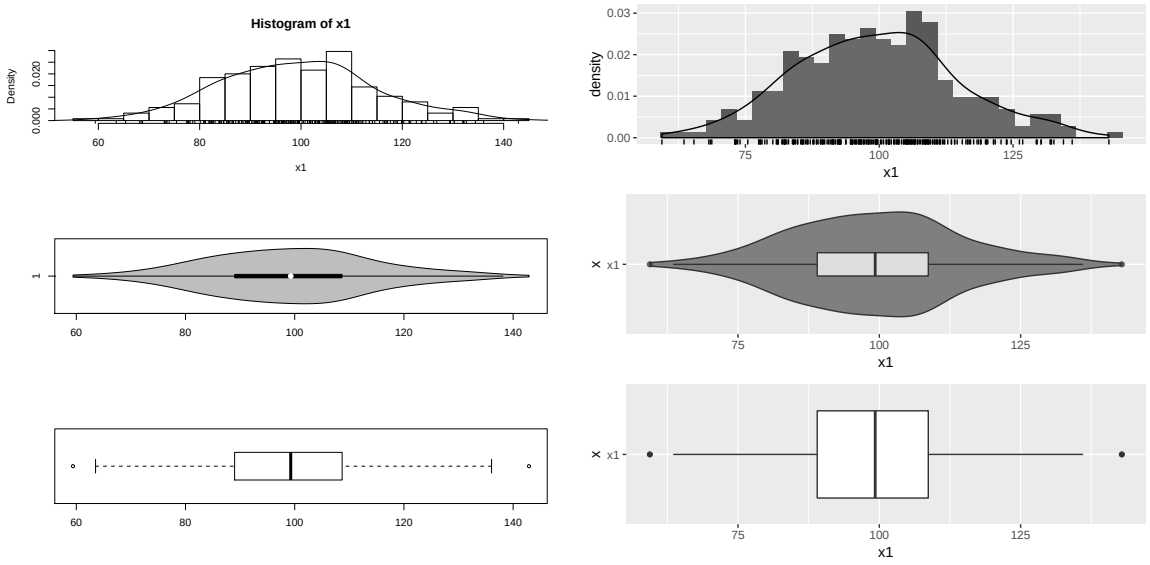
[2] "Distribution function" often refers to the "cumulative distribution function", which is a different (but one-to-one related) function than what I mean here.

play 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
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.
##   59.40   89.01   99.27   99.34  108.66  142.90
sd(x1)
## [1] 15.1007
skewness(x1)
## [1] 0.1432898
kurtosis(x1)
## [1] 2.883019
stem(x1)
```

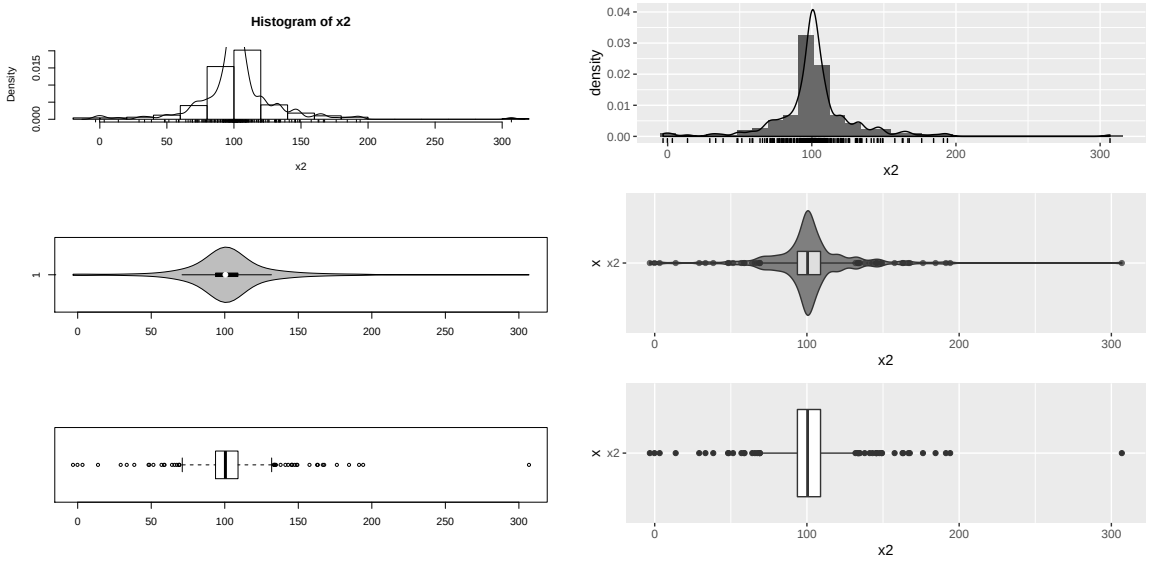
```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    5 | 9
##    6 | 4
##    6 | 5889
##    7 | 3333344
##    7 | 578888899
##    8 | 01111122222223344444
##    8 | 55555666667777888889999999
##    9 | 000111111122222233333344
##    9 | 555555555666666667777788888899999999
##   10 | 00000111222222233333344444
##   10 | 555555555666666667777777788888999999999
##   11 | 0000011111122233444
##   11 | 566677788999
##   12 | 00001123444
##   12 | 5679
##   13 | 00022234
##   13 | 6
##   14 | 3
```

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

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

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

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

# boxplot
boxplot(x2, horizontal=TRUE)
```

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

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

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

library(gridExtra)
grid.arrange(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.
##  -3.186  93.748 100.446 102.150 108.950 306.868
sd(x2)
## [1] 29.4546
```

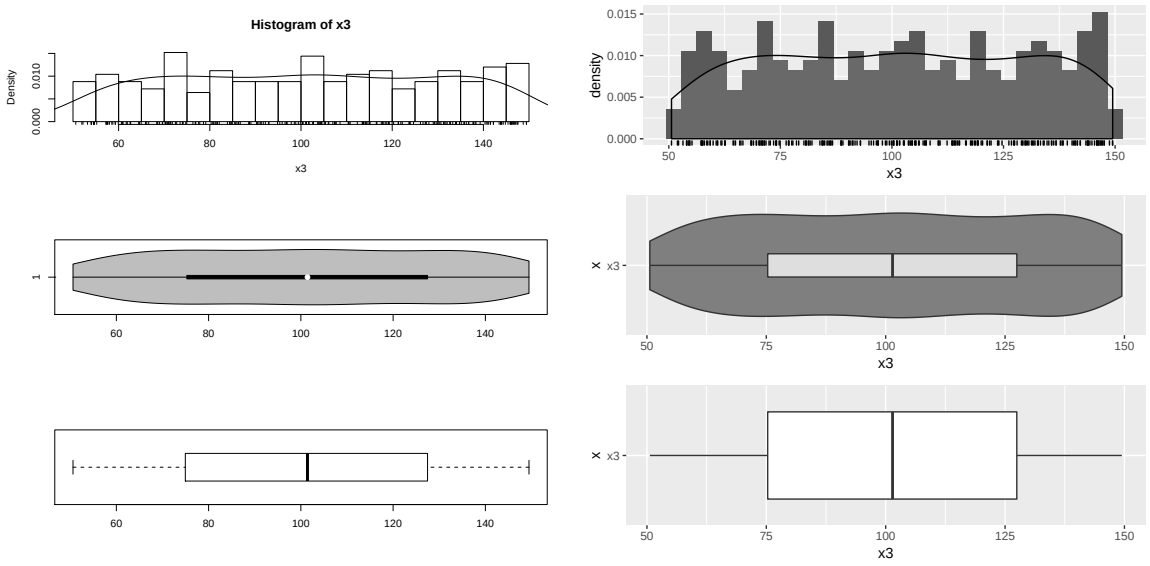
```
skewness(x2)
## [1] 1.124581
kurtosis(x2)
## [1] 13.88607
stem(x2)
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   -0 | 30
##    0 | 34
##    2 | 938
##    4 | 891799
##    6 | 4679991123334678899
##    8 | 00123334566788890012222223334444455556666677777888888999999999
##   10 | 000000000000000000000000000000011111111122222222222233333333444445+33
##   12 | 00000122245560112234445 8
##   14 | 135668998
##   16 | 33786
##   18 | 514
##   20 |
##   22 |
##   24 |
##   26 |
##   28 |
##   30 | 7
```

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

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

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

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

# boxplot
```

```
boxplot(x3, horizontal=TRUE)


# Histogram overlaid with kernel density curve
x3_df <- data.frame(x3)
p1 <- ggplot(x3_df, aes(x = x3))
  # Histogram with density instead of count on y-axis
p1 <- p1 + geom_histogram(aes(y=..density..))
p1 <- p1 + geom_density(alpha=0.1, fill="white")
p1 <- p1 + geom_rug()

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

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

library(gridExtra)
grid.arrange(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  101.44  101.31  127.46  149.46
sd(x3)
```

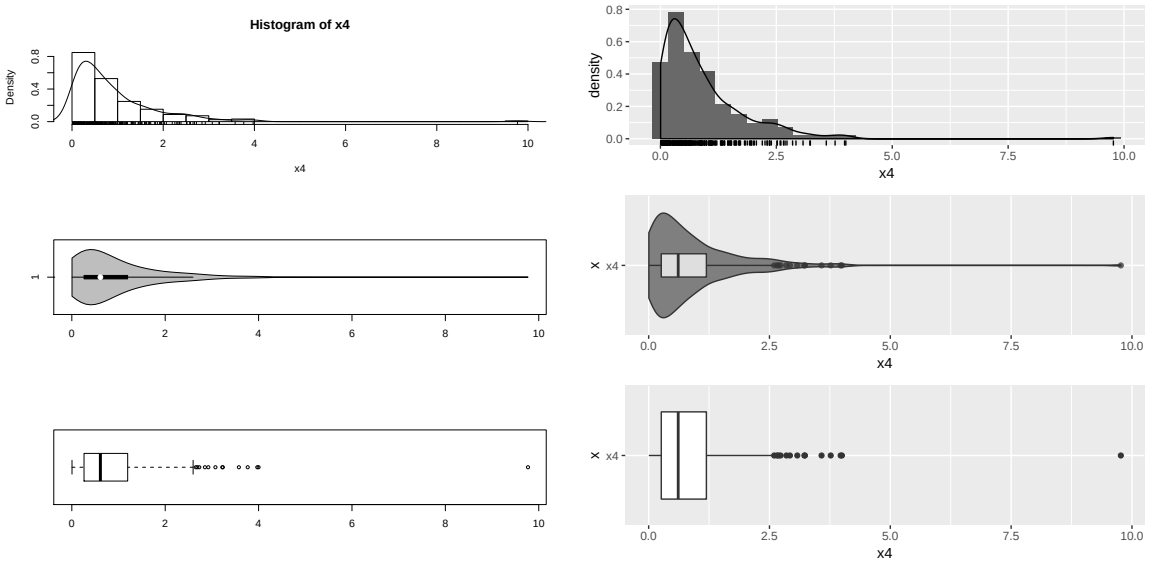
```
## [1] 29.02638
skewness(x3)
## [1] -0.00953667
kurtosis(x3)
## [1] 1.778113
stem(x3)
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    5 | 12234444
##    5 | 555577778889999
##    6 | 0111223334
##    6 | 556678899
##    7 | 0000011111122334444
##    7 | 5567778899
##    8 | 011111224444
##    8 | 5556666799999
##    9 | 0001112233
##    9 | 55667778999
##   10 | 00000111223334444
##   10 | 5555566777889
##   11 | 001123344444
##   11 | 55577888899999
##   12 | 001122444
##   12 | 5677778999
##   13 | 000011222333344
##   13 | 556667788889
##   14 | 01111222344444
##   14 | 55666666777788999
```

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

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

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

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

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

# boxplot
boxplot(x4, horizontal=TRUE)


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

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

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

library(gridExtra)
grid.arrange(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 0.908212 1.194486 9.769742
sd(x4)
## [1] 0.9963259
skewness(x4)
## [1] 3.596389
kurtosis(x4)
## [1] 27.45438
stem(x4)
##
##   The decimal point is at the |
##
##   0 | 000000000011111111111111111111112222222222222222222222222333333333+15
##   0 | 555555555555555555556666666666666677777777777777888888889999999999
##   1 | 00000001111111111122223333333444
##   1 | 5555555566666777788999
##   2 | 0000123333444
##   2 | 556677799
##   3 | 122
##   3 | 68
##   4 | 00
##   4 |
##   5 |
##   5 |
##   6 |
##   6 |
##   7 |
```

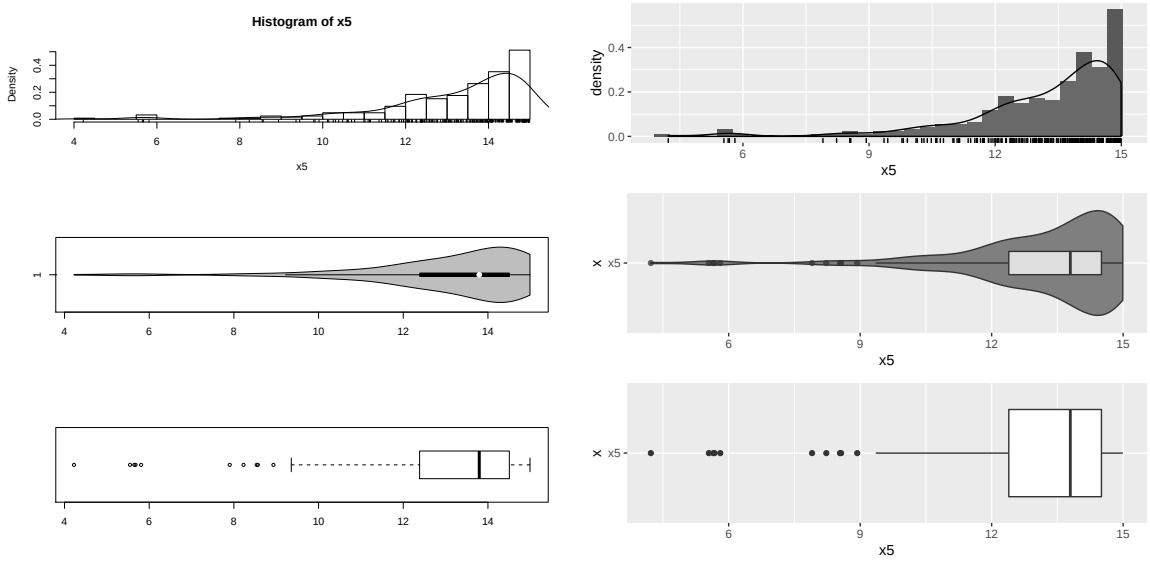
```
##   7 |
##   8 |
##   8 |
##   9 |
##   9 | 8
```

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

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

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

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

# boxplot
boxplot(x5, horizontal=TRUE)


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

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

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

```
library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3), ncol=1)
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```

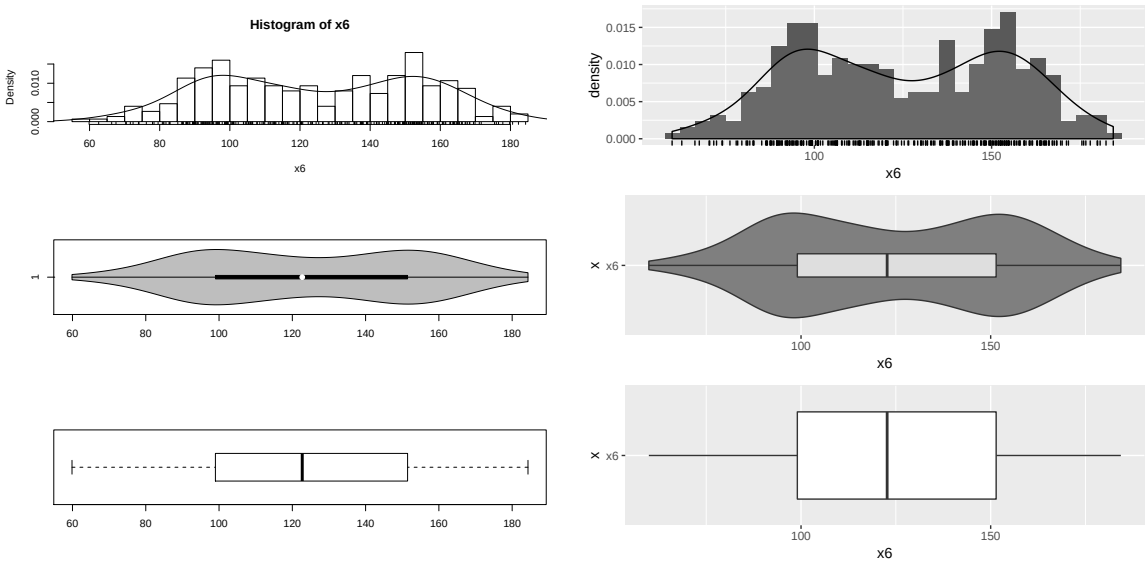



```
summary(x5)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.224  12.391  13.795  13.183  14.506  14.994
sd(x5)
## [1] 1.870509
skewness(x5)
## [1] -1.961229
kurtosis(x5)
## [1] 7.888622
stem(x5)
##
##   The decimal point is at the |
##
##    4 | 2
##    5 | 5678
##    6 |
##    7 | 9
##    8 | 2569
##    9 | 44889
##   10 | 11334566678
##   11 | 001124456677778899
##   12 | 000111122333444444444455666667788889999
##   13 | 000000111122222233334455566666666777777888888999999
##   14 | 00000000001111111111222222233333333334444445555555555555666667777777+25
##   15 | 000000000
```

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

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

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

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

# boxplot
boxplot(x6, horizontal=TRUE)


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

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

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

library(gridExtra)
grid.arrange(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.
##   59.87   99.03  122.71  124.71  151.45  184.32
sd(x6)
## [1] 29.59037
skewness(x6)
## [1] -0.005817938
kurtosis(x6)
## [1] 1.85249
stem(x6)
##
##   The decimal point is 1 digit(s) to the right of the |
##
##    5 |
##    6 | 0368
##    7 | 002244688
##    8 | 0111233356677788889999
##    9 | 000001112222223333444455555666677789999999999999
##   10 | 00111111224445556666666678899
##   11 | 000112223333345555555666678899
##   12 | 00000001123345567889
##   13 | 000111223344566667777888899999
##   14 | 00122233455556677788888999
##   15 | 000000011112222233333444444555556667778889999
##   16 | 011112333444444555556667788889
##   17 | 0125678
##   18 | 00124
```

The boxplot and histogram or stem-and-leaf display (or dotplot) are used **to-**

gether 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

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.

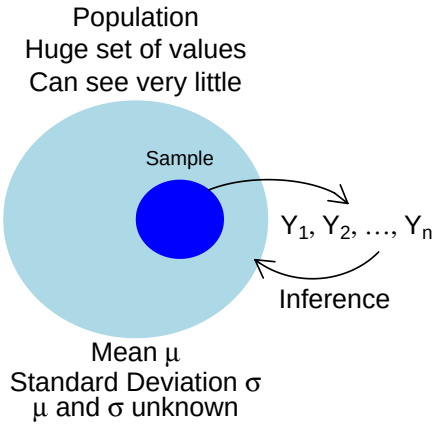
Achieving these goals contributes to mastery in these course learning outcomes:

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 measurements 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.

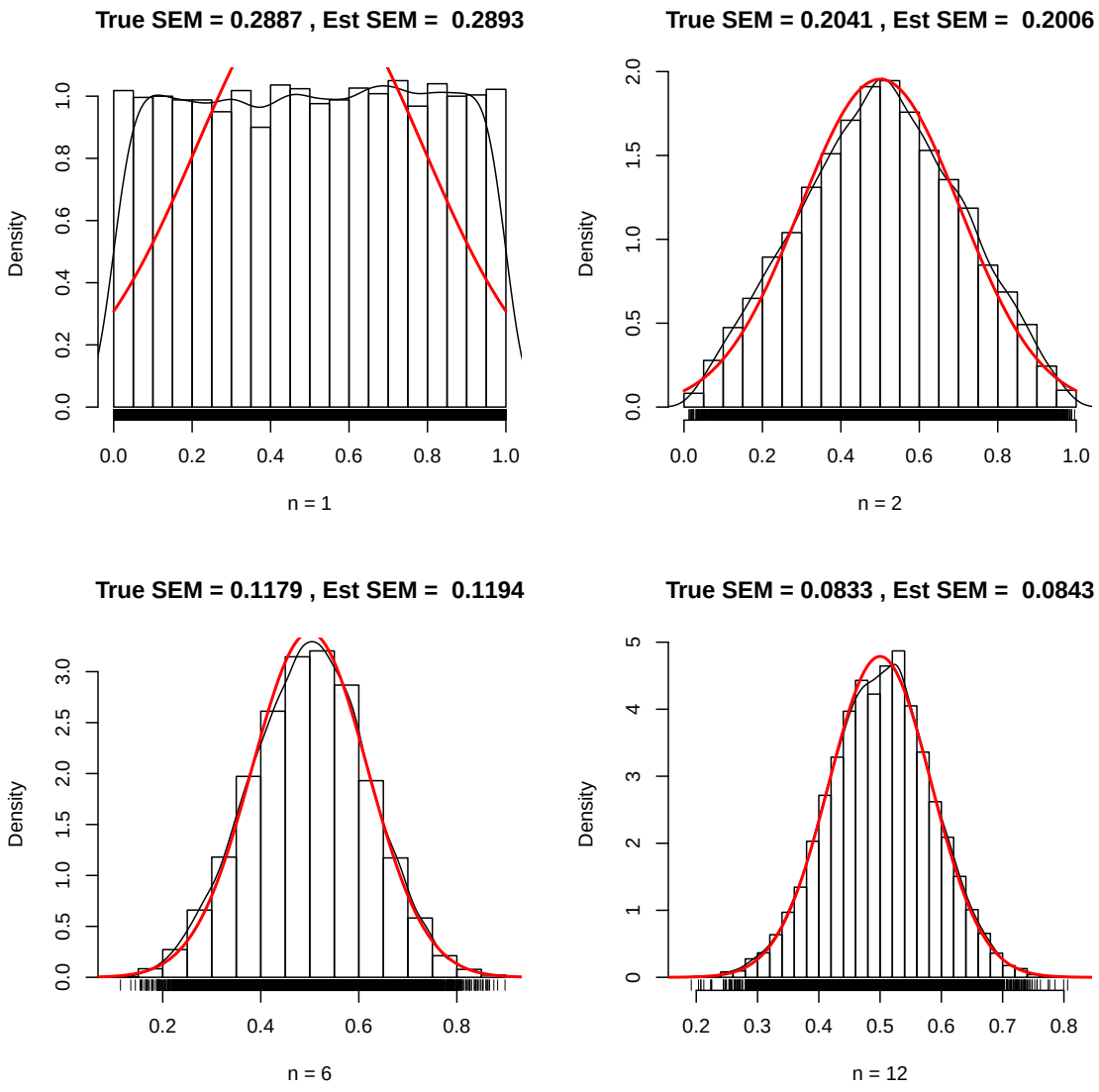
In probability theory, the **central limit theorem (CLT)** states that, given certain conditions, the mean of a sufficiently large number of independent random variables, each with finite mean and variance, will be approximately normally distributed[1].

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 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
```

[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.

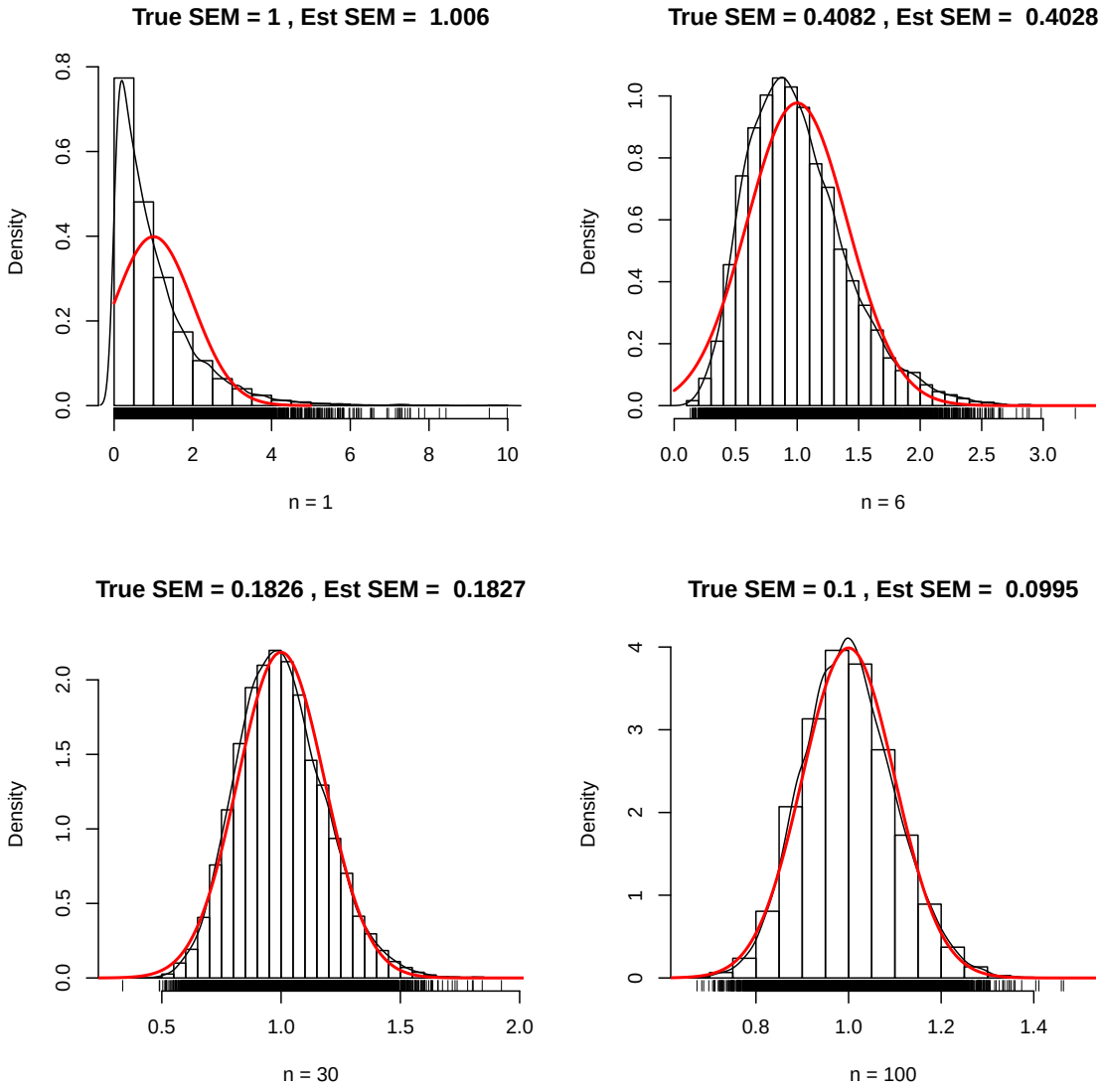
```
  rug(sam.mean)
}

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:

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

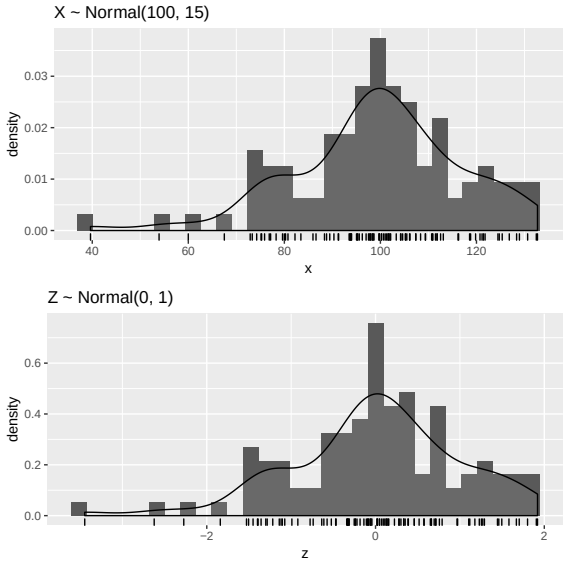
Below, the original variable x has a normal distribution with mean 100 and standard deviation 15, Normal$(100, 15^2)$, and z has a Normal$(0, 1)$ distribution.

```
# 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)
```

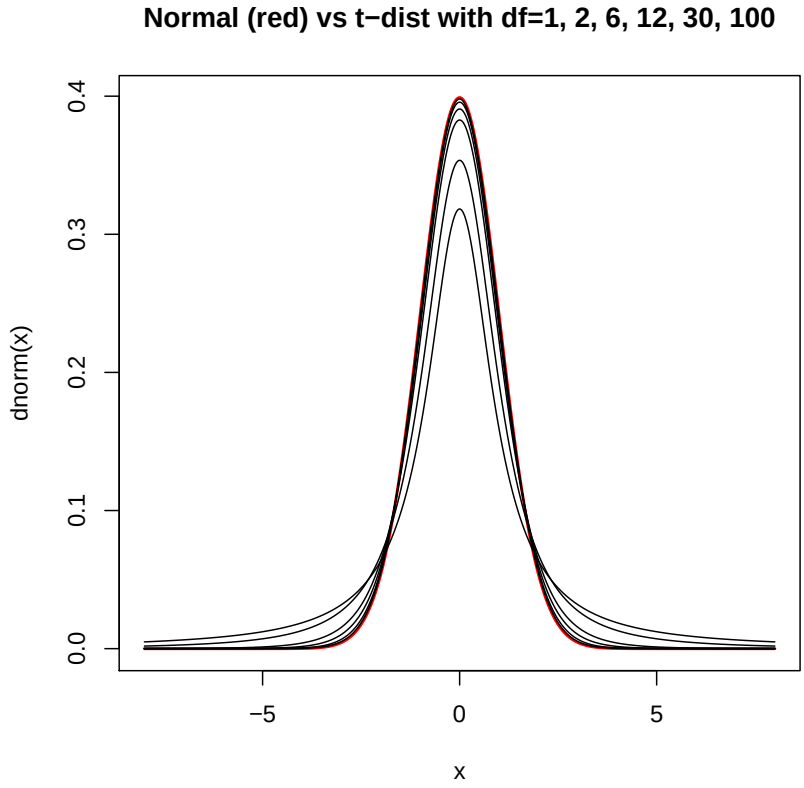
```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```



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.

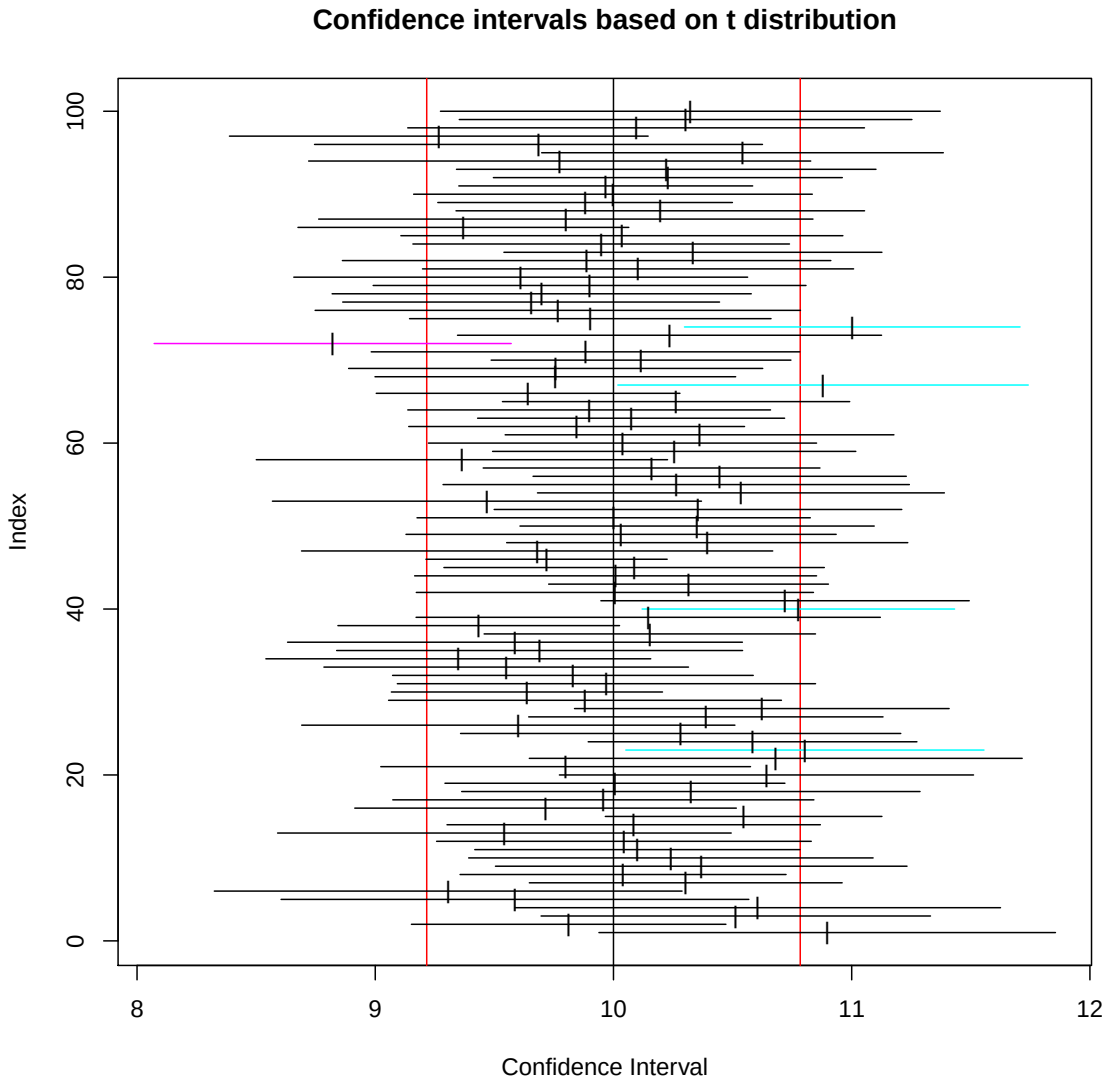
The length of the CI

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

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.

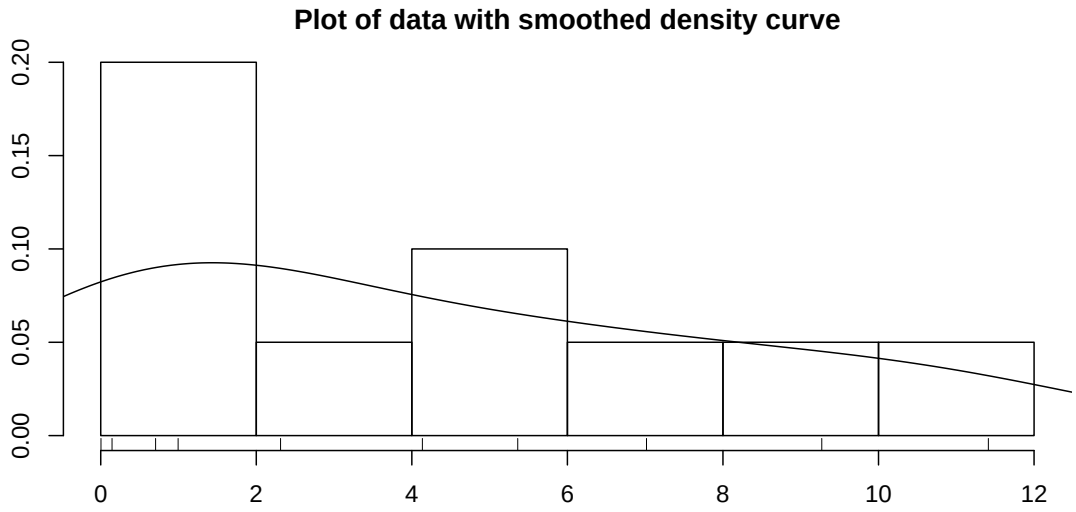
In fact, the assumptions are slightly looser than this, the population frequency curve can be anything provided the sample size is large enough that it's reasonable to assume that the **sampling distribution of the mean is normal**.

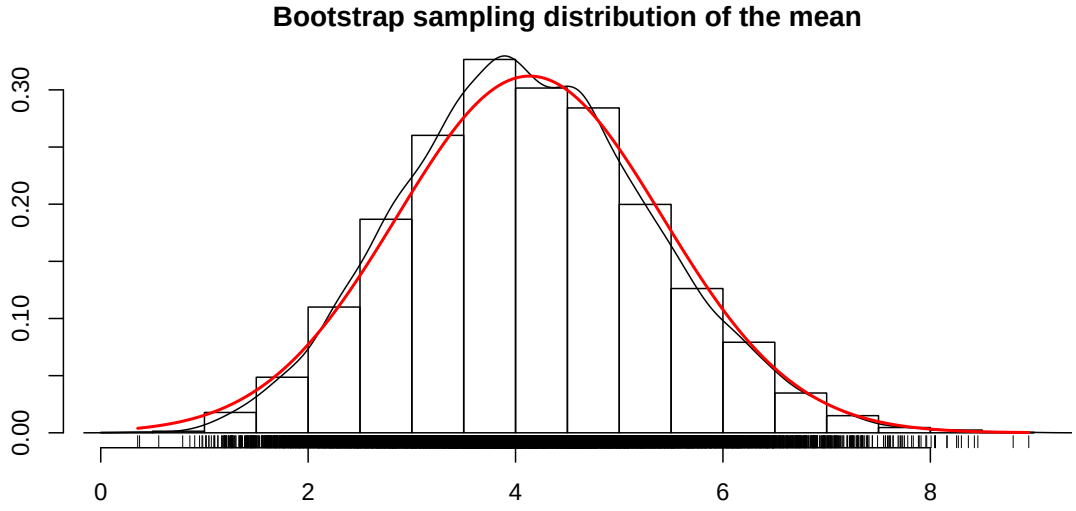
Assessing assumptions using the bootstrap

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.

■ CLICKER Qs — 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 $H_A : \mu \neq \mu_0$.

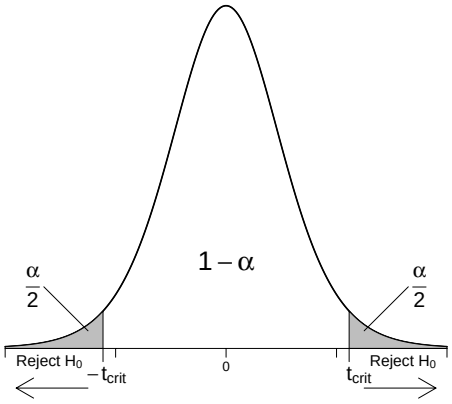
I will explore the ideas behind hypothesis testing later. At this point, I focus on the mechanics behind the test. The steps in carrying out the test are:

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).

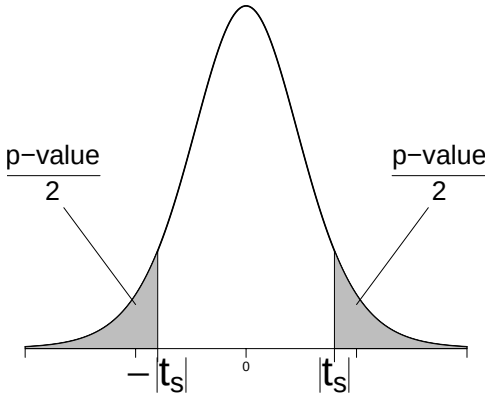
The process is represented graphically below. The area under the t-probability curve outside $\pm t_{\text{crit}}$ is the size of the test, α. One-half α is the area in each tail. You reject H_0 in favor of H_A only if the test statistic is outside $\pm t_{\text{crit}}$.



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.

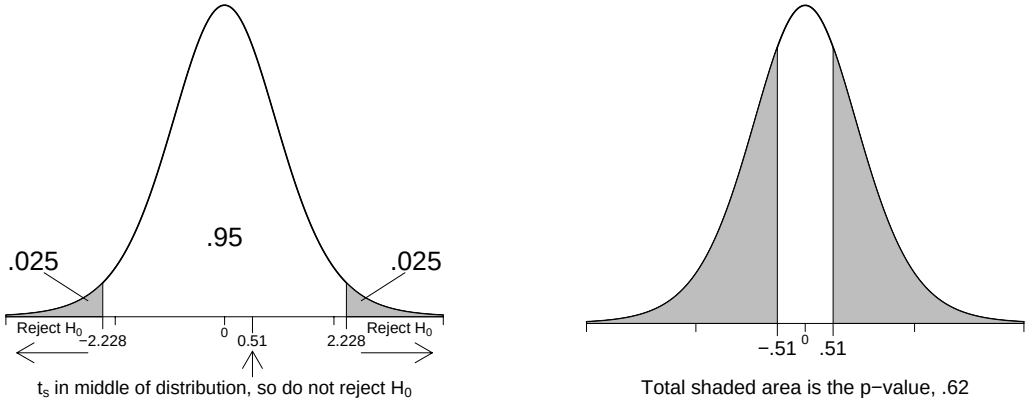
We are interested in testing $H_0 : \mu = 50$ against $H_A : \mu \neq 50$, so $\mu_0 = 50$.

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.

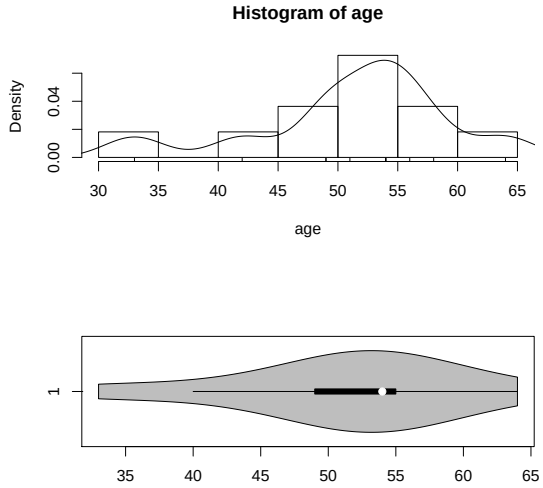


Example: Age at First Transplant R output for the heart transplant problem is given below. Let us look at the output and find all of the summaries we computed. Also, look at the graphical summaries to assess whether the t-test and CI are reasonable here.

```
#### 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
```

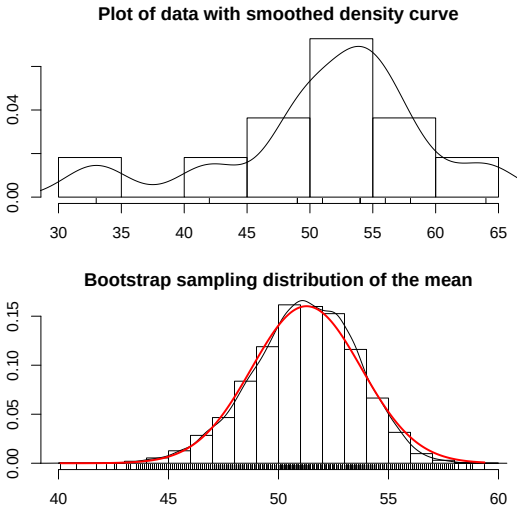
```
# look at help for t.test
?t.test
  # defaults include: alternative = "two.sided", conf.level = 0.95
t.summary <- t.test(age, mu = 50)
t.summary
```

```
##
##  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.

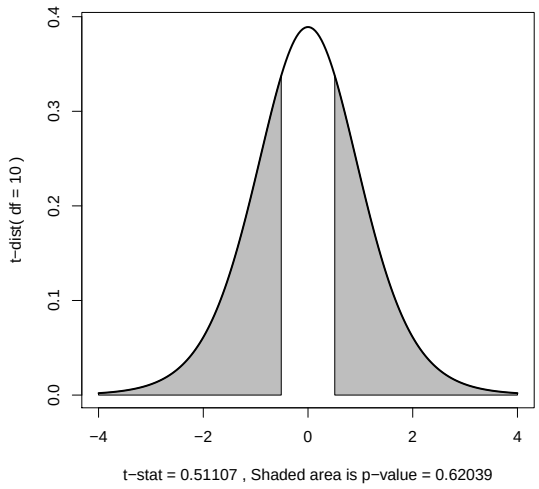
```
bs.one.samp.dist(age)
```



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

$$\mu = \text{mean cooling rate over all pieces of the Tocopilla meteorite.}$$

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 $SE_{\bar{Y}} = s/\sqrt{n} = 0.457$.

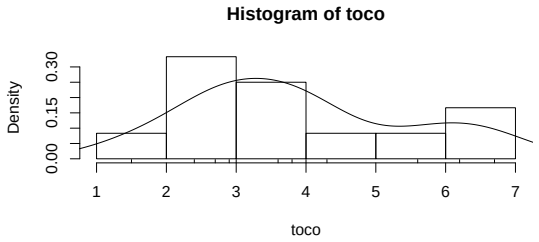
R output for this problem is given below. For a 5% test (i.e., $\alpha = 0.05$), you would reject H_0 in favor of H_A because the p-value ≤ 0.05. The data strongly suggest that $\mu \neq 0.54$. The 95% CI says that you are 95% confident that the population mean cooling rate for the Tocopilla meteorite 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.

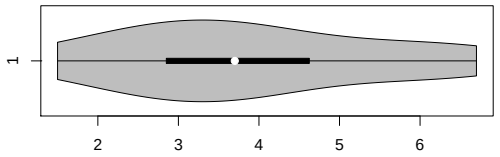
[2] `http://www.lpi.usra.edu/meteor/metbull.php?code=24204`
[3] `http://www.lpi.usra.edu/meteor/metbull.php?code=24138`
[4] `http://www.lpi.usra.edu/meteor/metbull.php?code=17001`

```
#### 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
```

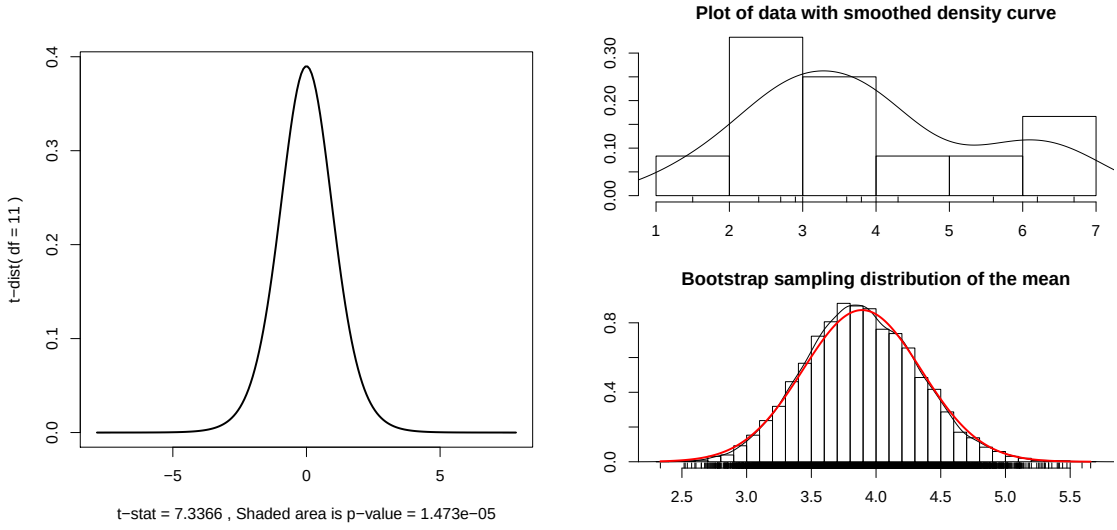
```
##
##  One Sample t-test
##
## data:  toco
## t = 7.3366, df = 11, p-value = 1.473e-05
## alternative hypothesis: true mean is not equal to 0.54
## 95 percent confidence interval:
##  2.886161 4.897172
## sample estimates:
## mean of x
##  3.891667
```

```
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 population 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.

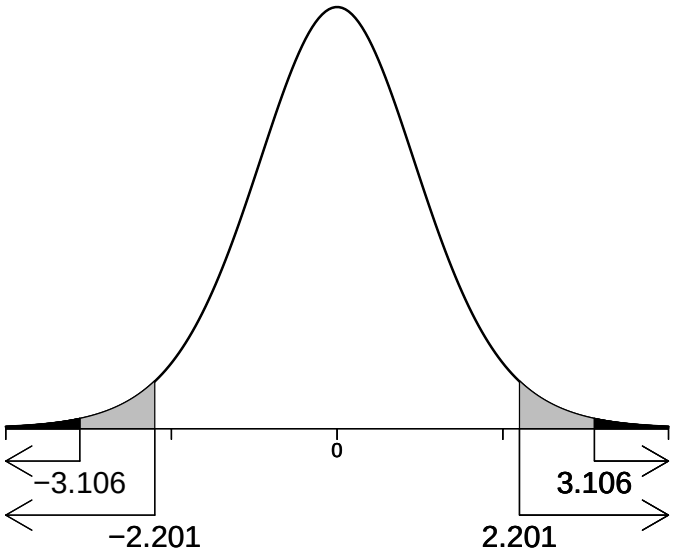
Let us piece together these ideas for the meteorite problem. Evolutionary history predicts $\mu = 0.54$. A scientist examining the validity of the theory is trying to decide whether $\mu = 0.54$ or $\mu \neq 0.54$. Good scientific practice dictates that rejecting another's claim when it is true is more 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.

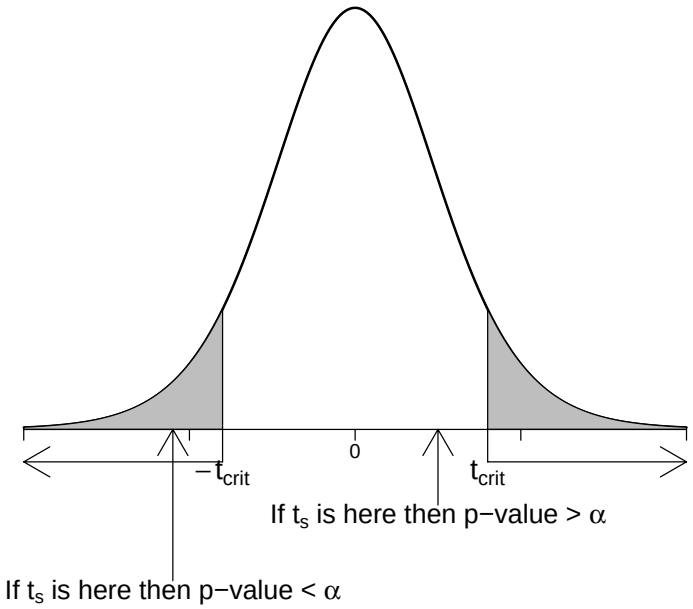
2.4 Two-sided tests, CI and p-values

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

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

size α test fails to reject $H_0 \Leftrightarrow 100(1-\alpha)\%$ CI contains $\mu_0 \Leftrightarrow$ 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.

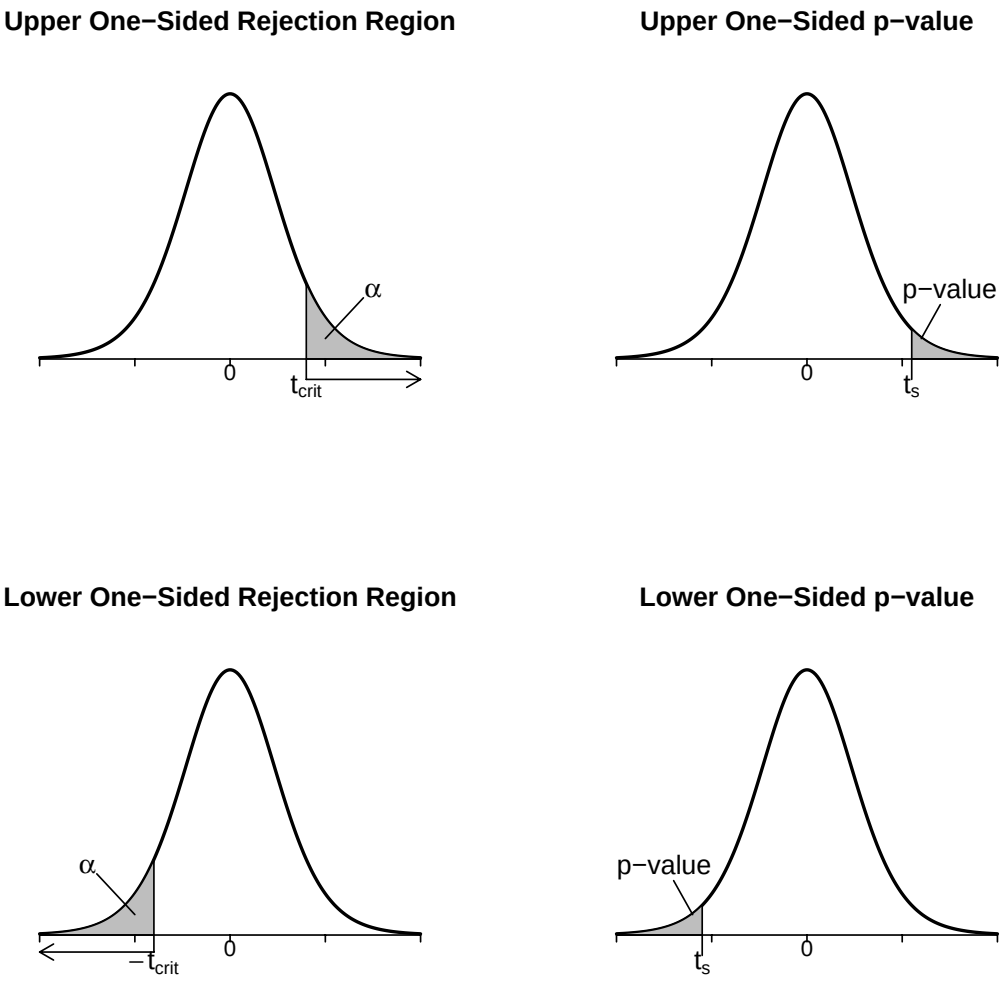
The **upper one-sided test** uses the **upper tail** of the t-distribution for a rejection region. The p-value calculation reflects the form of the rejection region. You will reject H_0 only for large positive values of t_s which require $\bar{Y}$ to be significantly greater than μ_0. Does this make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

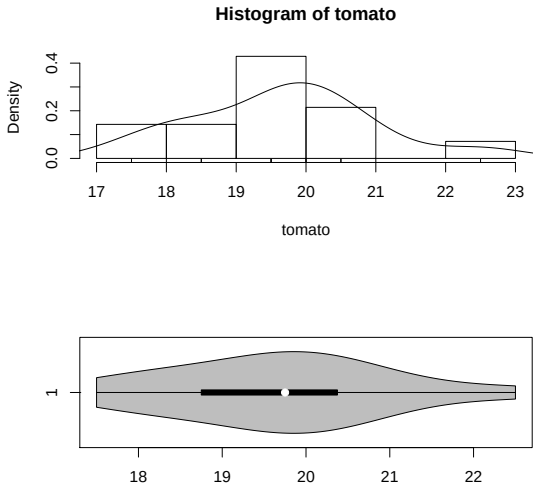
$$t_s = \frac{\bar{Y} - \mu_0}{SE_{\bar{Y}}} = \frac{19.679 - 20}{0.346} = -0.93.$$

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?

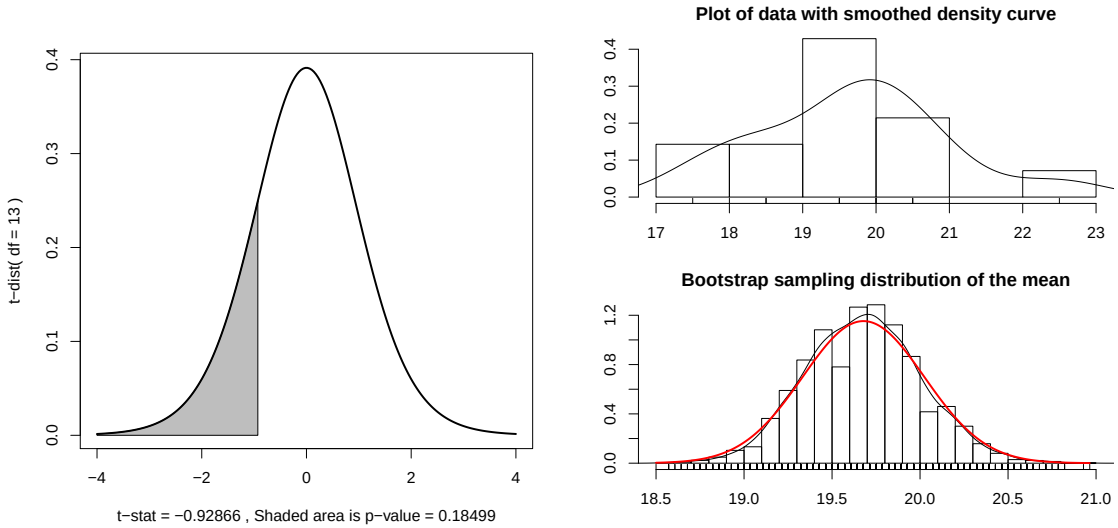
```
#### Example: Weights of canned tomatoes
tomato <- c(20.5, 18.5, 20.0, 19.5, 19.5, 21.0, 17.5
          , 22.5, 20.0, 19.5, 18.5, 20.0, 18.0, 20.5)

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

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.

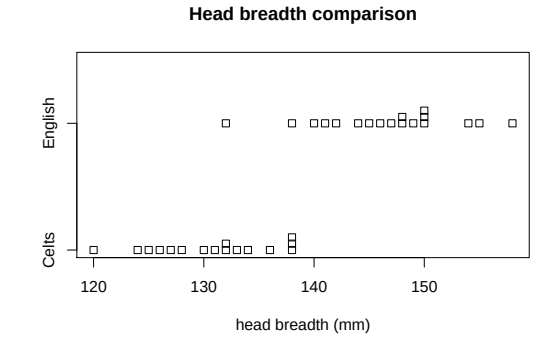
```
# stacked data as a vector of values and a vector of labels
HeadBreadth <- c(english, celts)
Group <- c(rep("English", length(english)), rep("Celts", length(celts)))
hb <- data.frame(HeadBreadth, Group)
hb
```

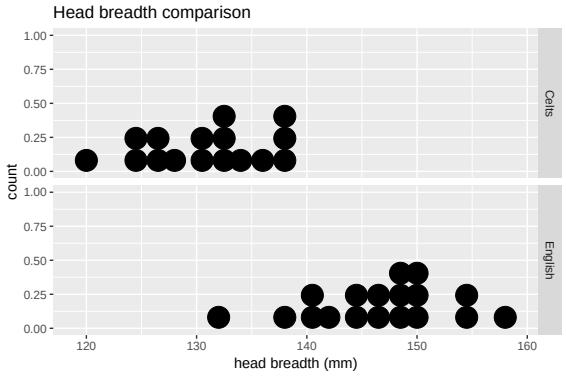
```
##    HeadBreadth   Group
## 1          141 English
## 2          148 English
## 3          132 English
## 4          138 English
## 5          154 English
## 6          142 English
## 7          150 English
## 8          146 English
## 9          155 English
## 10         158 English
## 11         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.

```r
#### 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)
```

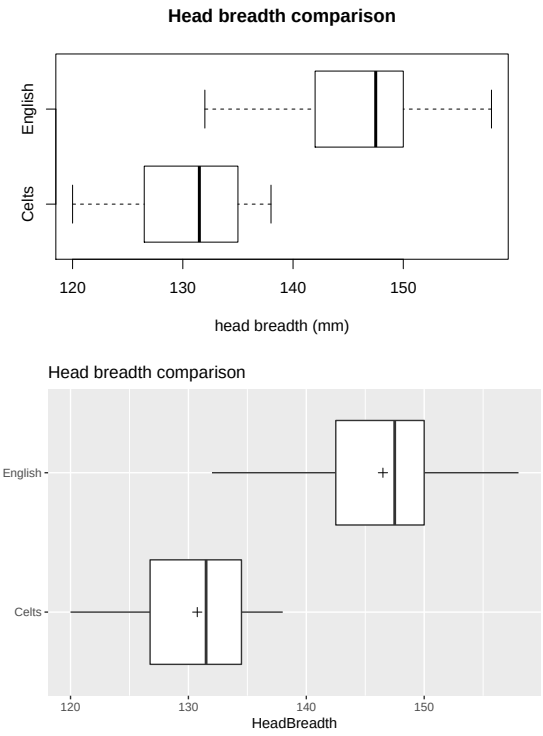




2. Boxplots for comparison are most helpful when plotted in the same axes.

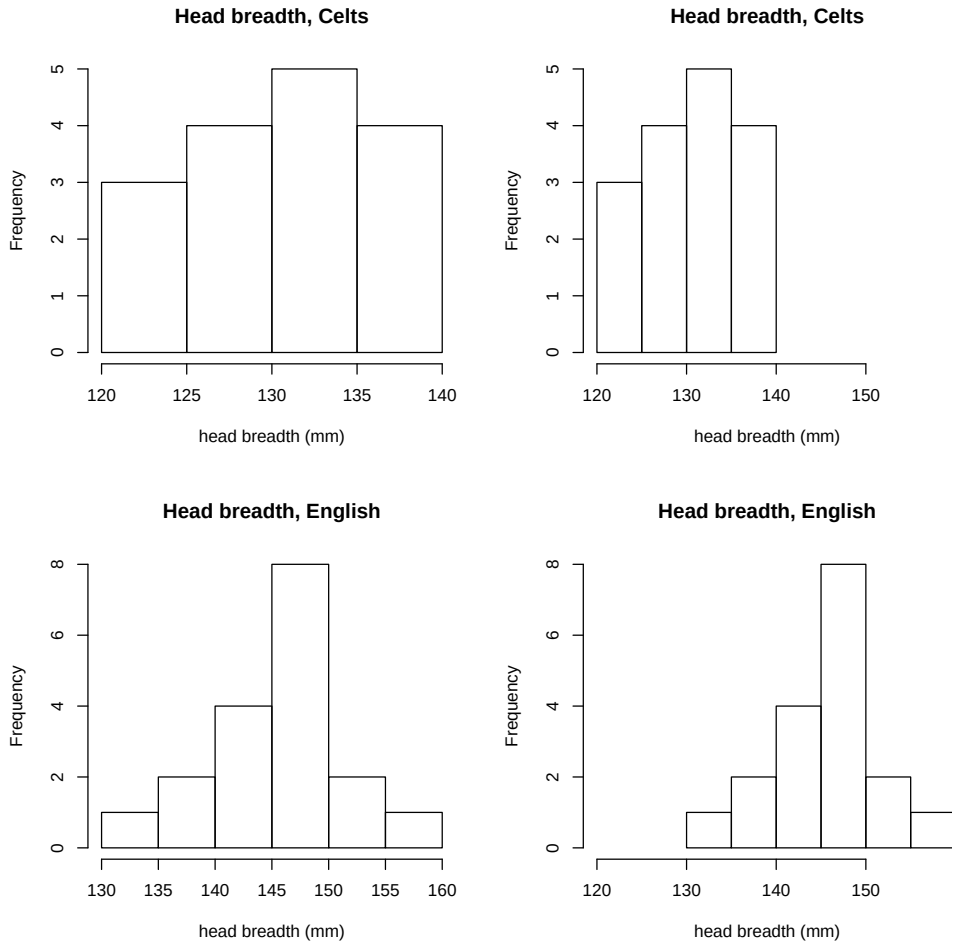
```r
# boxplot using R base graphics
boxplot(HeadBreadth ~ Group, method = "stack", data = hb,
  horizontal = TRUE,
  main = "Head breadth comparison", xlab = "head breadth (mm)")


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)
```

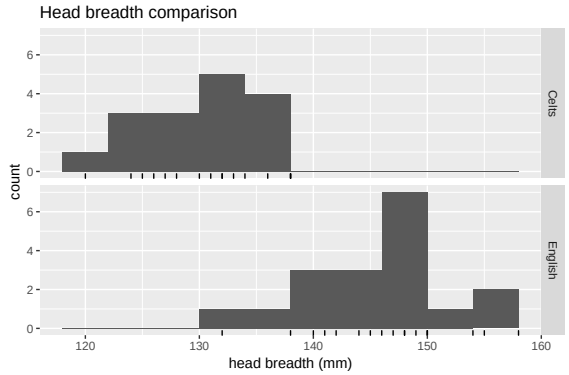


3. Histograms are hard to compare unless you make the scale and actual bins the same for both. Why is the pair on the right clearly preferable?

```
# 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)")
```




```r
# 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
```

```r
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.

```r
#### 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
```

```r
summary(celts)
```

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

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

```
## [1] 6.382421
```

```r
sd(celts)
```

```
## [1] 5.434458
```

```r
IQR(english)
```

```
## [1] 7.5
```

```r
IQR(celts)
```

```
## [1] 7.75
```

```r
# 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):

```r
# 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 consistent 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.

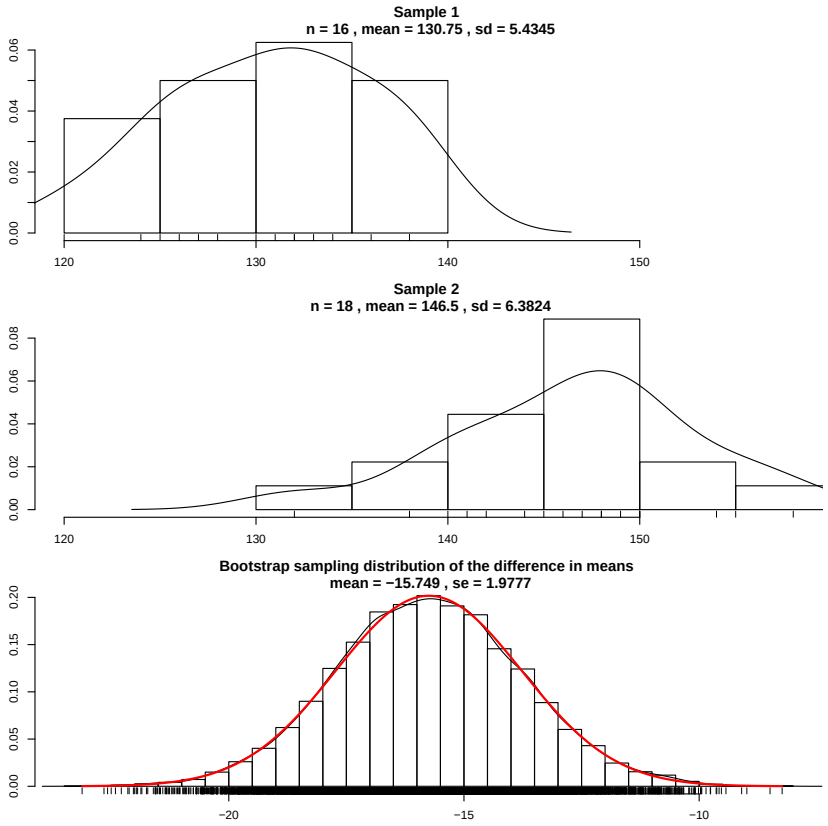
We can test the assumption that the distribution of $\bar{Y}_1 - \bar{Y}_2$ is normal using the bootstrap in the following function.

```r
#### 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,
```

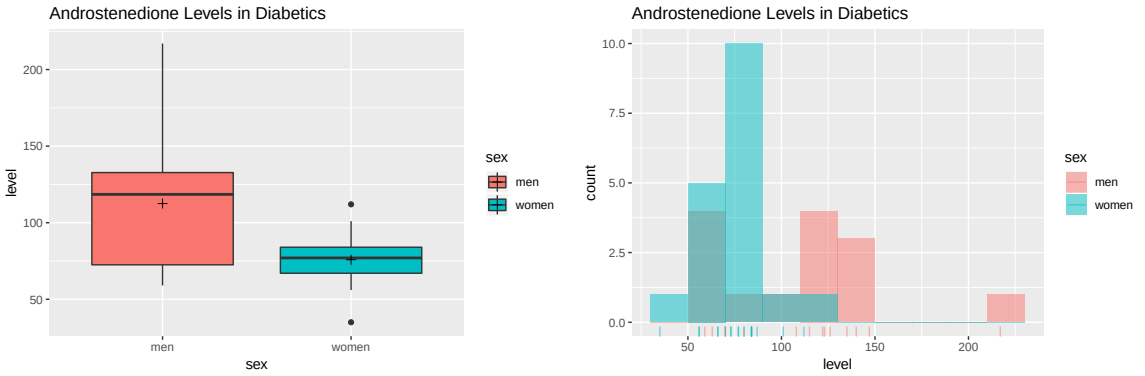
```
          147, 122, 108,  70)
women <- c( 84,  87,  77,  84,  73,  66,  70,  35,  77,  73,
            56, 112,  56,  84,  80, 101,  66,  84)
level <- c(men, women)
sex <- c(rep("men", length(men)), rep("women", length(women)))
andro <- data.frame(level, sex)
andro
##    level   sex
## 1    217   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))
```

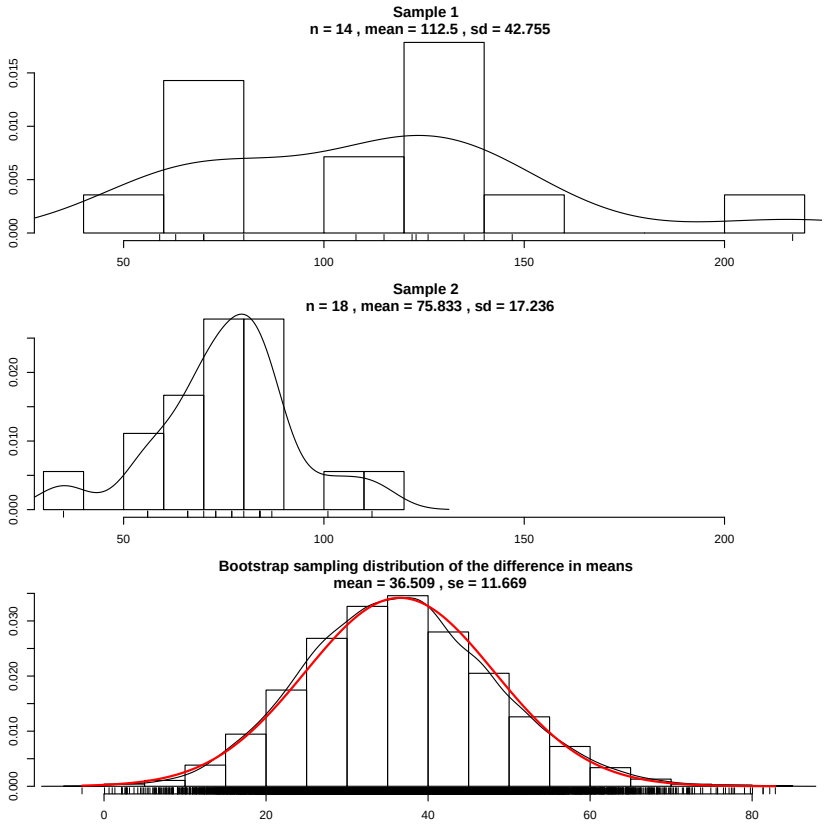
```
## [1] 42.75467 17.23625 60.25000 17.00000 14.00000 18.00000
```

```
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)

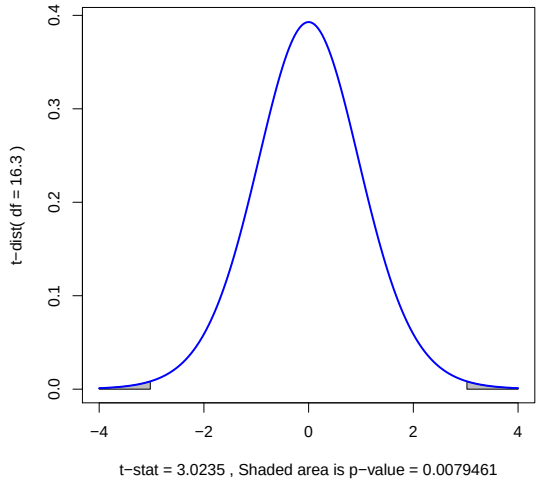
3. Specify confidence level, calculate t-stat, CI limits, p-value

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

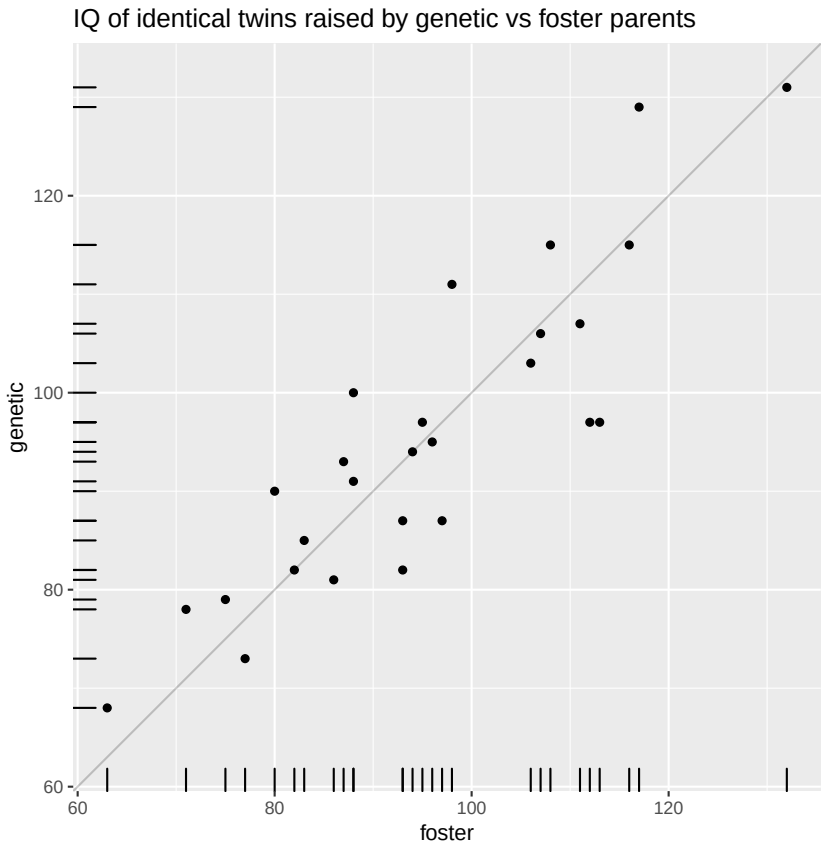
The most natural way to enter paired data is as two columns, one for each treatment group. You can then create a new column of differences, and do the usual one-sample graphical and inferential analysis on this column of differences, or you can do the paired analysis directly without this intermediate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

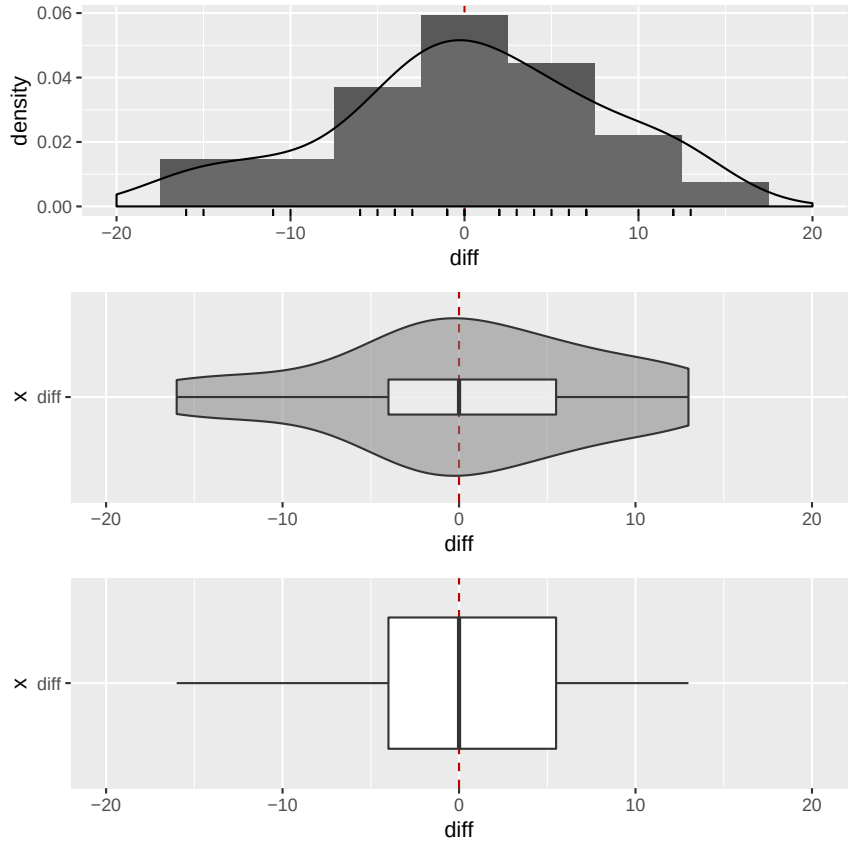


This plot of IQ scores shows that scores are related within pairs of twins. This is consistent with the need for a paired analysis.

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).

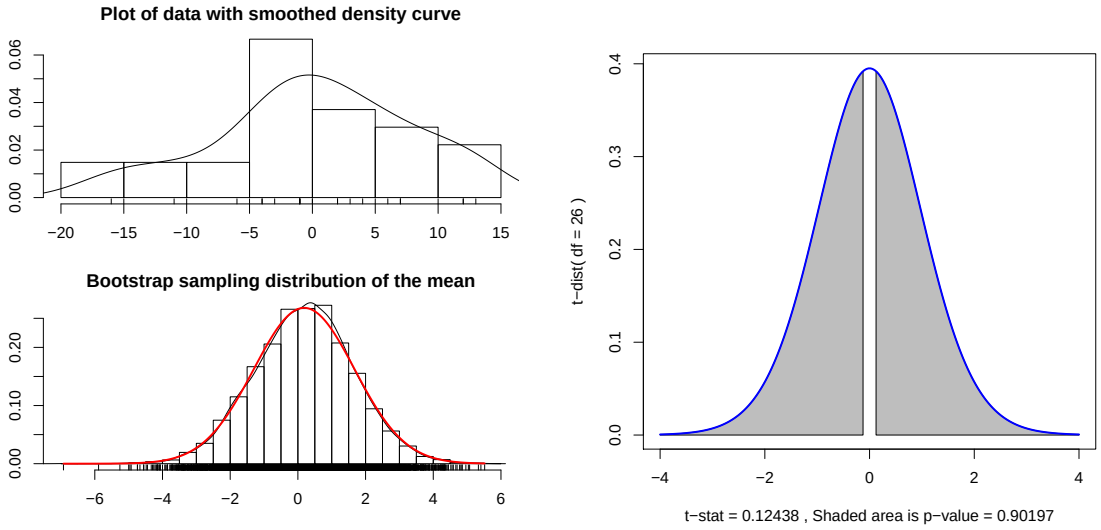
Given the sample of differences, I generated a one-sample CI and test. The hypothesis under test is $\mu_d = \mu_g - \mu_f = 0$. The p-value for this test is large. We do not have sufficient evidence to claim that the population mean IQs for twins raised apart

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 $\mu_d = \mu_B - \mu_A$.

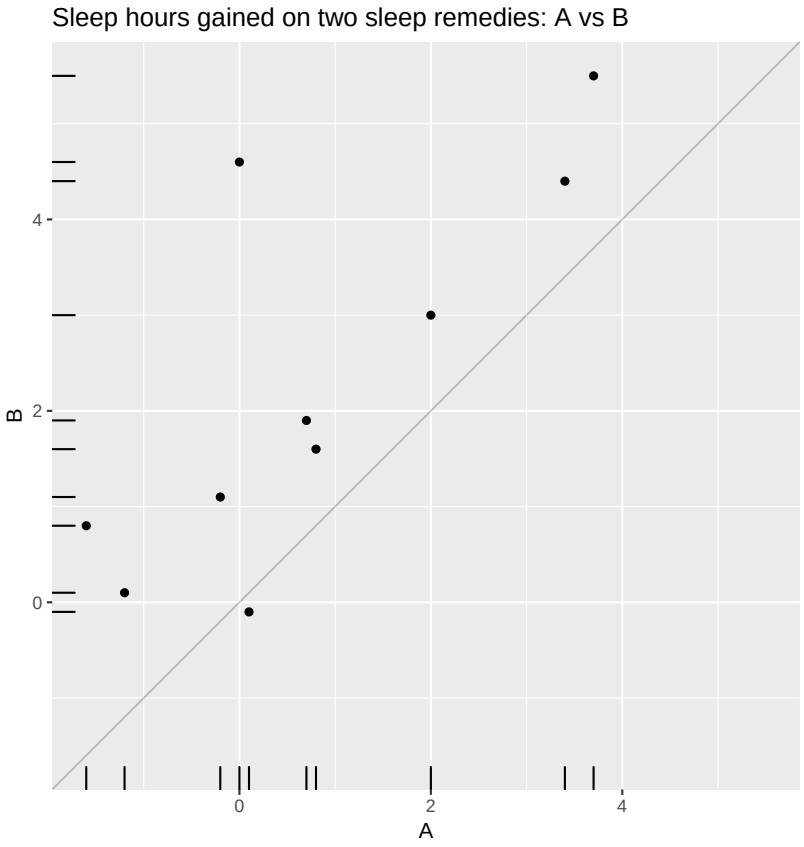
The observed distribution of differences between B and A is slightly skewed to the right, with a single outlier in the upper tail. The normality assumption of the standard one-sample t-test and CI are suspect here. I will continue with the analysis, anyways.

```
#### Example: Paired Differences on Sleep Remedies
# 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)
```

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

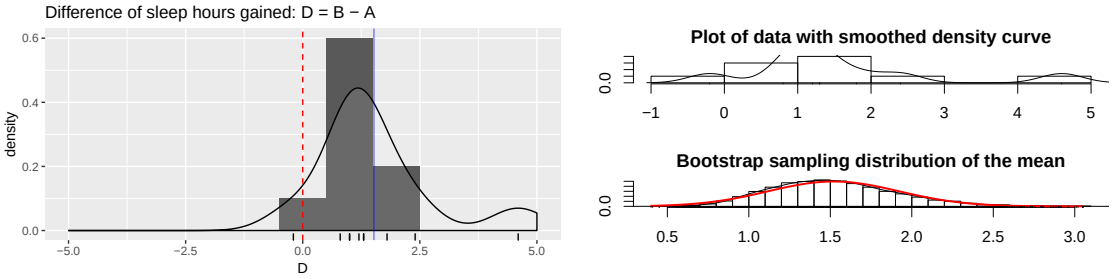
```
##       A    B    D
## 1   0.7  1.9  1.2
## 2  -1.6  0.8  2.4
## 3  -0.2  1.1  1.3
## 4  -1.2  0.1  1.3
## 5   0.1 -0.1 -0.2
## 6   3.4  4.4  1.0
## 7   3.7  5.5  1.8
## 8   0.8  1.6  0.8
## 9   0.0  4.6  4.6
## 10  2.0  3.0  1.0
```

```r
# 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)
```



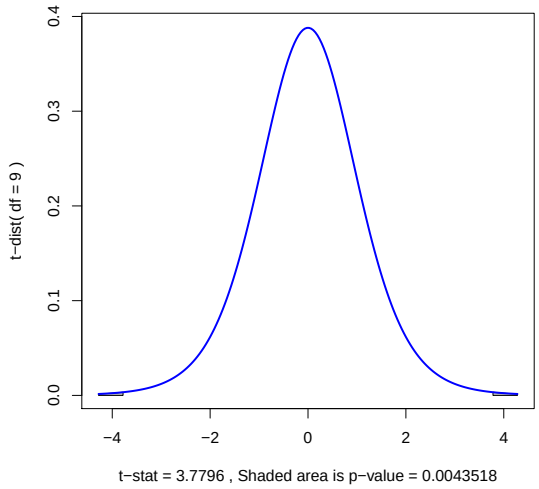
There is evidence here against the normality assumption of the sample mean. We'll continue anyway (in practice we'd use a nonparametric method, instead, in a later chapter).

```
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$D)
t.summary
##
##  One Sample t-test
##
## data:  sleep$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
# plot t-distribution with shaded p-value
t.dist.pval(t.summary)
```



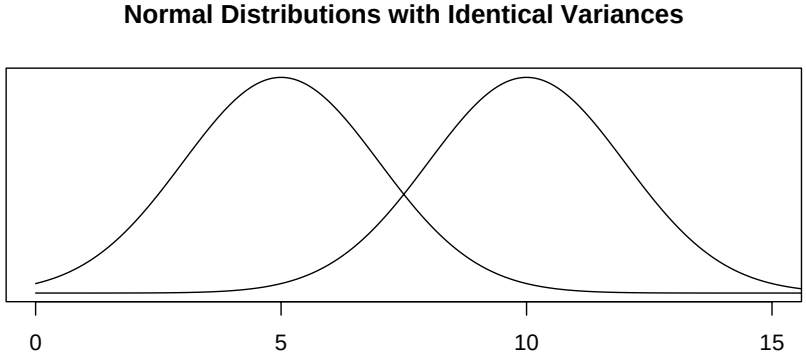
Question: In what order should the remedies be given to the patients?

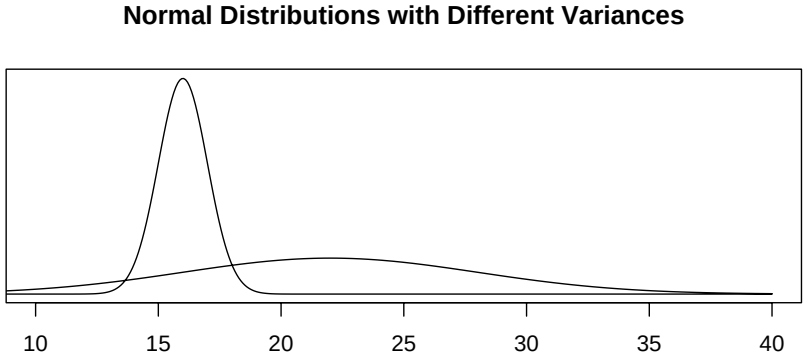
■ CLICKER Qs — Reporting results, STT.06.03.010 ■

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.


Normal Distributions with Identical Variances
0
5
10
15


Normal Distributions with Different Variances
10
15
20
25
30
35
40

Chapter 4

Checking Assumptions

Contents

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.

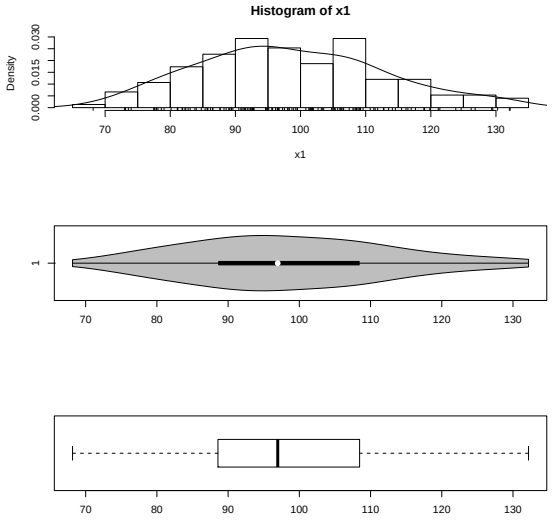
4.2 Testing Normality

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.

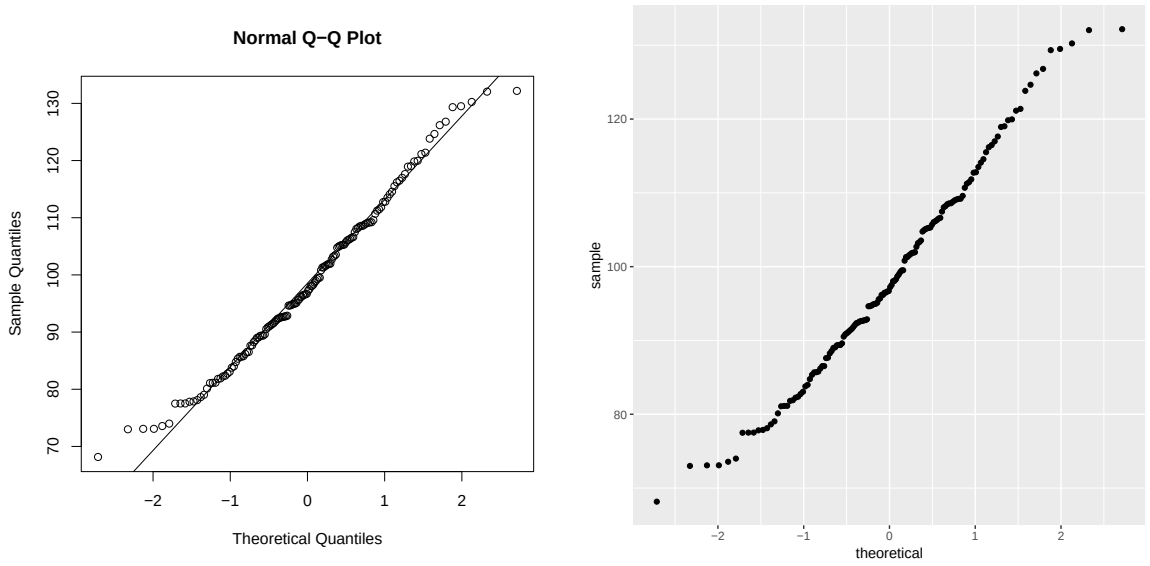
Let's start with some data from a normal distribution.

```
#### 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)
```

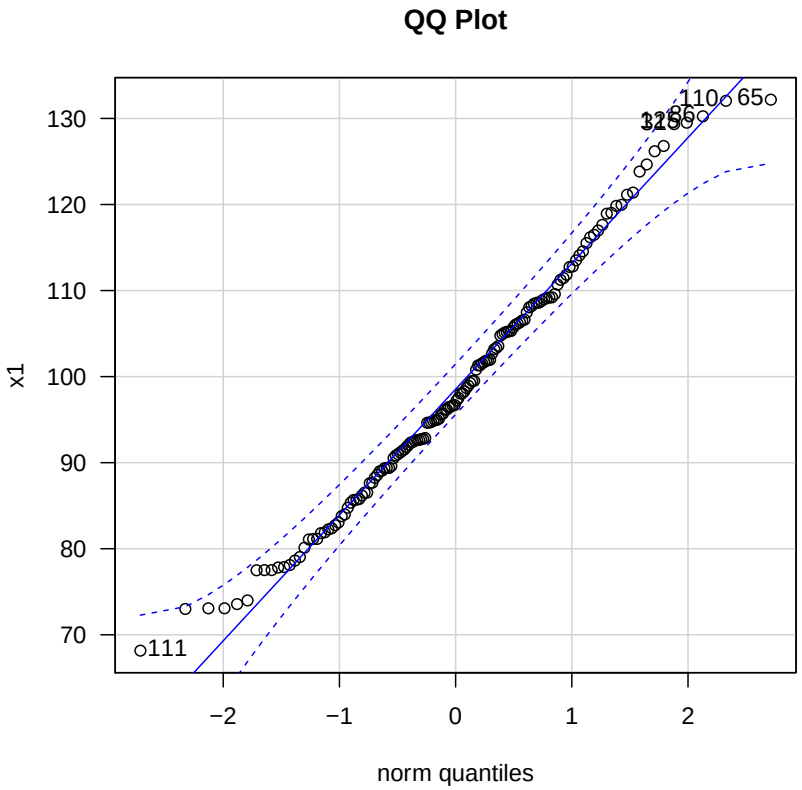


If you lay a straightedge along the bulk of the plot (putting in a regression line is not the right way to do it, even if it is easy), you see that the most extreme point on the right is a little below the line, and the last few points on the left a little above the line. What does this mean? The point 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)
## Loading required package:  carData
# 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 = list(n = 6, cex = 1), lwd = 1, main="QQ Plot")
## [1]  65 110  86 125  31 111
```



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.

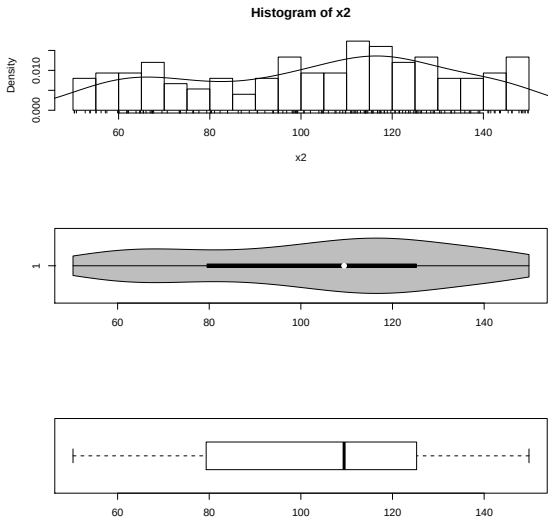
4.2.1 Normality tests on non-normal data

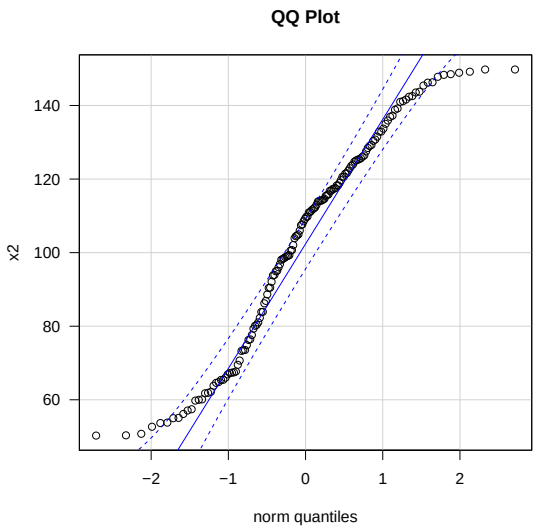
Let's turn to examples of sampling from other, non-normal distributions to see how the normal QQ plot identifies important features.

Light-tailed symmetric (Uniform)

```
#### Light-tailed symmetric (Uniform)
# sample from uniform distribution
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 = list(n = 0, cex = 1), lwd = 1, main="QQ Plot")
```

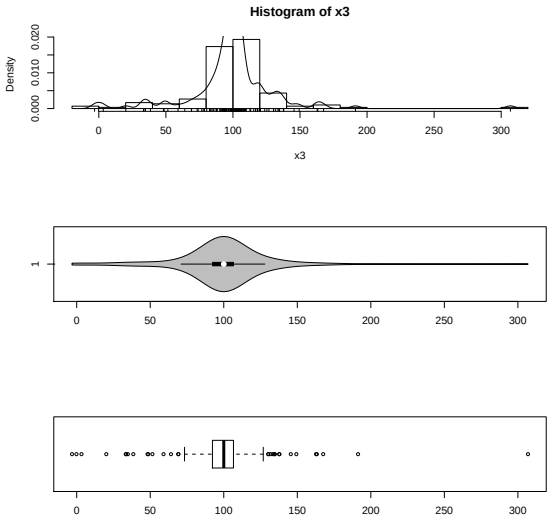


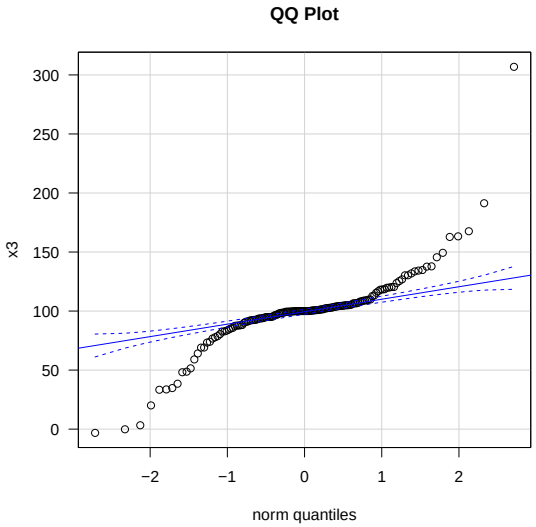


Heavy-tailed (fairly) symmetric (Normal-squared)

```
#### 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))
# 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 = list(n = 0, cex = 1), lwd = 1, main="QQ Plot")
```

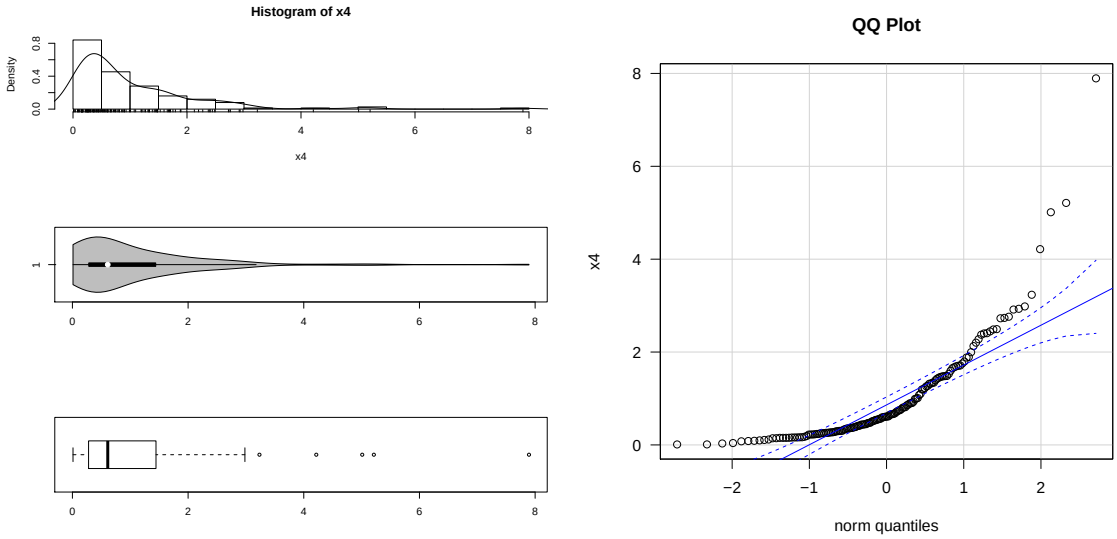




Right-skewed (Exponential)

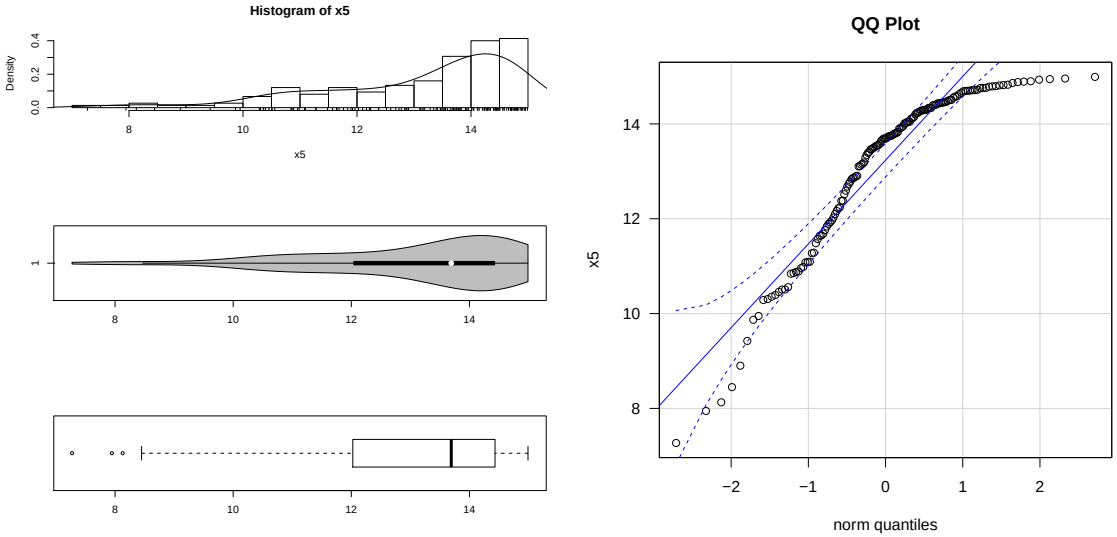
```
#### Right-skewed (Exponential)
# sample from exponential distribution
x4 <- rexp(150, rate = 1)

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

# 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 = list(n = 0, cex = 1), lwd = 1, main="QQ Plot")
```



Left-skewed (Exponential, reversed)

```
#### 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")

# boxplot
boxplot(x5, horizontal=TRUE)


par(mfrow=c(1,1))
qqPlot(x5, las = 1, id = list(n = 0, 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)
```

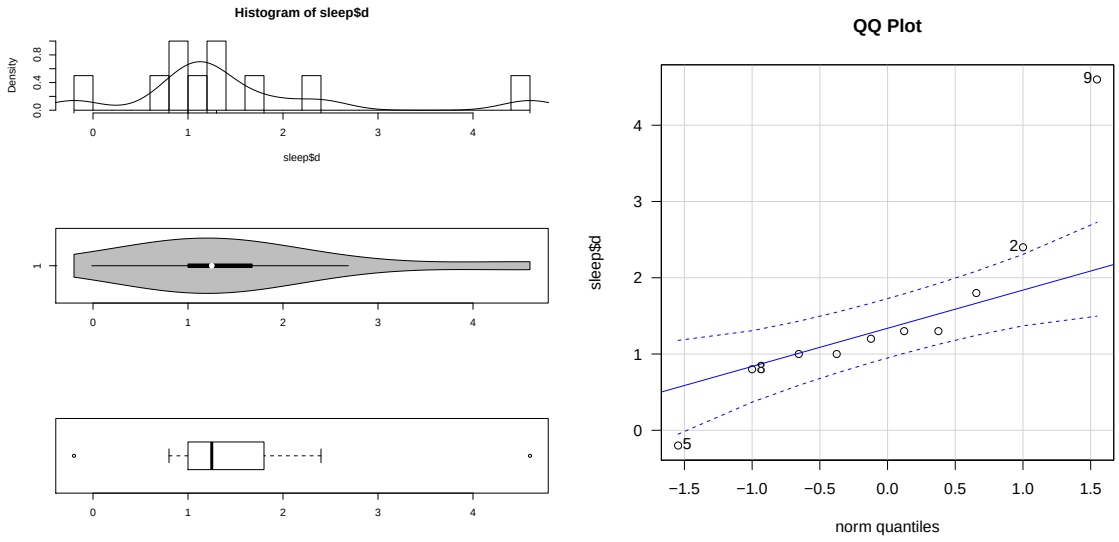
```
##
##  Anderson-Darling normality test
##
## data:  sleep$d
## A = 0.77378, p-value = 0.02898
```

```
# 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 = list(n = 4, cex = 1), lwd = 1, main="QQ Plot")
## [1] 9 5 2 8
```



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()
# 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))
```

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
```

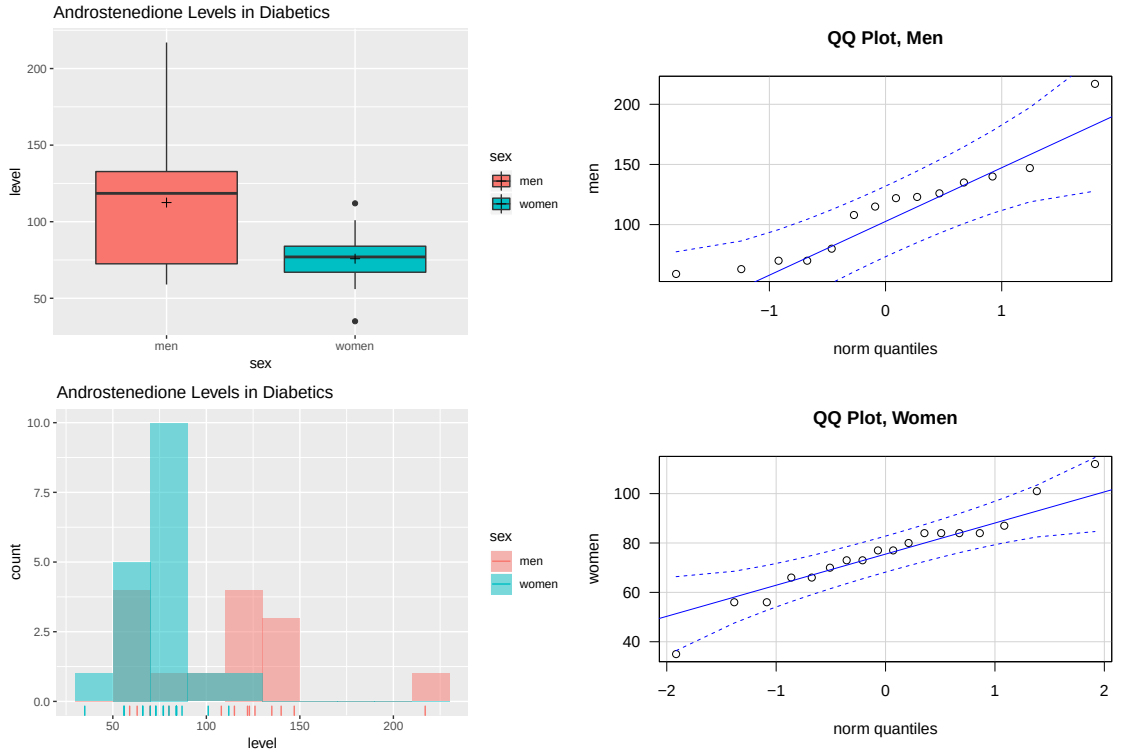
```
library(nortest)
ad.test(women)
```

```
##
##  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
```

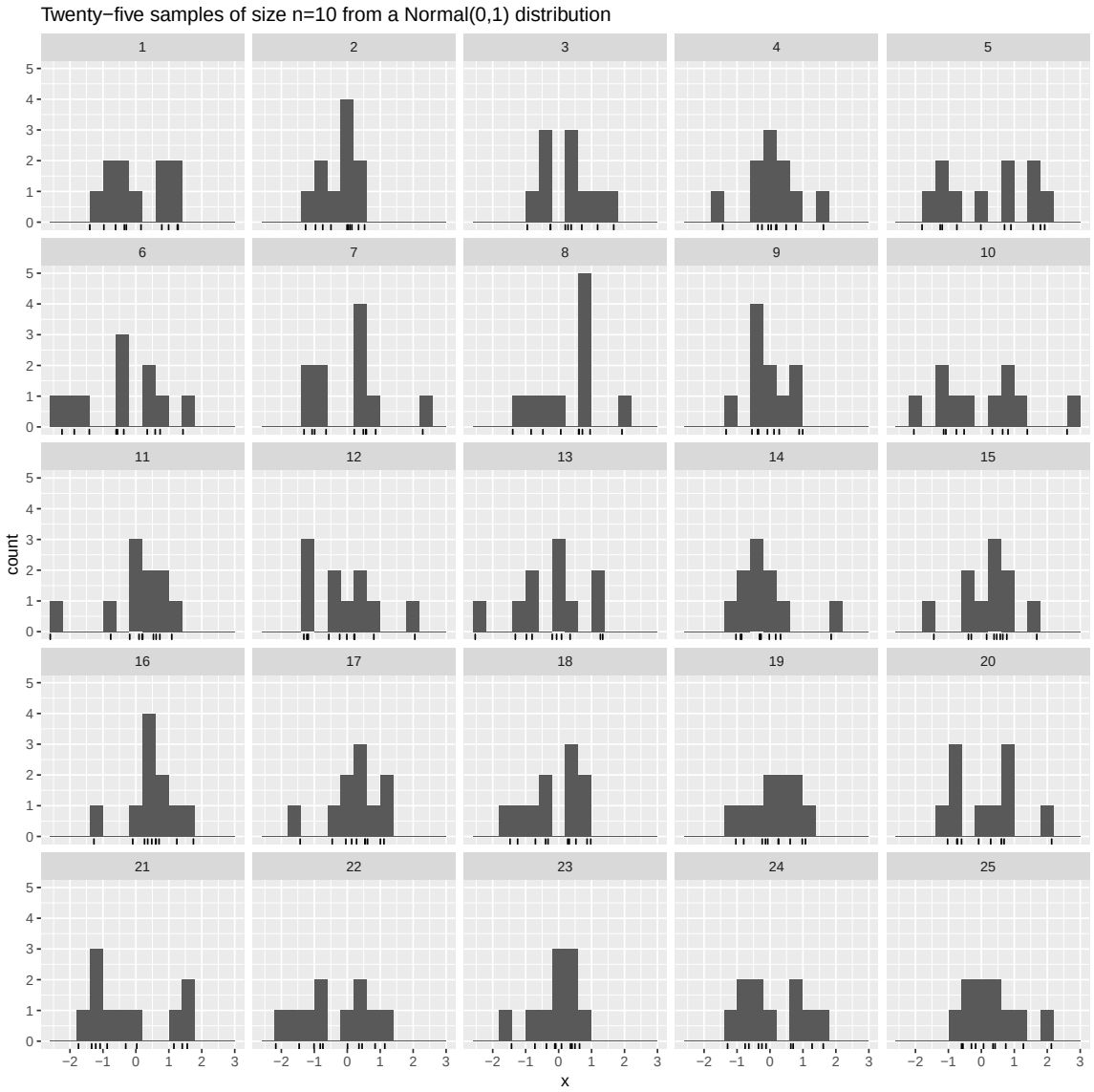
```
p2 <- p2 + labs(title = "Androstenedione Levels in Diabetics")
#print(p2)

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


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

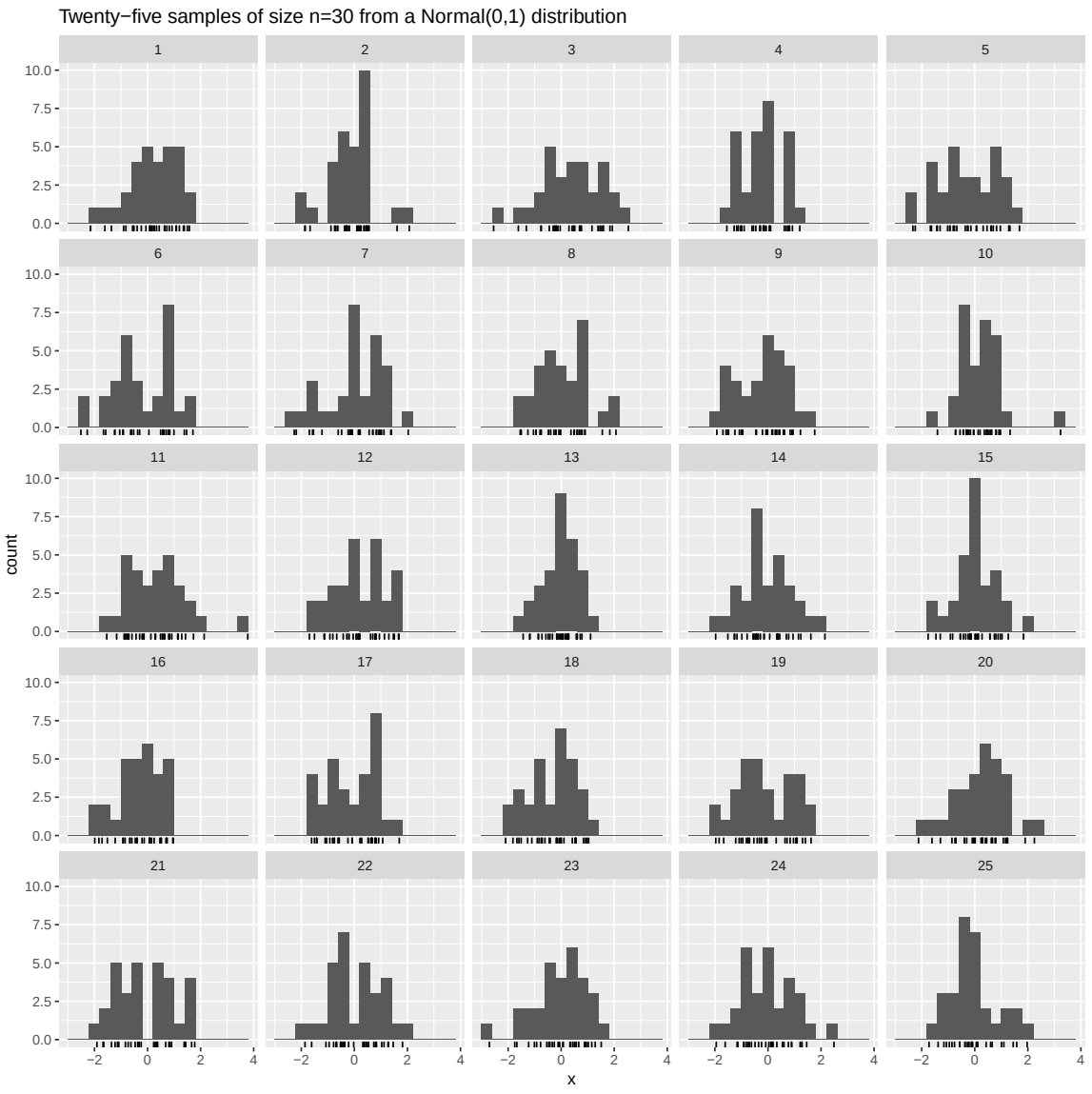
```
n = 10
r = 5
norm.many <- data.frame(id = rep(seq(1:r^2), n)
                        , x = rnorm(r^2 * n)
                         )
library(ggplot2)
p <- ggplot(norm.many, aes(x = x))
p <- p + geom_histogram(binwidth = 0.4)
p <- p + geom_rug()
p <- p + facet_wrap(~ id, ncol = r)
p <- p + labs(title = "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

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 $df(\text{Total}) = (k-1) + (n^* - k) = n^* - 1$.

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

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

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.

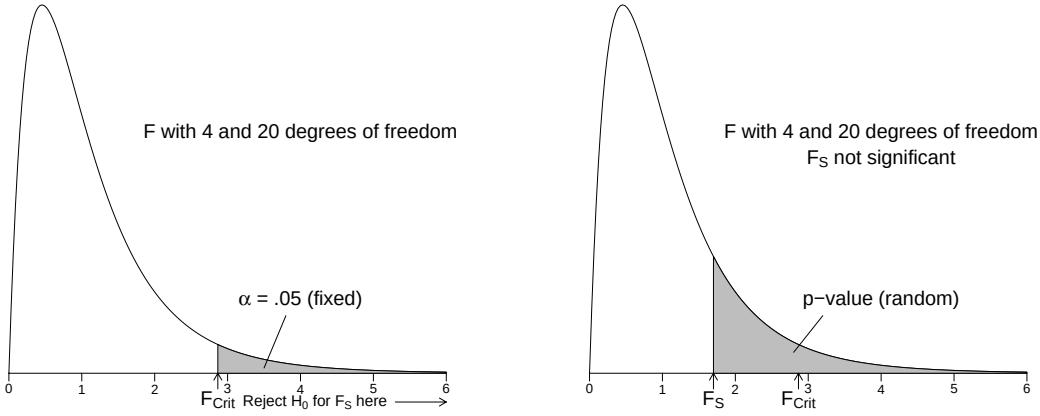
The decision on whether to reject $H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$ is based on the ratio of the MS(Between) and the MS(Within):

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

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.

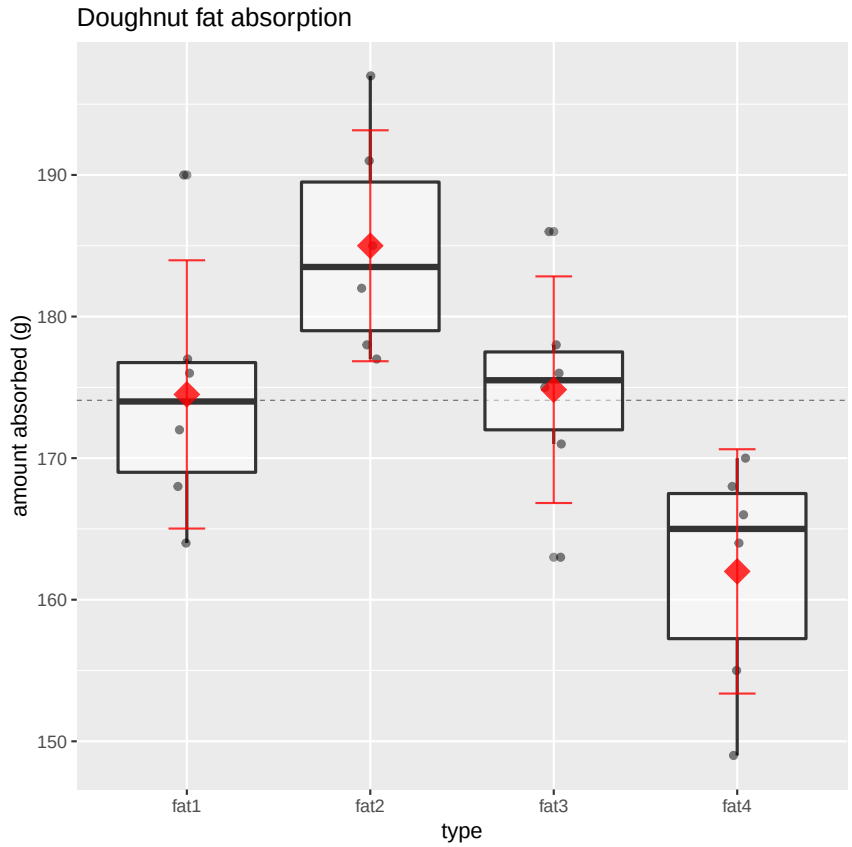
```
#### Back to ANOVA
# Calculate the mean, sd, n, and se for the four fats

# 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
```


■ CLICKER Qs — 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 = "none"`. 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.

■ CLICKER Qs — 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
----------------------
         ----------------------
```

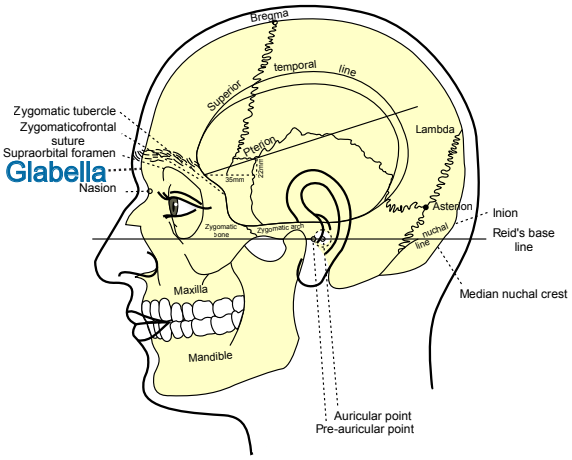
The Bonferroni method tends to produce "coarser" groups than the FSD method, because the individual comparisons are conducted at a lower (alpha/error) level. Equivalently, the minimum significant difference is inflated for the Bonferroni method. For example, in the doughnut problem with $FER \leq 0.05$, 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 152, we see that the sole comparison where the absolute difference between sample means exceeds 13.824 involves Fats 2 and 4.

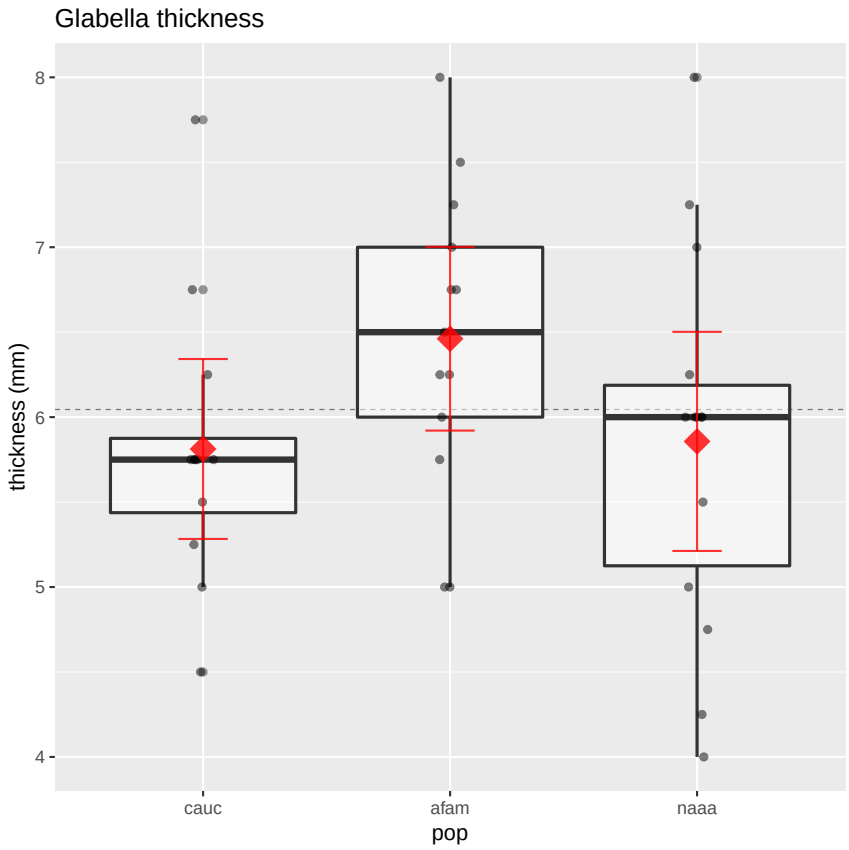
Example from Koopmans: glabella facial tissue thickness In an anthropological study of facial tissue thickness for different racial groups, data were taken during autopsy at several points on the faces of deceased individuals. The Glabella measurements taken at the bony ridge for samples of individuals 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
```

■ CLICKER Qs — ANOVA, multiple comparisons ■

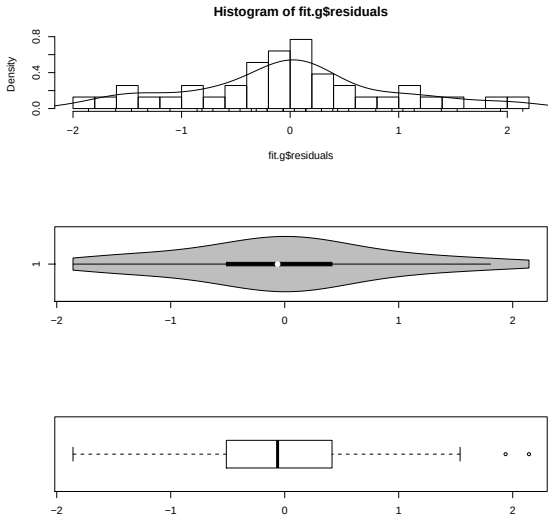
5.4 Checking Assumptions in ANOVA Problems

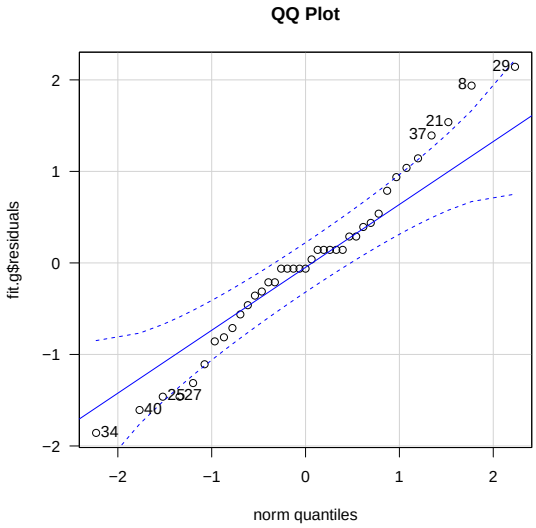
The classical ANOVA assumes that the populations have normal frequency curves and the populations have equal variances (or spreads). You can test the normality assumption using multiple normal QQ-plots and normal scores tests, which we discussed in Chapter 4. An alternative approach that is useful with 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 = list(n = 8, cex = 1), lwd = 1, main="QQ Plot")
## 29  8 34 40 21 25 27 37
## 26  8 31 37 19 23 25 34
```

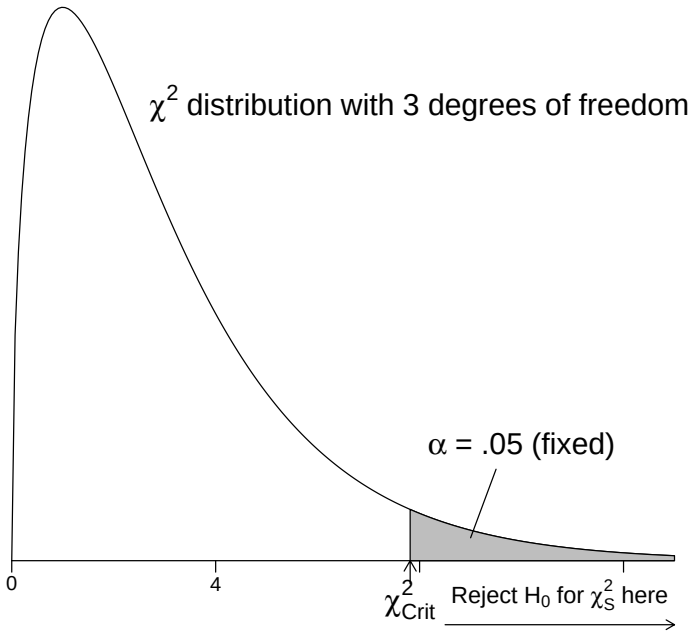





```
shapiro.test(fit.g$residuals)
##
##  Shapiro-Wilk normality test
##
## data:  fit.g$residuals
## W = 0.97693, p-value = 0.5927
library(nortest)
ad.test(fit.g$residuals)
##
##  Anderson-Darling normality test
##
```

```
## 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.

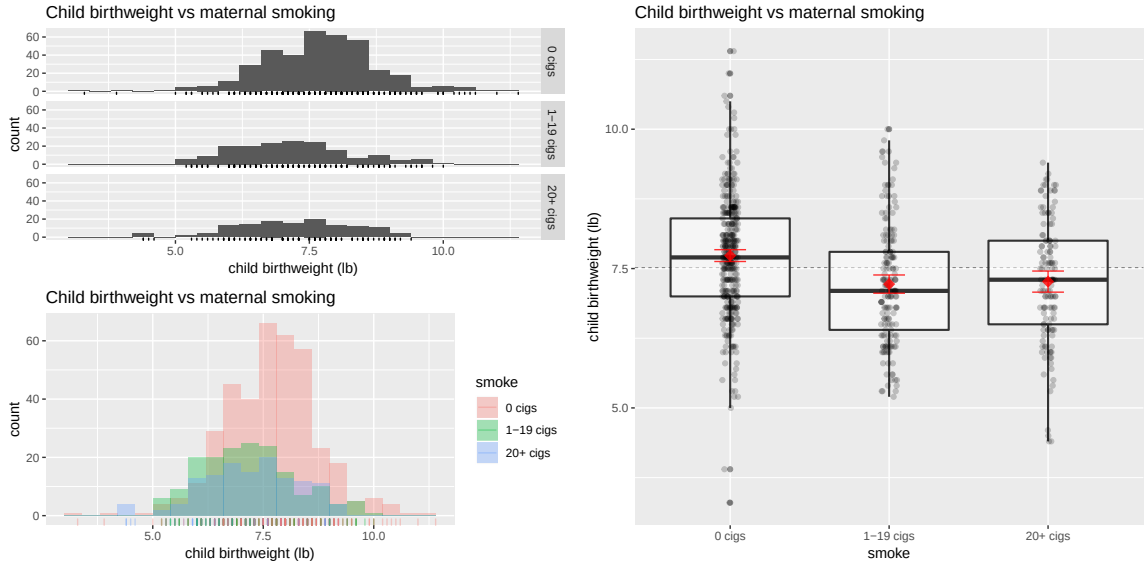
Let μ_i = pop mean birth weight (lb) for children in group i, $(i = 1, 2, 3)$. We wish to test $H_0 : \mu_1 = \mu_2 = \mu_3$ against H_A : not H_0.

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
```

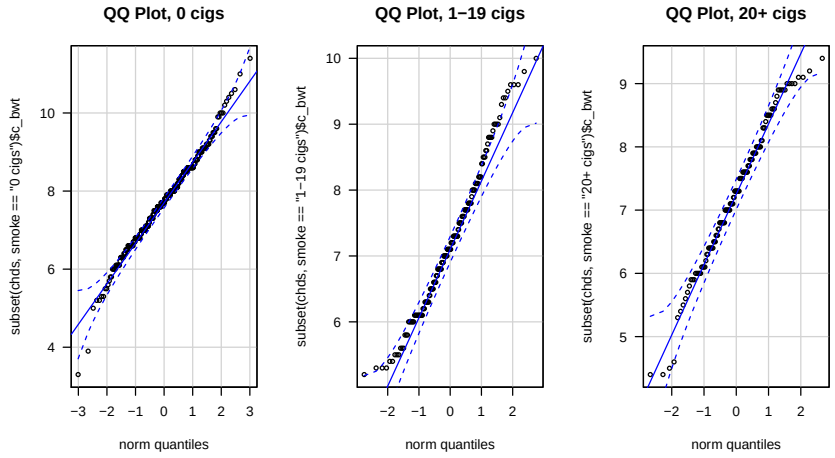
```
chds[(chds$m_smok == 0), "smoke"] <- "0 cigs";
# less than 1 pack (20 cigs = 1 pack)
chds[(chds$m_smok > 0) & (chds$m_smok < 20),"smoke"] <- "1-19 cigs";
# at least 1 pack (20 cigs = 1 pack)
chds[(chds$m_smok >= 20),"smoke"] <- "20+ cigs";
chds$smoke <- factor(chds$smoke)
```

```
# 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"   )$c_bwt, las = 1, id = list(n = 0, cex = 1),
  lwd = 1, main="QQ Plot, 0 cigs")
qqPlot(subset(chds, smoke == "1-19 cigs")$c_bwt, las = 1, id = list(n = 0, cex = 1),
  lwd = 1, main="QQ Plot, 1-19 cigs")
qqPlot(subset(chds, smoke == "20+ cigs" )$c_bwt, las = 1, id = list(n = 0, 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
```

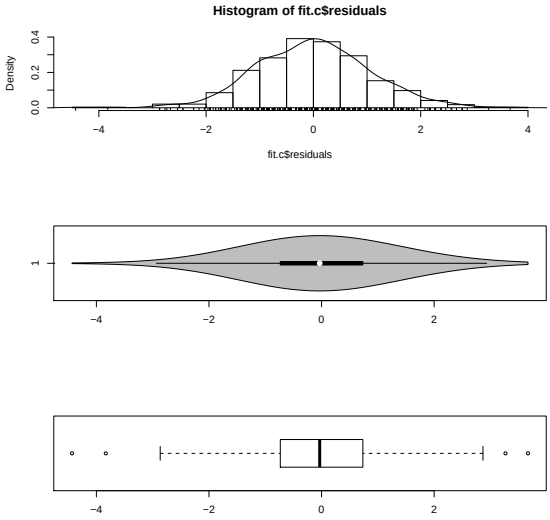
Fit the ANOVA (we'll look at the table below).

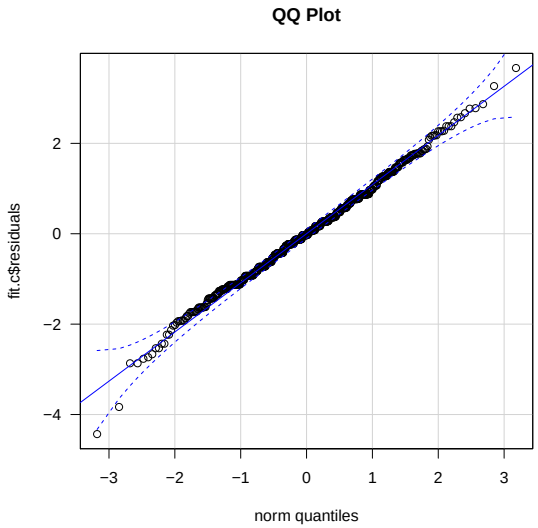
```
fit.c <- aov(c_bwt ~ smoke, data = chds)
```

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))
# 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 = list(n = 0, cex = 1), lwd = 1, main="QQ Plot")
```






```
shapiro.test(fit.c$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  fit.c$residuals
## W = 0.99553, p-value = 0.04758
```

```
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

ADA1: Nonparametric, categorical, and regression methods

Chapter 6

Nonparametric Methods

Contents

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

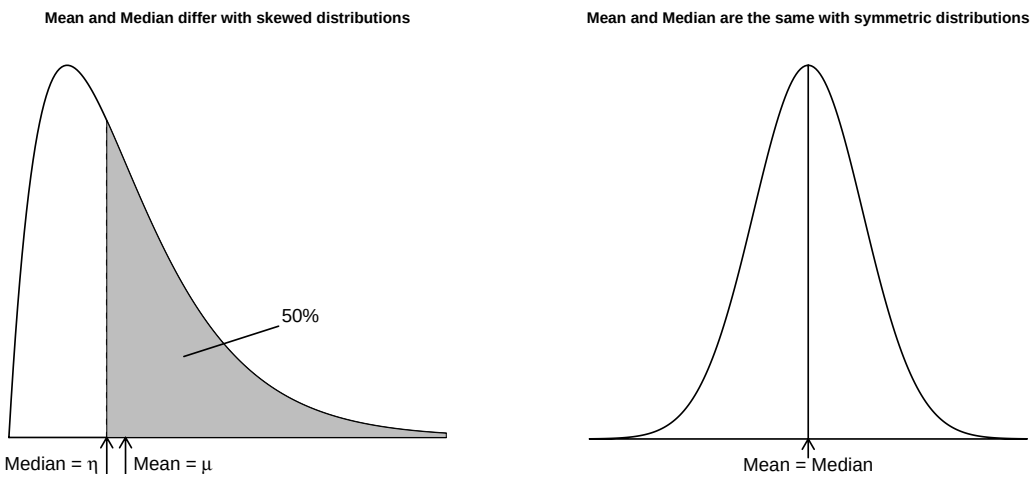
Nonparametric methods do not require the normality assumption of classical techniques. When the normality assumption is met, the ANOVA and t-test are most powerful, in that if the alternative is true these methods will make the correct decision with highest probability. However, if the normality assumption 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 $H_0 : \eta = \eta_0$.

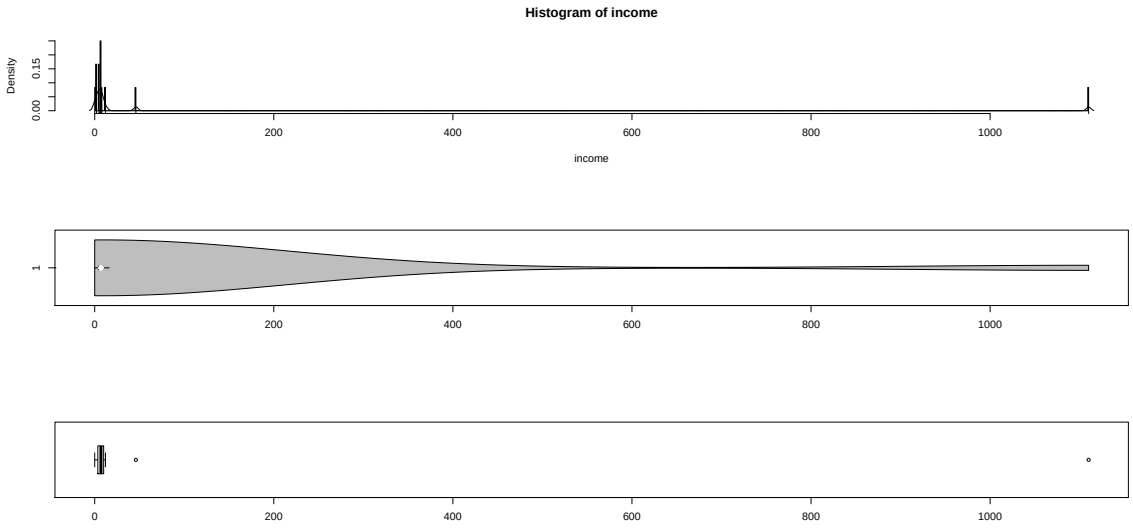
Not all test sizes and confidence levels are possible because the test statistic S is discrete valued. R's `SIGN.test()` in the `BSDA` package gives an exact p-value for the test, and approximates the desired confidence level using a linear interpolation algorithm.

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
```

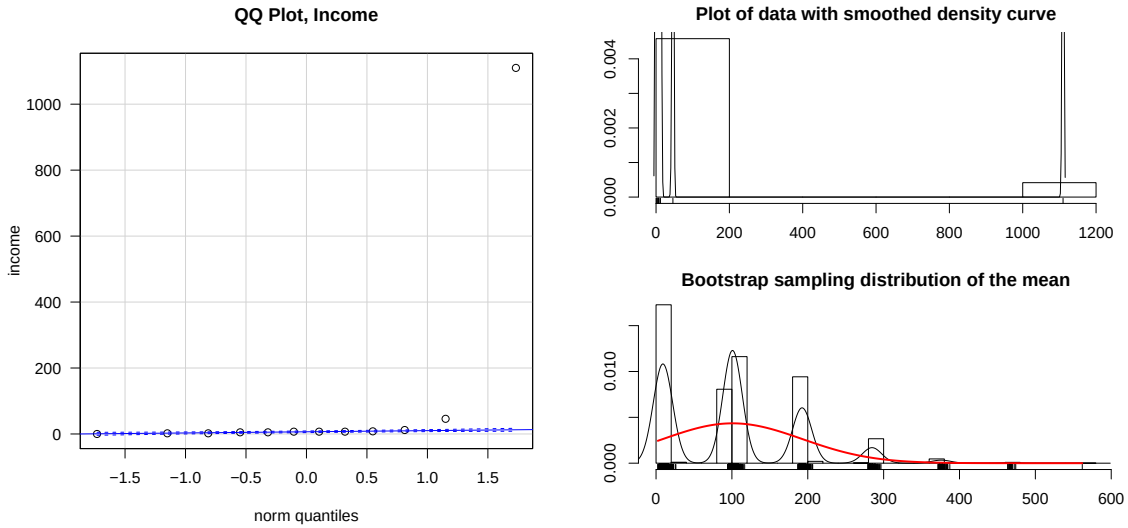
The income data is unimodal, skewed right, with two extreme outliers.

```
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 = list(n = 0, 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:carData':
##
##    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
```

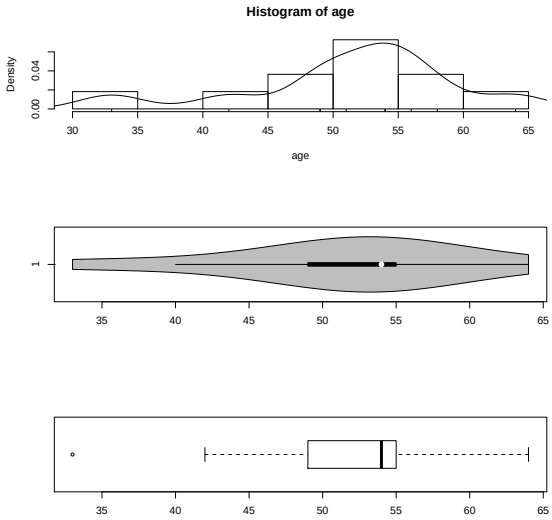
Example: Age at First Heart Transplant Recall that the distribution of ages is skewed to the left with a lower outlier. A question of interest is whether the "typical age" at first transplant is 50. This can be formulated as a test about the population median η or as a test about the population mean 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.

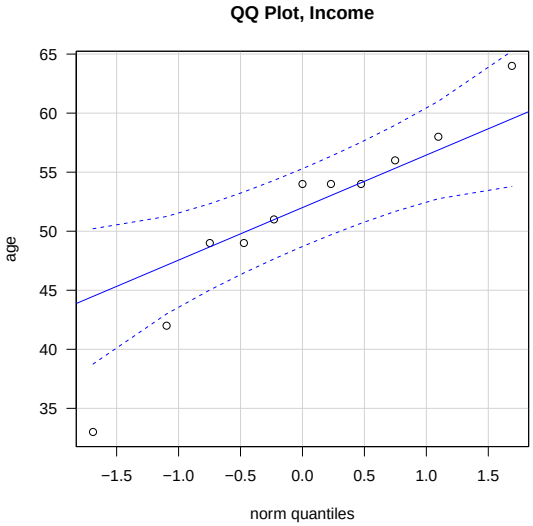
```
par(mfrow=c(3,1))
# Histogram overlaid with kernel density curve
hist(age, freq = FALSE, breaks = 10)
points(density(age), type = "l")
rug(age)

# violin plot
```

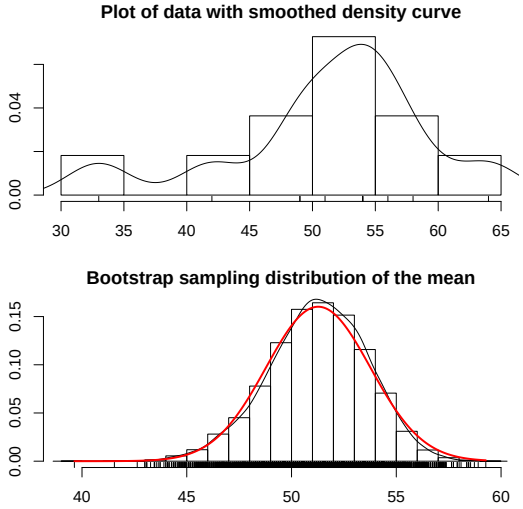
```
library(vioplot)
vioplot(age, horizontal=TRUE, col="gray")

# 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 = list(n = 0, 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**

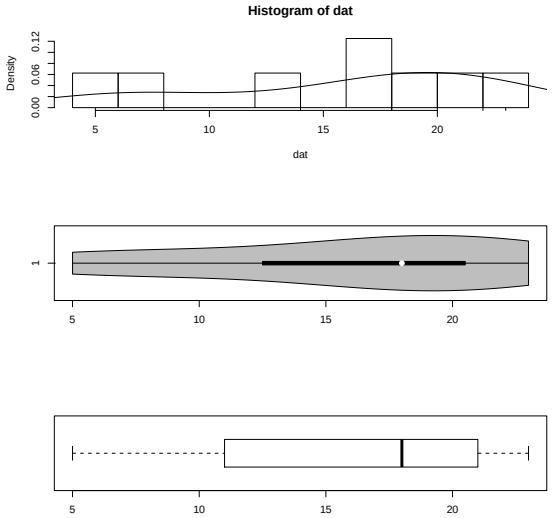
from the expected value of 18? To formally answer this question, we need to use the Wilcoxon procedures, which are implemented in R with `wilcox.test()`.

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
```

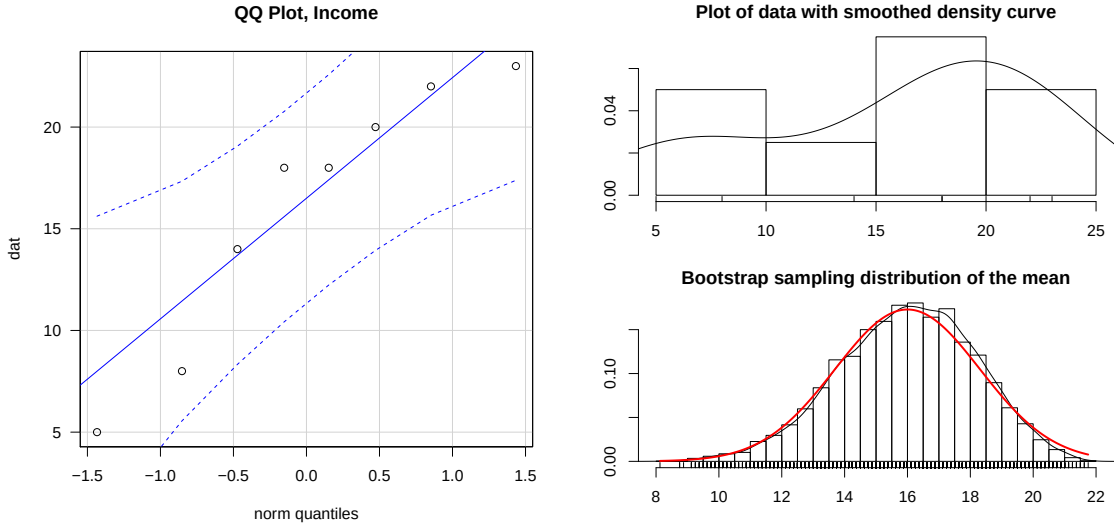
The dat data is unimodal, skewed left, no extreme outliers.

```
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 = list(n = 0, 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.

```r
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
```

```r
# 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
confidence 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
```

```r
# 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.

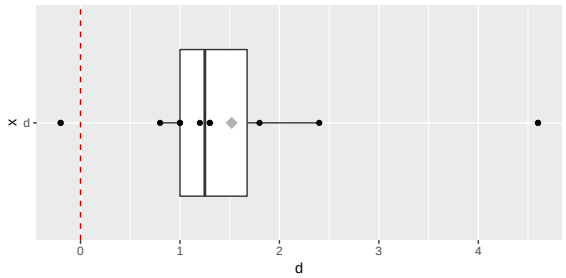
```
#### Example: Sleep Remedies
# Data and numerical summaries
a <- c( 0.7, -1.6, -0.2, -1.2,  0.1,  3.4,  3.7,  0.8,  0.0,  2.0)
b <- c( 1.9,  0.8,  1.1,  0.1, -0.1,  4.4,  5.5,  1.6,  4.6,  3.0)
d <- b - a;
sleep <- data.frame(a, b, d)
summary(sleep$d)
```

```
##    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**.

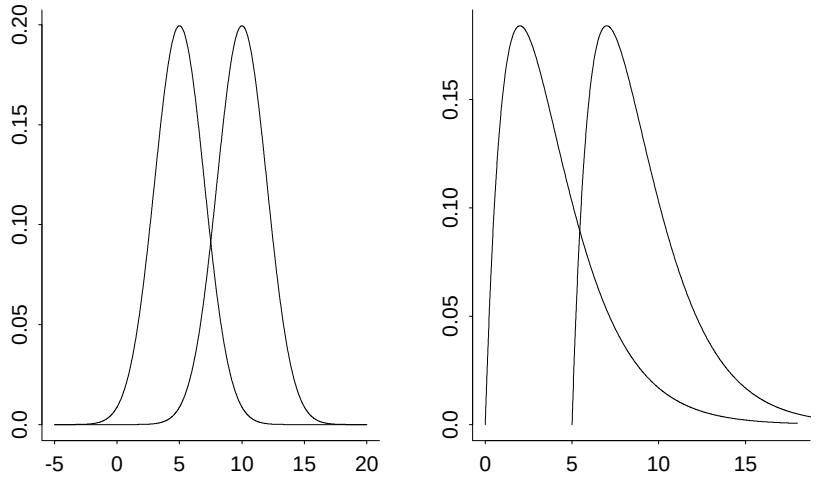
Nonparametric methods have gained widespread acceptance in many scientific disciplines, but not all. Scientists in some disciplines continue to use classical t-methods because they believe that the methods are robust to non-normality. As noted above, this is a robustness of validity, not sensitivity. 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 methods 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.

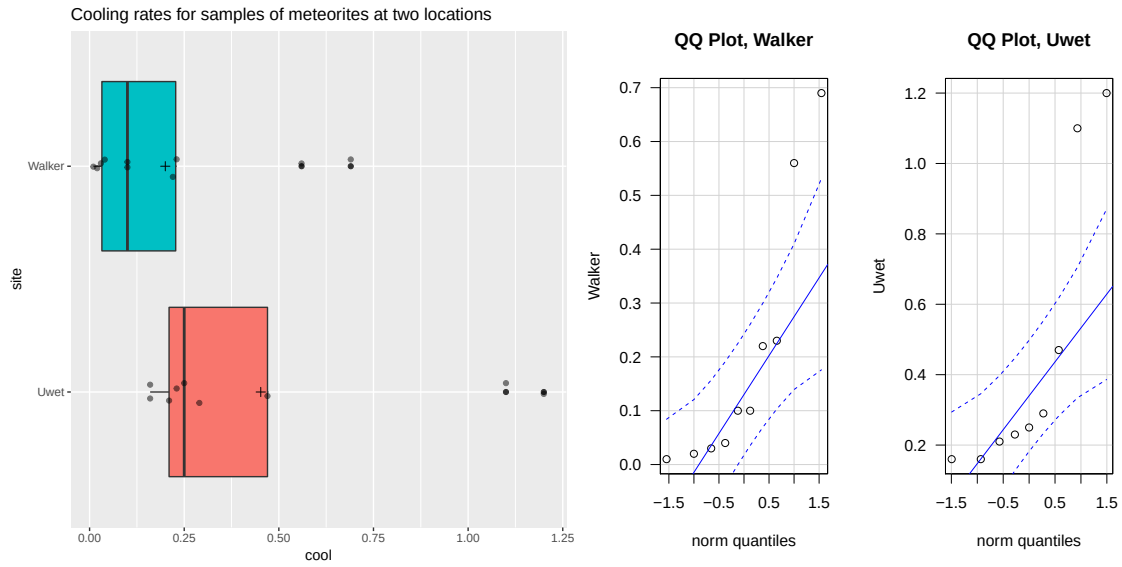
Example: Comparison of Cooling Rates of Uwet and Walker Co. Meteorites The Uwet[1] (Cross River, Nigeria) and Walker[2] County (Alabama, US) meteorite cooling rate data are below. A primary interest is comparing the population "typical" cooling rate measurements.

```
#### 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.

[1] `http://www.lpi.usra.edu/meteor/metbull.php?code=24138`
[2] `http://www.lpi.usra.edu/meteor/metbull.php?code=24204`

```
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 = list(n = 0, cex = 1), lwd = 1, main="QQ Plot, Walker")
qqPlot(Uwet, las = 1, id = list(n = 0, 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
```

```
confidence intervals with ties
```

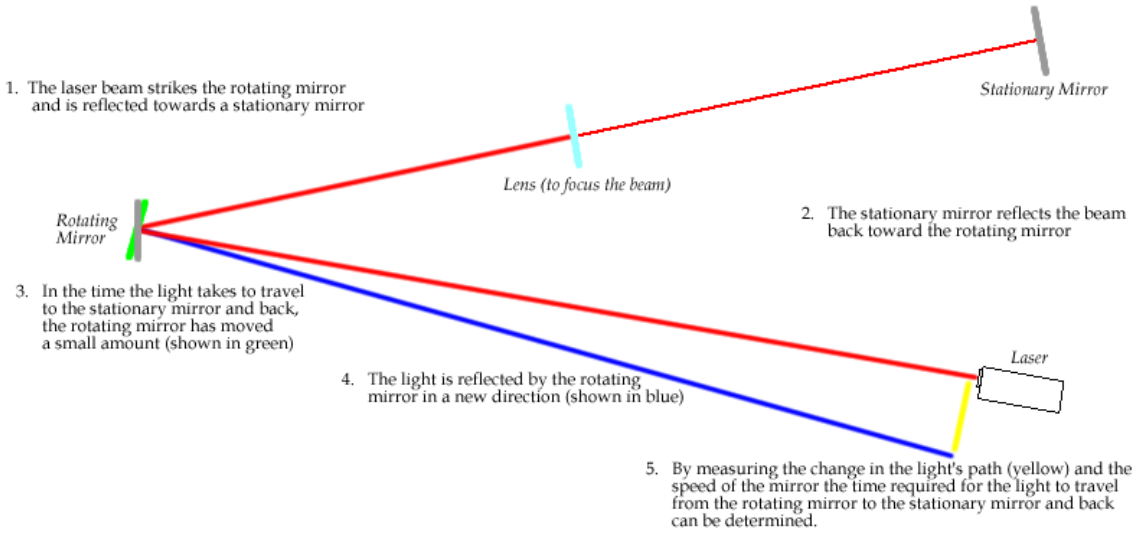
The difference between the WMW and t-test p-values and CI lengths (i.e. the WMW CI is narrower and the p-value smaller) reflects the effect of the outliers on the sensitivity of the standard tests and CI.

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.



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.

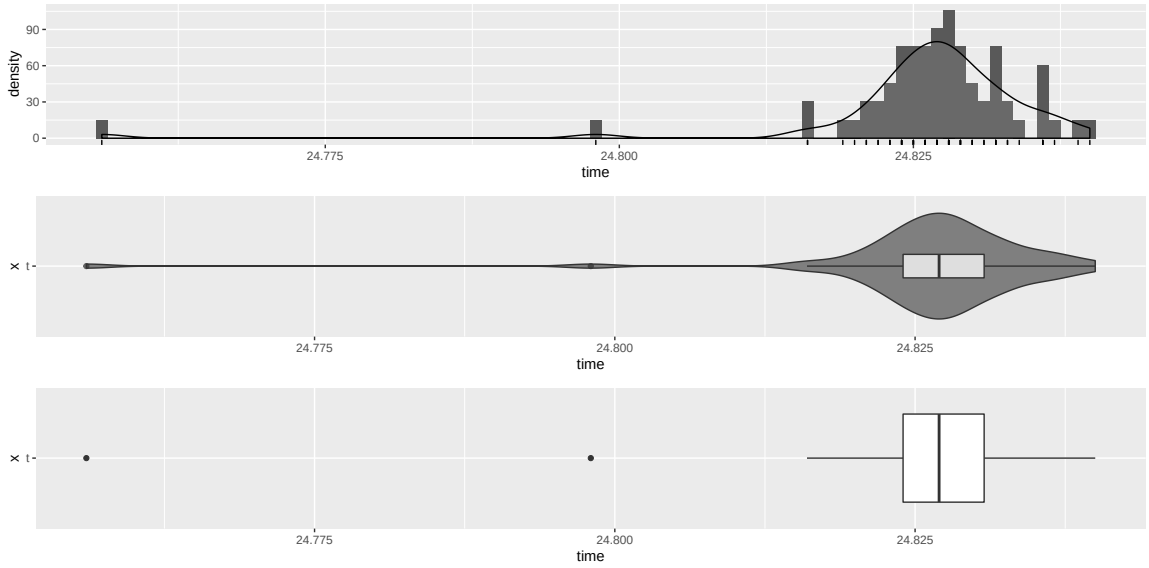
```
#### Example: Newcombe's Data
time <- c(24.828, 24.833, 24.834, 24.826, 24.824, 24.756
        , 24.827, 24.840, 24.829, 24.816, 24.798, 24.822
        , 24.824, 24.825, 24.823, 24.821, 24.830, 24.829
```

[3] http://en.wikipedia.org/wiki/File:Speed_of_light_(foucault).PNG

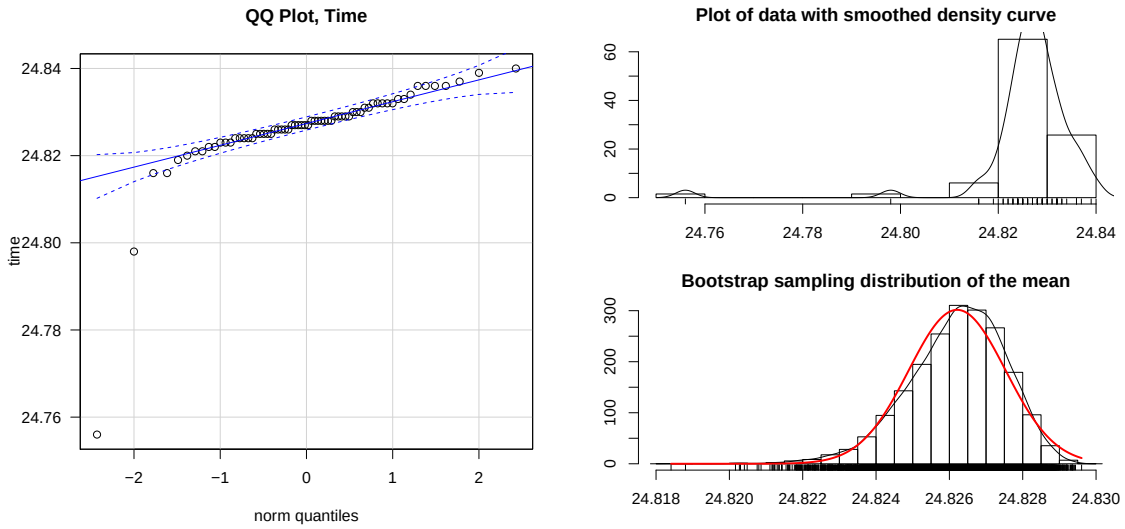
```
        , 24.831, 24.824, 24.836, 24.819, 24.820, 24.832
        , 24.836, 24.825, 24.828, 24.828, 24.821, 24.829
        , 24.837, 24.828, 24.830, 24.825, 24.826, 24.832
        , 24.836, 24.830, 24.836, 24.826, 24.822, 24.823
        , 24.827, 24.828, 24.831, 24.827, 24.827, 24.827
        , 24.826, 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 = list(n = 0, 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$conf.int
## [1] 24.82600 24.82849
## attr(,"conf.level")
## [1] 0.95
diff(s.sum$conf.int)
## [1] 0.00249297
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.

[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	$\sigma^2 = \mu^2$	$y' = \log(y)$

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 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.
```

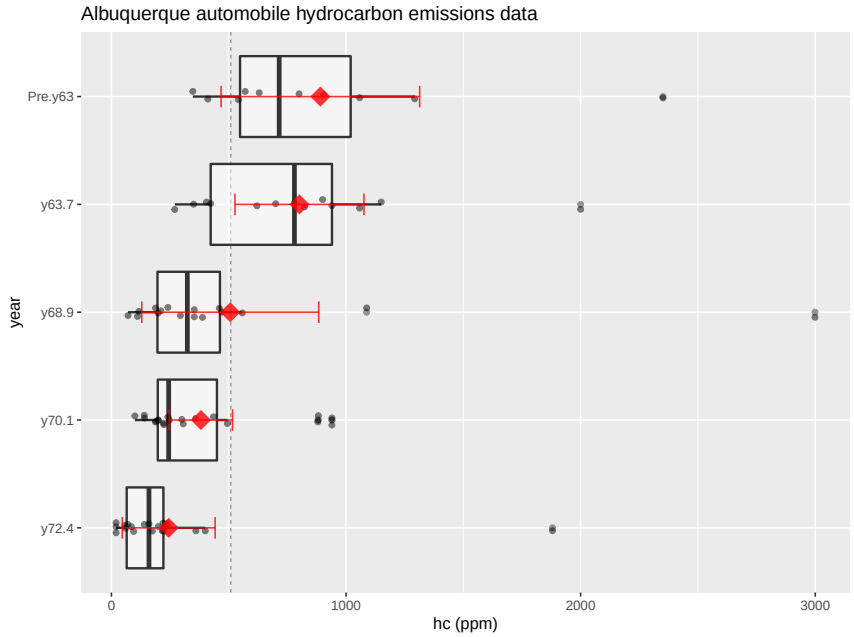
```
##   270.0   423.0   780.0   801.5   940.0  2000.0
## ------------------------------------------------------
## emis.long$year: y68.9
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    71.0   196.2   323.5   506.3   462.5  2999.0
## ------------------------------------------------------
## emis.long$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
```

```r
by(emis.long$loghc, emis.long$year, summary)
```

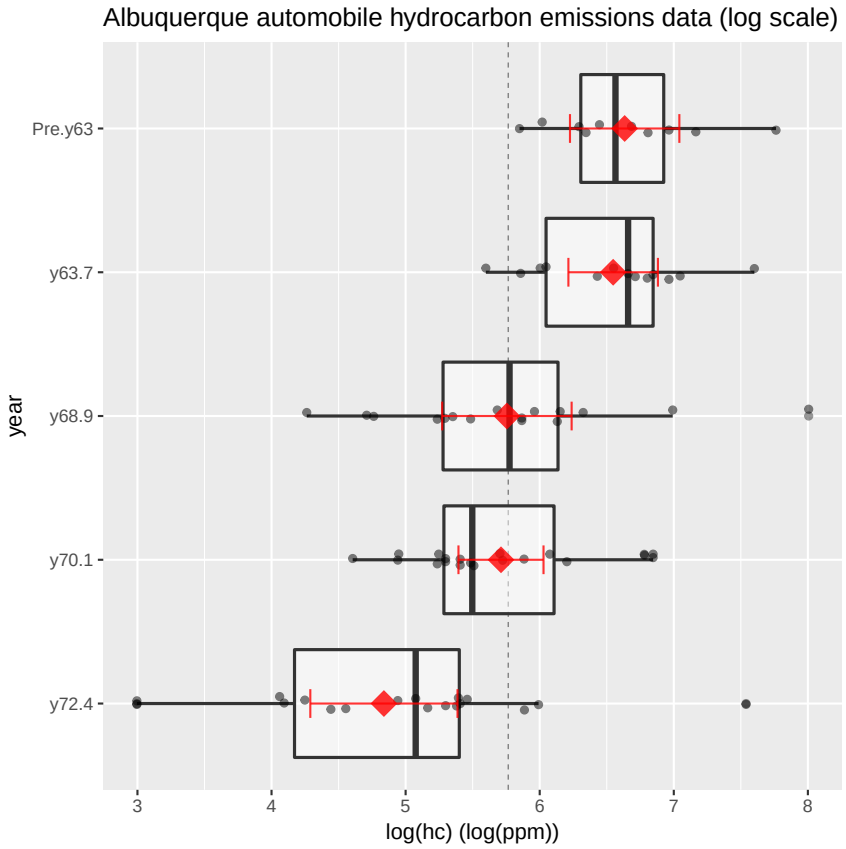
```
## emis.long$year: Pre.y63
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.849   6.306   6.565   6.634   6.925   7.763
## ------------------------------------------------------
## emis.long$year: y63.7
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.598   6.047   6.659   6.548   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
```

```r
# 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
```

```r
# 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).

[5]It is unethical to choose a method based on the results it gives.

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
  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
```

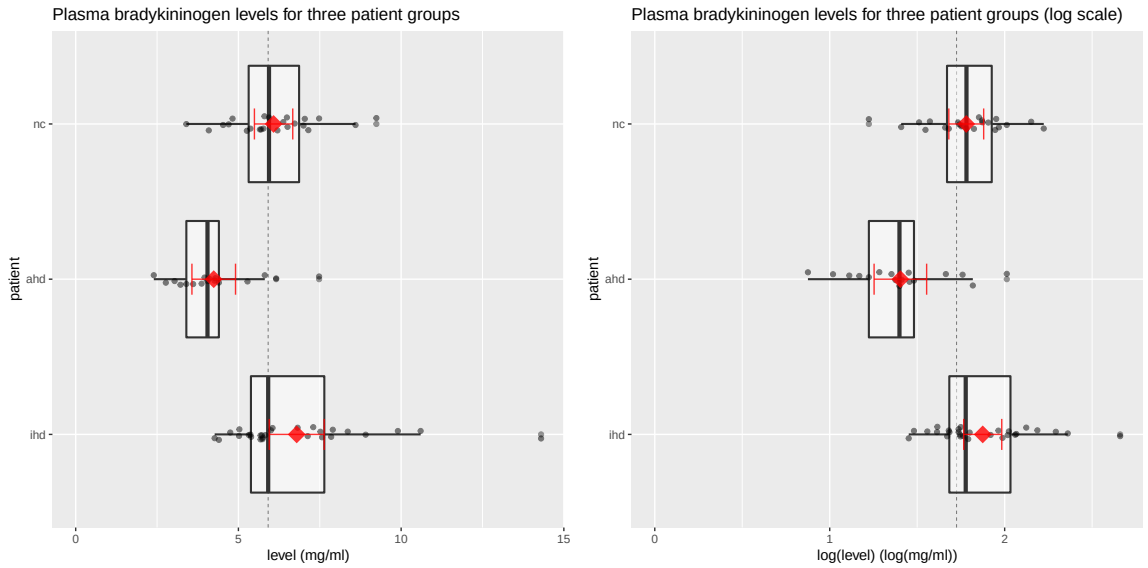
```r
# convert to long format
hd.long <- melt(hd,
                variable.name = "patient",
                value.name = "level",
                na.rm = TRUE
              )
## No id variables; using all as measure variables
# naming variables manually, the variable.name and value.name not working 11/2012
names(hd.long) <- c("patient", "level")
# summary of each patient
by(hd.long$level, hd.long$patient, summary)
## hd.long$patient: nc
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.400   5.315   5.940   6.081   6.865   9.240
## ------------------------------------------------------------
## hd.long$patient: ahd
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.400   3.400   4.050   4.242   4.400   7.480
## ------------------------------------------------------------
## hd.long$patient: ihd
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.270   5.385   5.915   6.791   7.640  14.300
# IQR and sd of each patient
by(hd.long$level, hd.long$patient, function(X) { c(IQR(X), sd(X), length(X)) })
## hd.long$patient: nc
## [1]  1.550000  1.362104 23.000000
## ------------------------------------------------------------
## hd.long$patient: ahd
## [1]  1.000000  1.302878 17.000000
## ------------------------------------------------------------
## hd.long$patient: ihd
## [1]  2.25500  2.17647 28.00000
```

```r
# 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):
cannot compute exact p-value with ties
## Warning in wilcox.test.default(hd$nc, hd$ahd, conf.int = TRUE, conf.level = 0.9833):
cannot 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):
cannot compute exact p-value with ties
## Warning in wilcox.test.default(hd$nc, hd$ihd, conf.int = TRUE, conf.level = 0.9833):
cannot compute exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  hd$nc and hd$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:
```

[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.

```
##  -530.0001  428.0000
## 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 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

[7] http://en.wikipedia.org/wiki/Resampling_(statistics)

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.

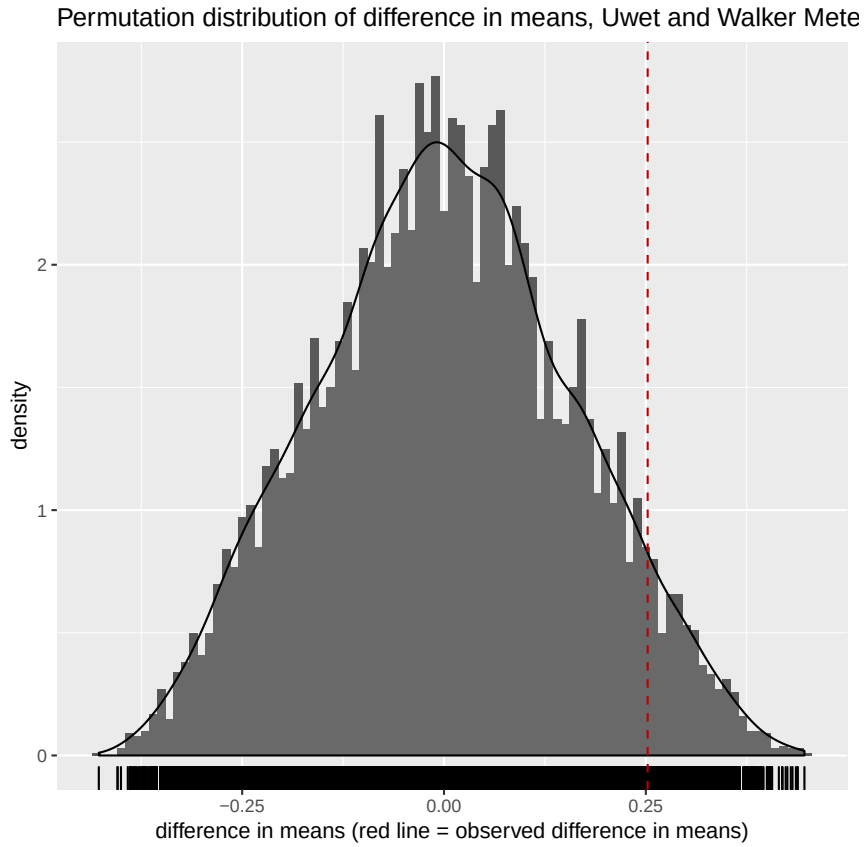
The test proceeds as follows. First, the difference in means between the two samples is calculated: this is the observed value of the test statistic, $T_{(\mathrm{obs})}$. Then the observations of groups Uwet and Walker are pooled.

```
#### 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 Meteorite
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] 0.0608
p.lower <- sum((Tperm <= -abs(Tobs))) / R
p.lower
## [1] 0.0569
p.twosided <- p.lower + p.upper
p.twosided
## [1] 0.1177
```

Note that the two-sided p-value of 0.1177 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)
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
# permutation test version
library(coin)
## Loading required package:  survival
# 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
  # examples of extracting values from coins S4 class objects
  coin::expectation(oneway.summary)
##     Uwet
## 2.875263
  coin::covariance(oneway.summary)
##           Uwet
## Uwet 0.5632881
  coin::pvalue(oneway.summary)
## [1] 0.1114144
  confint(oneway.summary)
## Error:  $ operator not defined for this S4 class
```

The permutation test gives a p-value of 0.1114 which is close to our manually calculated permuatation p-value of 0.1177.

For the emisions data, below we compare the ANOVA results (assuming normality) with a permutation test without distributional assumptions.

```
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] | year == fa

    # put p-value in matrix
    mc.pval[i1, i2] <- coin::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"
```

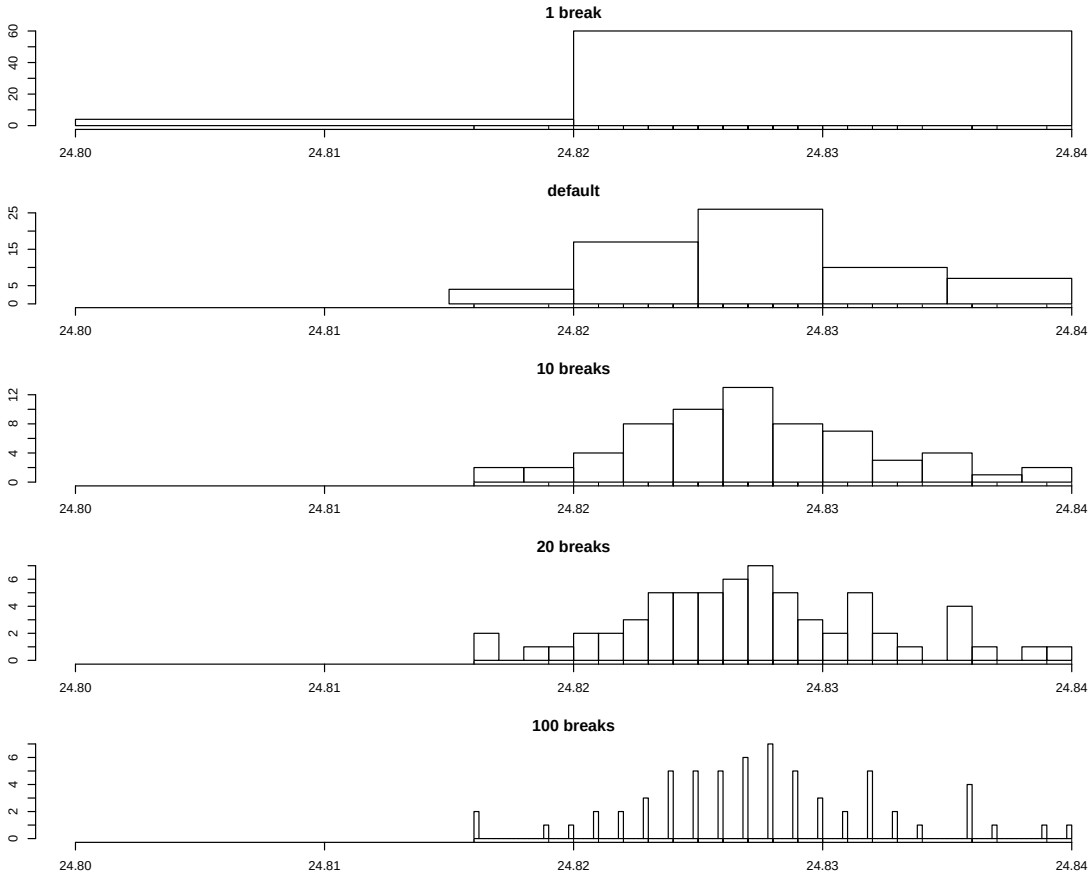
6.7 Density estimation

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)
```



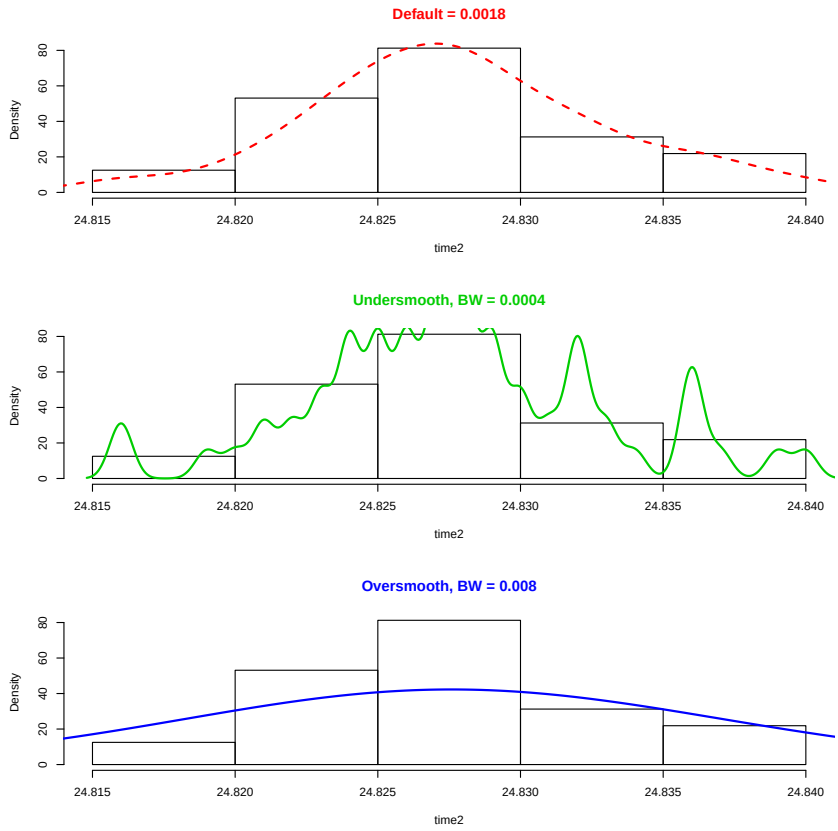
Notice that we are starting to see more and more bins that include only a single observation (or multiple observations at the precision of measurement). Taken to its extreme, this type of exercise gives in some sense a "perfect" fit to the data but is useless as an estimator of shape.

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.

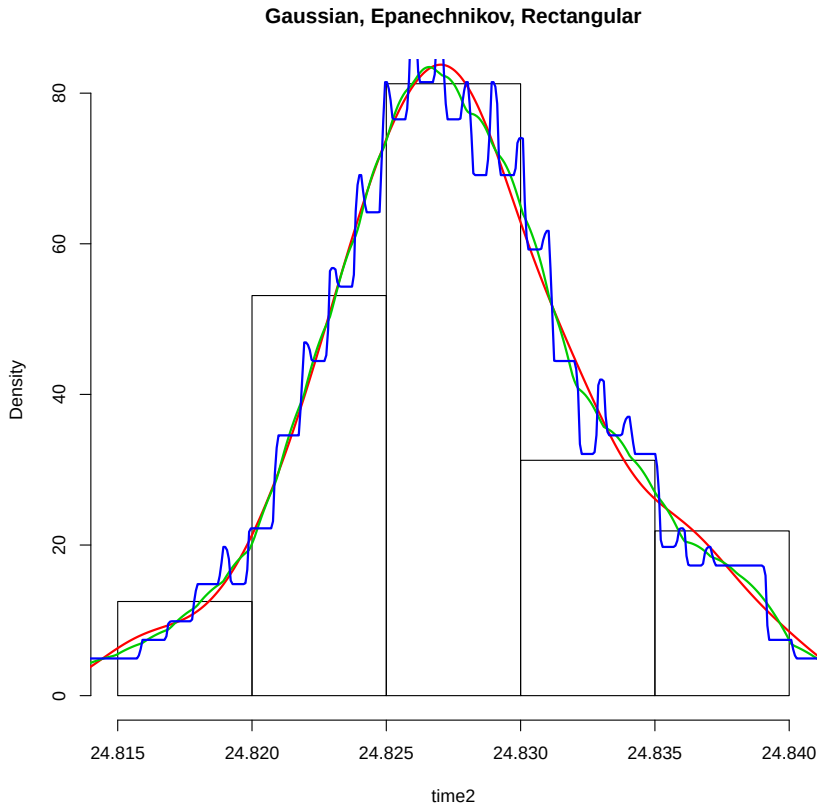
We've already used the `density()` function to provide a smooth curve to our histograms. So far, we've taken the default "bandwidth". Let's see what happens when we use different bandwidths.

```
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

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].

[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.

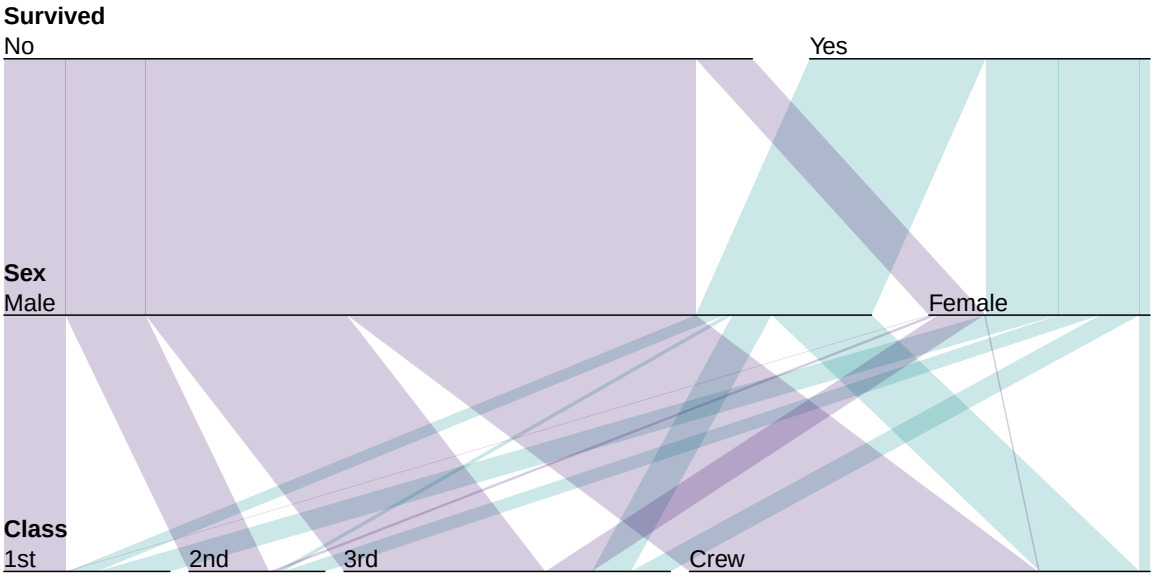
```r
# The Titanic dataset is a 4-dimensional table: Class, Sex, Age, Survived
library(datasets)
data(Titanic)

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
##
## , , 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 $\hat{p}$.

■ ClickerQs — 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})

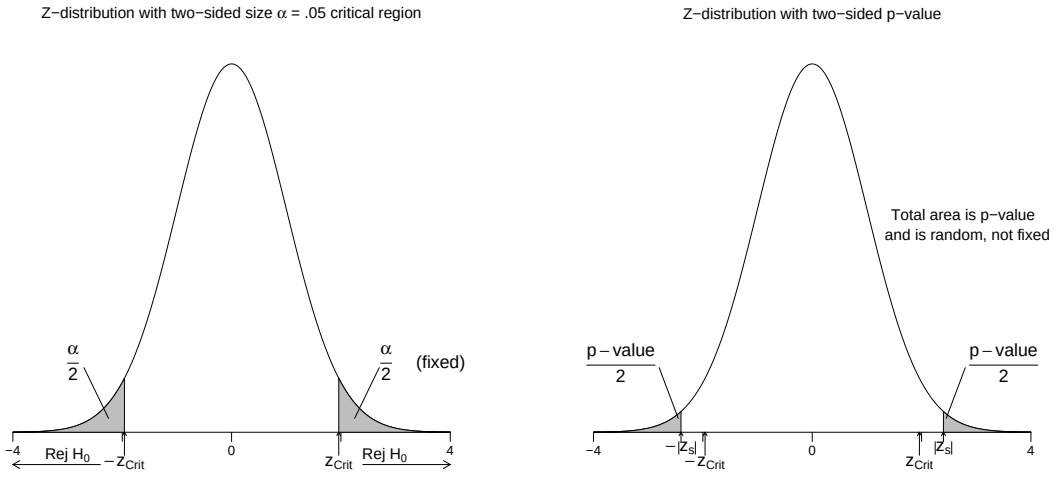
$$z_s = \frac{\hat{p} - p_0}{SE},$$

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

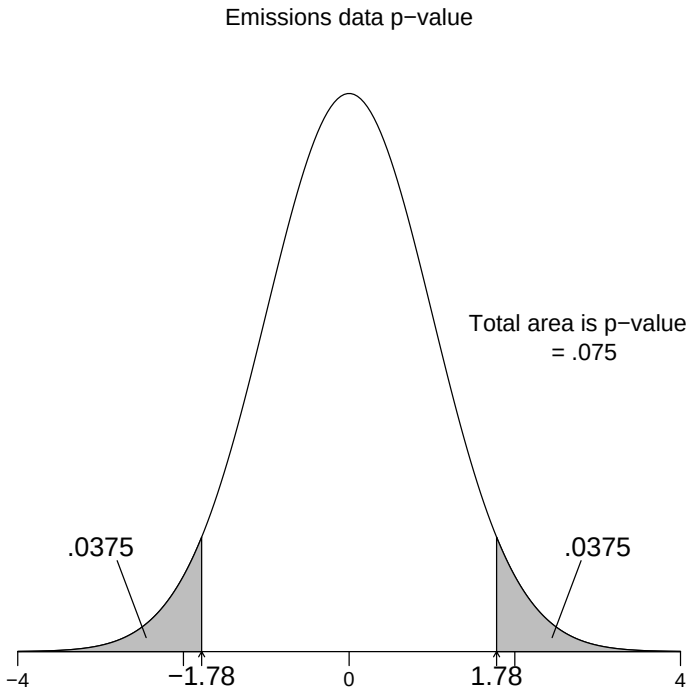
$$z_s = \frac{0.105 - 0.15}{0.02525} = -1.78,$$

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 $p_0 = 0.25$).

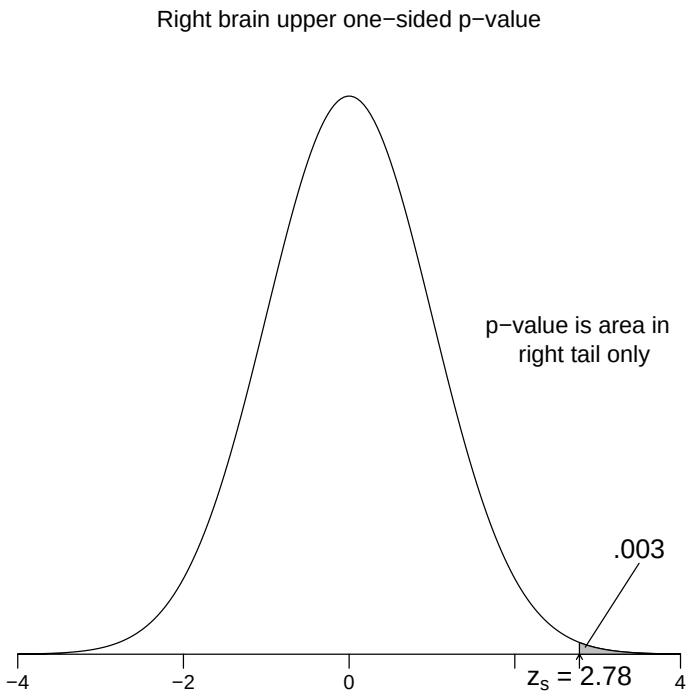
The proportion of children sampled with brains having larger right sides is $\hat{p} = 22/53 = 0.415$. The test statistic is

$$z_s = \frac{0.415 - 0.25}{0.0595} = 2.78,$$

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.

```
#### 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)



# display the data.frame
swim
```

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
```

■ CLICKERQs — 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:

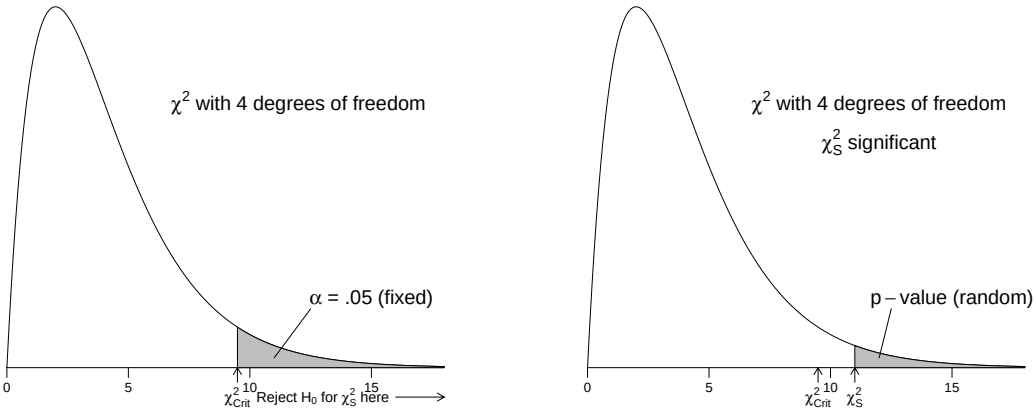
$$\chi^2_{\text{s}} = \sum_{i=1}^{r} Z_i^2,$$

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

$$\chi^2_s = (-6.48)^2 + (-7.38)^2 + (-3.45)^2 + 0.18^2 + 6.48^2 + 8.78^2 + (-1.99)^2 = 231.26$$

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)




# display the data.frame
jury
```

```
# 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
```

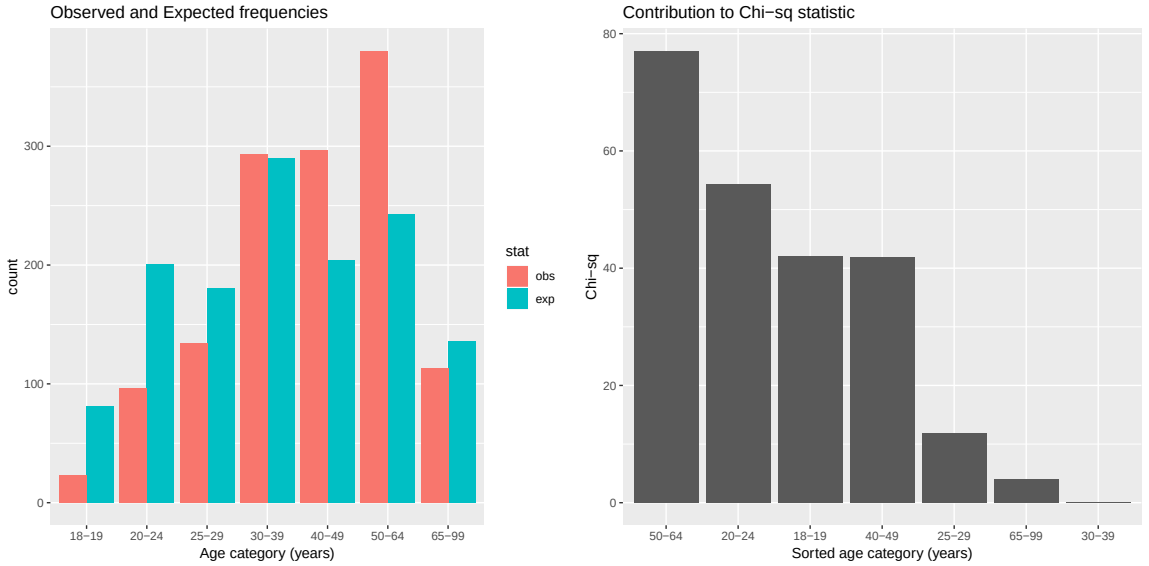
```
# use output in x.summary and create table
x.table <- data.frame(age = jury$Age
                      , obs = x.summary$observed
                      , exp = x.summary$expected
                      , res = x.summary$residuals
                      , chisq = x.summary$residuals^2
                      , stdres = x.summary$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.

```
#### Example: cervical dysplasia
# 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
```

[2] http://www.ncbi.nlm.nih.gov/pubmedhealth/PMH0002461/

```
## 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).

```
# 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
```

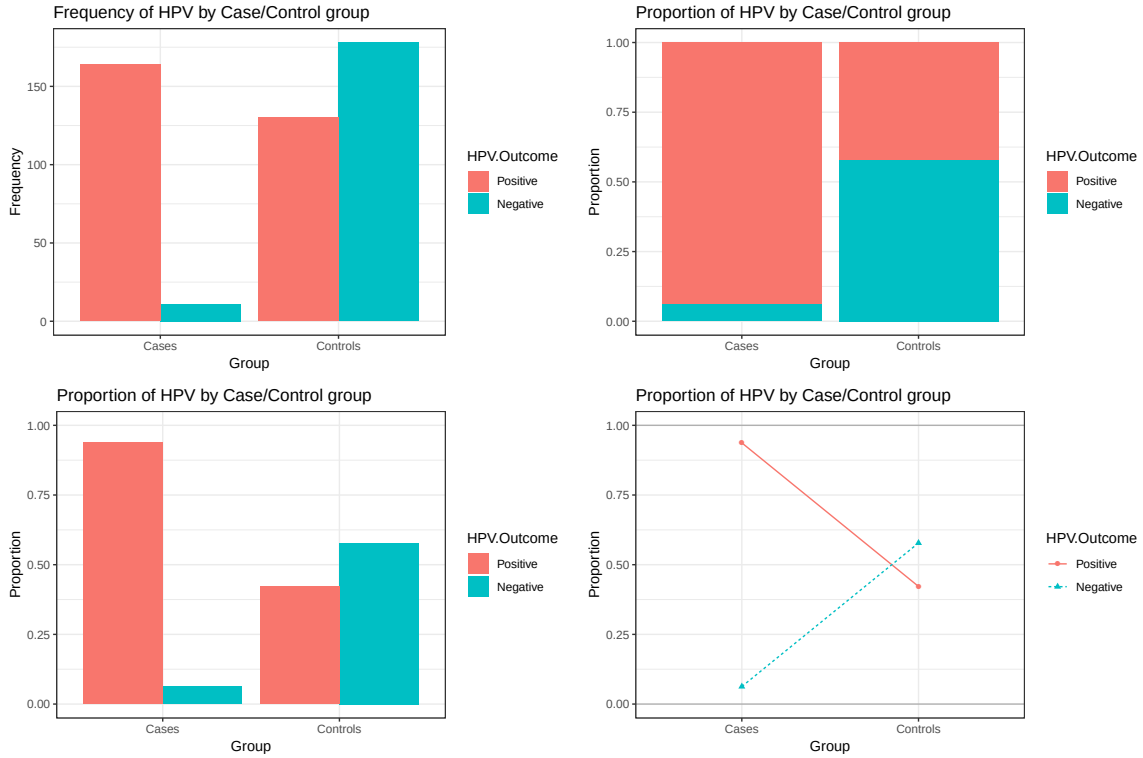
Add a column of proportions to our long-formatted data to plot these proportion values.

```
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).

```
# 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

[3]Borrowed graciously from UCLA Academic Technology Services at `http://www.ats.ucla.edu/stat/sas/faq/oratio.htm`

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, $p = 7/10 = 0.7$ 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 $p_{\mathrm{A}+} = p_{\mathrm{AA}}+p_{\mathrm{AD}}$. The population proportion that approved at the time of the second survey is $p_{+\mathrm{A}} = p_{\mathrm{AA}} + p_{\mathrm{DA}}$. The "+" sign used as a subscript means that the replaced subscript has been summed over.

Similarly, let $\hat{p}_{\mathrm{AA}}$, $\hat{p}_{\mathrm{AD}}$, $\hat{p}_{\mathrm{DA}}$, $\hat{p}_{\mathrm{DD}}$ be the sample proportion of voter-age Americans that fall into the four categories, and let $\hat{p}_{\mathrm{A}+} = \hat{p}_{\mathrm{AA}} + \hat{p}_{\mathrm{AD}}$ and $\hat{p}_{+\mathrm{A}} = \hat{p}_{\mathrm{AA}} + \hat{p}_{\mathrm{DA}}$ 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, $\hat{p}_{\mathrm{AA}} = 794/1600 = 0.496$ and $\hat{p}_{\mathrm{A}+} = 944/1600 = 0.496 + 0.094 = 0.590$. 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 $p_{\mathrm{A}+} - p_{+\mathrm{A}}$, given by $(\hat{p}_{\mathrm{A}+} - \hat{p}_{+\mathrm{A}}) \pm z_{\mathrm{crit}} SE_{\mathrm{CI}}(\hat{p}_{\mathrm{A}+} - \hat{p}_{+\mathrm{A}})$, where the CI standard error satisfies

$$SE_{\mathrm{CI}}(\hat{p}_{\mathrm{A}+} - \hat{p}_{+\mathrm{A}}) = \sqrt{\frac{\hat{p}_{\mathrm{A}+}(1-\hat{p}_{\mathrm{A}+}) + \hat{p}_{+\mathrm{A}}(1-\hat{p}_{+\mathrm{A}}) - 2(\hat{p}_{\mathrm{AA}}\hat{p}_{\mathrm{DD}} - \hat{p}_{\mathrm{AD}}\hat{p}_{\mathrm{DA}})}{n}}$$

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,

$$\begin{aligned} z_{\mathrm{crit}} SE_{\mathrm{CI}}(\hat{p}_{\mathrm{A}+} - \hat{p}_{+\mathrm{A}}) &= 1.96\sqrt{\frac{0.590 \times 0.410 + 0.550 \times 0.450 - 2(0.496 \times 0.356 - 0.094 \times 0.054)}{1600}} \\ &= 1.96 \times (0.0095) \;=\; 0.0186. \end{aligned}$$

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:

$$\mathrm{E} = \frac{\text{Row Total} \times \text{Column Total}}{\text{Total Sample Size}}.$$

2. Compute the Pearson test statistic

$$\chi^2_{\mathrm{s}} = \sum_{\text{all cells}} \frac{(O-E)^2}{E},$$

   where O is the **observed** count.
3. For a size α test, reject the hypothesis of homogeneity if $\chi^2_{\mathrm{s}} \geq \chi^2_{\mathrm{crit}}$, 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
```

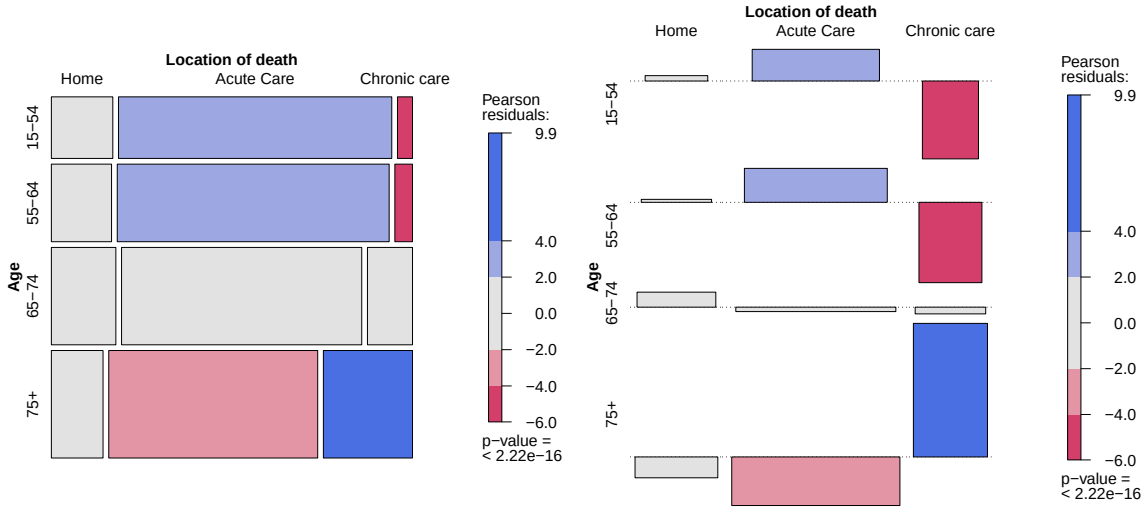
```
chisq.summary <- chisq.test(candeath, correct=FALSE)
chisq.summary
```

```
##
##  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)
```

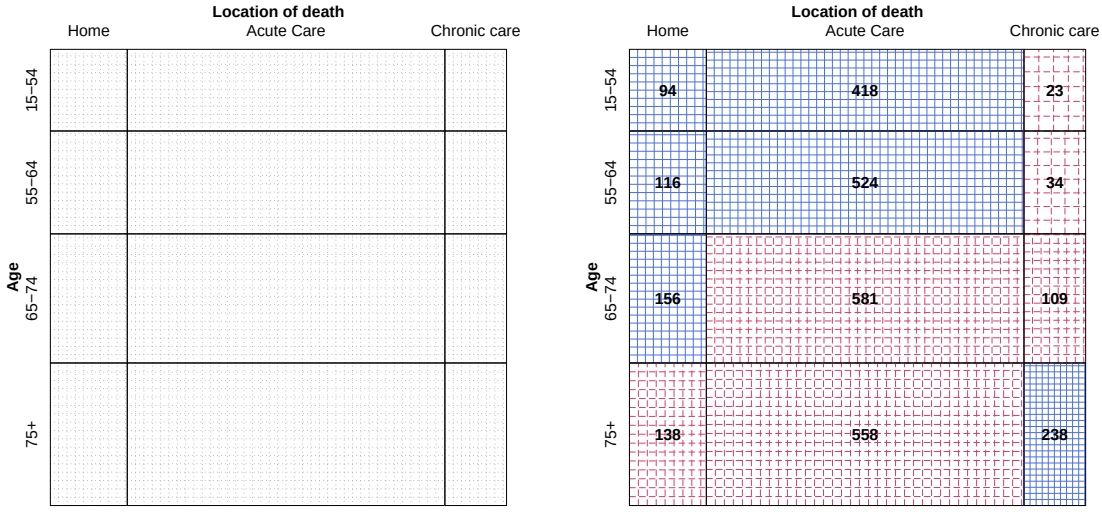


The `vcd` package provides a variety of methods for visualizing multivariate categorical data, inspired by Michael Friendly's wonderful "Visualizing Categorical Data". For more details, see The Strucplot Framework[4].

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 homo-

geneity 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

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.

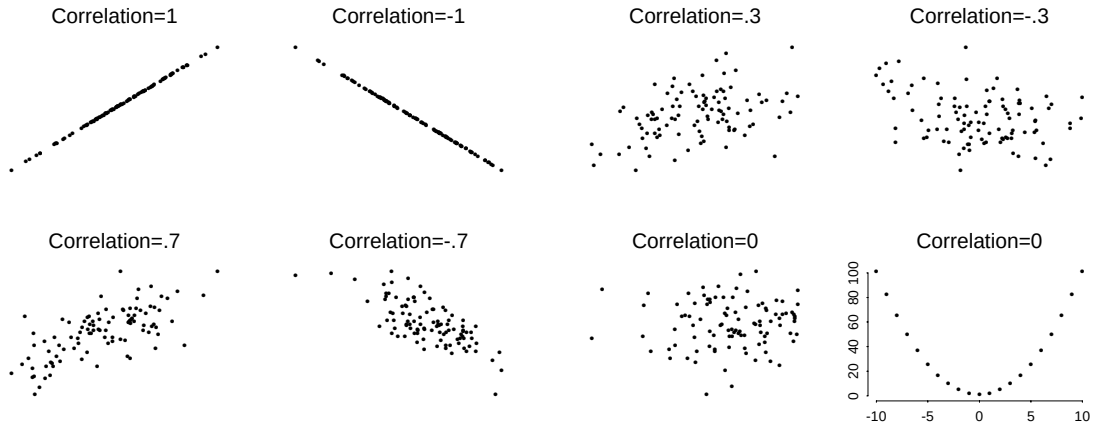
Important properties of r:

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 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, $\log(y) = (\log(2))x = 0.30x$.

[1] From: Gelman, Andrew and Deborah Nolan (2002). *Teaching statistics: A bag of tricks.* Oxford University Press.

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

$$\text{area} \quad = \quad (\text{circumference}/4)^2.$$

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

$$\text{surface area} \quad = \quad 6(\text{volume})^{2/3},$$

or, on the logarithmic scale,

$$\log(\text{surface area}) \quad = \quad \log(6) + (2/3)\log(\text{volume}).$$

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.

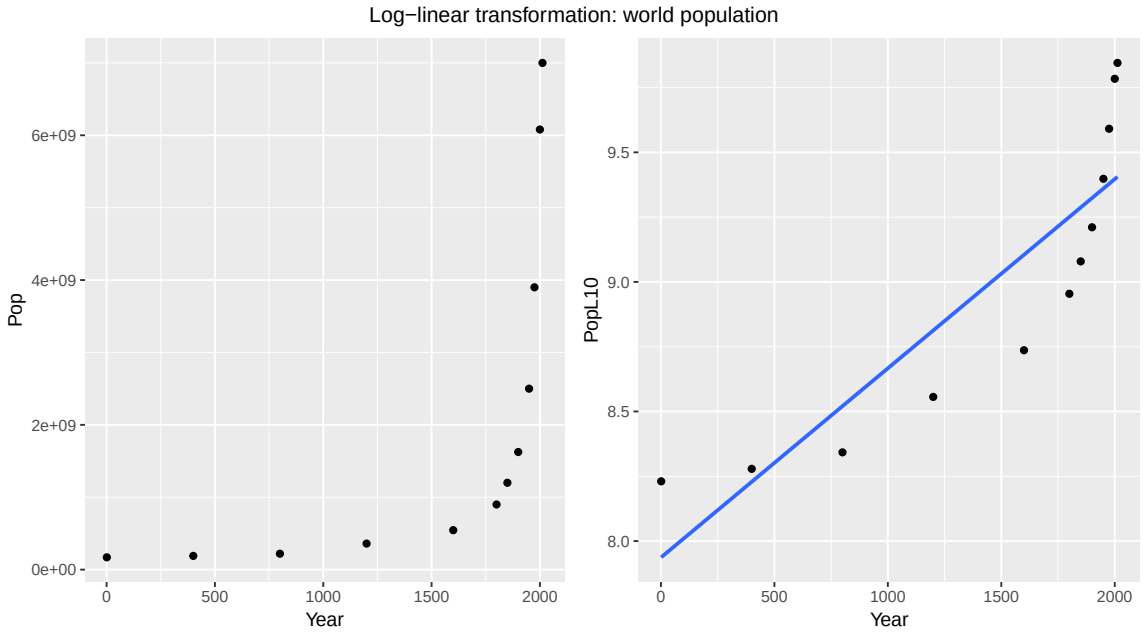
On the raw scale, all you can see is that the population has increased very fast recently. On the log scale, convex curvature is apparent — that is, the rate of increase has itself increased. On the logarithmic graph, the least-squares regression line is drawn. The estimated world population has been 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, 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
```

[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

```
## 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)
```

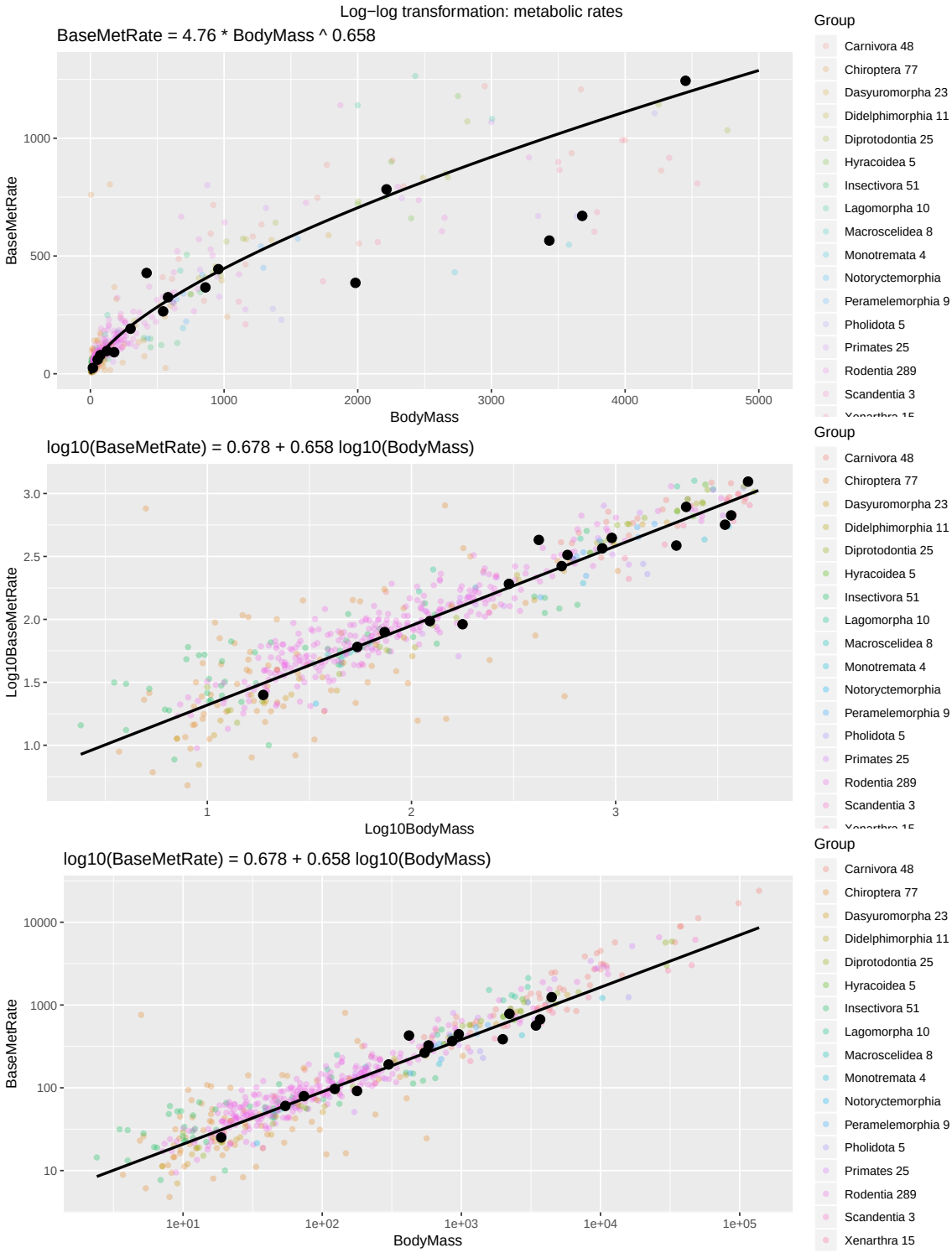
```
## 'data.frame': 634 obs. of  9 variables:
##  $ Group           : Factor w/ 18 levels "Artiodactyla 7",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Genus           : Factor w/ 88 levels "","Acrobatidae",..: 1 12 12 12 12 12 12 12 12 31 ...
##  $ Species         : Factor w/ 621 levels "","2n = 52","2n = 54",..: 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)", sep = ""))
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)", sep = ""))
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).
```



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

$$\log(\text{BaseMetRate}) \quad = \quad 0.678 + 0.658 \log(\text{BodyMass}).$$

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

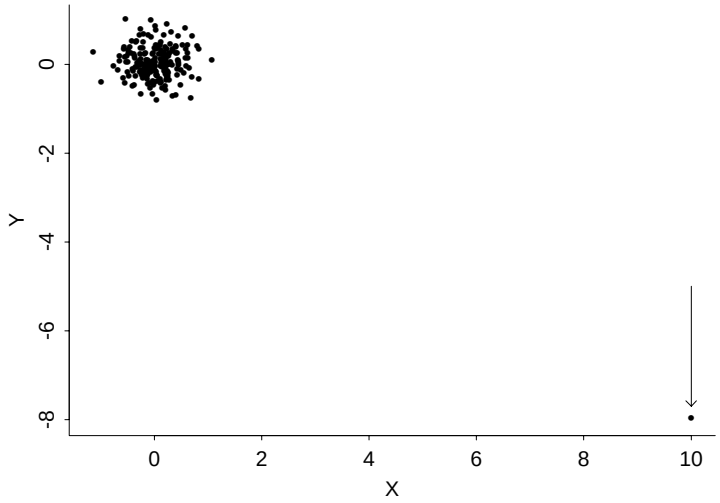
$$t_s = r\sqrt{\frac{n-2}{1-r^2}},$$

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.

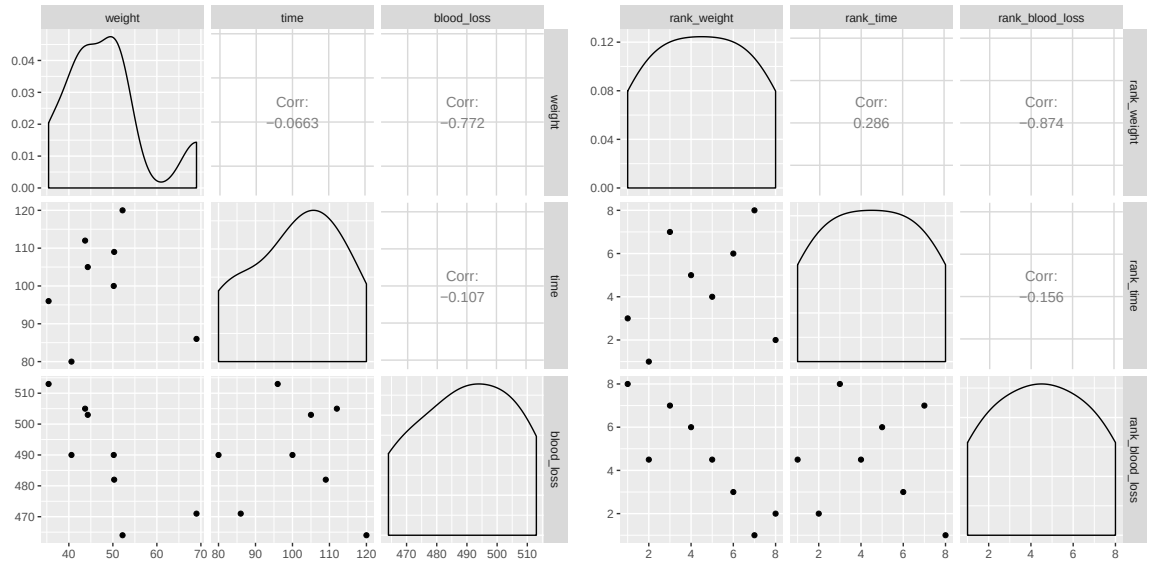
```
s.corr <- cor(thyroid, method = "spearman");
#s.corr

# 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], progress=FALSE)
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], progress=FALSE)
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.

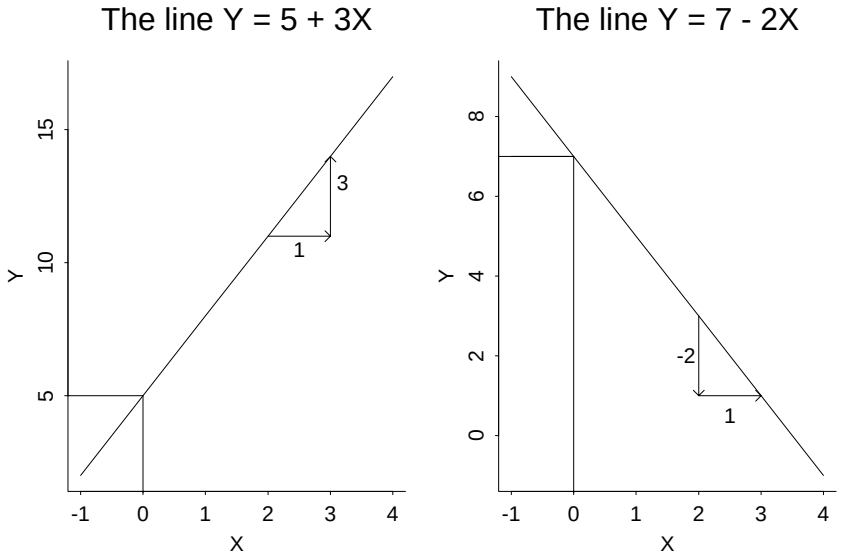
■ CLICKER Qs — Coney Island STT.02.02.050 ■

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 ■

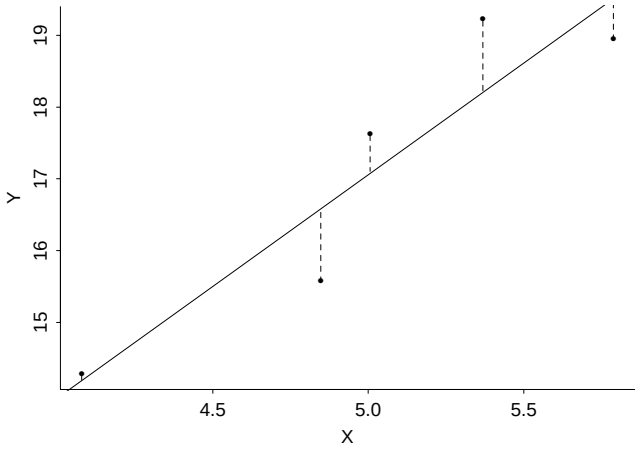
8.4.2 Least Squares

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).

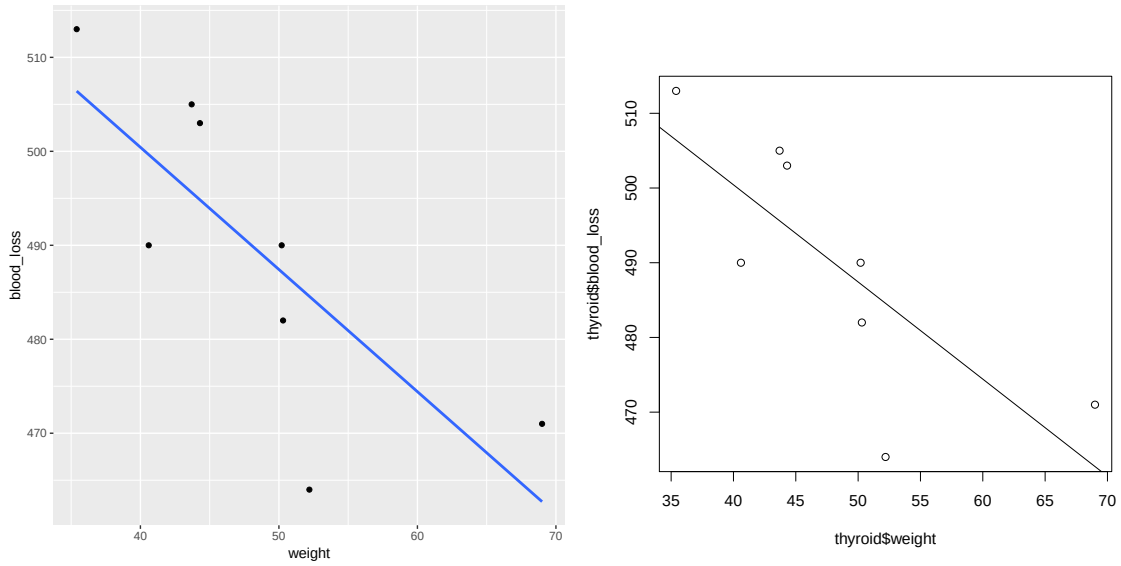
Special symbols b_0 and b_1 identify the LS intercept and slope to distinguish the LS line from the generic line $Y = \beta_0 + \beta_1 X$. You should think of $\hat{Y}$ as the **fitted value** at X, or the value of the LS line at X.

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

$$\text{Residual Sums of Squares} = \sum_{i=1}^{n}\{Y_i - (b_0 + b_1 X_i)\}^2.$$

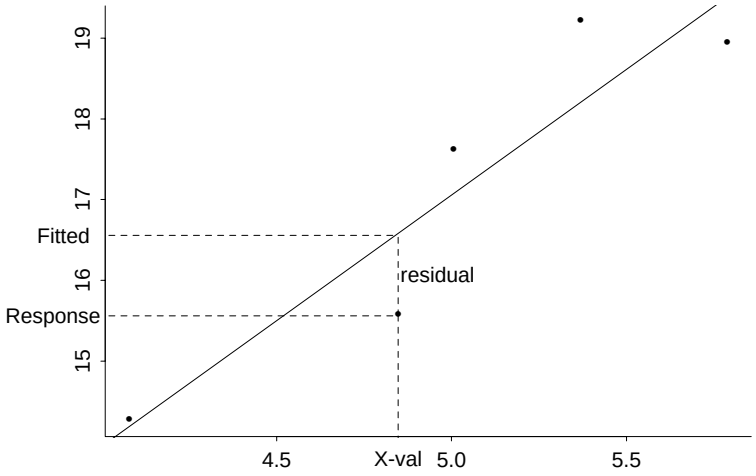
Several bits of notation are needed. Let

$$\hat{Y}_i = b_0 + b_1 X_i$$

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

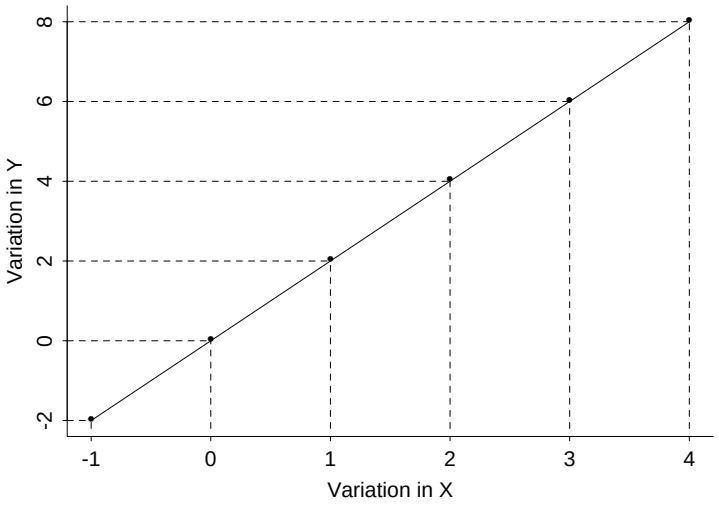
$$R^2 = \text{coefficient of determination} = \frac{\text{Reg SS}}{\text{Total SS}}.$$

Given the definitions of the Sums of Squares, we can show $0 \leq R^2 \leq 1$ and

$$R^2 = \text{square of Pearson correlation coefficient } = r^2.$$

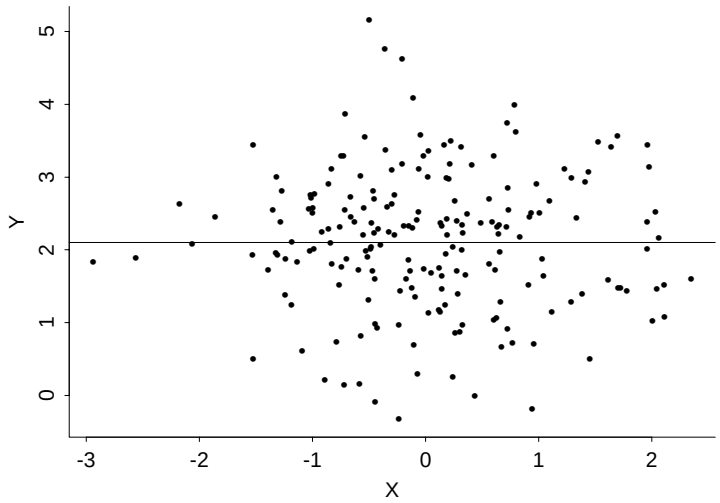
To understand the interpretation of R^2, at least in two extreme cases, note that

$$\begin{aligned}
\text{Reg SS} = \text{Total SS} &\Leftrightarrow \text{Res SS} = 0 \\
&\Leftrightarrow \text{all the data points fall on a straight line} \\
&\Leftrightarrow \text{all the variability in } Y \text{ is explained by the linear relationship with } X \\
&\quad\ \ \text{(which has variation)} \\
&\Leftrightarrow R^2 = 1. \quad \text{(see the picture below)}
\end{aligned}$$



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

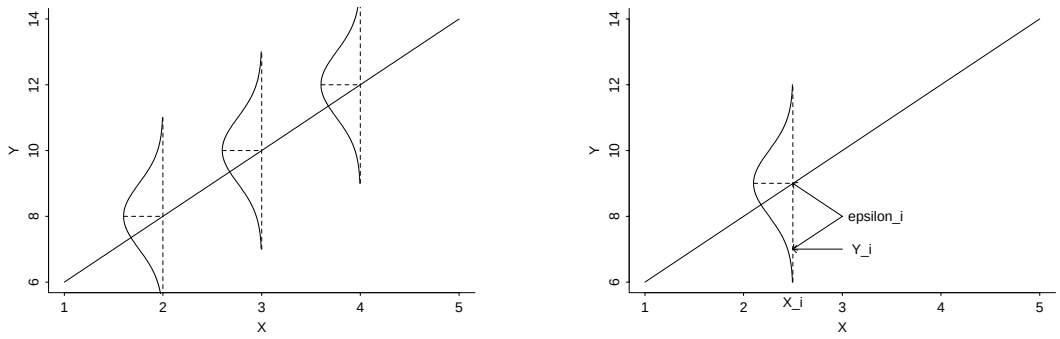
1. The mean in the population of all responses Y at a given X value (sometimes called $\mu_{Y|X}$) falls on a straight line, $\beta_0 + \beta_1 X$, called the population regression line.
2. The variation among responses Y at a given X value is the same for each X, and is denoted by $\sigma^2_{Y|X}$.
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

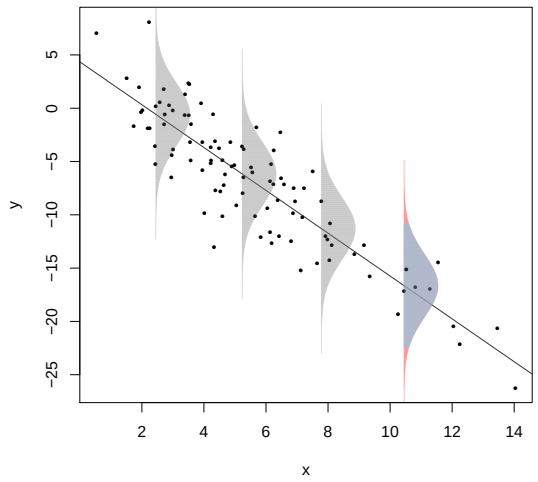
$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

(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 $x_i \sim$ Gamma$(3, 0.5)$ and $e_i \sim$ 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.

```
##
## Attaching package:  'scales'
## The following object is masked from 'package:coin':
##
##    pvalue
```



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!

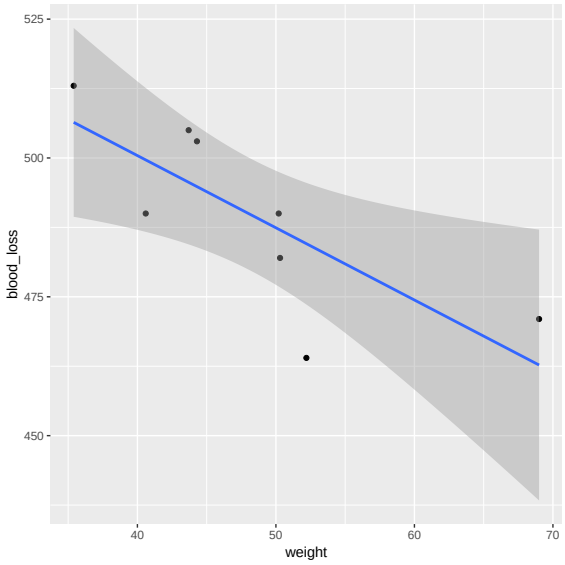
```
# CI for the mean and PI for a new observation at weight=50
predict(lm.blood.wt, data.frame(weight=50), interval = "confidence", level = 0.95)
##        fit      lwr      upr
## 1 487.4257 477.1575 497.6938
predict(lm.blood.wt, data.frame(weight=50), interval = "prediction", level = 0.95)
##        fit     lwr      upr
## 1 487.4257 457.098 517.7533
```

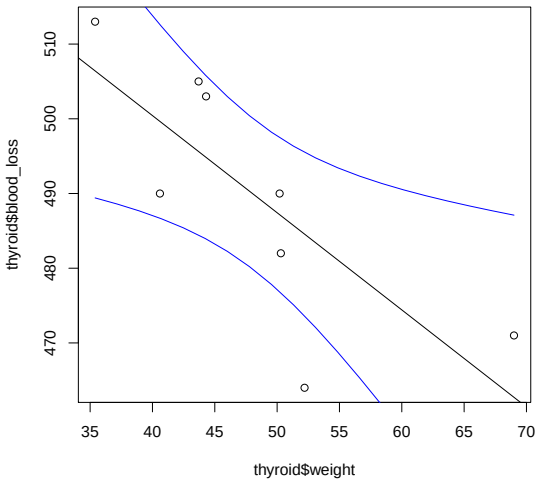
Comments

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.

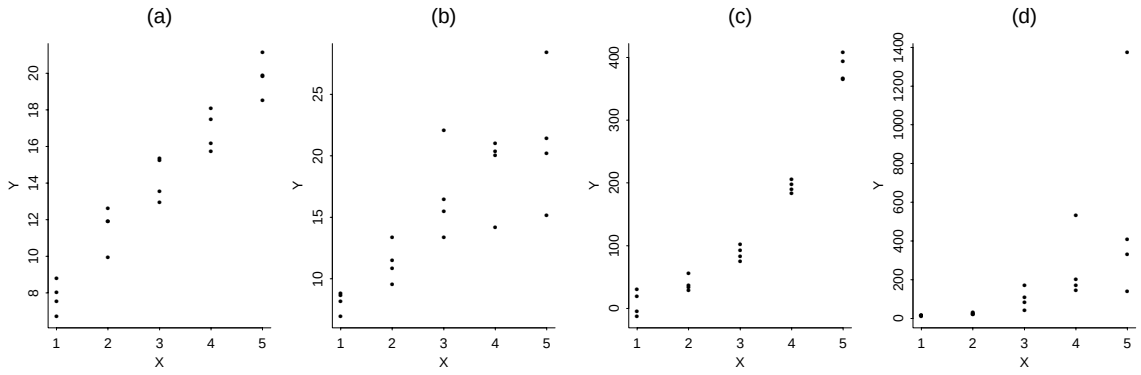


Figure (a) is the only plot that is consistent with the assumptions. The plot shows a linear relationship with constant variance. The other figures show one or more deviations. Figure (b) shows a linear relationship but the variability increases as the mean level increases. In Figure (c) we see a nonlinear 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

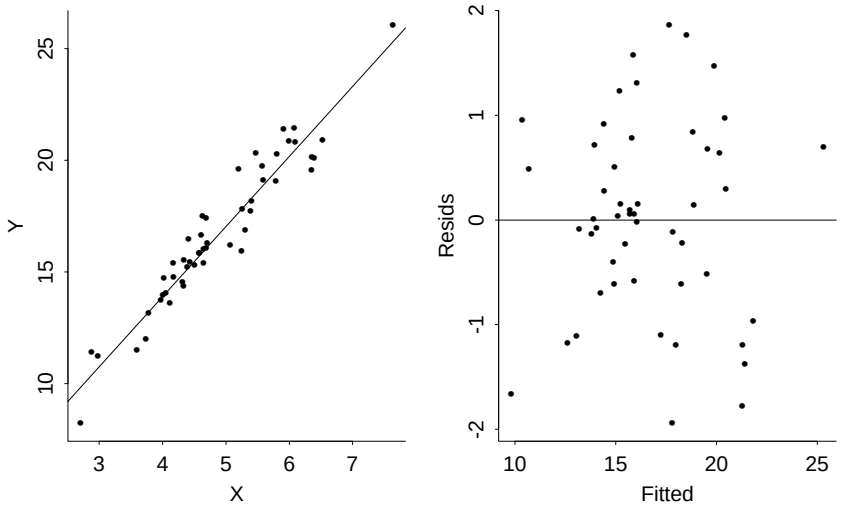
$$\text{SE}(\hat{Y}_i) \;=\; \text{SE}(e_i) \;=\; s_{Y|X}\sqrt{\frac{1}{n} + \frac{(X_i - \bar{X})^2}{\sum_j (X_j - \bar{X})^2}}.$$

So many statisticians prefer to plot the **studentized residuals** (sometimes called the standardized residuals or internally Studentized residual)

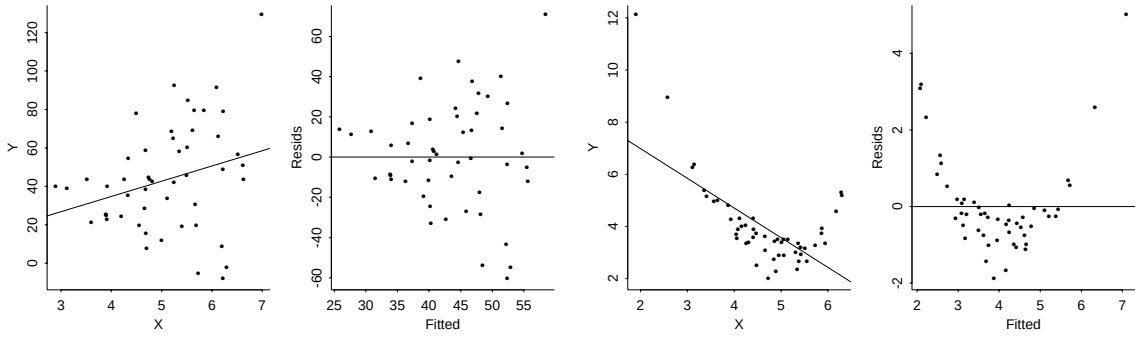
$$r_i = \frac{e_i}{SE(e_i)}.$$

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:

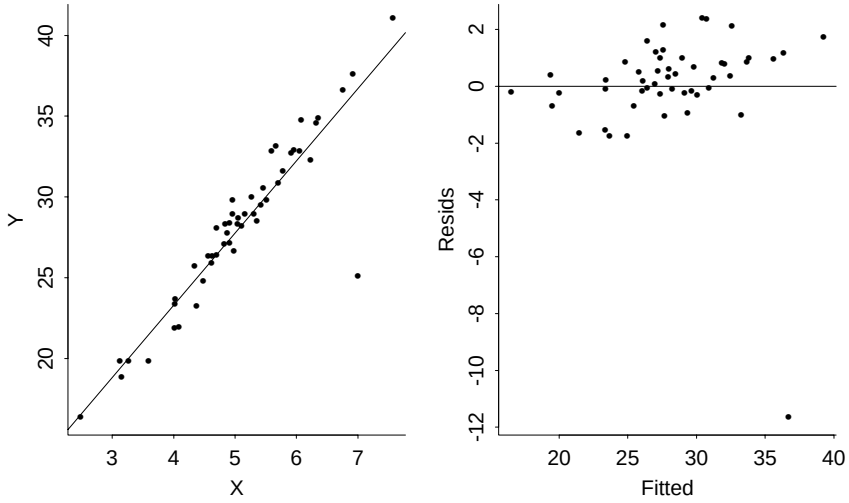


The following sequence of plots show how inadequacies in the data plot appear in a residual plot. The first plot shows a roughly linear relationship between Y and X with non-constant variance. The residual plot shows a megaphone shape rather than the ideal horizontal band. A possible remedy is a **weighted 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

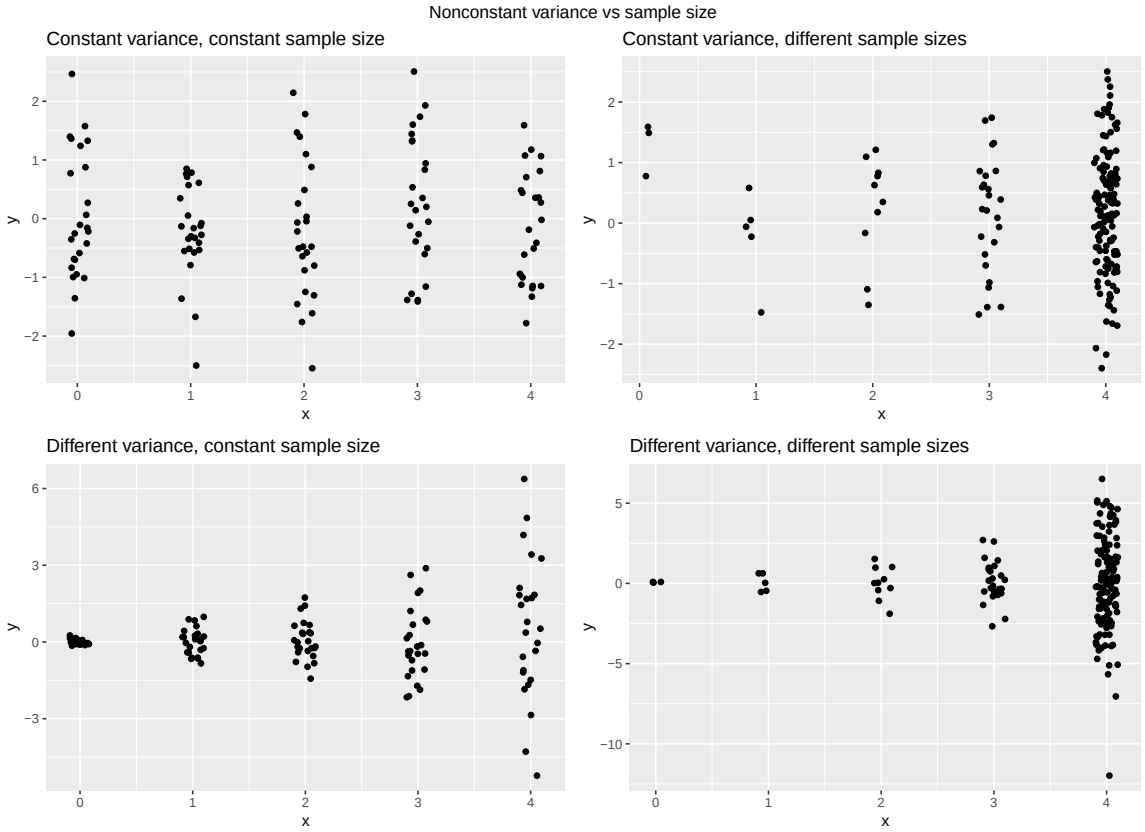
without using this observation to construct the fit. It is quite possible for the deleted residual to be huge when the raw residual is tiny.

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.

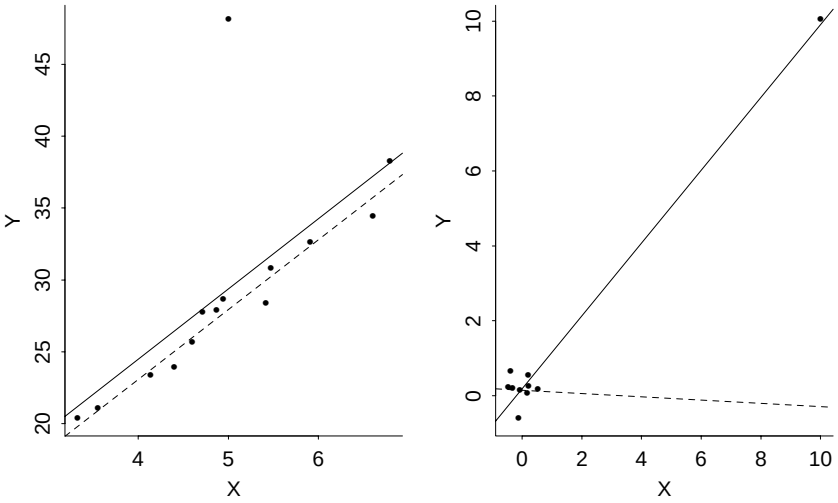
What do you do with outliers? Outliers may be due to incorrect recordings of the data or failure of the measuring device, or indications or a change in the mean or variance structure for one or more cases. Incorrect recordings should be fixed if possible, but otherwise deleted from the analysis.

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.

■ CLICKER QS — 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)`.

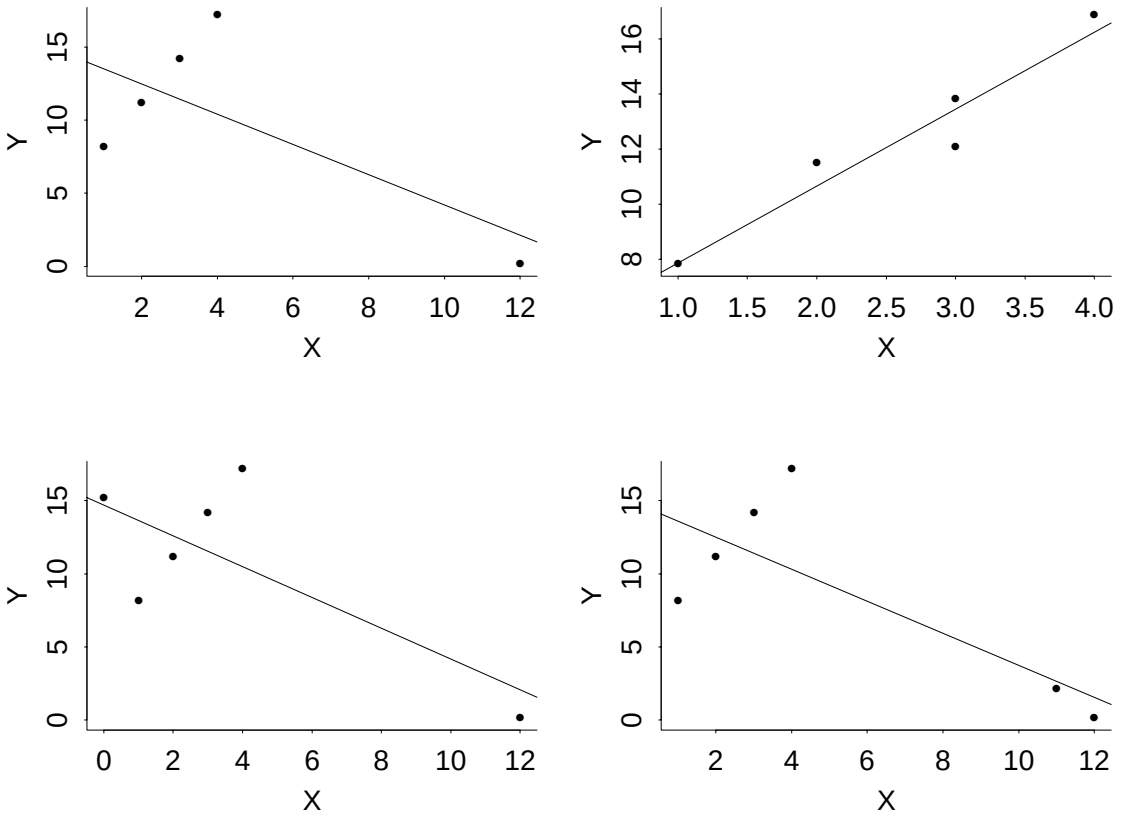
Given a regression problem, you should locate the points with the largest D_js and see whether holding these cases out has a decisive influence on the fit of the model or the conclusions of the analysis. You can examine the relative magnitudes of the D_js across cases without paying much attention to 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 misrecorded 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

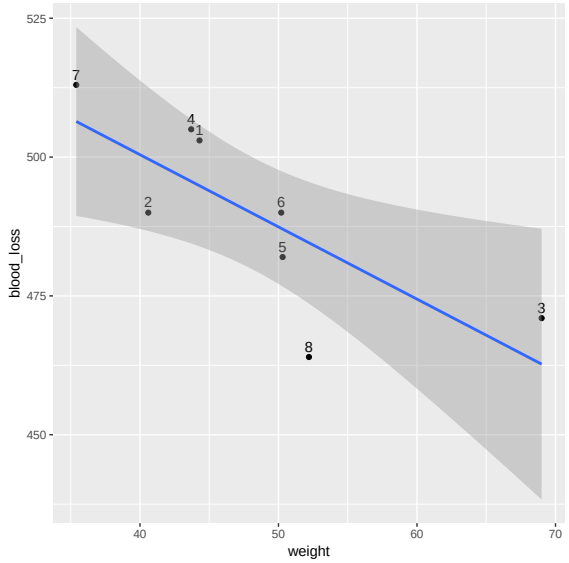
We looked at much of this before, but let us go through the above steps systematically. Recall the data set (we want to predict blood loss from weight):

	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 *
## ---
```

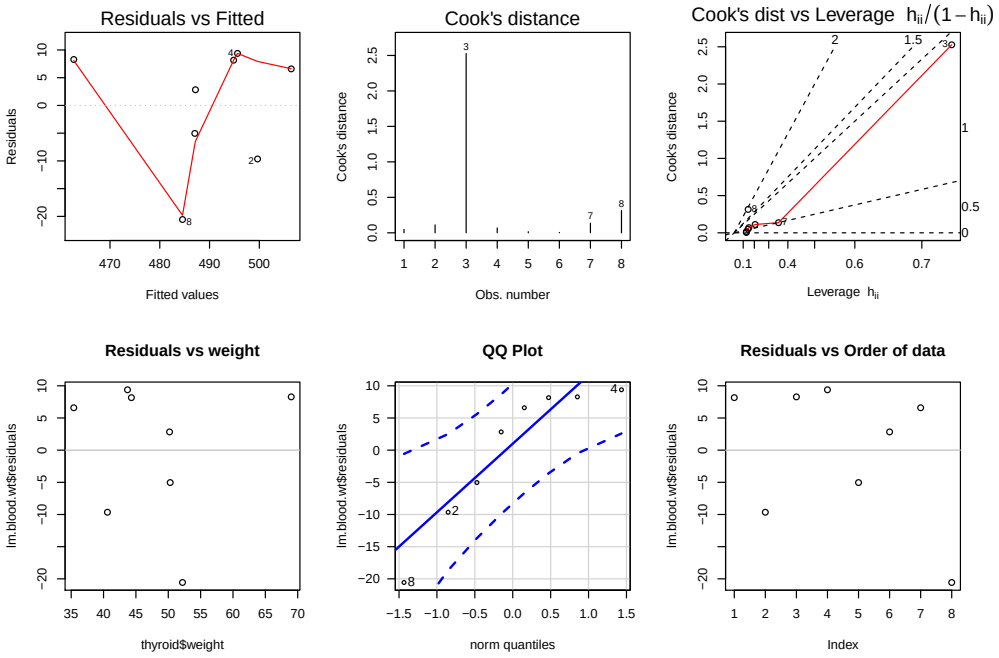
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.66 on 6 degrees of freedom
## Multiple R-squared:  0.5967,Adjusted R-squared:  0.5295
## F-statistic: 8.878 on 1 and 6 DF,  p-value: 0.02465
```

(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 = list(n = 3), main="QQ Plot")
```

```
## [1] 8 2 4
```

```
# 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.

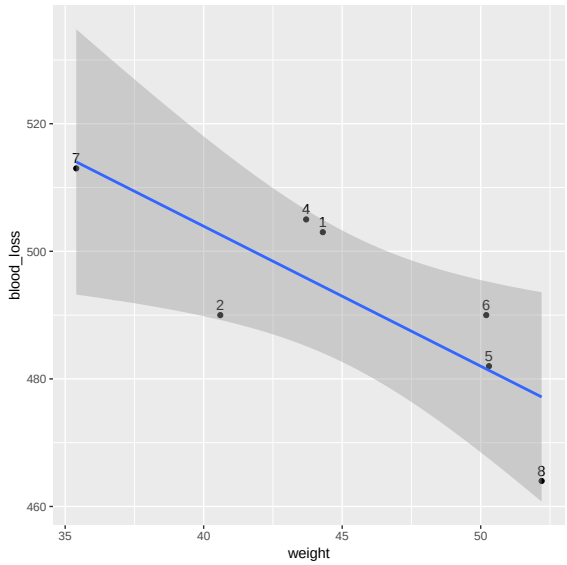
```
# wt = 1 for all except obs 3 where wt = 0
thyroid$wt <- as.numeric(!(thyroid$id == 3))
thyroid$wt

## [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)
```

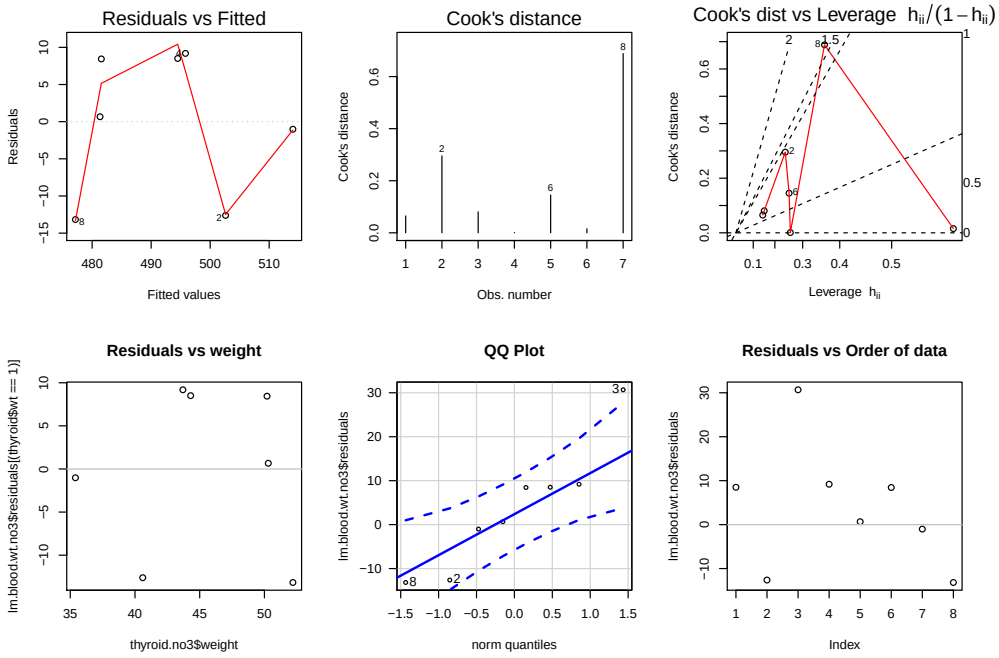


Nothing very striking shows up in the residual plots, and no Cook's D values are very large among the remaining observations.

```
# 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 = list(n = 3), main="QQ Plot")
```

```
## [1] 3 8 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

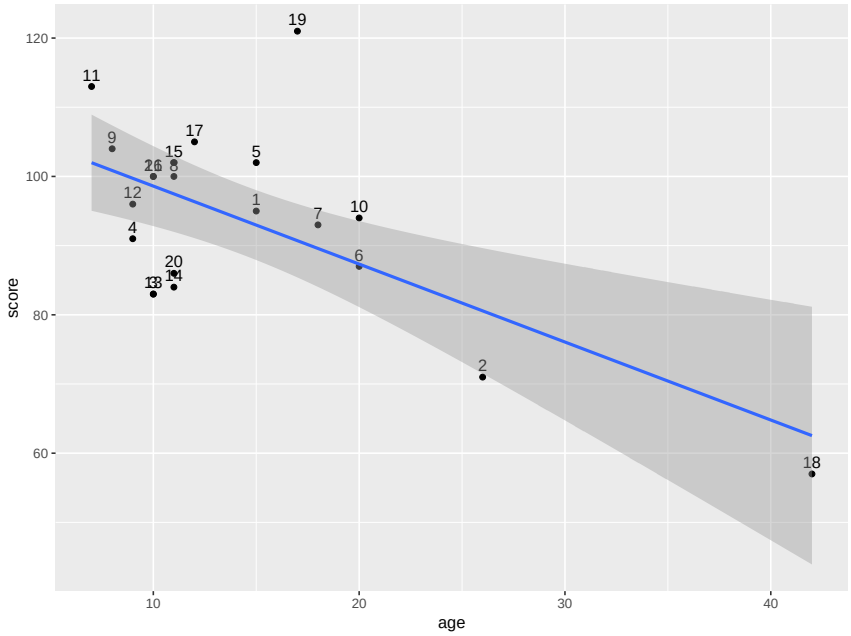
These data are from a UCLA study of cyanotic heart disease in children. The predictor is the age of the child in months at first word and the response variable is the Gesell adaptive score, for each of 21 children.

	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)
```

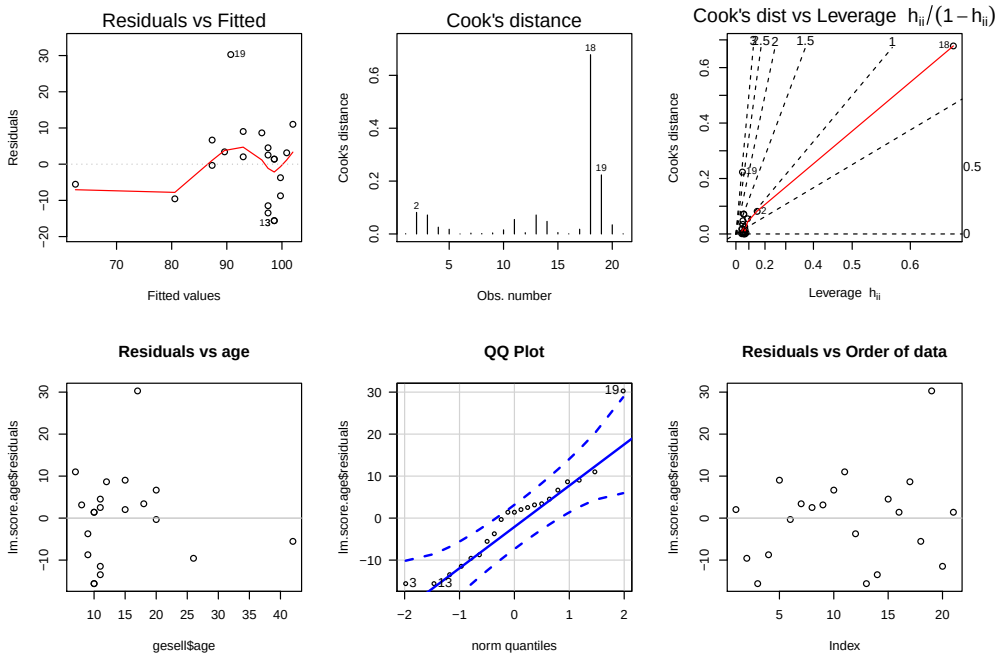


2. There are no obvious transformations to try here.
3. Fit a simple linear regression model. Provide an equation for the LS line. Does age at first word appear to be an "important predictor" of Gesell adaptive score? (i.e., is the estimated slope significantly different from zero?)

```
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 = list(n = 3), main="QQ Plot")
## [1] 19  3 13

# 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

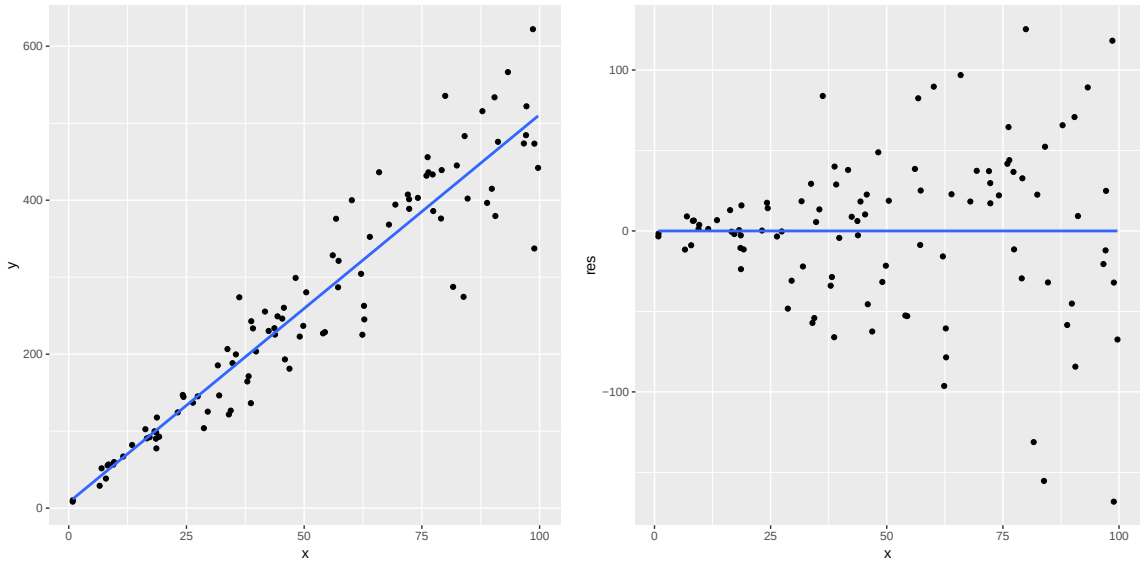
$$\sum_{i=1}^{n} w_i\{Y_i - (\beta_0 + \beta_1 X_i)\}^2$$

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.

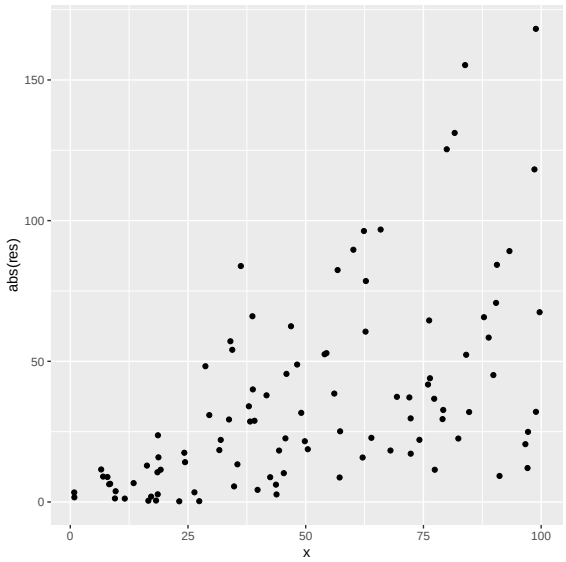
```
#### Weighted Least Squares
# R code to generate data
set.seed(7)
n <- 100
# 1s, Xs uniform 0 to 100
X <- matrix(c(rep(1,n),runif(n,0,100)), ncol=2)
# intercept and slope (5, 5)
beta <- matrix(c(5,5),ncol=1)
# errors are X*norm(0,1), so variance increases with X
e <- X[,2]*rnorm(n,0,1)
# response 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.

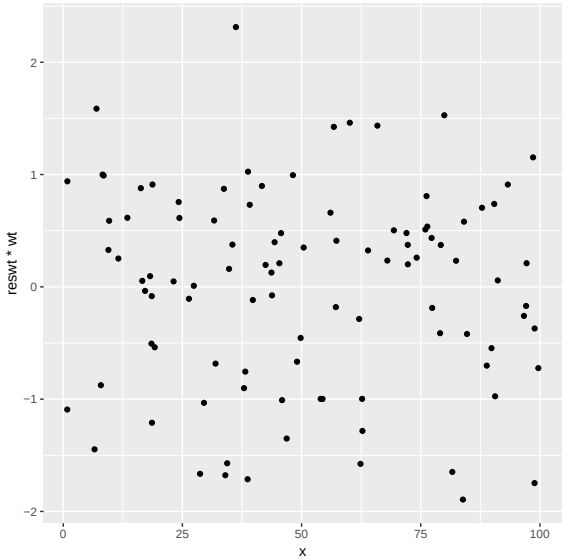
```
# ggplot: Plot the absolute value of the residuals
library(ggplot2)
p <- ggplot(wlsdat, aes(x = x, y = abs(res)))
p <- p + geom_point()
print(p)
```



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 (nonstandardized) 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

ADA1: Additional topics

Chapter 9

Introduction to the Bootstrap

Contents

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.

	(fatal plus non-fatal) heart attacks	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.

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.

[1] Efron (1979), "Bootstrap methods: another look at the jackknife." Ann. Statist. 7, 1–26

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

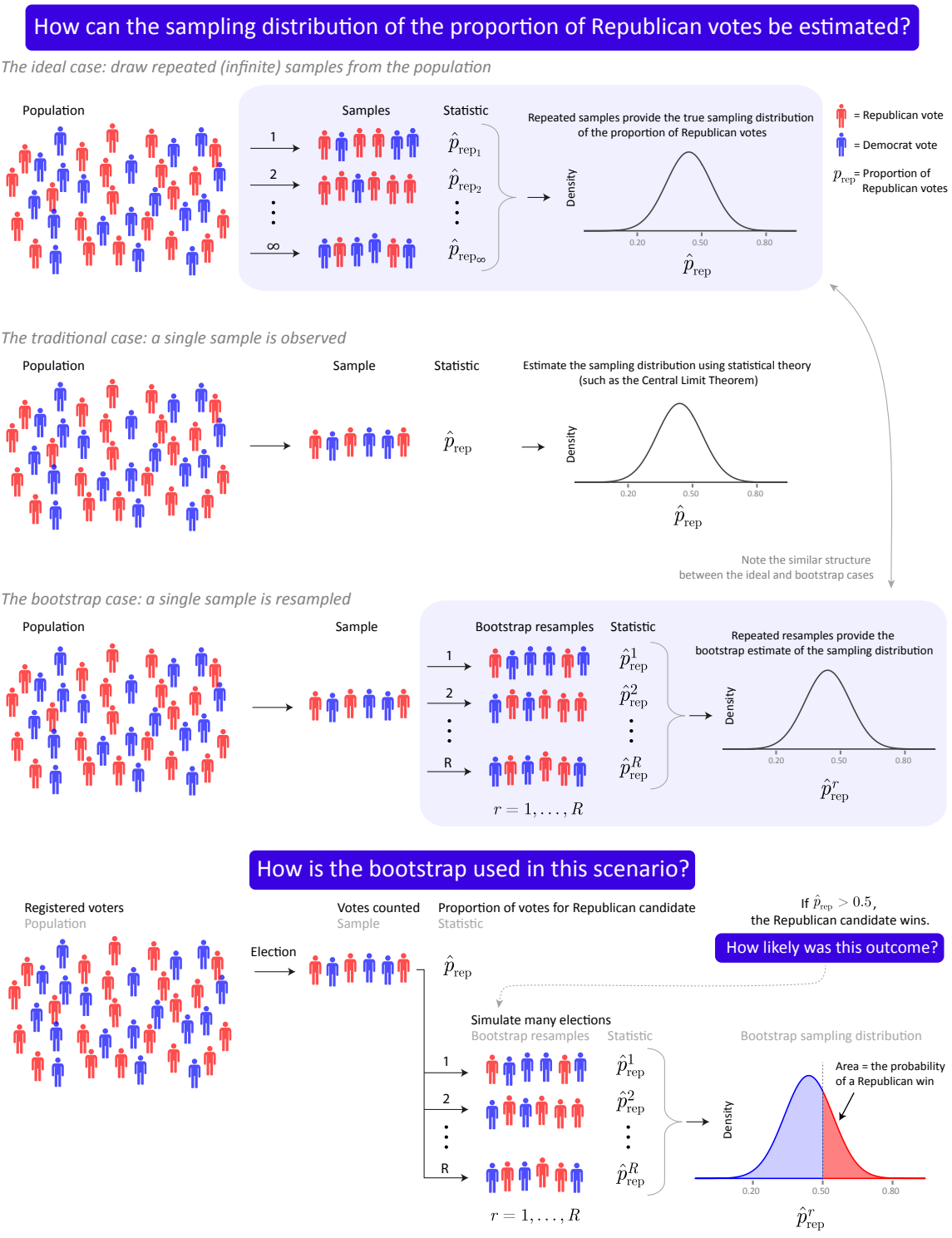
Ideal world

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
Population
Samples
Statistic
Repeated samples provide the true sampling distribution of the proportion of Republican votes
= Republican vote
= Democrat vote
p_{rep} = Proportion of Republican votes
Density
The traditional case: a single sample is observed
Population
Sample
Statistic
Estimate the sampling distribution using statistical theory (such as the Central Limit Theorem)
Note the similar structure between the ideal and bootstrap cases
The bootstrap case: a single sample is resampled
Population
Sample
Bootstrap resamples
Statistic
Repeated resamples provide the bootstrap estimate of the sampling distribution
$r = 1, \ldots, R$
How is the bootstrap used in this scenario?
Registered voters
Population
Election
Votes counted
Sample
Proportion of votes for Republican candidate
Statistic
If $\hat{p}_{rep} > 0.5$, the Republican candidate wins.
How likely was this outcome?
Simulate many elections
Bootstrap resamples
Statistic
Bootstrap sampling distribution
Area = the probability of a Republican win

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.

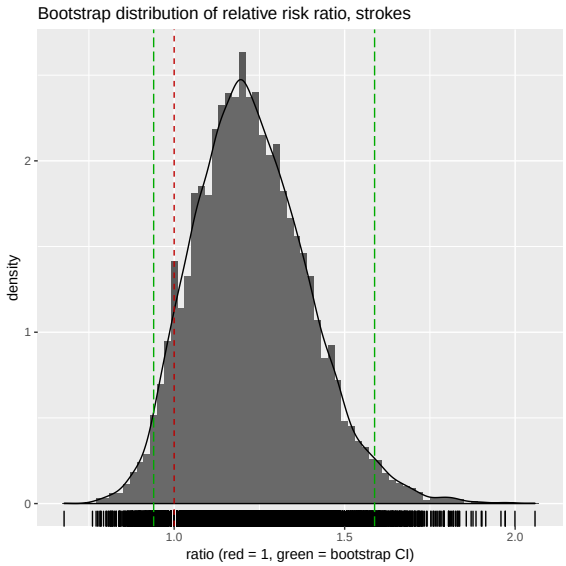
We draw with replacement a sample of 11037 items from the first population, and a sample of 11034 items from the second population. Each is called a *bootstrap sample*. From these we derive the bootstrap replicate of $\hat{\theta}$:

$$\hat{\theta}^* = \frac{\text{Proportion of ones in bootstrap sample 1}}{\text{Proportion of ones in bootstrap sample 2}}.$$

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
```

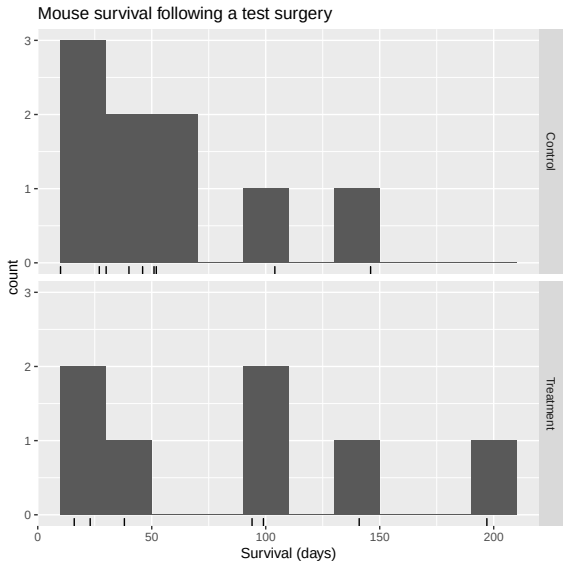
```
mice.summary
```

```
##       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.

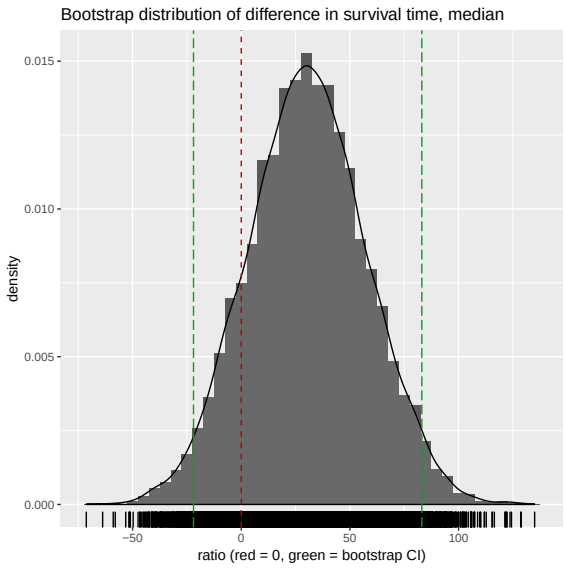
Example: Mouse survival, two-sample bootstrap, mean Here's how the bootstrap works in the two-sample mouse example. We draw with replacement from each sample, calculate the mean for each sample, then take the difference in means. Each is called a *bootstrap sample* of the difference in means. From these 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)
```



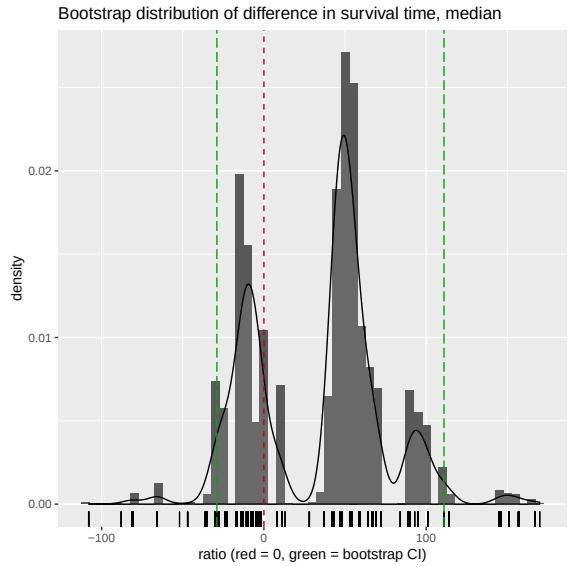
Example: Mouse survival, two-sample bootstrap, median For most statistics (such as the median) we don't have a formula for the limiting value of the standard error, but in fact no formula is needed. Instead, we use the numerical output of the bootstrap program. The summaries are in the code, followed 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
sort(treatment)
## [1]  16  23  38  94  99 141 197
# 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
# 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
```

```
## 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)
```



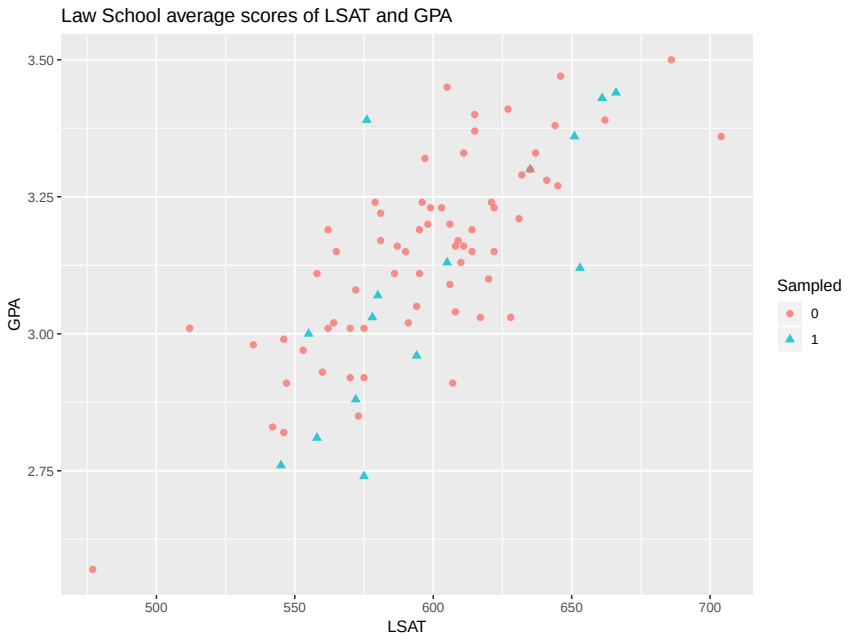
9.2.3 Comparing bootstrap sampling distribution from population and sample

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

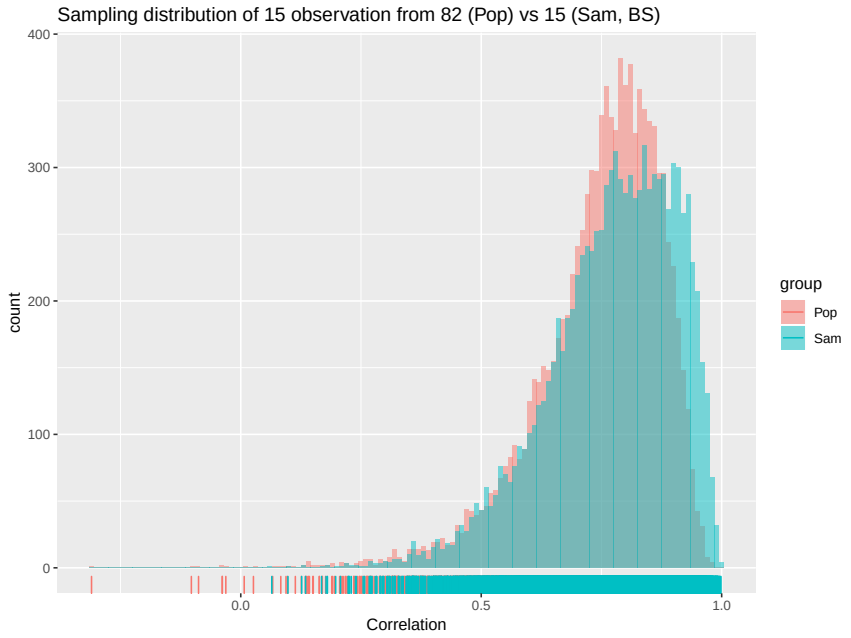
by drawing 15 random schools from the population of 82 schools and calculating the correlation. If the bootstrap works well (from our hopefully representative sample of 15), then the bootstrap sampling distribution from the 15 schools will be close to the true sampling distribution.

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)
```



Chapter 10

Power and Sample size

Contents

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

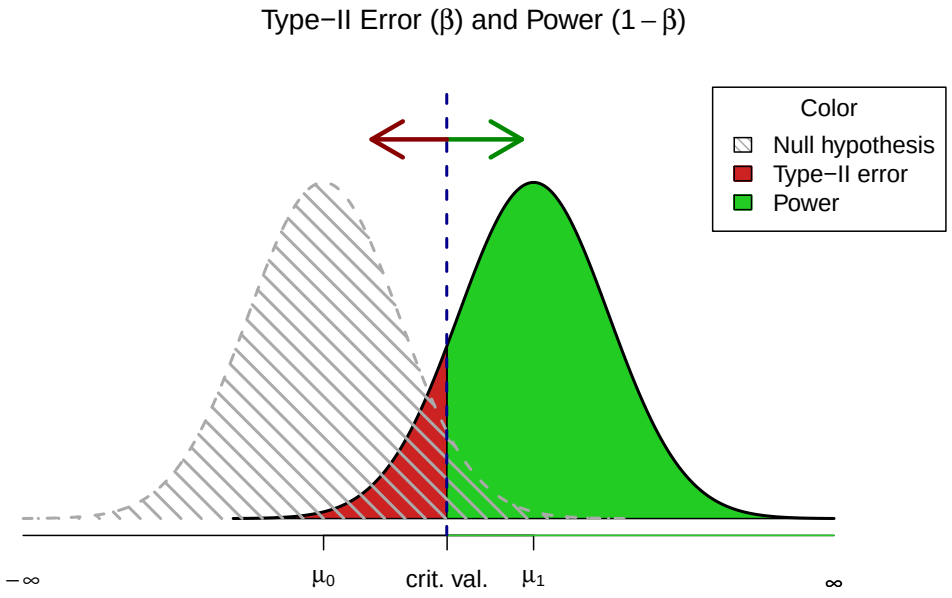
The meaning of statistical power **Power** *is the probability* $(1-\beta)$ *of detecting an effect, given that the effect is really there.* In other words, it is the probability of correctly rejecting the null hypothesis when it is in fact false. For example, let's say that we have a simple study with 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.)

```
#### One-sample power
# Power plot with two normal distributions
# http://stats.stackexchange.com/questions/14140/how-to-best-display-graphically-type-ii-beta-error-pow

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, borde
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

[1]Cohen, J. (1988). Statistical power analysis for the behavior sciences. (2nd ed.). Hillsdale, NJ: Erlbaum.

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:

- 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.

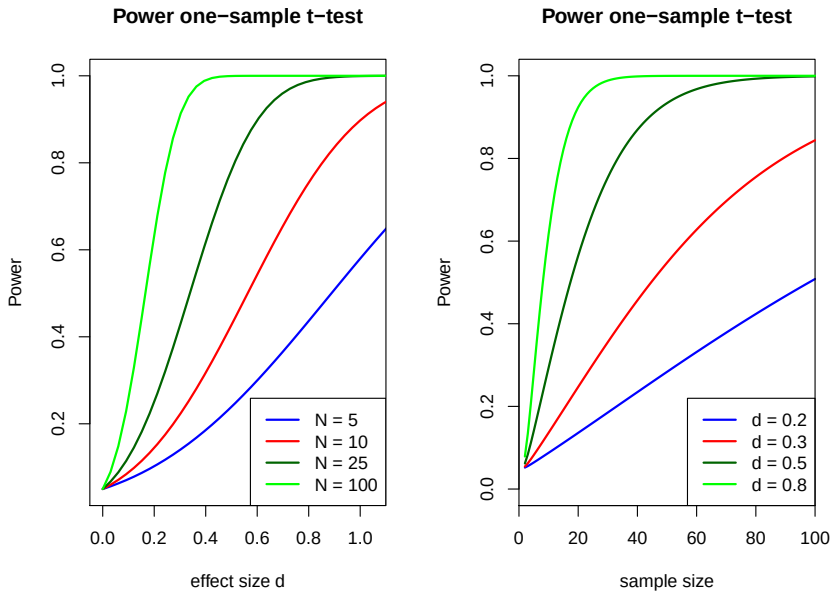
A common convention is to try to get at least enough data to have 80% power. So, how do you figure out how many subjects you need to have the desired amount of power. There are several methods, including:

- 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

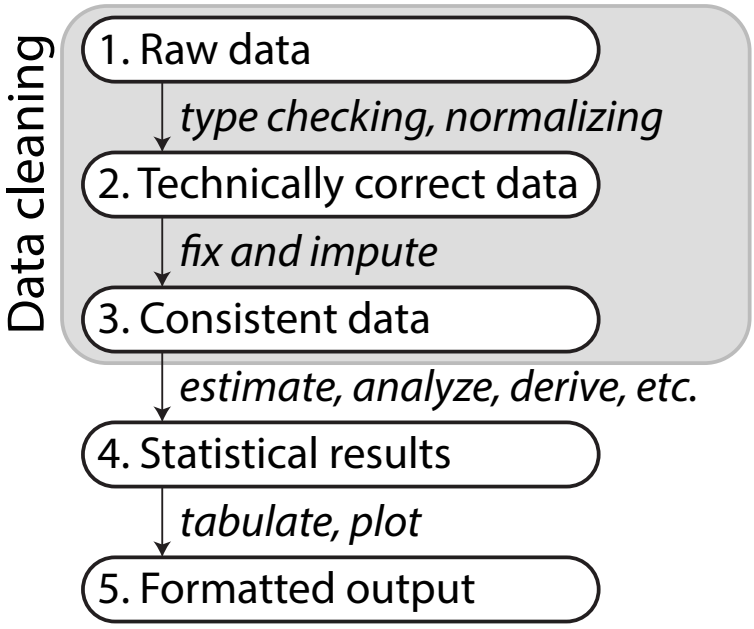
Data Cleaning

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.



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.

[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`

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.

```
#### Example: unnamed person text
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=FALSE`. 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.

```
#### Example: Dalton data
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
!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
```

[2] `http://en.wikipedia.org/wiki/Regular_expression`

```
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] "2018-10-10 00:15:10 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 character. 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
```

[3] `http://en.wikipedia.org/wiki/Regular_expression`

```
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.

[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`.

```
# 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
```

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 ±`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).

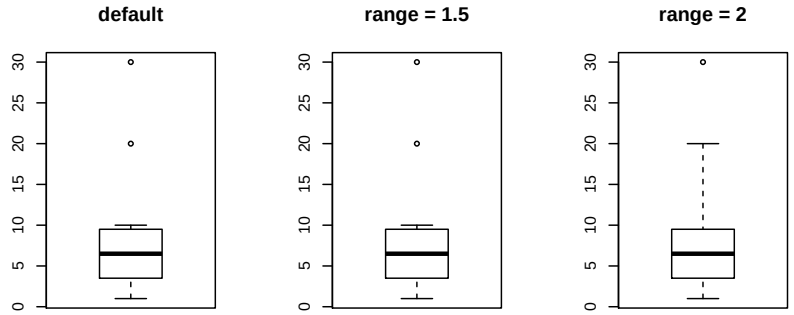
```
boxplot.stats(x, coef = 2)$out
```

```
## [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

$$h(x) \quad = \quad \max\left(\frac{x}{x^*}, \frac{x^*}{x}\right) \quad \geq \quad r, \quad \text{and} \quad x > 0.$$

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.

```
#### Example: Hiridoglou and Berthelot outlier detection function
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()`.

```
#### Example: people data for cleaning
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.

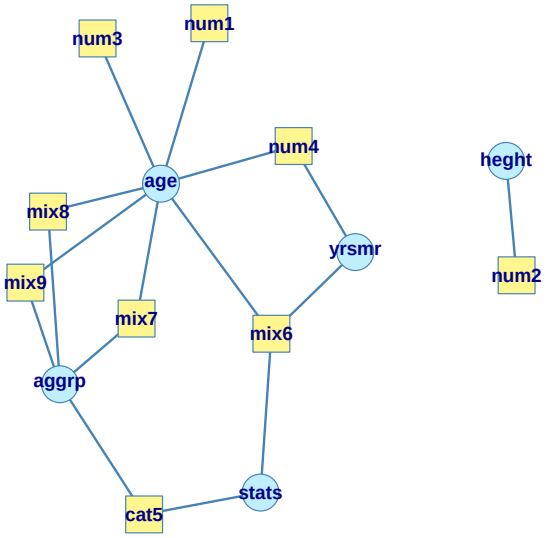
```
#### Edit rules for people data
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch18_edits.txt"
E <- editfile(fn.data)
E
```

```
##
## 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
```

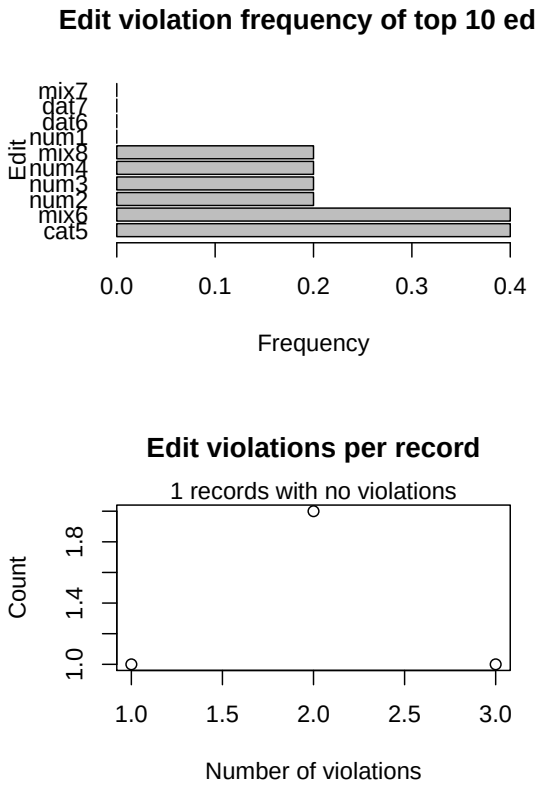


As the number of rules grows, looking at the full array produced by `violatedEdits()` becomes cumbersome. For this reason, `editrules` offers methods to summarize or visualize the result.

```
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.

```
#### Example: marx brothers data
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.

```
#### Conversions for marx bros data
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.

Part V

ADA2: Review of ADA1

Chapter 1

R statistical software and review

The purpose of this chapter is to discuss R in the context of a quick review of the topics we covered last semester in ADA1[1].

1.1 R

R is a programming language for programming, data management, and statistical analysis. So many people have written "An Introduction to R", that I refer you to the course website[2] for links to tutorials. I encourage you to learn R by (1) running the commands in the tutorials, (2) looking at the help for the commands (e.g., `?mean`), and (3) trying things on your own as you become curious. Make mistakes, figure out why some things don't work the way you expect, and keep trying. Persistence wins the day with programming (as does asking and searching for help).

R is more difficult to master (though, more rewarding) than some statistical packages (such as Minitab) for the following reasons: (1) R does not, in general, provide a point-and-click environment for statistical analysis. Rather, R uses syntax-based programs (i.e., code) to define, transform, and read data, and to define the procedures for analyzing data. (2) R does not really have a spreadsheet environment for data management. Rather, data are entered directly within an R program, read from a file, or imported from a spreadsheet. All manipulation, transformation, and selection of data is coded in the R program. Well done, this means that all the steps of the analysis are available to be repeatable and understood.

Take a minute to install the packages we'll need this semester by executing the following commands in R.

```
#### Install packages needed this semester
ADA2.package.list <- c("BSDA",      "Hmisc",    "MASS",         "NbClust",
```

[1] http://statacumen.com/teaching/ada1/
[2] http://statacumen.com/teaching/ada2/

```
                      "aod",        "candisc",  "car",           "cluster",
                      "ellipse",    "ggplot2",  "gridExtra",     "klaR",
                      "leaps",      "lsmeans",  "moments",       "multcomp",
                      "mvnormtest", "nortest",  "plyr",          "popbio",
                      "reshape",    "reshape2", "scatterplot3d", "vcd",
                      "vioplot",    "xtable")
install.packages(ADA2.package.list)
```

1.2 ADA1 Ch 0: R warm-up

This example illustrates several strategies for data summarization and analysis. The data for this example are from 15 turkeys raised on farms in either Virginia or Wisconsin.

You should use **help** to get more information on the functions demonstrated here. Many of the lines in the program have comments, which helps anyone reading the program to more easily follow the logic. I **strongly recommend** commenting any line of code that isn't absolutely obvious what is being done and why it is being done. I also recommend placing comments before blocks of code in order to describe what the code below is *meant* to do.

```
#### Example: Turkey, R warm-up
# Read the data file from the website and learn some functions

# filename
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch01_turkey.csv"
# read file and assign data to turkey variable
turkey <- read.csv(fn.data)
# examine the structure of the dataset, is it what you expected?
  # a data.frame containing integers, numbers, and factors
str(turkey)
## 'data.frame': 15 obs. of  3 variables:
##  $ age   : int  28 20 32 25 23 22 29 27 28 26 ...
##  $ weight: num  13.3 8.9 15.1 13.8 13.1 10.4 13.1 12.4 13.2 11.8 ...
##  $ orig  : Factor w/ 2 levels "va","wi": 1 1 1 2 2 1 1 1 1 1 ...
# print dataset to screen
turkey
##    age weight orig
## 1   28   13.3   va
## 2   20    8.9   va
## 3   32   15.1   va
## 4   25   13.8   wi
## 5   23   13.1   wi
## 6   22   10.4   va
## 7   29   13.1   va
## 8   27   12.4   va
```

```
## 9   28   13.2   va
## 10  26   11.8   va
## 11  21   11.5   wi
## 12  31   16.6   wi
## 13  27   14.2   wi
## 14  29   15.4   wi
## 15  30   15.9   wi
# Note: to view the age variable (column), there's a few ways to do that
turkey$age          # name the variable
##  [1] 28 20 32 25 23 22 29 27 28 26 21 31 27 29 30
turkey[, 1]         # give the column number
##  [1] 28 20 32 25 23 22 29 27 28 26 21 31 27 29 30
turkey[, "age"]     # give the column name
##  [1] 28 20 32 25 23 22 29 27 28 26 21 31 27 29 30
# and the structure is a vector
str(turkey$age)
##  int [1:15] 28 20 32 25 23 22 29 27 28 26 ...
# let's create an additional variable for later
# gt25mo will be a variable indicating whether the age is greater than 25 months
turkey$gt25mo <- (turkey$age > 25)
# now we also have a Boolean (logical) column
str(turkey)
## 'data.frame': 15 obs. of  4 variables:
##  $ age   : int  28 20 32 25 23 22 29 27 28 26 ...
##  $ weight: num  13.3 8.9 15.1 13.8 13.1 10.4 13.1 12.4 13.2 11.8 ...
##  $ orig  : Factor w/ 2 levels "va","wi": 1 1 1 2 2 1 1 1 1 1 ...
##  $ gt25mo: logi  TRUE FALSE TRUE FALSE FALSE FALSE ...
# there are a couple ways of subsetting the rows
turkey[(turkey$gt25mo == TRUE),]  # specify the rows
##    age weight orig gt25mo
## 1   28   13.3   va   TRUE
## 3   32   15.1   va   TRUE
## 7   29   13.1   va   TRUE
## 8   27   12.4   va   TRUE
## 9   28   13.2   va   TRUE
## 10  26   11.8   va   TRUE
## 12  31   16.6   wi   TRUE
## 13  27   14.2   wi   TRUE
## 14  29   15.4   wi   TRUE
## 15  30   15.9   wi   TRUE
subset(turkey, gt25mo == FALSE)   # use subset() to select the data.frame records
##    age weight orig gt25mo
## 2   20    8.9   va  FALSE
## 4   25   13.8   wi  FALSE
## 5   23   13.1   wi  FALSE
## 6   22   10.4   va  FALSE
## 11  21   11.5   wi  FALSE
```

Analyses can be then done on the entire dataset, or repeated for all subsets of a variable in the dataset.

```
# summaries of each variable in the entire dataset,
summary(turkey)
```

```
##       age            weight      orig     gt25mo
##  Min.   :20.00   Min.   : 8.90   va:8   Mode :logical
##  1st Qu.:24.00   1st Qu.:12.10   wi:7   FALSE:5
##  Median :27.00   Median :13.20          TRUE :10
##  Mean   :26.53   Mean   :13.25
##  3rd Qu.:29.00   3rd Qu.:14.65
##  Max.   :32.00   Max.   :16.60
```

```
# or summarize by a variable in the dataset.
by(turkey, turkey$orig, summary)
```

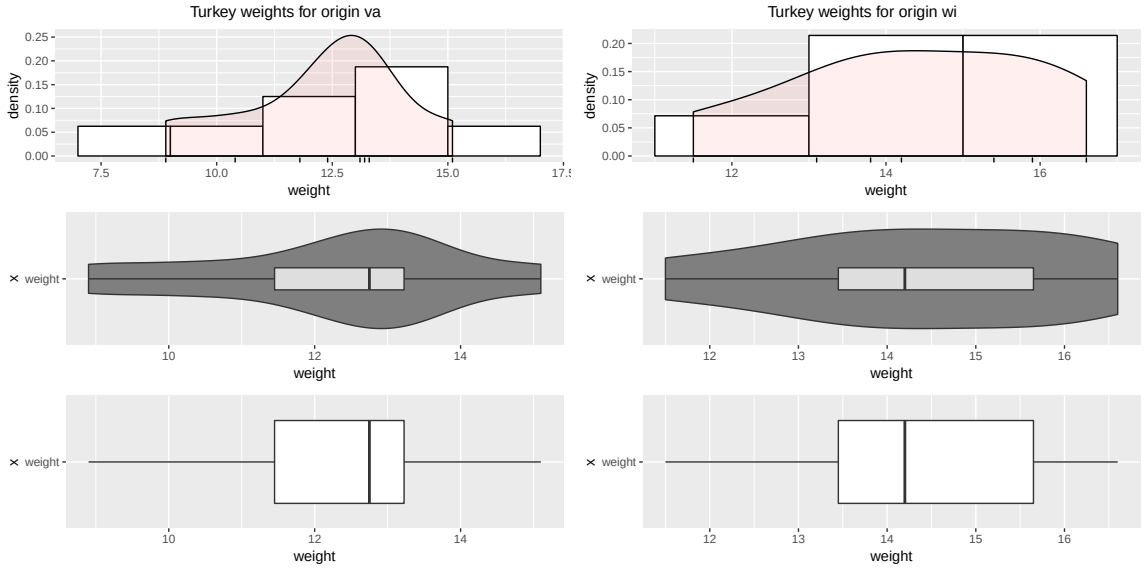
```
## turkey$orig: va
##       age            weight      orig     gt25mo
##  Min.   :20.00   Min.   : 8.90   va:8   Mode :logical
##  1st Qu.:25.00   1st Qu.:11.45   wi:0   FALSE:2
##  Median :27.50   Median :12.75          TRUE :6
##  Mean   :26.50   Mean   :12.28
##  3rd Qu.:28.25   3rd Qu.:13.22
##  Max.   :32.00   Max.   :15.10
## ------------------------------------------------------
## turkey$orig: wi
##       age            weight      orig     gt25mo
##  Min.   :21.00   Min.   :11.50   va:0   Mode :logical
##  1st Qu.:24.00   1st Qu.:13.45   wi:7   FALSE:3
##  Median :27.00   Median :14.20          TRUE :4
##  Mean   :26.57   Mean   :14.36
##  3rd Qu.:29.50   3rd Qu.:15.65
##  Max.   :31.00   Max.   :16.60
```

1.3 ADA1 Chapters 2, 4, 6: Estimation in one-sample problems

Plot the weights by origin.

```
#### Example: Turkey, Chapters 2, 4, 6
# subset the data for convenience
turkeyva <- subset(turkey, orig == "va")
turkeywi <- subset(turkey, orig == "wi")

library(ggplot2)
# Histogram overlaid with kernel density curve
p11 <- ggplot(turkeyva, aes(x = weight))
  # Histogram with density instead of count on y-axis
p11 <- p11 + geom_histogram(aes(y=..density..)
                        , binwidth=2
                        , colour="black", fill="white")
  # Overlay with transparent density plot
p11 <- p11 + geom_density(alpha=0.1, fill="#FF6666")
p11 <- p11 + geom_rug()

# violin plot
```

```
p12 <- ggplot(turkeyva, aes(x = "weight", y = weight))
p12 <- p12 + geom_violin(fill = "gray50")
p12 <- p12 + geom_boxplot(width = 0.2, alpha = 3/4)
p12 <- p12 + coord_flip()

# boxplot
p13 <- ggplot(turkeyva, aes(x = "weight", y = weight))
p13 <- p13 + geom_boxplot()
p13 <- p13 + coord_flip()

library(gridExtra)
#grid.arrange(p11, p12, p13, ncol=1, main="Turkey weights for origin va")
## add grobs = list(), and main= becomes top=
grid.arrange(grobs = list(p11, p12, p13), ncol=1, top="Turkey weights for origin va")

# Histogram overlaid with kernel density curve
p21 <- ggplot(turkeywi, aes(x = weight))
  # Histogram with density instead of count on y-axis
p21 <- p21 + geom_histogram(aes(y=..density..)
                            , binwidth=2
                            , colour="black", fill="white")
  # Overlay with transparent density plot
p21 <- p21 + geom_density(alpha=0.1, fill="#FF6666")
p21 <- p21 + geom_rug()

# violin plot
p22 <- ggplot(turkeywi, aes(x = "weight", y = weight))
p22 <- p22 + geom_violin(fill = "gray50")
p22 <- p22 + geom_boxplot(width = 0.2, alpha = 3/4)
p22 <- p22 + coord_flip()

# boxplot
p23 <- ggplot(turkeywi, aes(x = "weight", y = weight))
p23 <- p23 + geom_boxplot()
p23 <- p23 + coord_flip()

library(gridExtra)
grid.arrange(grobs = list(p21, p22, p23), ncol=1, top="Turkey weights for origin wi")
```

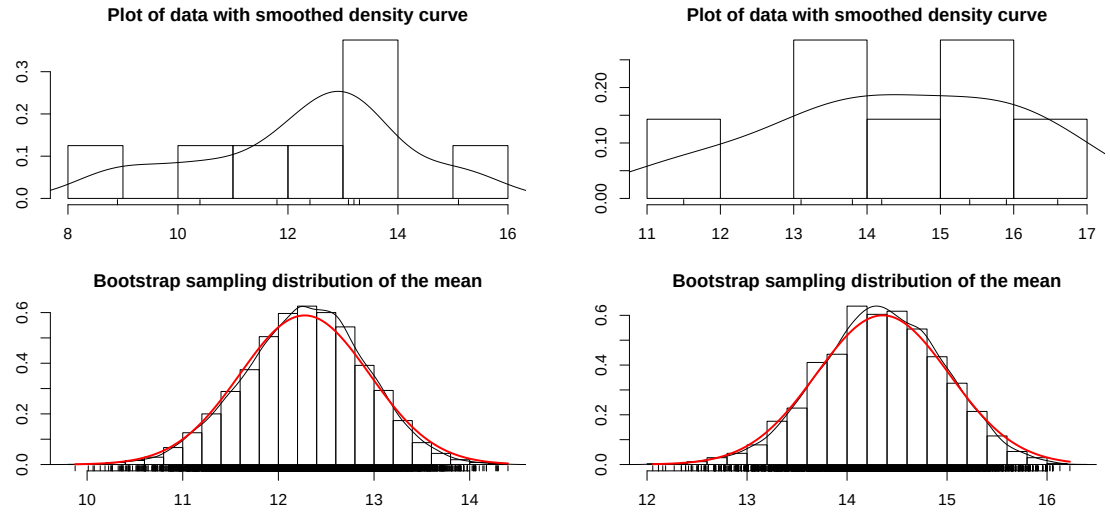


Check normality of each sample graphically with with bootstrap sampling distribution and normal quantile plot and formally with normality tests.

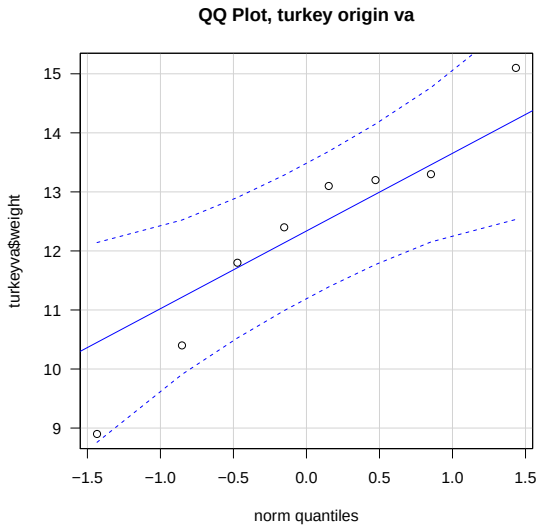
```
# a function to compare the bootstrap sampling distribution with
#   a normal distribution with mean and SEM estimated from the data
bs.one.samp.dist <- function(dat, N = 1e4) {
  n <- length(dat);
  # resample from data
  sam <- matrix(sample(dat, size = N * n, replace = TRUE), ncol=N);
  # draw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

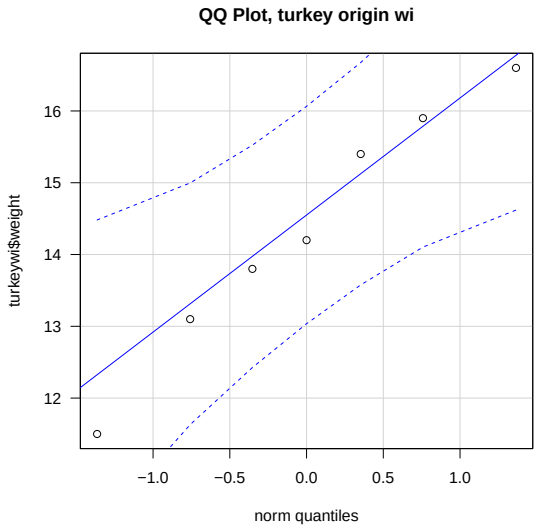
```
  old.par <- par(no.readonly = TRUE)
  # make smaller margins
  par(mfrow=c(2,1), mar=c(3,2,2,1), oma=c(1,1,1,1))
  # Histogram overlaid with kernel density curve
  hist(dat, freq = FALSE, breaks = 6
      , main = "Plot of data with smoothed density curve")
  points(density(dat), type = "l")
  rug(dat)

  hist(colMeans(sam), 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)
}
```

```
# Bootstrap sampling distribution
bs.one.samp.dist(turkeyva$weight)
bs.one.samp.dist(turkeywi$weight)
```




```
# normal quantile-quantile (QQ) plot of each orig sample
library(car)
# qq plot
#   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(turkeyva$weight, las = 1, id = list(n = 0, cex = 1), lwd = 1
     , main="QQ Plot, turkey origin va")
qqPlot(turkeywi$weight, las = 1, id = list(n = 0, cex = 1), lwd = 1
     , main="QQ Plot, turkey origin wi")
```






```
# Normality tests

# VA
shapiro.test(turkeyva$weight)
##
##  Shapiro-Wilk normality test
##
## data:  turkeyva$weight
## W = 0.95414, p-value = 0.7528
library(nortest)
ad.test(turkeyva$weight)
##
##  Anderson-Darling normality test
##
## data:  turkeyva$weight
## A = 0.283, p-value = 0.5339
# lillie.test(turkeyva$weight)
cvm.test(turkeyva$weight)
##
##  Cramer-von Mises normality test
##
## data:  turkeyva$weight
## W = 0.050135, p-value = 0.4642
# WI
shapiro.test(turkeywi$weight)
##
##  Shapiro-Wilk normality test
##
## data:  turkeywi$weight
```

```
## W = 0.97326, p-value = 0.9209
library(nortest)
ad.test(turkeywi$weight)
## Error in ad.test(turkeywi$weight):  sample size must be greater than 7
# lillie.test(turkeywi$weight)
cvm.test(turkeywi$weight)
## Error in cvm.test(turkeywi$weight):  sample size must be greater than 7
## Note: The errors above are expected.
```

Because we do not have any serious departures from normality (the data are consistent with being normal, as well as the sampling distribution of the mean) the t-test is appropriate. We will also look at a couple nonparametric methods.

```
# Is the average turkey weight 12 lbs?

# t-tests of the mean

# VA
t.summary <- t.test(turkeyva$weight, mu = 12)
t.summary
##
##  One Sample t-test
##
## data:  turkeyva$weight
## t = 0.40582, df = 7, p-value = 0.697
## alternative hypothesis: true mean is not equal to 12
## 95 percent confidence interval:
##  10.67264 13.87736
## sample estimates:
## mean of x
##    12.275
# WI
t.summary <- t.test(turkeywi$weight, mu = 12)
t.summary
##
##  One Sample t-test
##
## data:  turkeywi$weight
## t = 3.5442, df = 6, p-value = 0.01216
## alternative hypothesis: true mean is not equal to 12
## 95 percent confidence interval:
##  12.72978 15.98450
## sample estimates:
## mean of x
##  14.35714
# Sign test for the median

# VA
```

```r
library(BSDA)
SIGN.test(turkeyva$weight, md=12)
```

```
##
##  One-sample Sign-Test
##
## data:  turkeyva$weight
## s = 5, p-value = 0.7266
## alternative hypothesis: true median is not equal to 12
## 95 percent confidence interval:
##   9.9125 13.8850
## sample estimates:
## median of x
##       12.75
##
## Achieved and Interpolated Confidence Intervals:
##
##                   Conf.Level  L.E.pt U.E.pt
## Lower Achieved CI     0.9297 10.4000 13.300
## Interpolated CI       0.9500  9.9125 13.885
## Upper Achieved CI     0.9922  8.9000 15.100
```

```r
# WI
SIGN.test(turkeywi$weight, md=12)
```

```
##
##  One-sample Sign-Test
##
## data:  turkeywi$weight
## s = 6, p-value = 0.125
## alternative hypothesis: true median is not equal to 12
## 95 percent confidence interval:
##  12.00286 16.38000
## sample estimates:
## median of x
##        14.2
##
## Achieved and Interpolated Confidence Intervals:
##
##                   Conf.Level  L.E.pt U.E.pt
## Lower Achieved CI     0.8750 13.1000  15.90
## Interpolated CI       0.9500 12.0029  16.38
## Upper Achieved CI     0.9844 11.5000  16.60
```

```r
# Wilcoxon sign-rank test for the median (or mean, since symmetric assumption)

# VA
# with continuity correction in the normal approximation for the p-value
wilcox.test(turkeyva$weight, mu=12, conf.int=TRUE)
```

```
## Warning in wilcox.test.default(turkeyva$weight, mu = 12, conf.int = TRUE): cannot
compute exact p-value with ties
```

```
## Warning in wilcox.test.default(turkeyva$weight, mu = 12, conf.int = TRUE): cannot
compute exact confidence interval with ties
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  turkeyva$weight
## V = 21.5, p-value = 0.674
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  10.40005 14.09997
## sample estimates:
## (pseudo)median
##       12.47331
```

```
# without continuity correction
wilcox.test(turkeyva$weight, mu=12, conf.int=TRUE, correct=FALSE)
```

```
## Warning in wilcox.test.default(turkeyva$weight, mu = 12, conf.int = TRUE, :  cannot
compute exact p-value with ties
## Warning in wilcox.test.default(turkeyva$weight, mu = 12, conf.int = TRUE, :  cannot
compute exact confidence interval with ties
##
##  Wilcoxon signed rank test
##
## data:  turkeyva$weight
## V = 21.5, p-value = 0.6236
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  10.64999 13.75002
## sample estimates:
## (pseudo)median
##       12.47331
```

```
# WI
# with continuity correction in the normal approximation for the p-value
wilcox.test(turkeywi$weight, mu=12, conf.int=TRUE)
```

```
##
##  Wilcoxon signed rank test
##
## data:  turkeywi$weight
## V = 27, p-value = 0.03125
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  12.65 16.00
## sample estimates:
## (pseudo)median
##         14.375
```

```
# without continuity correction
wilcox.test(turkeywi$weight, mu=12, conf.int=TRUE, correct=FALSE)
```

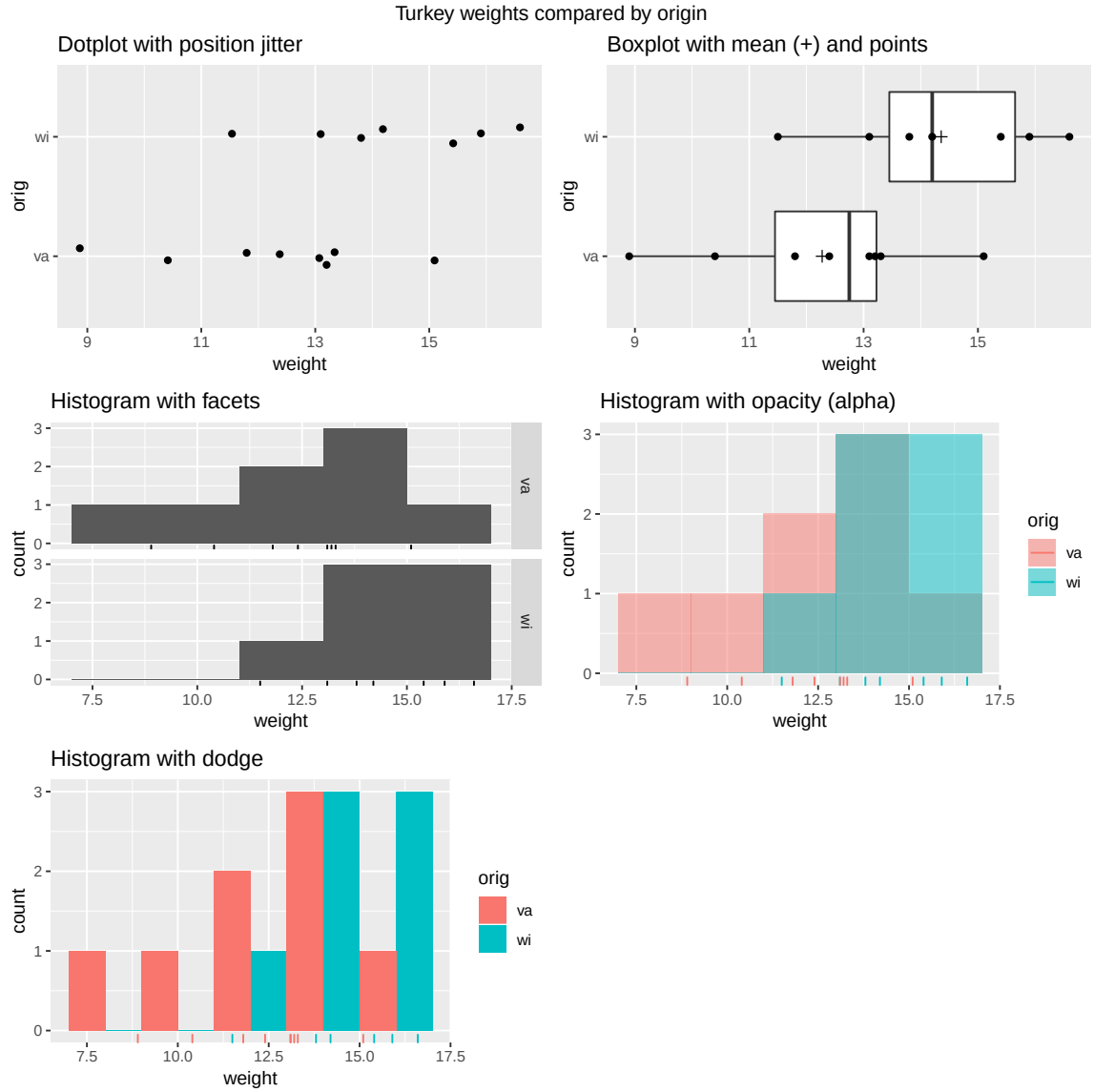
```
##
##  Wilcoxon signed rank test
##
## data:  turkeywi$weight
## V = 27, p-value = 0.03125
## alternative hypothesis: true location is not equal to 12
## 95 percent confidence interval:
##  12.65 16.00
## sample estimates:
## (pseudo)median
##         14.375
```

1.4 ADA1 Chapters 3, 4, 6: Two-sample inferences

Presume it is of interest to compare the center of the weight distributions between the origins. There are many ways to plot the data for visual comparisons.

```
#### Example: Turkey, Chapters 3, 4, 6
# stripchart (dotplot) using ggplot
library(ggplot2)
p1 <- ggplot(turkey, aes(x = weight, y = orig))
p1 <- p1 + geom_point(position = position_jitter(h=0.1))
p1 <- p1 + labs(title = "Dotplot with position jitter")

# boxplot
p2 <- ggplot(turkey, aes(x = orig, y = weight))
p2 <- p2 + geom_boxplot()
# add a "+" at the mean
p2 <- p2 + stat_summary(fun.y = mean, geom = "point", shape = 3, size = 2)
p2 <- p2 + geom_point()
p2 <- p2 + coord_flip()
p2 <- p2 + labs(title = "Boxplot with mean (+) and points")

# histogram using ggplot
p3 <- ggplot(turkey, aes(x = weight))
p3 <- p3 + geom_histogram(binwidth = 2)
p3 <- p3 + geom_rug()
p3 <- p3 + facet_grid(orig ~ .)
p3 <- p3 + labs(title = "Histogram with facets")

p4 <- ggplot(turkey, aes(x = weight, fill=orig))
p4 <- p4 + geom_histogram(binwidth = 2, alpha = 0.5, position="identity")
p4 <- p4 + geom_rug(aes(colour = orig))
p4 <- p4 + labs(title = "Histogram with opacity (alpha)")
```

```
p5 <- ggplot(turkey, aes(x = weight, fill=orig))
p5 <- p5 + geom_histogram(binwidth = 2, alpha = 1, position="dodge")
p5 <- p5 + geom_rug(aes(colour = orig))
p5 <- p5 + labs(title = "Histogram with dodge")

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4, p5), ncol=2, nrow=3
           , top="Turkey weights compared by origin")
```

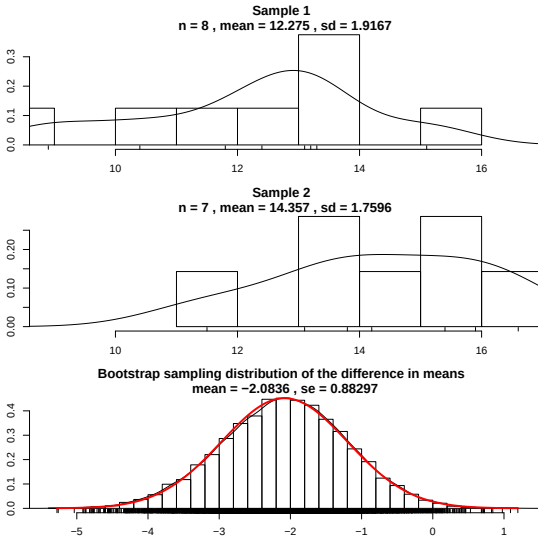


Using the two-sample t-test, first check the normality assumptions of the sampling distribution of the mean difference between the populations.

```
# 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)
}
```

```
bs.two.samp.diff.dist(turkeyva$weight, turkeywi$weight)
```



Two-sample t-test is appropriate since the bootstrap sampling distribution of the difference in means is approximately normal. This is the most powerful test and detects a difference at a 0.05 significance level.

```
# Two-sample t-test
## Equal variances
#   var.equal = FALSE is the default
# two-sample t-test specifying two separate vectors
t.summary.eqvar <- t.test(turkeyva$weight, turkeywi$weight, var.equal = TRUE)
t.summary.eqvar
##
##  Two Sample t-test
##
## data:  turkeyva$weight and turkeywi$weight
## t = -2.1796, df = 13, p-value = 0.04827
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.14595971 -0.01832601
## sample estimates:
## mean of x mean of y
##  12.27500  14.35714
# two-sample t-test with unequal variances (Welch = Satterthwaite)
#   specified using data.frame and a formula, HeadBreadth by Group
t.summary.uneqvar <- t.test(weight ~ orig, data = turkey, var.equal = FALSE)
t.summary.uneqvar
##
##  Welch Two Sample t-test
##
## data:  weight by orig
## t = -2.1929, df = 12.956, p-value = 0.04717
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.13407696 -0.03020875
## sample estimates:
## mean in group va mean in group wi
##         12.27500         14.35714
```

(Wilcoxon-)Mann-Whitney two-sample test is appropriate because the shapes of the two distributions are similar, though their locations are different. This is a less powerful test, but doesn't require normality, and fails to detect a difference at a 0.05 significance level.

```
# with continuity correction in the normal approximation for the p-value
wilcox.test(turkeyva$weight, turkeywi$weight, conf.int=TRUE)
## Warning in wilcox.test.default(turkeyva$weight, turkeywi$weight, conf.int = TRUE):
cannot compute exact p-value with ties
## Warning in wilcox.test.default(turkeyva$weight, turkeywi$weight, conf.int = TRUE):
cannot compute exact confidence intervals with ties
##
##  Wilcoxon rank sum test with continuity correction
```

```
##
## data:  turkeyva$weight and turkeywi$weight
## W = 11.5, p-value = 0.06384
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -4.19994493  0.09993686
## sample estimates:
## difference in location
##              -2.152771
# without continuity correction
wilcox.test(turkeyva$weight, turkeywi$weight, conf.int=TRUE, correct=FALSE)
## Warning in wilcox.test.default(turkeyva$weight, turkeywi$weight, conf.int = TRUE,
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(turkeyva$weight, turkeywi$weight, conf.int = TRUE,
:  cannot compute exact confidence intervals with ties
##
##  Wilcoxon rank sum test
##
## data:  turkeyva$weight and turkeywi$weight
## W = 11.5, p-value = 0.05598
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -4.100049e+00  1.445586e-05
## sample estimates:
## difference in location
##              -2.152771
```

1.5 ADA1 Chapters 5, 4, 6: One-way ANOVA

The Waste Run-up data[3] refer to five suppliers of the Levi-Strauss clothing manufacturing plant in Albuquerque. The firm's quality control department collects weekly data on percent-age waste (run-up) relative to what can be achieved by computer layouts of patterns on cloth. It is possible to have negative values, which indicate that the plant employees beat the computer in controlling waste. Under question are differences among the five supplier plants (PT1, . . . , PT5).

```
#### Example: Waste Run-up, Chapters 5, 4, 6
# convert to a data.frame by reading the text table
waste <- read.table(text = "
 PT1   PT2   PT3   PT4   PT5
 1.2  16.4  12.1  11.5  24.0
```

[3]From `http://lib.stat.cmu.edu/DASL/Stories/wasterunup.html`, the Data and Story Library (DASL, pronounced "dazzle") is an online library of datafiles and stories that illustrate the use of basic statistics methods. "Waste Run-up" dataset from L. Koopmans, Introduction to Contemporary Statistical Methods, Duxbury Press, 1987, p. 86.

```
10.1  -6.0   9.7  10.2  -3.7
-2.0 -11.6   7.4   3.8   8.2
 1.5  -1.3  -2.1   8.3   9.2
-3.0   4.0  10.1   6.6  -9.3
-0.7  17.0   4.7  10.2   8.0
 3.2   3.8   4.6   8.8  15.8
 2.7   4.3   3.9   2.7  22.3
-3.2  10.4   3.6   5.1   3.1
-1.7   4.2   9.6  11.2  16.8
 2.4   8.5   9.8   5.9  11.3
 0.3   6.3   6.5  13.0  12.3
 3.5   9.0   5.7   6.8  16.9
-0.8   7.1   5.1  14.5    NA
19.4   4.3   3.4   5.2    NA
 2.8  19.7  -0.8   7.3    NA
13.0   3.0  -3.9   7.1    NA
42.7   7.6   0.9   3.4    NA
 1.4  70.2   1.5   0.7    NA
 3.0   8.5    NA    NA    NA
 2.4   6.0    NA    NA    NA
 1.3   2.9    NA    NA    NA
", header=TRUE)
waste
```

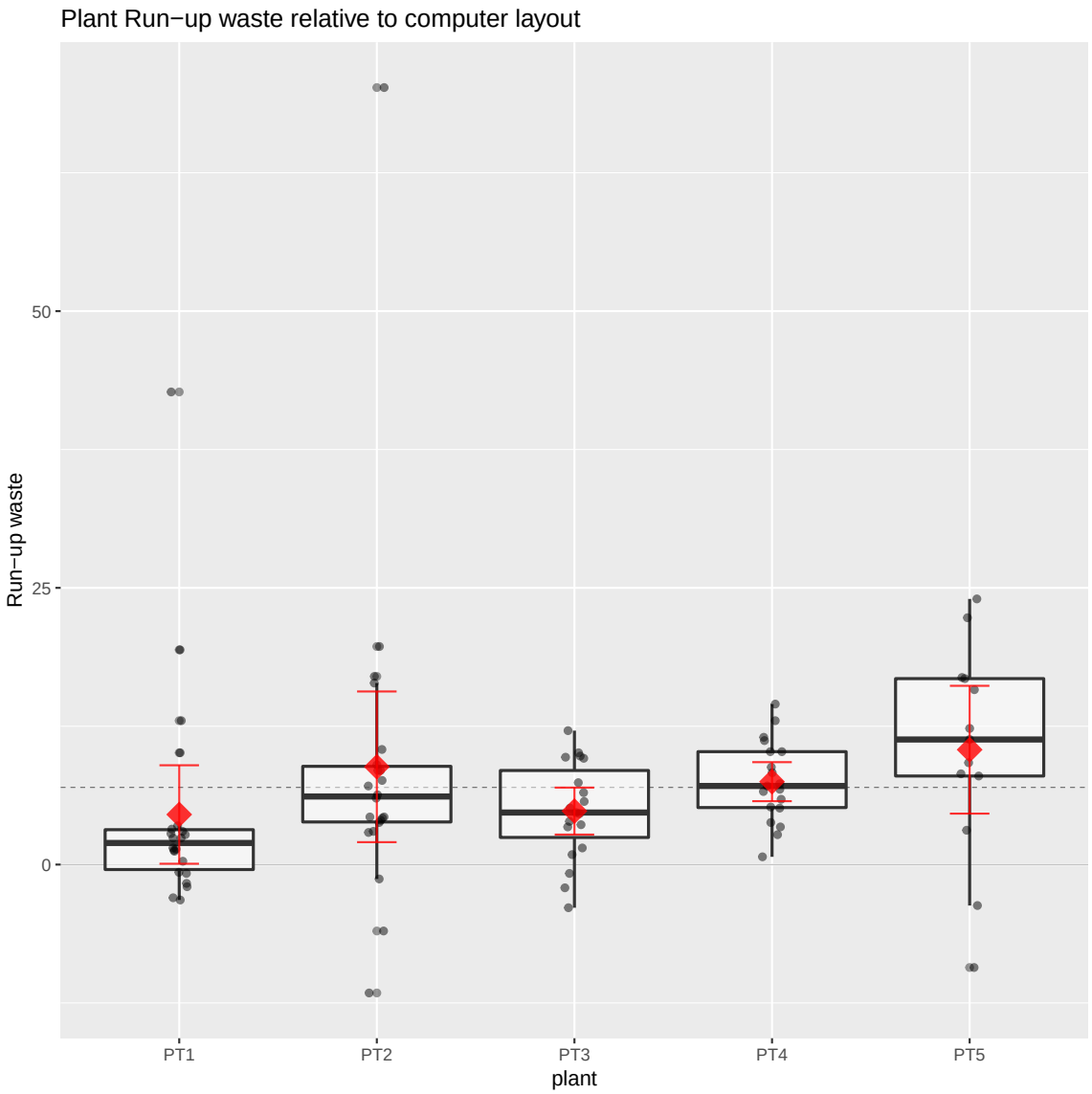
```
library(reshape2)
waste.long <- melt(waste,
```

```
              # id.vars: ID variables
              #   all variables to keep but not split apart on
              # id.vars=NULL,
              # measure.vars: The source columns
              #   (if unspecified then all other variables are measure.vars)
              # measure.vars = c("PT1","PT2","PT3","PT4","PT5"),
              # variable.name: Name of the destination column identifying each
              #   original column that the measurement came from
              variable.name = "plant",
              # value.name: column name for values in table
              value.name = "runup",
              # remove the NA values
              na.rm = TRUE
            )
## No id variables; using all as measure variables
str(waste.long)
## 'data.frame': 95 obs. of  2 variables:
##  $ plant: Factor w/ 5 levels "PT1","PT2","PT3",..: 1 1 1 1 1 1 1 1 1 ...
##  $ runup: num  1.2 10.1 -2 1.5 -3 -0.7 3.2 2.7 -3.2 -1.7 ...
head(waste.long)
##   plant runup
## 1   PT1   1.2
## 2   PT1  10.1
## 3   PT1  -2.0
## 4   PT1   1.5
## 5   PT1  -3.0
## 6   PT1  -0.7
tail(waste.long)
##     plant runup
## 96    PT5  22.3
## 97    PT5   3.1
## 98    PT5  16.8
## 99    PT5  11.3
## 100   PT5  12.3
## 101   PT5  16.9
# Calculate the mean, sd, n, and se for the plants

# 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
waste.summary <- ddply(waste.long,
                       "plant",
                       function(X) {
                         data.frame( m = mean(X$runup),
                                     s = sd(X$runup),
                                     n = length(X$runup)
```

```
                                  )
                    }
                  )
# standard errors
waste.summary$se  <- waste.summary$s/sqrt(waste.summary$n)
waste.summary$moe <- qt(1 - 0.05 / 2, df = waste.summary$n - 1) * waste.summary$se
# individual confidence limits
waste.summary$ci.l <- waste.summary$m - waste.summary$moe

waste.summary$ci.u <- waste.summary$m + waste.summary$moe
waste.summary
```

```
##   plant         m         s  n        se      moe       ci.l
## 1   PT1  4.522727 10.032041 22 2.1388383 4.447958 0.07476963
## 2   PT2  8.831818 15.353467 22 3.2733701 6.807346 2.02447241
## 3   PT3  4.831579  4.403162 19 1.0101547 2.122256 2.70932276
## 4   PT4  7.489474  3.657093 19 0.8389946 1.762662 5.72681139
## 5   PT5 10.376923  9.555030 13 2.6500884 5.774047 4.60287651
##        ci.u
## 1  8.970685
## 2 15.639164
## 3  6.953835
## 4  9.252136
## 5 16.150970
```

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(waste.long, aes(x = plant, y = runup))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(aes(yintercept = 0),
                    colour = "black", linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_hline(aes(yintercept = mean(runup)),
                    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 = "Plant Run-up waste relative to computer layout")
p <- p + ylab("Run-up waste")
print(p)
```



The outliers here suggest the ANOVA is not an appropriate model. The normality tests below suggest the distributions for the first two plants are not normal.

```
by(waste.long$runup, waste.long$plant, ad.test)
## waste.long$plant: PT1
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 2.8685, p-value = 1.761e-07
##
## ------------------------------------------------------
## waste.long$plant: PT2
```

```
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 2.5207, p-value = 1.334e-06
##
## ------------------------------------------------------
## waste.long$plant: PT3
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 0.23385, p-value = 0.7624
##
## ------------------------------------------------------
## waste.long$plant: PT4
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 0.12363, p-value = 0.9834
##
## ------------------------------------------------------
## waste.long$plant: PT5
##
##  Anderson-Darling normality test
##
## data:  dd[x, ]
## A = 0.27445, p-value = 0.6004
```

For review purposes, I'll fit the ANOVA, but we would count on the following nonparametric method for inference.

```
fit.w <- aov(runup ~ plant, data = waste.long)
summary(fit.w)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## plant        4    451  112.73    1.16  0.334
## Residuals   90   8749   97.21
```

```
fit.w
```

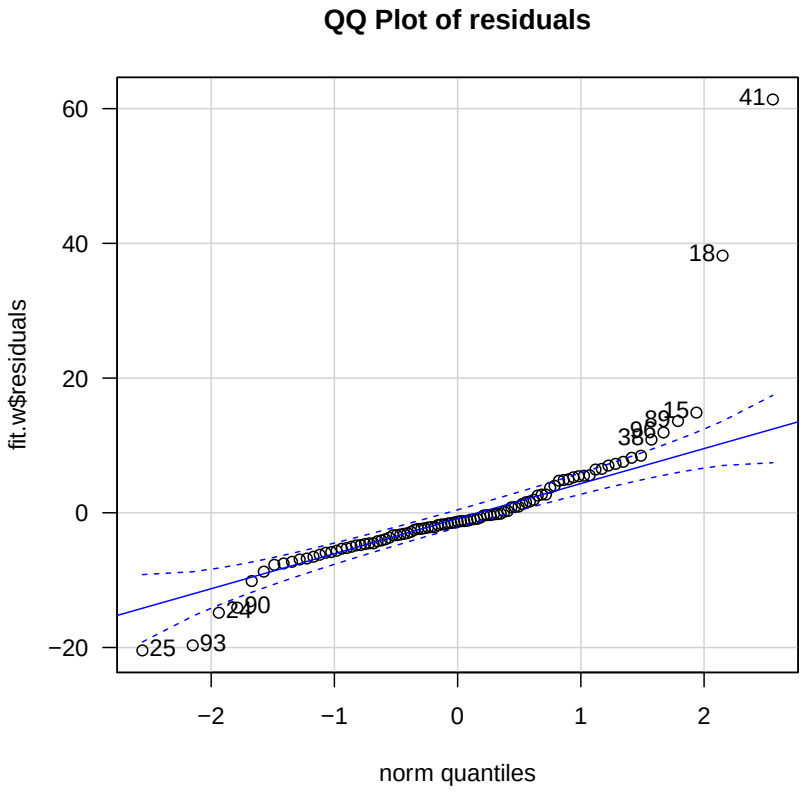
```
## Call:
##    aov(formula = runup ~ plant, data = waste.long)
##
## Terms:
##                     plant Residuals
## Sum of Squares   450.921  8749.088
## Deg. of Freedom        4        90
##
## Residual standard error: 9.859619
## Estimated effects may be unbalanced
```

```
# all pairwise comparisons among plants
# Fisher's LSD (FSD) uses "none"
pairwise.t.test(waste.long$runup, waste.long$plant,
                pool.sd = TRUE, p.adjust.method = "none")
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  waste.long$runup and waste.long$plant
##
##     PT1   PT2   PT3   PT4
## PT2 0.151 -     -     -
## PT3 0.921 0.198 -     -
## PT4 0.339 0.665 0.408 -
## PT5 0.093 0.655 0.122 0.418
##
## P value adjustment method: none
# Tukey 95% Individual p-values
TukeyHSD(fit.w)
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = runup ~ plant, data = waste.long)
##
## $plant
##                diff        lwr       upr     p adj
## PT2-PT1  4.3090909  -3.966713 12.584895 0.5976181
## PT3-PT1  0.3088517  -8.287424  8.905127 0.9999769
## PT4-PT1  2.9667464  -5.629529 11.563022 0.8718682
## PT5-PT1  5.8541958  -3.747712 15.456104 0.4408168
## PT3-PT2 -4.0002392 -12.596515  4.596036 0.6946720
## PT4-PT2 -1.3423445  -9.938620  7.253931 0.9924515
## PT5-PT2  1.5451049  -8.056803 11.147013 0.9915352
## PT4-PT3  2.6578947  -6.247327 11.563116 0.9203538
## PT5-PT3  5.5453441  -4.334112 15.424800 0.5251000
## PT5-PT4  2.8874494  -6.992007 12.766906 0.9258057
# Bonferroni 95% Individual p-values
# All Pairwise Comparisons among Levels of waste
pairwise.t.test(waste.long$runup, waste.long$plant,
                pool.sd = TRUE, p.adjust.method = "bonf")
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  waste.long$runup and waste.long$plant
##
##     PT1  PT2  PT3  PT4
## PT2 1.00 -    -    -
## PT3 1.00 1.00 -    -
```

```
## PT4 1.00 1.00 1.00 -
## PT5 0.93 1.00 1.00 1.00
##
## P value adjustment method: bonferroni
```

The residuals show many outliers

```
# QQ plot
par(mfrow=c(1,1))
library(car)
qqPlot(fit.w$residuals, las = 1, id = list(n = 10, cex = 1), lwd = 1
       , main="QQ Plot of residuals")
## 41 18 25 93 15 24 90 89 96 38
## 41 18 25 87 15 24 84 83 90 38
```



Kruskal-Wallis ANOVA is a non-parametric method for testing the hypothesis of equal population medians against the alternative that not all population medians are equal. It's still not perfect here because the distributional shapes are not all the same, but it is a better alternative than the ANOVA.

```
# KW ANOVA
fit.wk <- kruskal.test(runup ~ plant, data = waste.long)
fit.wk
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  runup by plant
## Kruskal-Wallis chi-squared = 15.319, df = 4, p-value =
## 0.004084
# Bonferroni 95% pairwise comparisions with continuity correction
#   in the normal approximation for the p-value
for (i1.pt in 1:4) {
  for (i2.pt in (i1.pt+1):5) {
    wt <- wilcox.test(waste[,names(waste)[i1.pt]], waste[,names(waste)[i2.pt]]
                      , conf.int=TRUE, conf.level = 1 - 0.05/choose(5,2))
    cat(names(waste)[i1.pt], names(waste)[i2.pt])
    print(wt)
  }
}
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT1 PT2
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 131.5, p-value = 0.009813
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -8.299958  1.599947
## sample estimates:
## difference in location
##              -4.399951
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT1 PT3
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 141.5, p-value = 0.07978
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -6.900028  2.700029
## sample estimates:
## difference in location
##              -2.500047
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
```

```
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT1 PT4
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 85, p-value = 0.001241
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -8.900056 -1.099910
## sample estimates:
## difference in location
##              -5.300051
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT1 PT5
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 76, p-value = 0.02318
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -15.50007   3.60001
## sample estimates:
## difference in location
##              -8.703538
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT2 PT3
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 238, p-value = 0.4562
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -3.50007  7.20000
## sample estimates:
## difference in location
##               1.352784
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT2 PT4
```

```
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 186, p-value = 0.5563
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -5.900014  3.800023
## sample estimates:
## difference in location
##              -1.099914
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT2 PT5
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 99, p-value = 0.1375
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -13.100001   7.400037
## sample estimates:
## difference in location
##              -4.630905
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT3 PT4
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 117, p-value = 0.06583
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -6.800001  1.699976
## sample estimates:
## difference in location
##              -2.400038
##
## PT3 PT5
##  Wilcoxon rank sum test
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 67, p-value = 0.03018
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
```

```
##  -13.4   1.7
## sample estimates:
## difference in location
##                   -6.6
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact p-value with ties
## Warning in wilcox.test.default(waste[, names(waste)[i1.pt]], waste[, names(waste)[i2.pt]],
:  cannot compute exact confidence intervals with ties
## PT4 PT5
##  Wilcoxon rank sum test with continuity correction
##
## data:  waste[, names(waste)[i1.pt]] and waste[, names(waste)[i2.pt]]
## W = 82, p-value = 0.1157
## alternative hypothesis: true location shift is not equal to 0
## 99.5 percent confidence interval:
##  -11.099965   4.799945
## sample estimates:
## difference in location
##              -4.000035
```

1.6 ADA1 Chapter 7: Categorical data analysis

Returning to the turkey dataset, below is the cross-classification of `orig` by `gt25mo`.

```
#### Example: Turkey, Chapter 7
# create a frequency table from two columns of categorical data
xt <- xtabs( ~ orig + gt25mo, data = turkey)
# display the table
xt
```

```
##     gt25mo
## orig FALSE TRUE
##   va     2    6
##   wi     3    4
```

```
# summary from xtabs() is the same as chisq.test() without continuity correction
summary(xt)
```

```
## Call: xtabs(formula = ~orig + gt25mo, data = turkey)
## Number of cases in table: 15
## Number of factors: 2
## Test for independence of all factors:
##  Chisq = 0.5357, df = 1, p-value = 0.4642
##  Chi-squared approximation may be incorrect
```

```
# same as xtabs()
x.summary <- chisq.test(xt, correct=FALSE)
```

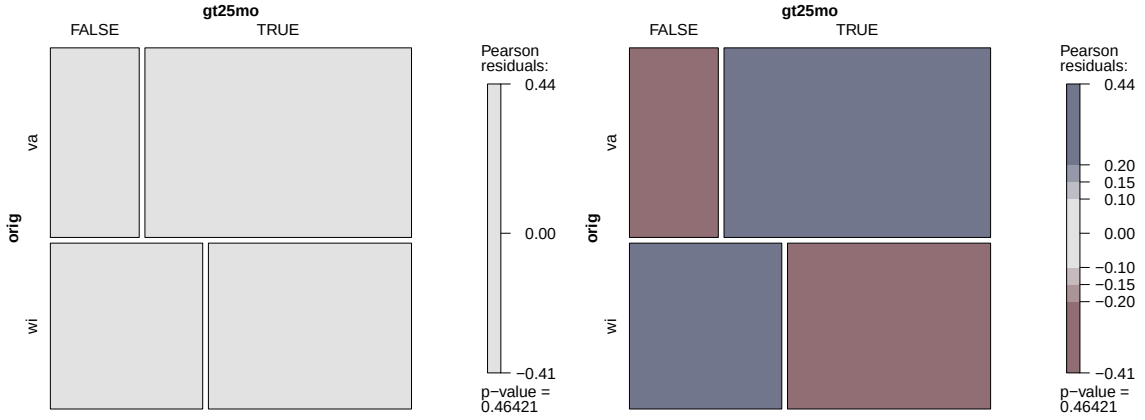
```
## Warning in chisq.test(xt, correct = FALSE): Chi-squared approximation may be incorrect
```

```
x.summary
```

```
##
##  Pearson's Chi-squared test
##
## data:  xt
## X-squared = 0.53571, df = 1, p-value = 0.4642
# the default is to perform Yates' continuity correction
chisq.test(xt)
## Warning in chisq.test(xt):  Chi-squared approximation may be incorrect
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  xt
## X-squared = 0.033482, df = 1, p-value = 0.8548
# Fisher's exact test
fisher.test(xt)
##
##  Fisher's Exact Test for Count Data
##
## data:  xt
## p-value = 0.6084
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.02687938 6.23767632
## sample estimates:
## odds ratio
##  0.4698172
```

A mosaic plot is for categorical data. Area represents frequency. The default shading is a good start, since colors only appear when there's evidence of association related to those cell values. In our example, there's insufficient evidence for association, so the default shading is all gray.

```
library(vcd)    # for mosaic()
# shading based on significance relative to appropriate chi-square distribution
mosaic(xt, shade=TRUE)
# you can define your own interpolated shading
mosaic(xt, shade=TRUE, gp_args = list(interpolate = seq(.1,.2,.05)))
```



From the `chisq.test()` above, we make a table to summarize important values from that analysis and compare the observed and expected frequencies in plots.

```
# use output in x.summary and create table
x.table <- data.frame(obs = x.summary$observed
                      , exp = x.summary$expected
                      , res = x.summary$residuals
                      , chisq = x.summary$residuals^2
                      , stdres = x.summary$stdres)
## Warning in data.frame(obs = x.summary$observed, exp = x.summary$expected, :  row
names were found from a short variable and have been discarded
# There are duplicate row and col identifiers in this x.table
#   because we're creating vectors from a two-way table
#   and columns identifying row and col names are automatically added.
# Can you figure out the naming scheme?
x.table
##   obs.orig obs.gt25mo obs.Freq exp.FALSE exp.TRUE res.orig res.gt25mo
## 1       va      FALSE        2  2.666667 5.333333       va      FALSE
## 2       wi      FALSE        3  2.333333 4.666667       wi      FALSE
## 3       va       TRUE        6  2.666667 5.333333       va       TRUE
## 4       wi       TRUE        4  2.333333 4.666667       wi       TRUE
##     res.Freq chisq.orig chisq.gt25mo chisq.Freq stdres.orig
## 1 -0.4082483         va        FALSE 0.16666667          va
## 2  0.4364358         wi        FALSE 0.19047619          wi
## 3  0.2886751         va         TRUE 0.08333333          va
## 4 -0.3086067         wi         TRUE 0.09523810          wi
##   stdres.gt25mo stdres.Freq
## 1         FALSE  -0.7319251
## 2         FALSE   0.7319251
## 3          TRUE   0.7319251
## 4          TRUE  -0.7319251
```

```r
# create a single column with a joint cell name
x.table$cellname <- paste(as.character(x.table$obs.orig)
                          , as.character(x.table$obs.gt25mo)
                          , sep="_")
# expected frequencies in a single column
x.table$exp <- c(x.table$exp.FALSE[1:2], x.table$exp.TRUE[3:4])
# create a simpler name for the obs, res, chisq, stdres
x.table$obs    <- x.table$obs.Freq
x.table$res    <- x.table$res.Freq
x.table$chisq  <- x.table$chisq.Freq
x.table$stdres <- x.table$stdres.Freq

# include only the "cleaned" columns
x.table <- subset(x.table, select = c(cellname, obs, exp, res, chisq, stdres))
x.table
```

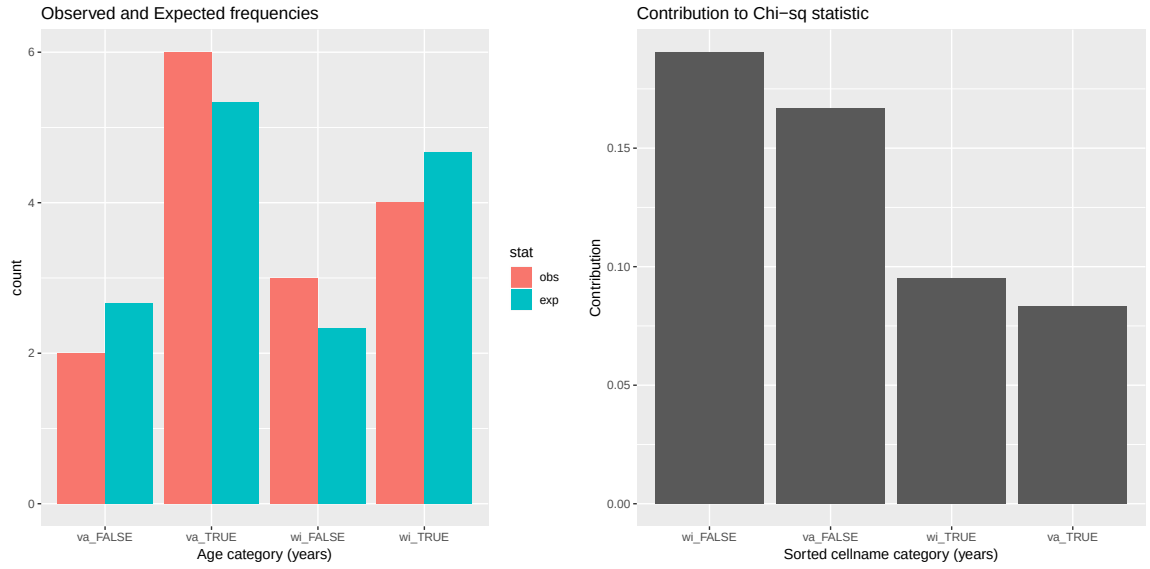
```
##   cellname obs      exp        res      chisq     stdres
## 1 va_FALSE   2 2.666667 -0.4082483 0.16666667 -0.7319251
## 2 wi_FALSE   3 2.333333  0.4364358 0.19047619  0.7319251
## 3  va_TRUE   6 5.333333  0.2886751 0.08333333  0.7319251
## 4  wi_TRUE   4 4.666667 -0.3086067 0.09523810 -0.7319251
```

```r
# reshape the data for plotting
library(reshape2)
x.table.obsexp <- melt(x.table,
              # id.vars: ID variables
              #   all variables to keep but not split apart on
              id.vars=c("cellname"),
              # 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"
            )
x.table.obsexp
```

```
##   cellname stat    value
## 1 va_FALSE  obs 2.000000
## 2 wi_FALSE  obs 3.000000
## 3  va_TRUE  obs 6.000000
## 4  wi_TRUE  obs 4.000000
## 5 va_FALSE  exp 2.666667
## 6 wi_FALSE  exp 2.333333
## 7  va_TRUE  exp 5.333333
## 8  wi_TRUE  exp 4.666667
```

Plot observed vs expected frequencies, and the contribution to chi-square statistic sorted decending.

```
# Observed vs Expected counts
library(ggplot2)
p <- ggplot(x.table.obsexp, aes(x = cellname, 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 cellname and chisq columns
x.table.chisq <- x.table[, c("cellname","chisq")]
# reorder the cellname categories to be descending relative to the chisq statistic
x.table.chisq$cellname <- with(x.table, reorder(cellname, -chisq))

p <- ggplot(x.table.chisq, aes(x = cellname, weight = chisq))
p <- p + geom_bar()
p <- p + labs(title = "Contribution to Chi-sq statistic")
p <- p + xlab("Sorted cellname category (years)")
p <- p + ylab("Contribution")
print(p)
```



1.7 ADA1 Chapter 8: Correlation and regression

Rocket Propellant Data A rocket motor is manufactured by bonding an igniter propellant and a sustainer propellant together inside a metal housing. The shear strength of the bond between the two types of propellant is an important quality characteristic. It is suspected that shear strength is related to the age in weeks of the

batch of sustainer propellant. Twenty observations on these two characteristics are given below. The first column is shear strength in psi, the second is age of propellant in weeks.

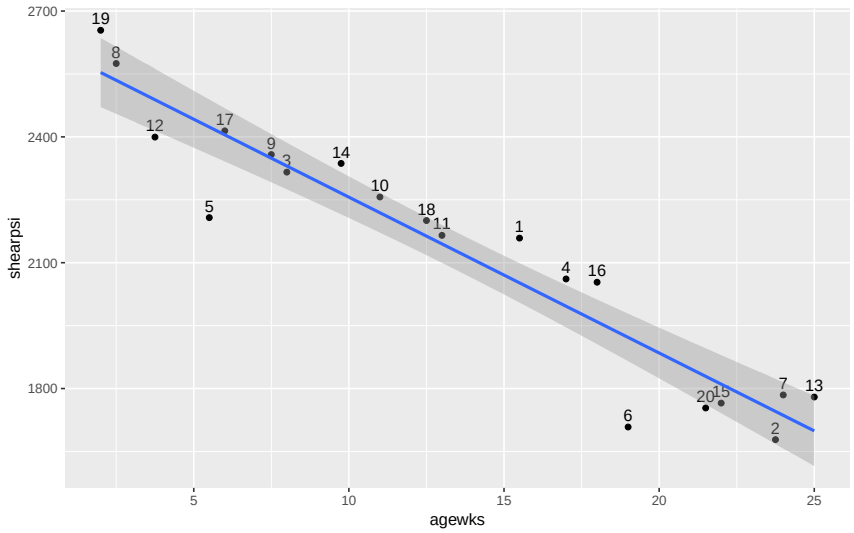
```
#### Example: Rocket, Chapter 8
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch01_rocket.dat"
# this file uses spaces as delimiters, so use read.table()
rocket <- read.table(fn.data, header = TRUE)
rocket$id <- 1:nrow(rocket) # add an id variable to identify observations
str(rocket)
```

```
## 'data.frame': 20 obs. of  3 variables:
##  $ shearpsi: num  2159 1678 2316 2061 2208 ...
##  $ agewks  : num  15.5 23.8 8 17 5.5 ...
##  $ id      : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
head(rocket)
```

```
##   shearpsi agewks id
## 1  2158.70  15.50  1
## 2  1678.15  23.75  2
## 3  2316.00   8.00  3
## 4  2061.30  17.00  4
## 5  2207.50   5.50  5
## 6  1708.30  19.00  6
```

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(rocket, aes(x = agewks, y = shearpsi, 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)
```



The data are reasonably linear, so fit the regression.

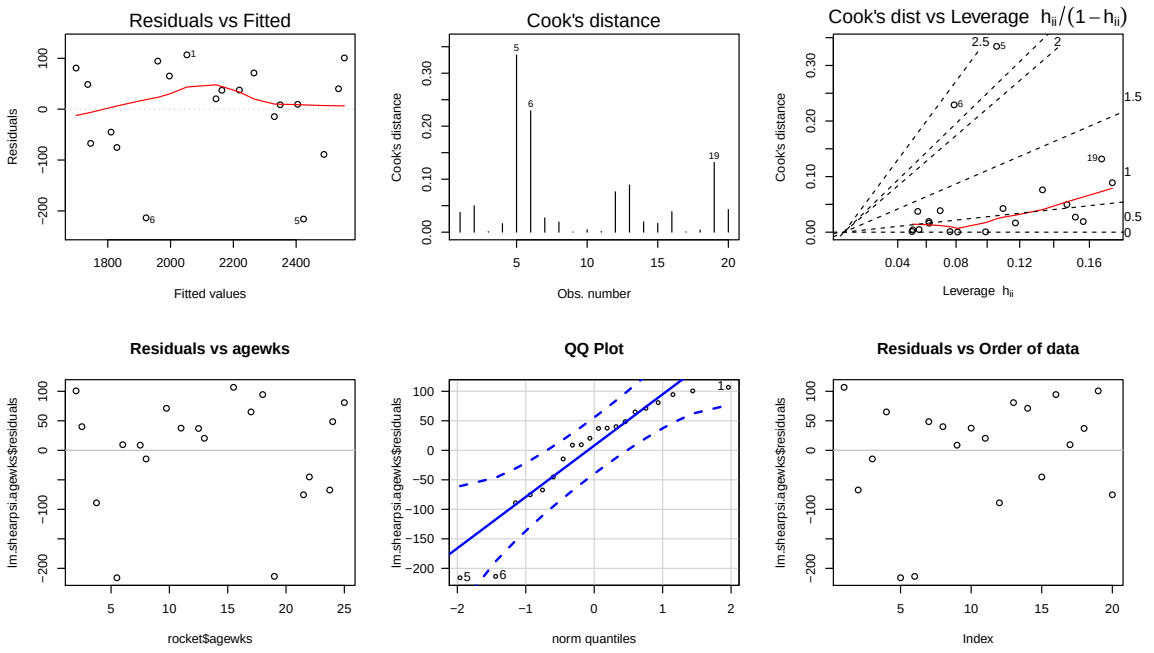
```
# fit the simple linear regression model
lm.shearpsi.agewks <- lm(shearpsi ~ agewks, data = rocket)
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm.shearpsi.agewks)
```

```
##
## Call:
## lm(formula = shearpsi ~ agewks, data = rocket)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -215.98  -50.68   28.74   66.61  106.76
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2627.822     44.184   59.48  < 2e-16 ***
## agewks       -37.154      2.889  -12.86 1.64e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 96.11 on 18 degrees of freedom
## Multiple R-squared:  0.9018,Adjusted R-squared:  0.8964
## F-statistic: 165.4 on 1 and 18 DF,  p-value: 1.643e-10
```

Plot diagnostics.

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.shearpsi.agewks, which = c(1,4,6))

# residuals vs weight
plot(rocket$agewks, lm.shearpsi.agewks$residuals, main="Residuals vs agewks")
```

```
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.shearpsi.agewks$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1] 5 6 1
# residuals vs order of data
plot(lm.shearpsi.agewks$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



The relationship between shear strength and age is fairly linear with predicted shear strength decreasing as the age of the propellant increases. The fitted LS line is

$$\text{Predicted shear strength} = 2627.8 - 37.2\ \text{Age}.$$

The test for $H_0 : \beta_1 = 0$ (zero slope for the population regression line) is highly significant: p-value< 0.0001. Also note that $R^2 = 0.9018$ so the linear relationship between shear strength and age explains about 90% of the variation in shear strength.

The data plot and residual information identify observations 5 and 6 as potential outliers ($r_5 = -2.38, r_6 = -2.32$). The predicted values for these observations are much greater than the observed shear strengths. These same observations appear as potential outliers in the normal scores plot and the plot of r_i against $\hat{Y}_i$. Observations 5 and 6 also have the largest influence on the analysis; see the Cook's distance values.

A sensible next step would be to repeat the analysis holding out the most influential case, observation 5. It should be somewhat clear that the influence of case 6 would increase dramatically once case 5 is omitted from the analysis. Since both cases have essentially the same effect on the positioning of the LS line, I will assess the impact of omitting both simultaneously.

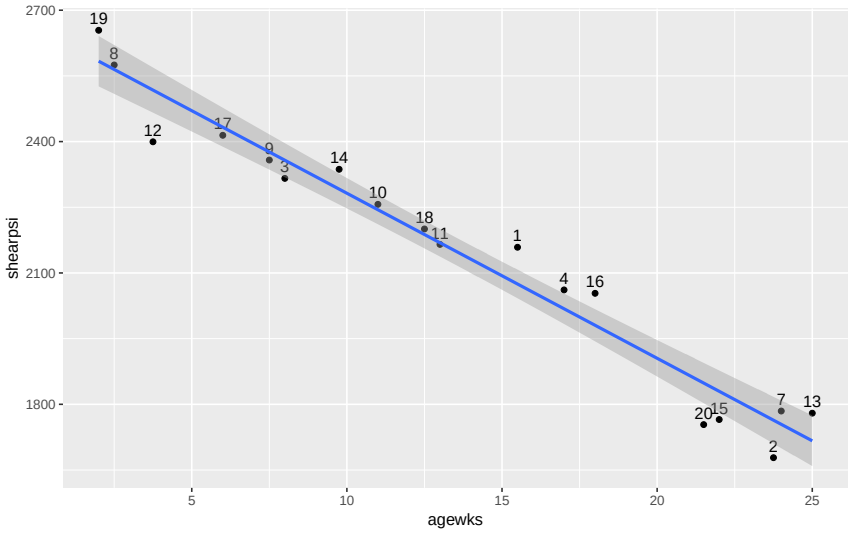
Before we hold out these cases, how do you think the LS line will change? My guess is these cases are pulling the LS line down, so the intercept of the LS line should increase once these cases are omitted. Holding out either case 5 or 6 would probably also affect the slope, but my guess is that when they are both omitted the slope will change little. (Is this my experience speaking, or have I already seen the output? Both.) What will happen to R^2 when we delete these points?

Exclude observations 5 and 6 and redo the analysis.

```
# exclude observations 5 and 6
rocket56 <- rocket[-c(5,6), ]
head(rocket56)
```

```
##   shearpsi agewks id
## 1  2158.70  15.50  1
## 2  1678.15  23.75  2
## 3  2316.00   8.00  3
## 4  2061.30  17.00  4
## 7  1784.70  24.00  7
## 8  2575.00   2.50  8
```

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(rocket56, aes(x = agewks, y = shearpsi, 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)
```



The data are reasonably linear, so fit the regression.

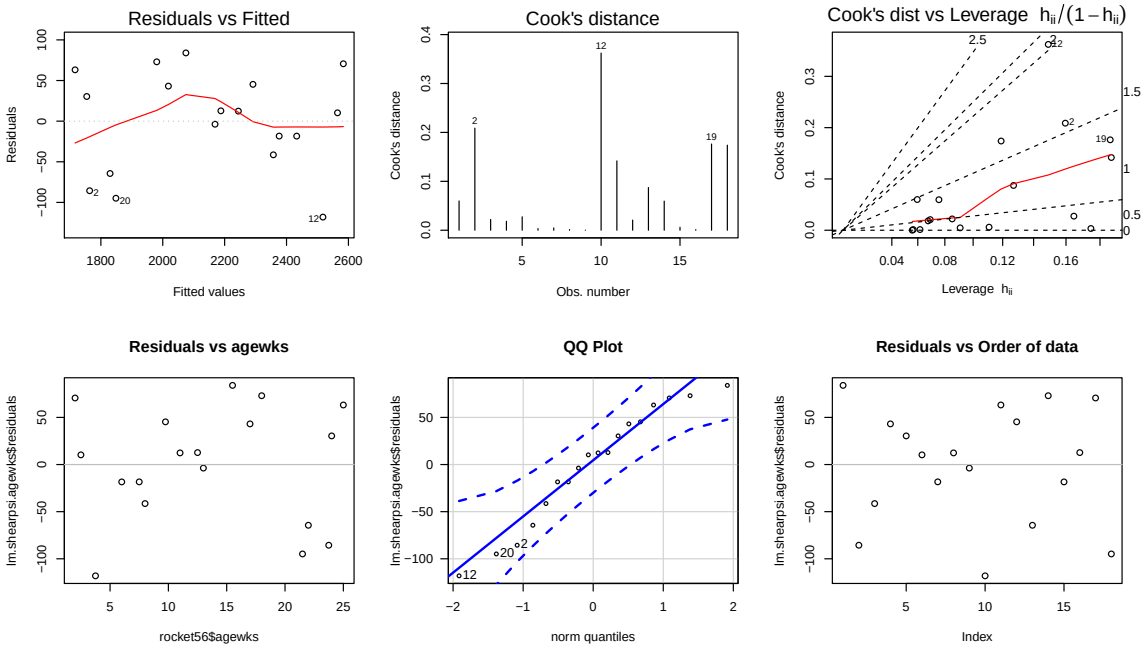
```
# fit the simple linear regression model
lm.shearpsi.agewks <- lm(shearpsi ~ agewks, data = rocket56)
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm.shearpsi.agewks)
```

```
##
## Call:
## lm(formula = shearpsi ~ agewks, data = rocket56)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -118.07  -35.67   11.31   44.75   83.98
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2658.973     30.533   87.08  < 2e-16 ***
## agewks       -37.694      1.979  -19.05 2.02e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 62.97 on 16 degrees of freedom
## Multiple R-squared:  0.9578,Adjusted R-squared:  0.9551
## F-statistic: 362.9 on 1 and 16 DF,  p-value: 2.023e-12
```

Plot diagnostics.

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.shearpsi.agewks, which = c(1,4,6))

# residuals vs weight
plot(rocket56$agewks, lm.shearpsi.agewks$residuals, main="Residuals vs agewks")
```

```
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.shearpsi.agewks$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## 12 20  2
## 10 18  2
# residuals vs order of data
plot(lm.shearpsi.agewks$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



Some summaries for the complete analysis, and when cases 5 and 6 are held out, are given below. The summaries lead to the following conclusions:

1. Holding out cases 5 and 6 has little effect on the estimated LS line. Predictions of shear strength are slightly larger after holding out these two cases (recall that intercept increased, but slope was roughly the same!)
2. Holding out these two cases decreases $\hat{\sigma}$ considerably, and leads to a modest increase in R^2. The complete data set will give wider CI and prediction intervals than the analysis which deletes case 5 and 6 because $\hat{\sigma}$ decreases when these points are omitted.
3. Once these cases are held out, the normal scores plot and plot of the studentized residuals against fitted values shows no significant problems. One observation

has a large Cook's D but does not appear to be extremely influential.

Without any substantive reason to explain the low shear strengths for cases 5 and 6, I am hesitant to delete either case from the analysis. I feel relatively confident that including these cases will not seriously limit the use of the model.

Feature	Full data	Omit 5 and 6
b_0	2627.82	2658.97
b_1	−37.15	−37.69
R^2	0.9018	0.9578
$\hat{\sigma}$	96.10	62.96
p-val for $H_0 : \beta_1 = 0$	0.0001	0.0001

Here is a comparison of the predicted or fitted values for selected observations in the data set, based on the two fits.

Observation	Actual Shear Strength	Pred, full	Pred, omit 5,6
1	2159	2052	2075
2	1678	1745	1764
4	2061	1996	2018
8	2575	2535	2565
10	2257	2219	2244
15	1765	1810	1830
18	2201	2163	2188
20	1754	1829	1849

Review complete

Now that we're warmed up, let's dive into new material!

Part VI

ADA2: Introduction to multiple regression and model selection

Chapter 2

Introduction to Multiple Linear Regression

In **multiple linear regression**, a linear combination of two or more predictor variables (xs) is used to explain the variation in a response. In essence, the additional predictors are used to explain the variation in the response not explained by a simple linear regression fit.

2.1 Indian systolic blood pressure example

Anthropologists conducted a study[1] to determine the long-term effects of an environmental change on systolic blood pressure. They measured the blood pressure and several other characteristics of 39 Indians who migrated from a very primitive environment high in the Andes into the mainstream of Peruvian society at a lower altitude. All of the Indians were males at least 21 years of age, and were born at a high altitude.

```
#### Example: Indian
# filename
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch02_indian.dat"
indian <- read.table(fn.data, header=TRUE)
# examine the structure of the dataset, is it what you expected?
  # a data.frame containing integers, numbers, and factors
str(indian)
```

```
## 'data.frame': 39 obs. of  11 variables:
##  $ id   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ age  : int  21 22 24 24 25 27 28 28 31 32 ...
##  $ yrmig: int  1 6 5 1 1 19 5 25 6 13 ...
##  $ wt   : num  71 56.5 56 61 65 62 53 53 65 57 ...
```

[1] This problem is from the Minitab handbook.

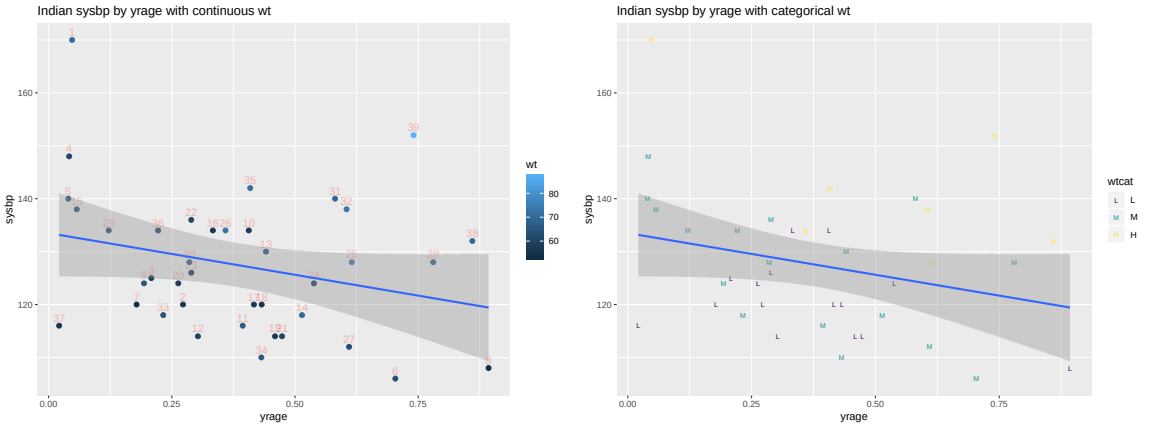
```
##  $ ht   : int  1629 1569 1561 1619 1566 1639 1494 1568 1540 1530 ...
##  $ chin : num  8 3.3 3.3 3.7 9 3 7.3 3.7 10.3 5.7 ...
##  $ fore : num  7 5 1.3 3 12.7 3.3 4.7 4.3 9 4 ...
##  $ calf : num  12.7 8 4.3 4.3 20.7 5.7 8 0 10 6 ...
##  $ pulse: int  88 64 68 52 72 72 64 80 76 60 ...
##  $ sysbp: int  170 120 125 148 140 106 120 108 124 134 ...
##  $ diabp: int  76 60 75 120 78 72 76 62 70 64 ...
# Description of variables
#    id = individual id
#   age = age in years           yrmig = years since migration
#    wt = weight in kilos           ht = height in mm
#  chin = chin skin fold in mm    fore = forearm skin fold in mm
#  calf = calf skin fold in mm   pulse = pulse rate-beats/min
# sysbp = systolic bp            diabp = diastolic bp

## print dataset to screen
#indian
```

	id	age	yrmig	wt	ht	chin	fore	calf	pulse	sysbp	diabp
1	1	21	1	71.00	1629	8.00	7.00	12.70	88	170	76
2	2	22	6	56.50	1569	3.30	5.00	8.00	64	120	60
3	3	24	5	56.00	1561	3.30	1.30	4.30	68	125	75
4	4	24	1	61.00	1619	3.70	3.00	4.30	52	148	120
5	5	25	1	65.00	1566	9.00	12.70	20.70	72	140	78
6	6	27	19	62.00	1639	3.00	3.30	5.70	72	106	72
7	7	28	5	53.00	1494	7.30	4.70	8.00	64	120	76
8	8	28	25	53.00	1568	3.70	4.30	0.00	80	108	62
9	9	31	6	65.00	1540	10.30	9.00	10.00	76	124	70
10	10	32	13	57.00	1530	5.70	4.00	6.00	60	134	64
11	11	33	13	66.50	1622	6.00	5.70	8.30	68	116	76
12	12	33	10	59.10	1486	6.70	5.30	10.30	73	114	74
13	13	34	15	64.00	1578	3.30	5.30	7.00	88	130	80
14	14	35	18	69.50	1645	9.30	5.00	7.00	60	118	68
15	15	35	2	64.00	1648	3.00	3.70	6.70	60	138	78
16	16	36	12	56.50	1521	3.30	5.00	11.70	72	134	86
17	17	36	15	57.00	1547	3.00	3.00	6.00	84	120	70
18	18	37	16	55.00	1505	4.30	5.00	7.00	64	120	76
19	19	37	17	57.00	1473	6.00	5.30	11.70	72	114	80
20	20	38	10	58.00	1538	8.70	6.00	13.00	64	124	64
21	21	38	18	59.50	1513	5.30	4.00	7.70	80	114	66
22	22	38	11	61.00	1653	4.00	3.30	4.00	76	136	78
23	23	38	11	57.00	1566	3.00	3.00	3.00	60	126	72
24	24	39	21	57.50	1580	4.00	3.00	5.00	64	124	62
25	25	39	24	74.00	1647	7.30	6.30	15.70	64	128	84
26	26	39	14	72.00	1620	6.30	7.70	13.30	68	134	92
27	27	41	25	62.50	1637	6.00	5.30	8.00	76	112	80
28	28	41	32	68.00	1528	10.00	5.00	11.30	60	128	82
29	29	41	5	63.40	1647	5.30	4.30	13.70	76	134	92
30	30	42	12	68.00	1605	11.00	7.00	10.70	88	128	90
31	31	43	25	69.00	1625	5.00	3.00	6.00	72	140	72
32	32	43	26	73.00	1615	12.00	4.00	5.70	68	138	74
33	33	43	10	64.00	1640	5.70	3.00	7.00	60	118	66
34	34	44	19	65.00	1610	8.00	6.70	7.70	74	110	70
35	35	44	18	71.00	1572	3.00	4.70	4.30	72	142	84
36	36	45	10	60.20	1534	3.00	3.00	3.30	56	134	70
37	37	47	1	55.00	1536	3.00	3.00	4.00	64	116	54
38	38	50	43	70.00	1630	4.00	6.00	11.70	72	132	90
39	39	54	40	87.00	1542	11.30	11.70	11.30	92	152	88

A question we consider concerns the long term effects of an environmental change on the systolic blood pressure. In particular, is there a relationship between the systolic blood pressure and how long the Indians lived in their new environment as measured by the fraction of their life spent in the new environment.

```
# Create the "fraction of their life" variable
#   yrage = years since migration divided by age
indian$yrage <- indian$yrmig / indian$age

# continuous color for wt
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(indian, aes(x = yrage, y = sysbp, label = id))
p <- p + geom_point(aes(colour=wt), size=2)
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5, alpha = 0.25, colour = 2)
# plot regression line and confidence band
```

```
p <- p + geom_smooth(method = lm)
p <- p + labs(title="Indian sysbp by yrage with continuous wt")
print(p)

# categorical color for wt
indian$wtcat <- rep(NA, nrow(indian))
indian$wtcat                    <- "M"  # init as medium
indian$wtcat[(indian$wt <  60)] <- "L"  # update low
indian$wtcat[(indian$wt >= 70)] <- "H"  # update high
# define as a factor variable with a specific order
indian$wtcat <- ordered(indian$wtcat, levels=c("L", "M", "H"))
#
library(ggplot2)
p <- ggplot(indian, aes(x = yrage, y = sysbp, label = id))
p <- p + geom_point(aes(colour=wtcat, shape=wtcat), size=2)
  library(R.oo) # for ascii code lookup
## Loading required package:  R.methodsS3
## R.methodsS3 v1.7.1 (2016-02-15) successfully loaded.  See ?R.methodsS3 for help.
## R.oo v1.22.0 (2018-04-21) successfully loaded.  See ?R.oo for help.
##
## Attaching package:  'R.oo'
## The following object is masked from 'package:igraph':
##
##    hierarchy
## The following objects are masked from 'package:gdata':
##
##    ll, trim
## The following objects are masked from 'package:methods':
##
##    getClasses, getMethods
## The following objects are masked from 'package:base':
##
##    attach, detach, gc, load, save
  p <- p + scale_shape_manual(values=charToInt(sort(unique(indian$wtcat))))
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
p <- p + labs(title="Indian sysbp by yrage with categorical wt")
print(p)
```



Fit the simple linear regression model reporting the ANOVA table ("Terms") and parameter estimate table ("Coefficients").

```
# fit the simple linear regression model
lm.sysbp.yrage <- lm(sysbp ~ yrage, data = indian)
# use Anova() from library(car) to get ANOVA table (Type 3 SS, df)
library(car)
Anova(lm.sysbp.yrage, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: sysbp
##             Sum Sq Df   F value  Pr(>F)
## (Intercept) 178221  1 1092.9484 < 2e-16 ***
## yrage          498  1    3.0544 0.08881 .
## Residuals     6033 37
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# use summary() to get t-tests of parameters (slope, intercept)
summary(lm.sysbp.yrage)
```

```
##
## Call:
## lm(formula = sysbp ~ yrage, data = indian)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.161 -10.987  -1.014   6.851  37.254
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  133.496      4.038  33.060   <2e-16 ***
## yrage        -15.752      9.013  -1.748   0.0888 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.77 on 37 degrees of freedom
```

```
## Multiple R-squared:  0.07626,Adjusted R-squared:  0.05129
## F-statistic: 3.054 on 1 and 37 DF,  p-value: 0.08881
```

A plot above of systolic blood pressure against `yrage` fraction suggests a weak linear relationship. Nonetheless, consider fitting the regression model

$$\text{sysbp} = \beta_0 + \beta_1 \text{ yrage} + \varepsilon.$$

The least squares line (already in the plot) is given by

$$\widehat{\text{sysbp}} = 133.5 + -15.75 \text{ yrage},$$

and suggests that average systolic blood pressure decreases as the fraction of life spent in modern society increases. However, the t-test of $H_0 : \beta_1 = 0$ is not significant at the 5% level (p-value=0.08881). That is, the weak linear relationship observed in the data is not atypical of a population where there is no linear relationship between systolic blood pressure and the fraction of life spent in a modern society.

Even if this test were significant, the small value of $R^2 = 0.07626$ suggests that `yrage` fraction does not explain a *substantial* amount of the variation in the systolic blood pressures. If we omit the individual with the highest blood pressure then the relationship would be weaker.

2.1.1 Taking Weight Into Consideration

At best, there is a weak relationship between systolic blood pressure and the `yrage` fraction. However, it is usually accepted that systolic blood pressure and weight are related. A natural way to take weight into consideration is to include `wt` (weight) and `yrage` fraction as predictors of systolic blood pressure in the multiple regression model:

$$\text{sysbp} = \beta_0 + \beta_1 \text{ yrage} + \beta_2 \text{ wt} + \varepsilon.$$

As in simple linear regression, the model is written in the form:

$$\text{Response} = \text{Mean of Response} + \text{Residual},$$

so the model implies that that average systolic blood pressure is a linear combination of `yrage` fraction and weight. As in simple linear regression, the standard multiple regression analysis assumes that the responses are normally distributed with a constant variance σ^2. The parameters of the regression model β_0, β_1, β_2, and σ^2 are estimated by least squares (LS).

Here is the multiple regression model with `yrage` and `wt` (weight) as predictors. Add `wt` to the right hand side of the previous formula statement.

```r
# fit the multiple linear regression model, (" + wt" added)
lm.sysbp.yrage.wt <- lm(sysbp ~ yrage + wt, data = indian)
library(car)
Anova(lm.sysbp.yrage.wt, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: sysbp
##             Sum Sq Df F value    Pr(>F)
## (Intercept) 1738.2  1  18.183 0.0001385 ***
## yrage       1314.7  1  13.753 0.0006991 ***
## wt          2592.0  1  27.115 7.966e-06 ***
## Residuals   3441.4 36
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(lm.sysbp.yrage.wt)
```

```
##
## Call:
## lm(formula = sysbp ~ yrage + wt, data = indian)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4330  -7.3070   0.8963   5.7275  23.9819
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  60.8959    14.2809   4.264 0.000138 ***
## yrage       -26.7672     7.2178  -3.708 0.000699 ***
## wt            1.2169     0.2337   5.207 7.97e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.777 on 36 degrees of freedom
## Multiple R-squared:  0.4731,Adjusted R-squared:  0.4438
## F-statistic: 16.16 on 2 and 36 DF,  p-value: 9.795e-06
```

2.1.2 Important Points to Notice About the Regression Output

1. The LS estimates of the intercept and the regression coefficient for `yrage` fraction, and their standard errors, change from the simple linear model to the multiple regression model. For the simple linear regression:

$$\widehat{\text{sysbp}} = 133.50 - 15.75 \text{ yrage}.$$

For the multiple regression model:

$$\widehat{\text{sysbp}} = 60.89 - 26.76 \text{ yrage} + 1.21 \text{ wt.}$$

2. Looking at the ANOVA tables for the simple linear and the multiple regression models we see that the Regression (model) *df* has increased from 1 to 2 (2=number of predictor variables) and the Residual (error) *df* has decreased from 37 to 36 ($=n-1-$ number of predictors). Adding a predictor increases the Regression *df* by 1 and decreases the Residual *df* by 1.
3. The Residual SS decreases by $6033.37 - 3441.36 = 2592.01$ upon adding the weight term. The Regression SS increased by 2592.01 upon adding the weight term term to the model. The Total SS does not depend on the number of predictors so it stays the same. The Residual SS, or the part of the variation in the response unexplained by the regression model, never increases when new predictors are added. (You can't add a predictor and explain less variation.)
4. The proportion of variation in the response explained by the regression model:

$$R^2 = \text{Regression SS/Total SS}$$

never decreases when new predictors are added to a model. The R^2 for the simple linear regression was 0.076, whereas

$$R^2 = 3090.08/6531.44 = 0.473$$

for the multiple regression model. Adding the weight variable to the model increases R^2 by 40%. That is, weight explains 40% of the variation in systolic blood pressure not already explained by fraction.
5. The estimated variability about the regression line

$$\text{Residual MS} = \hat{\sigma}^2$$

decreased dramatically after adding the weight effect. For the simple linear regression model $\hat{\sigma}^2 = 163.06$, whereas $\hat{\sigma}^2 = 95.59$ for the multiple regression model. This suggests that an important predictor has been added to model.
6. The F-statistic for the multiple regression model

$$F_{obs} = \text{Regression MS/Residual MS} = 1545.04/95.59 = 16.163$$

(which is compared to a F-table with 2 and 36 *df*) tests $H_0 : \beta_1 = \beta_2 = 0$ against H_A : not H_0. This is a test of no relationship between the average systolic blood pressure and fraction and weight, assuming the relationship is linear. If this test is significant, then either fraction or weight, or both, are important for explaining the variation in systolic blood pressure.

7. Given the model
$$\text{sysbp} = \beta_0 + \beta_1 \text{ yrage} + \beta_2 \text{ wt} + \varepsilon,$$
we are interested in testing $H_0 : \beta_2 = 0$ against $H_A : \beta_2 \neq 0$. The t-statistic for this test
$$t_{obs} = \frac{b_2 - 0}{\text{SE}(b_2)} = \frac{1.217}{0.234} = 5.207$$
is compared to a t-critical value with Residual $df = 36$. The test gives a p-value of < 0.0001, which suggests $\beta_2 \neq 0$. The t-test of $H_0 : \beta_2 = 0$ in the multiple regression model tests whether adding weight to the simple linear regression model explains a significant part of the variation in systolic blood pressure not explained by `yrage` fraction. In some sense, the t-test of $H_0 : \beta_1 = 0$ will be significant if the increase in R^2 (or decrease in Residual SS) obtained by adding weight to this simple linear regression model is substantial. We saw a big increase in R^2, which is deemed significant by the t-test. A similar interpretation is given to the t-test for $H_0 : \beta_1 = 0$.
8. The t-tests for $\beta_0 = 0$ and $\beta_1 = 0$ are conducted, assessed, and interpreted in the same manner. The p-value for testing $H_0 : \beta_0 = 0$ is 0.0001, whereas the p-value for testing $H_0 : \beta_1 = 0$ is 0.0007. This implies that fraction is important in explaining the variation in systolic blood pressure **after** weight is taken into consideration (by including weight in the model as a predictor).
9. We compute CIs for the regression parameters in the usual way: $b_i + t_{\text{crit}} SE(b_i)$, where t_{crit} is the t-critical value for the corresponding CI level with df = Residual df.

2.1.3 Understanding the Model

The t-test for $H_0 : \beta_1 = 0$ is highly significant (p-value=0.0007), which implies that fraction is important in explaining the variation in systolic blood pressure **after weight is taken into consideration** (by including weight in the model as a predictor). Weight is called a **suppressor variable**. Ignoring weight suppresses the relationship between systolic blood pressure and `yrage` fraction.

The implications of this analysis are enormous! Essentially, the correlation between a predictor and a response says very little about the importance of the predictor in a regression model with one or more additional predictors. This conclusion also holds in situations where the correlation is high, in the sense that a predictor that is highly correlated with the response may be unimportant in a multiple regression model once other predictors are included in the model. In multiple regression *"everything depends on everything else."*

I will try to convince you that this was expected, given the plot of systolic blood pressure against fraction. This plot used a weight category variable `wtcat` L, M, or H

as a plotting symbol. The relationship between systolic blood pressure and fraction is fairly linear within each weight category, and stronger than when we ignore weight. The slopes in the three groups are negative and roughly constant.

To see why `yrage` fraction is an important predictor after taking weight into consideration, let us return to the multiple regression model. The model implies that the average systolic blood pressure is a linear combination of `yrage` fraction and weight:

$$\widehat{\text{sysbp}} = \beta_0 + \beta_1 \text{ yrage} + \beta_2 \text{ wt}.$$

For each fixed weight, the average systolic blood pressure is linearly related to `yrage` fraction with a constant slope β_1, independent of weight. A similar interpretation holds if we switch the roles of `yrage` fraction and weight. That is, if we fix the value of fraction, then the average systolic blood pressure is linearly related to weight with a constant slope β_2, independent of `yrage` fraction.

To see this point, suppose that the LS estimates of the regression parameters are the true values

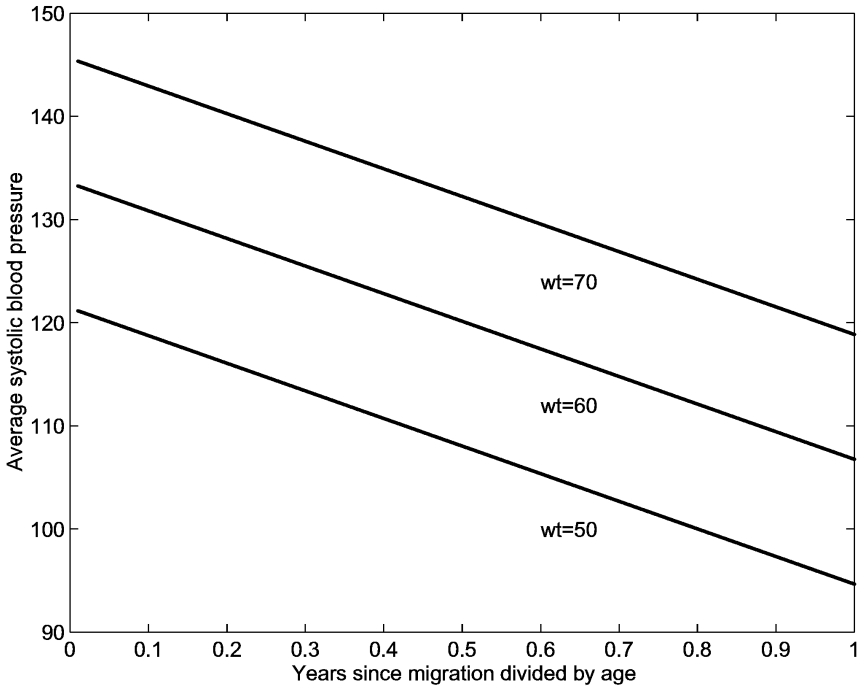
$$\widehat{\text{sysbp}} = 60.89 - 26.76 \text{ yrage} + 1.21 \text{ wt}.$$

If we restrict our attention to 50kg Indians, the average systolic blood pressure as a function of fraction is

$$\widehat{\text{sysbp}} = 60.89 - 26.76 \text{ yrage} + 1.21(50) = 121.39 - 26.76 \text{ yrage}.$$

For 60kg Indians,

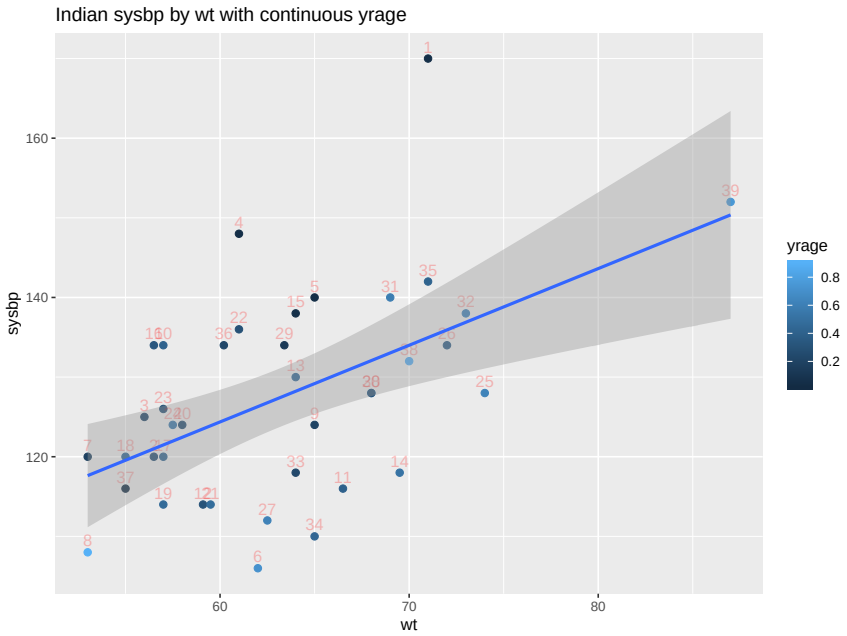
$$\widehat{\text{sysbp}} = 60.89 - 26.76 \text{ yrage} + 1.21(60) = 133.49 - 26.76 \text{ yrage}.$$



Hopefully the pattern is clear: the average systolic blood pressure decreases by 26.76 for each increase of 1 on fraction, regardless of one's weight. If we vary weight over its range of values, we get a set of parallel lines (i.e., equal slopes) when we plot average systolic blood pressure as a function of `yrage` fraction. The intercept increases by 1.21 for each increase of 1kg in weight.

Similarly, if we plot the average systolic blood pressure as a function of weight, for several fixed values of fraction, we see a set of parallel lines with slope 26.76, and intercepts decreasing by 26.76 for each increase of 1 in fraction.

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(indian, aes(x = wt, y = sysbp, label = id))
p <- p + geom_point(aes(colour=yrage), size=2)
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5, alpha = 0.25, colour = 2)
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
p <- p + labs(title="Indian sysbp by wt with continuous yrage")
print(p)
```



If we had more data we could check the model by plotting systolic blood pressure against fraction, broken down by individual weights. The plot should show a fairly linear relationship between systolic blood pressure and fraction, with a constant slope across weights. I grouped the weights into categories because of the limited number of observations. The same phenomenon should approximately hold, and it does. If the slopes for the different weight groups changed drastically with weight, but the relationships were linear, we would need to include an **interaction** or product variable

wt $\times$ yrage in the model, in addition to weight and `yrage` fraction. This is probably not warranted here.

A final issue that I wish to address concerns the interpretation of the estimates of the regression coefficients in a multiple regression model. For the fitted model

$$\widehat{\text{sysbp}} = 60.89 - 26.76 \text{ yrage} + 1.21 \text{ wt}$$

our interpretation is consistent with the explanation of the regression model given above. For example, focus on the `yrage` fraction coefficient. The negative coefficient indicates that the predicted systolic blood pressure decreases as `yrage` fraction increases **holding weight constant**. In particular, the predicted systolic blood pressure decreases by 26.76 for each unit increase in fraction, holding weight constant at any value. Similarly, the predicted systolic blood pressure increases by 1.21 for each unit increase in weight, holding `yrage` fraction constant at any level.

This example was meant to illustrate multiple regression. A more complete analysis of the data, including diagnostics, will be given later.

2.2 GCE exam score example

The data below are selected from a larger collection of data referring to candidates for the General Certificate of Education (GCE) who were being considered for a special award. Here, Y denotes the candidate's total mark, out of 1000, in the GCE exam, while X_1 is the candidate's score in the compulsory part of the exam, which has a maximum score of 200 of the 1000 points on the exam. X_2 denotes the candidates' score, out of 100, in a School Certificate English Language paper taken on a previous occasion.

```
#### Example: GCE
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch02_gce.dat"
gce <- read.table(fn.data, header=TRUE)
str(gce)
```

```
## 'data.frame': 15 obs. of  3 variables:
##  $ y : int  476 457 540 551 575 698 545 574 645 690 ...
##  $ x1: int  111 92 90 107 98 150 118 110 117 114 ...
##  $ x2: int  68 46 50 59 50 66 54 51 59 80 ...
```

```
## print dataset to screen
#gce
```

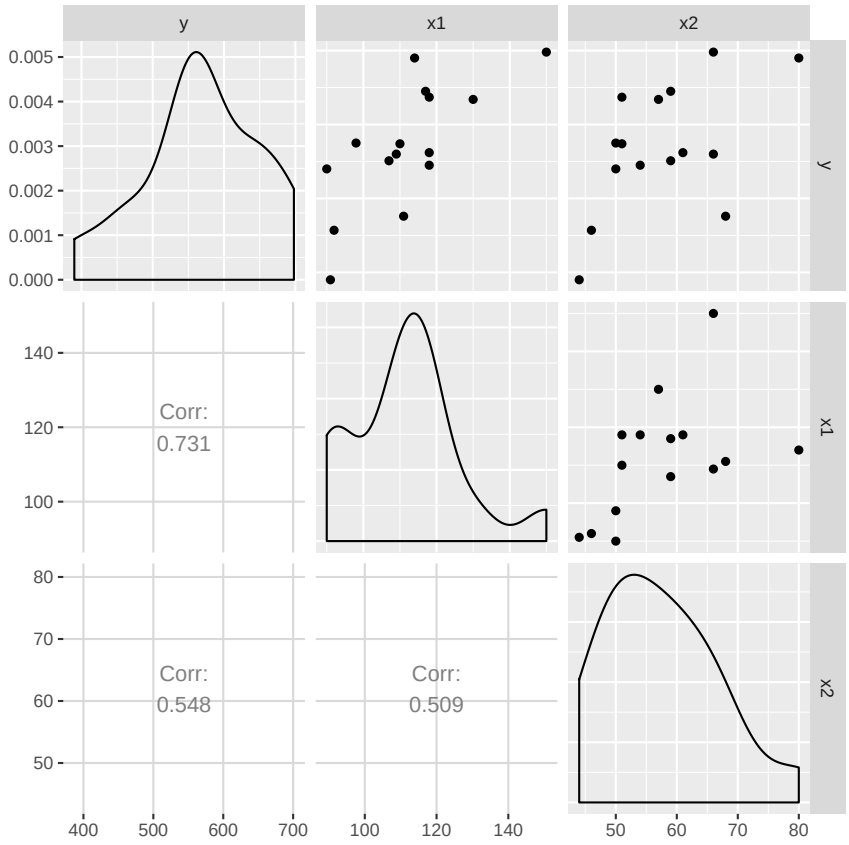
	y	x1	x2
1	476	111	68
2	457	92	46
3	540	90	50
4	551	107	59
5	575	98	50
6	698	150	66
7	545	118	54
8	574	110	51
9	645	117	59
10	690	114	80
11	634	130	57
12	637	118	51
13	390	91	44
14	562	118	61
15	560	109	66

A goal here is to compute a multiple regression of Y on X_1 and X_2, and make the necessary tests to enable you to comment intelligently on the extent to which current performance in the compulsory test (X_1) may be used to predict aggregate performance on the GCE exam (Y), and on whether previous performance in the School Certificate English Language (X_2) has any predictive value independently of what has already emerged from the current performance in the compulsory papers.

I will lead you through a number of steps to help you answer this question. Let us answer the following straightforward questions.

1. Plot Y against X_1 and X_2 individually, and comment on the form (i.e., linear, non-linear, logarithmic, etc.), strength, and direction of the relationships.
2. Plot X_1 against X_2 and comment on the form, strength, and direction of the relationship.
3. Compute the correlation between all pairs of variables. Do the correlations appear sensible, given the plots?

```
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
#p <- ggpairs(gce, progress=FALSE)
# put scatterplots on top so y axis is vertical
p <- ggpairs(gce, upper = list(continuous = "points")
                , lower = list(continuous = "cor")
                , progress=FALSE
             )
print(p)

  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```




```
# correlation matrix and associated p-values testing "H0: rho == 0"
library(Hmisc)
```

```
## Loading required package:  Formula
##
## Attaching package:  'Hmisc'
## The following objects are masked from 'package:xtable':
##
##    label, label<-
## The following objects are masked from 'package:plyr':
##
##    is.discrete, summarize
## The following objects are masked from 'package:TeachingDemos':
##
##    cnvrt.coords, subplot
## The following objects are masked from 'package:base':
##
##    format.pval, units
```

```
rcorr(as.matrix(gce))
```

```
##       y   x1   x2
## y  1.00 0.73 0.55
```

```
## x1 0.73 1.00 0.51
## x2 0.55 0.51 1.00
##
## n= 15
##
##
## P
##    y      x1     x2
## y         0.0020 0.0346
## x1 0.0020        0.0527
## x2 0.0346 0.0527
```

In parts 4 through 9, ignore the possibility that Y, X_1 or X_2 might ideally need to be transformed.

4. Which of X_1 and X_2 explains a larger proportion of the variation in Y? Which would appear to be a better predictor of Y? (Explain).
   Model $Y = \beta_0 + \beta_1 X_1 + \varepsilon$:

```
# y ~ x1
lm.y.x1 <- lm(y ~ x1, data = gce)
library(car)
Anova(lm.y.x1, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: y
##             Sum Sq Df F value   Pr(>F)
## (Intercept)   4515  1   1.246 0.284523
## x1           53970  1  14.895 0.001972 **
## Residuals    47103 13
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm.y.x1)
```

```
##
## Call:
## lm(formula = y ~ x1, data = gce)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -97.858 -33.637  -0.034  48.507 111.327
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  128.548    115.160   1.116  0.28452
## x1             3.948      1.023   3.859  0.00197 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.19 on 13 degrees of freedom
```

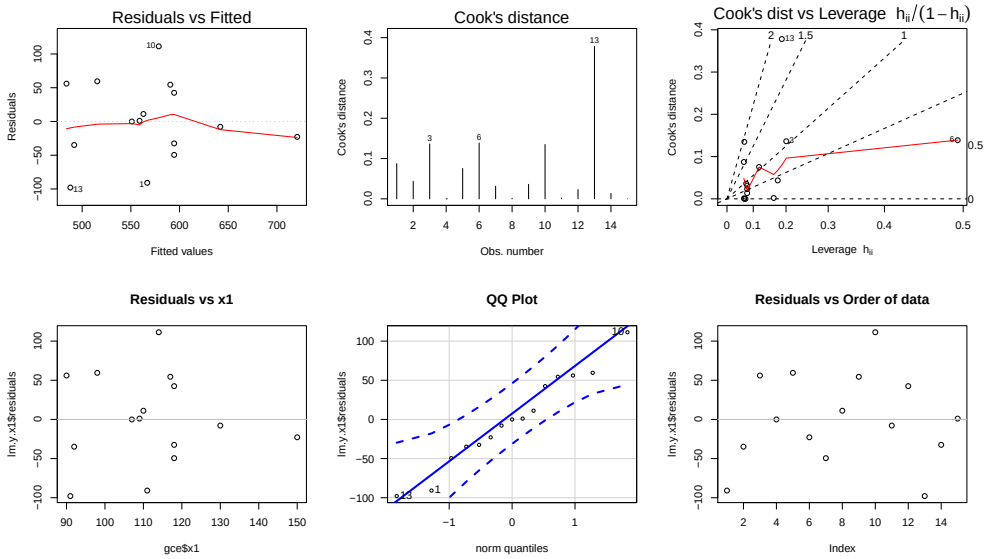
```
## Multiple R-squared:  0.534,Adjusted R-squared:  0.4981
## F-statistic:  14.9 on 1 and 13 DF,  p-value: 0.001972
```

Plot diagnostics.

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.y.x1, which = c(1,4,6))

plot(gce$x1, lm.y.x1$residuals, main="Residuals vs x1")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.y.x1$residuals, las = 1, id = list(n = 3), main="QQ Plot")
```

```
## [1] 10 13  1
```

```
# residuals vs order of data
plot(lm.y.x1$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



Model $Y = \beta_0 + \beta_1 X_2 + \varepsilon$:

```
# y ~ x2
lm.y.x2 <- lm(y ~ x2, data = gce)
library(car)
Anova(lm.y.x2, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: y
```

```
##             Sum Sq Df F value  Pr(>F)
## (Intercept)  32656  1  6.0001 0.02924 *
## x2           30321  1  5.5711 0.03455 *
## Residuals    70752 13
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

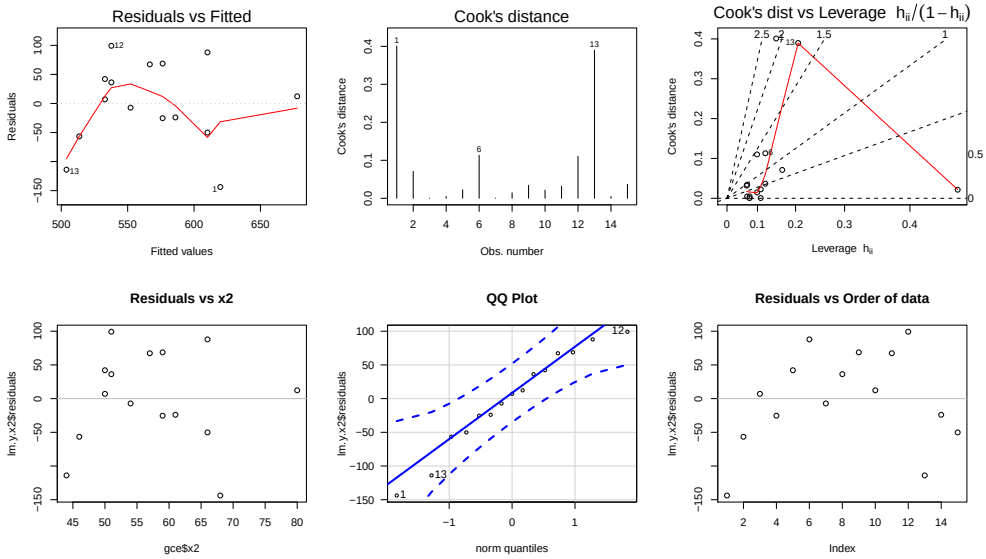
```
summary(lm.y.x2)
```

```
##
## Call:
## lm(formula = y ~ x2, data = gce)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -143.770  -37.725    7.103   54.711   99.276
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  291.586    119.038    2.45   0.0292 *
## x2             4.826      2.045    2.36   0.0346 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 73.77 on 13 degrees of freedom
## Multiple R-squared:    0.3,Adjusted R-squared:  0.2461
## F-statistic: 5.571 on 1 and 13 DF,  p-value: 0.03455
```

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.y.x2, which = c(1,4,6))

plot(gce$x2, lm.y.x2$residuals, main="Residuals vs x2")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.y.x2$residuals, las = 1, id = list(n = 3), main="QQ Plot")
```

```
## [1]  1 13 12
```

```
# residuals vs order of data
plot(lm.y.x2$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



Answer: R^2 is 0.53 for the model with X_1 and 0.30 with X_2. Equivilantly, the Model SS is larger for X_1 (53970) than for X_2 (30321). Thus, X_1 appears to be a better predictor of Y than X_2.

5. Consider 2 simple linear regression models for predicting Y, one with X_1 as a predictor, and the other with X_2 as the predictor. Do X_1 and X_2 individually appear to be important for explaining the variation in Y? (i.e., test that the slopes of the regression lines are zero). Which, if any, of the output, support, or contradicts, your answer to the previous question?
   Answer: The model with X_1 has a t-statistic of 3.86 with an associated p-value of 0.0020, while X_2 has a t-statistic of 2.36 with an associated p-value of 0.0346. Both predictors explain a significant amount of variability in Y. This is consistant with part (4).
6. Fit the multiple regression model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon.$$

   Test $H_0 : \beta_1 = \beta_2 = 0$ at the 5% level. Describe in words what this test is doing, and what the results mean here.
   Model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$:

```
# y ~ x1 + x2
lm.y.x1.x2 <- lm(y ~ x1 + x2, data = gce)
library(car)
Anova(lm.y.x1.x2, type=3)

## Anova Table (Type III tests)
##
## Response: y
```

```
##             Sum Sq Df F value  Pr(>F)
## (Intercept)   1571  1  0.4396 0.51983
## x1           27867  1  7.7976 0.01627 *
## x2            4218  1  1.1802 0.29866
## Residuals    42885 12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm.y.x1.x2)
```

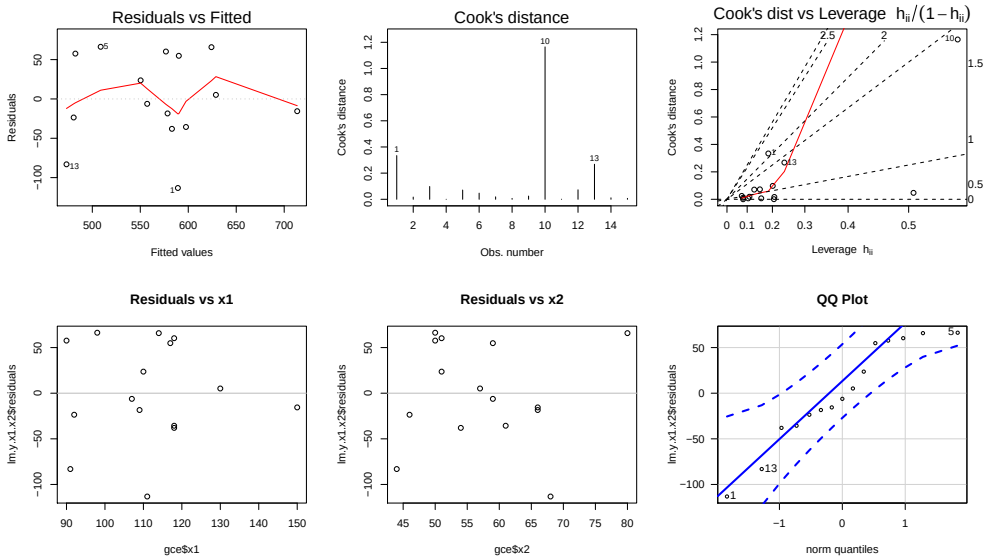
```
##
## Call:
## lm(formula = y ~ x1 + x2, data = gce)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -113.201  -29.605   -6.198   56.247   66.285
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   81.161    122.406   0.663   0.5198
## x1             3.296      1.180   2.792   0.0163 *
## x2             2.091      1.925   1.086   0.2987
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.78 on 12 degrees of freedom
## Multiple R-squared:  0.5757,Adjusted R-squared:  0.505
## F-statistic: 8.141 on 2 and 12 DF,  p-value: 0.005835
```

Diagnostic plots suggest the residuals are roughly normal with no substantial outliers, though the Cook's distance is substantially larger for observation 10. We may wish to fit the model without observation 10 to see whether conclusions change.

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.y.x1.x2, which = c(1,4,6))

plot(gce$x1, lm.y.x1.x2$residuals, main="Residuals vs x1")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(gce$x2, lm.y.x1.x2$residuals, main="Residuals vs x2")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.y.x1.x2$residuals, las = 1, id = list(n = 3), main="QQ Plot")
```

```
## [1]  1 13  5
```

```
## residuals vs order of data
#plot(lm.y.x1.x2$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Answer: The ANOVA table reports an F-statistic of 8.14 with associated p-value of 0.0058 indicating that the regression model with both X_1 and X_2 explains significantly more variability in Y than a model with the intercept, alone. That is, X_1 and X_2 explain variability in Y together. This does not tell us which of or whether X_1 or X_2 are individually important (recall the results of the Indian systolic blood pressure example).

7. In the multiple regression model, test $H_0 : \beta_1 = 0$ and $H_0 : \beta_2 = 0$ individually. Describe in words what these tests are doing, and what the results mean here.
   Answer: Each hypothesis is testing, conditional on all other predictors being in the model, whether the addition of the predictor being tested explains significantly more variability in Y than without it.
   For $H_0 : \beta_1 = 0$, the t-statistic is 2.79 with an associated p-value of 0.0163. Thus, we reject H_0 in favor of the alternative that the slope is statistically significantly different from 0 conditional on X_2 being in the model. That is, X_1 explains significantly more variability in Y given that X_2 is already in the model.
   For $H_0 : \beta_2 = 0$, the t-statistic is 1.09 with an associated p-value of 0.2987. Thus, we fail to reject H_0 concluding that there is insufficient evidence that the slope is different from 0 conditional on X_1 being in the model. That is, X_2 does not explain significantly more variability in Y given that X_1 is already in the model.
8. How does the R^2 from the multiple regression model compare to the R^2 from the individual simple linear regressions? Is what you are seeing here appear reasonable, given the tests on the individual coefficients?
   Answer: The R^2 for the model with only X_1 is 0.5340, only X_2 is 0.3000, and

both X_1 and X_2 is 0.5757. There is only a very small increase in R^2 from the model with only X_1 when X_2 is added, which is consistent with X_2 not being important given that X_1 is already in the model.

9. Do your best to answer the question posed above, in the paragraph after the data "A goal ...". Provide an equation (LS) for predicting Y.
   Answer: Yes, we've seen that X_1 may be used to predict Y, and that X_2 does not explain significantly more variability in the model with X_1. Thus, the preferred model has only X_1:

$$\hat{y} = 128.55 + 3.95X_1.$$

2.2.1 Some Comments on GCE Analysis

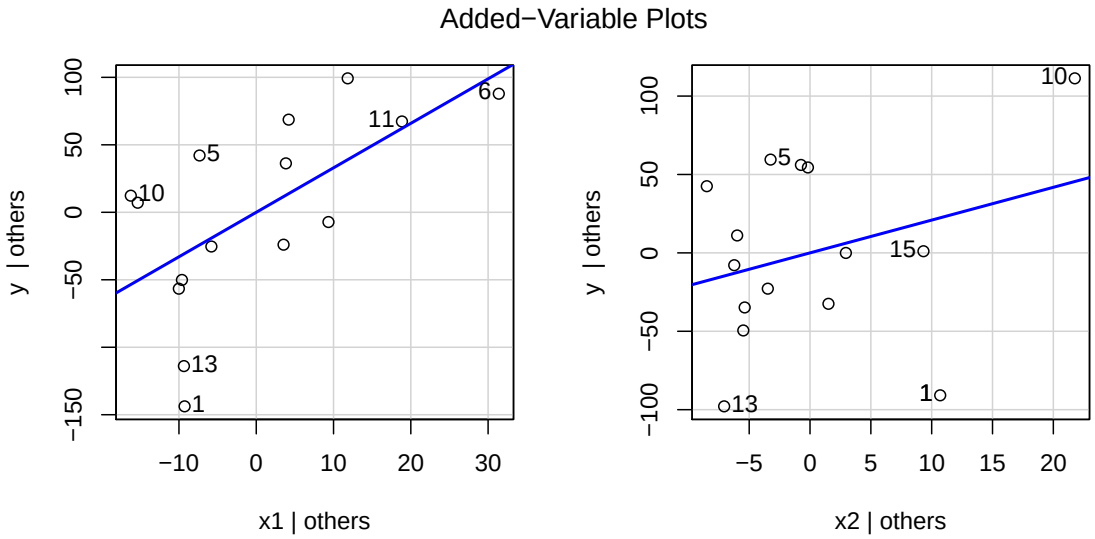
I will give you my thoughts on these data, and how I would attack this problem, keeping the ultimate goal in mind. I will examine whether transformations of the data are appropriate, and whether any important conclusions are dramatically influenced by individual observations. I will use some new tools to attack this problem, and will outline how they are used.

The plot of GCE (Y) against COMP (X_1) is fairly linear, but the trend in the plot of GCE (Y) against SCEL (X_2) is less clear. You might see a non-linear trend here, but the relationship is not very strong. When I assess plots I try to not allow a few observations affect my perception of trend, and with this in mind, I do not see any strong evidence at this point to transform any of the variables.

One difficulty that we must face when building a multiple regression model is that these two-dimensional (2D) plots of a response against individual predictors may have little information about the appropriate scales for a multiple regression analysis. In particular, the 2D plots only tell us whether we need to transform the data in a simple linear regression analysis. If a 2D plot shows a strong non-linear trend, I would do an analysis using the suggested transformations, including any other effects that are important. However, it might be that no variables need to be transformed in the multiple regression model.

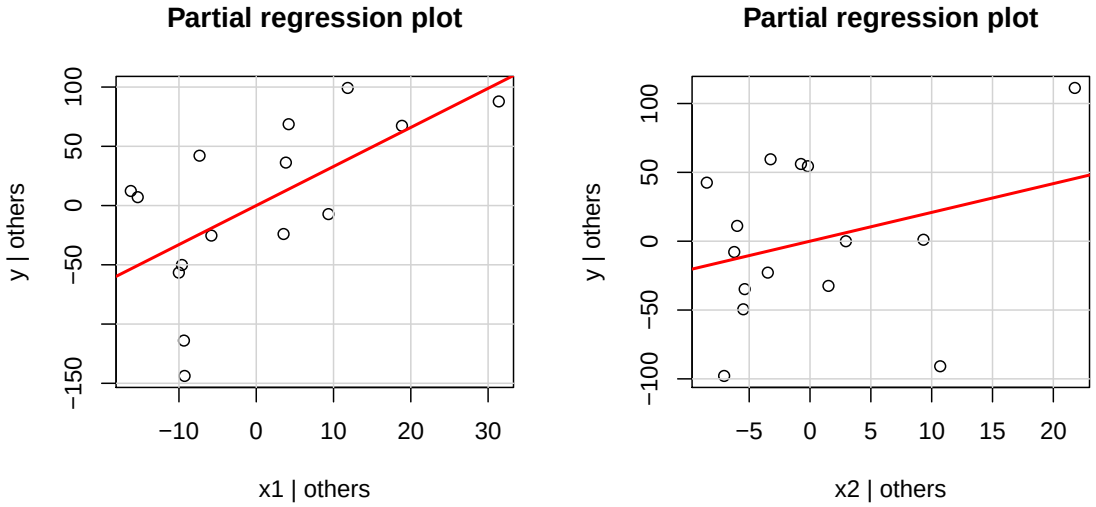
The **partial regression residual plot**, or added variable plot, is a graphical tool that provides information about the need for transformations in a multiple regression model. The following **reg** procedure generates diagnostics and the partial residual plots for each predictor in the multiple regression model that has COMP and SCEL as predictors of GCE.

```
library(car)
avPlots(lm.y.x1.x2, id = list(n = 3))
```



The partial regression residual plot compares the residuals from two model fits. First, we "adjust" Y for all the other predictors in the model except the selected one. Then, we "adjust" the selected variable X_{sel} for all the other predictors in the model. Lastly, plot the residuals from these two models against each other to see what relationship still exists between Y and X_{sel} after accounting for their relationships with the other predictors.

```
# function to create partial regression plot
partial.regression.plot <- function (y, x, sel, ...) {
  m <- as.matrix(x[, -sel])
  # residuals of y regressed on all x's except "sel"
  y1 <- lm(y ~ m)$res
  # residuals of x regressed on all other x's
  x1 <- lm(x[, sel] ~ m)$res
  # plot residuals of y vs residuals of x
  plot( y1 ~ x1, main="Partial regression plot", ylab="y | others", ...)
  # add grid
  grid(lty = "solid")
  # add red regression line
  abline(lm(y1 ~ x1), col = "red", lwd = 2)
}

par(mfrow=c(1, 2))
partial.regression.plot(gce$y, cbind(gce$x1, gce$x2), 1, xlab="x1 | others")
partial.regression.plot(gce$y, cbind(gce$x1, gce$x2), 2, xlab="x2 | others")
```



The first partial regression residual plot for COMP, given below, "**adjusts**" GCE (Y) and COMP (X_1) for their common dependence on all the other predictors in the model (only SCEL (X_2) here). This plot tells us whether we need to transform COMP in the multiple regression model, and whether any observations are influencing the significance of COMP in the fitted model. A roughly linear trend suggests that no transformation of COMP is warranted. The positive relationship seen here is consistent with the coefficient of COMP being positive in the multiple regression model. The partial residual plot for COMP shows little evidence of curvilinearity, and much less so than the original 2D plot of GCE against COMP. This indicates that there is no strong evidence for transforming COMP in a multiple regression model that includes SCEL.

Although SCEL appears to somewhat useful as a predictor of GCE on it's own, the multiple regression output indicates that SCEL does not explain a significant amount of the variation in GCE, once the effect of COMP has been taken into account. Put another way, previous performance in the School Certificate English Language (X_2) has little predictive value independently of what has already emerged from the current performance in the compulsory papers (X_1 or COMP). This conclusion is consistent with the fairly weak linear relationship between GCE against SCEL seen in the second partial residual plot.

Do diagnostics suggest any deficiencies associated with this conclusion? The partial residual plot of SCEL highlights observation 10, which has the largest value of Cook's distance in the multiple regression model. If we visually hold observation 10 out from this partial residual plot, it would appear that the relationship observed in this plot would weaken. This suggests that observation 10 is actually **enhancing** the significance of SCEL in the multiple regression model. That is, the p-value for testing

the importance of SCEL in the multiple regression model would be inflated by holding out observation 10. The following output confirms this conjecture. The studentized residuals, Cook's distances and partial residual plots show no serious deficiencies.

Model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$, excluding observation 10:

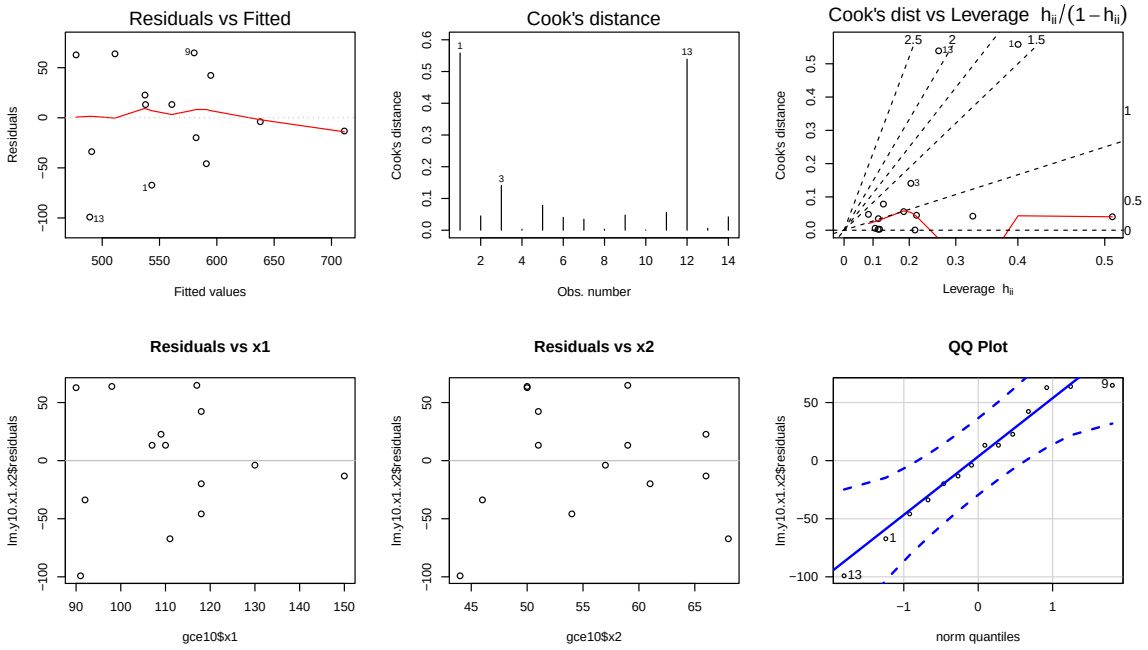
```
gce10 <- gce[-10,]
# y ~ x1 + x2
lm.y10.x1.x2 <- lm(y ~ x1 + x2, data = gce10)
library(car)
Anova(lm.y10.x1.x2, type=3)
## Anova Table (Type III tests)
##
## Response: y
##             Sum Sq Df F value   Pr(>F)
## (Intercept)   5280  1  1.7572 0.211849
## x1           37421  1 12.4540 0.004723 **
## x2             747  1  0.2486 0.627870
## Residuals    33052 11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.y10.x1.x2)
##
## Call:
## lm(formula = y ~ x1 + x2, data = gce10)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -99.117 -30.319   4.661  37.416  64.803
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  159.461    120.295   1.326  0.21185
## x1             4.241      1.202   3.529  0.00472 **
## x2            -1.280      2.566  -0.499  0.62787
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 54.82 on 11 degrees of freedom
## Multiple R-squared:  0.6128,Adjusted R-squared:  0.5424
## F-statistic: 8.706 on 2 and 11 DF,  p-value: 0.005413
```

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.y10.x1.x2, which = c(1,4,6))

plot(gce10$x1, lm.y10.x1.x2$residuals, main="Residuals vs x1")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(gce10$x2, lm.y10.x1.x2$residuals, main="Residuals vs x2")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
```

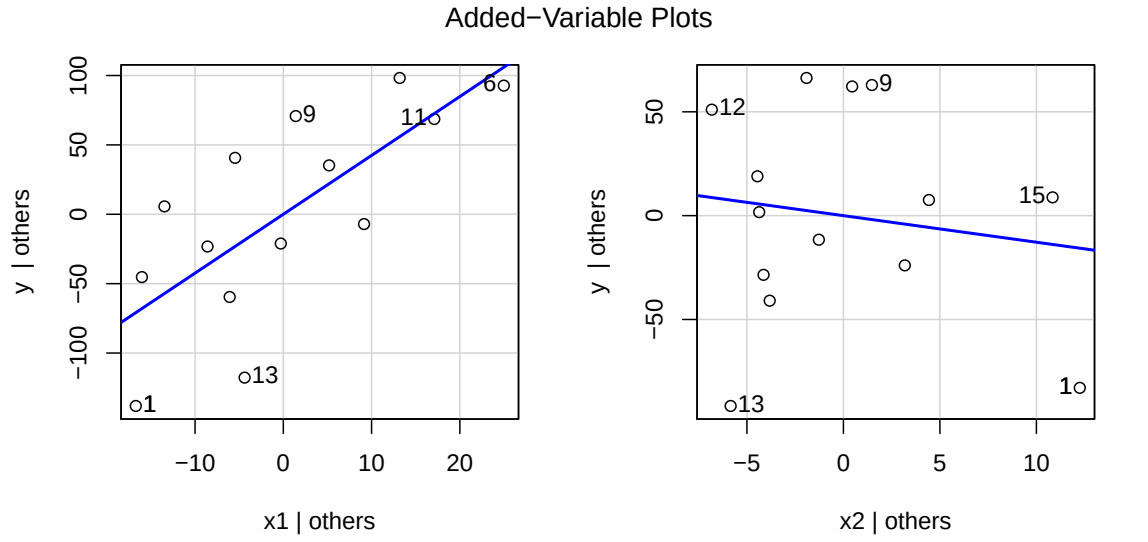
```
library(car)
qqPlot(lm.y10.x1.x2$residuals, las = 1, id = list(n = 3), main="QQ Plot")
```

```
## 13  1  9
## 12  1  9
```

```
## residuals vs order of data
#plot(lm.y10.x1.x2$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```




```
library(car)
avPlots(lm.y10.x1.x2, id = list(n = 3))
```



What are my conclusions? It would appear that SCEL (X_2) is not a useful predictor in the multiple regression model. For simplicity, I would likely use a simple linear regression model to predict GCE (Y) from COMP (X_1) only. The diagnostic analysis of the model showed no serious deficiencies.

Chapter 3

A Taste of Model Selection for Multiple Regression

3.1 Model

Given data on a response variable Y and k predictor variables $X_1, X_2, \ldots, X_k$, we wish to develop a regression model to predict Y. Assuming that the collection of variables is measured on the correct scale, and that the candidate list of predictors includes all the important predictors, the most general model is

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k + \varepsilon.$$

In most problems one or more of the predictors can be eliminated from this general or **full model** without (much) loss of information. We want to identify the important predictors, or equivalently, eliminate the predictors that are not very useful for explaining the variation in Y (conditional on the other predictors in the model).

We will study several **automated** methods for model selection, which, given a specific criterion for selecting a model, gives the best predictors. Before applying any of the methods, you should plot Y against each predictor $X_1, X_2, \ldots, X_k$ to see whether transformations are needed. If a transformation of X_i is suggested, include the transformation along with the original X_i in the candidate list. Note that you can transform the predictors differently, for example, $\log(X_1)$ and $\sqrt{X_2}$. However, if several transformations are suggested for the response, then you should consider doing one analysis for each suggested response scale before deciding on the final scale.

At this point, I will only consider the **backward elimination method**. Other approaches will be addressed later this semester.

3.2 Backward Elimination

The backward elimination procedure deletes unimportant variables, one at a time, starting from the full model. The steps is the procedure are:

1. Fit the full model

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k + \varepsilon.$$

2. Find the variable which when omitted from the full model (1) reduces R^2 the least, or equivalently, increases the Residual SS the least. This is the variable that gives the largest p-value for testing an individual regression coefficient $H_0 : \beta_i = 0$ for $i > 0$. Suppose this variable is X_k. If you reject H_0, stop and conclude that the full model is best. If you do not reject H_0, delete X_k from the full model, giving the **new full model**

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_{k-1} X_{k-1} + \varepsilon.$$

Repeat steps 1 and 2 sequentially until no further predictors can be deleted.

In backward elimination we isolate the least important predictor left in the model, and check whether it is important. If not, delete it and repeat the process. Otherwise, stop. A 0.10 significance level is common to use for this strategy.

Epidemiologists use a slightly different approach to building models. They argue strongly for the need to always include **confounding** variables in a model, regardless of their statistical significance. I will briefly discuss this issue, but you should recognize that there is no universally accepted best approach to building models. A related issue is that several sets of predictors might give nearly identical fits and predictions to those obtained using any model selection method. This should not be too surprising because predictors are often correlated with each other. However, this should make us question whether one could ever completely unravel which variables are important (and which are not) for predicting a response.

3.2.1 Maximum likelihood and AIC/BIC

The **Akaike information criterion (AIC)** and **Bayesian information criterion (BIC)** are related penalized-likelihood criteria of the relative goodness-of-fit of a statistical model to the observed data. For model selection, a parsimonious model minimizes (one of) these quantities, where the penalty term is larger in BIC ($k\ln(n)$) than in AIC ($2k$). They are defined as

$$\begin{aligned} \text{AIC} &= -2\ln(L) + 2k \qquad \text{and} \\ \text{BIC} &= -2\ln(L) + k\ln(n) \end{aligned}$$

where n is the number of observations, k is the number of model parameters, and L is the maximized value of the likelihood function for the estimated model.

Maximum-likelihood estimation (MLE) applied to a data set and given a statistical model, estimates the model's parameters (βs and σ^2 in regression). MLE finds the particular parametric values that make the observed data the most probable given the model. That is, it selects the set of values of the model parameters that maximizes the likelihood function.

In practice, start with a set of candidate models, and then find the models' corresponding AIC/BIC values. There will almost always be information lost due to using one of the candidate models to represent the "true" (unknown) model. We choose the model that minimizes the (estimated) information loss (the Kullback-Leibler divergence of the "true" unknown model represented with a candidate model).

The penalty discourages overfitting. Increasing the number of free parameters in the model will always improve the goodness-of-fit, regardless of the number of free parameters in the data-generating process. In the spirit of Occam's razor, the principle of parsimony, economy, or succinctness, the penalty helps balance the complexity of the model (low) with its ability to describe the data (high).

There are many methods for model selection. AIC or BIC are good tools for helping to choose among candidate models. A model selected by BIC will tend to have fewer parameters than one selected by AIC. Ultimately, you have to choose a model. I think of automated model selection as a starting point among the models I ultimately consider, and I may decide upon a different model than AIC, BIC, or another method.

3.3 Example: Peru Indian blood pressure

I will illustrate backward elimination on the Peru Indian data, using systolic blood pressure (`sysbp`) as the response, and seven candidate predictors: `wt` = weight in kilos; `ht` = height in mm; `chin` = chin skin fold in mm; `fore` = forearm skin fold in mm; `calf` = calf skin fold in mm; `pulse` = pulse rate-beats/min, and `yrage` = fraction.

The program given below generates simple summary statistics and plots. The plots do not suggest any apparent transformations of the response or the predictors, so we will analyze the data using the given scales. The correlations between the response and each potential predictor indicate that predictors are generally not highly correlated with each other (a few are).

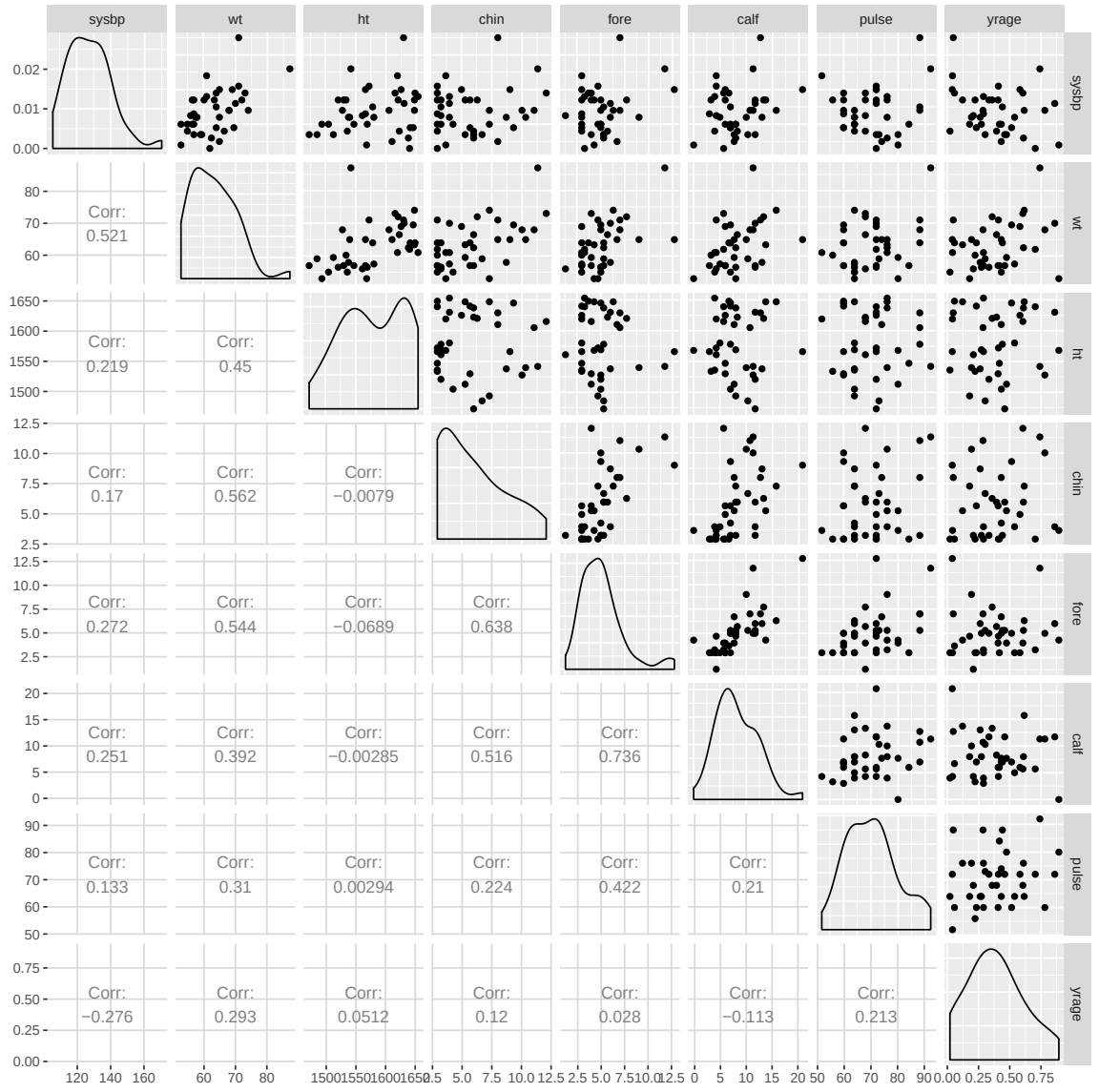
```
#### Example: Indian
# filename
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch02_indian.dat"
indian <- read.table(fn.data, header=TRUE)
```

```
# Create the "fraction of their life" variable
#   yrage = years since migration divided by age
indian$yrage <- indian$yrmig / indian$age

# subset of variables we want in our model
indian2 <- subset(indian, select=c("sysbp", "wt", "ht", "chin"
                                   , "fore", "calf", "pulse", "yrage")
                  )

str(indian2)
```

```
## 'data.frame': 39 obs. of  8 variables:
##  $ sysbp: int  170 120 125 148 140 106 120 108 124 134 ...
##  $ wt   : num  71 56.5 56 61 65 62 53 53 65 57 ...
##  $ ht   : int  1629 1569 1561 1619 1566 1639 1494 1568 1540 1530 ...
##  $ chin : num  8 3.3 3.3 3.7 9 3 7.3 3.7 10.3 5.7 ...
##  $ fore : num  7 5 1.3 3 12.7 3.3 4.7 4.3 9 4 ...
##  $ calf : num  12.7 8 4.3 4.3 20.7 5.7 8 0 10 6 ...
##  $ pulse: int  88 64 68 52 72 72 64 80 76 60 ...
##  $ yrage: num  0.0476 0.2727 0.2083 0.0417 0.04 ...
```

```
# Description of variables
#    id = individual id
#   age = age in years            yrmig = years since migration
#    wt = weight in kilos            ht = height in mm
#  chin = chin skin fold in mm     fore = forearm skin fold in mm
#  calf = calf skin fold in mm    pulse = pulse rate-beats/min
# sysbp = systolic bp             diabp = diastolic bp

## print dataset to screen
#indian2
```

```
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
#p <- ggpairs(indian2, progress=FALSE)
# put scatterplots on top so y axis is vertical
p <- ggpairs(indian2, upper = list(continuous = "points")
                    , lower = list(continuous = "cor")
                    , progress=FALSE
             )
print(p)

  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```




```
# correlation matrix and associated p-values testing "H0: rho == 0"
library(Hmisc)
rcorr(as.matrix(indian2))
```

```
##       sysbp   wt    ht  chin  fore  calf pulse yrage
## sysbp  1.00 0.52  0.22  0.17  0.27  0.25  0.13 -0.28
## wt     0.52 1.00  0.45  0.56  0.54  0.39  0.31  0.29
## ht     0.22 0.45  1.00 -0.01 -0.07  0.00  0.00  0.05
## chin   0.17 0.56 -0.01  1.00  0.64  0.52  0.22  0.12
## fore   0.27 0.54 -0.07  0.64  1.00  0.74  0.42  0.03
## calf   0.25 0.39  0.00  0.52  0.74  1.00  0.21 -0.11
## pulse  0.13 0.31  0.00  0.22  0.42  0.21  1.00  0.21
## yrage -0.28 0.29  0.05  0.12  0.03 -0.11  0.21  1.00
```

```
##
## n= 39
##
##
## P
##       sysbp  wt     ht     chin   fore   calf   pulse  yrage
## sysbp        0.0007 0.1802 0.3003 0.0936 0.1236 0.4211 0.0888
## wt    0.0007        0.0040 0.0002 0.0003 0.0136 0.0548 0.0702
## ht    0.1802 0.0040        0.9619 0.6767 0.9863 0.9858 0.7570
## chin  0.3003 0.0002 0.9619        0.0000 0.0008 0.1708 0.4665
## fore  0.0936 0.0003 0.6767 0.0000        0.0000 0.0075 0.8656
## calf  0.1236 0.0136 0.9863 0.0008 0.0000        0.1995 0.4933
## pulse 0.4211 0.0548 0.9858 0.1708 0.0075 0.1995        0.1928
## yrage 0.0888 0.0702 0.7570 0.4665 0.8656 0.4933 0.1928
```

Below I fit the linear model with all the selected main effects.

```
# fit full model
lm.indian2.full <- lm(sysbp ~ wt + ht + chin + fore + calf + pulse + yrage
                      , data = indian2)

library(car)
Anova(lm.indian2.full, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: sysbp
##              Sum Sq Df F value    Pr(>F)
## (Intercept)  389.46  1  3.8991 0.0572767 .
## wt          1956.49  1 19.5874 0.0001105 ***
## ht           131.88  1  1.3203 0.2593289
## chin         186.85  1  1.8706 0.1812390
## fore          27.00  1  0.2703 0.6068061
## calf           2.86  1  0.0287 0.8666427
## pulse         14.61  1  0.1463 0.7046990
## yrage       1386.76  1 13.8835 0.0007773 ***
## Residuals   3096.45 31
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm.indian2.full)
```

```
##
## Call:
## lm(formula = sysbp ~ wt + ht + chin + fore + calf + pulse + yrage,
##     data = indian2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.3993  -5.7916  -0.6907   6.9453  23.5771
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 106.45766   53.91303   1.975 0.057277 .
## wt            1.71095    0.38659   4.426 0.000111 ***
## ht           -0.04533    0.03945  -1.149 0.259329
## chin         -1.15725    0.84612  -1.368 0.181239
## fore         -0.70183    1.34986  -0.520 0.606806
## calf          0.10357    0.61170   0.169 0.866643
## pulse         0.07485    0.19570   0.383 0.704699
## yrage       -29.31810    7.86839  -3.726 0.000777 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.994 on 31 degrees of freedom
## Multiple R-squared:  0.5259,Adjusted R-squared:  0.4189
## F-statistic: 4.913 on 7 and 31 DF,  p-value: 0.0008079
```

Remarks on Step 0: The full model has 7 predictors so REG df = 7. The F-test in the full model ANOVA table ($F = 4.91$ with p-value = 0.0008) tests the hypothesis that the regression coefficient for each predictor variable is zero. This test is highly significant, indicating that one or more of the predictors is important in the model.

In the ANOVA table, the F-value column gives the square of the t-statistic (from the parameter [Coefficients] estimate table) for testing the significance of the individual predictors in the full model (conditional on all other predictors being in the model). The p-value is the same whether the t-statistic or F-value is shown.

The least important variable in the full model, as judged by the p-value, is calf skin fold. This variable, upon omission, reduces R^2 the least, or equivalently, increases the Residual SS the least. The p-value of 0.87 exceeds the default 0.10 cut-off, so $calf$ will be the first to be omitted from the model.

Below, we will continue in this way. After deleting `calf`, the six predictor model can be fitted. Manually, you can find that at least one of the predictors left is important, as judged by the overall F-test p-value. The least important predictor left is `pulse`. This variable is omitted from the model because the p-value for including it exceeds the 0.10 threshold.

This is repeated until all predictors remain significant at a 0.10 significance level.

```
# model reduction using update() and subtracting (removing) model terms
lm.indian2.red <- lm.indian2.full;

# remove calf
lm.indian2.red <- update(lm.indian2.red, ~ . - calf ); summary(lm.indian2.red);
##
## Call:
## lm(formula = sysbp ~ wt + ht + chin + fore + pulse + yrage, data = indian2)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.6993  -5.3152  -0.7725   7.2966  23.7240
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 106.13739   53.05581   2.000 0.053993 .
## wt            1.70900    0.38051   4.491 8.65e-05 ***
## ht           -0.04478    0.03871  -1.157 0.256008
## chin         -1.14165    0.82823  -1.378 0.177635
## fore         -0.56731    1.07462  -0.528 0.601197
## pulse         0.07103    0.19142   0.371 0.713018
## yrage       -29.54000    7.63983  -3.867 0.000509 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.841 on 32 degrees of freedom
## Multiple R-squared:  0.5255,Adjusted R-squared:  0.4365
## F-statistic: 5.906 on 6 and 32 DF,  p-value: 0.0003103
```

```
# remove pulse
lm.indian2.red <- update(lm.indian2.red, ~ . - pulse); summary(lm.indian2.red);
```

```
##
## Call:
## lm(formula = sysbp ~ wt + ht + chin + fore + yrage, data = indian2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.6147  -5.9803  -0.2065   6.6755  24.9269
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 110.27872   51.18665   2.154 0.038601 *
## wt            1.71825    0.37470   4.586 6.22e-05 ***
## ht           -0.04504    0.03820  -1.179 0.246810
## chin         -1.17716    0.81187  -1.450 0.156514
## fore         -0.43385    0.99933  -0.434 0.667013
## yrage       -28.98171    7.39172  -3.921 0.000421 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.712 on 33 degrees of freedom
## Multiple R-squared:  0.5234,Adjusted R-squared:  0.4512
## F-statistic: 7.249 on 5 and 33 DF,  p-value: 0.0001124
```

```
# remove fore
lm.indian2.red <- update(lm.indian2.red, ~ . - fore ); summary(lm.indian2.red);
```

```
##
## Call:
```

```
## lm(formula = sysbp ~ wt + ht + chin + yrage, data = indian2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.1030  -6.3484   0.2834   6.7766  24.8883
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 104.52292   48.84627   2.140 0.039629 *
## wt            1.64631    0.33203   4.958 1.94e-05 ***
## ht           -0.03957    0.03563  -1.111 0.274530
## chin         -1.31083    0.74220  -1.766 0.086348 .
## yrage       -28.32580    7.14879  -3.962 0.000361 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.595 on 34 degrees of freedom
## Multiple R-squared:  0.5207,Adjusted R-squared:  0.4643
## F-statistic: 9.235 on 4 and 34 DF,  p-value: 3.661e-05
```

```
# remove ht
lm.indian2.red <- update(lm.indian2.red, ~ . - ht   ); summary(lm.indian2.red);
```

```
##
## Call:
## lm(formula = sysbp ~ wt + chin + yrage, data = indian2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.6382  -6.6316   0.4521   6.3593  24.2086
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  52.9092    15.0895   3.506 0.001266 **
## wt            1.4407     0.2766   5.209 8.51e-06 ***
## chin         -1.0135     0.6945  -1.459 0.153407
## yrage       -27.3522     7.1185  -3.842 0.000491 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.627 on 35 degrees of freedom
## Multiple R-squared:  0.5033,Adjusted R-squared:  0.4608
## F-statistic: 11.82 on 3 and 35 DF,  p-value: 1.684e-05
```

```
# remove chin
lm.indian2.red <- update(lm.indian2.red, ~ . - chin ); summary(lm.indian2.red);
```

```
##
## Call:
## lm(formula = sysbp ~ wt + yrage, data = indian2)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4330  -7.3070   0.8963   5.7275  23.9819
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  60.8959    14.2809   4.264 0.000138 ***
## wt            1.2169     0.2337   5.207 7.97e-06 ***
## yrage       -26.7672     7.2178  -3.708 0.000699 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.777 on 36 degrees of freedom
## Multiple R-squared:  0.4731,Adjusted R-squared:  0.4438
## F-statistic: 16.16 on 2 and 36 DF,  p-value: 9.795e-06
# all are significant, stop.
# final model: sysbp ~ wt + yrage
lm.indian2.final <- lm.indian2.red
```

AIC/BIC automated model selection The AIC/BIC strategy is more commonly used for model selection, though resulting models are usually the same as the method described above. I use the `step()` function to perform backward selection using the AIC criterion (and give code for the BIC) then make some last-step decisions. Note that because the BIC has a larger penalty, it arrives at my chosen model directly.

At each step, the predictors are ranked (least significant to most significant) and then a decision of whether to keep the top predictor is made. `<none>` represents the current model.

```
## AIC
# option: test="F" includes additional information
#           for parameter estimate tests that we're familiar with
# option: for BIC, include k=log(nrow( [data.frame name] ))
lm.indian2.red.AIC <- step(lm.indian2.full, direction="backward", test="F")
## Start:  AIC=186.6
## sysbp ~ wt + ht + chin + fore + calf + pulse + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - calf   1      2.86 3099.3 184.64  0.0287 0.8666427
## - pulse  1     14.61 3111.1 184.79  0.1463 0.7046990
## - fore   1     27.00 3123.4 184.94  0.2703 0.6068061
## - ht     1    131.88 3228.3 186.23  1.3203 0.2593289
## <none>               3096.4 186.60
## - chin   1    186.85 3283.3 186.89  1.8706 0.1812390
## - yrage  1   1386.76 4483.2 199.04 13.8835 0.0007773 ***
## - wt     1   1956.49 5052.9 203.70 19.5874 0.0001105 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=184.64
## sysbp ~ wt + ht + chin + fore + pulse + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - pulse  1     13.34 3112.6 182.81  0.1377 0.7130185
## - fore   1     26.99 3126.3 182.98  0.2787 0.6011969
## - ht     1    129.56 3228.9 184.24  1.3377 0.2560083
## <none>               3099.3 184.64
## - chin   1    184.03 3283.3 184.89  1.9000 0.1776352
## - yrage  1   1448.00 4547.3 197.59 14.9504 0.0005087 ***
## - wt     1   1953.77 5053.1 201.70 20.1724 8.655e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=182.81
## sysbp ~ wt + ht + chin + fore + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - fore   1     17.78 3130.4 181.03  0.1885  0.667013
## - ht     1    131.12 3243.8 182.42  1.3902  0.246810
## <none>               3112.6 182.81
## - chin   1    198.30 3310.9 183.22  2.1023  0.156514
## - yrage  1   1450.02 4562.7 195.72 15.3730  0.000421 ***
## - wt     1   1983.51 5096.2 200.03 21.0290 6.219e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=181.03
## sysbp ~ wt + ht + chin + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - ht     1    113.57 3244.0 180.42  1.2334 0.2745301
## <none>               3130.4 181.03
## - chin   1    287.20 3417.6 182.45  3.1193 0.0863479 .
## - yrage  1   1445.52 4575.9 193.84 15.7000 0.0003607 ***
## - wt     1   2263.64 5394.1 200.25 24.5857 1.945e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=180.42
## sysbp ~ wt + chin + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>               3244.0 180.42
## - chin   1    197.37 3441.4 180.72  2.1295 0.1534065
## - yrage  1   1368.44 4612.4 192.15 14.7643 0.0004912 ***
```

```
## - wt     1   2515.33 5759.3 200.81 27.1384 8.512e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# BIC (not shown)
# step(lm.indian2.full, direction="backward", test="F", k=log(nrow(indian2)))
```

Remark on Summary Table: The partial R^2 is the reduction in R^2 achieved by omitting variables sequentially.

The backward elimination procedure eliminates five variables from the full model, in the following order: calf skin fold `calf`, pulse rate `pulse`, forearm skin fold `fore`, height `ht`, and chin skin fold `chin`. The model selected by backward elimination includes two predictors: weight `wt` and fraction `yrage`. As we progress from the full model to the selected model, R^2 decreases as follows: 0.53, 0.53, 0.52, 0.52, 0.50, and 0.47. The decrease is slight across this spectrum of models.

Using a mechanical approach, we are led to a model with weight and years by age fraction as predictors of systolic blood pressure. At this point we should closely examine this model.

3.3.1 Analysis for Selected Model

The summaries and diagnostics for the selected model follow.

Model sysbp $= \beta_0 + \beta_1$ wt $+ \beta_2$ yrage $+ \varepsilon$:

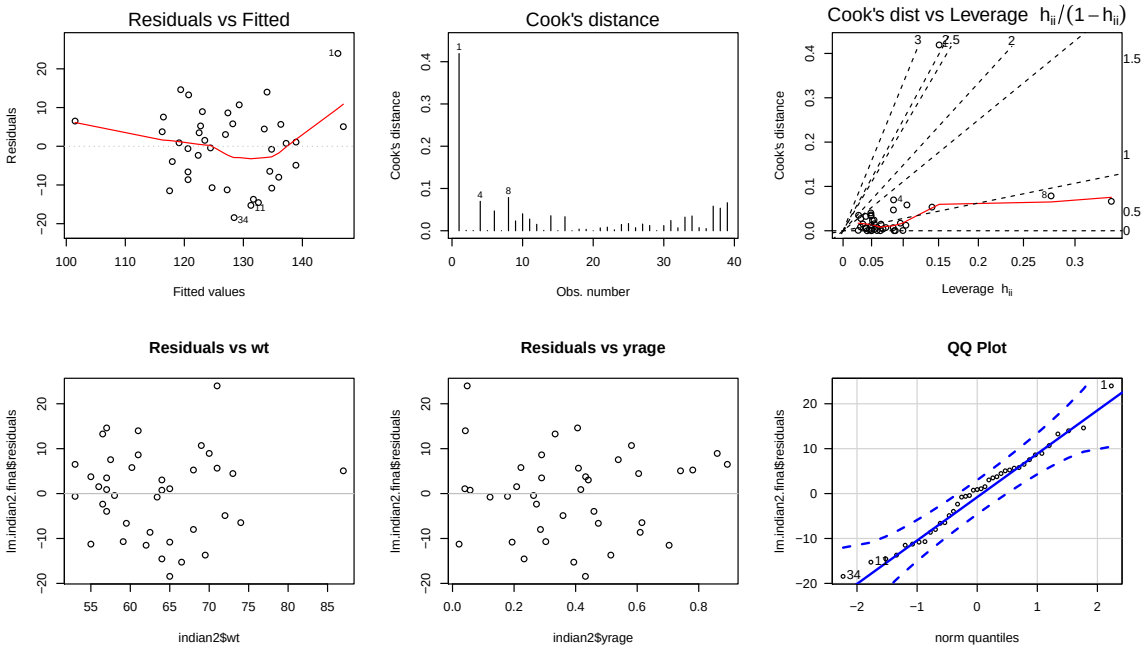
```
library(car)
Anova(lm.indian2.final, type=3)
## Anova Table (Type III tests)
##
## Response: sysbp
##             Sum Sq Df F value    Pr(>F)
## (Intercept) 1738.2  1  18.183 0.0001385 ***
## wt          2592.0  1  27.115 7.966e-06 ***
## yrage       1314.7  1  13.753 0.0006991 ***
## Residuals   3441.4 36
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.indian2.final)
##
## Call:
## lm(formula = sysbp ~ wt + yrage, data = indian2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4330  -7.3070   0.8963   5.7275  23.9819
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  60.8959    14.2809   4.264 0.000138 ***
## wt            1.2169     0.2337   5.207 7.97e-06 ***
## yrage       -26.7672     7.2178  -3.708 0.000699 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.777 on 36 degrees of freedom
## Multiple R-squared:  0.4731,Adjusted R-squared:  0.4438
## F-statistic: 16.16 on 2 and 36 DF,  p-value: 9.795e-06
```

Comments on the diagnostic plots below.

1. The individual with the highest systolic blood pressure (case 1) has a large studentized residual r_i and the largest Cook's D_i.
2. Except for case 1, the rankit plot and the plot of the studentized residuals against the fitted values show no gross abnormalities.
3. The plots of studentized residuals against the individual predictors show no patterns. The partial residual plots show roughly linear trends. These plots collectively do not suggest the need to transform either of the predictors. Although case 1 is prominent in the partial residual plots, it does not appear to be influencing the significance of these predictors.

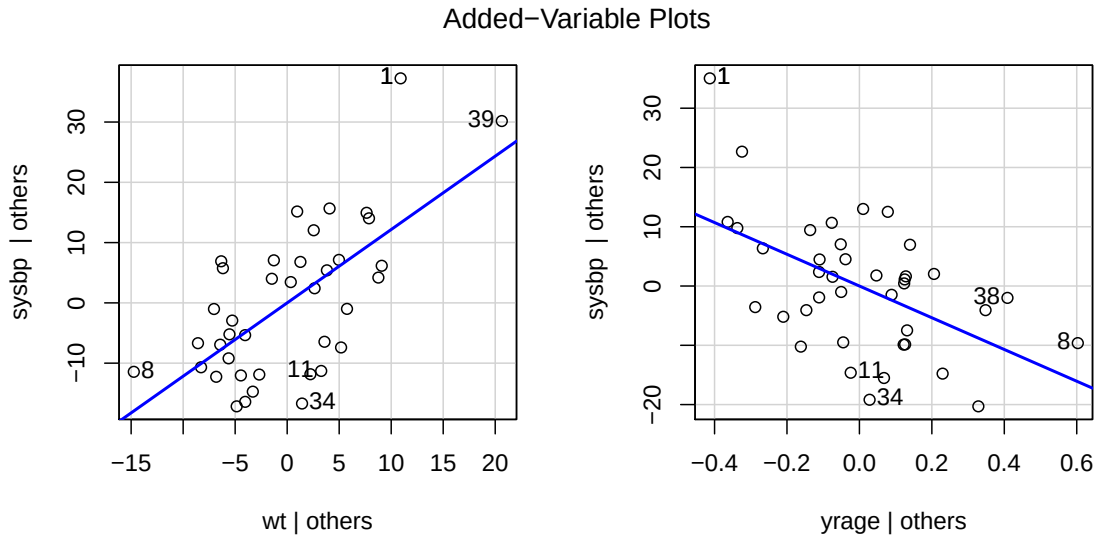
```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.indian2.final, which = c(1,4,6))

plot(indian2$wt, lm.indian2.final$residuals, main="Residuals vs wt")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(indian2$yrage, lm.indian2.final$residuals, main="Residuals vs yrage")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.indian2.final$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1]  1 34 11
## residuals vs order of data
#plot(lm.indian2.final$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Recall that the partial regression residual plot for weight, given below, **adjusts** systolic blood pressure and weight for their common dependence on all the other predictors in the model (only years by age fraction here). This plot tells us whether we need to transform weight in the multiple regression model, and whether any observations are influencing the significance of weight in the fitted model. A roughly linear trend, as seen here, suggests that no transformation of weight is warranted. The positive relationship seen here is consistent with the coefficient of weight being positive in the multiple regression model.

The partial residual plot for fraction exhibits a stronger relationship than is seen in the earlier 2D plot of systolic blood pressure against year by age fraction. This means that fraction is more useful as a predictor after taking an individual's weight into consideration.

```
library(car)
avPlots(lm.indian2.final, id = list(n = 3))
```



Model selection methods can be highly influenced by outliers and influential cases. We should hold out case 1, and rerun the backward procedure to see whether case 1 unduly influenced the selection of the two predictor model. If we hold out case 1, we find that the model with weight and fraction as predictors is suggested again. After holding out case 1, there are no large residuals, no extremely influential points, or any gross abnormalities in plots. The R^2 for the selected model is now $R^2 = 0.408$. This decrease in R^2 should have been anticipated. Why?[1]

The two analyses suggest that the "best model" for predicting systolic blood pressure is

$$\text{sysbp} \quad = \quad \beta_0 + \beta_1 \text{ wt} + \beta_2 \text{ yrage} + \varepsilon.$$

Should case 1 be deleted? I have not fully explored this issue, but I will note that eliminating this case does have a significant impact on the estimates of the regression coefficients, and on predicted values. What do you think?

3.4 Example: Dennis Cook's Rat Data

This example illustrates the importance of a careful diagnostic analysis.

An experiment was conducted to investigate the amount of a particular drug present in the liver of a rat. Nineteen (19) rats were randomly selected, weighed, placed under light ether anesthesia and given an oral dose of the drug. Because it was felt that large livers would absorb more of a given dose than small livers, the

[1]Obs 1 increases the SST, but greatly increases model relationship so greatly increases SSR.

actual dose an animal received was approximately determined as 40mg of the drug per kilogram of body weight. (Liver weight is known to be strongly related to body weight.) After a fixed length of time, each rat was sacrificed, the liver weighed, and the percent of the dose in the liver determined.

The experimental hypothesis was that, for the method of determining the dose, there is no relationship between the percentage of dose in the liver (Y) and the body weight, liver weight, and relative dose.

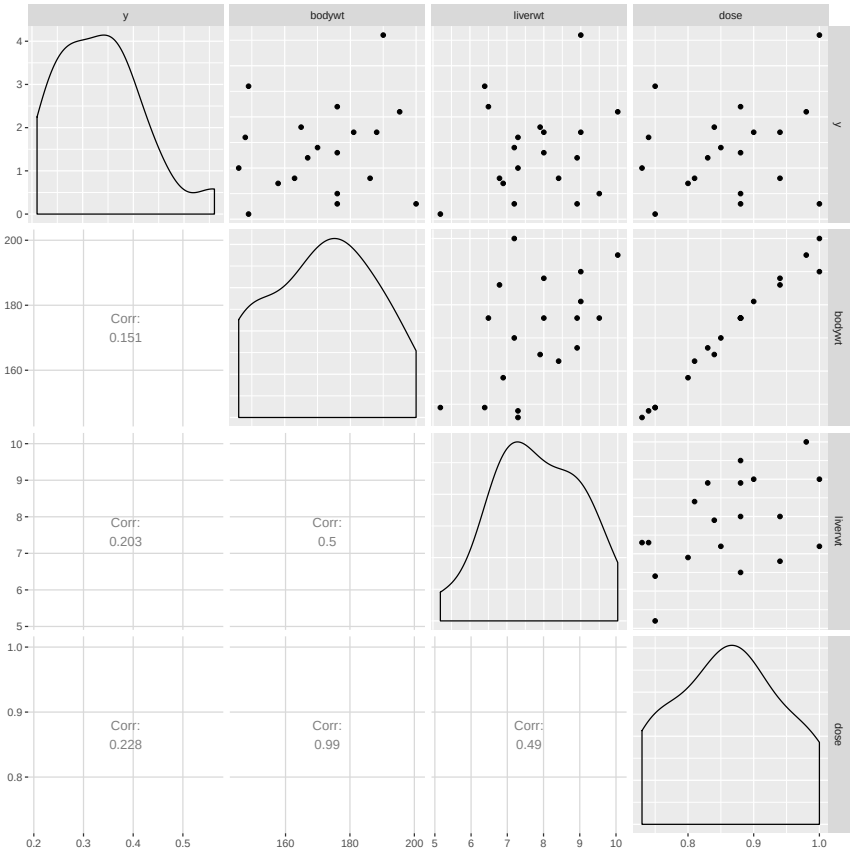
```
#### Example: Rat liver
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch03_ratliver.csv"
ratliver <- read.csv(fn.data)

ratliver <- ratliver[,c(4,1,2,3)] # reorder columns so response is the first

str(ratliver)
```

```
## 'data.frame': 19 obs. of  4 variables:
##  $ y      : num  0.42 0.25 0.56 0.23 0.23 0.32 0.37 0.41 0.33 0.38 ...
##  $ bodywt : int  176 176 190 176 200 167 188 195 176 165 ...
##  $ liverwt: num  6.5 9.5 9 8.9 7.2 8.9 8 10 8 7.9 ...
##  $ dose   : num  0.88 0.88 1 0.88 1 0.83 0.94 0.98 0.88 0.84 ...
```

	y	bodywt	liverwt	dose
1	0.42	176	6.50	0.88
2	0.25	176	9.50	0.88
3	0.56	190	9.00	1.00
4	0.23	176	8.90	0.88
5	0.23	200	7.20	1.00
6	0.32	167	8.90	0.83
7	0.37	188	8.00	0.94
8	0.41	195	10.00	0.98
9	0.33	176	8.00	0.88
10	0.38	165	7.90	0.84
11	0.27	158	6.90	0.80
12	0.36	148	7.30	0.74
13	0.21	149	5.20	0.75
14	0.28	163	8.40	0.81
15	0.34	170	7.20	0.85
16	0.28	186	6.80	0.94
17	0.30	146	7.30	0.73
18	0.37	181	9.00	0.90
19	0.46	149	6.40	0.75

```
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
#p <- ggpairs(ratliver, progress=FALSE)
# put scatterplots on top so y axis is vertical
p <- ggpairs(ratliver, upper = list(continuous = "points")
                     , lower = list(continuous = "cor")
                     , progress=FALSE
             )
print(p)
```

```
# detach package after use so reshape2 works (old reshape (v.1) conflicts)
#detach("package:GGally", unload=TRUE)
#detach("package:reshape", unload=TRUE)
```



The correlation between Y and each predictor is small, as expected.

```
# correlation matrix and associated p-values testing "H0: rho == 0"
library(Hmisc)
rcorr(as.matrix(ratliver))
```

```
##            y bodywt liverwt dose
## y       1.00   0.15    0.20 0.23
## bodywt  0.15   1.00    0.50 0.99
## liverwt 0.20   0.50    1.00 0.49
## dose    0.23   0.99    0.49 1.00
##
## n= 19
##
##
## P
##         y      bodywt liverwt dose
## y              0.5370 0.4038  0.3488
## bodywt  0.5370        0.0293  0.0000
```

```
## liverwt 0.4038 0.0293         0.0332
## dose    0.3488 0.0000 0.0332
```

Below I fit the linear model with all the selected main effects.

```
# fit full model
lm.ratliver.full <- lm(y ~ bodywt + liverwt + dose, data = ratliver)

library(car)
Anova(lm.ratliver.full, type=3)
## Anova Table (Type III tests)
##
## Response: y
##               Sum Sq Df F value  Pr(>F)
## (Intercept) 0.011157  1  1.8676 0.19188
## bodywt      0.042408  1  7.0988 0.01768 *
## liverwt     0.004120  1  0.6897 0.41930
## dose        0.044982  1  7.5296 0.01507 *
## Residuals   0.089609 15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.ratliver.full)
##
## Call:
## lm(formula = y ~ bodywt + liverwt + dose, data = ratliver)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.100557 -0.063233  0.007131  0.045971  0.134691
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.265922   0.194585   1.367   0.1919
## bodywt      -0.021246   0.007974  -2.664   0.0177 *
## liverwt      0.014298   0.017217   0.830   0.4193
## dose         4.178111   1.522625   2.744   0.0151 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07729 on 15 degrees of freedom
## Multiple R-squared:  0.3639,Adjusted R-squared:  0.2367
## F-statistic:  2.86 on 3 and 15 DF,  p-value: 0.07197
```

The backward elimination procedure selects weight and dose as predictors. The p-values for testing the importance of these variables, when added last to this two predictor model, are small, 0.019 and 0.015.

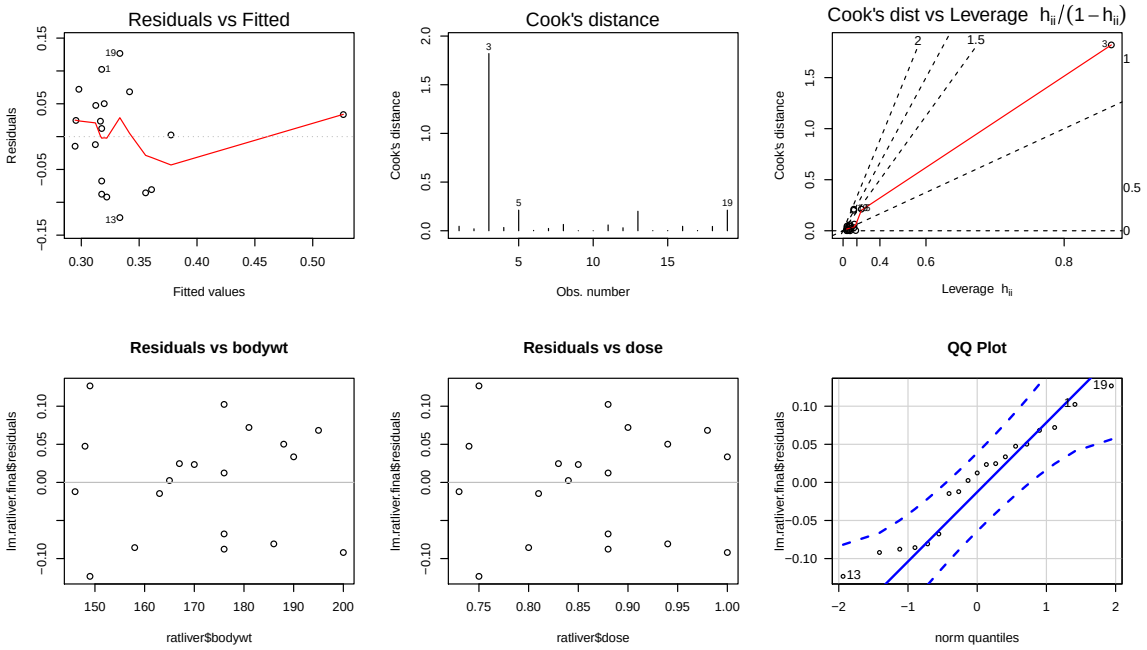
```
lm.ratliver.red.AIC <- step(lm.ratliver.full, direction="backward", test="F")
## Start:  AIC=-93.78
## y ~ bodywt + liverwt + dose
```

```
##
##           Df Sum of Sq      RSS     AIC F value  Pr(>F)
## - liverwt  1  0.004120 0.093729 -94.924  0.6897 0.41930
## <none>                 0.089609 -93.778
## - bodywt   1  0.042408 0.132017 -88.416  7.0988 0.01768 *
## - dose     1  0.044982 0.134591 -88.049  7.5296 0.01507 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=-94.92
## y ~ bodywt + dose
##
##          Df Sum of Sq      RSS     AIC F value  Pr(>F)
## <none>                0.093729 -94.924
## - bodywt  1  0.039851 0.133580 -90.192  6.8027 0.01902 *
## - dose    1  0.043929 0.137658 -89.621  7.4989 0.01458 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
lm.ratliver.final <- lm.ratliver.red.AIC
```

This cursory analysis leads to a conclusion that a combination of dose and body weight is associated with Y, but that neither of these predictors is important of its own (low correlations with Y). Although this commonly happens in regression problems, it is somewhat paradoxical here because dose was approximately a multiple of body weight, so to a first approximation, these predictors are linearly related and so only one of them should be needed in a linear regression model. Note that the correlation between dose and body weight is 0.99.

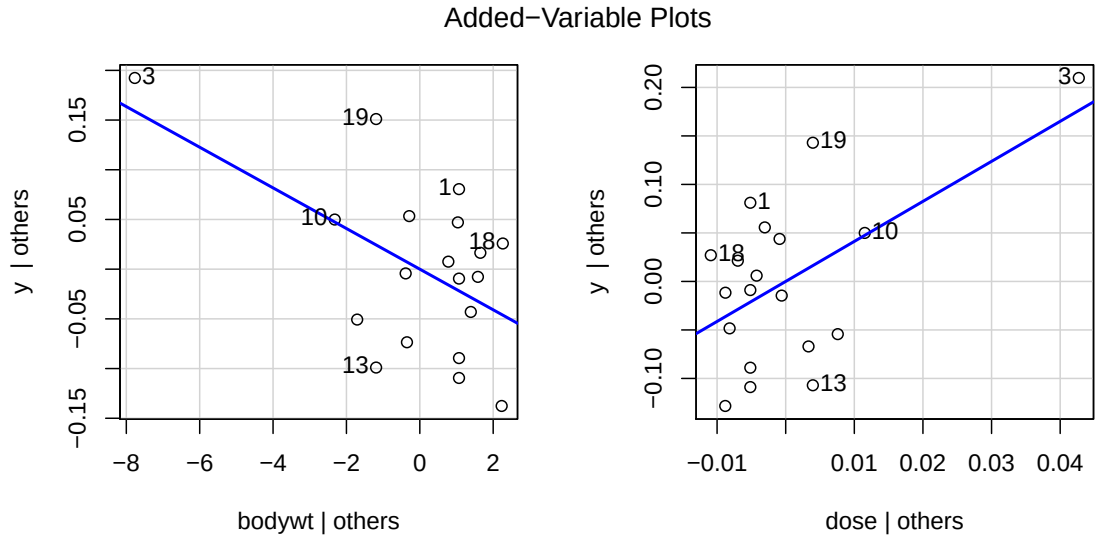
The apparent paradox can be resolved only with a careful diagnostic analysis! For the model with dose and body weight as predictors, there are no cases with large $|r_i|$ values, but case 3 has a relatively large Cook's D value.

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.ratliver.final, which = c(1,4,6))

plot(ratliver$bodywt, lm.ratliver.final$residuals, main="Residuals vs bodywt")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(ratliver$dose, lm.ratliver.final$residuals, main="Residuals vs dose")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.ratliver.final$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1] 19 13  1

## residuals vs order of data
#plot(lm.ratliver.final$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Further, the partial residual plot for `bodywt` clearly highlights case 3. Without this case we would see roughly a random scatter of points, suggesting that body weight is unimportant after taking dose into consideration. The importance of body weight as a predictor in the multiple regression model is due solely to the placement of case 3. The partial residual plot for `dose` gives the same message.

```
library(car)
avPlots(lm.ratliver.final, id = list(n = 3))
```



Removing case 3 If we delete this case and redo the analysis we find, as expected, no important predictors of Y. The output below shows that the backward elimination removes each predictor from the model. Thus, the apparent relationship between Y and body weight and dose in the initial analysis can be ascribed to Case 3 alone. Can you see this case in the plots?

```
# remove case 3
ratliver3 <- ratliver[-3,]

# fit full model
lm.ratliver3.full <- lm(y ~ bodywt + liverwt + dose, data = ratliver3)

lm.ratliver3.red.AIC <- step(lm.ratliver3.full, direction="backward", test="F")
## Start:  AIC=-88.25
## y ~ bodywt + liverwt + dose
##
##           Df  Sum of Sq      RSS     AIC F value Pr(>F)
## - dose     1 0.00097916 0.086696 -90.043  0.1599 0.6953
## - bodywt   1 0.00105871 0.086776 -90.026  0.1729 0.6838
## - liverwt  1 0.00142114 0.087138 -89.951  0.2321 0.6374
## <none>                 0.085717 -88.247
##
## Step:  AIC=-90.04
## y ~ bodywt + liverwt
##
##           Df  Sum of Sq      RSS     AIC F value Pr(>F)
## - bodywt   1 0.00035562 0.087052 -91.969  0.0615 0.8075
## - liverwt  1 0.00082681 0.087523 -91.872  0.1431 0.7106
## <none>                 0.086696 -90.043
##
## Step:  AIC=-91.97
## y ~ liverwt
##
##           Df  Sum of Sq      RSS     AIC F value Pr(>F)
## - liverwt  1 0.00050917 0.087561 -93.864  0.0936 0.7636
## <none>                 0.087052 -91.969
##
## Step:  AIC=-93.86
## y ~ 1
```

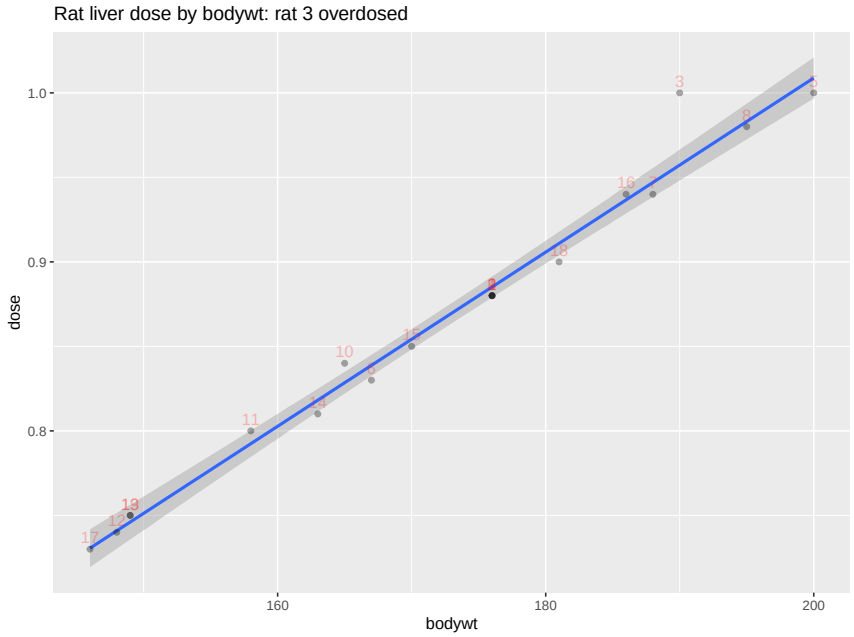
All variables are omitted!

In his text[2], Weisberg writes: The careful analyst must now try to understand exactly why the third case is so influential. Inspection of the data indicates that this rat with weight 190g, was reported to have received a full dose of 1.000, which was a larger dose than it should have received according to the rule for assigning doses, see

[2]Applied Linear Regression, 3rd Ed. by Sanford Weisberg, published by Wiley/Interscience in 2005 (ISBN 0-471-66379-4)

scatterplot below (e.g., rat 8 with a weight of 195g got a lower dose of 0.98).

```
# ggplot: Plot the data with linear regression fit and confidence bands
library(ggplot2)
p <- ggplot(ratliver, aes(x = bodywt, y = dose, label = 1:nrow(ratliver)))
# plot regression line and confidence band
p <- p + geom_smooth(method = lm)
p <- p + geom_point(alpha=1/3)
# plot labels next to points
p <- p + geom_text(hjust = 0.5, vjust = -0.5, alpha = 0.25, colour = 2)
p <- p + labs(title="Rat liver dose by bodywt: rat 3 overdosed")
print(p)
```



A number of causes for the result found in the first analysis are possible: (1) the dose or weight recorded for case 3 was in error, so the case should probably be deleted from the analysis, or (2) the regression fit in the second analysis is not appropriate except in the region defined by the 18 points excluding case 3. It is possible that the combination of dose and rat weight chosen was fortuitous, and that the lack of relationship found would not persist for any other combinations of them, since inclusion of a data point apparently taken under different conditions leads to a different conclusion. This suggests the need for collection of additional data, with dose determined by some rule other than a constant proportion of weight.

I hope the point of this analysis is clear! What have we learned from this analysis?

Part VII

ADA2: Experimental design and observational studies

Chapter 4

One Factor Designs and Extensions

This section describes an experimental design to compare the effectiveness of four insecticides to eradicate beetles. The primary interest is determining which treatment is most effective, in the sense of providing the lowest typical survival time.

In a **completely randomized design** (CRD), the scientist might select a sample of genetically identical beetles for the experiment, and then randomly assign a predetermined number of beetles to the treatment groups (insecticides). The sample sizes for the groups need not be equal. A power analysis is often conducted to determine sample sizes for the treatments. For simplicity, assume that 48 beetles will be used in the experiment, with 12 beetles assigned to each group.

After assigning the beetles to the four groups, the insecticide is applied (uniformly to all experimental units or beetles), and the individual survival times recorded. A natural analysis of the data collected from this **one factor** design would be to compare the survival times using a one-way ANOVA.

There are several important controls that should be built into this experiment. The same strain of beetles should be used to ensure that the four treatment groups are alike as possible, so that differences in survival times are attributable to the insecticides, and not due to genetic differences among beetles. Other factors that may influence the survival time, say the concentration of the insecticide or the age of the beetles, would be held constant, or fixed by the experimenter, if possible. Thus, the same concentration would be used with the four insecticides.

In complex experiments, there are always potential influences that are not realized or are thought to be unimportant that you do not or can not control. The **randomization** of beetles to groups ensures that there is no systematic dependence of the observed treatment differences on the uncontrolled influences. This is extremely important in studies where genetic and environmental influences can not be easily controlled (as in humans, more so than in bugs or mice). The randomization of beetles to insecticides tends to diffuse or greatly reduce the effect of the uncontrolled influences on the comparison of insecticides, in the sense that these effects become

part of the uncontrolled or error variation of the experiment.

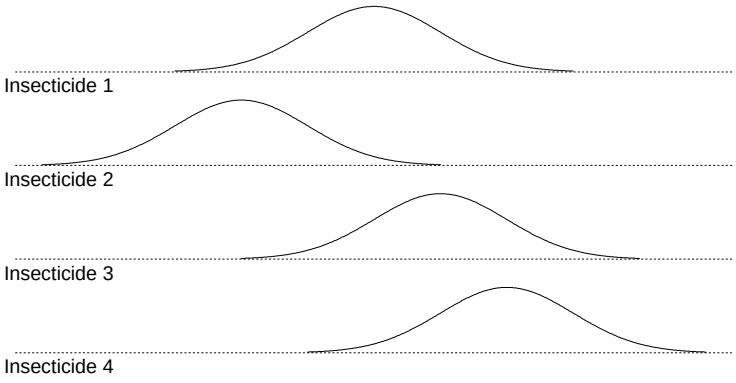
In summary, an **experiment** is to impose a treatment on experimental units to observe a response. Randomization and carefully controlling factors are important considerations.

Suppose y_{ij} is the response for the j^{th} experimental unit in the i^{th} treatment group, where $i = 1, 2, \ldots, I$. The statistical model for a **completely randomized one-factor design** that leads to a one-way ANOVA is given by:

$$y_{ij} = \mu_i + e_{ij},$$

where μ_i is the (unknown) population mean for all potential responses to the i^{th} treatment, and e_{ij} is the residual or deviation of the response from the population mean. The responses within and across treatments are assumed to be independent, normal random variables with constant variance.

For the insecticide experiment, y_{ij} is the survival time for the j^{th} beetle given the i^{th} insecticide, where $i = 1, 2, 3, 4$ and $j = 1, 2, \ldots, 12$. The random selection of beetles coupled with the randomization of beetles to groups ensures the independence assumptions. The assumed population distributions of responses for the $I = 4$ insecticides can be represented as follows:



Let

$$\mu = \frac{1}{I}\sum_i \mu_i$$

be the grand mean, or average of the population means. Let

$$\alpha_i = \mu_i - \mu$$

be the i^{th} group **treatment effect**. The treatment effects are constrained to add to zero, $\alpha_1 + \alpha_2 + \cdots + \alpha_I = 0$, and measure the difference between the treatment population means and the grand mean. Given this notation, the one-way ANOVA model is

$$y_{ij} = \mu + \alpha_i + e_{ij}.$$

The model specifies that the

Response = Grand Mean + Treatment Effect + Residual.

An hypothesis of interest is whether the population means are equal: $H_0 : \mu_1 = \cdots = \mu_I$, which is equivalent to the hypothesis of no treatment effects: $H_0 : \alpha_1 = \cdots = \alpha_I = 0$. If H_0 is true, then the one-way model is

$$y_{ij} = \mu + e_{ij},$$

where μ is the common population mean. You know how to test H_0 and do multiple comparisons of the treatments, so I will not review this material.

Most texts use treatment effects to specify ANOVA models, a convention that I will also follow. A difficulty with this approach is that the treatment effects must be constrained to be uniquely estimable from the data (because the I population means μ_i are modeled in terms of $I + 1$ parameters: $\mu_i = \mu + \alpha_i$). An infinite number of constraints can be considered each of which gives the same structure on the population means. The standard constraint where the treatment effects sum to zero was used above, but many statistical packages, impose the constraint $\alpha_I = 0$ (or sometimes $\alpha_1 = 0$). Although estimates of treatment effects depend on which constraint is chosen, the null and alternative models used with the ANOVA F-test, and pairwise comparisons of treatment effects, do not. I will downplay the discussion of estimating treatment effects to minimize problems.

Chapter 5

Paired Experiments and Randomized Block Experiments

A **randomized block design** is often used instead of a completely randomized design in studies where there is extraneous variation among the experimental units that may influence the response. A significant amount of the extraneous variation may be removed from the comparison of treatments by partitioning the experimental units into fairly **homogeneous subgroups** or **blocks**.

For example, suppose you are interested in comparing the effectiveness of four antibiotics for a bacterial infection. The recovery time after administering an antibiotic may be influenced by a patient's general health, the extent of their infection, or their age. Randomly allocating experimental subjects to the treatments (and then comparing them using a one-way ANOVA) may produce one treatment having a "favorable" sample of patients with features that naturally lead to a speedy recovery. Alternatively, if the characteristics that affect the recovery time are spread across treatments, then the variation within samples due to these uncontrolled features can dominate the effects of the treatment, leading to an inconclusive result.

A better way to design this experiment would be to **block** the subjects into groups of four patients who are alike as possible on factors other than the treatment that influence the recovery time. The four treatments are then randomly assigned to the patients (one per patient) within a block, and the recovery time measured. The blocking of patients usually produces a more sensitive comparison of treatments than does a completely randomized design because the variation in recovery times due to the blocks is eliminated from the comparison of treatments.

A randomized block design is a **paired experiment** when two treatments are compared. The usual analysis for a paired experiment is a parametric or nonparametric paired comparison.

Randomized block (RB) designs were developed to account for soil fertility gra-

dients in agricultural experiments. The experimental field would be separated into strips (blocks) of fairly constant fertility. Each strip is partitioned into equal size plots. The treatments, say varieties of corn, are randomly assigned to the plots, with each treatment occurring the same number of times (usually once) per block. All other factors that are known to influence the response would be controlled or fixed by the experimenter. For example, when comparing the mean yields, each plot would receive the same type and amount of fertilizer and the same irrigation plan.

The discussion will be limited to randomized block experiments with one factor. Two or more factors can be used with a randomized block design. For example, the agricultural experiment could be modified to compare four combinations of two corn varieties and two levels of fertilizer in each block instead of the original four varieties. In certain experiments, each experimental unit receives each treatment. The experimental units are "natural" blocks for the analysis.

Example: Comparison of Treatments for Itching Ten[1] male volunteers between 20 and 30 years old were used as a study group to compare seven treatments (5 drugs, a placebo, and no drug) to relieve itching. Each subject was given a different treatment on seven study days. The time ordering of the treatments was randomized across days. Except on the no-drug day, the subjects were given the treatment intravenously, and then itching was induced on their forearms using an effective itch stimulus called cowage. The subjects recorded the duration of itching, in seconds. The data are given in the table below. From left to right the drugs are: papaverine, morphine, aminophylline, pentobarbitol, tripelenamine.

The volunteers in the study were treated as blocks in the analysis. At best, the volunteers might be considered a representative sample of males between the ages of 20 and 30. This limits the extent of inferences from the experiment. The scientists can not, without sound medical justification, extrapolate the results to children or to senior citizens.

```
#### Example: Itching
itch <- read.csv("http://statacumen.com/teach/ADA2/ADA2_notes_Ch05_itch.csv")
```

[1] Beecher, 1959

	Patient	Nodrug	Placebo	Papv	Morp	Amino	Pento	Tripel
1	1	174	263	105	199	141	108	141
2	2	224	213	103	143	168	341	184
3	3	260	231	145	113	78	159	125
4	4	255	291	103	225	164	135	227
5	5	165	168	144	176	127	239	194
6	6	237	121	94	144	114	136	155
7	7	191	137	35	87	96	140	121
8	8	100	102	133	120	222	134	129
9	9	115	89	83	100	165	185	79
10	10	189	433	237	173	168	188	317

5.1 Analysis of a Randomized Block Design

Assume that you designed a randomized block experiment with I blocks and J treatments, where each treatment occurs once in each block. Let y_{ij} be the response for the j^{th} treatment within the i^{th} block. The model for the experiment is

$$y_{ij} = \mu_{ij} + e_{ij},$$

where μ_{ij} is the population mean response for the j^{th} treatment in the i^{th} block and e_{ij} is the deviation of the response from the mean. The population means are assumed to satisfy the additive model

$$\mu_{ij} = \mu + \alpha_i + \beta_j$$

where μ is a grand mean, α_i is the effect for the i^{th} block, and β_j is the effect for the j^{th} treatment. The responses are assumed to be independent across blocks, normally distributed and with constant variance. The randomized block model does not require the observations within a block to be independent, but does assume that the correlation between responses within a block is identical for each pair of treatments.

The model is sometimes written as

Response = Grand Mean + Treatment Effect + Block Effect + Residual.

Given the data, let $\bar{y}_{i\cdot}$ be the i^{th} block sample mean (the average of the responses in the i^{th} block), $\bar{y}_{\cdot j}$ be the j^{th} treatment sample mean (the average of the responses on the j^{th} treatment), and $\bar{y}_{\cdot\cdot}$ be the average response of all IJ observations in the experiment.

An ANOVA table for the randomized block experiment partitions the Model SS into SS for Blocks and Treatments.

Source	*df*	SS	MS
Blocks	$I-1$	$J\sum_i(\bar{y}_{i\cdot}-\bar{y}_{\cdot\cdot})^2$	
Treats	$J-1$	$I\sum_j(\bar{y}_{\cdot j}-\bar{y}_{\cdot\cdot})^2$	
Error	$(I-1)(J-1)$	$\sum_{ij}(y_{ij}-\bar{y}_{i\cdot}-\bar{y}_{\cdot j}+\bar{y}_{\cdot\cdot})^2$	
Total	$IJ-1$	$\sum_{ij}(y_{ij}-\bar{y}_{\cdot\cdot})^2$.	

A primary interest is testing whether the treatment effects are zero: $H_0 : \beta_1 = \cdots = \beta_J = 0$. The treatment effects are zero if in each block the population mean responses are identical for each treatment. A formal test of no treatment effects is based on the p-value from the F-statistic F_{obs} = MS Treat/MS Error. The p-value is evaluated in the usual way (i.e., as an upper tail area from an F-distribution with $J-1$ and $(I-1)(J-1)$ df.) This H_0 is rejected when the treatment averages $\bar{y}_{\cdot j}$ vary significantly relative to the error variation.

A test for no block effects ($H_0 : \alpha_1 = \cdots = \alpha_I = 0$) is often a secondary interest, because, if the experiment is designed well, the blocks will be, by construction, noticeably different. There are no block effects if the population mean response for an arbitrary treatment is identical across blocks. A formal test of no block effects is based on the p-value from the the F-statistic F_{obs} = MS Blocks/MS Error. This H_0 is rejected when the block averages $\bar{y}_{i\cdot}$ vary significantly relative to the error variation.

The randomized block model is easily fitted in the `lm()` function. Before illustrating the analysis on the itching data, let me mention five important points about randomized block analyses:

1. The F-test p-value for comparing $J = 2$ treatments is identical to the p-value for comparing the two treatments using a paired t-test.
2. The Block SS plus the Error SS is the Error SS from a one-way ANOVA comparing the J treatments. If the Block SS is large relative to the Error SS from the two-factor model, then the experimenter has eliminated a substantial portion of the variation that is used to assess the differences among the treatments. This leads to a more sensitive comparison of treatments than would have been obtained using a one-way ANOVA.
3. The RB model is equivalent to an additive or no interaction model for a two-factor experiment, where the blocks are levels of one of the factors. The analysis of a randomized block experiment under this model is the same analysis used for a two-factor experiment with no replication (one observation per cell). We will discuss the two-factor design soon.
4. Under the sum constraint on the parameters (i.e., $\sum_i \alpha_i = \sum_j \beta_j = 0$), the estimates of the grand mean, block effects, and treatment effects are $\hat{\mu} = \bar{y}_{\cdot\cdot}$, $\hat{\alpha}_i = \bar{y}_{i\cdot} - \bar{y}_{\cdot\cdot}$, and $\hat{\beta}_j = \bar{y}_{\cdot j} - \bar{y}_{\cdot\cdot}$, respectively. The estimated mean response for the $(i,j)^{th}$ cell is $\hat{\mu}_{ij} = \hat{\mu} + \hat{\alpha}_i + \hat{\beta}_j = \bar{y}_{i\cdot} + \bar{y}_{\cdot j} - \bar{y}_{\cdot\cdot}$.
5. The F-test for comparing treatments is appropriate when the responses within a block have the same correlation. This is a reasonable working assumption

in many analyses. A multivariate repeated measures model can be used to compare treatments when the constant correlation assumption is unrealistic, for example when the same treatment is given to an individual over time.

RB Analysis of the Itching Data First we reshape the data to long format so each observation is its own row in the `data.frame` and indexed by the `Patient` and `Treatment` variables.

```
library(reshape2)
itch.long <- melt(itch
                  , id.vars       = "Patient"
                  , variable.name = "Treatment"
                  , value.name    = "Seconds"
                )
str(itch.long)
```

```
## 'data.frame': 70 obs. of  3 variables:
##  $ Patient  : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Treatment: Factor w/ 7 levels "Nodrug","Placebo",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Seconds  : int  174 224 260 255 165 237 191 100 115 189 ...
```

```
head(itch.long, 3)
```

```
##   Patient Treatment Seconds
## 1       1    Nodrug     174
## 2       2    Nodrug     224
## 3       3    Nodrug     260
```

```
tail(itch.long, 3)
```

```
##    Patient Treatment Seconds
## 68       8    Tripel     129
## 69       9    Tripel      79
## 70      10    Tripel     317
```

```
# make Patient a factor variable
itch.long$Patient <- factor(itch.long$Patient)
str(itch.long)
```

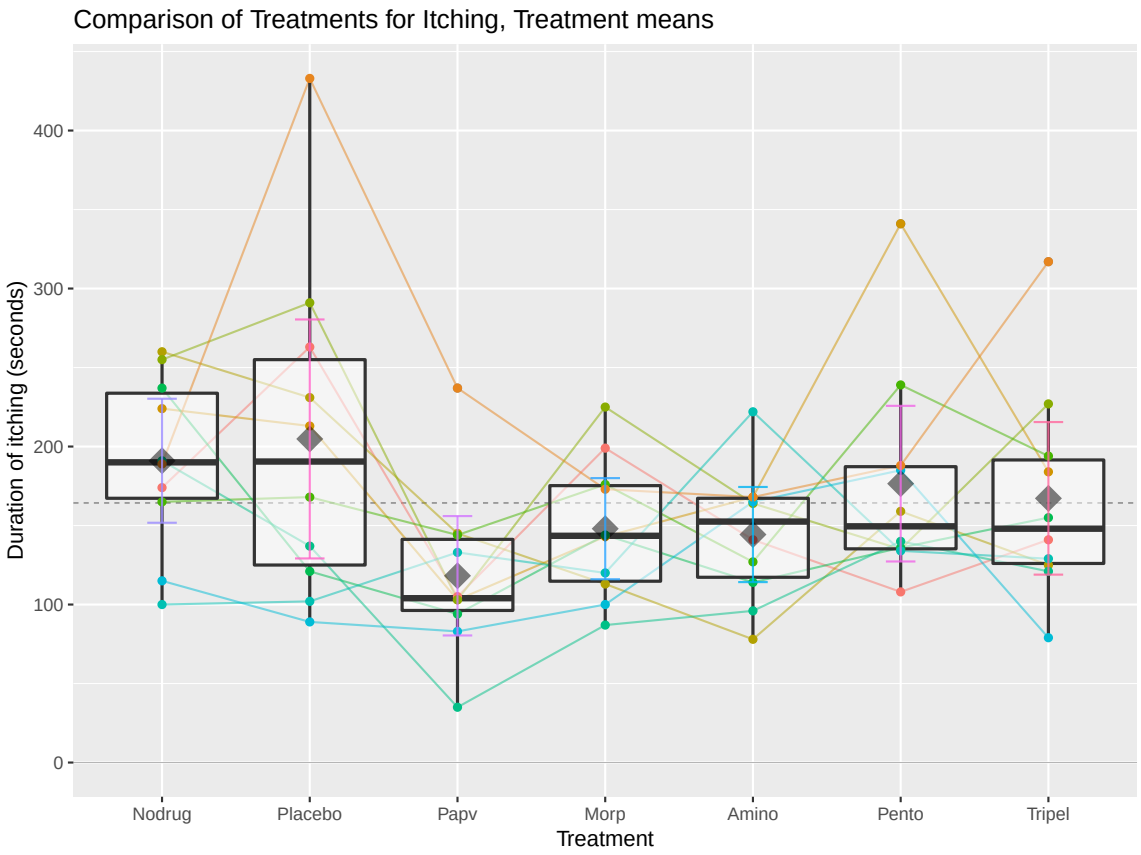
```
## 'data.frame': 70 obs. of  3 variables:
##  $ Patient  : Factor w/ 10 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Treatment: Factor w/ 7 levels "Nodrug","Placebo",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Seconds  : int  174 224 260 255 165 237 191 100 115 189 ...
```

As a first step, I made side-by-side boxplots of the itching durations across treatments. The boxplots are helpful for informally comparing treatments and visualizing the data. The differences in the level of the boxplots will usually be magnified by the F-test for comparing treatments because the variability within the boxplots includes block differences which are moved from the Error SS to the Block SS. The plot also includes the 10 Patients with lines connecting their measurements to see how common the treatment differences were over patients. I admit, this plot is a little too busy.

Each of the five drugs appears to have an effect, compared to the placebo and to no drug. Papaverine appears to be the most effective drug. The placebo and no

drug have similar medians. The relatively large spread in the placebo group suggests that some patients responded adversely to the placebo compared to no drug, whereas others responded positively.

```
# Plot the data using ggplot
library(ggplot2)
p <- ggplot(itch.long, aes(x = Treatment, y = Seconds))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(aes(yintercept = 0),
                    colour = "black", linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_hline(aes(yintercept = mean(Seconds)),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# colored line for each patient
p <- p + geom_line(aes(group = Patient, colour = Patient), 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(aes(colour = Patient))
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      alpha = 0.5)
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, aes(colour=Treatment), alpha = 0.8)
p <- p + labs(title = "Comparison of Treatments for Itching, Treatment means")
p <- p + ylab("Duration of itching (seconds)")
# removes legend
p <- p + theme(legend.position="none")
print(p)
```



To fit the RB model in `lm()`, you need to specify blocks (`Patient`) and treatments (`Treatment`) as **factor** variables, and include each to the right of the tilde symbol in the **formula** statement. The response variable **Seconds** appears to the left of the tilde.

```
lm.s.t.p <- lm(Seconds ~ Treatment + Patient, data = itch.long)
library(car)
Anova(lm.s.t.p, type=3)
## Anova Table (Type III tests)
##
## Response: Seconds
##             Sum Sq Df F value    Pr(>F)
## (Intercept) 155100  1 50.1133 3.065e-09 ***
## Treatment    53013  6  2.8548  0.017303 *
## Patient     103280  9  3.7078  0.001124 **
## Residuals   167130 54
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.s.t.p)
##
## Call:
```

```
## lm(formula = Seconds ~ Treatment + Patient, data = itch.long)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -81.286 -34.800  -8.393  30.900 148.914
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       188.286     26.598   7.079 3.07e-09 ***
## TreatmentPlacebo   13.800     24.880   0.555  0.58141
## TreatmentPapv     -72.800     24.880  -2.926  0.00501 **
## TreatmentMorp     -43.000     24.880  -1.728  0.08965 .
## TreatmentAmino    -46.700     24.880  -1.877  0.06592 .
## TreatmentPento    -14.500     24.880  -0.583  0.56245
## TreatmentTripel   -23.800     24.880  -0.957  0.34303
## Patient2           35.000     29.737   1.177  0.24436
## Patient3           -2.857     29.737  -0.096  0.92381
## Patient4           38.429     29.737   1.292  0.20176
## Patient5           11.714     29.737   0.394  0.69518
## Patient6          -18.571     29.737  -0.625  0.53491
## Patient7          -46.286     29.737  -1.557  0.12543
## Patient8          -27.286     29.737  -0.918  0.36292
## Patient9          -45.000     29.737  -1.513  0.13604
## Patient10          82.000     29.737   2.758  0.00793 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 55.63 on 54 degrees of freedom
## Multiple R-squared:  0.4832,Adjusted R-squared:  0.3397
## F-statistic: 3.367 on 15 and 54 DF,  p-value: 0.00052
```

The order to look at output follows the hierarchy of multi-parameter tests down to single-parameter tests.

1. The F-test at the bottom of the `summary()` tests for both no block effects and no treatment effects. If there are no block effects and no tretment effects then the mean itching time is independent of treatment and patients. The p-value of 0.0005 strongly suggests that the population mean itching times are not all equal.
2. The ANOVA table at top from `Anova()` partitions the Model SS into the SS for Blocks (Patients) and Treatments. The Mean Squares, F-statistics, and p-values for testing these effects are given. For a RB design with the same number of responses per block (i.e., no missing data), the Type I and Type III SS are identical, and correspond to the formulas given earlier. The distinction between Type I and Type III SS is important for unbalanced problems, an issue we discuss later. The F-tests show significant differences among the treatments (p-value=0.017) and among patients (p-value=0.001).

3. The individual parameter (coefficient) estimates in the `summary()` are likely of less interest since they test differences from the baseline group, only. The multiple comparisons in the next section will indicate which factor levels are different from others.

Multiple comparisons Multiple comparison and contrasts are not typically straightforward in R, though some newer packages are helping make them easier. Below I show one way that I think is relatively easy.

The package `multcomp` is used to specify which factor to perform multiple comparisons over and which p-value adjustment method to use. Below I use Tukey adjustments, first.

```
# multcomp has functions for multiple comparisons
library(multcomp)
## Loading required package:  mvtnorm
## Loading required package:  TH.data
## Loading required package:  MASS
##
## Attaching package:  'MASS'
## The following object is masked from 'package:sm':
##
##    muscle
##
## Attaching package:  'TH.data'
## The following object is masked from 'package:MASS':
##
##    geyser
## The following object is masked from 'package:sm':
##
##    geyser
##
## Attaching package:  'multcomp'
## The following object is masked _by_ '.GlobalEnv':
##
##    waste
# Use the ANOVA object and run a "General Linear Hypothesis Test"
# specifying a linfct (linear function) to be tested.
# The mpc (multiple comparison) specifies the factor and method.
# Here: correcting over Treatment using Tukey contrast corrections.
glht.itch.t <- glht(aov(lm.s.t.p), linfct = mcp(Treatment = "Tukey"))
summary(glht.itch.t)
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
```

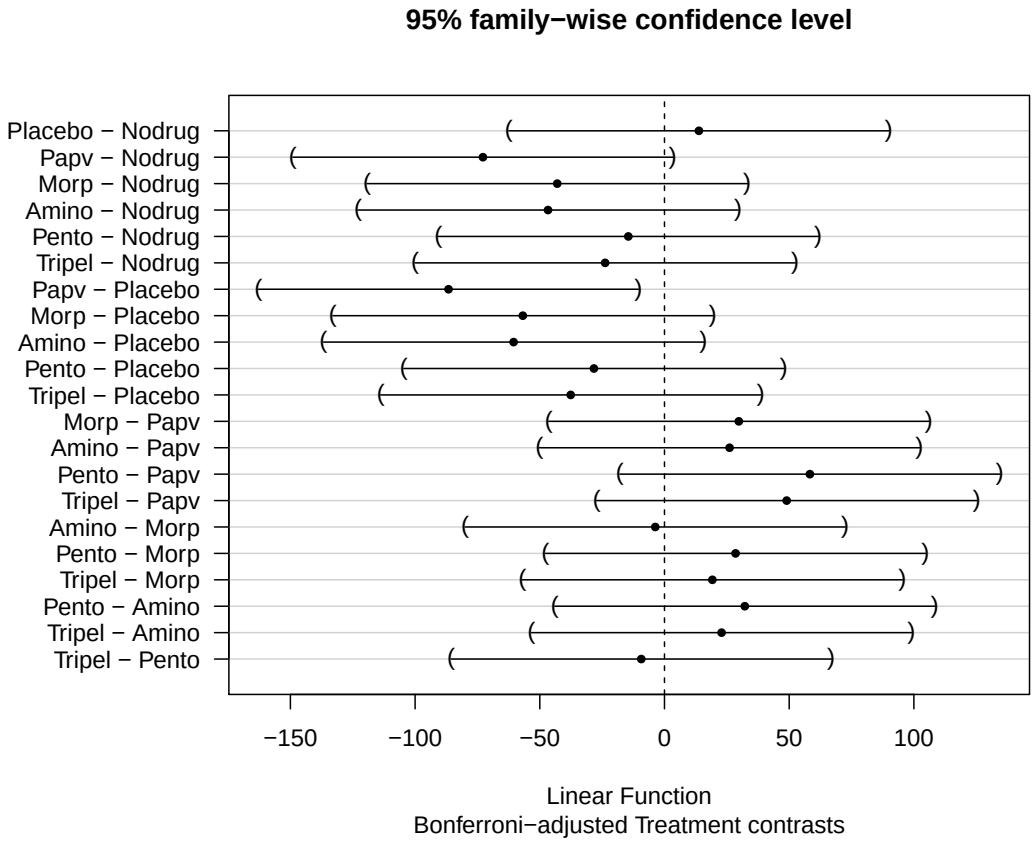
```
##
## Fit: aov(formula = lm.s.t.p)
##
## Linear Hypotheses:
##                        Estimate Std. Error t value Pr(>|t|)
## Placebo - Nodrug == 0    13.80      24.88   0.555   0.9978
## Papv - Nodrug == 0      -72.80      24.88  -2.926   0.0699 .
## Morp - Nodrug == 0      -43.00      24.88  -1.728   0.6003
## Amino - Nodrug == 0     -46.70      24.88  -1.877   0.5038
## Pento - Nodrug == 0     -14.50      24.88  -0.583   0.9971
## Tripel - Nodrug == 0    -23.80      24.88  -0.957   0.9610
## Papv - Placebo == 0     -86.60      24.88  -3.481   0.0162 *
## Morp - Placebo == 0     -56.80      24.88  -2.283   0.2710
## Amino - Placebo == 0    -60.50      24.88  -2.432   0.2054
## Pento - Placebo == 0    -28.30      24.88  -1.137   0.9135
## Tripel - Placebo == 0   -37.60      24.88  -1.511   0.7370
## Morp - Papv == 0         29.80      24.88   1.198   0.8920
## Amino - Papv == 0        26.10      24.88   1.049   0.9398
## Pento - Papv == 0        58.30      24.88   2.343   0.2434
## Tripel - Papv == 0       49.00      24.88   1.969   0.4456
## Amino - Morp == 0        -3.70      24.88  -0.149   1.0000
## Pento - Morp == 0        28.50      24.88   1.146   0.9108
## Tripel - Morp == 0       19.20      24.88   0.772   0.9867
## Pento - Amino == 0       32.20      24.88   1.294   0.8516
## Tripel - Amino == 0      22.90      24.88   0.920   0.9676
## Tripel - Pento == 0      -9.30      24.88  -0.374   0.9998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

With `summary()`, the p-value adjustment can be coerced into one of several popular methods, such as Bonferroni. Notice that the significance is lower (larger p-value) for Bonferroni below than Tukey above. Note comment at bottom of output that "`(Adjusted p values reported -- bonferroni method)`". Passing the summary to `plot()` will create a plot of the pairwise intervals for difference between factor levels.

Recall how the **Bonferroni** correction works. A comparison of c pairs of levels from one factor having a family error rate of 0.05 or less is attained by comparing pairs of treatments at the $0.05/c$ level. Using this criteria, the population mean response for factor levels (averaged over the other factor) are significantly different if the p-value for the test is $0.05/c$ or less. The output actually adjusts the p-values by reporting p-value$\times c$, so that the reported adjusted p-value can be compared to the 0.05 significance level.

```
summary(glht.itch.t, test = adjusted("bonferroni"))
##
```

```
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = lm.s.t.p)
##
## Linear Hypotheses:
##                         Estimate Std. Error t value Pr(>|t|)
## Placebo - Nodrug == 0    13.80      24.88   0.555    1.000
## Papv - Nodrug == 0      -72.80      24.88  -2.926    0.105
## Morp - Nodrug == 0      -43.00      24.88  -1.728    1.000
## Amino - Nodrug == 0     -46.70      24.88  -1.877    1.000
## Pento - Nodrug == 0     -14.50      24.88  -0.583    1.000
## Tripel - Nodrug == 0    -23.80      24.88  -0.957    1.000
## Papv - Placebo == 0     -86.60      24.88  -3.481    0.021 *
## Morp - Placebo == 0     -56.80      24.88  -2.283    0.554
## Amino - Placebo == 0    -60.50      24.88  -2.432    0.386
## Pento - Placebo == 0    -28.30      24.88  -1.137    1.000
## Tripel - Placebo == 0   -37.60      24.88  -1.511    1.000
## Morp - Papv == 0         29.80      24.88   1.198    1.000
## Amino - Papv == 0        26.10      24.88   1.049    1.000
## Pento - Papv == 0        58.30      24.88   2.343    0.479
## Tripel - Papv == 0       49.00      24.88   1.969    1.000
## Amino - Morp == 0        -3.70      24.88  -0.149    1.000
## Pento - Morp == 0        28.50      24.88   1.146    1.000
## Tripel - Morp == 0       19.20      24.88   0.772    1.000
## Pento - Amino == 0       32.20      24.88   1.294    1.000
## Tripel - Amino == 0      22.90      24.88   0.920    1.000
## Tripel - Pento == 0      -9.30      24.88  -0.374    1.000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
# plot the summary
  op <- par(no.readonly = TRUE) # the whole list of settable par's.
  # make wider left margin to fit contrast labels
  par(mar = c(5, 10, 4, 2) + 0.1)  # order is c(bottom, left, top, right)
# plot bonferroni-corrected difference intervals
plot(summary(glht.itch.t, test = adjusted("bonferroni"))
    , sub="Bonferroni-adjusted Treatment contrasts")
  par(op) # reset plotting options
```



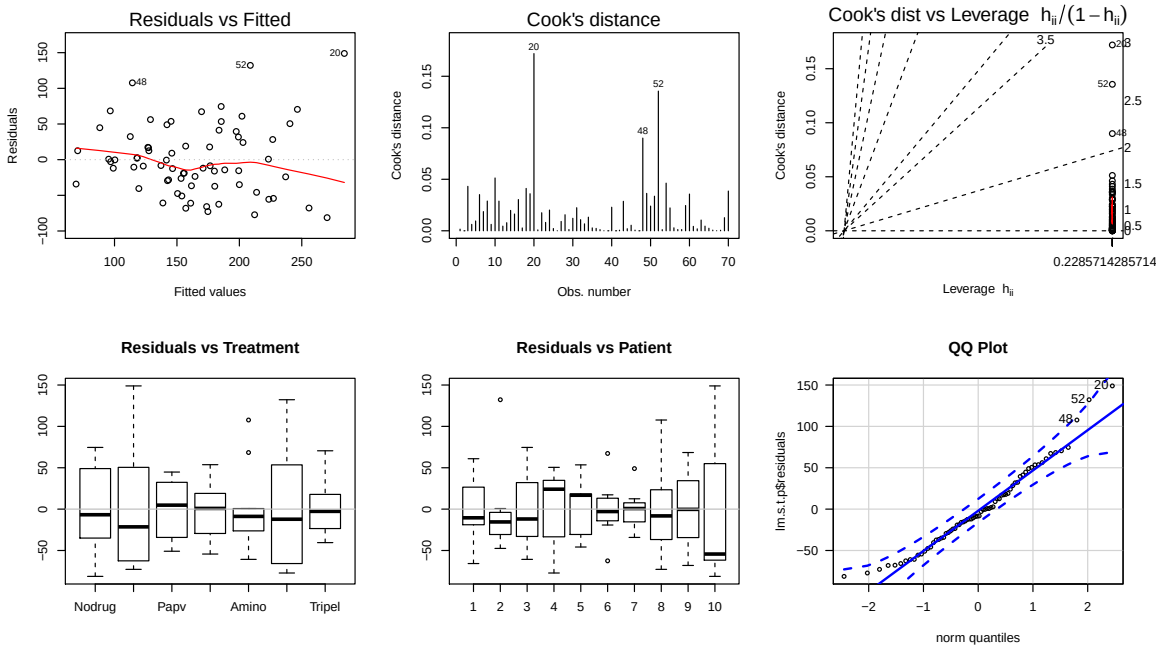
The Bonferroni comparisons for Treatment suggest that papaverine induces a lower mean itching time than placebo. All the other comparisons of treatments are insignificant. The comparison of Patient blocks is of less interest.

```
### Code for the less interesting contrasts.
### Testing multiple factors may be of interest in other problems.
### Note that the first block of code below corrects the p-values
###   for all the tests done for both factors together,
###   that is, the Bonferroni-corrected significance level is (alpha / (t + p))
###   where t = number of treatment comparisons
###   and   p = number of patient   comparisons.

# # correcting over Treatment and Patient
# glht.itch.tp <- glht(aov(lm.s.t.p), linfct = mcp(Treatment = "Tukey"
#                                                  , Patient   = "Tukey"))
# summary(glht.itch.tp, test = adjusted("bonferroni"))
# plot(summary(glht.itch.tp, test = adjusted("bonferroni")))

# # correcting over Patient, only
# glht.itch.p <- glht(aov(lm.s.t.p), linfct = mcp(Patient = "Tukey"))
# summary(glht.itch.p, test = adjusted("bonferroni"))
# plot(summary(glht.itch.p, test = adjusted("bonferroni")))
```

Diagnostic Analysis for the RB Analysis A diagnostic analysis of ANOVA-type models, including the RB model, is easily performed using the `lm()` output. The normal quantile (or QQ-plot) shows the residual distribution is slightly skewed to the right, in part, due to three cases that are not fitted well by the model (the outliers

in the boxplots). Except for these cases, which are also the most influential cases (Cook's distance), the plot of the studentized residuals against fitted values shows no gross abnormalities.

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.s.t.p, which = c(1,4,6))

plot(itch.long$Treatment, lm.s.t.p$residuals, main="Residuals vs Treatment")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(itch.long$Patient, lm.s.t.p$residuals, main="Residuals vs Patient")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.s.t.p$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1] 20 52 48

## residuals vs order of data
#plot(lm.s.t.p$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Although the F-test for comparing treatments is not overly sensitive to modest deviations from normality, I will present a non-parametric analysis as a backup, to see whether similar conclusions are reached about the treatments.

Non-parametric Analysis of a RB Experiment Milton Friedman developed a non-parametric test for comparing treatments in an unreplicated randomized block design where the normality assumption may be violated. An unreplicated complete block design has exactly one observation in y for each combination of levels of groups and blocks. The null hypothesis is that apart from an effect of blocks, the location

parameter of y is the same in each of the groups.

The output suggests significant differences among treatments, which supports the earlier conclusion.

```
# Friedman test for differences between groups conditional on blocks.
#   The formula is of the form a ~ b | c,
#     where a, b and c give the data values (a)
#     and corresponding groups (b) and blocks (c), respectively.
friedman.test(Seconds ~ Treatment | Patient, data = itch.long)
```

```
##
##  Friedman rank sum test
##
## data:  Seconds and Treatment and Patient
## Friedman chi-squared = 14.887, df = 6, p-value = 0.02115
```

```
# Quade test is very similar to the Friedman test (compare the help pages).
quade.test(Seconds ~ Treatment | Patient, data = itch.long)
```

```
##
##  Quade test
##
## data:  Seconds and Treatment and Patient
## Quade F = 3.7321, num df = 6, denom df = 54, p-value =
## 0.003542
```

5.2 Extending the One-Factor Design to Multiple Factors

The CRD (completely randomized design) for comparing insecticides below varies the levels of one factor (insecticide), while controlling other factors that influence survival time. The inferences from the one-way ANOVA apply to beetles with a given age from the selected strain that might be given the selected concentration of the insecticides. Any generalization of the conclusions to other situations must be justified scientifically, typically through further experimentation.

There are several ways to broaden the scope of the study. For example, several strains of beetles or several concentrations of the insecticide might be used. For simplicity, consider a simple two-factor experiment where three concentrations (Low, Medium, and High) are applied with each of the four insecticides. This is a completely crossed **two-factor experiment** where each of the $4 \times 3 = 12$ combinations of the two factors (insecticide and dose) are included in the comparison of survival times. With this experiment, the scientist can compare insecticides, compare concentrations, and check for an interaction between dose and insecticide.

Assuming that 48 beetles are available, the scientist would randomly assign them to the 12 experimental groups, giving prespecified numbers of beetles to the 12 groups.

For simplicity, assume that the experiment is **balanced**, that is, the same number of beetles (4) is assigned to each group ($12 \times 4 = 48$). This is a CRD with two factors.

5.2.1 Example: Beetle insecticide two-factor design

The data below were collected using the experimental design just described. The table gives survival times of groups of four beetles randomly allocated to twelve treatment groups obtained by crossing the levels of four insecticides (A, B, C, D) at each of three concentrations of the insecticides (1=Low, 2=Medium, 3=High). This is a balanced 4-by-3 **factorial** design (two-factor design) that is replicated four times. The unit of measure for the survival times is 10 hours, that is, 0.3 is a survival time of 3 hours.

```
#### Example: Beetles
beetles <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch05_beetles.dat"
                      , header = TRUE)
# make dose a factor variable and label the levels
beetles$dose <- factor(beetles$dose, labels = c("low","medium","high"))
```

	dose	insecticide	t1	t2	t3	t4
1	low	A	0.3100	0.4500	0.4600	0.4300
2	low	B	0.8200	1.1000	0.8800	0.7200
3	low	C	0.4300	0.4500	0.6300	0.7600
4	low	D	0.4500	0.7100	0.6600	0.6200
5	medium	A	0.3600	0.2900	0.4000	0.2300
6	medium	B	0.9200	0.6100	0.4900	1.2400
7	medium	C	0.4400	0.3500	0.3100	0.4000
8	medium	D	0.5600	1.0200	0.7100	0.3800
9	high	A	0.2200	0.2100	0.1800	0.2300
10	high	B	0.3000	0.3700	0.3800	0.2900
11	high	C	0.2300	0.2500	0.2400	0.2200
12	high	D	0.3000	0.3600	0.3100	0.3300

First we reshape the data to long format so each observation is its own row in the `data.frame` and indexed by the `dose` and `insecticide` variables.

```
library(reshape2)
beetles.long <- melt(beetles
                , id.vars       = c("dose", "insecticide")
                , variable.name = "number"
                , value.name    = "hours10"
             )
str(beetles.long)
```

```
## 'data.frame': 48 obs. of  4 variables:
##  $ dose       : Factor w/ 3 levels "low","medium",..: 1 1 1 1 2 2 2 2 3 3 ...
##  $ insecticide: Factor w/ 4 levels "A","B","C","D": 1 2 3 4 1 2 3 4 1 2 ...
##  $ number     : Factor w/ 4 levels "t1","t2","t3",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ hours10    : num  0.31 0.82 0.43 0.45 0.36 0.92 0.44 0.56 0.22 0.3 ...
```

```
head(beetles.long)
##     dose insecticide number hours10
## 1    low           A     t1    0.31
## 2    low           B     t1    0.82
## 3    low           C     t1    0.43
## 4    low           D     t1    0.45
## 5 medium           A     t1    0.36
## 6 medium           B     t1    0.92
```

The basic unit of analysis is the **cell means**, which are the averages of the 4 observations in each of the 12 treatment combinations. For example, in the table below, the sample mean survival for the 4 beetles given a low dose (dose=1) of insecticide A is 0.413. From the cell means we obtain the dose and insecticide **marginal means** by averaging over the levels of the other factor. For example, the marginal mean for insecticide A is the average of the cell means for the 3 treatment combinations involving insecticide A: $0.314 = (0.413 + 0.320 + 0.210)/3$.

Cell Means	Dose			
Insecticide	1	2	3	Insect marg
A	0.413	0.320	0.210	0.314
B	0.880	0.815	0.335	0.677
C	0.568	0.375	0.235	0.393
D	0.610	0.668	0.325	0.534
Dose marg	0.618	0.544	0.277	0.480

Because the experiment is balanced, a marginal mean is the average of all observations that receive a given treatment. For example, the marginal mean for insecticide A is the average survival time for the 16 beetles given insecticide A.

Looking at the table of means, the insecticides have noticeably different mean survival times averaged over doses, with insecticide A having the lowest mean survival time averaged over doses. Similarly, higher doses tend to produce lower survival times. A more formal approach to analyzing the table of means is given in the next section.

5.2.2 The Interaction Model for a Two-Factor Experiment

Assume that you designed a **balanced** two-factor experiment with K responses at each combination of the I levels of factor 1 (F1) with the J levels of factor 2 (F2). The total number of responses is KIJ, or K times the IJ treatment combinations.

Let y_{ijk} be the k^{th} response at the i^{th} level of F1 and the j^{th} level of F2. A generic model for the experiment expresses y_{ijk} as a mean response plus a residual:

$$y_{ijk} = \mu_{ij} + e_{ijk},$$

where μ_{ij} is the population mean response for the treatment defined by the i^{th} level of F1 combined with the j^{th} level of F2. As in a one-way ANOVA, the responses within

and across treatment groups are assumed to be independent, normally distributed, and have constant variance.

The **interaction model** expresses the population means as

$$\mu_{ij} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij},$$

where μ is a grand mean, α_i is the effect for the i^{th} level of F1, β_j is the effect for the j^{th} level of F2, and $(\alpha\beta)_{ij}$ is the interaction between the i^{th} level of F1 and the j^{th} level of F2. (Note that $(\alpha\beta)$ is an individual term distinct from α and β, $(\alpha\beta)$ is not their product.) The model is often written

$$y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + e_{ijk},$$

meaning

Response = Grand Mean + F1 effect + F2 effect + F1-by-F2 interaction + Residual.

The **additive model** having only main effects, no interaction terms, is $y_{ijk} = \mu + \alpha_i + \beta_j + e_{ijk}$, meaning

Response = Grand Mean + F1 effect + F2 effect + Residual.

The effects of F1 and F2 on the mean are additive.

Defining effects from cell means

The effects that define the population means and the usual hypotheses of interest can be formulated from the table of population means, given here.

	Level of F2				
Level of F1	1	2	$\cdots$	J	F1 marg
1	μ_{11}	μ_{12}	$\cdots$	μ_{1J}	$\bar{\mu}_{1\cdot}$
2	μ_{21}	μ_{22}	$\cdots$	μ_{2J}	$\bar{\mu}_{2\cdot}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
I	μ_{I1}	μ_{I2}	$\cdots$	μ_{IJ}	$\bar{\mu}_{I\cdot}$
F2 marg	$\bar{\mu}_{\cdot 1}$	$\bar{\mu}_{\cdot 2}$	$\cdots$	$\bar{\mu}_{\cdot J}$	$\bar{\mu}_{\cdot\cdot}$

The F1 **marginal population means** are averages within rows (over columns) of the table:

$$\bar{\mu}_{i\cdot} = \frac{1}{J}\sum_{c} \mu_{ic\cdot}$$

The F2 marginal population means are averages within columns (over rows):

$$\bar{\mu}_{\cdot j} = \frac{1}{I}\sum_{r} \mu_{rj\cdot}$$

The overall or **grand population mean** is the average of the cell means

$$\bar{\mu}_{\cdot\cdot} = \frac{1}{IJ}\sum_{rc}\mu_{rc} = \frac{1}{I}\sum_{i}\bar{\mu}_{i\cdot} = \frac{1}{J}\sum_{j}\bar{\mu}_{\cdot j}.$$

Using this notation, the effects in the interaction model are $\mu = \bar{\mu}_{\cdot\cdot}$, $\alpha_i = \bar{\mu}_{i\cdot} - \bar{\mu}_{\cdot\cdot}$, $\beta_j = \bar{\mu}_{\cdot j} - \bar{\mu}_{\cdot\cdot}$, and $(\alpha\beta)_{ij} = \mu_{ij} - \bar{\mu}_{i\cdot} - \bar{\mu}_{\cdot j} + \bar{\mu}_{\cdot\cdot}$. The effects sum to zero:

$$\sum_i \alpha_i = \sum_j \beta_j = \sum_{ij}(\alpha\beta)_{ij} = 0,$$

and satisfy $\mu_{ij} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij}$ (i.e., cell mean is sum of effects) required under the model.

The F1 and F2 effects are analogous to treatment effects in a one-factor experiment, except that here the treatment means are averaged over the levels of the other factor. The interaction effect will be interpreted later.

Estimating effects from the data

Let

$$\bar{y}_{ij} = \frac{1}{K}\sum_k y_{ijk} \qquad \text{and} \qquad s^2_{ij}$$

be the sample mean and variance, respectively, for the K responses at the i^{th} level of F1 and the j^{th} level of F2. Inferences about the population means are based on the table of sample means:

	Level of F2				
Level of F1	1	2	$\cdots$	J	F1 marg
1	$\bar{y}_{11}$	$\bar{y}_{12}$	$\cdots$	$\bar{y}_{1J}$	$\bar{y}_{1\cdot}$
2	$\bar{y}_{21}$	$\bar{y}_{22}$	$\cdots$	$\bar{y}_{2J}$	$\bar{y}_{2\cdot}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
I	$\bar{y}_{I1}$	$\bar{y}_{I2}$	$\cdots$	$\bar{y}_{IJ}$	$\bar{y}_{I\cdot}$
F2 marg	$\bar{y}_{\cdot 1}$	$\bar{y}_{\cdot 2}$	$\cdots$	$\bar{y}_{\cdot J}$	$\bar{y}_{\cdot\cdot}$

The F1 **marginal sample means** are averages within rows of the table:

$$\bar{y}_{i\cdot} = \frac{1}{J}\sum_c \bar{y}_{ic\cdot}$$

The F2 marginal sample means are averages within columns:

$$\bar{y}_{\cdot j} = \frac{1}{I}\sum_r \bar{y}_{rj\cdot}$$

Finally, $\bar{y}_{..}$ is the average of the cell sample means:

$$\bar{y}_{..} = \frac{1}{IJ}\sum_{ij}\bar{y}_{ij} = \frac{1}{I}\sum_{i}\bar{y}_{i\cdot} = \frac{1}{J}\sum_{j}\bar{y}_{\cdot j}.$$

The sample sizes in each of the IJ treatment groups are equal (K), so $\bar{y}_{i\cdot}$ is the sample average of all responses at the i^{th} level of F1, $\bar{y}_{\cdot j}$ is the sample average of all responses at the j^{th} level of F2, and $\bar{y}_{..}$ is the average response in the experiment.

Under the interaction model, the estimated population mean for the $(i,j)^{th}$ cell is the observed cell mean: $\hat{\mu}_{ij} = \bar{y}_{ij}$. This can be partitioned into estimated effects

$$\begin{aligned}
\hat{\mu} &= \bar{y}_{..} \quad \text{the estimated grand mean} \\
\hat{\alpha}_i &= \bar{y}_{i\cdot} - \bar{y}_{..} \quad \text{the estimated F1 effect } i = 1, 2, \ldots, I \\
\hat{\beta}_j &= \bar{y}_{\cdot j} - \bar{y}_{..} \quad \text{the estimated F2 effect } j = 1, 2, \ldots, J \\
\widehat{(\alpha\beta)}_{ij} &= \bar{y}_{ij} - \bar{y}_{i\cdot} - \bar{y}_{\cdot j} + \bar{y}_{..} \quad \text{the estimated cell interaction}
\end{aligned} \tag{5.1}$$

that satisfy

$$\hat{\mu}_{ij} = \hat{\mu} + \hat{\alpha}_i + \hat{\beta}_j + \widehat{(\alpha\beta)}_{ij}.$$

The ANOVA table

The ANOVA table for a balanced two-factor design decomposes the total variation in the data, as measured by the Total SS, into components that measure the variation of marginal sample means for F1 and F2 (the F1 SS and F2 SS), a component that measures the degree to which the factors interact (the F1-by-F2 Interaction SS), and a component that pools the sample variances across the IJ samples (the Error SS). Each SS has a df, given in the following ANOVA table. As usual, the MS for each source of variation is the corresponding SS divided by the df. The MS Error estimates the common population variance for the IJ treatments.

Source	*df*	SS	MS
F1	$I-1$	$KJ\sum_i(\bar{y}_{i\cdot}-\bar{y}_{..})^2$	MS F1=SS/df
F2	$J-1$	$KI\sum_j(\bar{y}_{\cdot j}-\bar{y}_{..})^2$	MS F2=SS/df
Interaction	$(I-1)(J-1)$	$K\sum_{ij}(y_{ij}-\bar{y}_{i\cdot}-\bar{y}_{\cdot j}+\bar{y}_{..})^2$	MS Inter=SS/df
Error	$IJ(K-1)$	$(K-1)\sum_{ij}s_{ij}^2$	MSE=SS/df
Total	$IJK-1$	$\sum_{ijk}(y_{ijk}-\bar{y}_{..})^2$.	

The standard tests in the two-factor analysis, and their interpretations are:

The test of no F1 effect: $H_0 : \alpha_1 = \cdots = \alpha_I = 0$ is equivalent to testing $H_0 : \bar{\mu}_{1\cdot} = \bar{\mu}_{2\cdot} = \cdots = \bar{\mu}_{I\cdot}$. The absence of an F1 effect implies that each level of F1 has the same population mean response **when the means are averaged over levels**

of F2. The test for no F1 effect is based on MS F1/MS Error, which is compared to the upper tail of an F-distribution with numerator and denominator df of $I-1$ and $IJ(K-1)$, respectively. H_0 is rejected when the F1 marginal means $\bar{y}_{i\cdot}$ vary significantly relative to the within sample variation. Equivalently, H_0 is rejected when the sum of squared F1 effects (between sample variation) is large relative to the within sample variation.

The test of no F2 effect: $H_0 : \beta_1 = \cdots = \beta_J = 0$ is equivalent to testing $H_0 : \bar{\mu}_{\cdot 1} = \bar{\mu}_{\cdot 2} = \cdots = \bar{\mu}_{\cdot J}$. The absence of a F2 effect implies that each level of F2 has the same population mean response **when the means are averaged over levels of** F1. The test for no F2 effect is based on MS F2/MS Error, which is compared to an F-distribution with numerator and denominator df of $J-1$ and $IJ(K-1)$, respectively. H_0 is rejected when the F2 marginal means $\bar{y}_{\cdot j}$ vary significantly relative to the within sample variation. Equivalently, H_0 is rejected when the sum of squared F2 effects (between sample variation) is large relative to the within sample variation.

The test of no interaction: $H_0 : (\alpha\beta)_{ij} = 0$ for all i and j is based on MS Interact/MS Error, which is compared to an F-distribution with numerator and denominator df of $(I-1)(J-1)$ and $IJ(K-1)$, respectively.

The interaction model places no restrictions on the population means μ_{ij}. Since the population means can be arbitrary, the interaction model can be viewed as a one factor model with IJ treatments. One connection between the two ways of viewing the two-factor analysis is that the F1, F2, and Interaction SS for the two-way interaction model sum to the Treatment or Model SS for comparing the IJ treatments. The Error SS for the two-way interaction model is identical to the Error SS for a one-way ANOVA of the IJ treatments. An overall test of no differences in the IJ population means is part of the two-way analysis.

I always summarize the data using the cell and marginal means instead of the estimated effects, primarily because means are the basic building blocks for the analysis. My discussion of the model and tests emphasizes both approaches to help you make the connection with the two ways this material is often presented in texts.

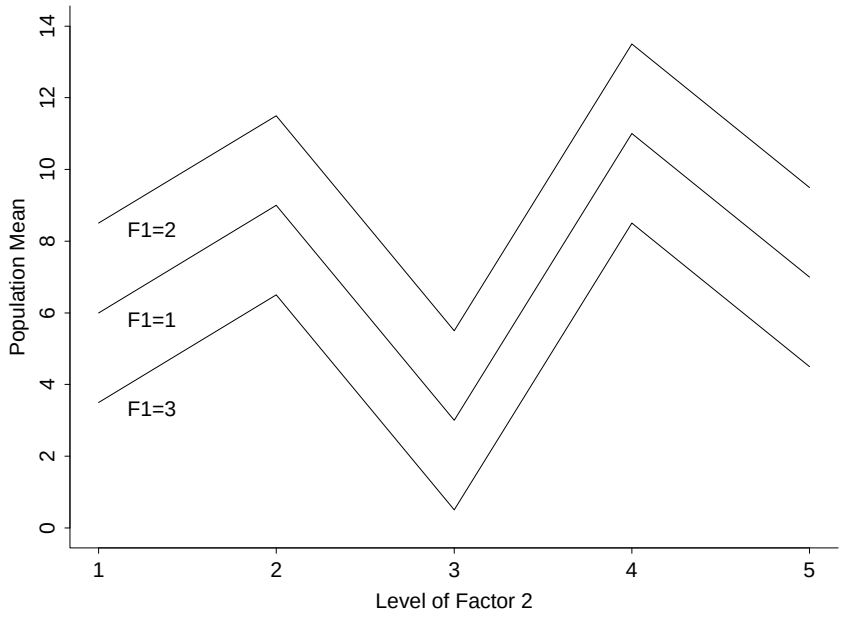
Understanding interaction

To understand interaction, suppose you (conceptually) plot the means in each row of the population table, giving what is known as the **population mean profile** plot. The F1 marginal population means average the population means within the F1 profiles. At each F2 level, the F2 marginal mean averages the population cell means across F1 profiles.

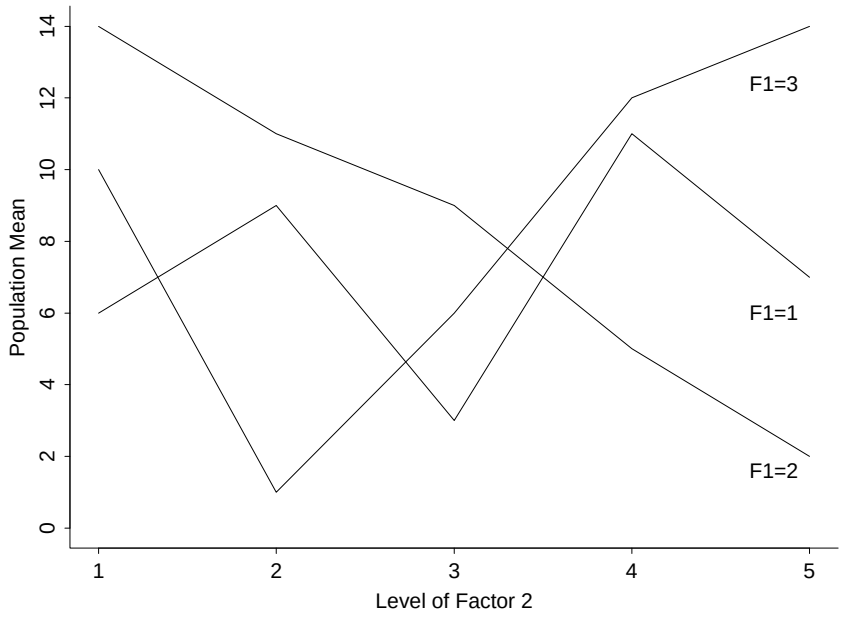
No interaction is present if the plot has perfectly **parallel** F1 profiles, as in the plot below for a 3×5 experiment. The levels of F1 and F2 **do not interact**. That

is,

parallel profiles $\Leftrightarrow$ $\mu_{ij} - \mu_{hj}$ is independent of j for each i and h
difference between levels of F1 does not depend on level of F2
$\Leftrightarrow$ $\mu_{ij} - \bar{\mu}_{i\cdot} = \mu_{hj} - \bar{\mu}_{h\cdot}$ for all i, j, h
difference between level of F2 j and F2 mean does not depend on level of F1
$\Leftrightarrow$ $\mu_{ij} - \bar{\mu}_{i\cdot} = \bar{\mu}_{\cdot j} - \bar{\mu}_{\cdot\cdot}$ for all i, j
difference between level of F2 j and F2 mean is the same for all levels of F1
$\Leftrightarrow$ $\mu_{ij} - \bar{\mu}_{i\cdot} - \bar{\mu}_{\cdot j} + \bar{\mu}_{\cdot\cdot} = 0$ for all i, j
interaction effect is 0
$\Leftrightarrow$ $(\alpha\beta)_{ij} = 0$ for all i, j
interaction effect is 0
$\Leftrightarrow$ no interaction term in model.



Interaction is present if the profiles are not **perfectly parallel**. An example of a profile plot for two-factor experiment (3×5) with interaction is given below.



The roles of F1 and F2 can be reversed in these plots without changing the assessment of a presence or absence of interaction. It is often helpful to view the interaction plot from both perspectives.

A qualitative check for interaction can be based on the **sample means profile plot**, but keep in mind that profiles of sample means are never perfectly parallel even when the factors do not interact in the population. The Interaction SS measures the extent of non-parallelism in the sample mean profiles. In particular, the Interaction SS is zero when the sample mean profiles are perfectly parallel because $\widehat{(\alpha\beta)}_{ij} = 0$ for all i and j.

5.2.3 Example: Survival Times of Beetles

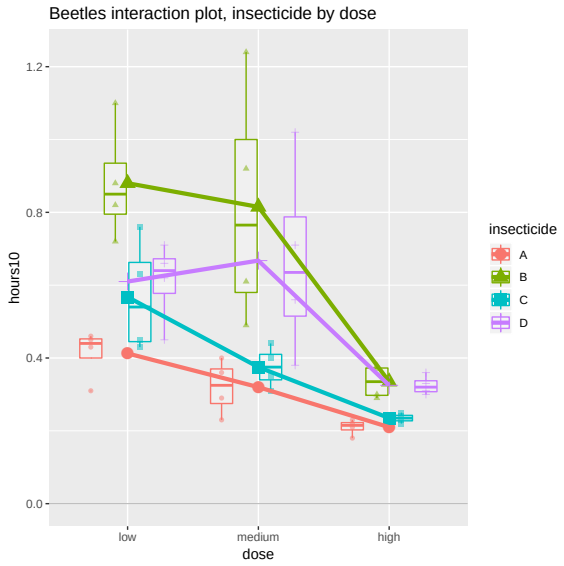
First we generate cell means and a sample means profile plot (interaction plot) for the beetle experiment. The `ddply()` function was used to obtain the 12 treatment cell means. Three variables were needed to represent each response in the data set: dose (1-3, categorical), insecticide (A-D, categorical), and time (the survival time).

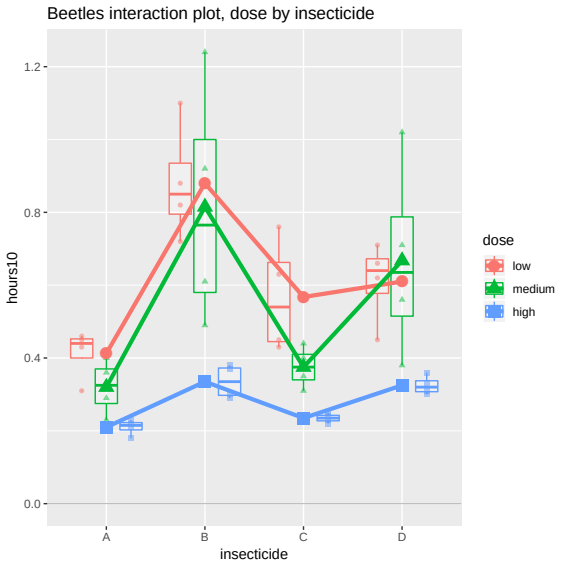
As noted earlier, the insecticides have noticeably different mean survival times averaged over doses, with insecticide A having the lowest mean. Similarly, higher doses tend to produce lower survival times. The sample means profile plot shows some evidence of interaction.

```
library(plyr)
# Calculate the cell means for each (dose, insecticide) combination
```

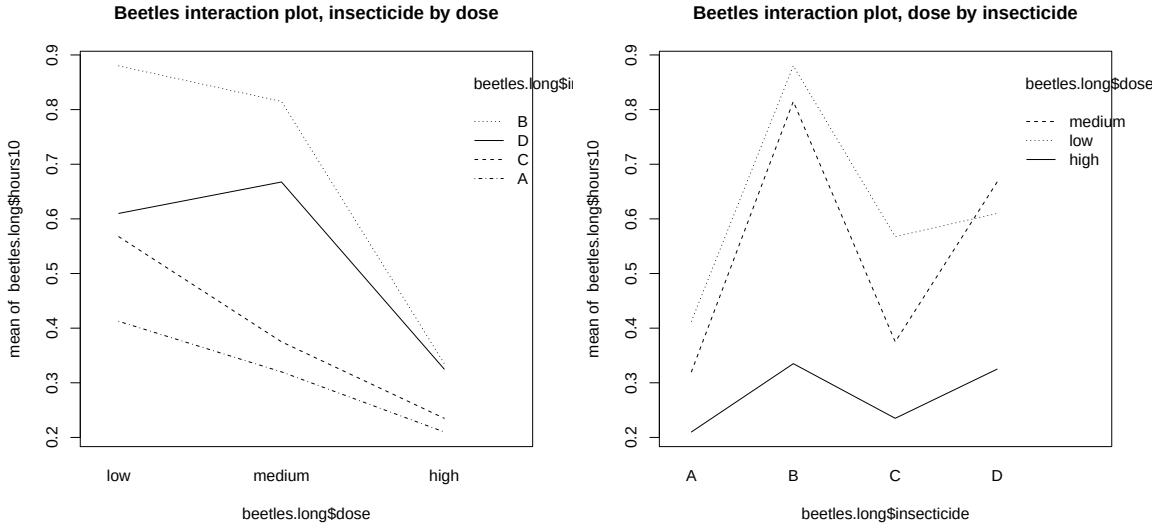
```
mean(beetles.long[, "hours10"])
## [1] 0.479375
beetles.mean    <- ddply(beetles.long, .(), summarise, m = mean(hours10))
beetles.mean
##    .id        m
## 1 <NA> 0.479375
beetles.mean.d  <- ddply(beetles.long, .(dose), summarise, m = mean(hours10))
beetles.mean.d
##     dose        m
## 1    low 0.617500
## 2 medium 0.544375
## 3   high 0.276250
beetles.mean.i  <- ddply(beetles.long, .(insecticide), summarise, m = mean(hours10))
beetles.mean.i
##   insecticide         m
## 1           A 0.3141667
## 2           B 0.6766667
## 3           C 0.3925000
## 4           D 0.5341667
beetles.mean.di <- ddply(beetles.long, .(dose,insecticide), summarise, m = mean(hours10))
beetles.mean.di
##      dose insecticide      m
## 1     low           A 0.4125
## 2     low           B 0.8800
## 3     low           C 0.5675
## 4     low           D 0.6100
## 5  medium           A 0.3200
## 6  medium           B 0.8150
## 7  medium           C 0.3750
## 8  medium           D 0.6675
## 9    high           A 0.2100
## 10   high           B 0.3350
## 11   high           C 0.2350
## 12   high           D 0.3250
# Interaction plots, ggplot

p <- ggplot(beetles.long, aes(x = dose, y = hours10, colour = insecticide, shape = insecticide))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                  , linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = beetles.mean.di, aes(y = m), size = 4)
p <- p + geom_line(data = beetles.mean.di, aes(y = m, group = insecticide), size = 1.5)
p <- p + labs(title = "Beetles interaction plot, insecticide by dose")
print(p)
```

```
p <- ggplot(beetles.long, aes(x = insecticide, y = hours10, colour = dose, shape = dose))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                    , linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = beetles.mean.di, aes(y = m), size = 4)
p <- p + geom_line(data = beetles.mean.di, aes(y = m, group = dose), size = 1.5)
p <- p + labs(title = "Beetles interaction plot, dose by insecticide")
print(p)
```






```
# Interaction plots, base graphics

interaction.plot(beetles.long$dose, beetles.long$insecticide, beetles.long$hours10
  , main = "Beetles interaction plot, insecticide by dose")
interaction.plot(beetles.long$insecticide, beetles.long$dose, beetles.long$hours10
  , main = "Beetles interaction plot, dose by insecticide")
```



In the `lm()` function below we specify a first-order model with interactions, including the main effects and two-way interactions. The interaction between dose and insecticide is indicated with `dose:insecticide`. The shorthand `dose*insecticide` expands to "`dose + insecticide + dose:insecticide`" for this first-order model.

The F-test at the bottom of the `summary()` tests for no differences among the population mean survival times for the 12 dose and insecticide combinations. The p-value of < 0.0001 strongly suggests that the population mean survival times are not all equal.

The next summary at the top gives two partitionings of the one-way ANOVA Treatment SS into the SS for Dose, Insecticide, and the Dose by Insecticide interaction. The Mean Squares, F-statistics and p-values for testing these effects are given. The p-values for the F-statistics indicate that the dose and insecticide effects are significant at the 0.01 level. The F-test for no dose by insecticide interaction is not significant at the 0.10 level (p-value=0.112). Thus, the interaction seen in the profile plot of the sample means might be due solely to chance or sampling variability.

```
lm.h.d.i.di <- lm(hours10 ~ dose + insecticide + dose:insecticide
                  , data = beetles.long)
# lm.h.d.i.di <- lm(hours10 ~ dose*insecticide, data = beetles.long) # equivalent
library(car)
Anova(lm.h.d.i.di, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: hours10
##                   Sum Sq Df F value    Pr(>F)
## (Intercept)      0.68063  1 30.6004 2.937e-06 ***
## dose             0.08222  2  1.8482 0.1721570
## insecticide      0.45395  3  6.8031 0.0009469 ***
```

```
## dose:insecticide 0.25014  6  1.8743 0.1122506
## Residuals        0.80072 36
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.h.d.i.di)
##
## Call:
## lm(formula = hours10 ~ dose + insecticide + dose:insecticide,
##     data = beetles.long)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32500 -0.04875  0.00500  0.04312  0.42500
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               0.41250    0.07457   5.532 2.94e-06 ***
## dosemedium               -0.09250    0.10546  -0.877   0.3862
## dosehigh                 -0.20250    0.10546  -1.920   0.0628 .
## insecticideB              0.46750    0.10546   4.433 8.37e-05 ***
## insecticideC              0.15500    0.10546   1.470   0.1503
## insecticideD              0.19750    0.10546   1.873   0.0692 .
## dosemedium:insecticideB   0.02750    0.14914   0.184   0.8547
## dosehigh:insecticideB    -0.34250    0.14914  -2.297   0.0276 *
## dosemedium:insecticideC -0.10000    0.14914  -0.671   0.5068
## dosehigh:insecticideC    -0.13000    0.14914  -0.872   0.3892
## dosemedium:insecticideD   0.15000    0.14914   1.006   0.3212
## dosehigh:insecticideD    -0.08250    0.14914  -0.553   0.5836
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1491 on 36 degrees of freedom
## Multiple R-squared:  0.7335,Adjusted R-squared:  0.6521
## F-statistic:  9.01 on 11 and 36 DF,  p-value: 1.986e-07
```

Since the interaction is not significant, I'll drop the interaction term and fit the additive model with main effects only. I update the model by removing the interaction term.

```
lm.h.d.i <- update(lm.h.d.i.di, ~ . - dose:insecticide )
library(car)
Anova(lm.h.d.i, type=3)
## Anova Table (Type III tests)
##
## Response: hours10
##              Sum Sq Df F value    Pr(>F)
## (Intercept) 1.63654  1  65.408 4.224e-10 ***
## dose        1.03301  2  20.643 5.704e-07 ***
```

```
## insecticide 0.92121  3  12.273 6.697e-06 ***
## Residuals   1.05086 42
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.h.d.i)
##
## Call:
## lm(formula = hours10 ~ dose + insecticide, data = beetles.long)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.25167 -0.09625 -0.01490  0.06177  0.49833
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.45229    0.05592   8.088 4.22e-10 ***
## dosemedium   -0.07313    0.05592  -1.308  0.19813
## dosehigh     -0.34125    0.05592  -6.102 2.83e-07 ***
## insecticideB  0.36250    0.06458   5.614 1.43e-06 ***
## insecticideC  0.07833    0.06458   1.213  0.23189
## insecticideD  0.22000    0.06458   3.407  0.00146 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1582 on 42 degrees of freedom
## Multiple R-squared:  0.6503,Adjusted R-squared:  0.6087
## F-statistic: 15.62 on 5 and 42 DF,  p-value: 1.123e-08
```

The Bonferroni multiple comparisons indicate which treatment effects are different.

```
# Testing multiple factors is of interest here.
# Note that the code below corrects the p-values
#   for all the tests done for both factors together,
#   that is, the Bonferroni-corrected significance level is (alpha / (d + i))
#   where d = number of dose        comparisons
#   and   i = number of insecticide comparisons.

# correcting over dose and insecticide
glht.beetle.di <- glht(aov(lm.h.d.i), linfct = mcp(dose = "Tukey"
                                                   , insecticide = "Tukey"))
summary(glht.beetle.di, test = adjusted("bonferroni"))
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
```

```
## Fit: aov(formula = lm.h.d.i)
##
## Linear Hypotheses:
##                            Estimate Std. Error t value Pr(>|t|)
## dose: medium - low == 0  -0.07313    0.05592  -1.308 1.000000
## dose: high - low == 0    -0.34125    0.05592  -6.102 2.55e-06 ***
## dose: high - medium == 0 -0.26812    0.05592  -4.794 0.000186 ***
## insecticide: B - A == 0   0.36250    0.06458   5.614 1.28e-05 ***
## insecticide: C - A == 0   0.07833    0.06458   1.213 1.000000
## insecticide: D - A == 0   0.22000    0.06458   3.407 0.013134 *
## insecticide: C - B == 0  -0.28417    0.06458  -4.400 0.000653 ***
## insecticide: D - B == 0  -0.14250    0.06458  -2.207 0.295702
## insecticide: D - C == 0   0.14167    0.06458   2.194 0.304527
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
```

```
# plot the summary
  op <- par(no.readonly = TRUE) # the whole list of settable par's.
  # make wider left margin to fit contrast labels
  par(mar = c(5, 10, 4, 2) + 0.1)  # order is c(bottom, left, top, right)
# plot bonferroni-corrected difference intervals
plot(summary(glht.beetle.di, test = adjusted("bonferroni"))
    , sub="Bonferroni-adjusted Treatment contrasts")
  par(op) # reset plotting options
```

Interpretation of the Dose and Insecticide Effects

The interpretation of the dose and insecticide **main effects** depends on whether interaction is present. The distinction is important, so I will give both interpretations to emphasize the differences. Given the test for interaction, I would likely summarize the main effects assuming no interaction.

The average survival time decreases as the dose increases, with estimated mean survival times of 0.618, 0.544, and 0.276, respectively. A Bonferroni comparison shows that the population mean survival time for the high dose (averaged over insecticides) is significantly less than the population mean survival times for the low and medium doses (averaged over insecticides). The two lower doses are not significantly different from each other. This leads to two dose groups:

```
Dose:          1=Low  2=Med  3=Hig
Marg Mean:     0.618  0.544  0.276
Groups:        ------------  -----
```

If dose and insecticide **interact**, you can conclude that beetles given a high dose of the insecticide typically survive for shorter periods of time **averaged over insecticides**. You can not, in general, conclude that the highest dose yields the lowest survival time **regardless** of insecticide. For example, the difference in the medium and high dose marginal means (0.544 - 0.276 = 0.268) estimates the typical decrease in survival time achieved by using the high dose instead of the medium dose, averaged over insecticides. If the two factors interact, then the difference in mean times between the medium and high doses on a given insecticide may be significantly greater than 0.268, significantly less than 0.268, or even negative. In the latter case the medium dose would be **better** than the high dose for the given insecticide, even though the high dose gives better performance averaged over insecticides. An interaction forces you to use the cell means to decide which combination of dose and insecticide gives the best results (and the multiple comparisons as they were done above do not give multiple comparisons of cell means; a single factor variable combining both factors would need to be created). Of course, our profile plot tells us that this hypothetical situation is probably not tenable here, but it could be so when a significant interaction is present.

If dose and insecticide **do not interact**, then the difference in marginal dose means averaged over insecticides also estimates the difference in population mean survival times between two doses, **regardless of the insecticide**. This follows from the parallel profiles definition of no interaction. Thus, the difference in the medium and high dose marginal means (0.544 - 0.276 = 0.268) estimates the expected decrease in survival time anticipated from using the high dose instead of the medium dose, **regardless of the insecticide** (and hence also when averaged over insecticides). A practical implication of no interaction is that you can conclude that the high dose is best, regardless of the insecticide used. The difference in marginal means for two doses estimates the difference in average survival expected, regardless of the insecticide.

An ordering of the mean survival times on the four insecticides (averaged over the three doses) is given below. Three groups are obtained from the Bonferroni comparisons, with any two insecticides separated by one or more other insecticides in the ordered string having significantly different mean survival times averaged over doses.

If interaction is present, you can conclude that insecticide A is no better than C, but significantly better than B or D, when performance is **averaged over doses**. If the interaction is absent, then A is not significantly better than C, but is significantly better than B or D, **regardless of the dose**. Furthermore, for example, the difference in marginal means for insecticides B and A of 0.677 - 0.314 = 0.363 is the expected decrease in survival time from using A instead of B, regardless of dose. This is also the expected decrease in survival times when averaged over doses.

```
Insect:          B      D      C      A
Marg Mean:   0.677  0.534  0.393  0.314
Groups:      ------------
                    ------------
                           ------------
```

5.2.4 Example: Output voltage for batteries

The maximum output voltage for storage batteries is thought to be influenced by the temperature in the location at which the battery is operated and the material used in the plates. A scientist designed a two-factor study to examine this hypothesis, using three temperatures (50, 65, 80), and three materials for the plates (1, 2, 3). Four batteries were tested at each of the 9 combinations of temperature and material type. The maximum output voltage was recorded for each battery. This is a balanced 3-by-3 factorial experiment with four observations per treatment.

```
#### Example: Output voltage for batteries
battery <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch05_battery.dat"
                      , header = TRUE)
battery$material <- factor(battery$material)
battery$temp     <- factor(battery$temp)
```

	material	temp	v1	v2	v3	v4
1	1	50	130	155	74	180
2	1	65	34	40	80	75
3	1	80	20	70	82	58
4	2	50	150	188	159	126
5	2	65	136	122	106	115
6	2	80	25	70	58	45
7	3	50	138	110	168	160
8	3	65	174	120	150	139
9	3	80	96	104	82	60

```
library(reshape2)
battery.long <- melt(battery
                , id.vars       = c("material", "temp")
                , variable.name = "battery"
                , value.name    = "maxvolt"
             )
str(battery.long)
```

```
## 'data.frame': 36 obs. of  4 variables:
##  $ material: Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 3 3 3 1 ...
##  $ temp    : Factor w/ 3 levels "50","65","80": 1 2 3 1 2 3 1 2 3 1 ...
##  $ battery : Factor w/ 4 levels "v1","v2","v3",..: 1 1 1 1 1 1 1 1 1 2 ...
##  $ maxvolt : int  130 34 20 150 136 25 138 174 96 155 ...
```

The overall F-test at the bottom indicates at least one parameter in the model is significant. The two-way ANOVA table indicates that the main effect of temperature and the interaction are significant at the 0.05 level, the main effect of material is not.

```
lm.m.m.t.mt <- lm(maxvolt ~ material*temp, data = battery.long)
library(car)
Anova(lm.m.m.t.mt, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: maxvolt
##               Sum Sq Df  F value    Pr(>F)
## (Intercept)    72630  1 107.5664 6.456e-11 ***
## material         886  2   0.6562 0.5268904
## temp           15965  2  11.8223 0.0002052 ***
## material:temp   9614  4   3.5595 0.0186112 *
## Residuals      18231 27
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm.m.m.t.mt)
```

```
##
## Call:
## lm(formula = maxvolt ~ material * temp, data = battery.long)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -60.750 -14.625   1.375  17.938  45.250
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         134.75      12.99  10.371 6.46e-11 ***
## material2            21.00      18.37   1.143 0.263107
## material3             9.25      18.37   0.503 0.618747
## temp65              -77.50      18.37  -4.218 0.000248 ***
## temp80              -77.25      18.37  -4.204 0.000257 ***
## material2:temp65     41.50      25.98   1.597 0.121886
```

```
## material3:temp65    79.25      25.98   3.050 0.005083 **
## material2:temp80   -29.00      25.98  -1.116 0.274242
## material3:temp80    18.75      25.98   0.722 0.476759
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25.98 on 27 degrees of freedom
## Multiple R-squared:  0.7652,Adjusted R-squared:  0.6956
## F-statistic:    11 on 8 and 27 DF,  p-value: 9.426e-07
```

The cell means plots of the material profiles have different slopes, which is consistent with the presence of a temperature-by-material interaction.

```
library(plyr)
# Calculate the cell means for each (material, temp) combination
battery.mean    <- ddply(battery.long, .(), summarise, m = mean(maxvolt))
battery.mean
```

```
##    .id        m
## 1 <NA> 105.5278
```

```
battery.mean.m  <- ddply(battery.long, .(material), summarise, m = mean(maxvolt))
battery.mean.m
```

```
##   material         m
## 1        1  83.16667
## 2        2 108.33333
## 3        3 125.08333
```

```
battery.mean.t  <- ddply(battery.long, .(temp), summarise, m = mean(maxvolt))
battery.mean.t
```

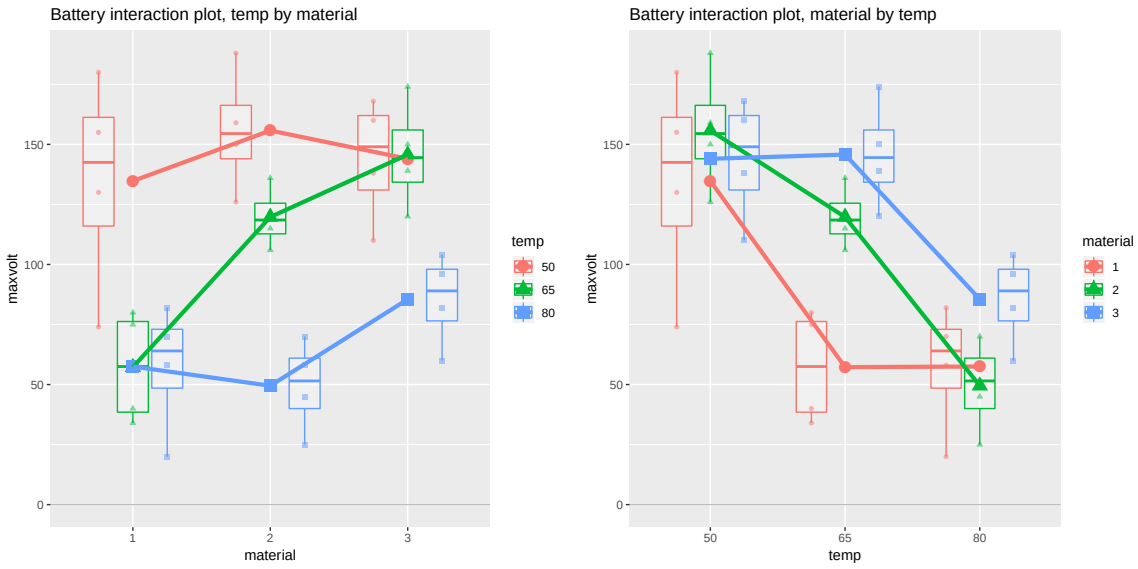
```
##   temp         m
## 1   50 144.83333
## 2   65 107.58333
## 3   80  64.16667
```

```
battery.mean.mt <- ddply(battery.long, .(material,temp), summarise, m = mean(maxvolt))
battery.mean.mt
```

```
##   material temp      m
## 1        1   50 134.75
## 2        1   65  57.25
## 3        1   80  57.50
## 4        2   50 155.75
## 5        2   65 119.75
## 6        2   80  49.50
## 7        3   50 144.00
## 8        3   65 145.75
## 9        3   80  85.50
```

```
# Interaction plots, ggplot
p <- ggplot(battery.long, aes(x = material, y = maxvolt, colour = temp, shape = temp))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                  , linetype = "solid", size = 0.2, alpha = 0.3)
```

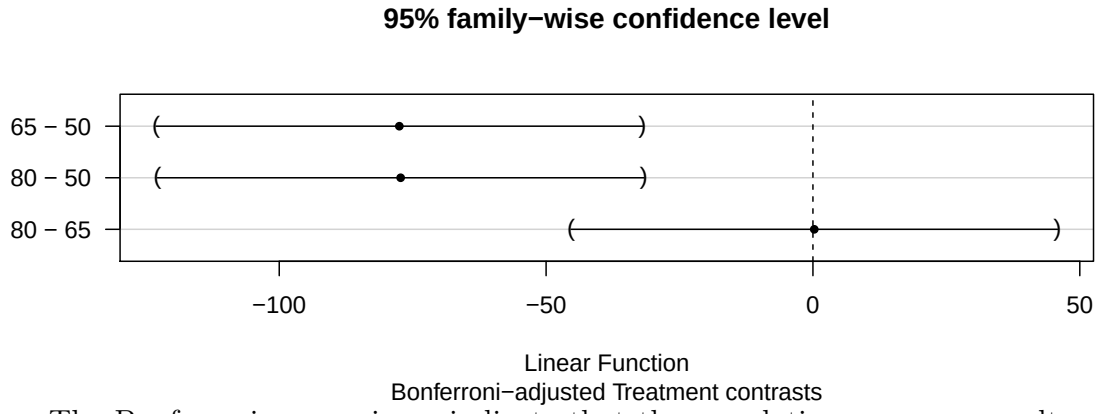
```
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = battery.mean.mt, aes(y = m), size = 4)
p <- p + geom_line(data = battery.mean.mt, aes(y = m, group = temp), size = 1.5)
p <- p + labs(title = "Battery interaction plot, temp by material")
print(p)

p <- ggplot(battery.long, aes(x = temp, y = maxvolt, colour = material, shape = material))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                    , linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = battery.mean.mt, aes(y = m), size = 4)
p <- p + geom_line(data = battery.mean.mt, aes(y = m, group = material), size = 1.5)
p <- p + labs(title = "Battery interaction plot, material by temp")
print(p)
```



The Bonferroni multiple comparisons may be inappropriate because of covariate interactions. That is, interactions make the main effects less meaningful (or their interpretation unclear) since the change in response when one factor is changed depends on what the second factor is.

The significant interaction between temperature and material implies that you can not directly conclude that batteries stored at 50 degrees have the highest average output regardless of the material. Nor can you directly conclude that material 3 has a higher average output than material 1 regardless of the temperature. You can only conclude that the differences are significant when averaged over the levels of the other factor.

```
# correcting over temp
glht.battery.t <- glht(aov(lm.m.m.t.mt), linfct = mcp(temp = "Tukey"))
## Warning in mcp2matrix(model, linfct = linfct):  covariate interactions found -- default
contrast might be inappropriate
summary(glht.battery.t, test = adjusted("bonferroni"))
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = lm.m.m.t.mt)
##
## Linear Hypotheses:
##              Estimate Std. Error t value Pr(>|t|)
## 65 - 50 == 0   -77.50      18.37  -4.218 0.000744 ***
## 80 - 50 == 0   -77.25      18.37  -4.204 0.000772 ***
## 80 - 65 == 0     0.25      18.37   0.014 1.000000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
# plot bonferroni-corrected difference intervals
plot(summary(glht.battery.t, test = adjusted("bonferroni"))
    , sub="Bonferroni-adjusted Treatment contrasts")
```



The Bonferroni comparisons indicate that the population mean max voltage for the three temperatures **averaged over material types** decreases as the temperature increases:

```
      Temp:    80      65      50
 Marg mean:  64.17 107.58 144.83
     Group: ------------- ------
```

However, you can compare materials at each temperature, and you can compare temperatures for each material. At individual temperatures, material 2 and 3 (or 1 and 2) might be significantly different even though they are not significantly different

when averaged over temperatures. For example, material 2 might produce a significantly higher average output than the other two material types at 50 degrees. This comparison of cell means is relevant if you are interested in using the batteries at 50 degrees! Comparing cell means is possible using "**lsmeans**", a point I will return to later.

5.2.5 Checking assumptions in a two-factor experiment

The normality and constant variance assumptions for a two-factor design can be visually checked using side-by-side boxplots (as was produced in the `ggplot()` interaction plots) and residual plots. Another useful tool for checking constant variances is to plot the sample deviations for each group against the group means.

Let us check the distributional assumptions for the insecticide experiment. The sampling design suggests that the independence assumptions are reasonable. The group sample sizes are small, so the residual plots are likely to be more informative than the side-by-side boxplots and the plot of the standard deviations.

The code below generates plots and summary statistics for the survival times. I used `ddply()` to store the means $\bar{y}_{ij}$ and standard deviations s_{ij} for the 12 treatment combinations. The diagnostic plots we've been using for `lm()` displays residual plots. Only the relevant output is presented.

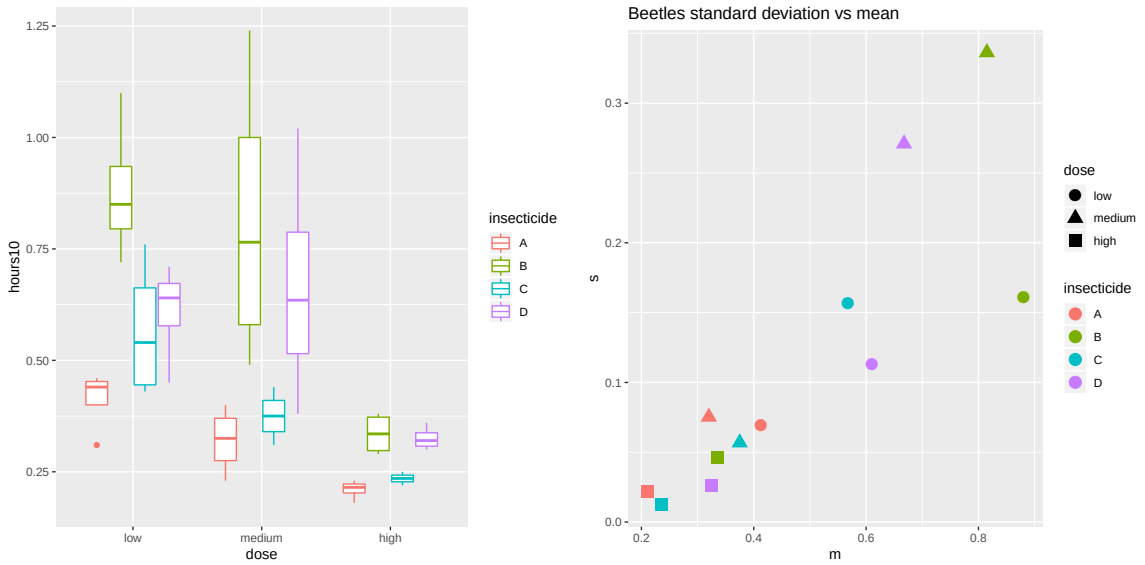
The set of box plots (each representing 4 points) for each insecticide/dose combination indicates both that means and standard deviations of treatments seem different. Also, there appears to be less variability for dose=3 (high) than for doses 1 and 2 in the table; the model assumes that variability is the same and does not depend on treatment. The plot of the standard deviation vs mean shows an increasing trend.

```
#### Example: Beetles, checking assumptions
# boxplots, ggplot
p <- ggplot(beetles.long, aes(x = dose, y = hours10, colour = insecticide))
p <- p + geom_boxplot()
print(p)

# mean vs sd plot
library(plyr)
# means and standard deviations for each dose/interaction cell
beetles.meansd.di <- ddply(beetles.long, .(dose,insecticide), summarise
                        , m = mean(hours10), s = sd(hours10))
beetles.meansd.di
```

```
##      dose insecticide      m          s
## 1     low           A 0.4125 0.06946222
## 2     low           B 0.8800 0.16083117
## 3     low           C 0.5675 0.15671099
## 4     low           D 0.6100 0.11284207
## 5  medium           A 0.3200 0.07527727
```

```
## 6  medium          B 0.8150 0.33630343
## 7  medium          C 0.3750 0.05686241
## 8  medium          D 0.6675 0.27097048
## 9    high          A 0.2100 0.02160247
## 10   high          B 0.3350 0.04654747
## 11   high          C 0.2350 0.01290994
## 12   high          D 0.3250 0.02645751
```

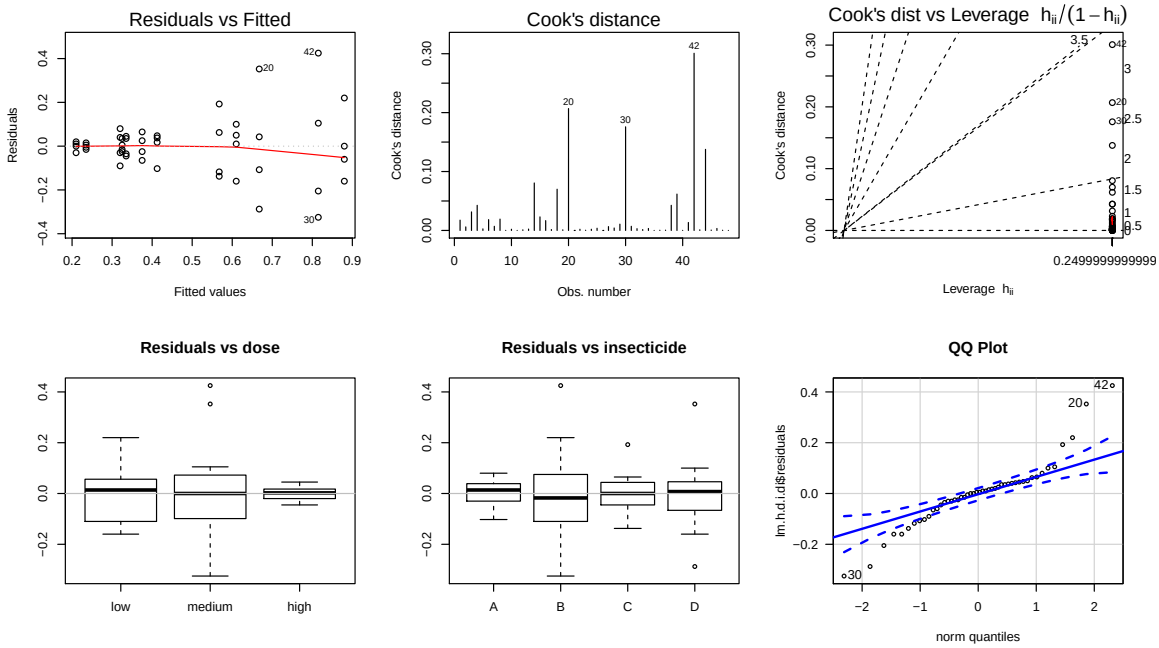
```
p <- ggplot(beetles.meansd.di, aes(x = m, y = s, shape = dose, colour = insecticide))
p <- p + geom_point(size=4)
p <- p + labs(title = "Beetles standard deviation vs mean")
print(p)
```



Diagnostic plots show the following features. The normal quantile plot shows an "S" shape rather than a straight line, suggesting the residuals are not normal, but have higher kurtosis (more peaky) than a normal distribution. The residuals vs the fitted (predicted) values show that the higher the predicted value the more variability (horn shaped). The plot of the Cook's distances indicate a few influential observations.

```
# interaction model
lm.h.d.i.di <- lm(hours10 ~ dose*insecticide, data = beetles.long)

# plot diagnostics
par(mfrow=c(2,3))
plot(lm.h.d.i.di, which = c(1,4,6))

plot(beetles.long$dose, lm.h.d.i.di$residuals, main="Residuals vs dose")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(beetles.long$insecticide, lm.h.d.i.di$residuals, main="Residuals vs insecticide")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.h.d.i.di$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1] 42 20 30
```

```
## residuals vs order of data
#plot(lm.h.d.i.dif$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Survival times are usually right skewed, with the spread or variability in the distribution increasing as the mean or median increases. Ideally, the distributions should be symmetric, normal, and the standard deviation should be fairly constant across groups.

The boxplots (note the ordering) and the plot of the s_{ij} against $\bar{y}_{ij}$ show the tendency for the spread to increase with the mean. This is reinforced by the residual plot, where the variability increases as the predicted values (the cell means under the two-factor interaction model) increase.

As noted earlier, the QQ-plot of the studentized residuals is better suited to examine normality here than the boxplots which are constructed from 4 observations. Not surprisingly, the boxplots do not suggest non-normality. Looking at the QQ-plot we clearly see evidence of non-normality.

5.2.6 A Remedy for Non-Constant Variance

A plot of cell standard deviations against the cell means is sometimes used as a diagnostic tool for suggesting transformations of the data. Here are some suggestions for transforming non-negative measurements to make the variability independent of the mean (i.e., stabilize the variance). The transformations also tend to reduce skewness, if present (and may induce skewness if absent!). As an aside, some statisticians prefer

to plot the IQR against the median to get a more robust view of the dependence of spread on typical level because s_{ij} and $\bar{y}_{ij}$ are sensitive to outliers.

1. If s_{ij} increases linearly with $\bar{y}_{ij}$, use a **log** transformation of the response.
2. If s_{ij} increases as a quadratic function of $\bar{y}_{ij}$, use a reciprocal (**inverse**) transformation of the response.
3. If s_{ij} increases as a square root function of $\bar{y}_{ij}$, use a **square root** transformation of the response.
4. If s_{ij} is roughly independent of $\bar{y}_{ij}$, do not transform the response. This idea does not require the response to be non-negative!

A logarithmic transformation or a reciprocal (inverse) transformation of the survival times might help to stabilize the variance. The survival time distributions are fairly symmetric, so these nonlinear transformations may destroy the symmetry. As a first pass, I will consider the reciprocal transformation because the inverse survival time has a natural interpretation as the dying rate. For example, if you survive 2 hours, then 1/2 is the proportion of your remaining lifetime expired in the next hour. The unit of time is actually 10 hours, so 0.1/time is the actual rate. The 0.1 scaling factor has no effect on the analysis provided you appropriately rescale the results on the mean responses.

Create the **rate** variable.

```
#### Example: Beetles, non-constant variance
# create the rate variable (1/hours10)
beetles.long$rate <- 1/beetles.long$hours10
```

Redo the analysis replacing `hours10` by `rate`.

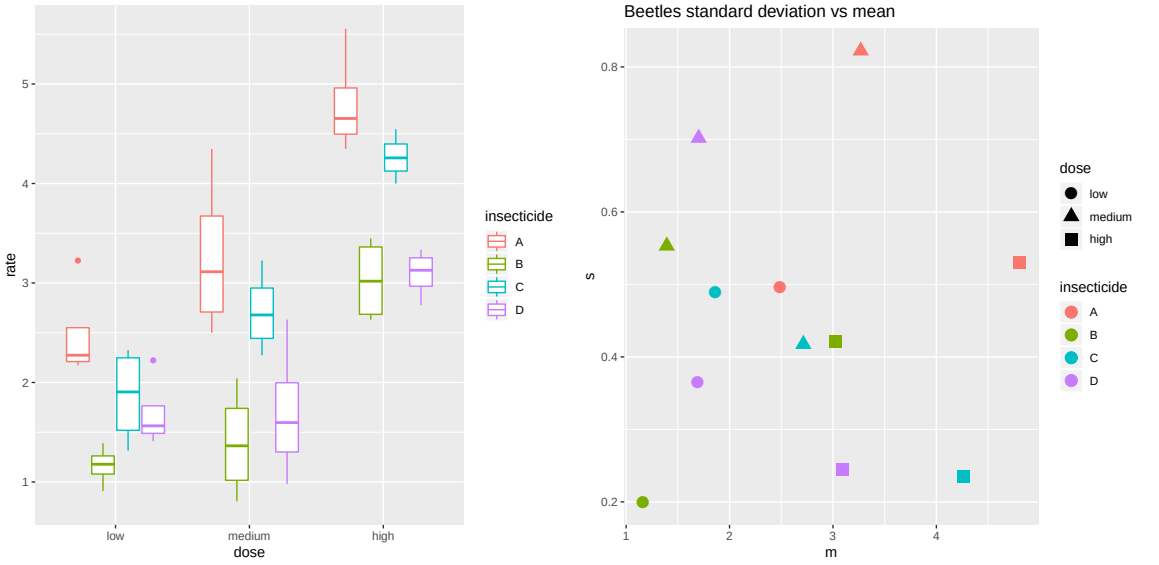
The standard deviations of rate appear much more similar than those of time did.

```
# boxplots, ggplot
p <- ggplot(beetles.long, aes(x = dose, y = rate, colour = insecticide))
p <- p + geom_boxplot()
print(p)

# mean vs sd plot
library(plyr)
# means and standard deviations for each dose/interaction cell
beetles.meansd.di.rate <- ddply(beetles.long, .(dose,insecticide), summarise
                        , m = mean(rate), s = sd(rate))
beetles.meansd.di.rate
```

```
##      dose insecticide        m         s
## 1     low           A 2.486881 0.4966627
## 2     low           B 1.163464 0.1994976
## 3     low           C 1.862724 0.4893774
## 4     low           D 1.689682 0.3647127
## 5  medium           A 3.268470 0.8223269
## 6  medium           B 1.393392 0.5531885
## 7  medium           C 2.713919 0.4175138
```

```
## 8  medium           D 1.701534 0.7019053
## 9    high           A 4.802685 0.5296355
## 10   high           B 3.028973 0.4214358
## 11   high           C 4.264987 0.2348115
## 12   high           D 3.091805 0.2440546
```

```
p <- ggplot(beetles.meansd.di.rate, aes(x = m, y = s, shape = dose
          , colour = insecticide))
p <- p + geom_point(size=4)
p <- p + labs(title = "Beetles standard deviation vs mean")
print(p)
```



The profile plots and ANOVA table indicate that the main effects are significant but the interaction is not.

```
library(plyr)
# Calculate the cell means for each (dose, insecticide) combination
beetles.mean    <- ddply(beetles.long, .(), summarise, m = mean(rate))
beetles.mean
```

```
##    .id        m
## 1 <NA> 2.622376
```

```
beetles.mean.d  <- ddply(beetles.long, .(dose), summarise, m = mean(rate))
beetles.mean.d
```

```
##     dose        m
## 1    low 1.800688
## 2 medium 2.269329
## 3   high 3.797112
```

```
beetles.mean.i  <- ddply(beetles.long, .(insecticide), summarise, m = mean(rate))
beetles.mean.i
```

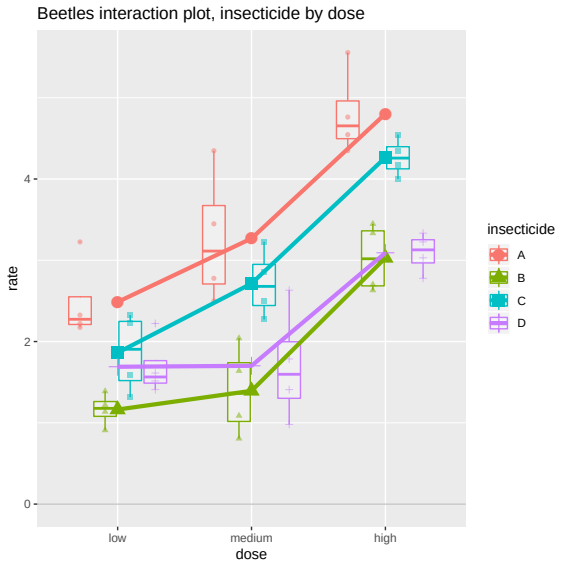
```
##   insecticide        m
## 1           A 3.519345
## 2           B 1.861943
## 3           C 2.947210
## 4           D 2.161007
```

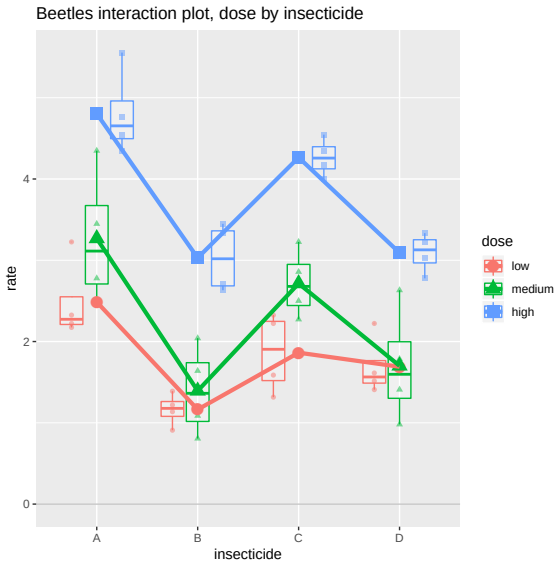
```
beetles.mean.di <- ddply(beetles.long, .(dose,insecticide), summarise, m = mean(rate))
beetles.mean.di
```

```
##      dose insecticide        m
## 1     low           A 2.486881
## 2     low           B 1.163464
## 3     low           C 1.862724
## 4     low           D 1.689682
## 5  medium           A 3.268470
```

```
## 6  medium          B 1.393392
## 7  medium          C 2.713919
## 8  medium          D 1.701534
## 9    high          A 4.802685
## 10   high          B 3.028973
## 11   high          C 4.264987
## 12   high          D 3.091805
```

```
# Interaction plots, ggplot

p <- ggplot(beetles.long, aes(x = dose, y = rate, colour = insecticide, shape = insecticide))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                , linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = beetles.mean.di, aes(y = m), size = 4)
p <- p + geom_line(data = beetles.mean.di, aes(y = m, group = insecticide), size = 1.5)
p <- p + labs(title = "Beetles interaction plot, insecticide by dose")
print(p)

p <- ggplot(beetles.long, aes(x = insecticide, y = rate, colour = dose, shape = dose))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                , linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = beetles.mean.di, aes(y = m), size = 4)
p <- p + geom_line(data = beetles.mean.di, aes(y = m, group = dose), size = 1.5)
p <- p + labs(title = "Beetles interaction plot, dose by insecticide")
print(p)
```






```
lm.r.d.i.di <- lm(rate ~ dose*insecticide, data = beetles.long) # equivalent
library(car)
Anova(lm.r.d.i.di, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: rate
##                    Sum Sq Df  F value    Pr(>F)
## (Intercept)      24.7383  1 103.0395 4.158e-12 ***
## dose             11.1035  2  23.1241 3.477e-07 ***
## insecticide       3.5723  3   4.9598  0.005535 **
## dose:insecticide  1.5708  6   1.0904  0.386733
## Residuals         8.6431 36
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.r.d.i.di)
##
## Call:
## lm(formula = rate ~ dose * insecticide, data = beetles.long)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.76847 -0.29642 -0.06914  0.25458  1.07936
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               2.48688    0.24499  10.151 4.16e-12 ***
## dosemedium                0.78159    0.34647   2.256 0.030252 *
## dosehigh                  2.31580    0.34647   6.684 8.56e-08 ***
## insecticideB             -1.32342    0.34647  -3.820 0.000508 ***
## insecticideC             -0.62416    0.34647  -1.801 0.080010 .
## insecticideD             -0.79720    0.34647  -2.301 0.027297 *
## dosemedium:insecticideB -0.55166    0.48999  -1.126 0.267669
## dosehigh:insecticideB   -0.45030    0.48999  -0.919 0.364213
## dosemedium:insecticideC  0.06961    0.48999   0.142 0.887826
## dosehigh:insecticideC    0.08646    0.48999   0.176 0.860928
## dosemedium:insecticideD -0.76974    0.48999  -1.571 0.124946
## dosehigh:insecticideD   -0.91368    0.48999  -1.865 0.070391 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.49 on 36 degrees of freedom
## Multiple R-squared:  0.8681,Adjusted R-squared:  0.8277
## F-statistic: 21.53 on 11 and 36 DF,  p-value: 1.289e-12
```

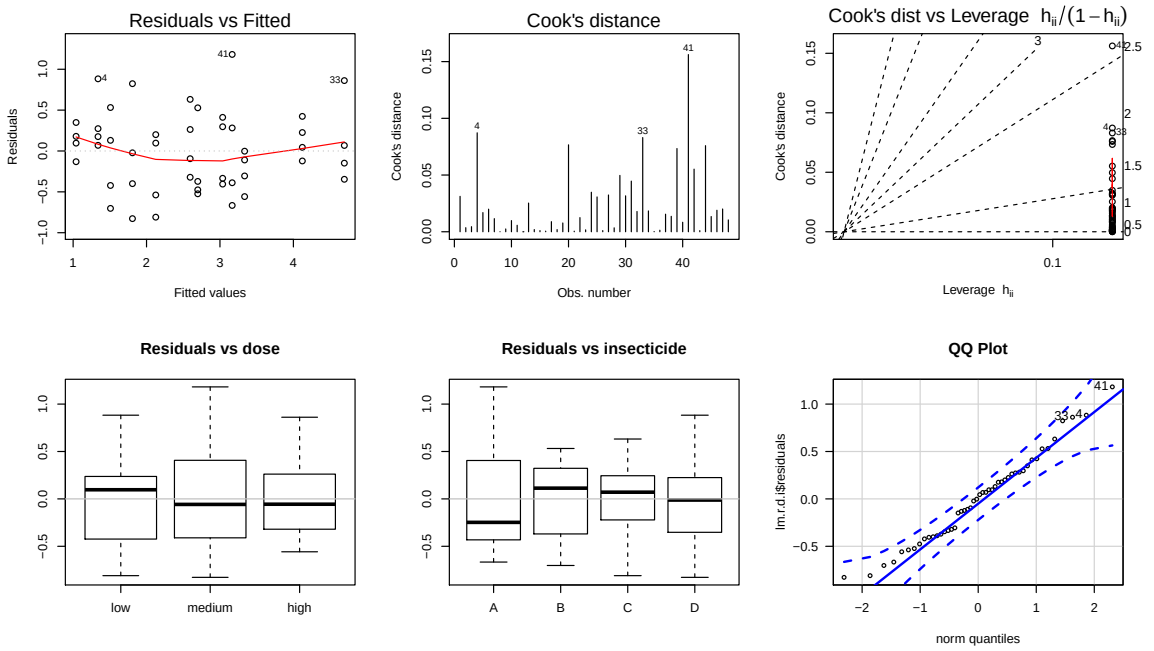
Drop the nonsignificant interaction term.

```
lm.r.d.i <- update(lm.r.d.i.di, ~ . - dose:insecticide)
library(car)
Anova(lm.r.d.i, type=3)
## Anova Table (Type III tests)
##
## Response: rate
##             Sum Sq Df F value    Pr(>F)
## (Intercept) 58.219  1 239.399 < 2.2e-16 ***
## dose        34.877  2  71.708 2.865e-14 ***
## insecticide 20.414  3  27.982 4.192e-10 ***
## Residuals   10.214 42
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.r.d.i)
```

```
##
## Call:
## lm(formula = rate ~ dose + insecticide, data = beetles.long)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.82757 -0.37619  0.02116  0.27568  1.18153
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.6977     0.1744  15.473  < 2e-16 ***
## dosemedium     0.4686     0.1744   2.688  0.01026 *
## dosehigh       1.9964     0.1744  11.451 1.69e-14 ***
## insecticideB  -1.6574     0.2013  -8.233 2.66e-10 ***
## insecticideC  -0.5721     0.2013  -2.842  0.00689 **
## insecticideD  -1.3583     0.2013  -6.747 3.35e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4931 on 42 degrees of freedom
## Multiple R-squared:  0.8441,Adjusted R-squared:  0.8255
## F-statistic: 45.47 on 5 and 42 DF,  p-value: 6.974e-16
```

Unlike the original analysis, the residual plots do not show any gross deviations from assumptions. Also, no case seems relatively influential.

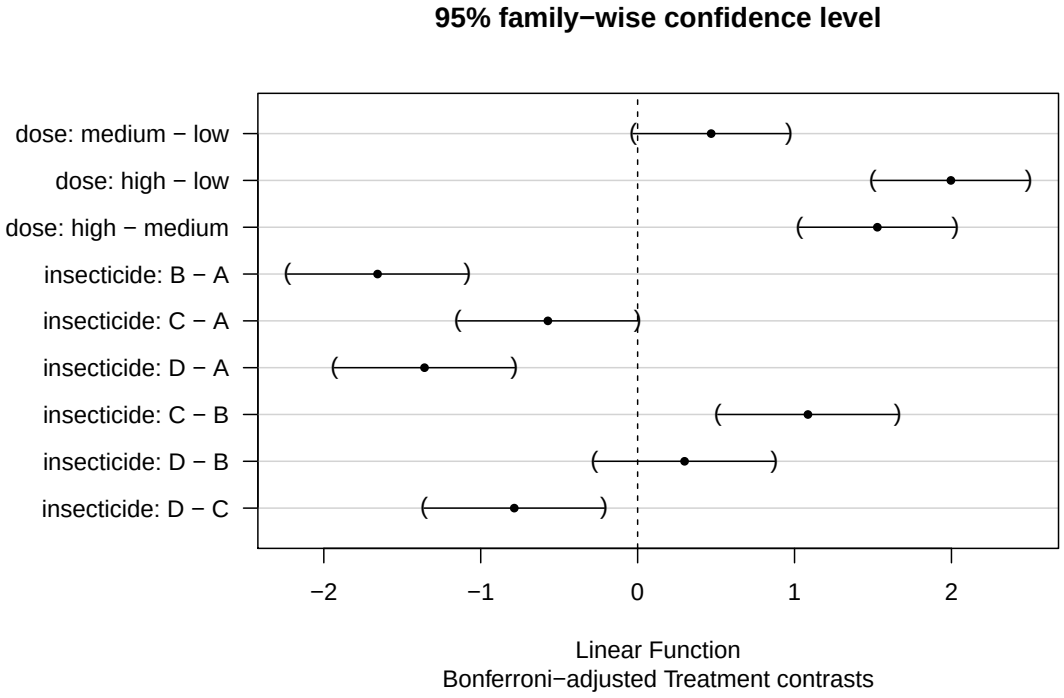
```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.r.d.i, which = c(1,4,6))

plot(beetles.long$dose, lm.r.d.i$residuals, main="Residuals vs dose")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(beetles.long$insecticide, lm.r.d.i$residuals, main="Residuals vs insecticide")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.r.d.i$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1] 41  4 33
## residuals vs order of data
#plot(lm.r.d.i$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Bonferroni multiple comparisons imply differences about the mean rates.

```
# Testing multiple factors is of interest here.
# Note that the code below corrects the p-values
#   for all the tests done for both factors together,
#   that is, the Bonferroni-corrected significance level is (alpha / (d + i))
#   where d = number of dose        comparisons
#   and   i = number of insecticide comparisons.

# correcting over dose and insecticide
glht.beetle.di.rate <- glht(aov(lm.r.d.i), linfct = mcp(dose = "Tukey"
                                                        , insecticide = "Tukey"))
summary(glht.beetle.di.rate, test = adjusted("bonferroni"))
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = lm.r.d.i)
##
## Linear Hypotheses:
##                           Estimate Std. Error t value Pr(>|t|)
## dose: medium - low == 0    0.4686     0.1744   2.688  0.09236 .
## dose: high - low == 0      1.9964     0.1744  11.451 1.52e-13 ***
## dose: high - medium == 0   1.5278     0.1744   8.763 4.46e-10 ***
## insecticide: B - A == 0   -1.6574     0.2013  -8.233 2.39e-09 ***
```

```
## insecticide: C - A == 0   -0.5721     0.2013  -2.842  0.06203 .
## insecticide: D - A == 0   -1.3583     0.2013  -6.747 3.01e-07 ***
## insecticide: C - B == 0    1.0853     0.2013   5.391 2.67e-05 ***
## insecticide: D - B == 0    0.2991     0.2013   1.485  1.00000
## insecticide: D - C == 0   -0.7862     0.2013  -3.905  0.00302 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
par(mfrow=c(1,1))
# plot the summary
  op <- par(no.readonly = TRUE) # the whole list of settable par's.
  # make wider left margin to fit contrast labels
  par(mar = c(5, 10, 4, 2) + 0.1)  # order is c(bottom, left, top, right)
# plot bonferroni-corrected difference intervals
plot(summary(glht.beetle.di.rate, test = adjusted("bonferroni"))
    , sub="Bonferroni-adjusted Treatment contrasts")
  par(op) # reset plotting options
```



Comments on the Two Analyses of Survival Times

The effects of the transformation are noticeable. For example, the comparisons among doses and insecticides are less sensitive (differences harder to distinguish) on the original scale (look at the Bonferroni groupings). A comparison of the interaction p-values and profile plots for the two analyses suggests that the transformation eliminates much

of the observed interaction between the main effects. Although the interaction in the original analysis was not significant at the 10% level (p-value=0.112), the small sample sizes suggest that power for detecting interaction might be low. To be on the safe side, one might interpret the main effects in the original analysis as if an interaction were present. This need appears to be less pressing with the rates.

The statistical assumptions are reasonable for an analysis of the rates. I think that the simplicity of the main effects interpretation is a strong motivating factor for preferring the analysis of the transformed data to the original analysis. You might disagree, especially if you believe that the original time scale is most relevant for analysis.

Given the suitability of the inverse transformation, I did not consider the logarithmic transformation.

5.3 Multiple comparisons: balanced (means) vs unbalanced (lsmeans)

The **lsmeans** provides a way to compare cell means (combinations of factors), something that is not possible directly with `glht()`, which compares marginal means.

Using the battery example, we compare the multiple comparison methods using means (`glht()`) and lsmeans[2] (`lsmeans()`). When there are only main effects, the two methods agree.

```
#### Multiple comparisons
#### Example: Battery
# fit additive (main effects) model (same as before)
lm.m.m.t <- lm(maxvolt ~ material + temp, data = battery.long)

### comparing means (must be balanced or have only one factor)
# correcting over temp
glht.battery.t <- glht(aov(lm.m.m.t), linfct = mcp(temp = "Tukey"))
summary(glht.battery.t, test = adjusted("bonferroni"))
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = lm.m.m.t)
##
## Linear Hypotheses:
##              Estimate Std. Error t value Pr(>|t|)
```

[2] `lsmeans` is a package written by Russell V. Lenth, PhD of UNM 1975, and well-known for his online power calculators.

```
## 65 - 50 == 0   -37.25      12.24  -3.044  0.01417 *
## 80 - 50 == 0   -80.67      12.24  -6.593 6.89e-07 ***
## 80 - 65 == 0   -43.42      12.24  -3.548  0.00377 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
```

```
### comparing lsmeans (may be unbalanced)
library(lsmeans)
```

```
## The 'lsmeans' package is being deprecated.
## Users are encouraged to switch to 'emmeans'.
## See help('transition') for more information, including how
## to convert 'lsmeans' objects and scripts to work with 'emmeans'.
```

```
## compare levels of main effects
# temp
lsmeans(lm.m.m.t, list(pairwise ~ temp), adjust = "bonferroni")
```

```
## $`lsmeans of temp`
##  temp    lsmean      SE df  lower.CL upper.CL
##  50   144.83333 8.65164 31 127.18820 162.4785
##  65   107.58333 8.65164 31  89.93820 125.2285
##  80    64.16667 8.65164 31  46.52153  81.8118
##
## Results are averaged over the levels of: material
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast estimate       SE df t.ratio p.value
##  50 - 65  37.25000 12.23527 31   3.044  0.0142
##  50 - 80  80.66667 12.23527 31   6.593  <.0001
##  65 - 80  43.41667 12.23527 31   3.548  0.0038
##
## Results are averaged over the levels of: material
## P value adjustment: bonferroni method for 3 tests
```

When there are model interactions, the comparisons of the main effects are inappropriate, and give different results depending on the method of comparison.

```
# fit interaction model (same as before)
lm.m.m.t.mt <- lm(maxvolt ~ material*temp, data = battery.long)

### comparing means (must be balanced or have only one factor)
# correcting over temp
glht.battery.t <- glht(aov(lm.m.m.t.mt), linfct = mcp(temp = "Tukey"))
```

```
## Warning in mcp2matrix(model, linfct = linfct):  covariate interactions found -- default
contrast might be inappropriate
```

```
summary(glht.battery.t, test = adjusted("bonferroni"))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
```

```
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = lm.m.m.t.mt)
##
## Linear Hypotheses:
##              Estimate Std. Error t value Pr(>|t|)
## 65 - 50 == 0   -77.50      18.37  -4.218 0.000744 ***
## 80 - 50 == 0   -77.25      18.37  -4.204 0.000772 ***
## 80 - 65 == 0     0.25      18.37   0.014 1.000000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- bonferroni method)
```

```
### comparing lsmeans (may be unbalanced)
library(lsmeans)
## compare levels of main effects
# temp
lsmeans(lm.m.m.t.mt, list(pairwise ~ temp), adjust = "bonferroni")
```

```
## NOTE: Results may be misleading due to involvement in interactions
## $`lsmeans of temp`
##  temp    lsmean       SE df  lower.CL  upper.CL
##  50   144.83333 7.501183 27 129.44218 160.22449
##  65   107.58333 7.501183 27  92.19218 122.97449
##  80    64.16667 7.501183 27  48.77551  79.55782
##
## Results are averaged over the levels of: material
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast estimate       SE df t.ratio p.value
##  50 - 65  37.25000 10.60827 27   3.511  0.0048
##  50 - 80  80.66667 10.60827 27   7.604  <.0001
##  65 - 80  43.41667 10.60827 27   4.093  0.0010
##
## Results are averaged over the levels of: material
## P value adjustment: bonferroni method for 3 tests
```

When there are model interactions and you want to compare cell means, levels of one factor at each level of another factor separately, then you must use `lsmeans()`.

```
# fit interaction model (same as before)
lm.m.m.t.mt <- lm(maxvolt ~ material*temp, data = battery.long)

### comparing lsmeans (may be unbalanced)
library(lsmeans)
## compare levels of one factor at each level of another factor separately
# material at levels of temp
lsmeans(lm.m.m.t.mt, list(pairwise ~ material | temp), adjust = "bonferroni")
```

```
## $`lsmeans of material | temp`
## temp = 50:
##  material lsmean       SE df  lower.CL  upper.CL
##  1        134.75 12.99243 27 108.09174 161.40826
##  2        155.75 12.99243 27 129.09174 182.40826
##  3        144.00 12.99243 27 117.34174 170.65826
##
## temp = 65:
##  material lsmean       SE df  lower.CL  upper.CL
##  1         57.25 12.99243 27  30.59174  83.90826
##  2        119.75 12.99243 27  93.09174 146.40826
##  3        145.75 12.99243 27 119.09174 172.40826
##
## temp = 80:
##  material lsmean       SE df  lower.CL  upper.CL
##  1         57.50 12.99243 27  30.84174  84.15826
##  2         49.50 12.99243 27  22.84174  76.15826
##  3         85.50 12.99243 27  58.84174 112.15826
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast, temp | temp`
## temp = 50:
##  contrast estimate       SE df t.ratio p.value
##  1 - 2      -21.00 18.37407 27  -1.143  0.7893
##  1 - 3       -9.25 18.37407 27  -0.503  1.0000
##  2 - 3       11.75 18.37407 27   0.639  1.0000
##
## temp = 65:
##  contrast estimate       SE df t.ratio p.value
##  1 - 2      -62.50 18.37407 27  -3.402  0.0063
##  1 - 3      -88.50 18.37407 27  -4.817  0.0001
##  2 - 3      -26.00 18.37407 27  -1.415  0.5055
##
## temp = 80:
##  contrast estimate       SE df t.ratio p.value
##  1 - 2        8.00 18.37407 27   0.435  1.0000
##  1 - 3      -28.00 18.37407 27  -1.524  0.4175
##  2 - 3      -36.00 18.37407 27  -1.959  0.1814
##
## P value adjustment: bonferroni method for 3 tests
# temp at levels of material
lsmeans(lm.m.m.t.mt, list(pairwise ~ temp | material), adjust = "bonferroni")
## $`lsmeans of temp | material`
## material = 1:
##  temp lsmean       SE df  lower.CL  upper.CL
##  50   134.75 12.99243 27 108.09174 161.40826
##  65    57.25 12.99243 27  30.59174  83.90826
```

```
##  80    57.50 12.99243 27  30.84174  84.15826
##
## material = 2:
##  temp lsmean       SE df  lower.CL  upper.CL
##  50   155.75 12.99243 27 129.09174 182.40826
##  65   119.75 12.99243 27  93.09174 146.40826
##  80    49.50 12.99243 27  22.84174  76.15826
##
## material = 3:
##  temp lsmean       SE df  lower.CL  upper.CL
##  50   144.00 12.99243 27 117.34174 170.65826
##  65   145.75 12.99243 27 119.09174 172.40826
##  80    85.50 12.99243 27  58.84174 112.15826
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast, material | material`
## material = 1:
##  contrast estimate       SE df t.ratio p.value
##  50 - 65     77.50 18.37407 27   4.218  0.0007
##  50 - 80     77.25 18.37407 27   4.204  0.0008
##  65 - 80     -0.25 18.37407 27  -0.014  1.0000
##
## material = 2:
##  contrast estimate       SE df t.ratio p.value
##  50 - 65     36.00 18.37407 27   1.959  0.1814
##  50 - 80    106.25 18.37407 27   5.783  <.0001
##  65 - 80     70.25 18.37407 27   3.823  0.0021
##
## material = 3:
##  contrast estimate       SE df t.ratio p.value
##  50 - 65     -1.75 18.37407 27  -0.095  1.0000
##  50 - 80     58.50 18.37407 27   3.184  0.0109
##  65 - 80     60.25 18.37407 27   3.279  0.0086
##
## P value adjustment: bonferroni method for 3 tests
```

Finally, an important point demonstrated in the next section is that the cell and marginal averages given by the **means** and **lsmeans** methods agree here for the main effects model because the design is balanced. For unbalanced designs with two or more factors, **lsmeans** and **means** compute different averages. I will argue that **lsmeans** are the appropriate averages for unbalanced analyses. You should use the **means** statement with caution — it is OK for balanced or unbalanced one-factor designs, and for the balanced two-factor designs (including the RB) that we have discussed.

5.4 Unbalanced Two-Factor Designs and Analysis

Sample sizes are usually unequal, or **unbalanced**, for the different treatment groups in an experiment. Although this has no consequence on the specification of a model, you have to be more careful with the **analysis of unbalanced experiments** that have two or more factors. With unbalanced designs, the Type I and Type III SS differ, as do the main effect averages given by **means** and **lsmeans**. Inferences might critically depend on which summaries are used. I will use the following example to emphasize the differences between the types of SS and averages.

5.4.1 Example: Rat insulin

The experiment consists of measuring insulin levels in rats a certain length of time after a fixed dose of insulin was injected into their jugular or portal veins. This is a two-factor study with two vein types (jugular, portal) and three time levels (0, 30, and 60). An unusual feature of this experiment is that the rats used in the six vein and time combinations are distinct. I will fit a two-factor interaction model, which assumes that the responses are independent within and across treatments (other models may be preferred). The design is **unbalanced**, with sample sizes varying from 3 to 12.

An alternative experimental design might randomly assign rats to the two vein groups, and then measure the insulin levels of each rat at the three time points. Depending on the questions of interest, you could compare veins using a one-way MANOVA, or use other multivariate techniques that allow correlated responses within rats.

```
#### Example: Rat insulin
rat <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch05_ratinsulin.dat"
                  , header = TRUE)
# make time a factor variable and label the levels
rat$time<- factor(rat$time)
str(rat)
## 'data.frame': 48 obs. of  3 variables:
##  $ vein   : Factor w/ 2 levels "j","p": 1 1 1 1 1 1 1 1 1 1 ...
##  $ time   : Factor w/ 3 levels "0","30","60": 1 1 1 1 1 2 2 2 2 2 ...
##  $ insulin: int  18 36 12 24 43 61 116 63 132 68 ...
head(rat, 3)
##   vein time insulin
## 1    j    0      18
## 2    j    0      36
## 3    j    0      12
tail(rat, 3)
##    vein time insulin
## 46    p   60     105
## 47    p   60      71
```

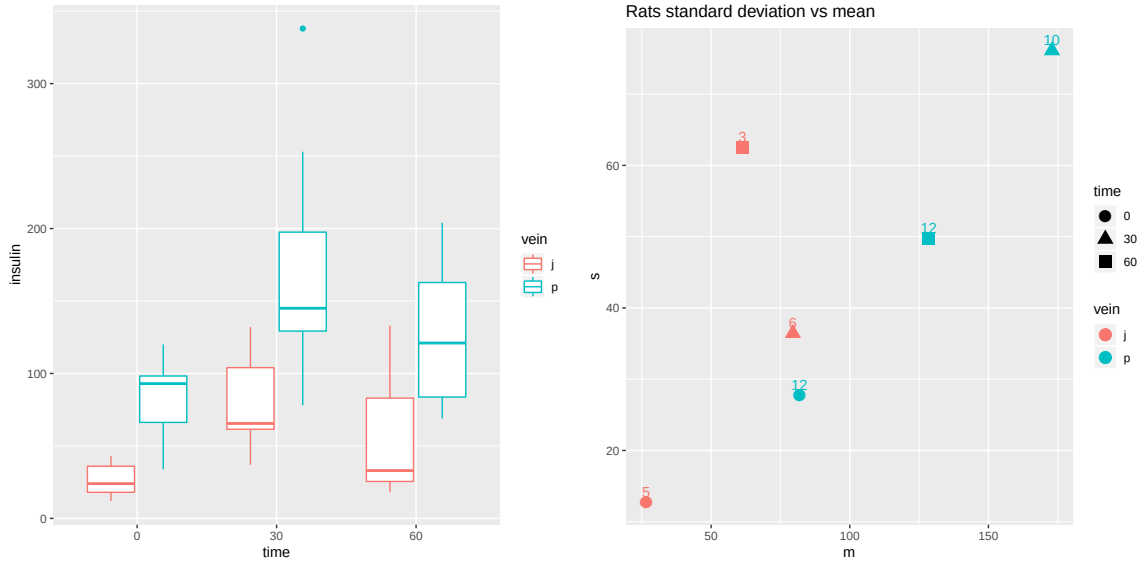
```
## 48    p   60      83
```

It appears the standard deviation increases with the mean.

```
# boxplots, ggplot
p <- ggplot(rat, aes(x = time, y = insulin, colour = vein))
p <- p + geom_boxplot()
print(p)

# mean vs sd plot
library(plyr)
# means and standard deviations for each time/interaction cell
rat.meansd.tv <- ddply(rat, .(time,vein), summarise
                       , m = mean(insulin), s = sd(insulin), n = length(insulin))
rat.meansd.tv
```

```
##   time vein         m        s  n
## 1    0    j  26.60000 12.75931  5
## 2    0    p  81.91667 27.74710 12
## 3   30    j  79.50000 36.44585  6
## 4   30    p 172.90000 76.11753 10
## 5   60    j  61.33333 62.51666  3
## 6   60    p 128.50000 49.71830 12
```

```
p <- ggplot(rat.meansd.tv, aes(x = m, y = s, shape = time, colour = vein, label=n))
p <- p + geom_point(size=4)
# labels are sample sizes
p <- p + geom_text(hjust = 0.5, vjust = -0.5)
p <- p + labs(title = "Rats standard deviation vs mean")
print(p)
```



We take the log of insulin to correct the problem. The variances are more constant now, except for one sample with only 3 observations which has a larger standard

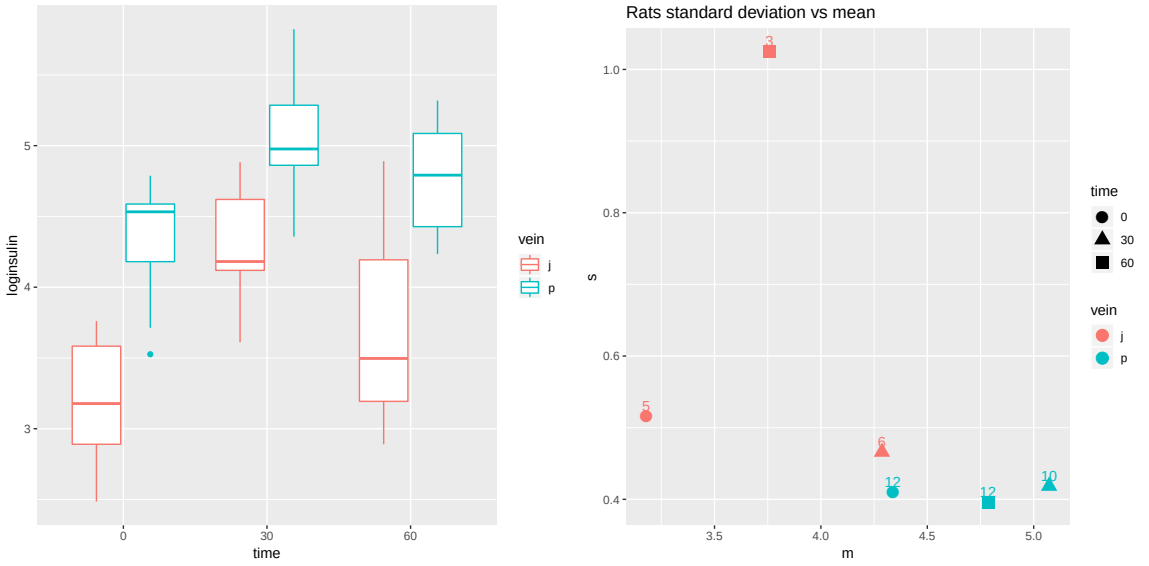
deviation than the others, but because this is based on such a small sample size, it's not of much concern.

```
rat$loginsulin <- log(rat$insulin)
```

```
# boxplots, ggplot
p <- ggplot(rat, aes(x = time, y = loginsulin, colour = vein))
p <- p + geom_boxplot()
print(p)

# mean vs sd plot
library(plyr)
# means and standard deviations for each time/interaction cell
rat.meansd.tv <- ddply(rat, .(time,vein), summarise
                       , m = mean(loginsulin)
                       , s = sd(loginsulin)
                       , n = length(loginsulin))
rat.meansd.tv
```

```
##   time vein        m         s  n
## 1    0    j 3.179610 0.5166390  5
## 2    0    p 4.338230 0.4096427 12
## 3   30    j 4.286804 0.4660571  6
## 4   30    p 5.072433 0.4185221 10
## 5   60    j 3.759076 1.0255165  3
## 6   60    p 4.785463 0.3953252 12
```

```
p <- ggplot(rat.meansd.tv, aes(x = m, y = s, shape = time, colour = vein, label=n))
p <- p + geom_point(size=4)
# labels are sample sizes
p <- p + geom_text(hjust = 0.5, vjust = -0.5)
p <- p + labs(title = "Rats standard deviation vs mean")
print(p)
```



Type I and Type III SS

We can request ANOVA tables including Type I or Type III SS[3] for each effect in a model. The Type I SS is the **sequential reduction** in Error SS achieved when an effect is added to a model that includes only the prior effects listed in the **model** statement. The Type III SS are more difficult to define explicitly, but they roughly correspond to the reduction in Error SS achieved when an effect is added **last** to the model.

Type I SS and Type III SS are equal for balanced designs and for one-way ANOVA, but are typically different for unbalanced designs, where there is no unique way to define the SS for an effect. The problem here is similar to multiple regression, where the SS for a predictor X is the decrease in Residual SS when X is added to a model. This SS is not unique because the change in the Residual SS depends on which predictors are included in the model prior to X. In a regression analysis, the standard tests for effects in a model are based on Type III SS and not on the Type I SS.

For the insulin analysis, the Type I and Type III interaction SS are identical because this effect was added last to the **model** statement. The Type I and III SS for the main effects are not equal. Also note that the Type I SS for the main effects and interaction add to the model SS, but the Type III SS do not.

Looking at the output, we see the Type I and Type III SS are different, except for the interaction term.

```
lm.i.t.v.tv <- lm(loginsulin ~ time*vein, data = rat
                  , contrasts = list(time = contr.sum, vein = contr.sum))
```

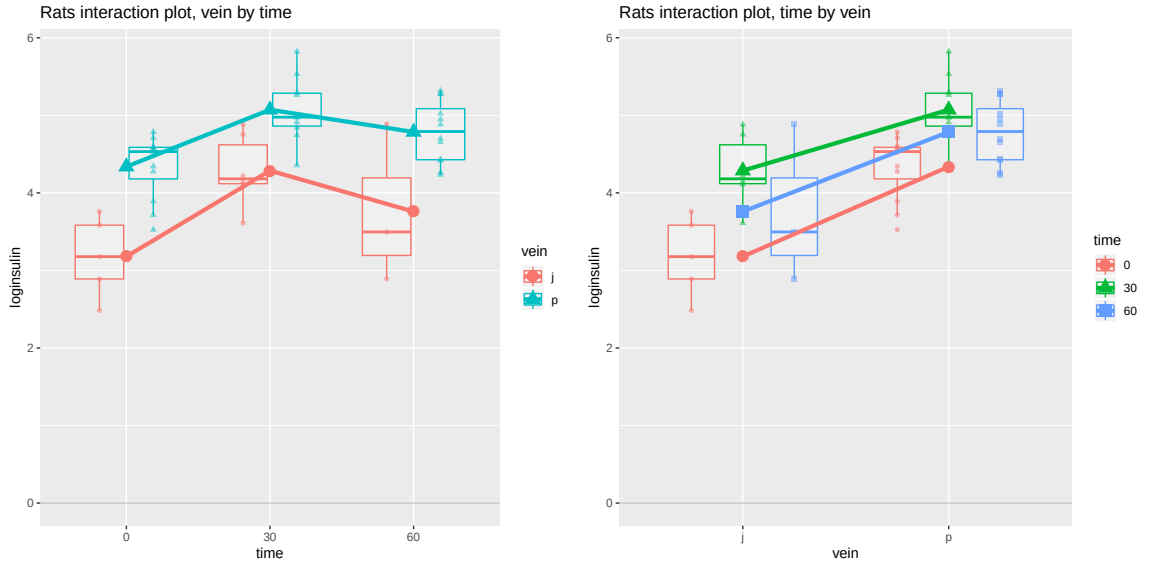
[3]For the ugly details, see `http://goanna.cs.rmit.edu.au/~fscholer/anova.php`.

```
## CRITICAL!!!  Unbalanced design warning.
##   The contrast statement above must be included identifying
##   each main effect with "contr.sum" in order for the correct
##   Type III SS to be computed.
##   See http://goanna.cs.rmit.edu.au/~fscholer/anova.php
library(car)
# type I   SS  (intercept SS not shown)
summary(aov(lm.i.t.v.tv))
##             Df Sum Sq Mean Sq F value   Pr(>F)
## time         2  5.450   2.725   12.18 6.74e-05 ***
## vein         1  9.321   9.321   41.66 8.82e-08 ***
## time:vein    2  0.259   0.130    0.58    0.565
## Residuals   42  9.399   0.224
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# type III SS
Anova(lm.i.t.v.tv, type=3)
## Anova Table (Type III tests)
##
## Response: loginsulin
##             Sum Sq Df   F value    Pr(>F)
## (Intercept) 668.54  1 2987.5842 < 2.2e-16 ***
## time          6.18  2   13.7996 2.475e-05 ***
## vein          9.13  1   40.7955 1.101e-07 ***
## time:vein     0.26  2    0.5797    0.5645
## Residuals     9.40 42
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Because the profile plot lines all seem parallel, and because of the interaction Type III SS p-value above, it appears there is not sufficient evidence for a vein-by-time interaction. For now we'll keep the interaction in the model for the purpose of discussing differences between means and lsmeans and Type I and Type III SS.

```
# calculate means for plot
library(plyr)
rat.mean.tv <- ddply(rat, .(time,vein), summarise, m = mean(loginsulin))

# Interaction plots, ggplot
p <- ggplot(rat, aes(x = time, y = loginsulin, colour = vein, shape = vein))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                  , linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = rat.mean.tv, aes(y = m), size = 4)
p <- p + geom_line(data = rat.mean.tv, aes(y = m, group = vein), size = 1.5)
p <- p + labs(title = "Rats interaction plot, vein by time")
print(p)
```

```
p <- ggplot(rat, aes(x = vein, y = loginsulin, colour = time, shape = time))
p <- p + geom_hline(aes(yintercept = 0), colour = "black"
                    , linetype = "solid", size = 0.2, alpha = 0.3)
p <- p + geom_boxplot(alpha = 0.25, outlier.size=0.1)
p <- p + geom_point(alpha = 0.5, position=position_dodge(width=0.75))
p <- p + geom_point(data = rat.mean.tv, aes(y = m), size = 4)
p <- p + geom_line(data = rat.mean.tv, aes(y = m, group = time), size = 1.5)
p <- p + labs(title = "Rats interaction plot, time by vein")
print(p)
```



Means versus lsmeans

The **lsmeans** (sometimes called **adjusted means**) for a single factor is an arithmetic average of cell means. For example, the mean responses in the jugular vein at times 0, 30, and 60 are 3.18, 4.29, and 3.76, respectively. The **lsmeans** for the jugular vein is thus $3.74 = (3.18 + 4.29 + 3.76)/3$. This average gives equal weight to the 3 times even though the sample sizes at these times differ (5, 6, and 3). The **means** of 3.78 for the jugular is the average of the 14 jugular responses, ignoring time. If the cell sample sizes were equal, the **lsmeans** and **means** averages would agree.

The **means** and **lsmeans** for individual cells (i.e., for the 6 **vein*time** combinations) are identical, and equal to cell means.

```
#### lsmeans
library(plyr)
# unbalanced, don't match
rat.mean.t  <- ddply(rat, .(time), summarise, m = mean(loginsulin))
rat.mean.t
##   time        m
```

```
## 1    0 3.997460
## 2   30 4.777822
## 3   60 4.580186
```

```
library(lsmeans)
lsmeans(lm.i.t.v.tv, list(pairwise ~ time), adjust = "bonferroni")
```

```
## NOTE: Results may be misleading due to involvement in interactions
## $`lsmeans of time`
##  time   lsmean        SE df lower.CL upper.CL
##  0    3.758920 0.1258994 42 3.504845 4.012996
##  30   4.679619 0.1221403 42 4.433130 4.926108
##  60   4.272270 0.1526754 42 3.964158 4.580381
##
## Results are averaged over the levels of: vein
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast   estimate        SE df t.ratio p.value
##  0 - 30   -0.9206985 0.1754107 42  -5.249  <.0001
##  0 - 60   -0.5133494 0.1978900 42  -2.594  0.0390
##  30 - 60   0.4073491 0.1955199 42   2.083  0.1300
##
## Results are averaged over the levels of: vein
## P value adjustment: bonferroni method for 3 tests
```

```
# unbalanced, don't match
rat.mean.v  <- ddply(rat, .(vein), summarise, m = mean(loginsulin))
rat.mean.v
```

```
##   vein        m
## 1    j 3.778293
## 2    p 4.712019
```

```
  # compare jugular mean above (3.778) with the lsmeans average below (3.742)
(3.179610 + 4.286804 + 3.759076)/3
```

```
## [1] 3.74183
```

```
lsmeans(lm.i.t.v.tv, list(pairwise ~ vein), adjust = "bonferroni")
```

```
## NOTE: Results may be misleading due to involvement in interactions
## $`lsmeans of vein`
##  vein   lsmean         SE df lower.CL upper.CL
##  j    3.741830 0.13192664 42 3.475592 4.008069
##  p    4.732042 0.08142689 42 4.567716 4.896369
##
## Results are averaged over the levels of: time
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast   estimate        SE df t.ratio p.value
##  j - p    -0.9902121 0.1550322 42  -6.387  <.0001
##
```

```
## Results are averaged over the levels of: time
```

```
# unbalanced, but highest-order interaction cell means will match
rat.mean.tv <- ddply(rat, .(time,vein), summarise, m = mean(loginsulin))
rat.mean.tv
```

```
##   time vein        m
## 1    0    j 3.179610
## 2    0    p 4.338230
## 3   30    j 4.286804
## 4   30    p 5.072433
## 5   60    j 3.759076
## 6   60    p 4.785463
```

```
lsmeans(lm.i.t.v.tv, list(pairwise ~ time | vein), adjust = "bonferroni")
```

```
## $`lsmeans of time | vein`
## vein = j:
##  time   lsmean        SE df lower.CL upper.CL
##  0    3.179610 0.2115533 42 2.752678 3.606542
##  30   4.286804 0.1931208 42 3.897071 4.676538
##  60   3.759076 0.2731141 42 3.207910 4.310243
##
## vein = p:
##  time   lsmean        SE df lower.CL upper.CL
##  0    4.338230 0.1365570 42 4.062647 4.613814
##  30   5.072433 0.1495908 42 4.770547 5.374320
##  60   4.785463 0.1365570 42 4.509880 5.061047
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast, vein | vein`
## vein = j:
##  contrast   estimate        SE df t.ratio p.value
##  0 - 30   -1.1071941 0.2864445 42  -3.865  0.0011
##  0 - 60   -0.5794659 0.3454650 42  -1.677  0.3027
##  30 - 60   0.5277282 0.3344951 42   1.578  0.3664
##
## vein = p:
##  contrast   estimate        SE df t.ratio p.value
##  0 - 30   -0.7342029 0.2025468 42  -3.625  0.0023
##  0 - 60   -0.4472330 0.1931208 42  -2.316  0.0766
##  30 - 60   0.2869699 0.2025468 42   1.417  0.4917
##
## P value adjustment: bonferroni method for 3 tests
```

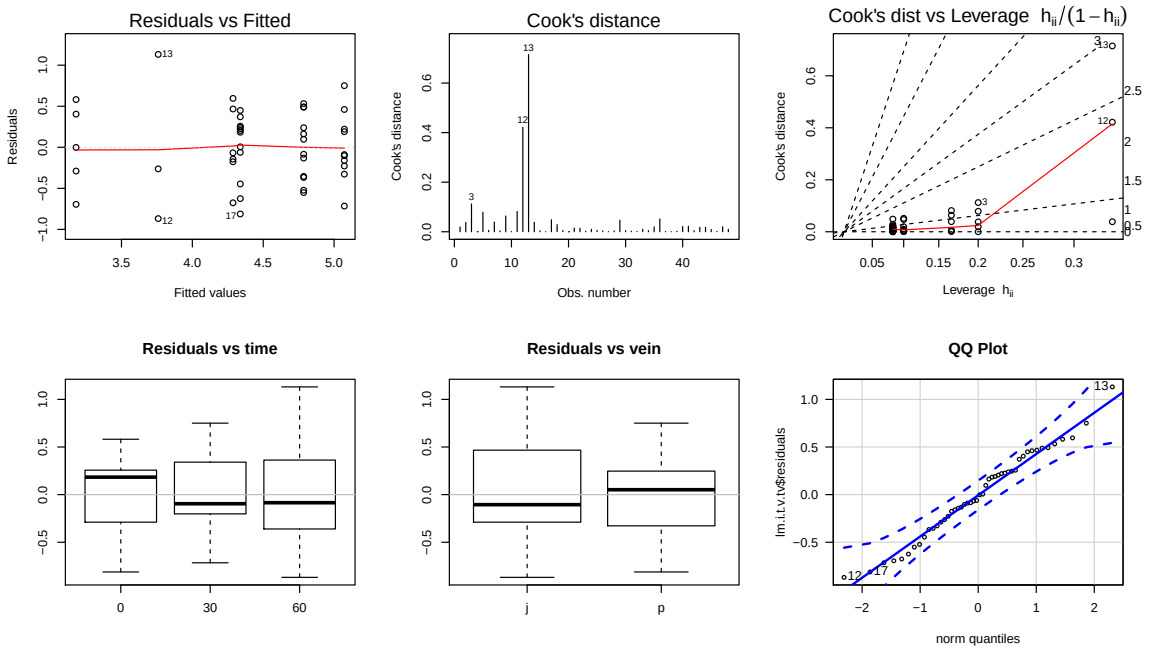
```
lsmeans(lm.i.t.v.tv, list(pairwise ~ vein | time), adjust = "bonferroni")
```

```
## $`lsmeans of vein | time`
## time = 0:
##  vein   lsmean        SE df lower.CL upper.CL
##  j    3.179610 0.2115533 42 2.752678 3.606542
```

```
##  p    4.338230 0.1365570 42 4.062647 4.613814
##
## time = 30:
##  vein   lsmean        SE df lower.CL upper.CL
##  j    4.286804 0.1931208 42 3.897071 4.676538
##  p    5.072433 0.1495908 42 4.770547 5.374320
##
## time = 60:
##  vein   lsmean        SE df lower.CL upper.CL
##  j    3.759076 0.2731141 42 3.207910 4.310243
##  p    4.785463 0.1365570 42 4.509880 5.061047
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast, time | time`
## time = 0:
##  contrast   estimate        SE df t.ratio p.value
##  j - p    -1.1586202 0.2517988 42  -4.601  <.0001
##
## time = 30:
##  contrast   estimate        SE df t.ratio p.value
##  j - p    -0.7856289 0.2442807 42  -3.216  0.0025
##
## time = 60:
##  contrast   estimate        SE df t.ratio p.value
##  j - p    -1.0263873 0.3053508 42  -3.361  0.0017
```

For completeness, these diagnostic plots are mostly fine, though the plot of the Cook's distances indicate a couple influential observations.

```
# interaction model
lm.i.t.v.tv <- lm(loginsulin ~ time*vein, data = rat
                  , contrasts = list(time = contr.sum, vein = contr.sum))

# plot diagnostics
par(mfrow=c(2,3))
plot(lm.i.t.v.tv, which = c(1,4,6))

plot(rat$time, lm.i.t.v.tv$residuals, main="Residuals vs time")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(rat$vein, lm.i.t.v.tv$residuals, main="Residuals vs vein")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.i.t.v.tv$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1] 13 12 17
## residuals vs order of data
#plot(lm.i.t.v.tv$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Should I use means or lsmeans, Type I or Type III SS? *Use lsmeans and Type III SS.*

Regardless of whether the design is balanced, the basic building blocks for a two-factor analysis are cell means, and the marginal means, defined as the average of the cell means over the levels of the other factor.

The F-statistics based on Type III SSs are appropriate for unbalanced two-factor designs because they test the same hypotheses that were considered in balanced designs. That is, the Type III F-tests on the main effects check for equality in population means averaged over levels of the other factor. The Type III F-test for no interaction checks for parallel profiles. Given that the Type III F-tests for the main effects check for equal population cell means averaged over the levels of the other factor, multiple comparisons for main effects should be based on **lsmeans**.

The Type I SS and F-tests and the multiple comparisons based on **means should be ignored** because they do not, in general, test meaningful hypotheses. The problem with using the **means** output is that the experimenter has fixed the sample sizes for a two-factor experiment, so comparisons of **means**, which ignore the second factor, introduces a potential bias due to choice of sample sizes. Put another way, any differences seen in the **means** in the jugular and portal could be solely due to the sample sizes used in the experiment and not due to differences in the veins.

Focusing on the Type III SS, the F-tests indicate that the vein and time effects are significant, but that the interaction is not significant. The jugular and portal

profiles are reasonably parallel, which is consistent with a lack of interaction. What can you conclude from the **lsmeans** comparisons of veins and times?

Answer: significant differences between veins, and between times 0 and 30.

5.5 Writing factor model equations and interpretting coefficients

This section is an exercise in writing statistical factor models, plotting predicted values, and interpretting model coefficients. You'll need pen (preferrably multi-colored) and paper.

In class I'll discuss indicator variables and writing these models. Together, we'll plot the model predicted values, write the model for each factor combination, and indicate the β coefficients on the plot (βs are vertical differences)[4]. From the exercise, I hope that coefficient interpretation will become clear. Assume a balanced design with n_i observations for each treatment combination.

5.5.1 One-way ANOVA, 1 factor with 3 levels

1. Write ANOVA factor model (general and indicator-variables)
2. Write model for each factor level with βs and predicted values
3. Plot the predicted values on axes
4. Label the βs on the plot
5. Calculate marginal and grand means in table and label in plot

Level	1	2	3	
$\hat{y}$	5	4	6	

5.5.2 Two-way ANOVA, 2 factors with 3 and 2 levels, additive model

1. Write two-way ANOVA factor model (general and indicator-variables)
2. Write model for each factor level with βs and predicted values
3. Plot the predicted values on axes
4. Label the βs on the plot
5. Calculate marginal and grand means in table and label in plot

$\hat{y}$	Factor 1			
Factor 2	1	2	3	
1	5	4	6	
2	8	7	9	

[4]Please attempt by hand before looking at the solutions at `http://statacumen.com/teach/ADA2/ADA2_05_PairedAndBlockDesigns_CoefScan.pdf`.

5.5.3 Two-way ANOVA, 2 factors with 3 and 2 levels, interaction model

1. Write two-way ANOVA factor model (general and indicator-variables)
2. Write model for each factor level with βs and predicted values
3. Plot the predicted values on axes
4. Label the βs on the plot
5. Calculate marginal and grand means in table and label in plot

$\hat{y}$	Factor 1			
Factor 2	1	2	3	
1	5	4	6	
2	8	10	3	

Chapter 6

A Short Discussion of Observational Studies

"Thou shall adjust for what thou can not control."

In most scientific studies, the groups being compared do not consist of identical experimental units that have been randomly assigned to receive a treatment. Instead, the groups might be extremely heterogeneous on factors that might be related to a specific response on which you wish to compare the groups. Inferences about the nature of differences among groups in such **observational studies** can be flawed if this heterogeneity is ignored in the statistical analysis.

The following problem emphasizes the care that is needed when analyzing **observational studies**, and highlights the distinction between the **means** and **lsmeans** output for a two-way table. The **data are artificial**, but the conclusions are consistent with an interesting analysis conducted by researchers at Sandia National Laboratories.

A representative sample of 550 high school seniors was selected in 1970. A similar sample of 550 was selected in 1990. The final SAT scores (on a 1600 point scale) were obtained for each student[1].

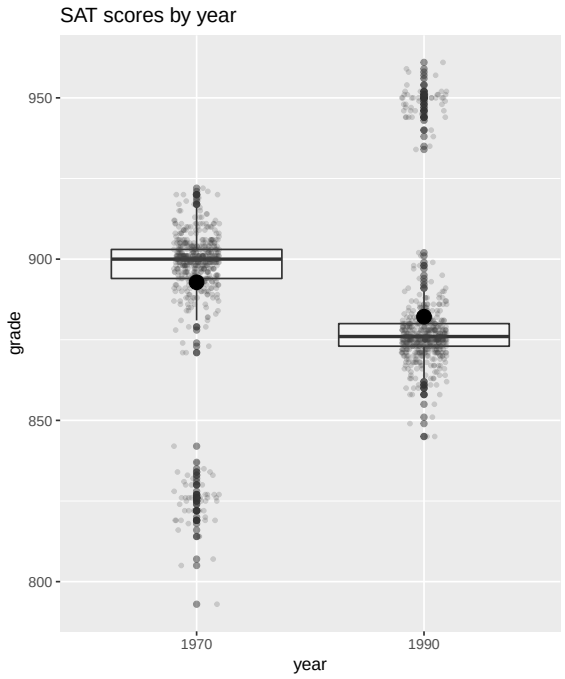
The boxplots for the two samples show heavy-tailed distributions with similar spreads. Given the large sample sizes, the F-test comparing populations is approximately valid even though the population distributions are non-normal.

```
#### Example: SAT
sat <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch06_sat.dat", header = TRUE)
sat$year <- factor(sat$year)
```

[1]The fake-data example in this chapter is similar to a real-world SAT example illustrated in this paper: "Minority Contributions to the SAT Score Turnaround: An Example of Simpson's Paradox" by Howard Wainer, Journal of Educational Statistics, Vol. 11, No. 4 (Winter, 1986), pp. 239–244 http://www.jstor.org/stable/1164696.

```
sat$eth  <- factor(sat$eth )

# calculate means by year (also calculated below to illustrate lsmeans())
library(plyr)
sat.mean.y  <- ddply(sat, .(year), summarise, m = mean(grade))
```

```
# Interaction plots, ggplot
p <- ggplot(sat, aes(x = year, y = grade))
p <- p + geom_boxplot(alpha = 0.5)
p <- p + geom_point(position = position_jitter(w = 0.1, h = 0), colour="gray25", size=1, alpha = 0.2)
p <- p + geom_point(data = sat.mean.y, aes(y = m), size = 4)
#p <- p + geom_line(data = sat.mean.y, aes(y = m), size = 1.5)
p <- p + labs(title = "SAT scores by year")
print(p)
```



A simple analysis might compare the average SAT scores for the two years, to see whether students are scoring higher, lower, or about the same, over time. The one-way **lsmeans** and **means** breakdowns of the SAT scores are identical; the average SAT scores for 1970 and 1990 are 892.8 and 882.2, respectively. The one-way ANOVA, combined with the observed averages, indicates that the typical SAT score has decreased significantly (10.7 points) over the 20 year period.

```
lm.g.y <- lm(grade ~ year, data = sat
             , contrasts = list(year = contr.sum))
library(car)
# type III SS
Anova(lm.g.y, type=3)
## Anova Table (Type III tests)
##
## Response: grade
```

```
##                Sum Sq   Df    F value    Pr(>F)
## (Intercept) 866418325    1 1.7076e+06 < 2.2e-16 ***
## year            31410    1 6.1904e+01 8.591e-15 ***
## Residuals      557117 1098
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
library(plyr)
# balanced with respect to year, so means and lsmeans match
sat.mean.y  <- ddply(sat, .(year), summarise, m = mean(grade))
sat.mean.y
##   year        m
## 1 1970 892.8418
## 2 1990 882.1545
library(lsmeans)
lsmeans(lm.g.y, list(pairwise ~ year), adjust = "bonferroni")
## $`lsmeans of year`
##  year   lsmean        SE   df lower.CL upper.CL
##  1970 892.8418 0.9604853 1098 890.9572 894.7264
##  1990 882.1545 0.9604853 1098 880.2700 884.0391
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast    estimate       SE   df t.ratio p.value
##  1970 - 1990 10.68727 1.358331 1098   7.868  <.0001
```

Should we be alarmed? Should we be concerned that students entering college have fewer skills than students 20 years ago? Should we be pumping billions of dollars into the bloated bureaucracies of our public school systems with the hope that a few of these dollars might be put to good use in programs to enhance performance? This is the consensus among some people in the know, all of whom wax eloquently about the impending inability of the U.S. to compete in the new global economy.

The SAT study is not a well-designed experiment, where a scientist has controlled all the factors that might affect the response (the SAT score) other than the treatment (the year). Even without these controls, there is no randomization of treatments to students selected from a target population.

The SAT study is an **observational study** of two distinct populations. The observed differences in SAT scores may indeed be due to a decrease in performance. The differences might also be due to factors that make the two populations incomparable for assessing changes in performance over time.

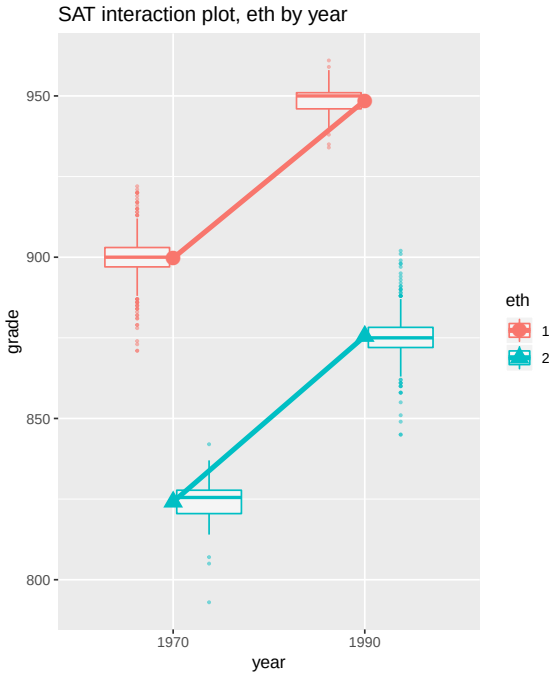
My hypothetical populations have students from two ethnic groups (1 and 2). If you construct box-plots of the SAT scores for the four combinations of ethnicity and year, you see that the typical SAT score **within** each ethnic group has **increased** over time, whereas the typical SAT score **ignoring** ethnicity decreased over time. Is

this a paradox, and what are appropriate conclusions in the analysis?

```
sat.mean.ye <- ddply(sat, .(year,eth), summarise, m = mean(grade))
sat.mean.ye
```

```
##   year eth       m
## 1 1970   1 899.712
## 2 1970   2 824.140
## 3 1990   1 948.560
## 4 1990   2 875.514
```

```
# Interaction plots, ggplot
library(ggplot2)
p <- ggplot(sat, aes(x = year, y = grade, colour = eth, shape = eth))
p <- p + geom_boxplot(alpha = 0.5, outlier.size=0.5)
p <- p + geom_point(data = sat.mean.ye, aes(y = m), size = 4)
p <- p + geom_line(data = sat.mean.ye, aes(y = m, group = eth), size = 1.5)
p <- p + labs(title = "SAT interaction plot, eth by year")
print(p)

#p <- ggplot(sat, aes(x = eth, y = grade, colour = year, shape = year))
#p <- p + geom_boxplot(alpha = 0.5, outlier.size=0.5)
#p <- p + geom_point(data = sat.mean.ye, aes(y = m), size = 4)
#p <- p + geom_line(data = sat.mean.ye, aes(y = m, group = year), size = 1.5)
#p <- p + labs(title = "SAT interaction plot, year by eth")
#print(p)
```



I fit a two-factor model with year and ethnicity effects plus an interaction. The two-factor model gives a method to compare the SAT scores over time, after **adjusting** for the effect of ethnicity on performance. The F-test for comparing years adjusts for ethnicity because it is based on comparing the average SAT scores across years after averaging the cell means over ethnicities, thereby eliminating from the comparison of years any effects due to changes in the ethnic composition of the populations. The two-way analysis is preferable to the unadjusted one-way analysis which **ignores** ethnicity.

```
lm.g.y.e.ye <- lm(grade ~ year * eth, data = sat
                  , contrasts = list(year = contr.sum, eth = contr.sum))
```

```
## CRITICAL!!!  Unbalanced design warning.
##   The contrast statement above must be included identifying
##   each main effect with "contr.sum" in order for the correct
##   Type III SS to be computed.
##   See http://goanna.cs.rmit.edu.au/~fscholer/anova.php
library(car)
# type III SS
Anova(lm.g.y.e.ye, type=3)
## Anova Table (Type III tests)
##
## Response: grade
##                Sum Sq   Df    F value  Pr(>F)
## (Intercept) 286085884    1 5.7022e+06 < 2e-16 ***
## year           228283    1 4.5501e+03 < 2e-16 ***
## eth            501984    1 1.0005e+04 < 2e-16 ***
## year:eth          145    1 2.8904e+00 0.08939 .
## Residuals       54988 1096
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The year and ethnicity main effects are significant in the two factor model, but the interaction is not. The marginal **lsmeans** indicate that the average SAT score increased significantly over time when averaged over ethnicities. This is consistent with the cell mean SAT scores increasing over time **within** each ethnic group. Given the lack of a significant interaction, the expected increase in SAT scores from 1970 to 1990 **within each** ethnic group is the difference in marginal averages: 912.0 - 861.9 = 50.1.

```
library(plyr)
# unbalanced, don't match (lsmeans is correct)
sat.mean.y  <- ddply(sat, .(year), summarise, m = mean(grade))
sat.mean.y
##   year        m
## 1 1970 892.8418
## 2 1990 882.1545
library(lsmeans)
lsmeans(lm.g.y.e.ye, list(pairwise ~ year), adjust = "bonferroni")
## NOTE: Results may be misleading due to involvement in interactions
## $`lsmeans of year`
##  year  lsmean        SE   df lower.CL upper.CL
##  1970 861.926 0.5253021 1096 860.8953 862.9567
##  1990 912.037 0.5253021 1096 911.0063 913.0677
##
## Results are averaged over the levels of: eth
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
```

```
##  contrast    estimate        SE   df t.ratio p.value
##  1970 - 1990  -50.111 0.7428893 1096 -67.454  <.0001
##
## Results are averaged over the levels of: eth
```

```
# unbalanced, don't match (lsmeans is correct)
sat.mean.e  <- ddply(sat, .(eth), summarise, m = mean(grade))
sat.mean.e
```

```
##   eth        m
## 1   1 904.1527
## 2   2 870.8436
```

```
lsmeans(lm.g.y.e.ye, list(pairwise ~ eth), adjust = "bonferroni")
```

```
## NOTE: Results may be misleading due to involvement in interactions
## $`lsmeans of eth`
##  eth  lsmean        SE   df lower.CL upper.CL
##  1   924.136 0.5253021 1096 923.1053 925.1667
##  2   849.827 0.5253021 1096 848.7963 850.8577
##
## Results are averaged over the levels of: year
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast estimate        SE   df t.ratio p.value
##  1 - 2      74.309 0.7428893 1096 100.027  <.0001
##
## Results are averaged over the levels of: year
```

```
# unbalanced, but highest-order interaction cell means will match
sat.mean.ye <- ddply(sat, .(year,eth), summarise, m = mean(grade))
sat.mean.ye
```

```
##   year eth       m
## 1 1970   1 899.712
## 2 1970   2 824.140
## 3 1990   1 948.560
## 4 1990   2 875.514
```

```
lsmeans(lm.g.y.e.ye, list(pairwise ~ year | eth), adjust = "bonferroni")
```

```
## $`lsmeans of year | eth`
## eth = 1:
##  year  lsmean        SE   df lower.CL upper.CL
##  1970 899.712 0.3167691 1096 899.0905 900.3335
##  1990 948.560 1.0017118 1096 946.5945 950.5255
##
## eth = 2:
##  year  lsmean        SE   df lower.CL upper.CL
##  1970 824.140 1.0017118 1096 822.1745 826.1055
##  1990 875.514 0.3167691 1096 874.8925 876.1355
##
## Confidence level used: 0.95
```

```
##
## $`pairwise differences of contrast, eth | eth`
## eth = 1:
##  contrast    estimate       SE   df t.ratio p.value
##  1970 - 1990  -48.848 1.050604 1096 -46.495  <.0001
##
## eth = 2:
##  contrast    estimate       SE   df t.ratio p.value
##  1970 - 1990  -51.374 1.050604 1096 -48.899  <.0001
lsmeans(lm.g.y.e.ye, list(pairwise ~ eth | year), adjust = "bonferroni")
## $`lsmeans of eth | year`
## year = 1970:
##  eth  lsmean        SE   df lower.CL upper.CL
##  1   899.712 0.3167691 1096 899.0905 900.3335
##  2   824.140 1.0017118 1096 822.1745 826.1055
##
## year = 1990:
##  eth  lsmean        SE   df lower.CL upper.CL
##  1   948.560 1.0017118 1096 946.5945 950.5255
##  2   875.514 0.3167691 1096 874.8925 876.1355
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast, year | year`
## year = 1970:
##  contrast estimate       SE   df t.ratio p.value
##  1 - 2      75.572 1.050604 1096  71.932  <.0001
##
## year = 1990:
##  contrast estimate       SE   df t.ratio p.value
##  1 - 2      73.046 1.050604 1096  69.528  <.0001
```

As noted in the insulin analysis, the marginal **lsmeans** and **means** are different for unbalanced two-factor analyses. The marginal **means ignore** the levels of the other factors when averaging responses. The marginal **lsmeans** are averages of cell means over the levels of the other factor. Thus, for example, the 1970 **mean** SAT score of 892.8 is the average of the 550 scores selected that year. The 1970 **lsmeans** SAT score of 861.9 is midway between the average 1970 SAT scores for the two ethnic groups: $861.9 = (899.7 + 824.1)/2$. Hopefully, this discussion also clarifies why the year marginal **means** are identical in the one and two-factor analyses, but the year **lsmeans** are not.

The 1970 and 1990 marginal **means** estimate the typical SAT score ignoring all factors that may influence performance. These marginal averages are not relevant for **understanding** any trends in performance over time because they do not account for changes in the composition of the population that may be related to performance.

The average SAT scores (ignoring ethnicity) decreased from 1970 to 1990 because

the ethnic composition of the student population changed. Ten out of every eleven students sampled in 1970 were from the first ethnic group. Only one out of eleven students sampled in 1990 was from this group. Students in the second ethnic group are underachievers, but they are becoming a larger portion of the population over time. The decrease in average (**means**) performance inferred from comparing 1970 to 1990 is **confounded** with the increased representation of the underachievers over time. Once ethnicity was taken into consideration, the typical SAT scores were shown to have increased, rather than decreased.

In summary, the one-way analysis ignoring ethnicity is valid, and allows you to conclude that the typical SAT score has decreased over time, but it does not provide any insight into the nature of the changes that have occurred. A two-factor analysis backed up with a comparison of the marginal **lsmeans** is needed to compare performances over time, adjusting for the changes in ethnic composition.

The Sandia study reached the same conclusion. The Sandia team showed that the widely reported decreases in SAT scores over time are due to changes in the ethnic distribution of the student population over time, with individuals in historically underachieving ethnic groups becoming a larger portion of the student population over time.

A more complete analysis of the SAT study would adjust the SAT scores to account for other potential confounding factors, such as sex, and differences due to the number of times the exam was taken. These confounding effects are taken into consideration by including them as effects in the model.

The interpretation of the results from an observational study with several effects of interest, and several confounding variables, is greatly simplified by eliminating the insignificant effects from the model. For example, the year by ethnicity interaction in the SAT study might be omitted from the model to simplify interpretation. The year effects would then be estimated after fitting a two-way additive model with year and ethnicity effects only. The same approach is sometimes used with designed experiments, say the insulin study that we analyzed earlier.

An important caveat The ideas that we discussed on the design and analysis of experiments and observational studies are universal. They apply regardless of whether you are analyzing categorical data, counts, or measurements.

Part VIII

ADA2: ANCOVA and logistic regression

Chapter 7

Analysis of Covariance: Comparing Regression Lines

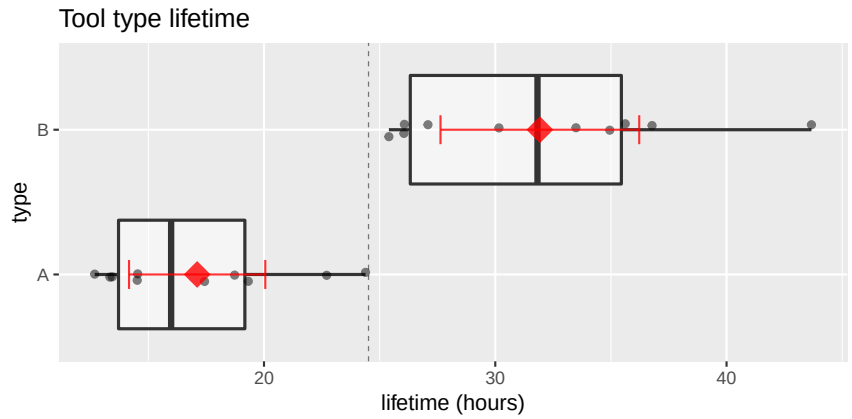
Suppose that you are interested in comparing the typical lifetime (hours) of two tool types (A and B). A simple analysis of the data given below would consist of making side-by-side boxplots followed by a two-sample test of equal means (or medians). The standard two-sample test using the pooled variance estimator is a special case of the one-way ANOVA with two groups. The summaries suggest that the distribution of lifetimes for the tool types are different. In the output below, μ_i is population mean lifetime for tool type i ($i = A, B$).

```
#### Example: Tool lifetime
tools <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch07_tools.dat"
                    , header = TRUE)
str(tools)
```

```
## 'data.frame': 20 obs. of  3 variables:
##  $ lifetime: num  18.7 14.5 17.4 14.5 13.4 ...
##  $ rpm     : int  610 950 720 840 980 530 680 540 890 730 ...
##  $ type    : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
```

	lifetime	rpm	type		lifetime	rpm	type
1	18.7300	610	A	11	30.1600	670	B
2	14.5200	950	A	12	27.0900	770	B
3	17.4300	720	A	13	25.4000	880	B
4	14.5400	840	A	14	26.0500	1000	B
5	13.4400	980	A	15	33.4900	760	B
6	24.3900	530	A	16	35.6200	590	B
7	13.3400	680	A	17	26.0700	910	B
8	22.7100	540	A	18	36.7800	650	B
9	12.6800	890	A	19	34.9500	810	B
10	19.3200	730	A	20	43.6700	500	B

```
library(ggplot2)
p <- ggplot(tools, aes(x = type, y = lifetime))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(aes(yintercept = mean(lifetime)),
                    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 = "Tool type lifetime") + ylab("lifetime (hours)")
p <- p + coord_flip()
print(p)
```



A two sample t-test comparing mean lifetimes of tool types indicates a difference between means.

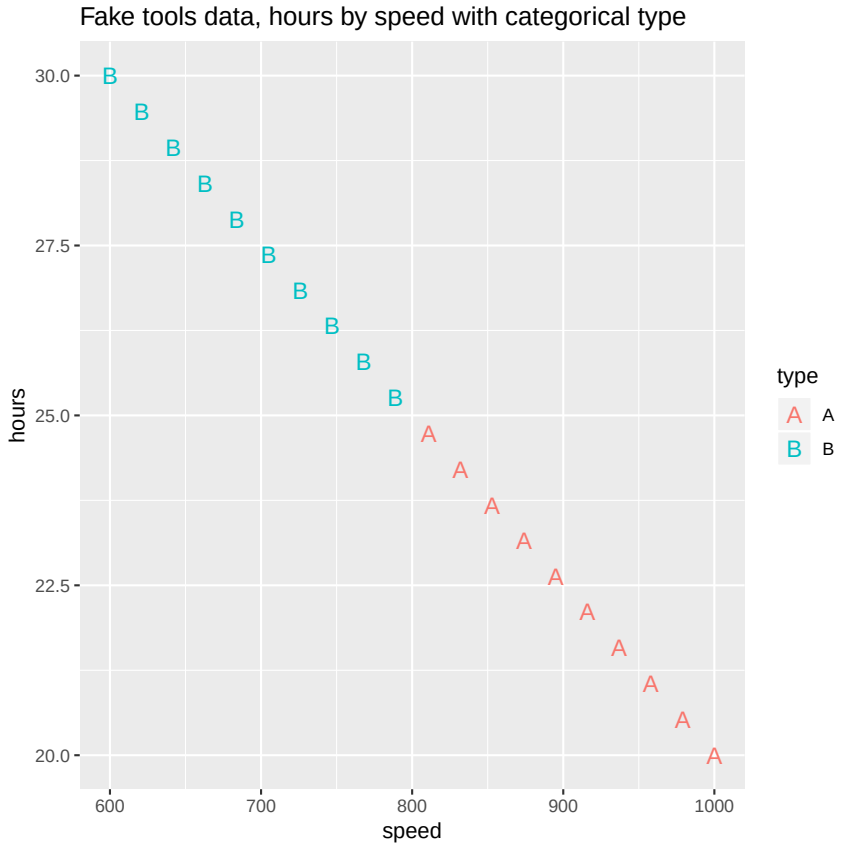
```
t.summary <- t.test(lifetime ~ type, data = tools)
t.summary
```

```
##
##  Welch Two Sample t-test
##
## data:  lifetime by type
## t = -6.435, df = 15.93, p-value = 8.422e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -19.70128  -9.93472
## sample estimates:
## mean in group A mean in group B
##          17.110          31.928
```

This comparison is potentially misleading because the samples are not comparable. A one-way ANOVA is most appropriate for designed experiments where all the factors influencing the response, other than the treatment (tool type), are fixed by the experimenter. The tools were operated at different speeds. If speed influences lifetime, then the observed differences in lifetimes could be due to differences in speeds at which the two tool types were operated.

Fake example For example, suppose speed is inversely related to lifetime of the tool. Then, the differences seen in the boxplots above could be due to tool type B being operated at lower speeds than tool type A. To see how this is possible, consider the data plot given below, where the relationship between lifetime and speed is identical in each sample. A simple linear regression model relating hours to speed, ignoring tool type, fits the data exactly, yet the lifetime distributions for the tool types, ignoring speed, differ dramatically. (The data were generated to fall exactly on a straight line). The regression model indicates that you would expect identical mean lifetimes for tool types A and B, if they were, or could be, operated at identical speeds. This is not exactly what happens in the actual data. However, I hope the point is clear.

```
#### Example: Tools, fake
toolsfake <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch07_toolsfake.dat"
                        , header = TRUE)
library(ggplot2)
p <- ggplot(toolsfake, aes(x = speed, y = hours, colour = type, shape = type))
p <- p + geom_point(size=4)
  library(R.oo) # for ascii code lookup
  p <- p + scale_shape_manual(values=charToInt(sort(unique(toolsfake$type))))
p <- p + labs(title="Fake tools data, hours by speed with categorical type")
print(p)
```



As noted in the Chapter 6 SAT example, you should be wary of group comparisons where important factors that influence the response have not been accounted for or

controlled. In the SAT example, the differences in scores were affected by a change in the ethnic composition over time. A two-way ANOVA with two factors, time and ethnicity, gave the most sensible analysis.

For the tool lifetime problem, you should compare groups (tools) after adjusting the lifetimes to account for the influence of a measurement variable, speed. The appropriate statistical technique for handling this problem is called **analysis of covariance** (ANCOVA).

7.1 ANCOVA

A natural way to account for the effect of speed is through a multiple regression model with lifetime as the response and two predictors, speed and tool type. A binary **categorical variable**, here tool type, is included in the model as a **dummy variable** or **indicator variable** (a $\{0, 1\}$ variable).

Consider the model

$$\text{Tool lifetime} = \beta_0 + \beta_1 \text{ typeB} + \beta_2 \text{ rpm} + e,$$

where typeB is 0 for type A tools, and 1 for type B tools. For type A tools, the model simplifies to:

$$\begin{aligned} \text{Tool lifetime} &= \beta_0 + \beta_1(0) + \beta_2 \text{ rpm} + e \\ &= \beta_0 + \beta_2 \text{ rpm} + e. \end{aligned}$$

For type B tools, the model simplifies to:

$$\begin{aligned} \text{Tool lifetime} &= \beta_0 + \beta_1(1) + \beta_2 \text{ rpm} + e \\ &= (\beta_0 + \beta_1) + \beta_2 \text{ rpm} + e. \end{aligned}$$

This ANCOVA model fits two regression lines, one for each tool type, but restricts the slopes of the regression lines to be identical. To see this, let us focus on the interpretation of the regression coefficients. For the ANCOVA model,

β_2 = slope of population regression lines for tool types A and B.

and

β_0 = intercept of population regression line for tool A (called the reference group).

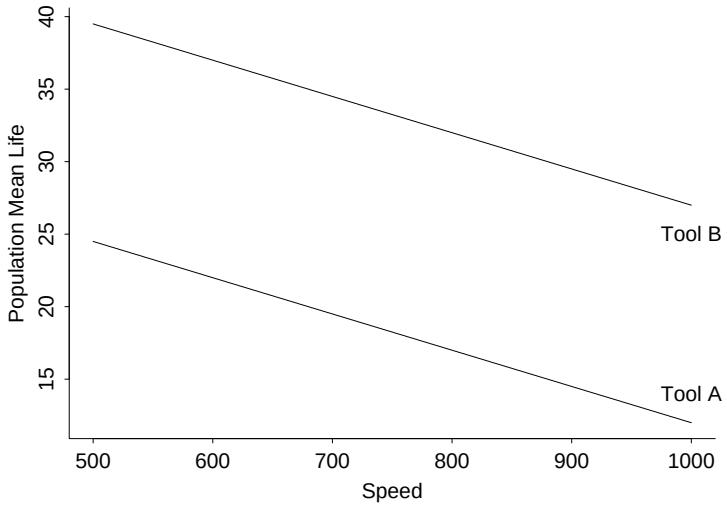
Given that

$\beta_0 + \beta_1$ = intercept of population regression line for tool B,

it follows that

β_1 = difference between tool B and tool A intercepts.

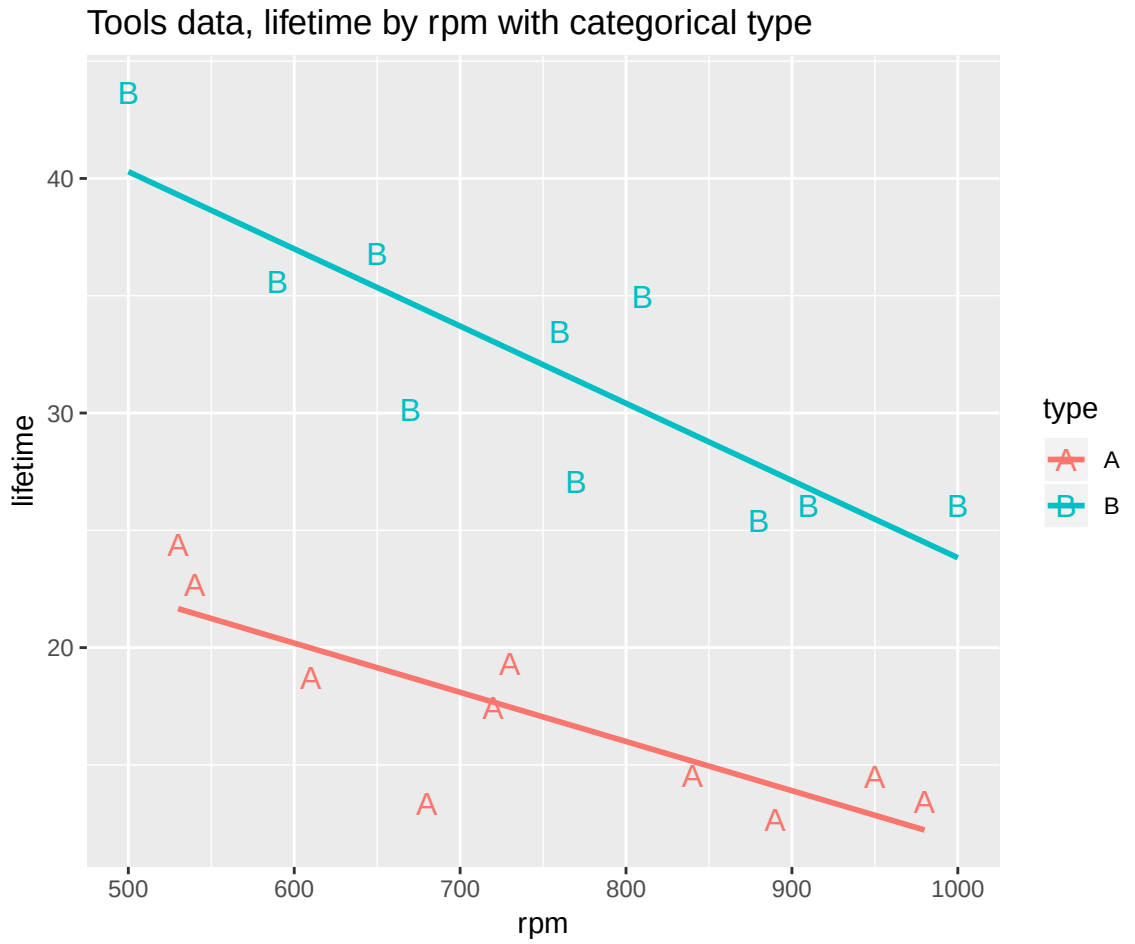
A picture of the population regression lines for one version of the model is given below.



An important feature of the ANCOVA model is that β_1 measures the difference in mean response for the tool types, regardless of the speed. A test of $H_0 : \beta_1 = 0$ is the primary interest, and is interpreted as a **comparison of the tool types, after adjusting or allowing for the speeds at which the tools were operated**.

The ANCOVA model is plausible. The relationship between lifetime and speed is roughly linear within tool types, with similar slopes but unequal intercepts across groups. The plot of the studentized residuals against the fitted values shows no gross abnormalities, but suggests that the variability about the regression line for tool type A is somewhat smaller than the variability for tool type B. The model assumes that the variability of the responses is the same for each group. The QQ-plot does not show any gross deviations from a straight line.

```
#### Example: Tool lifetime
library(ggplot2)
p <- ggplot(tools, aes(x = rpm, y = lifetime, colour = type, shape = type))
p <- p + geom_point(size=4)
  library(R.oo) # for ascii code lookup
  p <- p + scale_shape_manual(values=charToInt(sort(unique(tools$type))))
p <- p + geom_smooth(method = lm, se = FALSE)
p <- p + labs(title="Tools data, lifetime by rpm with categorical type")
print(p)
```




```
lm.l.r.t <- lm(lifetime ~ rpm + type, data = tools)
#library(car)
#Anova(aov(lm.l.r.t), type=3)
summary(lm.l.r.t)
```

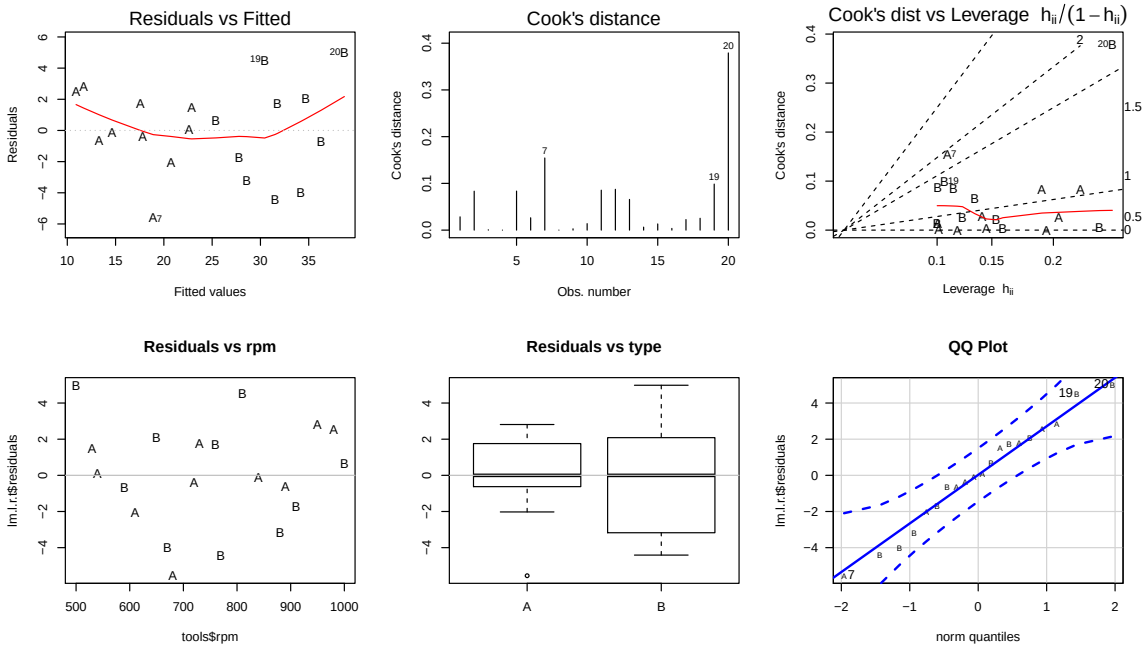
```
##
## Call:
## lm(formula = lifetime ~ rpm + type, data = tools)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.5527 -1.7868 -0.0016  1.8395  4.9838
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 36.98560    3.51038  10.536 7.16e-09 ***
## rpm         -0.02661    0.00452  -5.887 1.79e-05 ***
## typeB       15.00425    1.35967  11.035 3.59e-09 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.039 on 17 degrees of freedom
## Multiple R-squared:  0.9003,Adjusted R-squared:  0.8886
## F-statistic: 76.75 on 2 and 17 DF,  p-value: 3.086e-09
```

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.l.r.t, which = c(1,4,6), pch=as.character(tools$type))

plot(tools$rpm, lm.l.r.t$residuals, main="Residuals vs rpm", pch=as.character(tools$type))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(tools$type, lm.l.r.t$residuals, main="Residuals vs type")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.l.r.t$residuals, las = 1, id = list(n = 3), main="QQ Plot", pch=as.character(tools$type))
```

```
## [1]  7 20 19
```

```
## residuals vs order of data
#plot(lm.l.r.t$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



The fitted relationship for the combined data set is

$$\text{Predicted Lifetime} = 36.99 + 15.00\ \text{typeB} - 0.0266\ \text{rpm}.$$

Assigning the LS estimates to the appropriate parameters, the fitted relationships for the two tool types must be, for tool type B:

$$\begin{aligned}\text{Predicted Lifetime} &= (36.99 + 15.00) - 0.0266\ \text{rpm} \\ &= 51.99 - 0.0266\ \text{rpm},\end{aligned}$$

and for tool type A:

$$\text{Predicted Lifetime} \;=\; 36.99 - 0.0266 \text{ rpm}.$$

The t-test of $H_0 : \beta_1 = 0$ checks whether the intercepts for the population regression lines are equal, assuming equal slopes. The t-test p-value < 0.0001 suggests that the population regression lines for tools A and B have unequal intercepts. The LS lines indicate that the average lifetime of either type tool decreases by 0.0266 hours for each increase in 1 RPM. Regardless of the lathe speed, the model predicts that type B tools will last 15 hours longer (i.e., the regression coefficient for the typeB predictor) than type A tools. Summarizing this result another way, the t-test suggests that there is a significant difference between the lifetimes of the two tool types, after adjusting for the effect of the speeds at which the tools were operated. The estimated difference in average lifetime is 15 hours, regardless of the lathe speed.

7.2 Generalizing the ANCOVA Model to Allow Unequal Slopes

I will present a flexible approach for checking equal slopes and equal intercepts in ANCOVA-type models. The algorithm also provides a way to build regression models in studies where the primary interest is comparing the regression lines across groups rather than comparing groups after adjusting for a regression effect. The approach can be applied to an arbitrary number of groups and predictors. For simplicity, I will consider a problem with three groups and a single regression effect.

The data[1] below are the IQ scores of identical twins, one raised in a foster home (IQF) and the other raised by natural parents (IQN). The 27 pairs are divided into three groups by social status of the natural parents (H=high, M=medium, L=low). I will examine the regression of IQF on IQN for each of the three social classes.

There is no **a priori** reason to assume that the regression lines for the three groups have equal slopes or equal interepts. These are, however, reasonable hypotheses to examine. The easiest way to check these hypotheses is to fit a multiple regression model to the combined data set, and check whether certain carefully defined regression effects are zero. The most general model has six parameters, and corresponds to fitting a simple linear regression model to the three groups separately ($3 \times 2 = 6$).

Two indicator variables are needed to uniquely identify each observation by social class. For example, let $I_1 = 1$ for H status families and $I_1 = 0$ otherwise, and let $I_2 = 1$ for M status families and $I_2 = 0$ otherwise. The indicators I_1 and I_2 jointly assume 3 values:

[1]The data were originally analyzed by Sir Cyril Burt.

Status	I_1	I_2
L	0	0
M	0	1
H	1	0

Given the indicators I_1 and I_2 and the predictor IQN, define two **interaction** or **product effects**: $I_1 \times$ IQN and $I_2 \times$ IQN.

7.2.1 Unequal slopes ANCOVA model

The most general model allows separate slopes and intercepts for each group:

$$\text{IQF} = \beta_0 + \beta_1 I_1 + \beta_2 I_2 + \beta_3 \text{ IQN} + \beta_4 I_1 \text{ IQN} + \beta_5 I_2 \text{ IQN} + e. \quad (7.1)$$

This model is best understood by considering the three status classes separately. If status = L, then $I_1 = I_2 = 0$. For these families

$$\text{IQF} = \beta_0 + \beta_3 \text{ IQN} + e.$$

If status = M, then $I_1 = 0$ and $I_2 = 1$. For these families

$$\begin{aligned} \text{IQF} &= \beta_0 + \beta_2(1) + \beta_3 \text{ IQN} + \beta_5 \text{ IQN} + e \\ &= (\beta_0 + \beta_2) + (\beta_3 + \beta_5) \text{ IQN} + e. \end{aligned}$$

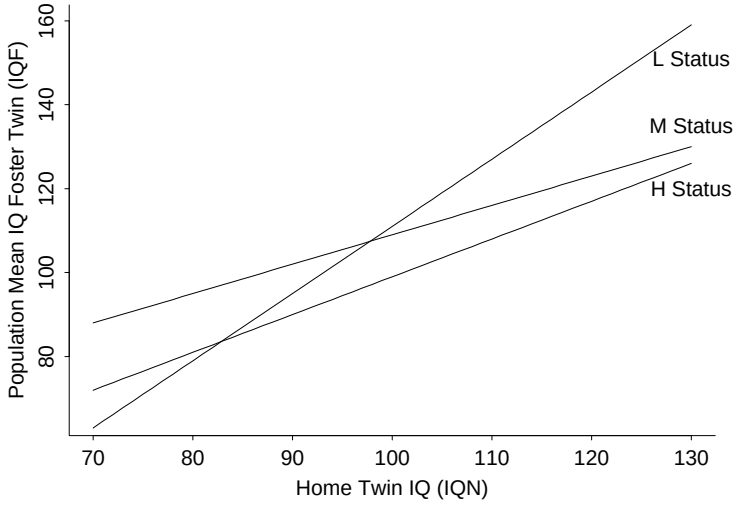
Finally, if status = H, then $I_1 = 1$ and $I_2 = 0$. For these families

$$\begin{aligned} \text{IQF} &= \beta_0 + \beta_1(1) + \beta_3 \text{ IQN} + \beta_4 \text{ IQN} + e \\ &= (\beta_0 + \beta_1) + (\beta_3 + \beta_4) \text{ IQN} + e. \end{aligned}$$

The regression coefficients β_0 and β_3 are the intercept and slope for the L status population regression line. The other parameters measure differences in intercepts and slopes across the three groups, using L status families as a **baseline** or **reference group**. In particular:

β_1 = difference between the intercepts of the H and L population regression lines.
β_2 = difference between the intercepts of the M and L population regression lines.
β_4 = difference between the slopes of the H and L population regression lines.
β_5 = difference between the slopes of the M and L population regression lines.

The plot gives a possible picture of the population regression lines corresponding to the general model (7.1).

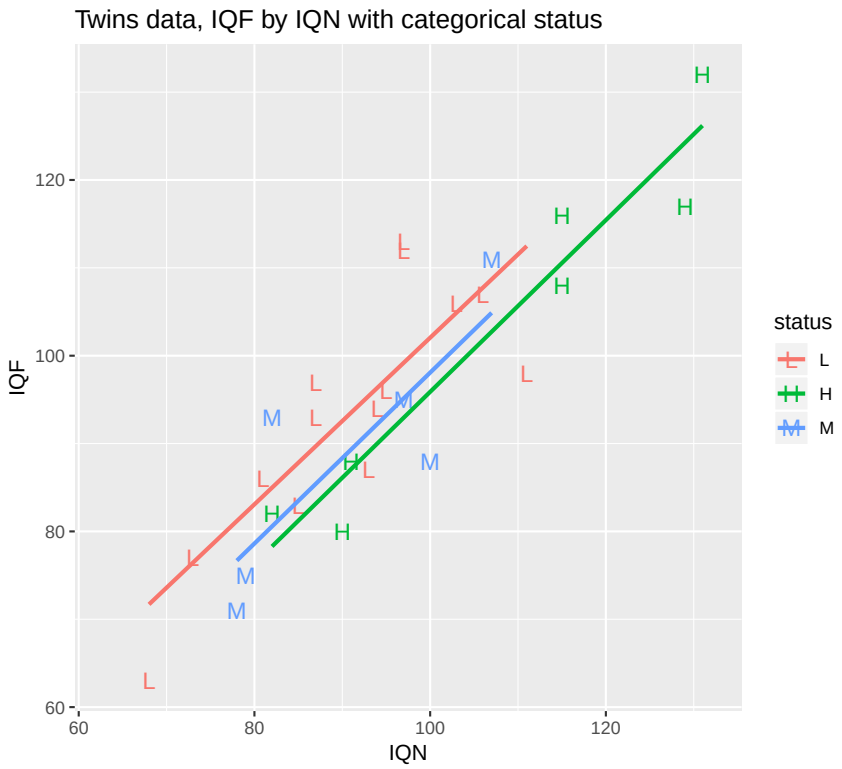


We fit the general model to the twins data.

```
#### Example: Twins
twins <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch07_twins.dat"
                    , header = TRUE)
# set "L" as baseline level
twins$status <- relevel(twins$status, "L")
str(twins)
```

```
## 'data.frame': 27 obs. of  3 variables:
##  $ IQF   : int  82 80 88 108 116 117 132 71 75 93 ...
##  $ IQN   : int  82 90 91 115 115 129 131 78 79 82 ...
##  $ status: Factor w/ 3 levels "L","H","M": 2 2 2 2 2 2 2 3 3 3 ...
```

```
library(ggplot2)
p <- ggplot(twins, aes(x = IQN, y = IQF, colour = status, shape = status))
p <- p + geom_point(size=4)
  library(R.oo) # for ascii code lookup
  p <- p + scale_shape_manual(values=charToInt(sort(unique(twins$status))))
p <- p + geom_smooth(method = lm, se = FALSE)
p <- p + labs(title="Twins data, IQF by IQN with categorical status")
  # equal axes since x- and y-variables are same quantity
  dat.range <- range(twins[,c("IQF","IQN")])
  p <- p + xlim(dat.range) + ylim(dat.range) + coord_equal(ratio=1)
print(p)
```



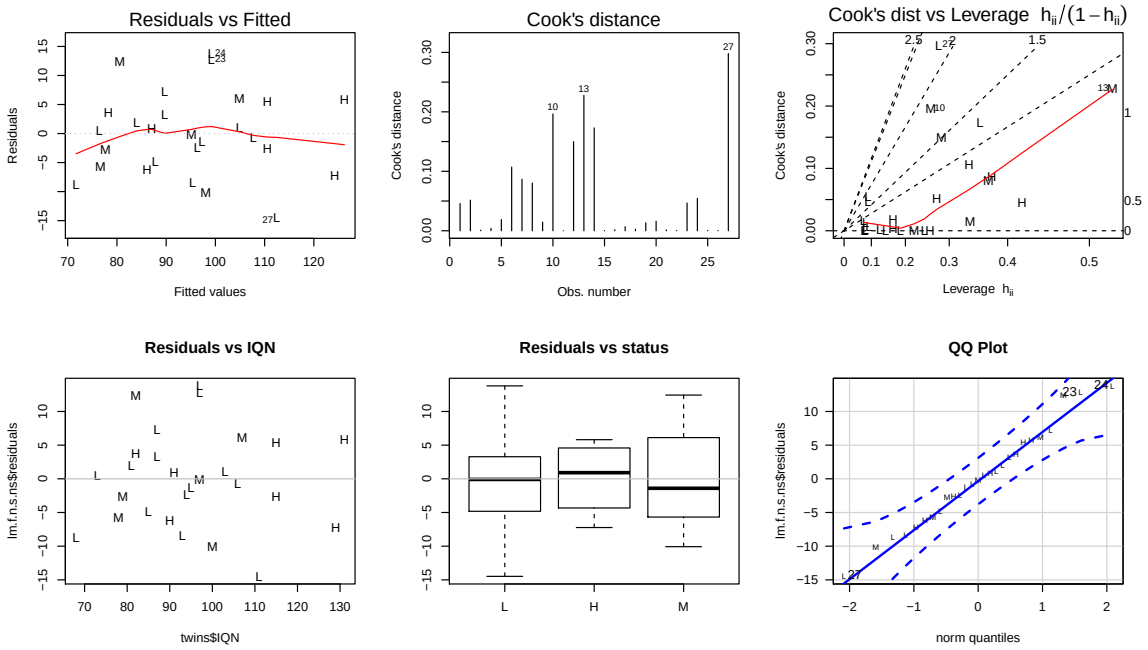

```
lm.f.n.s.ns <- lm(IQF ~ IQN*status, data = twins)
library(car)
Anova(aov(lm.f.n.s.ns), type=3)
## Anova Table (Type III tests)
##
## Response: IQF
##              Sum Sq Df F value   Pr(>F)
## (Intercept)   11.61  1  0.1850   0.6715
## IQN         1700.39  1 27.1035 3.69e-05 ***
## status         8.99  2  0.0716   0.9311
## IQN:status     0.93  2  0.0074   0.9926
## Residuals   1317.47 21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.f.n.s.ns)
##
## Call:
## lm(formula = IQF ~ IQN * status, data = twins)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -14.479  -5.248  -0.155   4.582  13.798
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.20461   16.75126   0.430    0.672
## IQN          0.94842    0.18218   5.206 3.69e-05 ***
## statusH     -9.07665   24.44870  -0.371    0.714
## statusM     -6.38859   31.02087  -0.206    0.839
## IQN:statusH  0.02914    0.24458   0.119    0.906
## IQN:statusM  0.02414    0.33933   0.071    0.944
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.921 on 21 degrees of freedom
## Multiple R-squared:  0.8041,Adjusted R-squared:  0.7574
## F-statistic: 17.24 on 5 and 21 DF,  p-value: 8.31e-07
```

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.f.n.s.ns, which = c(1,4,6), pch=as.character(twins$status))

plot(twins$IQN, lm.f.n.s.ns$residuals, main="Residuals vs IQN", pch=as.character(twins$status))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(twins$status, lm.f.n.s.ns$residuals, main="Residuals vs status")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.f.n.s.ns$residuals, las = 1, id = list(n = 3), main="QQ Plot", pch=as.character(twins$status))
```

```
## [1] 27 24 23
```

```
## residuals vs order of data
#plot(lm.f.n.s.ns$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



The natural way to express the fitted model is to give separate prediction equations for the three status groups. Here is an easy way to get the separate fits. For the general model (7.1), the predicted IQF satisfies

$$\begin{aligned}\text{Predicted IQF} &= (\text{Intercept} + \text{Coeff for Status Indicator}) \\ &\quad + (\text{Coeff for Status Product Effect} + \text{Coeff for IQN}) \times \text{IQN}.\end{aligned}$$

For the baseline group, use 0 as the coefficients for the status indicator and product effect.

Thus, for the baseline group with status = L,

$$\begin{aligned}\text{Predicted IQF} &= 7.20 + 0 + (0.948 + 0)\ \text{IQN} \\ &= 7.20 + 0.948\ \text{IQN}.\end{aligned}$$

For the M status group with indicator I_2 and product effect $I_2 \times$ IQN:

$$\begin{aligned}\text{Predicted IQF} &= 7.20 - 6.39 + (0.948 + 0.024)\ \text{IQN} \\ &= 0.81 + 0.972\ \text{IQN}.\end{aligned}$$

For the H status group with indicator I_1 and product effect $I_1 \times$ IQN:

$$\begin{aligned}\text{Predicted IQF} &= 7.20 - 9.08 + (0.948 + 0.029)\ \text{IQN} \\ &= -1.88 + 0.977\ \text{IQN}.\end{aligned}$$

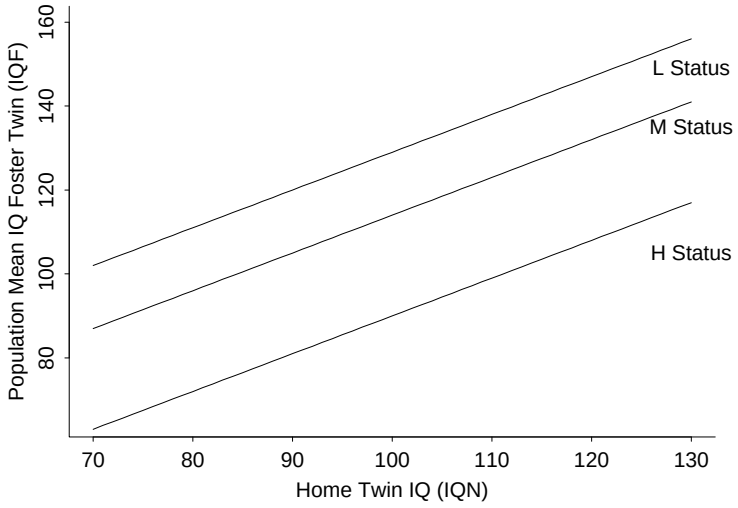
The LS lines are identical to separately fitting simple linear regressions to the three groups.

7.2.2 Equal slopes ANCOVA model

There are three other models of potential interest besides the general model. The **equal slopes** ANCOVA model

$$\text{IQF} \quad = \quad \beta_0 + \beta_1 I_1 + \beta_2 I_2 + \beta_3 \text{ IQN} + e$$

is a special case of (7.1) with $\beta_4 = \beta_5 = 0$ (no interaction). In the ANCOVA model, β_3 is the slope for all three regression lines. The other parameters have the same interpretation as in the general model (7.1), see the plot above. Output from the ANCOVA model is given below.




```
lm.f.n.s <- lm(IQF ~ IQN + status, data = twins)
library(car)
Anova(aov(lm.f.n.s), type=3)
```

```
## Anova Table (Type III tests)
##
## Response: IQF
##             Sum Sq Df F value    Pr(>F)
## (Intercept)   18.2  1  0.3181    0.5782
## IQN         4674.7  1 81.5521 5.047e-09 ***
## status       175.1  2  1.5276    0.2383
## Residuals   1318.4 23
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm.f.n.s)
```

```
##
## Call:
## lm(formula = IQF ~ IQN + status, data = twins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.8235  -5.2366  -0.1111   4.4755  13.6978
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.6188     9.9628   0.564    0.578
## IQN           0.9658     0.1069   9.031 5.05e-09 ***
## statusH      -6.2264     3.9171  -1.590    0.126
## statusM      -4.1911     3.6951  -1.134    0.268
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.571 on 23 degrees of freedom
## Multiple R-squared:  0.8039,Adjusted R-squared:  0.7784
## F-statistic: 31.44 on 3 and 23 DF,  p-value: 2.604e-08
```

For the ANCOVA model, the predicted IQF for the three groups satisfies

$$\begin{aligned}\text{Predicted IQF} &= (\text{Intercept} + \text{Coeff for Status Indicator}) \\ &\quad +(\text{Coeff for IQN}) \times \text{IQN}.\end{aligned}$$

As with the general model, use 0 as the coefficients for the status indicator and product effect for the baseline group.

For L status families:

$$\text{Predicted IQF} = 5.62 + 0.966 \text{ IQN},$$

for M status:

$$\begin{aligned}\text{Predicted IQF} &= 5.62 - 4.19 + 0.966 \text{ IQN} \\ &= 1.43 + 0.966 \text{ IQN},\end{aligned}$$

and for H status:

$$\begin{aligned}\text{Predicted IQF} &= 5.62 - 6.23 + 0.966 \text{ IQN} \\ &= -0.61 + 0.966 \text{ IQN}.\end{aligned}$$

7.2.3 Equal slopes and equal intercepts ANCOVA model

The model with **equal slopes** and **equal intercepts**

$$\text{IQF} = \beta_0 + \beta_3 \text{ IQN} + e$$

is a special case of the ANCOVA model with $\beta_1 = \beta_2 = 0$. This model does not distinguish among social classes. The common intercept and slope for the social classes are β_0 and β_3, respectively.

The predicted IQF for this model is

$$\text{IQF} \quad = \quad 9.21 + 0.901 \text{ IQN}$$

for each social class.

```
lm.f.n <- lm(IQF ~ IQN, data = twins)
#library(car)
#Anova(aov(lm.f.n), type=3)
summary(lm.f.n)
```

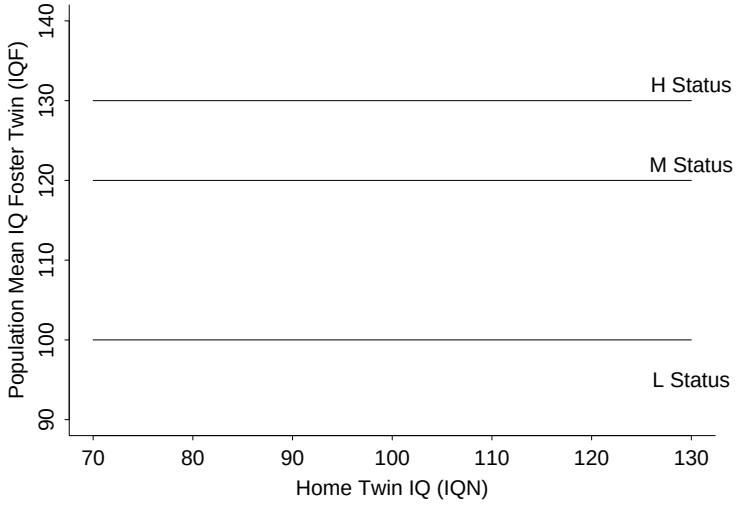
```
##
## Call:
## lm(formula = IQF ~ IQN, data = twins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3512  -5.7311   0.0574   4.3244  16.3531
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.20760    9.29990   0.990    0.332
## IQN          0.90144    0.09633   9.358  1.2e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.729 on 25 degrees of freedom
## Multiple R-squared:  0.7779,Adjusted R-squared:  0.769
## F-statistic: 87.56 on 1 and 25 DF,  p-value: 1.204e-09
```

7.2.4 No slopes, but intercepts ANCOVA model

The model with **no predictor** (IQN) effects

$$\text{IQF} \quad = \quad \beta_0 + \beta_1 I_1 + \beta_2 I_2 + e$$

is a special case of the ANCOVA model with $\beta_3 = 0$. In this model, social status has an effect on IQF but IQN does not. This model of **parallel regression lines** with **zero slopes** is identical to a one-way ANOVA model for the three social classes, where the intercepts play the role of the population means, see the plot below.



For the ANOVA model, the predicted IQF for the three groups satisfies

$$\text{Predicted IQF} \quad = \quad \text{Intercept} + \text{Coeff for Status Indicator}$$

Again, use 0 as the coefficients for the baseline status indicator.

For L status families:

$$\text{Predicted IQF} \quad = \quad 93.71,$$

for M status:

$$\begin{aligned} \text{Predicted IQF} &= 93.71 - 4.88 \\ &= 88.83, \end{aligned}$$

and for H status:

$$\begin{aligned} \text{Predicted IQF} &= 93.71 + 9.57 \\ &= 103.28. \end{aligned}$$

The predicted IQFs are the mean IQFs for the three groups.

```
lm.f.s <- lm(IQF ~ status, data = twins)
library(car)
Anova(aov(lm.f.s), type=3)
```

```
## Anova Table (Type III tests)
##
## Response: IQF
##             Sum Sq Df  F value Pr(>F)
## (Intercept) 122953  1 492.3772 <2e-16 ***
## status         732  2   1.4648 0.2511
## Residuals     5993 24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.f.s)
##
## Call:
## lm(formula = IQF ~ status, data = twins)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -30.714 -12.274   2.286  12.500  28.714
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   93.714      4.223  22.190   <2e-16 ***
## statusH        9.571      7.315   1.308    0.203
## statusM       -4.881      7.711  -0.633    0.533
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.8 on 24 degrees of freedom
## Multiple R-squared:  0.1088,Adjusted R-squared:  0.03452
## F-statistic: 1.465 on 2 and 24 DF,  p-value: 0.2511
```

7.3 Relating Models to Two-Factor ANOVA

Recall the multiple regression formulation of the general model (7.1):

$$\text{IQF} = \beta_0 + \beta_1 I_1 + \beta_2 I_2 + \beta_3\ \text{IQN} + \beta_4 I_1\ \text{IQN} + \beta_5 I_2\ \text{IQN} + e. \tag{7.2}$$

If you think of β_0 as a grand mean, $\beta_1 I_1 + \beta_2 I_2$ as the status effect (i.e., the two indicators I_1 and I_2 allow you to differentiate among social classes), β_3 IQN as the IQN effect and $\beta_4 I_1$ IQN $+ \beta_5 I_2$ IQN as the status by IQN interaction, then you can represent the model as

$$\begin{aligned} \text{IQF} = {} & \text{Grand Mean} + \text{Status Effect} + \text{IQN effect} \\ & + \text{Status} \times \text{IQN interaction} + \text{Residual}. \end{aligned} \tag{7.3}$$

This representation has the same form as a two-factor ANOVA model with interaction, except that IQN is a quantitative effect rather than a qualitative (i.e., categorical)

effect. The general model has the same structure as a two-factor interaction ANOVA model because the plot of the population means allows non-parallel profiles. However, the general model is a special case of the two-factor interaction ANOVA model because it restricts the means to change linearly with IQN.

The ANCOVA model has main effects for status and IQN but no interaction:

$$\text{IQF} \quad = \quad \text{Grand Mean} + \text{Status Effect} + \text{IQN effect} + \text{Residual}. \tag{7.4}$$

The ANCOVA model is a special case of the additive two-factor ANOVA model because the plot of the population means has parallel profiles, but is not equivalent to the additive two-factor ANOVA model.

The model with equal slopes and intercepts has no main effect for status nor an interaction between status and IQN:

$$\text{IQF} \quad = \quad \text{Grand Mean} + \text{IQN effect} + \text{Residual}. \tag{7.5}$$

The one-way ANOVA model has no main effect for IQN nor an interaction between status and IQN:

$$\text{IQF} \quad = \quad \text{Grand Mean} + \text{Status Effect} + \text{Residual}. \tag{7.6}$$

I will expand on these ideas later, as they are useful for understanding the connections between regression and ANOVA models.

7.4 Choosing Among Models

I will suggest a backward sequential method to select which of models (7.1), (7.4), and (7.5) fits best. You would typically be interested in the one-way ANOVA model (7.6) only when the effect of IQN was negligible.

Step 1: Fit the full model (7.1) and test the hypothesis of equal slopes $H_0 : \beta_4 = \beta_5 = 0$. (aside: t-tests are used to test **either** $\beta_4 = 0$ or $\beta_5 = 0$.) To test H_0, eliminate the predictor variables I_1 IQN and I_2 IQN associated with β_4 and β_5 from the full model (7.1). Then fit the reduced model (7.4) with equal slopes. Reject $H_0 : \beta_4 = \beta_5 = 0$ if the increase in the Residual SS obtained by deleting I_1 IQN and I_2 IQN from the full model is significant. Formally, compute the F-statistic:

$$F_{obs} = \frac{(\text{ERROR SS for reduced model} - \text{ERROR SS for full model})/2}{\text{ERROR MS for full model}}$$

and compare it to an upper-tail critical value for an F-distribution with 2 and df degrees of freedom, where df is the Residual df for the full model. The F-test is a direct extension of the single degree-of-freedom F-tests in the stepwise fits. A p-value

for F-test is obtained from `library(car)` with `Anova(aov(LMOBJECT), type=3)` for the interaction. If H_0 is rejected, stop and conclude that the population regression lines have different slopes (and then I do not care whether the intercepts are equal). Otherwise, proceed to step 2.
Step 2: Fit the equal slopes or ANCOVA model (7.4) and test for equal intercepts $H_0 : \beta_1 = \beta_2 = 0$. Follow the procedure outlined in Step 1, treating the ANCOVA model as the full model and the model $IQF = \beta_0 + \beta_3$ IQN$+e$ with equal slopes and intercepts as the reduced model. See the intercept term using `library(car)` with `Anova(aov(LMOBJECT), type=3)`. If H_0 is rejected, conclude that that population regression lines are parallel with unequal intercepts. Otherwise, conclude that regression lines are identical.
Step 3: Estimate the parameters under the appropriate model, and conduct a diagnostic analysis. Summarize the fitted model by status class.

A comparison of regression lines across $k > 3$ groups requires $k - 1$ indicator variables to define the groups, and $k - 1$ interaction variables, assuming the model has a single predictor. The comparison of models mimics the discussion above, except that the numerator of the F-statistic is divided by $k-1$ instead of 2, and the numerator df for the F-test is $k - 1$ instead of 2. If $k = 2$, the F-tests for comparing the three models are equivalent to $t-$tests given with the parameter estimates summary. For example, recall how you tested for equal intercepts in the tools problems.

The plot of the twins data shows fairly linear relationships within each social class. The linear relationships appear to have similar slopes and similar intercepts. The p-value for testing the hypothesis that the slopes of the population regression lines are equal is essentially 1. The observed data are consistent with the reduced model of equal slopes.

The p-value for comparing the model of equal slopes and equal intercepts to the ANCOVA model is 0.238, so there is insufficient evidence to reject the reduced model with equal slopes and intercepts. The estimated regression line, regardless of social class, is:

$$\text{Predicted IQF} = 9.21 + 0.901\text{*IQN}.$$

There are no serious inadequacies with this model, based on a diagnostic analysis (not shown).

An interpretation of this analysis is that the natural parents' social class has no impact on the relationship between the IQ scores of identical twins raised apart. What other interesting features of the data would be interesting to explore? For example, what values of the intercept and slope of the population regression line are of intrinsic interest?

7.4.1 Simultaneous testing of regression parameters

In the twins example, we have this full interaction model,

$$\text{IQF} \quad = \quad \beta_0 + \beta_1 I_1 + \beta_2 I_2 + \beta_3 \text{ IQN} + \beta_4 I_1 \text{ IQN} + \beta_5 I_2 \text{ IQN} + e, \qquad (7.7)$$

where $I_1 = 1$ indicates H, and $I_2 = 1$ indicates M, and L is the baseline status.

Consider these two specific hypotheses:

1. H_0 : equal regression lines for status M and L
2. H_0 : equal regression lines for status M and H

That is, the intercept and slope for the regression lines are equal for the pairs of status groups.

First, it is necessary to formulate these hypotheses in terms of testable parameters. That is, find the β values that make the null hypothesis true in terms of the model equation.

1. $H_0 : \beta_2 = 0$ and $\beta_5 = 0$
2. $H_0 : \beta_1 = \beta_2$ and $\beta_4 = \beta_5$

Using linear model theory, there are methods for testing these multiple-parameter hypothesis tests.

One strategy is to use the Wald test of null hypothesis $\mathbf{r}\underset{\sim}{\beta} = \underset{\sim}{r}$, where $\mathbf{r}$ is a matrix of contrast coefficients (typically $+1$ or -1), $\underset{\sim}{\beta}$ is our vector of regression β coefficients, and $\underset{\sim}{r}$ is a hypothesized vector of what the linear system $\mathbf{r}\underset{\sim}{\beta}$ equals. For our first hypothesis test, the linear system we're testing in matrix notation is

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Let's go about testing another hypothesis, first, using the Wald test, then we'll test our two simultaneous hypotheses above.

- H_0 : equal slopes for all status groups
- $H_0 : \beta_4 = \beta_5 = 0$

```
lm.f.n.s.ns <- lm(IQF ~ IQN*status, data = twins)
library(car)
Anova(aov(lm.f.n.s.ns), type=3)
## Anova Table (Type III tests)
##
## Response: IQF
##              Sum Sq Df F value   Pr(>F)
```

```
## (Intercept)   11.61  1  0.1850   0.6715
## IQN         1700.39  1 27.1035 3.69e-05 ***
## status         8.99  2  0.0716   0.9311
## IQN:status     0.93  2  0.0074   0.9926
## Residuals   1317.47 21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# beta coefficients (term positions: 1, 2, 3, 4, 5, 6)
coef(lm.f.n.s.ns)
```

```
## (Intercept)         IQN     statusH     statusM IQN:statusH
##  7.20460986  0.94842244 -9.07665352 -6.38858548  0.02913971
## IQN:statusM
##  0.02414450
```

The test for the interaction above (IQN:status) has a p-value=0.9926, which indicates that common slope is reasonable. In the Wald test notation, we want to test whether those last two coefficients (term positions 5 and 6) both equal 0. Here we get the same result as the ANOVA table.

```
library(aod) # for wald.test()
```

```
##
## Attaching package:  'aod'
## The following object is masked _by_ '.GlobalEnv':
##
##    mice
## The following object is masked from 'package:survival':
##
##    rats
```

```
# Typically, we are interested in testing whether individual parameters or
#   set of parameters are all simultaneously equal to 0s
# However, any null hypothesis values can be included in the vector coef.test.values.
coef.test.values <- rep(0, length(coef(lm.f.n.s.ns)))
wald.test(b = coef(lm.f.n.s.ns) - coef.test.values
        , Sigma = vcov(lm.f.n.s.ns)
        , Terms = c(5,6))
```

```
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 0.015, df = 2, P(> X2) = 0.99
```

Now to our two simultaneous hypotheses. In hypothesis 1 we are testing $\beta_2 = 0$ and $\beta_5 = 0$, which are the 3rd and 6th position for coefficients in our original equation (7.7). *However, we need to choose the correct positions based on the* `coef()` *order, and these are positions 4 and 6.* The large p-value=0.55 suggests that M and L can be described by the same regression line, same slope and intercept.

```
library(aod) # for wald.test()
coef.test.values <- rep(0, length(coef(lm.f.n.s.ns)))
wald.test(b = coef(lm.f.n.s.ns) - coef.test.values
```

```
        , Sigma = vcov(lm.f.n.s.ns)
        , Terms = c(4,6))
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 1.2, df = 2, P(> X2) = 0.55
# Another way to do this is to define the matrix r and vector r, manually.
mR <- as.matrix(rbind(c(0, 0, 0, 1, 0, 0), c(0, 0, 0, 0, 0, 1)))
mR
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    0    0    0    1    0    0
## [2,]    0    0    0    0    0    1
vR <- c(0, 0)
vR
## [1] 0 0
wald.test(b = coef(lm.f.n.s.ns)
        , Sigma = vcov(lm.f.n.s.ns)
        , L = mR, H0 = vR)
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 1.2, df = 2, P(> X2) = 0.55
```

In hypothesis 2 we are testing $\beta_1 - \beta_2 = 0$ and $\beta_4 - \beta_5 = 0$ which are the difference of the 2nd and 3rd coefficients and the difference of the 5th and 6th coefficients. *However, we need to choose the correct positions based on the* `coef()` *order, and these are positions 3 and 4, and 5 and 6.* The large p-value=0.91 suggests that M and H can be described by the same regression line, same slope and intercept.

```
mR <- as.matrix(rbind(c(0, 0, 1, -1, 0, 0), c(0, 0, 0, 0, 1, -1)))
mR
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    0    0    1   -1    0    0
## [2,]    0    0    0    0    1   -1
vR <- c(0, 0)
vR
## [1] 0 0
wald.test(b = coef(lm.f.n.s.ns)
        , Sigma = vcov(lm.f.n.s.ns)
        , L = mR, H0 = vR)
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 0.19, df = 2, P(> X2) = 0.91
```

The results of these tests are not surprising, given our previous analysis where we found that the status effect is not significant for all three groups.

Any simultaneous linear combination of parameters can be tested in this way.

7.5 Comments on Comparing Regression Lines

In the twins example, I defined two indicator variables (plus two interaction variables) from an ordinal categorical variable: status (H, M, L). Many researchers would assign numerical codes to the status groups and use the coding as a predictor in a regression model. For status, a "natural" coding might be to define NSTAT=0 for L, 1 for M, and 2 for H status families. This suggests building a multiple regression model with a single status variable (i.e., single df):

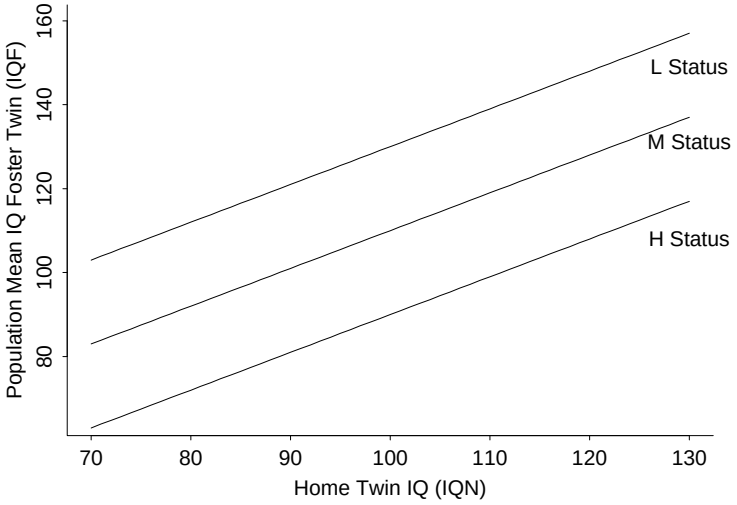
$$\text{IQF} = \beta_0 + \beta_1 \text{ IQN} + \beta_2 \text{NSTAT} + e.$$

If you consider the status classes separately, the model implies that

$$\begin{array}{lll}
\text{IQF} = \beta_0 + \beta_1 \text{ IQN} + \beta_2(0) + e = \beta_0 + \beta_1 \text{ IQN} + e & & \text{for L status,} \\
\text{IQF} = \beta_0 + \beta_1 \text{ IQN} + \beta_2(1) + e = (\beta_0 + \beta_2) + \beta_1 \text{ IQN} + e & & \text{for M status,} \\
\text{IQF} = \beta_0 + \beta_1 \text{ IQN} + \beta_2(2) + e = (\beta_0 + 2\beta_2) + \beta_1 \text{ IQN} + e & & \text{for H status.}
\end{array}$$

The model assumes that the IQF by IQN regression lines are parallel for the three groups, and are separated by a constant β_2. This model is more restrictive (and less reasonable) than the ANCOVA model with equal slopes but arbitrary intercepts. Of course, this model is a easier to work with because it requires keeping track of only one status variable instead of two status indicators.

A plot of the population regression lines under this model is given above, assuming $\beta_2 < 0$.



7.6 Three-way interaction

In this example, a three-way interaction is illustrated with two categorical variables and one continuous variable. Let a take values 0 or 1 (it's an indicator variable), b take values 0 or 1, and c be a continuous variable taking any value.

Below are five models:

(1) Interactions: ab. All lines parallel, different intercepts for each (a, b) combination.
(2) Interactions: ab, ac. (a, c) combinations have parallel lines, different intercepts for each (a, b) combination.
(3) Interactions: ab, bc. (b, c) combinations have parallel lines, different intercepts for each (a, b) combination.
(4) Interactions: ab, ac, bc. All combinations may have different slope lines with different intercepts, but difference in slope between $b = 0$ and $b = 1$ is similar for each a group (and vice versa).
(5) Interactions: ab, ac, bc, abc. All combinations may have different slope lines with different intercepts.

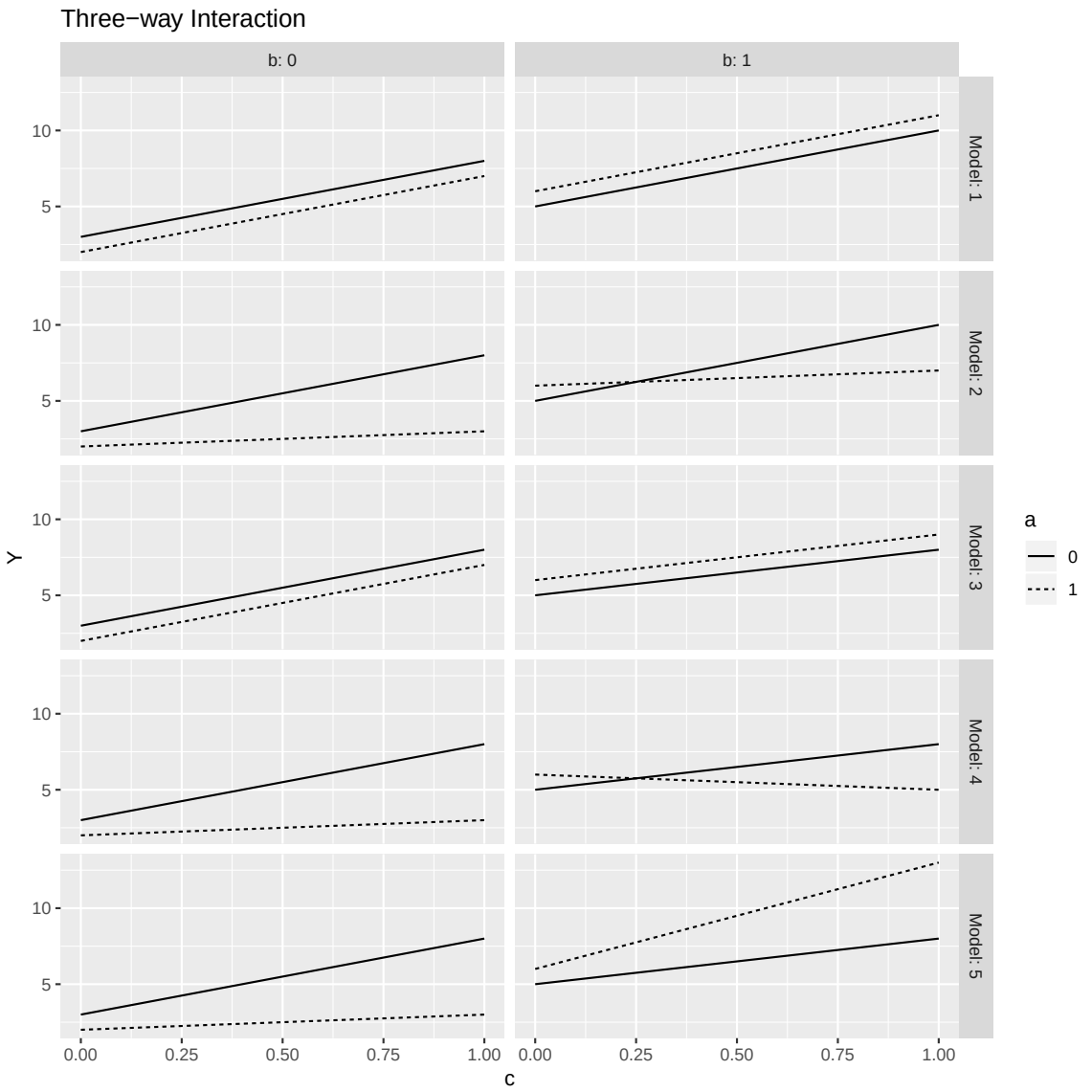
Model		Intercepts				Slopes for c			
(1)	$y =$	β_0	$+\beta_1 a$	$+\beta_2 b$	$+\beta_3 ab$	$+\beta_4 c$			
(2)	$y =$	β_0	$+\beta_1 a$	$+\beta_2 b$	$+\beta_3 ab$	$+\beta_4 c$	$+\beta_5 ac$		
(3)	$y =$	β_0	$+\beta_1 a$	$+\beta_2 b$	$+\beta_3 ab$	$+\beta_4 c$		$+\beta_6 bc$	
(4)	$y =$	β_0	$+\beta_1 a$	$+\beta_2 b$	$+\beta_3 ab$	$+\beta_4 c$	$+\beta_5 ac$	$+\beta_6 bc$	
(5)	$y =$	β_0	$+\beta_1 a$	$+\beta_2 b$	$+\beta_3 ab$	$+\beta_4 c$	$+\beta_5 ac$	$+\beta_6 bc$	$+\beta_7 abc$

```
X <- expand.grid(c(0,1),c(0,1),c(0,1))
X <- cbind(1, X)
colnames(X) <- c("one", "a", "b", "c")
X$ab  <- X$a * X$b
X$ac  <- X$a * X$c
X$bc  <- X$b * X$c
X$abc <- X$a * X$b * X$c
X <- as.matrix(X)
X <- X[,c(1,2,3,5,4,6,7,8)] # reorder columns to be consistent with table above
#£

vbeta <- matrix(c(3, -1, 2, 2, 5, -4, -2, 8), ncol = 1)
rownames(vbeta) <- paste("beta", 0:7, sep="")

Beta <- matrix(vbeta, nrow = dim(vbeta)[1], ncol = 5)
rownames(Beta) <- rownames(vbeta)

# Beta vector for each model
Beta[c(6,7,8), 1] <- 0
Beta[c(  7,8), 2] <- 0
Beta[c(6,  8), 3] <- 0
Beta[c(    8), 4] <- 0

colnames(Beta) <- 1:5 #paste("model", 1:5, sep="")

# Calculate response values
Y <- X %*% Beta

library(reshape2)
YX <- data.frame(cbind(melt(Y), X[,"a"], X[,"b"], X[,"c"]))
colnames(YX) <- c("obs", "Model", "Y", "a", "b", "c")
YX$a <- factor(YX$a)
YX$b <- factor(YX$b)
```

These are the β values used for this example.

beta0	beta1	beta2	beta3	beta4	beta5	beta6	beta7
3	−1	2	2	5	−4	−2	8

```
library(ggplot2)
p <- ggplot(YX, aes(x = c, y = Y, group = a))
#p <- p + geom_point()
p <- p + geom_line(aes(linetype = a))
p <- p + labs(title = "Three-way Interaction")
p <- p + facet_grid(Model ~ b, labeller = "label_both")
print(p)
```


Three−way Interaction
b: 0
b: 1
Model: 1
Model: 2
Model: 3
Model: 4
Model: 5
Y
c
a
0
1
10
5
0.00
0.25
0.50
0.75
1.00

Chapter 8

Polynomial Regression

8.1 Polynomial Models with One Predictor

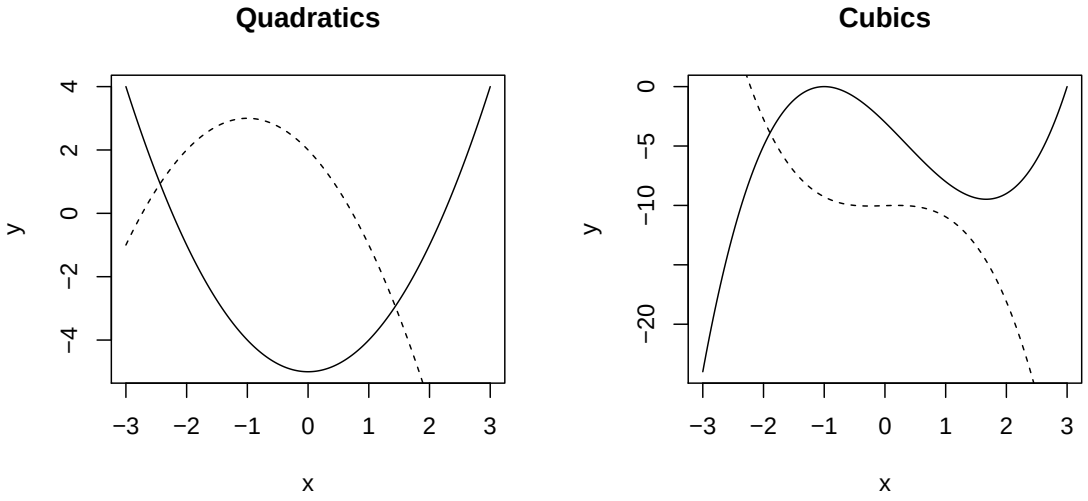
A p^{th} order polynomial model relating a dependent variable Y to a predictor X is given by

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_p X^p + \varepsilon.$$

This is a multiple regression model with predictors $X, X^2, \ldots, X^p$. For $p = 2$, 3, 4, and 5 we have quadratic, cubic, quartic and quintic relationships, respectively.

A second order polynomial (quadratic) allows at most one local maximum or minimum (i.e., a point where trend changes direction from increasing to decreasing, or from decreasing to increasing). A third order polynomial (cubic) allows at most two local maxima or minima. In general, a p^{th} order polynomial allows at most $p-1$ local maxima or minima. The two panels below illustrate different quadratic and cubic relationships.

```
#### Creating polynomial plots
# R code for quadratic and cubic plots
x <- seq(-3,3,0.01);
y21 <- x^2-5;
y22 <- -(x+1)^2+3;
y31 <- (x+1)^2*(x-3);
y32 <- -(x-.2)^2*(x+.5)-10;

plot(  x, y21, type="l", main="Quadratics", ylab="y")
points(x, y22, type="l", lt=2)
plot(  x, y31, type="l", main="Cubics", ylab="y")
points(x, y32, type="l", lt=2)
```



It is important to recognize that not all, or even none, of the "turning-points" in a polynomial may be observed if the range of X is suitably restricted.

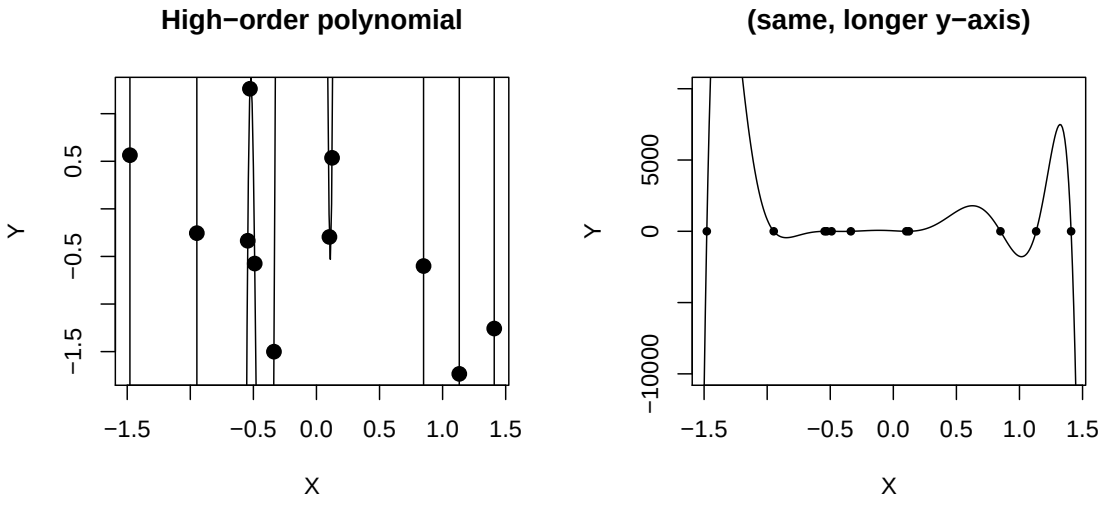
Although polynomial models allow a rich class of non-linear relationships between Y and X (by virtue of Taylor's Theorem in calculus), some caution is needed when fitting polynomials. In particular, the extreme X-values can be highly influential, numerical instabilities occur when fitting high order models, and predictions based on high order polynomial models can be woeful.

To illustrate the third concern, consider a data set (Y_i, X_i) for $i = 1, 2, \ldots, n$ where the X_is are distinct. One can show mathematically that an $(n-1)^{st}$ degree polynomial will fit the observed data exactly. However, for a high order polynomial to fit exactly, the fitted curve must oscillate wildly between data points. In the picture below, I show the 10^{th} degree polynomial that fits exactly the 11 distinct data points. Although $R^2 = 1$, I would not use the fitted model to make predictions with new data. (If possible, models should always be validated using new data.) Intuitively, a quadratic or a lower order polynomial would likely be significantly better. In essence, the 10^{th} degree polynomial is modelling the variability in the data, rather than the trend.

```
# R code for quadratic and cubic plots

X <- rnorm(11); Y <- rnorm(11);  # observed
X1  <- X^1 ;
X2  <- X^2 ;
X3  <- X^3 ;
X4  <- X^4 ;
X5  <- X^5 ;
X6  <- X^6 ;
X7  <- X^7 ;
X8  <- X^8 ;
X9  <- X^9 ;
X10 <- X^10;

fit <- lm(Y~X+X2+X3+X4+X5+X6+X7+X8+X9+X10)
fit$coefficients
##  (Intercept)           X          X2          X3          X4
```

```
##     36.70206   -461.55109   -620.55094  13030.85848  29149.14342
##           X5           X6           X7           X8           X9
## -26416.29553 -81282.20211  15955.10270  70539.53467  -3396.10960
##          X10
## -18290.46769
```

```
x <- seq(-2.5,2.5,0.01);
x1  <- x^1 ;
x2  <- x^2 ;
x3  <- x^3 ;
x4  <- x^4 ;
x5  <- x^5 ;
x6  <- x^6 ;
x7  <- x^7 ;
x8  <- x^8 ;
x9  <- x^9 ;
x10 <- x^10;

xx <- matrix(c(rep(1,length(x)),x1,x2,x3,x4,x5,x6,x7,x8,x9,x10),ncol=11)

y <- xx %*% fit$coefficients;

plot(  X, Y, main="High-order polynomial", pch=20, cex=2)
points(x, y, type="l", lt=1)
plot(  X, Y, main="(same, longer y-axis)", pch=20, cex=1, ylim=c(-10000,10000))
points(x, y, type="l", lt=1)
```



Another concern is that the importance of lower order terms (i.e., X, X^2, ..., X^{p-1}) depends on the scale in which we measure X. For example, suppose for some chemical reaction,

$$\text{Time to reaction} = \beta_0 + \beta_1 \text{ Temp} + \beta_2 \text{ Temp}^2 + \varepsilon.$$

The significance level for the estimate of the Temp coefficient depends on whether we measure temperature in degrees Celsius or Fahrenheit.

To avoid these problems, I recommend the following:

1. Center the X data at $\bar{X}$ and fit the model

$$Y = \beta_0 + \beta_1(X - \bar{X}) + \beta_2(X - \bar{X})^2 + \cdots + \beta_p(X - \bar{X})^p + \varepsilon.$$

This is usually important only for cubic and higher order models.

2. Restrict attention to low order models, say quartic or less. If a fourth-order polynomial does not fit, a transformation may provide a more succinct summary.
3. Pay careful attention to diagnostics.
4. Add or delete variables using the natural hierarchy among powers of X and include all lower order terms if a higher order term is needed. For example, in a forward-selection type algorithm, add terms X, X^2, ..., sequentially until no additional term is significant, but do not delete powers that were entered earlier. Similarly, with a backward-elimination type algorithm, start with the model of maximum acceptable order (for example a fourth or third order polynomial) and consider deleting terms in the order X^p, X^{p-1}, ..., until no further terms can be omitted. The **select=backward** option in the **reg** procedure **does not allow** you to invoke the hierarchy principle with backward elimination. The backward option sequentially eliminates the least significant effect in the model, regardless of what other effects are included.

8.1.1 Example: Cloud point and percent I-8

The cloud point of a liquid is a measure of the degree of crystallization in a stock, and is measured by the refractive index [1]. It has been suggested that the percent of I-8 (variable "i8") in the base stock is an excellent predictor of the cloud point using a second order (quadratic) model:

$$\text{Cloud point} = \beta_0 + \beta_1 \text{ I8} + \beta_2 \text{ I8}^2 + \varepsilon.$$

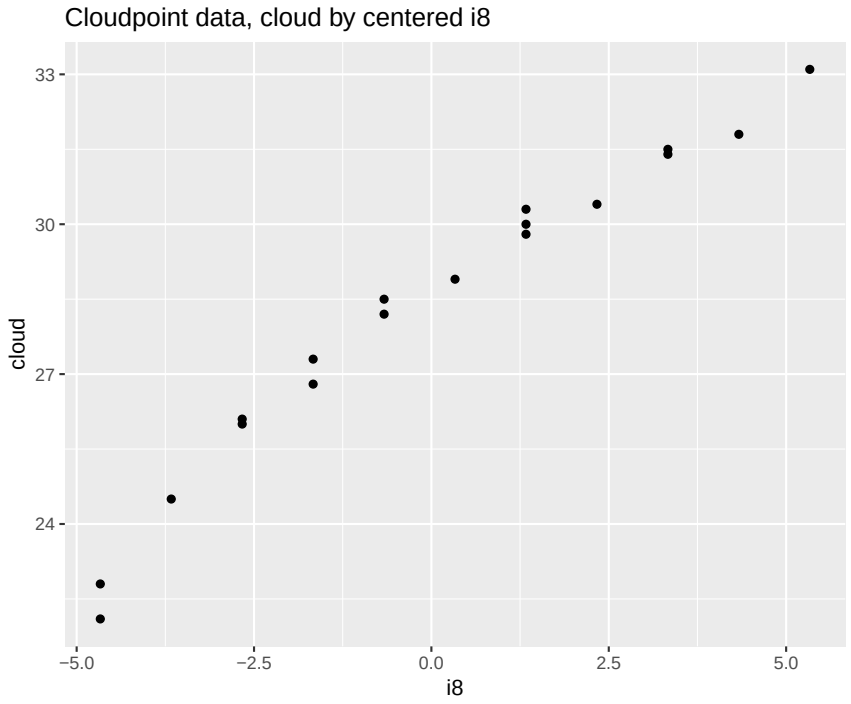
Data were collected to examine this model.

```
#### Example: Cloud point
cloudpoint <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch08_cloudpoint.dat"
                         , header = TRUE)
# center i8 by subracting the mean
cloudpoint$i8 <- cloudpoint$i8 - mean(cloudpoint$i8)
```

The plot of the data suggest a departure from a linear relationship.

```
library(ggplot2)
p <- ggplot(cloudpoint, aes(x = i8, y = cloud))
p <- p + geom_point()
p <- p + labs(title="Cloudpoint data, cloud by centered i8")
print(p)
```

[1] Draper and Smith 1966, p. 162

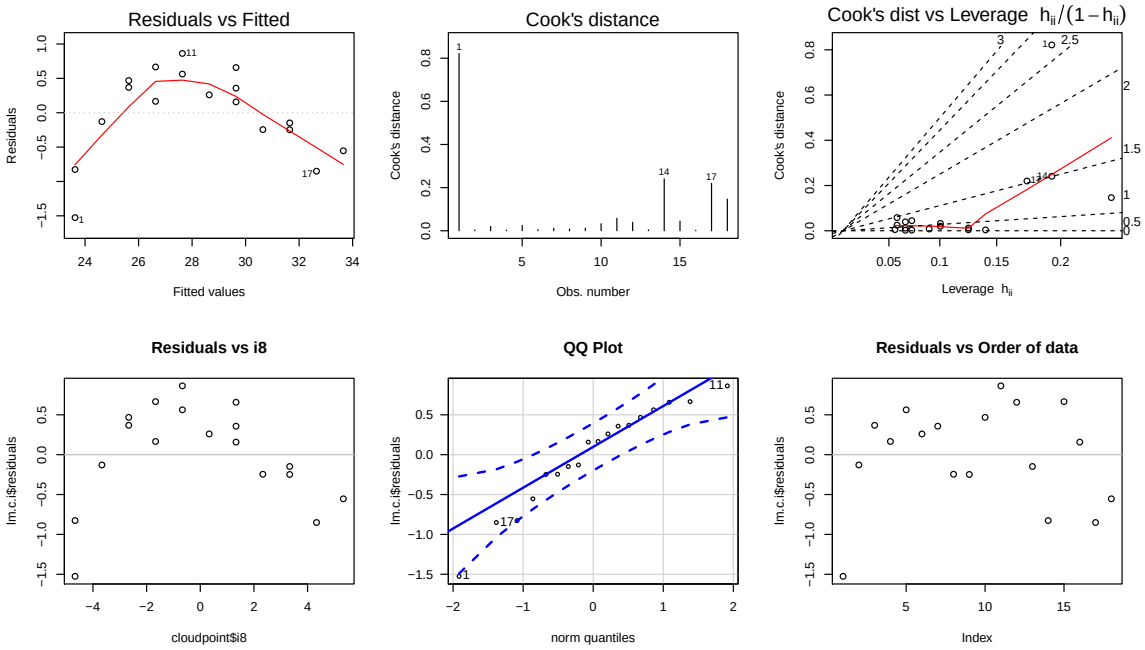


Fit the simple linear regression model and plot the residuals.

```
lm.c.i <- lm(cloud ~ i8, data = cloudpoint)
#library(car)
#Anova(aov(lm.c.i), type=3)
#summary(lm.c.i)
```

The data plot is clearly nonlinear, suggesting that a simple linear regression model is inadequate. This is confirmed by a plot of the studentized residuals against the fitted values from a simple linear regression of Cloud point on i8. Also by the residuals against the i8 values. We do not see any local maxima or minima, so a second order model is likely to be adequate. To be sure, we will first fit a cubic model, and see whether the third order term is important.

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.c.i, which = c(1,4,6), pch=as.character(cloudpoint$type))

plot(cloudpoint$i8, lm.c.i$residuals, main="Residuals vs i8", pch=as.character(cloudpoint$type))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.c.i$residuals, las = 1, id = list(n = 3), main="QQ Plot", pch=as.character(cloudpoint$type))
## [1]  1 11 17

# residuals vs order of data
plot(lm.c.i$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



The output below shows that the cubic term improves the fit of the quadratic model (i.e., the cubic term is important when added last to the model). The plot of the studentized residuals against the fitted values does not show any extreme abnormalities. Furthermore, no individual point is poorly fitted by the model. Case 1 has the largest studentized residual: $r_1 = -1.997$.

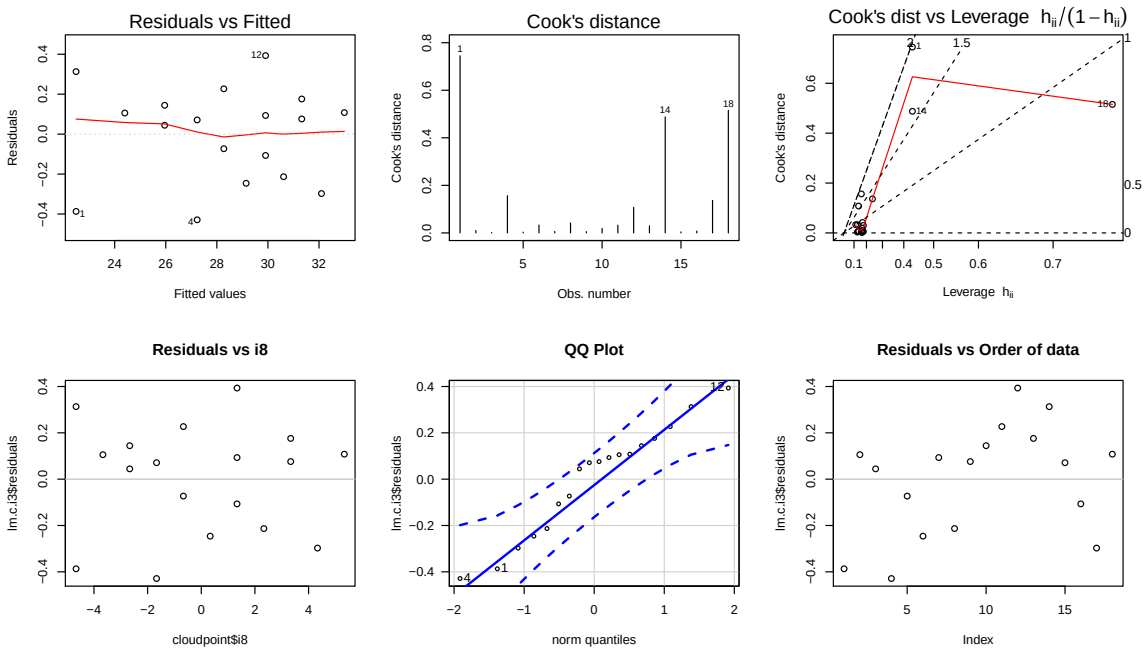
```
# I() is used to create an interpreted object treated "as is"
#   so we can include quadratic and cubic terms in the formula
#   without creating separate columns in the dataset of these terms
lm.c.i3 <- lm(cloud ~ i8 + I(i8^2) + I(i8^3), data = cloudpoint)
#library(car)
#Anova(aov(lm.c.i3), type=3)
summary(lm.c.i3)
```

```
##
## Call:
## lm(formula = cloud ~ i8 + I(i8^2) + I(i8^3), data = cloudpoint)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.42890 -0.18658  0.07355  0.13536  0.39328
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 28.870451   0.088364 326.723  < 2e-16 ***
## i8           0.847889   0.048536  17.469 6.67e-11 ***
## I(i8^2)     -0.065998   0.007323  -9.012 3.33e-07 ***
```

```
## I(i8^3)      0.009735   0.002588   3.762   0.0021 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2599 on 14 degrees of freedom
## Multiple R-squared:  0.9943,Adjusted R-squared:  0.9931
## F-statistic: 812.9 on 3 and 14 DF,  p-value: 6.189e-16
```

Below are plots of the data and the studentized residuals.

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.c.i3, which = c(1,4,6), pch=as.character(cloudpoint$type))

plot(cloudpoint$i8, lm.c.i3$residuals, main="Residuals vs i8", pch=as.character(cloudpoint$type))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.c.i3$residuals, las = 1, id = list(n = 3), main="QQ Plot", pch=as.character(cloudpoint$type))
## [1]  4 12  1
# residuals vs order of data
plot(lm.c.i3$residuals, main="Residuals vs Order of data")
  # horizontal line at zero
  abline(h = 0, col = "gray75")
```



The first and last observations have the lowest and highest values of I8, given by 0 and 10, respectively. These cases are also the most influential points in the data set (largest Cook's D). If we delete these cases and redo the analysis we find that the cubic term is no longer important (p-value=0.55) when added after the quadratic term. One may reasonably conclude that the significance of the cubic term in the original analysis is solely due to the two extreme I8 values, and that the quadratic

model appears to fit well over the smaller range of $1 \leq \text{I8} \leq 9$.

```
# remove points for minimum and maximum i8 values
cloudpoint2 <- cloudpoint[!(cloudpoint$i8 == min(cloudpoint$i8) |
                            cloudpoint$i8 == max(cloudpoint$i8)), ]
lm.c.i2 <- lm(cloud ~ i8 + I(i8^2) + I(i8^3), data = cloudpoint2)
#library(car)
#Anova(aov(lm.c.i2), type=3)
summary(lm.c.i2)
```

```
##
## Call:
## lm(formula = cloud ~ i8 + I(i8^2) + I(i8^3), data = cloudpoint2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36620 -0.12845  0.03737  0.14031  0.33737
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 28.857039   0.089465 322.551  < 2e-16 ***
## i8           0.904515   0.058338  15.505 8.04e-09 ***
## I(i8^2)     -0.060714   0.012692  -4.784 0.000568 ***
## I(i8^3)      0.003168   0.005166   0.613 0.552200
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2313 on 11 degrees of freedom
## Multiple R-squared:  0.9917,Adjusted R-squared:  0.9894
## F-statistic: 436.3 on 3 and 11 DF,  p-value: 1.032e-11
```

8.2 Polynomial Models with Two Predictors

Polynomial models are sometimes fit to data collected from experiments with two or more predictors. For simplicity, consider the general quadratic model, with two predictors:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 + \varepsilon.$$

The model, which can be justified as a second order approximation to a smooth trend, includes quadratic terms in X_1 and X_2 and the product or interaction of X_1 and X_2.

8.2.1 Example: Mooney viscosity

The data below give the Mooney viscosity at 100 degrees Celsius (Y) as a function of the filler level (X_1) and the naphthenic oil (X_2) level for an experiment involving filled and plasticized elastomer compounds.

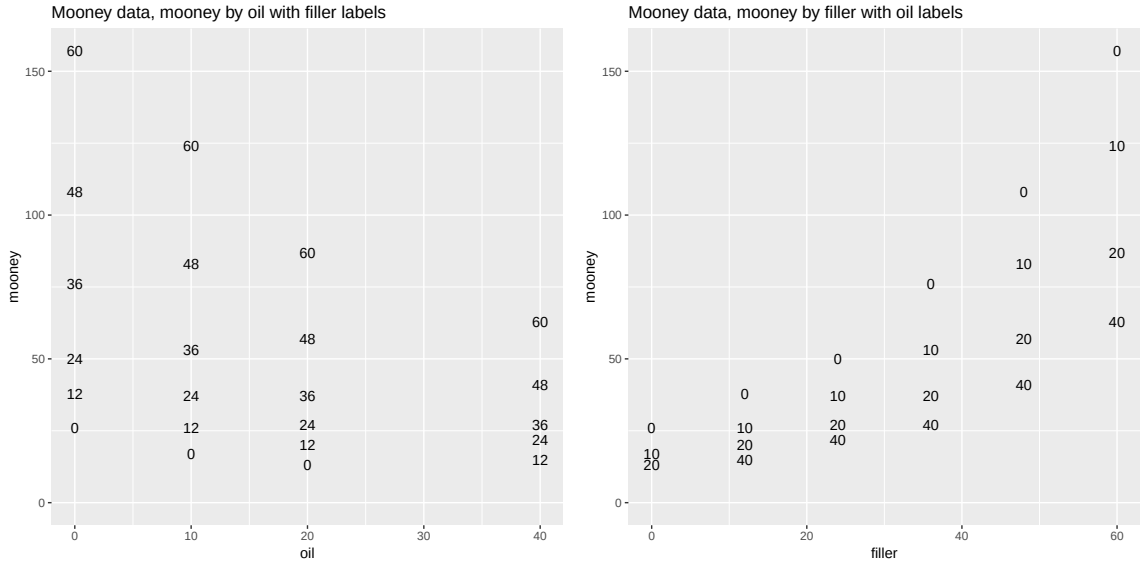
```
#### Example: Mooney viscosity
mooney <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch08_mooney.dat"
                     , header = TRUE)
```

```
library(ggplot2)
p <- ggplot(mooney, aes(x = oil, y = mooney, label = filler))
p <- p + geom_text()
p <- p + scale_y_continuous(limits = c(0, max(mooney$mooney, na.rm=TRUE)))
p <- p + labs(title="Mooney data, mooney by oil with filler labels")
print(p)
```

```
## Warning:  Removed 1 rows containing missing values (geom_text).
```

```
library(ggplot2)
p <- ggplot(mooney, aes(x = filler, y = mooney, label = oil))
p <- p + geom_text()
p <- p + scale_y_continuous(limits = c(0, max(mooney$mooney, na.rm=TRUE)))
p <- p + labs(title="Mooney data, mooney by filler with oil labels")
print(p)
```

```
## Warning:  Removed 1 rows containing missing values (geom_text).
```



At each of the 4 oil levels, the relationship between the Mooney viscosity and filler level (with 6 levels) appears to be quadratic. Similarly, the relationship between the Mooney viscosity and oil level appears quadratic for each filler level (with 4 levels). This supports fitting the general quadratic model as a first step in the analysis.

The output below shows that each term is needed in the model. Although there are potentially influential points (cases 6 and 20), deleting either or both cases does not change the significance of the effects in the model (not shown).

```
# I create each term separately
lm.m.o2.f2 <- lm(mooney ~ oil + filler + I(oil^2) + I(filler^2) + I(oil * filler),
                 data = mooney)
summary(lm.m.o2.f2)
```

```
##
## Call:
## lm(formula = mooney ~ oil + filler + I(oil^2) + I(filler^2) +
##     I(oil * filler), data = mooney)
##
```

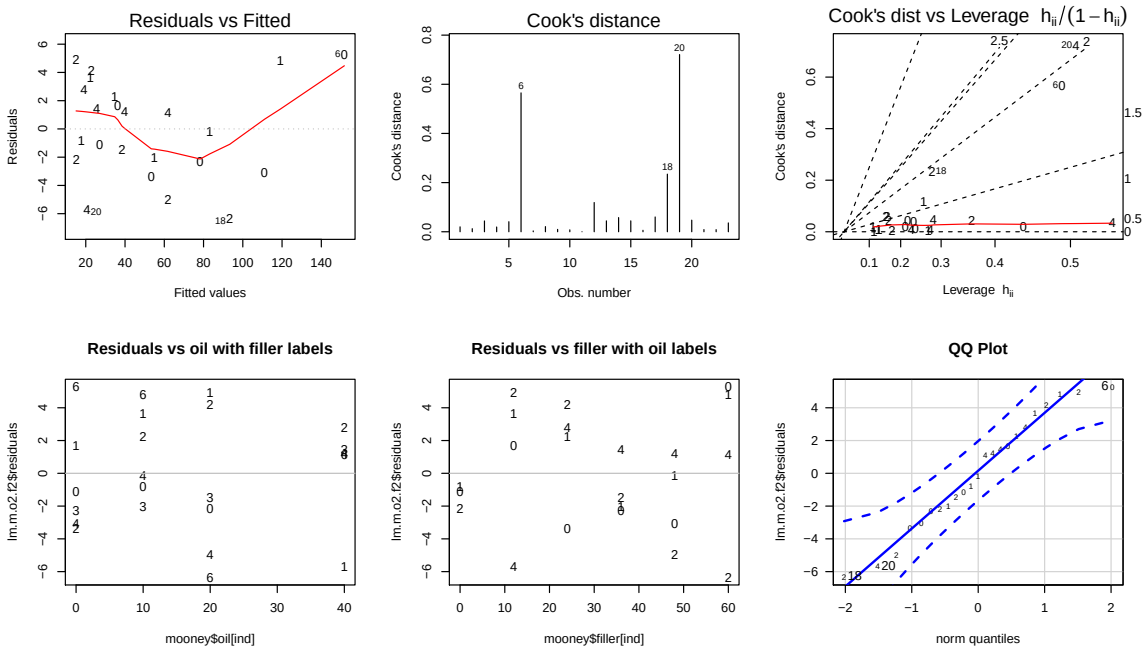
```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3497 -2.2231 -0.1615  2.5424  5.2749
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     27.144582   2.616779  10.373 9.02e-09 ***
## oil             -1.271442   0.213533  -5.954 1.57e-05 ***
## filler           0.436984   0.152658   2.862   0.0108 *
## I(oil^2)         0.033611   0.004663   7.208 1.46e-06 ***
## I(filler^2)      0.027323   0.002410  11.339 2.38e-09 ***
## I(oil * filler) -0.038659   0.003187 -12.131 8.52e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.937 on 17 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.9917,Adjusted R-squared:  0.9892
## F-statistic: 405.2 on 5 and 17 DF,  p-value: < 2.2e-16
```

```
## poly() will evaluate variables and give joint polynomial values
##        which is helpful when you have many predictors
#head(mooney, 10)
#head(poly(mooney$oil, mooney$filler, degree = 2, raw = TRUE), 10)
## This model is equivalent to the one above
#lm.m.o2.f2 <- lm(mooney ~ poly(oil, filler, degree = 2, raw = TRUE), data = mooney)
#summary(lm.m.o2.f2)
```

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.m.o2.f2, which = c(1,4,6), pch=as.character(mooney$oil))

# because of one missing value, get the indices of non-missing
ind <- as.numeric(names(lm.m.o2.f2$residuals))

plot(mooney$oil[ind], lm.m.o2.f2$residuals, main="Residuals vs oil with filler labels", pch=as.character(mooney$filler[ind]))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(mooney$filler[ind], lm.m.o2.f2$residuals, main="Residuals vs filler with oil labels", pch=as.character(mooney$oil[ind]))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.m.o2.f2$residuals, las = 1, id = list(n = 3), main="QQ Plot", pch=as.character(mooney$oil[ind]))
```

```
## 18 20  6
## 18 19  6
```

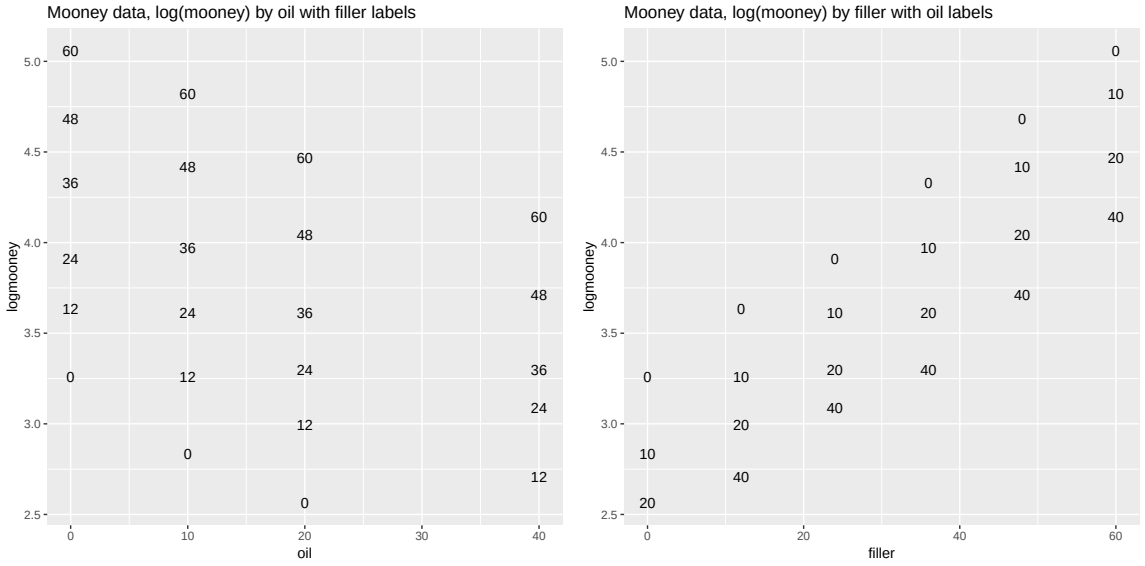
```
## residuals vs order of data
#plot(lm.m.o2.f2$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



8.2.2 Example: Mooney viscosity on log scale

As noted earlier, transformations can often be used instead of polynomials. For example, the original data plots suggest transforming the Moody viscosity to a log scale. If we make this transformation and replot the data, we see that the log Mooney viscosity is roughly linearly related to the filler level at each oil level, but is a quadratic function of oil at each filler level. The plots of the transformed data suggest that a simpler model will be appropriate.

```
# log transform the response
mooney$logmooney <- log(mooney$mooney)

library(ggplot2)
p <- ggplot(mooney, aes(x = oil, y = logmooney, label = filler))
p <- p + geom_text()
#p <- p + scale_y_continuous(limits = c(0, max(mooney$logmooney, na.rm=TRUE)))
p <- p + labs(title="Mooney data, log(mooney) by oil with filler labels")
print(p)
## Warning:  Removed 1 rows containing missing values (geom_text).
library(ggplot2)
p <- ggplot(mooney, aes(x = filler, y = logmooney, label = oil))
p <- p + geom_text()
#p <- p + scale_y_continuous(limits = c(0, max(mooney$logmooney, na.rm=TRUE)))
p <- p + labs(title="Mooney data, log(mooney) by filler with oil labels")
print(p)
## Warning:  Removed 1 rows containing missing values (geom_text).
```



To see that a simpler model is appropriate, we fit the full quadratic model. The interaction term can be omitted here, without much loss of predictive ability (R-squared is similar). The p-value for the interaction term in the quadratic model is 0.34.

```
# I create each term separately
lm.lm.o2.f2 <- lm(logmooney ~ oil + filler + I(oil^2) + I(filler^2) + I(oil * filler),
                  data = mooney)
summary(lm.lm.o2.f2)
```

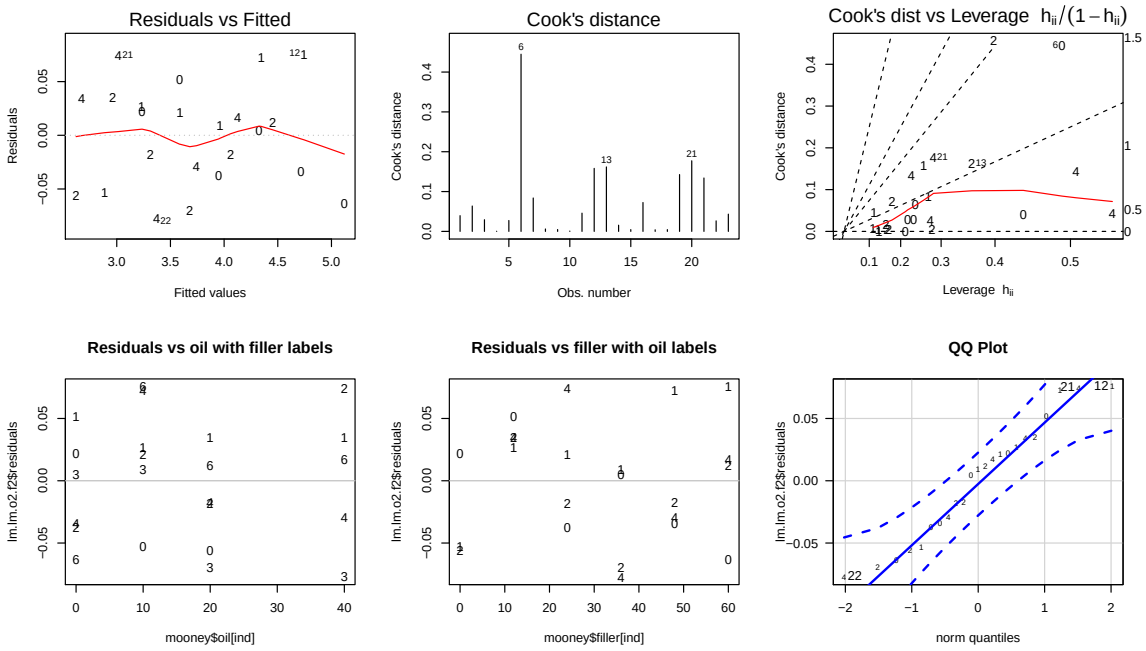
```
##
## Call:
## lm(formula = logmooney ~ oil + filler + I(oil^2) + I(filler^2) +
##     I(oil * filler), data = mooney)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.077261 -0.035795  0.009193  0.030641  0.075640
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.236e+00  3.557e-02  90.970  < 2e-16 ***
## oil             -3.921e-02  2.903e-03 -13.507 1.61e-10 ***
## filler           2.860e-02  2.075e-03  13.781 1.18e-10 ***
## I(oil^2)         4.227e-04  6.339e-05   6.668 3.96e-06 ***
## I(filler^2)      4.657e-05  3.276e-05   1.421    0.173
## I(oil * filler) -4.231e-05  4.332e-05  -0.977    0.342
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05352 on 17 degrees of freedom
```

```
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.9954,Adjusted R-squared:  0.9941
## F-statistic:   737 on 5 and 17 DF,  p-value: < 2.2e-16
```

```
# plot diagnistics
par(mfrow=c(2,3))
plot(lm.lm.o2.f2, which = c(1,4,6), pch=as.character(mooney$oil))

# because of one missing value, get the indices of non-missing
ind <- as.numeric(names(lm.lm.o2.f2$residuals))

plot(mooney$oil[ind], lm.lm.o2.f2$residuals, main="Residuals vs oil with filler labels", pch=as.character(mooney$filler[ind]))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(mooney$filler[ind], lm.lm.o2.f2$residuals, main="Residuals vs filler with oil labels", pch=as.character(mooney$oil[ind]))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.lm.o2.f2$residuals, las = 1, id = list(n = 3), main="QQ Plot", pch=as.character(mooney$oil[ind]))
```

```
## 22 12 21
## 21 12 20
```

```
## residuals vs order of data
#plot(lm.lm.o2.f2$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



After omitting the interaction term, the quadratic effect in filler is not needed in the model (output not given). Once these two effects are removed, each of the remaining effects is significant.

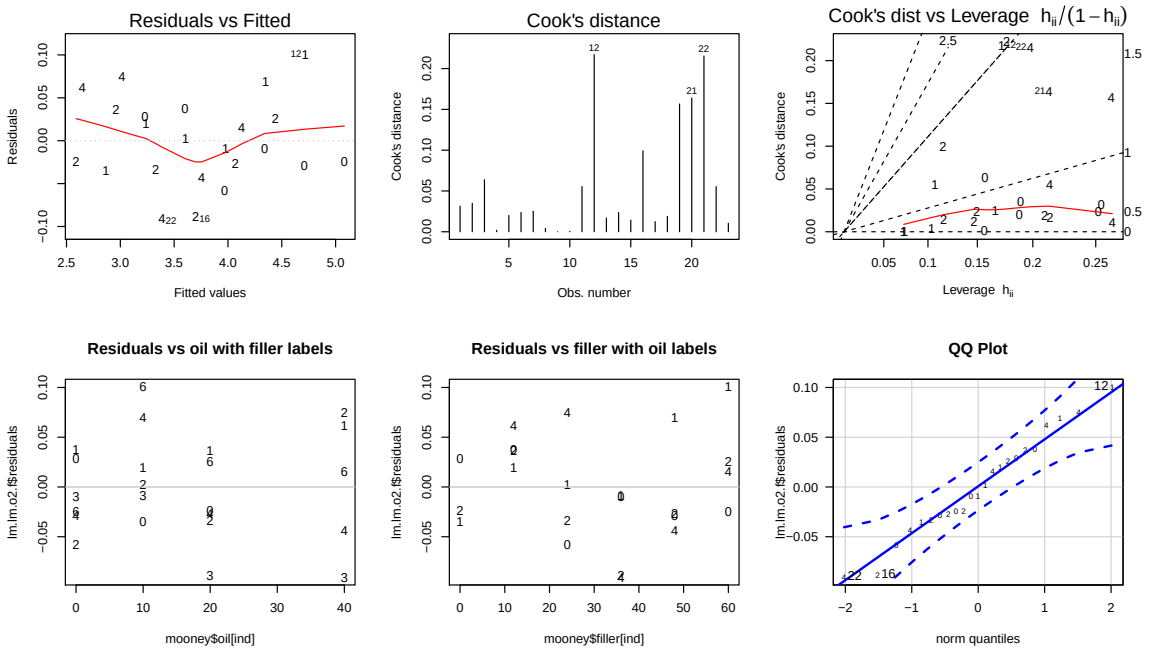
```
# I create each term separately
lm.lm.o2.f <- lm(logmooney ~ oil + filler + I(oil^2),
                 data = mooney)
summary(lm.lm.o2.f)
```

```
##
## Call:
## lm(formula = logmooney ~ oil + filler + I(oil^2), data = mooney)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.090796 -0.031113 -0.008831  0.032533  0.100587
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.230e+00  2.734e-02 118.139  < 2e-16 ***
## oil         -4.024e-02  2.702e-03 -14.890 6.26e-12 ***
## filler       3.086e-02  5.716e-04  53.986  < 2e-16 ***
## I(oil^2)     4.097e-04  6.356e-05   6.446 3.53e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05423 on 19 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.9947,Adjusted R-squared:  0.9939
## F-statistic:  1195 on 3 and 19 DF,  p-value: < 2.2e-16
```

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.lm.o2.f, which = c(1,4,6), pch=as.character(mooney$oil))

# because of one missing value, get the indices of non-missing
ind <- as.numeric(names(lm.lm.o2.f$residuals))

plot(mooney$oil[ind], lm.lm.o2.f$residuals, main="Residuals vs oil with filler labels", pch=as.character(mooney$filler[ind]))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(mooney$filler[ind], lm.lm.o2.f$residuals, main="Residuals vs filler with oil labels", pch=as.character(mooney$oil[ind]))
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.lm.o2.f$residuals, las = 1, id = list(n = 3), main="QQ Plot", pch=as.character(mooney$oil[ind]))
```

```
## 12 22 16
## 12 21 16
```

```
## residuals vs order of data
#plot(lm.lm.o2.f£residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



The model does not appear to have inadequacies. Assuming no difficulties, an important effect of the transformation is that the resulting model is simpler than the model selected on the original scale. This is satisfying to me, but you may not agree. Note that the selected model, with linear effects due to the oil and filler levels and a quadratic effect due to the oil level, agrees with our visual assessment of the data. Assuming no inadequacies, the predicted log Moody viscosity is given by

$$\widehat{\log(\text{Moody viscosity})} = 3.2297 - 0.0402\ \text{Oil} + 0.0004\ \text{Oil}^2 + 0.0309\ \text{Filler}.$$

Quadratic models with two or more predictors are often used in industrial experiments to estimate the optimal combination of predictor values to maximize or minimize the response, over the range of predictor variable values where the model is reasonable. (This strategy is called "response surface methodology".) For example, we might wish to know what combination of oil level between 0 and 40 and filler level between 0 and 60 provides the lowest predicted Mooney viscosity (on the original or log scale). We can visually approximate the minimizer using the data plots, but one can do a more careful job of analysis using standard tools from calculus.

Chapter 9

Discussion of Response Models with Factors and Predictors

We have considered simple models for designed experiments and observational studies where a response variable is modeled as a linear combination of effects due to **factors** or **predictors**, or both. With designed experiments, where only qualitative factors are considered, we get a "pure ANOVA" model. For example, in the experiment comparing survival times of beetles, the potential effects of **insecticide** (with levels A, B, C, and D) and **dose** (with levels 1=low, 2=medium, and 3=high) are included in the model as **factors** because these variables are qualitative. The natural model to consider is a two-way ANOVA with effects for dose and insecticide and a dose-by-insecticide interaction. If, however, the dose given to each beetle was recorded on a measurement scale, then the dosages can be used to define a predictor variable which can be used as a "regression effect" in a model. That is, the dose or some function of dose can be used as a (quantitative) predictor instead of as a qualitative effect.

For simplicity, assume that the doses are 10, 20, and 30, but the actual levels are irrelevant to the discussion. The simple additive model, or ANCOVA model, assumes that there is a linear relationship between mean survival time and dose, with different intercepts for the four insecticides. If data set includes the survival time (times) for each beetle, the insecticide (insect: an alphanumeric variable, with values A, B, C, and D), and dose, you would fit the ANCOVA model this way

```
beetles$insect <- factor(beetles$insect)
lm.t.i.d <- lm(times ~ insect + dose, data = beetles)
```

A more complex model that allows separate regression lines for each insecticide is specified as follows:

```
beetles$insect <- factor(beetles$insect)
lm.t.i.d.id <- lm(times ~ insect + dose + insect:dose, data = beetles)
```

It is important to recognize that the `factor()` statement defines which variables in the model are treated as **factors**. Each effect of `Factor` data type is treated as a factor. Effects in the model statement that are numeric data types are treated as

predictors. To treat a measurement variable as a factor (with one level for each distinct observed value of the variable) instead of a predictor, convert that varible type to a factor using `factor()`. Thus, in the survival time experiment, these models

```
beetles$insect <- factor(beetles$insect)
beetles$dose   <- factor(beetles$dose)
# call this (A) for additive
lm.t.i.d <- lm(times ~ insect + dose, data = beetles)
# call this (I) for interaction
lm.t.i.d.id <- lm(times ~ insect + dose + insect:dose, data = beetles)
```

give the analysis for a two-way ANOVA model without interaction and with interaction, respectively, where both dose and insecticide are treated as factors (since dose and insect are both converted to factors), even though we just defined dose on a measurement scale!

Is there a basic connection between the ANCOVA and separate regression line models for dose and two-way ANOVA models where dose and insecticide are treated as factors? Yes — I mentioned a connection when discussing ANCOVA and I will try now to make the connection more explicit.

For the moment, let us simplify the discussion and assume that only one insecticide was used at three dose levels. The LS estimates of the mean responses from the quadratic model

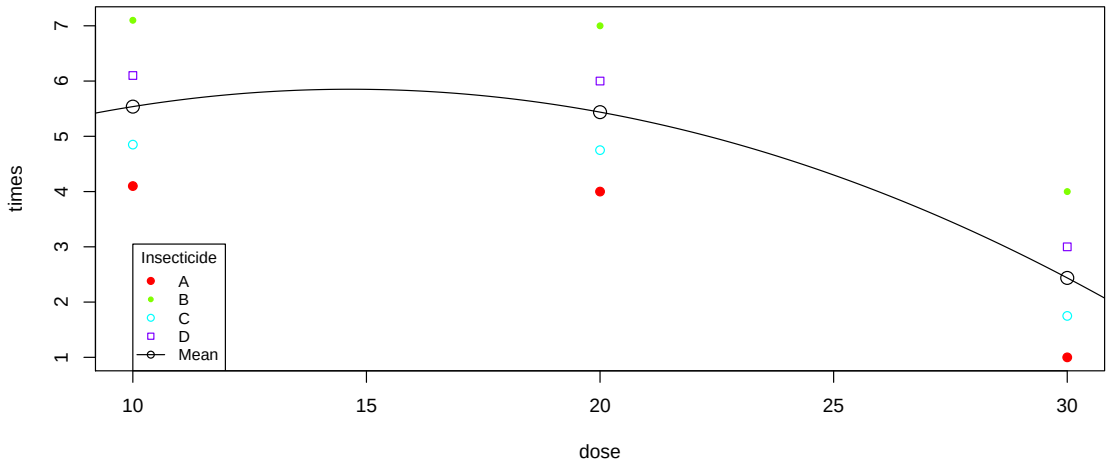
$$\text{Times} = \beta_0 + \beta_1 \text{ Dose} + \beta_2 \text{ Dose}^2 + \varepsilon$$

are the observed average survival times at the three dose levels. The LS curve goes through the mean survival time at each dose, as illustrated in the picture below.

If we treat dose as a **factor**, and fit the one-way ANOVA model

$$\text{Times} = \text{Grand Mean} + \text{Dose Effect} + \text{Residual},$$

then the LS estimates of the population mean survival times are the observed mean survival times. The two models are mathematically equivalent, but the parameters have different interpretations. In essence, the one-way ANOVA model places no restrictions on the values of the population means (no *a priori* relation between them) at the three doses, and neither does the quadratic model! (WHY?)



In a one-way ANOVA, the standard hypothesis of interest is that the dose effects are zero. This can be tested using the one-way ANOVA F-test, or by testing H_0 : $\beta_1 = \beta_2 = 0$ in the quadratic model. With three dosages, the absence of a linear or quadratic effect implies that all the population mean survival times must be equal. An advantage of the polynomial model over the one-way ANOVA is that it provides an easy way to **quantify** how dose impacts the mean survival, and a convenient way to check whether a simple description such as a simple linear regression model is adequate to describe the effect.

More generally, if dose has p levels, then the one-way ANOVA model

$$\text{Times} = \text{Grand Mean} + \text{Dose Effect} + \text{Residual},$$

is equivalent to the $(p-1)^{st}$ degree polynomial

$$\text{Times} = \beta_0 + \beta_1 \text{ Dose} + \beta_2 \text{ Dose}^2 + \cdots + \beta_{p-1} \text{ Dose}^{(p-1)} + \varepsilon$$

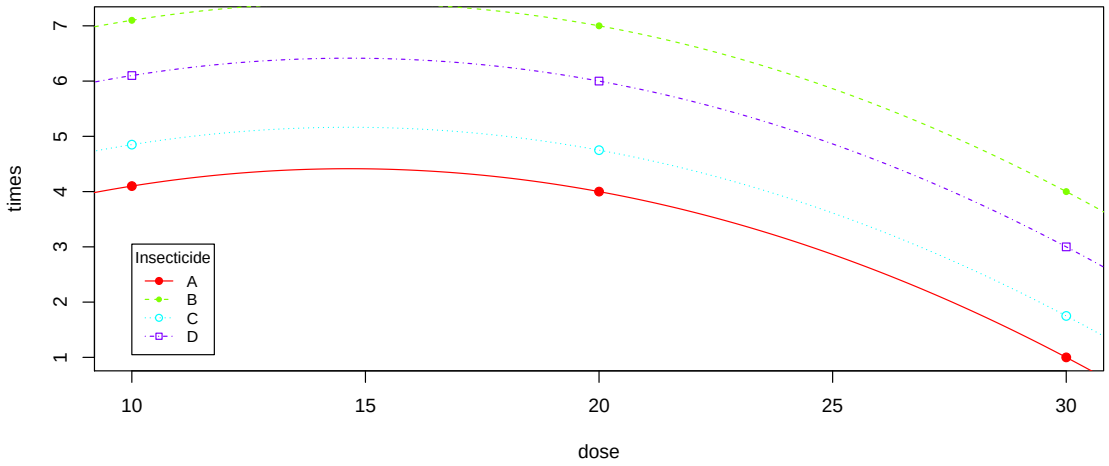
and the one-way ANOVA F-test for no treatment effects is equivalent to testing $H_0 : \beta_1 = \beta_2 = \cdots = \beta_{p-1} = 0$ in this polynomial.

Returning to the original experiment with 4 insecticides and 3 doses, I can show the following two equivalences. First, the two-way additive ANOVA model, with insecticide and dose as factors, i.e., model (A), is mathematically equivalent to an additive model with insecticide as a factor, and a quadratic effect in dose:

```
beetles$insect <- factor(beetles$insect)
lm.t.i.d.d2 <- lm(times ~ insect + dose + I(dose^2), data = beetles)
```

Thinking of dose^2 as a quadratic term in dose, rather than as an interaction, this model has an additive insecticide effect, but the dose effect is not differentiated across insecticides. That is, the model assumes that the quadratic curves for the four insecticides differ only in level (i.e., different intercepts) and that the coefficients

for the dose and dose^2 effects are identical across insecticides. This is an additive model, because the population means plot has parallel profiles. A possible pictorial representation of this model is given below.



Second, the two-way ANOVA interaction model, with insecticide and dose as factors, i.e., model (I), is mathematically equivalent to an interaction model with insecticide as a factor, and a quadratic effect in dose.

```
beetles$insect <- factor(beetles$insect)
lm.t.i.d.d2.id.id2 <- lm(times ~ insect + dose + I(dose^2)
                                + insect:dose + insect:I(dose^2), data = beetles)
```

This model fits separate quadratic relationships for each of the four insecticides, by including interactions between insecticides and the linear and quadratic terms in dose. Because dose has three levels, this model places no restrictions on the mean responses.

To summarize, we have established that

- The additive two-way ANOVA model with insecticide and dose as factors is mathematically identical to an additive model with an insecticide factor and a quadratic effect in dose. The ANCOVA model with a linear effect in dose is a special case of these models, where the quadratic effect is omitted.
- The two-way ANOVA interaction model with insecticide and dose as factors is mathematically identical to a model with an insecticide factor, a quadratic effect in dose, and interactions between the insecticide and the linear and quadratic dose effects. The separate regression lines model with a linear effect in dose is a special case of these models, where the quadratic dose effect and the interaction of the quadratic term with insecticide are omitted.

Recall that response models with **factors** and **predictors** as effects can be fit using the `lm()` procedure, but each factor or interaction involving a factor must be

represented in the model using indicator variables or product terms. The number of required indicators or product effects is one less than the number of distinct levels of the factor. For example, to fit the model with "parallel" quadratic curves in dose, you can define (in the `data.frame()`) three indicator variables for the insecticide effect, say I_1, I_2, and I_3, and fit the model

$$\text{Times} = \beta_0 + \beta_1 I_1 + \beta_2 I_2 + \beta_3 I_3 + \beta_4 \text{ Dose} + \beta_5 \text{ Dose}^2 + \varepsilon.$$

For the "quadratic interaction model", you must define 6 interaction or product terms between the 3 indicators and the 2 dose terms:

$$\begin{aligned} \text{Times} \;=\; & \beta_0 + \beta_1 I_1 + \beta_2 I_2 + \beta_3 I_3 + \beta_4 \text{ Dose} + \beta_5 \text{ Dose}^2 \\ & + \beta_6 I_1 \text{ Dose} + \beta_7 I_2 \text{ Dose} + \beta_8 I_3 \text{ Dose} \\ & + \beta_9 I_1 \text{ Dose}^2 + \beta_{10} I_2 \text{ Dose}^2 + \beta_{11} I_3 \text{ Dose}^2 + \varepsilon. \end{aligned}$$

The ($\beta_6 I_1$ Dose+$\beta_7 I_2$ Dose+$\beta_8 I_3$ Dose) component in the model formally corresponds to the $insect * dose$ interaction, whereas the ($\beta_9 I_1$ Dose$^2 + \beta_{10} I_2$ Dose$^2 + \beta_{11} I_3$ Dose^2) component is equivalent to the $insect * dose * dose$ interaction (i.e., testing $H_0 : \beta_9 = \beta_{10} = \beta_{11} = 0$).

This discussion is not intended to confuse, but rather to impress upon you the intimate connection between regression and ANOVA, and to convince you of the care that is needed when modelling variation even in simple studies. Researchers are usually faced with more complex modelling problems than we have examined, where many variables might influence the response. If experimentation is possible, a scientist will often control the levels of variables that influence the response but that are not of primary interest. This can result in a manageable experiment with, say, four or fewer qualitative or quantitative variables that are systematically varied in a scientifically meaningful way. In observational studies, where experimentation is not possible, the scientist builds models to assess the effects of interest on the response, adjusting the response for all the uncontrolled variables that might be important. The uncontrolled variables are usually a mixture of factors and predictors. Ideally, the scientist knows what variables to control in an experiment and which to vary, and what variables are important to collect in an observational study.

The level of complexity that I am describing here might be intimidating, but certain basic principles can be applied to many of the studies you will see. Graduate students in statistics often take several courses (5+) in experimental design, regression analysis, and linear model theory to master the breadth of models, and the subtleties of modelling, that are needed to be a good data analyst. I can only scratch the surface here. I will discuss a reasonably complex study having multiple factors and multiple predictors. The example focuses on strategies for building models, with little attempt to do careful diagnostic analyses. Hopefully, the example will give you an

appreciation for statistical modelling, but **please be careful — these tools are dangerous!**

9.1 Some Comments on Building Models

A primary goal in many statistical analyses is to build a model or models to understand the variation in a response. Fortunately, or unfortunately, there is no consensus on how this should be done. Ideally, theory would suggest models to compare, but in many studies the goal is to provide an initial model that will be refined, validated, or refuted, by further experimentation. An extreme view is that the selected model(s) should only include effects that are "**statistically important**", whereas another extreme suggests that all effects that might be "**scientifically important**" should be included.

A difficulty with implementing either approach is that importance is relative to specific goals (i.e., Why are you building the model and what do you plan to use the model for? Is the model a prescription or device to make predictions? Is the model a tool to understand the effect that one or more variables have on a response, after adjusting for uninteresting, but important effects that can not be controlled? etc.) Madigan and Raftery, in the 1994 edition of *The Journal of the American Statistical Association*, comment that "Science is an iterative process in which competing models of reality are compared on the basis of how well they predict what is observed; models that predict much less well than their competitors are discarded." They argue that models should be selected using Occum's razor, a widely accepted norm in scientific investigations whereby the simplest plausible model among all reasonable models, given the data, is preferred. Madigan and Raftery's ideas are fairly consistent with the first extreme, but can be implemented in a variety of ways, depending on how you measure prediction adequacy. They propose a Bayesian approach, based on model averaging and prior beliefs on the plausibility of different models. An alternative method using Mallow's C_p criterion will be discussed later.

A simple compromise between the two extremes might be to start the model building process with the most complex model that is scientifically reasonable, but still interpretable, and systematically eliminate effects using backward elimination. The initial or **maximal** model might include polynomial effects for predictors, main effects and interactions (2 factor, 3 factor, etc.) between factors, and products or interactions between predictors and factors. This approach might appear to be less than ideal because the importance of effects is assessed using hypothesis tests and no attempt is made to assess the effect of changes on predictions. However, one can show that the average squared error in predictions is essentially reduced by eliminating **insignificant** regression effects from the model, so this approach seems tenable.

It might be sensible to only assess significance of effects specified in the model

statement. However, many of these effects consist of several degrees-of-freedom. That is, the effect corresponds to several regression coefficients in the model. (Refer to the discussion following the displayed equations on page 621). The individual regression variables that comprise an effect could also be tested individually. However, if the effect is a factor (with 3+ levels) or an interaction involving a factor, then the interpretation of tests on individual regression coefficients depends on the level of the factor that was selected to be the baseline category. The Type III F-test on the entire effect does not depend on the baseline category. In essence, two researchers can start with different representations of the same mathematical model (i.e., the parameters are defined differently for different choices of baseline categories), use the same algorithm for selecting a model, yet come to different final models for the data.

Statisticians often follow the **hierarchy** principle, which states that a lower order term (be it a factor or a predictor) may be considered for exclusion from a model only if no higher order effects that include the term are present in the model. For example, given an initial model with effects A, B, C, and the $A * B$ interaction, the only candidates for omission at the first step are C and $A * B$. If you follow the hierarchy principle, and test an entire effect rather than test the single degree-of-freedom components that comprise an effect, then the difficulty described above can not occur. The hierarchy principle is most appealing with **pure** ANOVA models (such as the three-factor model in the example below), where all the regression variables are indicators. In ANOVA models, the ANOVA effects are of interest because they imply certain structure on the means. The individual regression variables that define the effects are not usually a primary interest.

A non-hierarchical backward elimination algorithm where single degree-of-freedom effects are eliminated independently of the other effects in the model is implemented in the `step()` procedure. Recall our discussion of backwards elimination from Chapter 3 earlier this semester.

9.2 Example: The Effect of Sex and Rank on Faculty Salary

The data in this example were collected from the personnel files of faculty at a small college in the 1970s. The data were collected to assess whether women were being discriminated against (consciously or unconsciously) in salary. The sample consists of tenured and tenure-stream faculty only. Temporary faculty were excluded from consideration (because they were already being discriminated against).

The variables below are id (individual identification numbers from 1 to 52), sex (coded 1 for female and 0 for male), rank (coded 1 for Asst. Professor, 2 for Assoc. Professor and 3 for Full Professor), year (number of years in current rank), degree

(coded 1 for Doctorate, 0 else), yd (number of years since highest degree was earned), and salary (academic year salary in dollars).

```
#### Example: Faculty salary
faculty <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch09_faculty.dat"
                      , header = TRUE)
head(faculty)
```

```
##   id sex rank year degree yd salary
## 1  1   0    3   25      1 35  36350
## 2  2   0    3   13      1 22  35350
## 3  3   0    3   10      1 23  28200
## 4  4   1    3    7      1 27  26775
## 5  5   0    3   19      0 30  33696
## 6  6   0    3   16      1 21  28516
```

```
str(faculty)
```

```
## 'data.frame': 52 obs. of  7 variables:
##  $ id    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ sex   : int  0 0 0 1 0 0 1 0 0 0 ...
##  $ rank  : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ year  : int  25 13 10 7 19 16 0 16 13 13 ...
##  $ degree: int  1 1 1 1 0 1 0 1 0 0 ...
##  $ yd    : int  35 22 23 27 30 21 32 18 30 31 ...
##  $ salary: int  36350 35350 28200 26775 33696 28516 24900 31909 31850 32850 ...
```

```
faculty$sex    <- factor(faculty$sex   , labels=c("Male", "Female"))
# ordering the rank variable so Full is the baseline, then descending.
faculty$rank   <- factor(faculty$rank  , levels=c(3,2,1)
                         , labels=c("Full", "Assoc", "Asst"))
faculty$degree <- factor(faculty$degree, labels=c("Other", "Doctorate"))
head(faculty)
```

```
##   id    sex rank year    degree yd salary
## 1  1   Male Full   25 Doctorate 35  36350
## 2  2   Male Full   13 Doctorate 22  35350
## 3  3   Male Full   10 Doctorate 23  28200
## 4  4 Female Full    7 Doctorate 27  26775
## 5  5   Male Full   19     Other 30  33696
## 6  6   Male Full   16 Doctorate 21  28516
```

```
str(faculty)
```

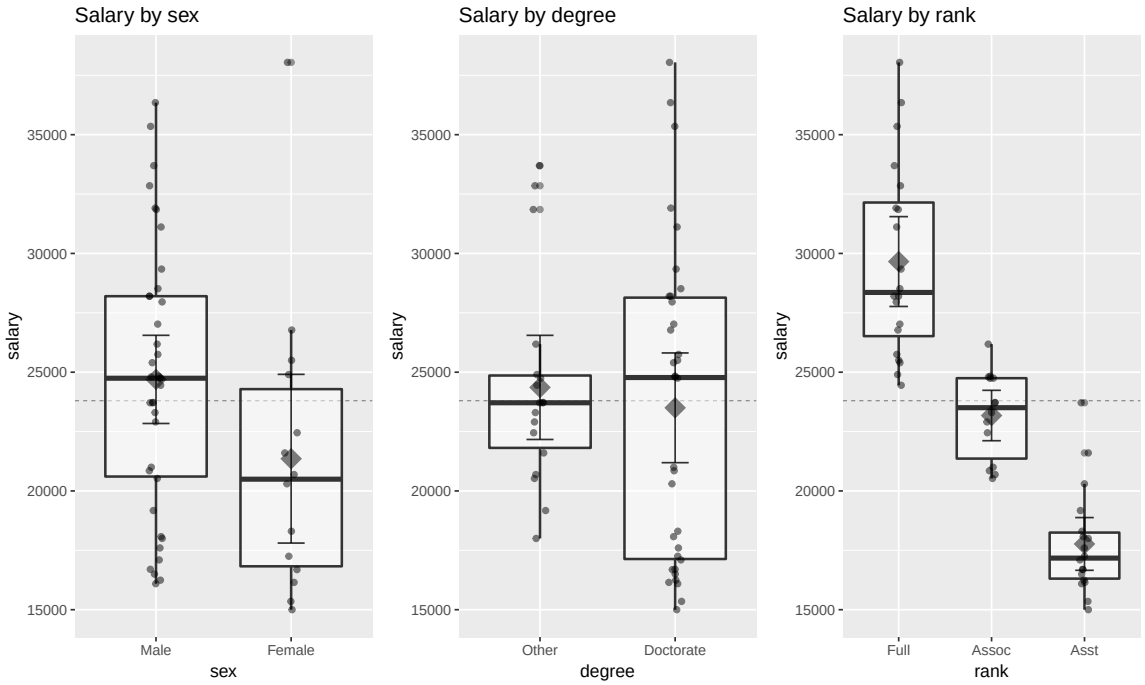
```
## 'data.frame': 52 obs. of  7 variables:
##  $ id    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ sex   : Factor w/ 2 levels "Male","Female": 1 1 1 2 1 1 2 1 1 1 ...
##  $ rank  : Factor w/ 3 levels "Full","Assoc",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ year  : int  25 13 10 7 19 16 0 16 13 13 ...
##  $ degree: Factor w/ 2 levels "Other","Doctorate": 2 2 2 2 1 2 1 2 1 1 ...
##  $ yd    : int  35 22 23 27 30 21 32 18 30 31 ...
##  $ salary: int  36350 35350 28200 26775 33696 28516 24900 31909 31850 32850 ...
```

The data includes two potential predictors of salary (year and yd), and three

factors (sex, rank, and degree). A primary statistical interest is whether males and females are compensated equally, on average, after adjusting salary for rank, years in rank, and the other given effects. Furthermore, we wish to know whether an effect due to sex is the same for each rank, or not.

Before answering these questions, let us look at the data. I will initially focus on the effect of the individual factors (sex, rank, and degree) on salary. A series of box-plots is given below. Looking at the boxplots, notice that women tend to earn less than men, that faculty with Doctorates tend to earn more than those without Doctorates (median), and that salary tends to increase with rank.

```
# plot marginal boxplots

# Plot the data using ggplot
library(ggplot2)
p1 <- ggplot(faculty, aes(x = sex, y = salary, group = sex))
# plot a reference line for the global mean (assuming no groups)
p1 <- p1 + geom_hline(aes(yintercept = mean(salary)),
                      colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p1 <- p1 + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p1 <- p1 + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p1 <- p1 + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                        alpha = 0.5)
# confidence limits based on normal distribution
p1 <- p1 + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                        width = .2, alpha = 0.8)
p1 <- p1 + labs(title = "Salary by sex")

# Plot the data using ggplot
library(ggplot2)
p2 <- ggplot(faculty, aes(x = degree, y = salary, group = degree))
# plot a reference line for the global mean (assuming no groups)
p2 <- p2 + geom_hline(aes(yintercept = mean(salary)),
                      colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p2 <- p2 + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p2 <- p2 + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p2 <- p2 + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                        alpha = 0.5)
# confidence limits based on normal distribution
p2 <- p2 + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                        width = .2, alpha = 0.8)
p2 <- p2 + labs(title = "Salary by degree")

# Plot the data using ggplot
library(ggplot2)
p3 <- ggplot(faculty, aes(x = rank, y = salary, group = rank))
# plot a reference line for the global mean (assuming no groups)
p3 <- p3 + geom_hline(aes(yintercept = mean(salary)),
                      colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.75 to stand out behind CI
p3 <- p3 + geom_boxplot(size = 0.75, alpha = 0.5)
# points for observed data
p3 <- p3 + geom_point(position = position_jitter(w = 0.05, h = 0), alpha = 0.5)
# diamond at mean for each group
p3 <- p3 + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                        alpha = 0.5)
# confidence limits based on normal distribution
p3 <- p3 + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                        width = .2, alpha = 0.8)
p3 <- p3 + labs(title = "Salary by rank")

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



9.2.1 A Three-Way ANOVA on Salary Data

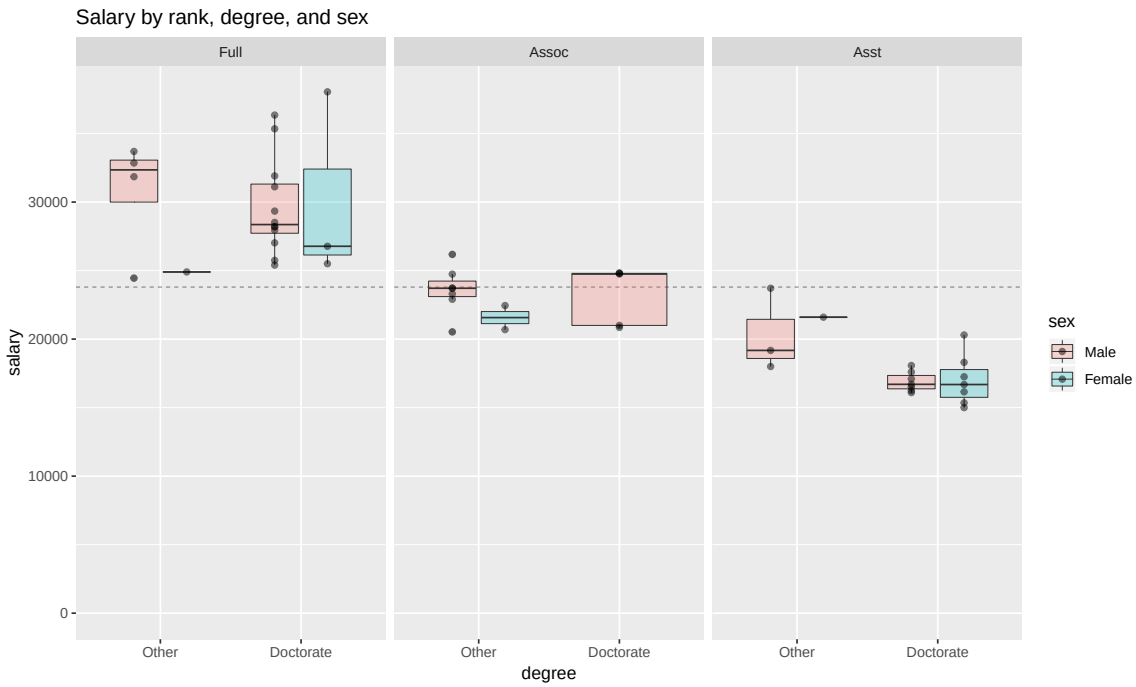
Hopefully, our earlier analyses have cured you of the desire to claim that a sex effect exists before considering whether the differences between male and female salaries might be due to other factors. The output below gives the sample sizes, means, and standard deviations for the 11 combinations of sex, rank, and degree observed in the data. Side-by-side boxplots of the salaries for the 11 combinations are also provided. One combination of the three factors was not observed: female Associate Professors without Doctorates.

Looking at the summaries, the differences between sexes **within** each combination of rank and degree appear to be fairly small. There is a big difference in the ranks of men and women, with a higher percentage of men in the more advanced ranks. This might explain the differences between male and female salaries, when other factors are ignored.

```
library(plyr)
fac.sum <- ddply(faculty, .(sex, rank, degree), function(.df) {
                    data.frame(n = length(.df$salary)
                               , m = mean(.df$salary)
                               , s = sd(.df$salary)
                                )
                    })
fac.sum
```

```
##       sex  rank    degree  n        m         s
## 1    Male  Full     Other  4 30711.50 4241.9723
## 2    Male  Full Doctorate 12 29592.75 3479.9566
## 3    Male Assoc     Other  7 23584.57 1733.2365
## 4    Male Assoc Doctorate  5 23246.20 2120.2684
## 5    Male  Asst     Other  3 20296.00 3016.9642
## 6    Male  Asst Doctorate  7 16901.14  728.9962
## 7  Female  Full     Other  1 24900.00        NA
## 8  Female  Full Doctorate  3 30106.67 6904.2927
## 9  Female Assoc     Other  2 21570.00 1244.5079
## 10 Female  Asst     Other  1 21600.00        NA
## 11 Female  Asst Doctorate  7 17005.71 1834.6791
```

```
# plot marginal boxplots

library(ggplot2)
# create position dodge offset for plotting points
pd <- position_dodge(0.75) # 0.75 puts dots up center of boxplots

p <- ggplot(faculty, aes(x = degree, y = salary, fill = sex))
# plot a reference line for the global mean (assuming no groups)
p <- p + geom_hline(aes(yintercept = mean(salary)),
                    colour = "black", linetype = "dashed", size = 0.3, alpha = 0.5)
# boxplot, size=.25 for thin lines
p <- p + geom_boxplot(size = 0.25, alpha = 0.25)
# points for observed data
p <- p + geom_point(position = pd, alpha = 0.5)
p <- p + facet_grid(. ~ rank)
p <- p + scale_y_continuous(limits = c(0, max(faculty$salary)))
p <- p + labs(title = "Salary by rank, degree, and sex")
print(p)
```



I will consider two simple analyses of these data. The first analysis considers the effect of the three factors on salary. The second analysis considers the effect of the predictors. A complete analysis using both factors and predictors is then considered.

I am doing the three factor analysis because the most complex pure ANOVA problem we considered this semester has two factors — the analysis is for illustration **only**!!

The full model for a three-factor study includes the three main effects, the three possible two-factor interactions, plus the three-factor interaction. Identifying the factors by S (sex), D (degree) and R (rank), we write the full model as

$$\begin{aligned} \text{Salary} \;=\; & \text{Grand mean} + \text{S effect} + \text{D effect} + \text{R effect} \\ & + \text{S*D interaction} + \text{S*R interaction} + \text{R*D interaction} \\ & + \text{S*D*R interaction} + \text{Residual}. \end{aligned}$$

You should understand what main effects and two-factor interactions measure, but what about the three-factor term? If you look at the two levels of degree separately, then a three-factor interaction is needed if the interaction between sex and rank is different for the two degrees. (i.e., the profile plots are different for the two degrees). Not surprisingly, three-factor interactions are hard to interpret.

I considered a hierarchical backward elimination of effects (see Chapter 3 for details). Individual regression variables are not considered for deletion, unless they correspond to an effect in the model statement. All tests were performed at the 0.10 level, but this hardly matters here.

The first step in the elimination is to fit the full model and check whether the three-factor term is significant. The three-factor term was not significant (in fact, it couldn't be fit because one category had zero observations). After eliminating this effect, I fit the model with all three two-factor terms, and then sequentially deleted the least important effects, one at a time, while still adhering to the hierarchy principle using the AIC criterion from the `step()` function. The final model includes only an effect due to rank. Finally, I compute the `lsmeans()` to compare salary for all pairs of rank.

```
# fit full model
lm.faculty.factor.full <- lm(salary ~ sex*rank*degree, data = faculty)

## Note that there are not enough degrees-of-freedom to estimate all these effects
##   because we have 0 observations for Female/Assoc/Doctorate
library(car)
Anova(lm.faculty.factor.full, type=3)
## Error in Anova.III.lm(mod, error, singular.ok = singular.ok, ...):  there are aliased
coefficients in the model
summary(lm.faculty.factor.full)
##
## Call:
## lm(formula = salary ~ sex * rank * degree, data = faculty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -6261.5 -1453.0  -225.9  1349.7  7938.3
##
## Coefficients: (1 not defined because of singularities)
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          30711.5     1485.0  20.681  < 2e-16
## sexFemale                            -5811.5     3320.7  -1.750 0.087581
## rankAssoc                            -7126.9     1861.6  -3.828 0.000433
## rankAsst                            -10415.5     2268.4  -4.591 4.13e-05
## degreeDoctorate                      -1118.8     1714.8  -0.652 0.517774
## sexFemale:rankAssoc                   3796.9     4086.3   0.929 0.358229
## sexFemale:rankAsst                    7115.5     4773.7   1.491 0.143734
## sexFemale:degreeDoctorate             6325.4     3834.4   1.650 0.106653
## rankAssoc:degreeDoctorate              780.4     2442.3   0.320 0.750952
## rankAsst:degreeDoctorate             -2276.1     2672.3  -0.852 0.399304
## sexFemale:rankAssoc:degreeDoctorate       NA         NA      NA       NA
## sexFemale:rankAsst:degreeDoctorate   -7524.8     5383.7  -1.398 0.169720
##
## (Intercept)                         ***
## sexFemale                           .
## rankAssoc                           ***
## rankAsst                            ***
## degreeDoctorate
## sexFemale:rankAssoc
## sexFemale:rankAsst
## sexFemale:degreeDoctorate
## rankAssoc:degreeDoctorate
## rankAsst:degreeDoctorate
## sexFemale:rankAssoc:degreeDoctorate
## sexFemale:rankAsst:degreeDoctorate
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2970 on 41 degrees of freedom
## Multiple R-squared:  0.7975,Adjusted R-squared:  0.7481
## F-statistic: 16.14 on 10 and 41 DF,  p-value: 2.989e-11
## AIC
# option: test="F" includes additional information
#           for parameter estimate tests that we're familiar with
# option: for BIC, include k=log(nrow( [data.frame name] ))
lm.faculty.factor.red.AIC <- step(lm.faculty.factor.full, direction="backward", test="F")
## Start:  AIC=841.26
## salary ~ sex * rank * degree
##
##                   Df Sum of Sq       RSS    AIC F value Pr(>F)
## <none>                          361677227 841.26
## - sex:rank:degree  1  17233177 378910404 841.68  1.9536 0.1697
## Because the full model can not be fit, the step() procedure does not work
## Below we remove the three-way interaction, then the step() procedure will
```

```
##   do the rest of the work for us.
```

Remove the three-way interaction, then use `step()` to perform backward selection based on AIC.

```
# model reduction using update() and subtracting (removing) model terms
lm.faculty.factor.red <- lm.faculty.factor.full;

# remove variable
lm.faculty.factor.red <- update(lm.faculty.factor.red, ~ . - sex:rank:degree );
Anova(lm.faculty.factor.red, type=3)
## Anova Table (Type III tests)
##
## Response: salary
##                 Sum Sq Df  F value    Pr(>F)
## (Intercept) 3932650421  1 435.9113 < 2.2e-16 ***
## sex           11227674  1   1.2445 0.2709438
## rank         196652264  2  10.8989 0.0001539 ***
## degree         421614  1   0.0467 0.8298945
## sex:rank       2701493  2   0.1497 0.8614045
## sex:degree     7661926  1   0.8493 0.3620198
## rank:degree   33433415  2   1.8529 0.1693627
## Residuals    378910404 42
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# AIC backward selection
lm.faculty.factor.red.AIC <- step(lm.faculty.factor.red, direction="backward", test="F")
## Start:  AIC=841.68
## salary ~ sex + rank + degree + sex:rank + sex:degree + rank:degree
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - sex:rank     2   2701493 381611896 838.05  0.1497 0.8614
## - sex:degree   1   7661926 386572329 840.72  0.8493 0.3620
## <none>                     378910404 841.68
## - rank:degree  2  33433415 412343819 842.08  1.8529 0.1694
##
## Step:  AIC=838.05
## salary ~ sex + rank + degree + sex:degree + rank:degree
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - sex:degree   1  12335789 393947686 837.71  1.4223 0.2394
## <none>                     381611896 838.05
## - rank:degree  2  32435968 414047864 838.29  1.8699 0.1662
##
## Step:  AIC=837.71
## salary ~ sex + rank + degree + rank:degree
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
```

```
## - sex          1   3009036 396956722 836.10  0.3437 0.5606
## - rank:degree  2  27067985 421015671 837.16  1.5460 0.2242
## <none>                     393947686 837.71
##
## Step:  AIC=836.1
## salary ~ rank + degree + rank:degree
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - rank:degree  2  31019255 427975976 836.01  1.7973 0.1772
## <none>                     396956722 836.10
##
## Step:  AIC=836.01
## salary ~ rank + degree
##
##          Df  Sum of Sq        RSS    AIC F value   Pr(>F)
## - degree  1   10970082  438946058 835.33  1.2304   0.2729
## <none>                  427975976 836.01
## - rank    2 1349072233 1777048209 906.04 75.6532 1.45e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=835.33
## salary ~ rank
##
##        Df  Sum of Sq        RSS    AIC F value    Pr(>F)
## <none>                 438946058 835.33
## - rank  2 1346783800 1785729858 904.30  75.171 1.174e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# all are significant, stop.
# final model: salary ~ rank
lm.faculty.factor.final <- lm.faculty.factor.red.AIC

library(car)
Anova(lm.faculty.factor.final, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: salary
##                 Sum Sq Df  F value    Pr(>F)
## (Intercept) 1.7593e+10  1 1963.932 < 2.2e-16 ***
## rank        1.3468e+09  2   75.171 1.174e-15 ***
## Residuals   4.3895e+08 49
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm.faculty.factor.final)
```

```
##
## Call:
```

```
## lm(formula = salary ~ rank, data = faculty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5209.0 -1819.2  -417.8  1586.6  8386.1
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  29659.0      669.3  44.316  < 2e-16 ***
## rankAssoc    -6483.0     1043.0  -6.216 1.09e-07 ***
## rankAsst    -11890.3      972.4 -12.228  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2993 on 49 degrees of freedom
## Multiple R-squared:  0.7542,Adjusted R-squared:  0.7442
## F-statistic: 75.17 on 2 and 49 DF,  p-value: 1.174e-15
```

All ranks are different with salaries increasing with rank.

```
### comparing lsmeans (may be unbalanced)
library(lsmeans)
## compare levels of main effects
lsmeans(lm.faculty.factor.final, list(pairwise ~ rank), adjust = "bonferroni")
## $`lsmeans of rank`
##  rank    lsmean       SE df lower.CL upper.CL
##  Full  29658.95 669.2564 49 28314.03 31003.87
##  Assoc 23175.93 799.9144 49 21568.44 24783.42
##  Asst  17768.67 705.4582 49 16351.00 19186.34
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast      estimate       SE df t.ratio p.value
##  Full - Assoc  6483.021 1042.961 49   6.216  <.0001
##  Full - Asst  11890.283  972.407 49  12.228  <.0001
##  Assoc - Asst  5407.262 1066.553 49   5.070  <.0001
##
## P value adjustment: bonferroni method for 3 tests
```

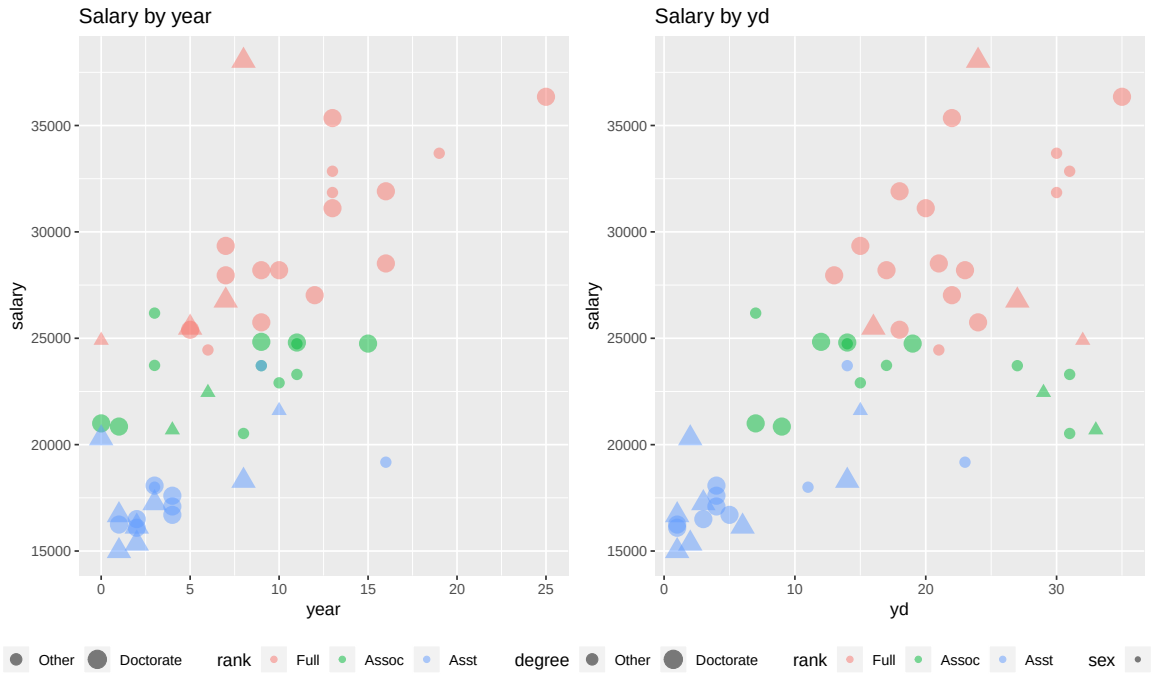
This analysis suggests that sex is not predictive of salary, once other factors are taken into account. In particular, faculty rank appears to be the sole important effect, in the sense that once salaries are adjusted for rank no other factors explain a significant amount of the unexplained variation in salaries.

As noted earlier, *the analysis was meant to illustrate a three-factor ANOVA and backward selection.* The analysis is likely flawed, because it ignores the effects of year and year since degree on salary.

9.2.2 Using Year and Year Since Degree to Predict Salary

Plots of the salary against years in rank and years since degree show fairly strong associations with salary. The variability in salaries appears to be increasing with year and with year since degree, which might be expected. You might think to transform the salaries to a log scale to eliminate this effect, but doing so has little impact on the conclusions (not shown).

```
library(ggplot2)

p1 <- ggplot(faculty, aes(x = year, y = salary, colour = rank, shape = sex, size = degree))
p1 <- p1 + scale_size_discrete(range=c(3,5))
## Warning:  Using size for a discrete variable is not advised.
p1 <- p1 + geom_point(alpha = 0.5)
p1 <- p1 + labs(title = "Salary by year")
p1 <- p1 + theme(legend.position = "bottom")
#print(p1)

p2 <- ggplot(faculty, aes(x = yd, y = salary, colour = rank, shape = sex, size = degree))
p2 <- p2 + scale_size_discrete(range=c(3,5))
## Warning:  Using size for a discrete variable is not advised.
p2 <- p2 + geom_point(alpha = 0.5)
p2 <- p2 + labs(title = "Salary by yd")
p2 <- p2 + theme(legend.position = "bottom")
#print(p2)

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



As a point of comparison with the three-factor ANOVA, I fit a multiple regression model with year and years since degree as predictors of salary. These two predictors are important for explaining the variation in salaries, but together they explain much less of the variation (58%) than rank does on its own (75%).

```r
# interaction model
lm.s.y.yd.yyd <- lm(salary ~ year*yd, data = faculty)
summary(lm.s.y.yd.yyd)
```

```
##
## Call:
## lm(formula = salary ~ year * yd, data = faculty)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10368.5  -2361.5   -505.7   2363.1  12211.6
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16287.391   1395.049  11.675 1.25e-15 ***
## year          561.155    275.243   2.039  0.04700 *
## yd            235.415     83.266   2.827  0.00683 **
## year:yd        -3.089     10.412  -0.297  0.76796
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3958 on 48 degrees of freedom
## Multiple R-squared:  0.579,Adjusted R-squared:  0.5527
## F-statistic:    22 on 3 and 48 DF,  p-value: 4.17e-09
```

```r
# interaction is not significant
lm.s.y.yd <- lm(salary ~ year + yd, data = faculty)
summary(lm.s.y.yd)
```

```
##
## Call:
## lm(formula = salary ~ year + yd, data = faculty)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10321.2  -2347.2   -332.7   2298.8  12240.9
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  16555.7     1052.4  15.732  < 2e-16 ***
## year           489.3      129.6   3.777 0.000431 ***
## yd             222.2       69.8   3.184 0.002525 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3921 on 49 degrees of freedom
## Multiple R-squared:  0.5782,Adjusted R-squared:  0.561
## F-statistic: 33.58 on 2 and 49 DF,  p-value: 6.532e-10
```

9.2.3 Using Factors and Predictors to Model Salaries

The plots we looked at helped us to understand the data. In particular, the plot of salary against years in rank, using rank as a plotting symbol, suggests that a combination of predictors and factors will likely be better for modelling faculty salaries than either of the two models that we proposed up to this point.

There is no evidence of non-linearity in the plots of salary against the predictors, so I will not consider transforming years since degree, years in rank, or salary. Note that the increasing variability in salaries for increasing years in rank and increasing years since degree is partly due to differences in the relationships across ranks. The non-constant variance should be less of a concern in any model that includes rank and either years in rank or years since degree as effects.

I started the model building process with a **maximal** or full model with the five main effects plus the 10 possible interactions between two effects, regardless of whether the effects were factors or predictors. Notationally, this model is written as follows:

$$\begin{aligned}\text{Salary} = {} & \text{Grand mean} + \text{S effect} + \text{D effect} + \text{R effect} + \text{YEAR effect} + \text{YD effect} \\ & + \text{S*D interaction} + \text{S*R interaction} + \text{S*YEAR interaction} + \text{S*YD interaction} \\ & + \text{D*R interaction} + \text{D*YEAR interaction} + \text{D*YD interaction} \\ & + \text{R*YEAR interaction} + \text{R*YD interaction} + \text{YEAR*YD interaction} + \text{Residual},\end{aligned}$$

where the year and year since degree effects (YD) are linear terms (as in the multiple regression model we considered). To check whether any important effects might have been omitted, I added individual three-factor terms to this model. All of the three factor terms were insignificant (not shown), so I believe that my choice for the "maximal" model is sensible.

The output below gives the fit to the maximal model, and subsequent fits, using the hierarchy principle. Only selected summaries are provided.

```
# fit full model with two-way interactions
lm.faculty.full <- lm(salary ~ (sex + rank + degree + year + yd)^2, data = faculty)
library(car)
Anova(lm.faculty.full, type=3)
```

```
## Anova Table (Type III tests)
##
## Response: salary
##                Sum Sq Df F value  Pr(>F)
## (Intercept)  22605087  1  3.6916 0.06392 .
## sex           4092995  1  0.6684 0.41984
## rank          5731837  2  0.4680 0.63059
## degree        4137628  1  0.6757 0.41735
## year          2022246  1  0.3302 0.56966
## yd            3190911  1  0.5211 0.47578
```

```
## sex:rank       932237  2  0.0761 0.92688
## sex:degree    7164815  1  1.1701 0.28773
## sex:year      7194388  1  1.1749 0.28676
## sex:yd        2024210  1  0.3306 0.56947
## rank:degree  13021265  2  1.0632 0.35759
## rank:year     1571933  2  0.1284 0.88001
## rank:yd       9822382  2  0.8020 0.45750
## degree:year   4510249  1  0.7366 0.39735
## degree:yd     6407880  1  1.0465 0.31424
## year:yd         50921  1  0.0083 0.92793
## Residuals   189825454 31
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# This time I use BIC for model reduction by specifying k=
#   (compare this to model result using AIC --
#      too many nonsignificant parameters left in model)

## BIC
# option: test="F" includes additional information
#           for parameter estimate tests that we're familiar with
# option: for BIC, include k=log(nrow( [data.frame name] ))
lm.faculty.red.BIC <- step(lm.faculty.full, direction="backward", test="F"
                           , k=log(nrow(faculty)))
## Start:  AIC=868.72
## salary ~ (sex + rank + degree + year + yd)^2
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - sex:rank     2    932237 190757690 861.07  0.0761 0.9269
## - rank:year    2   1571933 191397386 861.24  0.1284 0.8800
## - rank:yd      2   9822382 199647836 863.44  0.8020 0.4575
## - rank:degree  2  13021265 202846719 864.26  1.0632 0.3576
## - year:yd      1     50921 189876375 864.78  0.0083 0.9279
## - sex:yd       1   2024210 191849663 865.32  0.3306 0.5695
## - degree:year  1   4510249 194335703 865.99  0.7366 0.3974
## - degree:yd    1   6407880 196233334 866.49  1.0465 0.3142
## - sex:degree   1   7164815 196990268 866.69  1.1701 0.2877
## - sex:year     1   7194388 197019841 866.70  1.1749 0.2868
## <none>                     189825454 868.72
##
## Step:  AIC=861.07
## salary ~ sex + rank + degree + year + yd + sex:degree + sex:year +
##     sex:yd + rank:degree + rank:year + rank:yd + degree:year +
##     degree:yd + year:yd
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - rank:year    2   4480611 195238301 854.37  0.3876 0.6818
## - rank:yd      2  14587933 205345624 857.00  1.2618 0.2964
## - year:yd      1     25889 190783580 857.12  0.0045 0.9470
```

```
## - rank:degree  2  16365099 207122790 857.45  1.4155 0.2572
## - sex:yd       1   3293276 194050966 858.01  0.5697 0.4557
## - degree:year  1   4428068 195185758 858.31  0.7660 0.3878
## - degree:yd    1   6525075 197282766 858.87  1.1288 0.2957
## - sex:year     1  10462381 201220071 859.89  1.8099 0.1877
## - sex:degree   1  10654937 201412628 859.94  1.8432 0.1838
## <none>                     190757690 861.07
##
## Step:  AIC=854.37
## salary ~ sex + rank + degree + year + yd + sex:degree + sex:year +
##     sex:yd + rank:degree + rank:yd + degree:year + degree:yd +
##     year:yd
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - year:yd      1    582367 195820669 850.58  0.1044 0.7485
## - rank:degree  2  18612514 213850816 851.21  1.6683 0.2032
## - sex:yd       1   3008739 198247041 851.22  0.5394 0.4676
## - rank:yd      2  20258184 215496486 851.60  1.8158 0.1777
## - degree:year  1   7497925 202736226 852.38  1.3441 0.2542
## - degree:yd    1   8179958 203418259 852.56  1.4664 0.2340
## - sex:degree   1  12500896 207739197 853.65  2.2410 0.1434
## - sex:year     1  12669105 207907406 853.69  2.2712 0.1408
## <none>                     195238301 854.37
##
## Step:  AIC=850.58
## salary ~ sex + rank + degree + year + yd + sex:degree + sex:year +
##     sex:yd + rank:degree + rank:yd + degree:year + degree:yd
##
##               Df Sum of Sq       RSS    AIC F value  Pr(>F)
## - sex:yd       1   2456466 198277134 847.27  0.4516 0.50587
## - rank:degree  2  21836322 217656990 848.17  2.0072 0.14912
## - degree:year  1   7414066 203234734 848.56  1.3630 0.25069
## - degree:yd    1   9232872 205053541 849.02  1.6974 0.20090
## - sex:degree   1  12831931 208652600 849.93  2.3590 0.13330
## - sex:year     1  13646799 209467467 850.13  2.5089 0.12196
## <none>                     195820669 850.58
## - rank:yd      2  41051000 236871669 852.57  3.7734 0.03253 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=847.27
## salary ~ sex + rank + degree + year + yd + sex:degree + sex:year +
##     rank:degree + rank:yd + degree:year + degree:yd
##
##               Df Sum of Sq       RSS    AIC F value  Pr(>F)
## - rank:degree  2  21157939 219435073 844.64  1.9741 0.15324
## - degree:year  1   8497324 206774458 845.50  1.5857 0.21583
## - degree:yd    1   9463400 207740534 845.75  1.7659 0.19202
```

```
## - sex:degree   1  10394382 208671516 845.98  1.9397 0.17202
## <none>                     198277134 847.27
## - sex:year     1  22789419 221066553 848.98  4.2527 0.04626 *
## - rank:yd      2  42516602 240793736 849.47  3.9670 0.02749 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=844.64
## salary ~ sex + rank + degree + year + yd + sex:degree + sex:year +
##     rank:yd + degree:year + degree:yd
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - degree:yd    1    361929 219797002 840.78  0.0643 0.8011
## - degree:year  1    855102 220290175 840.89  0.1520 0.6988
## - sex:degree   1   1616150 221051223 841.07  0.2872 0.5950
## - rank:yd      2  24391011 243826084 842.22  2.1675 0.1281
## - sex:year     1  10569795 230004869 843.14  1.8786 0.1783
## <none>                     219435073 844.64
##
## Step:  AIC=840.78
## salary ~ sex + rank + degree + year + yd + sex:degree + sex:year +
##     rank:yd + degree:year
##
##               Df Sum of Sq       RSS    AIC F value  Pr(>F)
## - sex:degree   1   3112507 222909509 837.56  0.5664 0.45609
## - degree:year  1   4414318 224211320 837.86  0.8033 0.37546
## - rank:yd      2  24695126 244492128 838.41  2.2471 0.11889
## - sex:year     1  16645026 236442028 840.62  3.0292 0.08947 .
## <none>                     219797002 840.78
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=837.56
## salary ~ sex + rank + degree + year + yd + sex:year + rank:yd +
##     degree:year
##
##               Df Sum of Sq       RSS    AIC F value Pr(>F)
## - degree:year  1   2585275 225494784 834.21  0.4755 0.4943
## - rank:yd      2  25367664 248277174 835.26  2.3330 0.1098
## - sex:year     1  14770974 237680484 836.94  2.7168 0.1069
## <none>                     222909509 837.56
##
## Step:  AIC=834.21
## salary ~ sex + rank + degree + year + yd + sex:year + rank:yd
##
##            Df Sum of Sq       RSS    AIC F value Pr(>F)
## - rank:yd   2  24905278 250400062 831.75  2.3194 0.1108
## - degree    1   8902098 234396882 832.27  1.6581 0.2049
```

```
## - sex:year  1  14134386 239629170 833.42  2.6326 0.1122
## <none>                 225494784 834.21
##
## Step:  AIC=831.75
## salary ~ sex + rank + degree + year + yd + sex:year
##
##            Df Sum of Sq       RSS    AIC F value    Pr(>F)
## - sex:year  1   8458303 258858365 829.53  1.4863   0.22929
## - degree    1  11217823 261617885 830.08  1.9712   0.16734
## - yd        1  16309342 266709404 831.08  2.8659   0.09755 .
## <none>                 250400062 831.75
## - rank      2 406263292 656663354 873.98 35.6941 6.144e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=829.53
## salary ~ sex + rank + degree + year + yd
##
##          Df Sum of Sq       RSS    AIC F value    Pr(>F)
## - sex     1   9134971 267993336 827.38  1.5880    0.2141
## - degree  1  10687589 269545954 827.68  1.8579    0.1796
## - yd      1  14868158 273726523 828.48  2.5847    0.1149
## <none>                258858365 829.53
## - year    1 144867403 403725768 848.69 25.1838 8.654e-06 ***
## - rank    2 399790682 658649047 870.19 34.7499 7.485e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=827.38
## salary ~ rank + degree + year + yd
##
##          Df Sum of Sq       RSS    AIC F value    Pr(>F)
## - degree  1   6684984 274678320 824.71  1.1475    0.2897
## - yd      1   7871680 275865016 824.93  1.3511    0.2511
## <none>                267993336 827.38
## - year    1 147642871 415636208 846.25 25.3423 7.839e-06 ***
## - rank    2 404108665 672102002 867.29 34.6818 6.544e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=824.71
## salary ~ rank + year + yd
##
##        Df Sum of Sq       RSS    AIC F value    Pr(>F)
## - yd    1   2314414 276992734 821.19   0.396    0.5322
## <none>              274678320 824.71
## - year  1 141105647 415783967 842.32  24.145 1.126e-05 ***
## - rank  2 478539101 753217421 869.26  40.941 5.067e-11 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=821.19
## salary ~ rank + year
##
##        Df Sum of Sq       RSS    AIC F value    Pr(>F)
## <none>              276992734 821.19
## - year  1 161953324 438946058 841.18  28.065 2.905e-06 ***
## - rank  2 632056217 909048951 875.09  54.764 4.103e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The `add1()` function will indicate whether a variable from the "full" model should be added to the current model. In our case, our BIC-backward selected model appears adequate.

```
add1(lm.faculty.red.BIC, . ~ (sex + rank + degree + year + yd)^2, test="F")
## Single term additions
##
## Model:
## salary ~ rank + year
##           Df Sum of Sq       RSS    AIC F value Pr(>F)
## <none>                 276992734 813.39
## sex        1   2304648 274688086 814.95  0.3943 0.5331
## degree     1   1127718 275865016 815.18  0.1921 0.6632
## yd         1   2314414 274678320 814.95  0.3960 0.5322
## rank:year  2  15215454 261777280 814.45  1.3368 0.2727
```

Let's look carefully at our resulting model.

```
# all are significant, stop.
# final model: salary ~ year + rank
lm.faculty.final <- lm.faculty.red.BIC

library(car)
Anova(lm.faculty.final, type=3)
## Anova Table (Type III tests)
##
## Response: salary
##                 Sum Sq Df F value    Pr(>F)
## (Intercept) 4422688839  1 766.407 < 2.2e-16 ***
## rank         632056217  2  54.764 4.103e-13 ***
## year         161953324  1  28.065 2.905e-06 ***
## Residuals    276992734 48
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.faculty.final)
```

```
##
## Call:
## lm(formula = salary ~ rank + year, data = faculty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3462.0 -1302.8  -299.2   783.5  9381.6
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 25657.79     926.81  27.684  < 2e-16 ***
## rankAssoc   -5192.24     871.83  -5.956 2.93e-07 ***
## rankAsst    -9454.52     905.83 -10.437 6.12e-14 ***
## year          375.70      70.92   5.298 2.90e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2402 on 48 degrees of freedom
## Multiple R-squared:  0.8449,Adjusted R-squared:  0.8352
## F-statistic: 87.15 on 3 and 48 DF,  p-value: < 2.2e-16
```

```
### comparing lsmeans (may be unbalanced)
library(lsmeans)
## compare levels of main effects
lsmeans(lm.faculty.final, list(pairwise ~ rank), adjust = "bonferroni")
```

```
## $`lsmeans of rank`
##  rank    lsmean       SE df lower.CL upper.CL
##  Full  28468.28 582.2789 48 27297.53 29639.03
##  Assoc 23276.05 642.2996 48 21984.62 24567.48
##  Asst  19013.76 613.0513 48 17781.14 20246.38
##
## Confidence level used: 0.95
##
## $`pairwise differences of contrast`
##  contrast     estimate       SE df t.ratio p.value
##  Full - Assoc 5192.239 871.8328 48   5.956  <.0001
##  Full - Asst  9454.523 905.8301 48  10.437  <.0001
##  Assoc - Asst 4262.285 882.8914 48   4.828  <.0001
##
## P value adjustment: bonferroni method for 3 tests
```

9.2.4 Discussion of the Salary Analysis

The selected model is a simple ANCOVA model with a rank effect and a linear effect due to years in rank. Note that the maximal model has 20 single df effects with an $R^2 = 0.89$ while the selected model has 3 single df effects with $R^2 = 0.84$.

Looking at the parameter estimates table, all of the single df effects in the selected

model are significant. The baseline group is Full Professors, with rank=3. Predicted salaries for the different ranks are given by:

$$\begin{aligned} \text{Full:} \quad & \widehat{\text{salary}} = 25658 + 375.70\ \text{year} \\ \text{Assoc:} \quad & \widehat{\text{salary}} = 25658 - 5192 + 375.70\ \text{year} = 20466 + 375.70\ \text{year} \\ \text{Assis:} \quad & \widehat{\text{salary}} = 25658 - 9454 + 375.70\ \text{year} = 16204 + 375.70\ \text{year} \end{aligned}$$

Do you remember how to interpret the **lsmeans**, and the p-values for comparing **lsmeans**?

You might be tempted to conclude that rank and years in rank are the only effects that are predictive of salaries, and that differences in salaries by sex are insignificant, once these effects have been taken into account. However, you must be careful because you have not done a diagnostic analysis. The following two issues are also important to consider.

A sex effect may exist even though there is insufficient evidence to support it based on these data. (**Lack of power corrupts; and absolute lack of power corrupts absolutely!**) If we are interested in the possibility of a sex effect, I think that we would do better by focusing on how large the effect might be, and whether it is important. A simple way to check is by constructing a confidence interval for the sex effect, based on a simple additive model that includes sex plus the effects that were selected as statistically significant, rank and year in rank. I am choosing this model because the omitted effects are hopefully small, and because the regression coefficient for a sex indicator is easy to interpret in an additive model. Other models might be considered for comparison. Summary output from this model is given below.

```
# add sex to the model
lm.faculty.final.sex <- update(lm.faculty.final, . ~ . + sex)
summary(lm.faculty.final.sex)
```

```
##
## Call:
## lm(formula = salary ~ rank + year + sex, data = faculty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3286.3 -1311.8  -178.4   939.1  9002.7
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 25390.65    1025.14  24.768  < 2e-16 ***
## rankAssoc   -5109.93     887.12  -5.760 6.20e-07 ***
## rankAsst    -9483.84     912.79 -10.390 9.19e-14 ***
## year          390.94      75.38   5.186 4.47e-06 ***
## sexFemale     524.15     834.69   0.628    0.533
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2418 on 47 degrees of freedom
## Multiple R-squared:  0.8462,Adjusted R-squared:  0.8331
## F-statistic: 64.64 on 4 and 47 DF,  p-value: < 2.2e-16
```

Men are the baseline group for the sex effect, so the predicted salaries for men are 524 dollars **less** than that for women, adjusting for rank and year. A rough 95% CI for the sex differential is the estimated sex coefficient plus or minus two standard errors, or $524 \pm 2 * (835)$, or -1146 to 2194 dollars. The range of plausible values for the sex effect would appear to contain values of practical importance, so further analysis is warranted here.

Another concern, and potentially a more important issue, was raised by M. O. Finkelstein in a 1980 discussion in the **Columbia Law Review** on the use of regression in discrimination cases: "... **[a] variable may reflect a position or status bestowed by the employer, in which case if there is discrimination in the award of the position or status, the variable may be 'tainted'.**" *Thus, if women are unfairly held back from promotion through the faculty ranks, then using faculty rank to adjust salary before comparing sexes may not be acceptable to the courts.* This suggests that an analysis comparing sexes but ignoring rank effects might be justifiable. What happens if this is done?

```
lm.faculty.sex.yd <- lm(salary ~ sex + yd, data = faculty)
library(car)
Anova(lm.faculty.sex.yd, type=3)
## Anova Table (Type III tests)
##
## Response: salary
##                 Sum Sq Df  F value    Pr(>F)
## (Intercept) 4275963832  1 231.4448 < 2.2e-16 ***
## sex           67178787  1   3.6362   0.06241 .
## yd           766344185  1  41.4799 4.883e-08 ***
## Residuals    905279453 49
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.faculty.sex.yd)
##
## Call:
## lm(formula = salary ~ sex + yd, data = faculty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9631.7 -2529.4     3.5  2298.0 13125.7
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 18355.23    1206.52  15.213  < 2e-16 ***
## sexFemale   -2572.53    1349.08  -1.907   0.0624 .
## yd            380.69      59.11   6.440 4.88e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4298 on 49 degrees of freedom
## Multiple R-squared:  0.493,Adjusted R-squared:  0.4724
## F-statistic: 23.83 on 2 and 49 DF,  p-value: 5.911e-08
```

Similar result as before, insufficient evidence between sexes (due to large proportion of variability in salary explained by yd [which I'm using in place of year since year is paired with rank]). Furthermore (not shown), there is insufficient evidence for a sex:yd interaction. However, rank and sex are (potentially) confounded. This data can not resolve this question. Instead, data on promotions would help resolve this issue.

Chapter 10

Automated Model Selection for Multiple Regression

Given data on a response variable Y and k predictors or binary variables $X_1, X_2, \ldots, X_k$, we wish to develop a regression model to predict Y. Assuming that the collection of variables is measured on the correct scale, and that the candidate list of effects includes all the important predictors or binary variables, the most general model is

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k + \varepsilon.$$

In most problems one or more of the effects can be eliminated from the full model without loss of information. We want to identify the important effects, or equivalently, eliminate the effects that are not very useful for explaining the variation in Y.

We will study several automated non-hierarchical methods for model selection. Given a specific criterion for selecting a model, a method gives the best model. *Before applying these methods*, plot Y against each predictor to see whether transformations are needed. Although transformations of binary variables are not necessary, side-by-side boxplots of the response across the levels of a factor give useful information on the predictive ability of the factor. If a transformation of X_i is suggested, include the transformation along with the original X_i in the candidate list. Note that we can transform the predictors differently, for example, $\log(X_1)$ and $\sqrt{X_2}$. However, if several transformations are suggested for the response, then one should consider doing one analysis for each suggested response scale before deciding on the final scale.

Different criteria for selecting models lead to different "best models." Given a collection of candidates for the best model, we make a choice of model on the basis of (1) a direct comparison of models, if possible (2) examination of model adequacy (residuals, influence, etc.) (3) simplicity — all things being equal, simpler models are preferred, and (4) scientific plausibility.

I view the various criteria as a means to generate interesting models for further consideration. I do not take any of them literally as best.

You should recognize that automated model selection methods should not replace scientific theory when building models! Automated methods are best suited for exploratory analyses, in situations where the researcher has little scientific information as a guide.

AIC/BIC were discussed in Section 3.2.1 for stepwise procedures and were used in examples in Chapter 9. In those examples, I included the corresponding F-tests in the ANOVA table as a criterion for dropping variables from a model. The next few sections cover these methods in more detail, then discuss other criteria and selections strategies, finishing with a few examples.

10.1 Forward Selection

In forward selection we add variables to the model one at a time. The steps in the procedure are:

1. Find the variable in the candidate list with the largest correlation (ignoring the sign) with Y. This variable gives a simple linear regression model with the largest R^2. Suppose this is X_1. Then fit the model

$$Y = \beta_0 + \beta_1 X_1 + \varepsilon \tag{10.1}$$

   and test $H_0 : \beta_1 = 0$. If we reject H_0, go to step 2. Otherwise stop and conclude that no variables are important. A t-test can be used here, or the equivalent ANOVA F-test.
2. Find the remaining variable which when added to model (10.1) increases R^2 the most (or equivalently decreases Residual SS the most). Suppose this is X_2. Fit the model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon \tag{10.2}$$

   and test $H_0 : \beta_2 = 0$. If we do not reject H_0, stop and use model (10.1) to predict Y. If we reject H_0, replace model (10.1) with (10.2) and repeat step 2 sequentially until no further variables are added to the model.

In forward selection we sequentially isolate the most important effect left in the pool, and check whether it is needed in the model. If it is needed we continue the process. Otherwise we stop.

The F-test default level for the tests on the individual effects is sometimes set as high as $\alpha = 0.50$ (SAS default). This may seem needlessly high. However, in many problems certain variables may be important only in the presence of other variables. If we force the forward selection to test at standard levels then the process will never get "going" when none of the variables is important on its own.

10.2 Backward Elimination

The backward elimination procedure (discussed earlier this semester) deletes unimportant variables, one at a time, starting from the full model. The steps is the procedure are:

1. Fit the full model

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k + \varepsilon. \quad (10.3)$$

2. Find the variable which when omitted from the full model (10.3) reduces R^2 the least, or equivalently, increases the Residual SS the least. This is the variable that gives the largest p-value for testing an individual regression coefficient $H_0 : \beta_j = 0$ for $j > 0$. Suppose this variable is X_k. If you reject H_0, stop and conclude that the full model is best. If you do not reject H_0, delete X_k from the full model, giving the new full model

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_{k-1} X_{k-1} + \varepsilon$$

to replace (10.3). Repeat steps 1 and 2 sequentially until no further variables can be deleted.

In backward elimination we isolate the least important effect left in the model, and check whether it is important. If not, delete it and repeat the process. Otherwise, stop. The default test level on the individual variables is sometimes set at $\alpha = 0.10$ (SAS default).

10.3 Stepwise Regression

Stepwise regression combines features of forward selection and backward elimination. A deficiency of forward selection is that variables can not be omitted from the model once they are selected. This is problematic because many variables that are initially important are not important once several other variables are included in the model. In stepwise regression, we add variables to the model as in forward regression, but include a backward elimination step every time a new variable is added. That is, every time we add a variable to the model we ask whether any of the variables added earlier can be omitted. If variables can be omitted, they are placed back into the candidate pool for consideration at the next step of the process. The process continues until no additional variables can be added, and none of the variables in the model can be excluded. The procedure can start from an empty model, a full model, or an intermediate model, depending on the software.

The p-values used for including and excluding variables in stepwise regression are usually taken to be equal (why is this reasonable?), and sometimes set at $\alpha = 0.15$ (SAS default).

10.3.1 Example: Indian systolic blood pressure

We revisit the example first introduced in Chapter 2. Anthropologists conducted a study to determine the long-term effects of an environmental change on systolic blood pressure. They measured the blood pressure and several other characteristics of 39 Indians who migrated from a very primitive environment high in the Andes into the mainstream of Peruvian society at a lower altitude. All of the Indians were males at least 21 years of age, and were born at a high altitude.

Let us *illustrate* the three model selection methods to build a regression model, using systolic blood pressure (*sysbp*) as the response, and seven candidate predictors: wt = weight in kilos; ht = height in mm; $chin$ = chin skin fold in mm; $fore$ = forearm skin fold in mm; $calf$ = calf skin fold in mm; $pulse$ = pulse rate-beats/min, and $yrage$ = fraction, which is the proportion of each individual's lifetime spent in the new environment.

Below I generate simple summary statistics and plots. The plots do not suggest any apparent transformations of the response or the predictors, so we will analyze the data using the given scales.

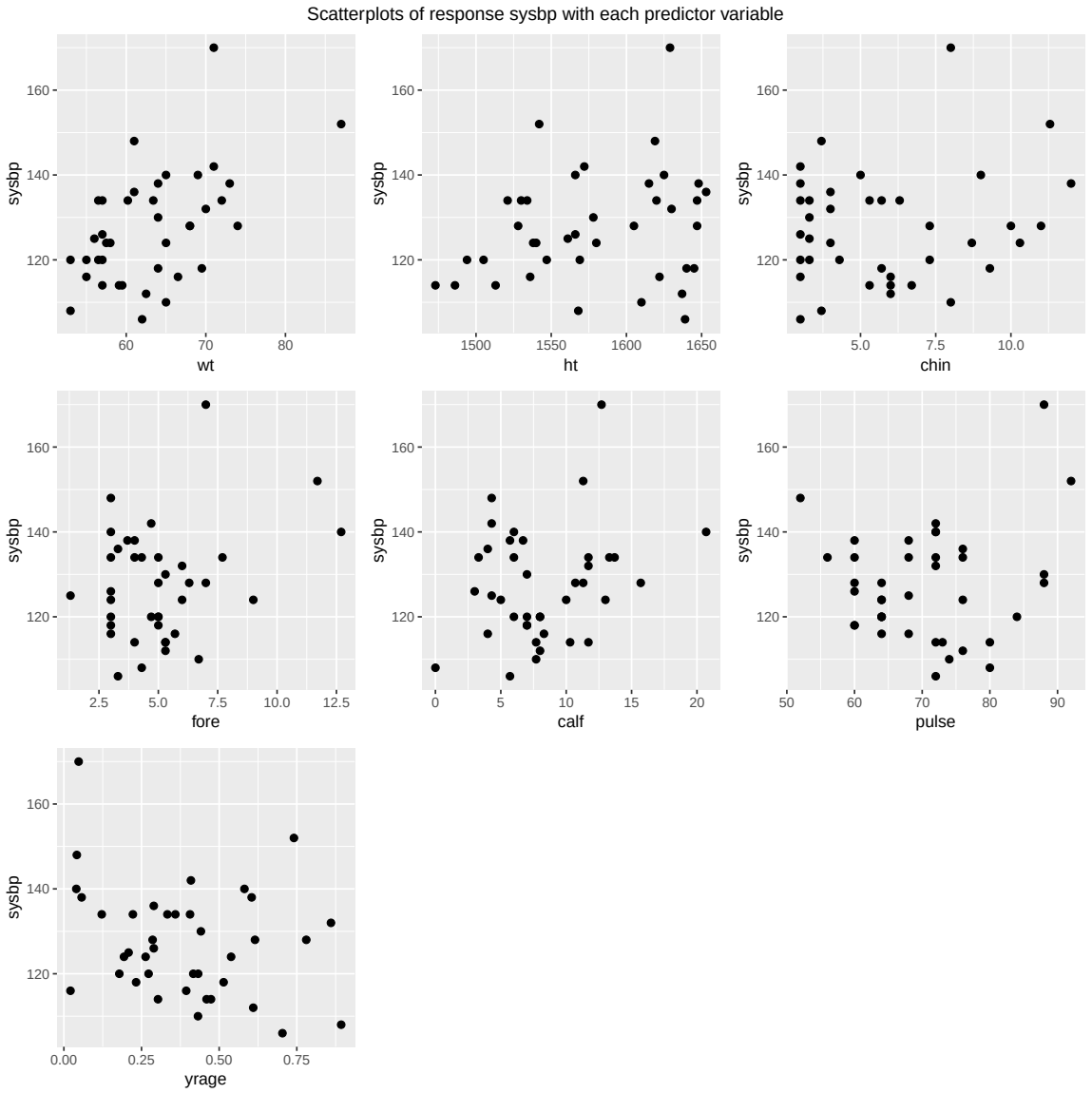
```
#### Example: Indian
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch02_indian.dat"
indian <- read.table(fn.data, header=TRUE)

# Description of variables
#    id = individual id
#   age = age in years           yrmig = years since migration
#    wt = weight in kilos           ht = height in mm
#  chin = chin skin fold in mm    fore = forearm skin fold in mm
#  calf = calf skin fold in mm   pulse = pulse rate-beats/min
# sysbp = systolic bp            diabp = diastolic bp

# Create the "fraction of their life" variable
#   yrage = years since migration divided by age
indian$yrage <- indian$yrmig / indian$age
```

```
# correlation matrix and associated p-values testing "H0: rho == 0"
library(Hmisc)
i.cor <- rcorr(as.matrix(indian[,c("sysbp", "wt", "ht", "chin"
                                   , "fore", "calf", "pulse", "yrage")]))
# print correlations with the response to 3 significant digits
signif(i.cor$r[1, ], 3)
```

```
##  sysbp     wt     ht   chin   fore   calf  pulse  yrage
##  1.000  0.521  0.219  0.170  0.272  0.251  0.133 -0.276
```

```
# scatterplots
library(ggplot2)
p1 <- ggplot(indian, aes(x = wt   , y = sysbp)) + geom_point(size=2)
p2 <- ggplot(indian, aes(x = ht   , y = sysbp)) + geom_point(size=2)
```

```
p3 <- ggplot(indian, aes(x = chin , y = sysbp)) + geom_point(size=2)
p4 <- ggplot(indian, aes(x = fore , y = sysbp)) + geom_point(size=2)
p5 <- ggplot(indian, aes(x = calf , y = sysbp)) + geom_point(size=2)
p6 <- ggplot(indian, aes(x = pulse, y = sysbp)) + geom_point(size=2)
p7 <- ggplot(indian, aes(x = yrage, y = sysbp)) + geom_point(size=2)

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4, p5, p6, p7), ncol=3
  , top = "Scatterplots of response sysbp with each predictor variable")
```



The `step()` function provides the forward, backward, and stepwise procedures based on AIC or BIC, and provides corresponding F-tests.

```
## step() function specification
## The first two arguments of step(object, scope, ...) are
# object = a fitted model object.
# scope  = a formula giving the terms to be considered for adding or dropping
## default is AIC
# for BIC, include k = log(nrow( [data.frame name] ))
# test="F" includes additional information
#          for parameter estimate tests that we're familiar with
```

Forward selection output The output for the forward selection method is below. BIC is our selection criterion, though similar decisions are made as if using F-tests.

Step 1 Variable *wt* =weight is entered first because it has the highest correlation with *sysbp* =sys bp. The corresponding F-value is the square of the t-statistic for testing the significance of the weight predictor in this simple linear regression model.

Step 2 Adding *yrage* =fraction to the simple linear regression model with weight as a predictor increases R^2 the most, or equivalently, decreases Residual SS (RSS) the most.

Step 3 The last table has "<none>" as the first row indicating that the current model (no change to current model) is the best under the current selection criterion.

```
# start with an empty model (just the intercept 1)
lm.indian.empty <- lm(sysbp ~ 1, data = indian)
# Forward selection, BIC with F-tests
lm.indian.forward.red.BIC <- step(lm.indian.empty
  , sysbp ~ wt + ht + chin + fore + calf + pulse + yrage
  , direction = "forward", test = "F", k = log(nrow(indian)))
## Start:  AIC=203.38
## sysbp ~ 1
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## + wt     1   1775.38 4756.1 194.67 13.8117 0.0006654 ***
## <none>               6531.4 203.38
## + yrage  1    498.06 6033.4 203.95  3.0544 0.0888139 .
## + fore   1    484.22 6047.2 204.03  2.9627 0.0935587 .
## + calf   1    410.80 6120.6 204.51  2.4833 0.1235725
## + ht     1    313.58 6217.9 205.12  1.8660 0.1801796
## + chin   1    189.19 6342.2 205.89  1.1037 0.3002710
## + pulse  1    114.77 6416.7 206.35  0.6618 0.4211339
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Step:  AIC=194.67
## sysbp ~ wt
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## + yrage  1   1314.69 3441.4 185.71 13.7530 0.0006991 ***
## <none>               4756.1 194.67
## + chin   1    143.63 4612.4 197.14  1.1210 0.2967490
## + calf   1     16.67 4739.4 198.19  0.1267 0.7240063
## + pulse  1      6.11 4749.9 198.28  0.0463 0.8308792
## + ht     1      2.01 4754.0 198.31  0.0152 0.9024460
## + fore   1      1.16 4754.9 198.32  0.0088 0.9257371
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=185.71
## sysbp ~ wt + yrage
##
##         Df Sum of Sq    RSS    AIC F value Pr(>F)
## <none>               3441.4 185.71
## + chin   1   197.372 3244.0 187.07  2.1295 0.1534
## + fore   1    50.548 3390.8 188.80  0.5218 0.4749
## + calf   1    30.218 3411.1 189.03  0.3101 0.5812
## + ht     1    23.738 3417.6 189.11  0.2431 0.6251
## + pulse  1     5.882 3435.5 189.31  0.0599 0.8081
summary(lm.indian.forward.red.BIC)
##
## Call:
## lm(formula = sysbp ~ wt + yrage, data = indian)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4330  -7.3070   0.8963   5.7275  23.9819
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  60.8959    14.2809   4.264 0.000138 ***
## wt            1.2169     0.2337   5.207 7.97e-06 ***
## yrage       -26.7672     7.2178  -3.708 0.000699 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.777 on 36 degrees of freedom
## Multiple R-squared:  0.4731,Adjusted R-squared:  0.4438
## F-statistic: 16.16 on 2 and 36 DF,  p-value: 9.795e-06
```

Backward selection output The output for the backward elimination method is below. BIC is our selection criterion, though similar decisions are made as if using

F-tests.

Step 0 The full model has 7 predictors so REG df = 7. The F-test in the full model ANOVA table ($F = 4.91$ with p-value=0.0008) tests the hypothesis that the regression coefficient for each predictor variable is zero. This test is highly significant, indicating that one or more of the predictors is important in the model.

The t-value column gives the t-statistic for testing the significance of the individual predictors in the full model conditional on the other variables being in the model.

```
# start with a full model
lm.indian.full <- lm(sysbp ~ wt + ht + chin + fore + calf + pulse + yrage, data = indian)
summary(lm.indian.full)
```

```
##
## Call:
## lm(formula = sysbp ~ wt + ht + chin + fore + calf + pulse + yrage,
##     data = indian)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.3993  -5.7916  -0.6907   6.9453  23.5771
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 106.45766   53.91303   1.975 0.057277 .
## wt            1.71095    0.38659   4.426 0.000111 ***
## ht           -0.04533    0.03945  -1.149 0.259329
## chin         -1.15725    0.84612  -1.368 0.181239
## fore         -0.70183    1.34986  -0.520 0.606806
## calf          0.10357    0.61170   0.169 0.866643
## pulse         0.07485    0.19570   0.383 0.704699
## yrage       -29.31810    7.86839  -3.726 0.000777 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.994 on 31 degrees of freedom
## Multiple R-squared:  0.5259,Adjusted R-squared:  0.4189
## F-statistic: 4.913 on 7 and 31 DF,  p-value: 0.0008079
```

The least important variable in the full model, as judged by the p-value, is $calf$ =calf skin fold. This variable, upon omission, reduces R^2 the least, or equivalently, increases the Residual SS the least. So $calf$ is the first to be omitted from the model.

Step 1 After deleting $calf$, the six predictor model is fitted. At least one of the predictors left is important, as judged by the overall F-test p-value. The least important predictor left is $pulse$ =pulse rate.

```r
# Backward selection, BIC with F-tests
lm.indian.backward.red.BIC <- step(lm.indian.full
  , direction = "backward", test = "F", k = log(nrow(indian)))
```

```
## Start:  AIC=199.91
## sysbp ~ wt + ht + chin + fore + calf + pulse + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - calf   1      2.86 3099.3 196.28  0.0287 0.8666427
## - pulse  1     14.61 3111.1 196.43  0.1463 0.7046990
## - fore   1     27.00 3123.4 196.59  0.2703 0.6068061
## - ht     1    131.88 3228.3 197.88  1.3203 0.2593289
## - chin   1    186.85 3283.3 198.53  1.8706 0.1812390
## <none>               3096.4 199.91
## - yrage  1   1386.76 4483.2 210.68 13.8835 0.0007773 ***
## - wt     1   1956.49 5052.9 215.35 19.5874 0.0001105 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=196.28
## sysbp ~ wt + ht + chin + fore + pulse + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - pulse  1     13.34 3112.6 192.79  0.1377 0.7130185
## - fore   1     26.99 3126.3 192.96  0.2787 0.6011969
## - ht     1    129.56 3228.9 194.22  1.3377 0.2560083
## - chin   1    184.03 3283.3 194.87  1.9000 0.1776352
## <none>               3099.3 196.28
## - yrage  1   1448.00 4547.3 207.57 14.9504 0.0005087 ***
## - wt     1   1953.77 5053.1 211.69 20.1724 8.655e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=192.79
## sysbp ~ wt + ht + chin + fore + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - fore   1     17.78 3130.4 189.35  0.1885  0.667013
## - ht     1    131.12 3243.8 190.73  1.3902  0.246810
## - chin   1    198.30 3310.9 191.53  2.1023  0.156514
## <none>               3112.6 192.79
## - yrage  1   1450.02 4562.7 204.04 15.3730  0.000421 ***
## - wt     1   1983.51 5096.2 208.35 21.0290 6.219e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=189.35
## sysbp ~ wt + ht + chin + yrage
##
```

```
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - ht     1    113.57 3244.0 187.07  1.2334 0.2745301
## - chin   1    287.20 3417.6 189.11  3.1193 0.0863479 .
## <none>               3130.4 189.35
## - yrage  1   1445.52 4575.9 200.49 15.7000 0.0003607 ***
## - wt     1   2263.64 5394.1 206.90 24.5857 1.945e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=187.07
## sysbp ~ wt + chin + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## - chin   1    197.37 3441.4 185.71  2.1295 0.1534065
## <none>               3244.0 187.07
## - yrage  1   1368.44 4612.4 197.14 14.7643 0.0004912 ***
## - wt     1   2515.33 5759.3 205.80 27.1384 8.512e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step:  AIC=185.71
## sysbp ~ wt + yrage
##
##         Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>               3441.4 185.71
## - yrage  1    1314.7 4756.1 194.67  13.753 0.0006991 ***
## - wt     1    2592.0 6033.4 203.95  27.115 7.966e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
summary(lm.indian.backward.red.BIC)
##
## Call:
## lm(formula = sysbp ~ wt + yrage, data = indian)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4330  -7.3070   0.8963   5.7275  23.9819
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  60.8959    14.2809   4.264 0.000138 ***
## wt            1.2169     0.2337   5.207 7.97e-06 ***
## yrage       -26.7672     7.2178  -3.708 0.000699 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.777 on 36 degrees of freedom
## Multiple R-squared:  0.4731,Adjusted R-squared:  0.4438
```

```
## F-statistic: 16.16 on 2 and 36 DF,  p-value: 9.795e-06
```

In the final table we are unable to drop yrage or wt from the model.

Stepwise selection output The output for the stepwise selection is given below.

Variables are listed in the output tables in order that best improves the AIC/BIC criterion. In the stepwise case, BIC will decrease (improve) by considering variables to drop or add (indicated in the first column by − and +). Rather than printing a small table at each step of the `step()` procedure, we use `lm.XXX$anova` to print a summary of the drop/add choices made.

```
# Stepwise (both) selection, BIC with F-tests, starting with intermediate model
# (this is a purposefully chosen "opposite" model,
#    from the forward and backward methods this model
#    includes all the variables dropped and none kept)
lm.indian.intermediate <- lm(sysbp ~ ht + fore + calf + pulse, data = indian)
# option: trace = 0 does not print each step of the selection
lm.indian.both.red.BIC <- step(lm.indian.intermediate
  , sysbp ~ wt + ht + chin + fore + calf + pulse + yrage
  , direction = "both", test = "F", k = log(nrow(indian)), trace = 0)
# the anova object provides a summary of the selection steps in order
lm.indian.both.red.BIC$anova
```

```
##      Step Df    Deviance Resid. Df Resid. Dev      AIC
## 1         NA          NA        34   5651.131 212.3837
## 2 - pulse  1    2.874432        35   5654.005 208.7400
## 3  - calf  1   21.843631        36   5675.849 205.2268
## 4    + wt -1  925.198114        35   4750.651 201.9508
## 5 + yrage -1 1439.707117        34   3310.944 191.5335
## 6    - ht  1   79.870793        35   3390.815 188.7995
## 7  - fore  1   50.548149        36   3441.363 185.7131
```

```
summary(lm.indian.both.red.BIC)
```

```
##
## Call:
## lm(formula = sysbp ~ wt + yrage, data = indian)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4330  -7.3070   0.8963   5.7275  23.9819
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  60.8959    14.2809   4.264 0.000138 ***
## wt            1.2169     0.2337   5.207 7.97e-06 ***
## yrage       -26.7672     7.2178  -3.708 0.000699 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 9.777 on 36 degrees of freedom
## Multiple R-squared:  0.4731,Adjusted R-squared:  0.4438
## F-statistic: 16.16 on 2 and 36 DF,  p-value: 9.795e-06
```

Summary of three section methods

All three methods using BIC choose the same final model, sysbp $= \beta_0 + \beta_1$ wt $+$ β_2 yrage. Using the AIC criterion, you will find different results.

10.4 Other Model Selection Procedures

10.4.1 R^2 Criterion

R^2 is the proportion of variation explained by the model over the grand mean, and we wish to maximize this. A substantial increase in R^2 is usually observed when an "important" effect is added to a regression model. With the R^2 criterion, variables are added to the model until further additions give inconsequential increases in R^2. The R^2 criterion is not well-defined in the sense of producing a single best model. All other things being equal, I prefer the simpler of two models with similar values of R^2. If several models with similar complexity have similar R^2s, then there may be no good reason, at this stage of the analysis, to prefer one over the rest.

10.4.2 Adjusted-R^2 Criterion, maximize

The adjusted-R^2 criterion gives a way to compare R^2 across models with different numbers of variables, and we want to maximize this. This eliminates some of the difficulty with calibrating R^2, which increases even when unimportant predictors are added to a model. For a model with p variables and an intercept, the adjusted-R^2 is defined by

$$\bar{R}^2 = 1 - \frac{n-1}{n-p-1}(1-R^2),$$

where n is the sample size.

There are four properties of $\bar{R}^2$ worth mentioning:

1. $\bar{R}^2 \leq R^2$,
2. if two models have the same number of variables, then the model with the larger R^2 has the larger $\bar{R}^2$,
3. if two models have the same R^2, then the model with fewer variables has the larger adjusted-R^2. Put another way, $\bar{R}^2$ *penalizes complex models with many variables.* And

4. $\bar{R}^2$ can be less than zero for models that explain little of the variation in Y.

The adjusted R^2 is easier to calibrate than R^2 because it tends to *decrease* when unimportant variables are added to a model. The model with the maximum $\bar{R}^2$ is judged best by this criterion. As I noted before, I do not take any of the criteria literally, and would also choose other models with $\bar{R}^2$ near the maximum value for further consideration.

10.4.3 Mallows' C_p Criterion, minimize

Mallows' C_p measures the adequacy of predictions from a model, relative to those obtained from the full model, and we want to minimize C_p. Mallows' C_p statistic is defined for a given model with p variables by

$$C_p = \frac{\text{Residual SS}}{\hat{\sigma}^2_{\text{FULL}}} - \text{Residual df} + (p+1)$$

where $\hat{\sigma}^2_{\text{FULL}}$ is the Residual MS from the full model with k variables X_1, X_2, ..., X_k.

If all the important effects from the candidate list are included in the model, then the difference between the first two terms of C_p should be approximately zero. Thus, if the model under consideration includes all the important variables from the candidate list, then C_p should be approximately $p+1$ (the number of variables in model plus one), or less. If important variables from the candidate list are excluded, C_p will tend to be much greater than $p+1$.

Two important properties of C_p are

1. the full model has $C_p = p+1$, where $p = k$, and
2. if two models have the same number of variables, then the model with the larger R^2 has the smaller C_p.

Models with $C_p \approx p+1$, or less, merit further consideration. As with R^2 and $\bar{R}^2$, I prefer simpler models that satisfy this condition. The "best" model by this criterion has the minimum C_p.

10.5 Illustration with Peru Indian data

R^2 Criterion

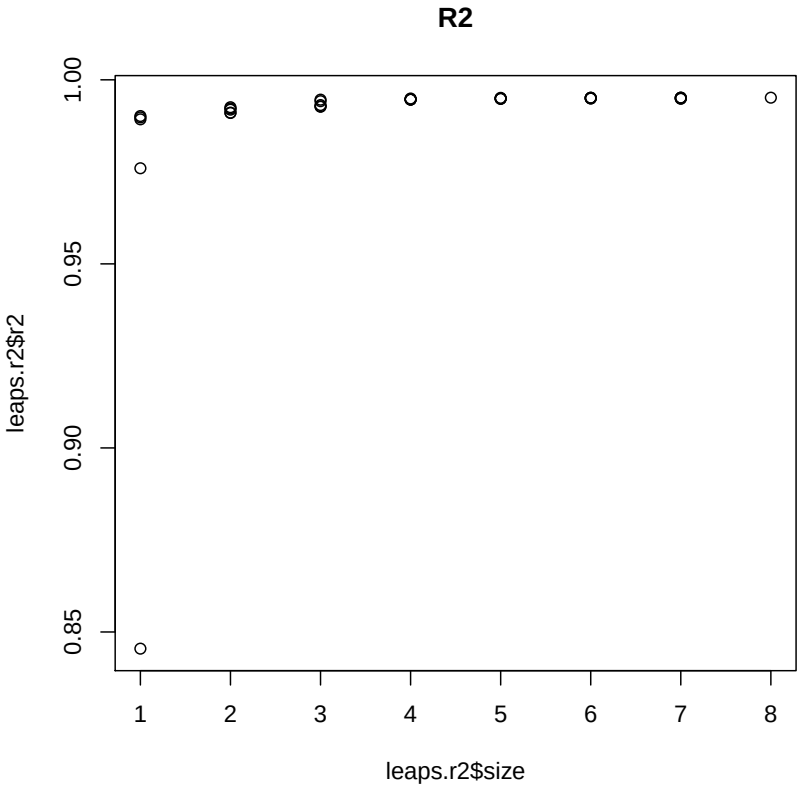
```
# The leaps package provides best subsets with other selection criteria.
library(leaps)

# First, fit the full model
lm.indian.full <- lm(sysbp ~ wt + ht + chin + fore + calf + pulse + yrage, data = indian)

# Second, create the design matrix which leap uses as argument
```

```
#   using model.matrix(lm.XXX) as input to leaps()


# R^2 -- for each model size, report best subset of size 5
leaps.r2 <- leaps(x = model.matrix(lm.indian.full), y = indian$sysbp
                , method = 'r2'
                , int = FALSE, nbest = 5, names = colnames(model.matrix(lm.indian.full)))
str(leaps.r2)
```

```
## List of 4
##  $ which: logi [1:36, 1:8] FALSE TRUE FALSE FALSE FALSE TRUE ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:36] "1" "1" "1" "1" ...
##   .. ..$ : chr [1:8] "(Intercept)" "wt" "ht" "chin" ...
##  $ label: chr [1:8] "(Intercept)" "wt" "ht" "chin" ...
##  $ size : num [1:36] 1 1 1 1 1 2 2 2 2 2 ...
##  $ r2   : num [1:36] 0.99 0.99 0.989 0.976 0.845 ...
```

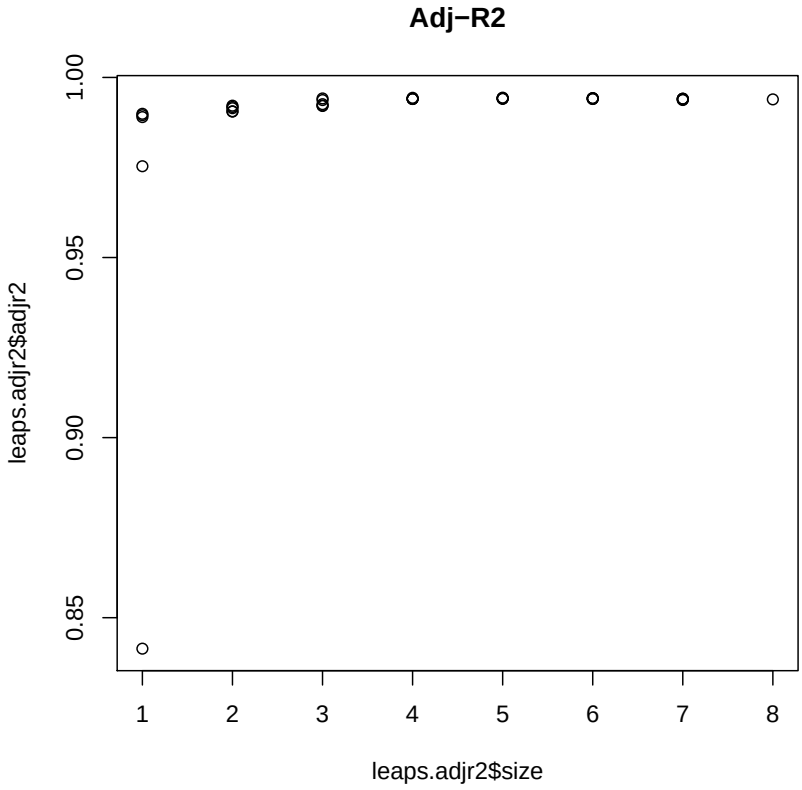
```
# plot model R^2 vs size of model
plot(leaps.r2$size, leaps.r2$r2, main = "R2")
# report the best model (indicate which terms are in the model)
best.model.r2 <- leaps.r2$which[which((leaps.r2$r2 == max(leaps.r2$r2))),]
# these are the variable names for the best model
names(best.model.r2)[best.model.r2]
```

```
## [1] "(Intercept)" "wt"          "ht"          "chin"
## [5] "fore"        "calf"        "pulse"       "yrage"
```



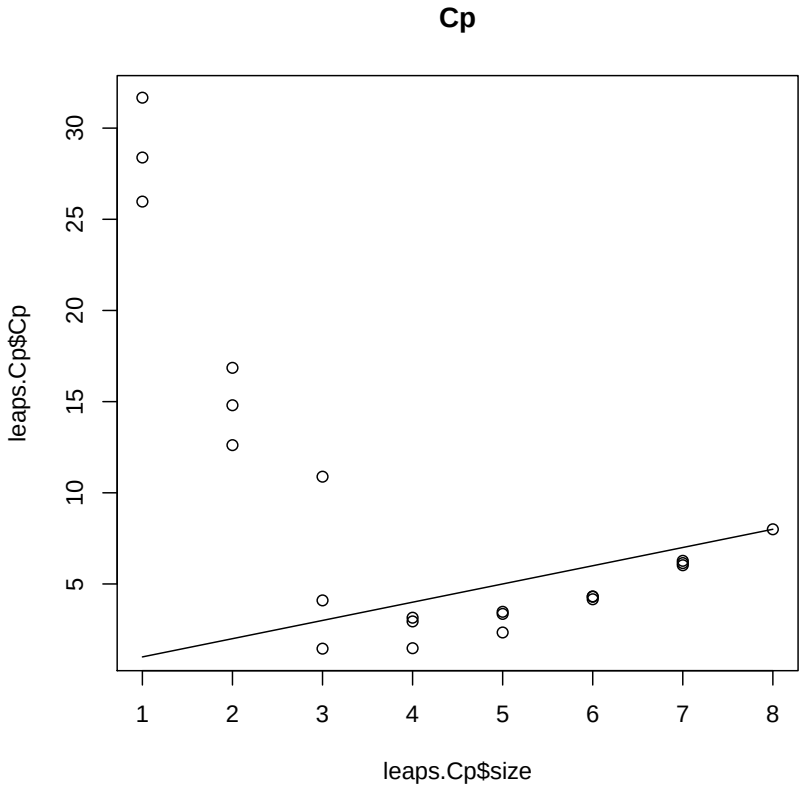
Adjusted-R^2 Criterion, maximize

```
# adj-R^2 -- for each model size, report best subset of size 5
leaps.adjr2 <- leaps(x = model.matrix(lm.indian.full), y = indian$sysbp
                , method = 'adjr2'
                , int = FALSE, nbest = 5, names = colnames(model.matrix(lm.indian.full)))
# plot model R^2 vs size of model
plot(leaps.adjr2$size, leaps.adjr2$adjr2, main = "Adj-R2")
# report the best model (indicate which terms are in the model)
best.model.adjr2 <- leaps.adjr2$which[which((leaps.adjr2$adjr2 == max(leaps.adjr2$adjr2))),]
# these are the variable names for the best model
names(best.model.adjr2)[best.model.adjr2]
## [1] "(Intercept)" "wt"          "ht"          "chin"
## [5] "yrage"
```



Mallows' C_p Criterion, minimize

```
# Cp -- for each model size, report best subset of size 3
leaps.Cp <- leaps(x = model.matrix(lm.indian.full), y = indian$sysbp
                , method = 'Cp'
                , int = FALSE, nbest = 3, names = colnames(model.matrix(lm.indian.full)))
# plot model R^2 vs size of model
plot(leaps.Cp$size, leaps.Cp$Cp, main = "Cp")
  lines(leaps.Cp$size, leaps.Cp$size)       # adds the line for Cp = p
# report the best model (indicate which terms are in the model)
best.model.Cp <- leaps.Cp$which[which((leaps.Cp$Cp == min(leaps.Cp$Cp))),]
# these are the variable names for the best model
names(best.model.Cp)[best.model.Cp]
## [1] "(Intercept)" "wt"          "yrage"
```



All together The function below takes `regsubsets()` output and formats it into a table.

```
# best subset, returns results sorted by BIC
f.bestsubset <- function(form, dat, nbest = 5){
  library(leaps)
  bs <- regsubsets(form, data=dat, nvmax=30, nbest=nbest, method="exhaustive");
  bs2 <- cbind(summary(bs)$which, (rowSums(summary(bs)$which)-1)
               , summary(bs)$rss, summary(bs)$rsq
               , summary(bs)$adjr2, summary(bs)$cp, summary(bs)$bic);
  cn <- colnames(bs2);
    cn[(dim(bs2)[2]-5):dim(bs2)[2]] <- c("SIZE", "rss", "r2", "adjr2", "cp", "bic");
    colnames(bs2) <- cn;
  ind <- sort.int(summary(bs)$bic, index.return=TRUE); bs2 <- bs2[ind$ix,];
  return(bs2);
}
# perform on our model
i.best <- f.bestsubset(formula(sysbp ~ wt + ht + chin + fore + calf + pulse + yrage)
          , indian)
  op <- options(); # saving old options
  options(width=90) # setting command window output text width wider
i.best
```

```
##   (Intercept) wt ht chin fore calf pulse yrage SIZE      rss         r2      adjr2
## 2           1  1  0    0    0    0     0     1    2 3441.363 0.47310778 0.44383599
## 3           1  1  0    1    0    0     0     1    3 3243.990 0.50332663 0.46075463
```

```
## 3           1  1  0    0    1    0     0     1    3 3390.815 0.48084699 0.43634816
## 3           1  1  0    0    0    1     0     1    3 3411.145 0.47773431 0.43296868
## 3           1  1  1    0    0    0     0     1    3 3417.624 0.47674226 0.43189159
## 3           1  1  0    0    0    0     1     1    3 3435.481 0.47400828 0.42892328
## 4           1  1  1    1    0    0     0     1    4 3130.425 0.52071413 0.46432755
## 4           1  1  0    1    0    0     1     1    4 3232.168 0.50513668 0.44691747
## 4           1  1  0    1    1    0     0     1    4 3243.771 0.50336023 0.44493203
## 4           1  1  0    1    0    1     0     1    4 3243.988 0.50332702 0.44489490
## 4           1  1  1    0    1    0     0     1    4 3310.944 0.49307566 0.43343750
## 5           1  1  1    1    1    0     0     1    5 3112.647 0.52343597 0.45122930
## 5           1  1  1    1    0    0     1     1    5 3126.303 0.52134520 0.44882174
## 5           1  1  1    1    0    1     0     1    5 3128.297 0.52103997 0.44847027
## 5           1  1  0    1    1    0     1     1    5 3228.867 0.50564205 0.43073933
## 5           1  1  0    1    0    1     1     1    5 3231.936 0.50517225 0.43019835
## 1           1  1  0    0    0    0     0     0    1 4756.056 0.27182072 0.25214020
## 6           1  1  1    1    1    0     1     1    6 3099.310 0.52547798 0.43650510
## 6           1  1  1    1    1    1     0     1    6 3111.060 0.52367894 0.43436875
## 6           1  1  1    1    0    1     1     1    6 3123.448 0.52178233 0.43211651
## 2           1  1  0    1    0    0     0     0    2 4612.426 0.29381129 0.25457859
## 6           1  1  0    1    1    1     1     1    6 3228.324 0.50572524 0.41304872
## 2           1  1  0    0    0    1     0     0    2 4739.383 0.27437355 0.23406097
## 2           1  1  0    0    0    0     1     0    2 4749.950 0.27275566 0.23235320
## 2           1  1  1    0    0    0     0     0    2 4754.044 0.27212880 0.23169151
## 6           1  1  1    0    1    1     1     1    6 3283.293 0.49730910 0.40305455
## 7           1  1  1    1    1    1     1     1    7 3096.446 0.52591643 0.41886531
## 1           1  0  0    0    0    0     0     1    1 6033.372 0.07625642 0.05129038
## 1           1  0  0    0    1    0     0     0    1 6047.218 0.07413652 0.04911319
## 1           1  0  0    0    0    1     0     0    1 6120.639 0.06289527 0.03756811
## 1           1  0  1    0    0    0     0     0    1 6217.854 0.04801119 0.02228176
##          cp         bic
## 2  1.453122 -13.9989263
## 3  1.477132 -12.6388375
## 3  2.947060 -10.9124614
## 3  3.150596 -10.6793279
## 3  3.215466 -10.6053171
## 3  3.394238 -10.4020763
## 4  2.340175 -10.3650555
## 4  3.358774  -9.1176654
## 4  3.474934  -8.9779148
## 4  3.477106  -8.9753065
## 4  4.147436  -8.1785389
## 5  4.162196  -6.9236046
## 5  4.298910  -6.7528787
## 5  4.318869  -6.7280169
## 5  5.325728  -5.4939516
## 5  5.356448  -5.4569068
## 1 12.615145  -5.0439888
## 6  6.028670  -3.4275113
## 6  6.146308  -3.2799319
## 6  6.270326  -3.1249499
## 2 13.177196  -2.5763538
## 6  7.320289  -1.8369533
## 2 14.448217  -1.5173924
## 2 14.554009  -1.4305331
## 2 14.595000  -1.3969306
## 6  7.870614  -1.1784808
## 7  8.000000   0.1999979
## 1 25.402961   4.2336138
## 1 25.541579   4.3230122
## 1 26.276637   4.7936744
## 1 27.249897   5.4082455
```

```
options(op); # reset (all) initial options
```

10.5.1 R^2, $\bar{R}^2$, and C_p Summary for Peru Indian Data

Discussion of R^2 results:

1. The single predictor model with the highest value of R^2 has wt = weight as a predictor: $R^2 = 0.272$. All the other single predictor models have $R^2 < 0.10$.
2. The two predictor model with the highest value of R^2 has weight and $yrage$ = fraction as predictors: $R^2 = 0.473$. No other two predictor model has R^2 close to this.
3. All of the best three predictor models include weight and fraction as predictors. However, the increase in R^2 achieved by adding a third predictor is minimal.
4. None of the more complex models with four or more predictors provides a significant increase in R^2.

A good model using the R^2 criterion has two predictors, weight and yrage. The same conclusion is reached with $\bar{R}^2$, albeit the model with maximum $\bar{R}^2$ includes wt (weight), ht (height), $chin$ (chin skin fold), and $yrage$ (fraction) as predictors.

Discussion of C_p results:

1. None of the single predictor models is adequate. Each has $C_p \gg 1 + 1 = 2$, the target value.
2. The only adequate two predictor model has wt = weight and $yrage$ = fraction as predictors: $C_p = 1.45 < 2 + 1 = 3$. This is the minimum C_p model.
3. Every model with weight and fraction is adequate. Every model that excludes either weight or fraction is inadequate: $C_p \gg p + 1$.

According to C_p, any reasonable model must include both weight and fraction as predictors. Based on simplicity, I would select the model with these two predictors as a starting point. I can always add predictors if subsequent analysis suggests this is necessary!

10.5.2 Peru Indian Data Summary

The model selection procedures suggest three models that warrant further consideration.

Predictors	Methods suggesting model
wt, yrage	BIC via stepwise, forward, and backward elimination, and C_p
wt, yrage, chin	AIC via stepwise and backward selection
wt, yrage, chin, ht	AIC via forward selection, Adj-R2

I will give three reasons why I feel that the simpler model is preferable at this point:

1. It was suggested by 4 of the 5 methods (ignoring R^2).
2. Forward selection often chooses predictors that are not important, even when the significance level for inclusion is reduced from the default $\alpha = 0.50$ level.
3. The AIC/BIC forward and backward elimination outputs suggest that neither chin skin fold nor height is significant at any of the standard levels of significance used in practice. Look at the third and fourth steps of forward selection to see this.

Using a mechanical approach, we are led to a model with weight and yrage as predictors of systolic blood pressure. At this point we should closely examine this model. We did this earlier this semester and found that observation 1 (the individual with the largest systolic blood pressure) was fitted poorly by the model and potentially influential.

As noted earlier this semester, model selection methods can be highly influenced by outliers and influential cases. Thus, we should hold out case 1, and re-evaluate the various procedures to see whether case 1 unduly influenced the models selected. I will just note (not shown) that the selection methods point to the same model when case 1 is held out. After deleting case 1, there are no large residuals, extremely influential points, or any gross abnormalities in plots.

Both analyses suggest that the "best model" for predicting systolic blood pressure is

$$\text{sysbp} = \beta_0 + \beta_1 \text{ wt} + \beta_2 \text{ yrage} + \varepsilon.$$

Should case 1 be deleted? I have not fully explored this issue, but I will note that eliminating this case does have a significant impact on the least squares estimates of the regression coefficients, and on predicted values. What do you think?

10.6 Example: Oxygen Uptake

An experiment was conducted to model oxygen uptake (o2up), in milligrams of oxygen per minute, from five chemical measurements: biological oxygen demand (bod), total Kjeldahl nitrogen (tkn), total solids (ts), total volatile solids (tvs), which is a component of ts, and chemical oxygen demand (cod), each measured in milligrams per liter. The data were collected on samples of dairy wastes kept in suspension in water in a laboratory for 220 days. All observations were on the same sample over time. We desire an equation relating o2up to the other variables. The goal is to find variables that should be further studied with the eventual goal of developing a prediction equation (day should not be considered as a predictor).

We are interested in developing a regression model with o2up, or some function of o2up, as a response. The researchers believe that the predictor variables are more likely to be linearly related to $\log_{10}$(o2up) rather than o2up, so $\log_{10}$(o2up) was included in the data set. As a first step, we should plot o2up against the different

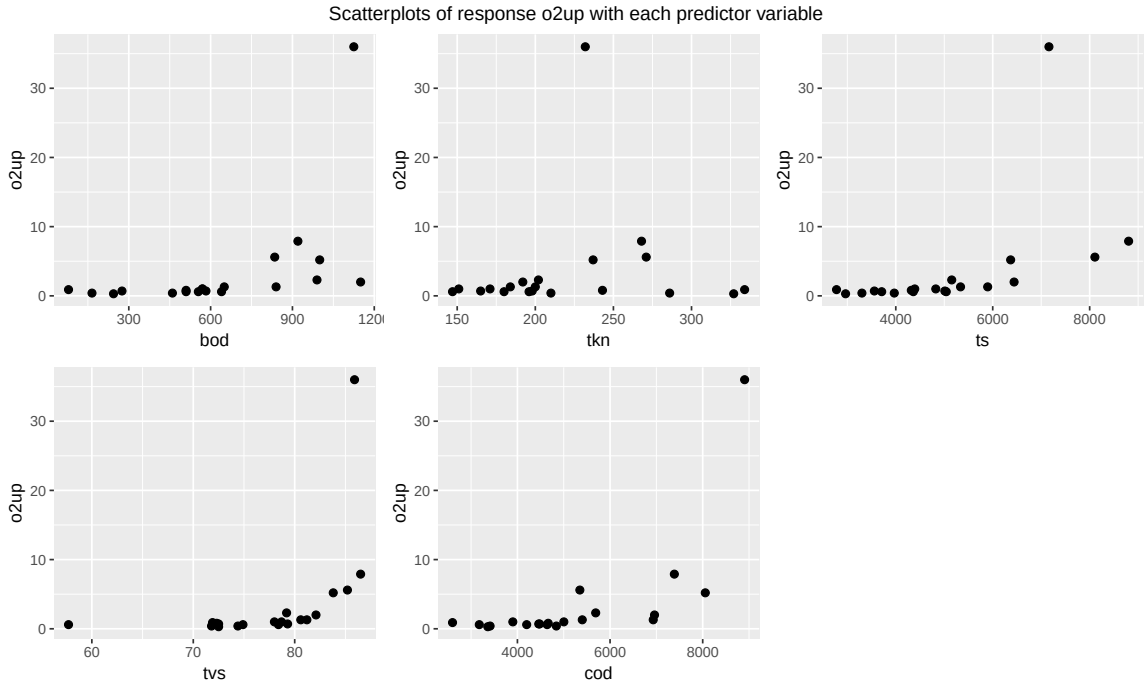
predictors, and see whether the relationship between o2up and the individual predictors is roughly linear. If not, we will consider appropriate transformations of the response and/or predictors.

```
#### Example: Oxygen uptake
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch10_oxygen.dat"
oxygen <- read.table(fn.data, header=TRUE)
```

	day	bod	tkn	ts	tvs	cod	o2up	logup
1	0	1125	232	7160	85.90	8905	36.00	1.56
2	7	920	268	8804	86.50	7388	7.90	0.90
3	15	835	271	8108	85.20	5348	5.60	0.75
4	22	1000	237	6370	83.80	8056	5.20	0.72
5	29	1150	192	6441	82.10	6960	2.00	0.30
6	37	990	202	5154	79.20	5690	2.30	0.36
7	44	840	184	5896	81.20	6932	1.30	0.11
8	58	650	200	5336	80.60	5400	1.30	0.11
9	65	640	180	5041	78.40	3177	0.60	−0.22
10	72	583	165	5012	79.30	4461	0.70	−0.15
11	80	570	151	4825	78.70	3901	1.00	0.00
12	86	570	171	4391	78.00	5002	1.00	0.00
13	93	510	243	4320	72.30	4665	0.80	−0.10
14	100	555	147	3709	74.90	4642	0.60	−0.22
15	107	460	286	3969	74.40	4840	0.40	−0.40
16	122	275	198	3558	72.50	4479	0.70	−0.15
17	129	510	196	4361	57.70	4200	0.60	−0.22
18	151	165	210	3301	71.80	3410	0.40	−0.40
19	171	244	327	2964	72.50	3360	0.30	−0.52
20	220	79	334	2777	71.90	2599	0.90	−0.05

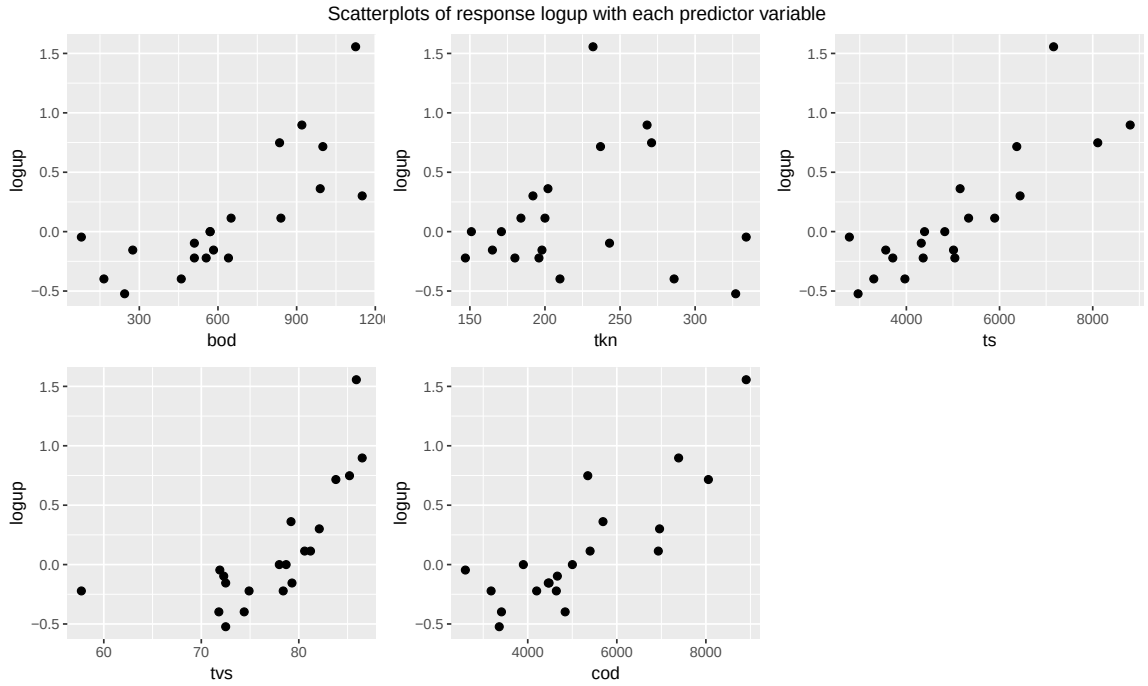
The plots showed an exponential relationship between o2up and the predictors. To shorten the output, only one plot is given. An exponential relationship can often be approximately linearized by transforming o2up to the $\log_{10}$(o2up) scale suggested by the researchers. The extreme skewness in the marginal distribution of o2up also gives an indication that a transformation might be needed.

```
# scatterplots
library(ggplot2)
p1 <- ggplot(oxygen, aes(x = bod, y = o2up)) + geom_point(size=2)
p2 <- ggplot(oxygen, aes(x = tkn, y = o2up)) + geom_point(size=2)
p3 <- ggplot(oxygen, aes(x = ts , y = o2up)) + geom_point(size=2)
p4 <- ggplot(oxygen, aes(x = tvs, y = o2up)) + geom_point(size=2)
p5 <- ggplot(oxygen, aes(x = cod, y = o2up)) + geom_point(size=2)

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4, p5), nrow=2
  , top = "Scatterplots of response o2up with each predictor variable")
```



After transformation, several plots show a roughly linear relationship. A sensible next step would be to build a regression model using log(o2up) as the response variable.

```
# scatterplots
library(ggplot2)
p1 <- ggplot(oxygen, aes(x = bod, y = logup)) + geom_point(size=2)
p2 <- ggplot(oxygen, aes(x = tkn, y = logup)) + geom_point(size=2)
p3 <- ggplot(oxygen, aes(x = ts , y = logup)) + geom_point(size=2)
p4 <- ggplot(oxygen, aes(x = tvs, y = logup)) + geom_point(size=2)
p5 <- ggplot(oxygen, aes(x = cod, y = logup)) + geom_point(size=2)

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4, p5), nrow=2
  , top = "Scatterplots of response logup with each predictor variable")
```



Correlation between response and each predictor.

```
# correlation matrix and associated p-values testing "HO: rho == 0"
library(Hmisc)
o.cor <- rcorr(as.matrix(oxygen[,c("logup", "bod", "tkn", "ts", "tvs", "cod")]))
# print correlations with the response to 3 significant digits
signif(o.cor$r[1, ], 3)
```

```
##  logup    bod    tkn     ts    tvs    cod
## 1.0000 0.7740 0.0906 0.8350 0.7110 0.8320
```

I used several of the model selection procedures to select out predictors. The model selection criteria below point to a more careful analysis of the model with ts and cod as predictors. This model has the minimum C_p and is selected by the backward and stepwise procedures. Furthermore, no other model has a substantially higher R^2 or $\bar{R}^2$. The fit of the model will not likely be improved substantially by adding any of the remaining three effects to this model.

```
# perform on our model
o.best <- f.bestsubset(formula(logup ~ bod + tkn + ts + tvs + cod)
          , oxygen, nbest = 3)
  op <- options(); # saving old options
  options(width=90) # setting command window output text width wider
o.best
```

```
##   (Intercept) bod tkn ts tvs cod SIZE       rss        r2     adjr2        cp       bic
## 2           1   0   0  1   0   1    2 1.0850469 0.7857080 0.7604972  1.738781 -21.82112
## 3           1   0   1  1   0   1    3 0.9871461 0.8050430 0.7684886  2.318714 -20.71660
## 3           1   0   0  1   1   1    3 1.0633521 0.7899926 0.7506163  3.424094 -19.22933
```

```
## 3           1   1   0  1   0   1    3 1.0643550 0.7897946 0.7503810  3.438642 -19.21047
## 4           1   0   1  1   1   1    4 0.9652627 0.8093649 0.7585288  4.001291 -18.16922
## 1           1   0   0  1   0   0    1 1.5369807 0.6964531 0.6795894  6.294153 -17.85292
## 2           1   0   0  0   1   1    2 1.3287305 0.7375816 0.7067088  5.273450 -17.76910
## 4           1   1   1  1   0   1    4 0.9871272 0.8050467 0.7530592  4.318440 -17.72124
## 1           1   0   0  0   0   1    1 1.5563027 0.6926371 0.6755614  6.574421 -17.60306
## 4           1   1   0  1   1   1    4 1.0388337 0.7948349 0.7401242  5.068450 -16.70015
## 2           1   0   1  0   0   1    2 1.4388035 0.7158426 0.6824124  6.870078 -16.17735
## 5           1   1   1  1   1   1    5 0.9651737 0.8093824 0.7413048  6.000000 -15.17533
## 1           1   1   0  0   0   0    1 2.0337926 0.5983349 0.5760202 13.500489 -12.25127
  options(op); # reset (all) initial options
```

These comments must be taken with a grain of salt because we have not critically assessed the underlying assumptions (linearity, normality, independence), nor have we considered whether the data contain influential points or outliers.

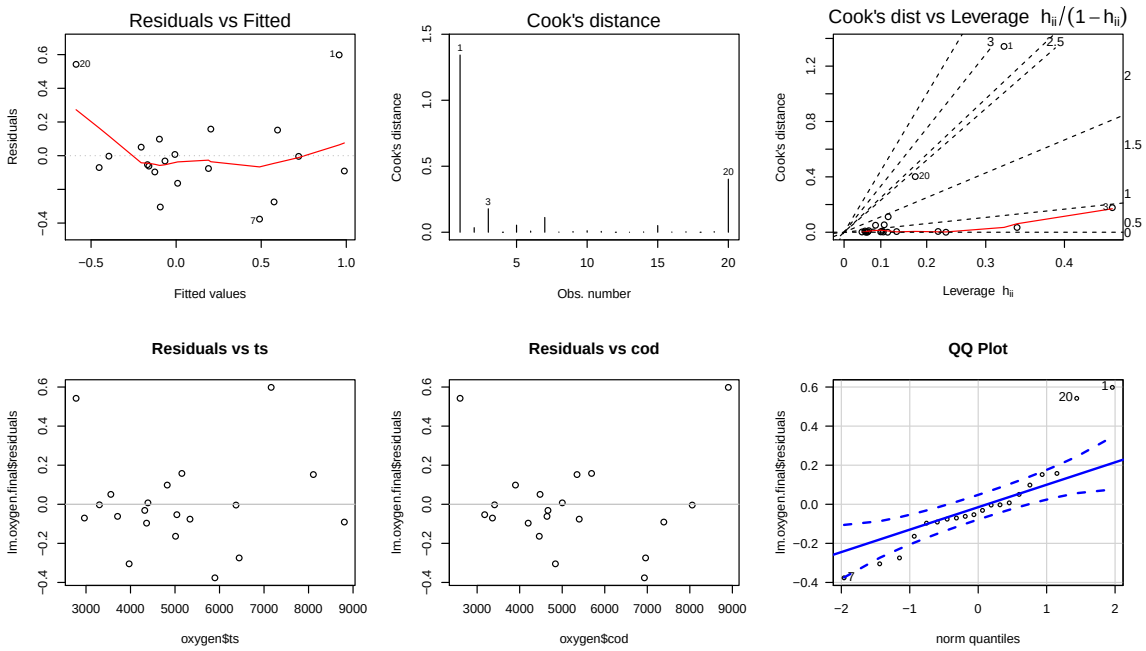
```
lm.oxygen.final <- lm(logup ~ ts + cod, data = oxygen)
summary(lm.oxygen.final)
```

```
##
## Call:
## lm(formula = logup ~ ts + cod, data = oxygen)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.37640 -0.09238 -0.04229  0.06256  0.59827
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.370e+00  1.969e-01  -6.960  2.3e-06 ***
## ts           1.492e-04  5.489e-05   2.717   0.0146 *
## cod          1.415e-04  5.318e-05   2.661   0.0165 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2526 on 17 degrees of freedom
## Multiple R-squared:  0.7857,Adjusted R-squared:  0.7605
## F-statistic: 31.17 on 2 and 17 DF,  p-value: 2.058e-06
```

The p-values for testing the importance of the individual predictors are small, indicating that both predictors are important. However, two observations (1 and 20) are poorly fitted by the model (both have $r_i > 2$) and are individually most influential (largest D_is). Recall that this experiment was conducted over 220 days, so these observations were the first and last data points collected. We have little information about the experiment, but it is reasonable to conjecture that the experiment may not have reached a steady state until the second time point, and that the experiment was ended when the experimental material dissipated. The end points of the experiment may not be typical of conditions under which we are interested in modelling oxygen

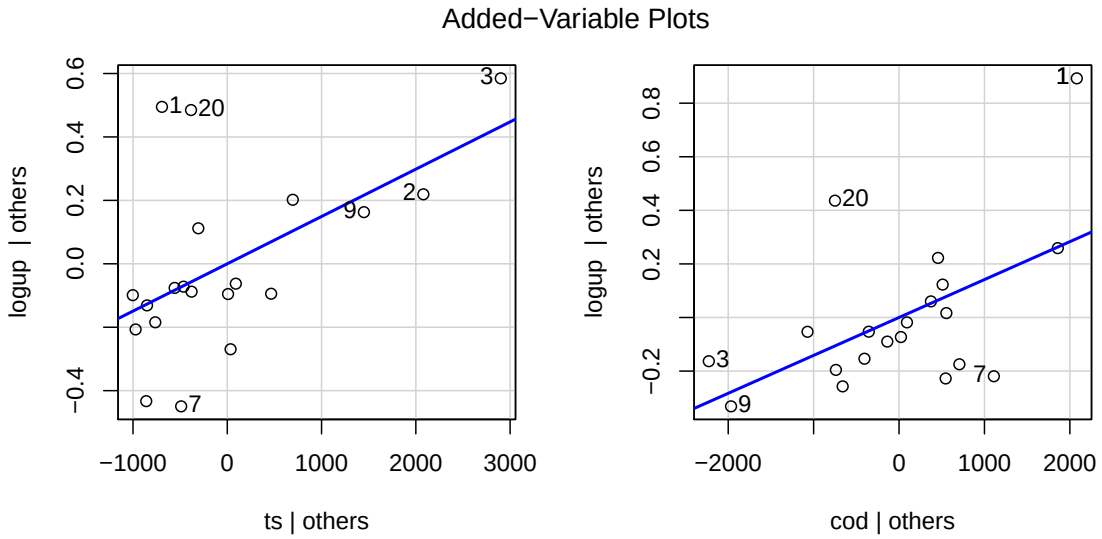
uptake. A sensible strategy here is to delete these points and redo the entire analysis to see whether our model changes noticeably.

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.oxygen.final, which = c(1,4,6))

plot(oxygen$ts, lm.oxygen.final$residuals, main="Residuals vs ts")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(oxygen$cod, lm.oxygen.final$residuals, main="Residuals vs cod")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
library(car)
qqPlot(lm.oxygen.final$residuals, las = 1, id = list(n = 3), main="QQ Plot")
## [1]  1 20  7

## residuals vs order of data
#plot(lm.oxygen.final$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```



Further, the partial residual plot for both `ts` and `cod` clearly highlights outlying cases 1 and 20.

```
library(car)
avPlots(lm.oxygen.final, id = list(n = 3))
```



10.6.1 Redo analysis excluding first and last observations

For more completeness, we exclude the end observations and repeat the model selection steps. Summaries from the model selection are provided.

The model selection criteria again suggest ts and cod as predictors. After deleting observations 1 and 20 the R^2 for this two predictor model jumps from 0.786 to 0.892. Also note that the LS coefficients change noticeably after these observations are deleted.

```
# exclude observations 1 and 20
oxygen2 <- oxygen[-c(1,20),]
```

Correlation between response and each predictor.

```
# correlation matrix and associated p-values testing "H0: rho == 0"
library(Hmisc)
o.cor <- rcorr(as.matrix(oxygen2[,c("logup", "bod", "tkn", "ts", "tvs", "cod")]))
# print correlations with the response to 3 significant digits
signif(o.cor$r[1, ], 3)
```

```
## logup   bod   tkn    ts   tvs   cod
## 1.000 0.813 0.116 0.921 0.717 0.806
```

```
# perform on our model
o.best <- f.bestsubset(formula(logup ~ bod + tkn + ts + tvs + cod)
          , oxygen2, nbest = 3)
  op <- options(); # saving old options
  options(width=90) # setting command window output text width wider
o.best
```

```
##   (Intercept) bod tkn ts tvs cod SIZE       rss        r2     adjr2         cp       bic
## 2           1   0   0  1   0   1    2 0.3100332 0.8923243 0.8779675  0.1755108 -31.44424
```

```
## 3           1   0   0  1   1   1    3 0.3060781 0.8936979 0.8709189  2.0201863 -28.78498
## 3           1   1   0  1   0   1    3 0.3092614 0.8925923 0.8695764  2.1452002 -28.59874
## 3           1   0   1  1   0   1    3 0.3095352 0.8924973 0.8694610  2.1559501 -28.58281
## 1           1   0   0  1   0   0    1 0.4346881 0.8490312 0.8395956  3.0709102 -28.25153
## 2           1   1   0  1   0   0    2 0.3832495 0.8668960 0.8491488  3.0508321 -27.62812
## 2           1   0   0  1   1   0    2 0.4182487 0.8547406 0.8353727  4.4253075 -26.05510
## 4           1   1   0  1   1   1    4 0.3056566 0.8938443 0.8611810  4.0036314 -25.91941
## 4           1   0   1  1   1   1    4 0.3057789 0.8938018 0.8611255  4.0084356 -25.91221
## 4           1   1   1  1   0   1    4 0.3091249 0.8926398 0.8596058  4.1398376 -25.71632
## 5           1   1   1  1   1   1    5 0.3055641 0.8938764 0.8496583  6.0000000 -23.03449
## 1           1   1   0  0   0   0    1 0.9766926 0.6607910 0.6395904 24.3563087 -13.67975
## 1           1   0   0  0   0   1    1 1.0100759 0.6491968 0.6272716 25.6673251 -13.07480
```

```
  options(op); # reset (all) initial options
```

Below is the model with ts and cod as predictors, after omitting the end observations. Both predictors are significant at the 0.05 level. Furthermore, there do not appear to be any extreme outliers. The QQ-plot, and the plot of studentized residuals against predicted values do not show any extreme abnormalities.

```
lm.oxygen2.final <- lm(logup ~ ts + cod, data = oxygen2)
summary(lm.oxygen2.final)
```

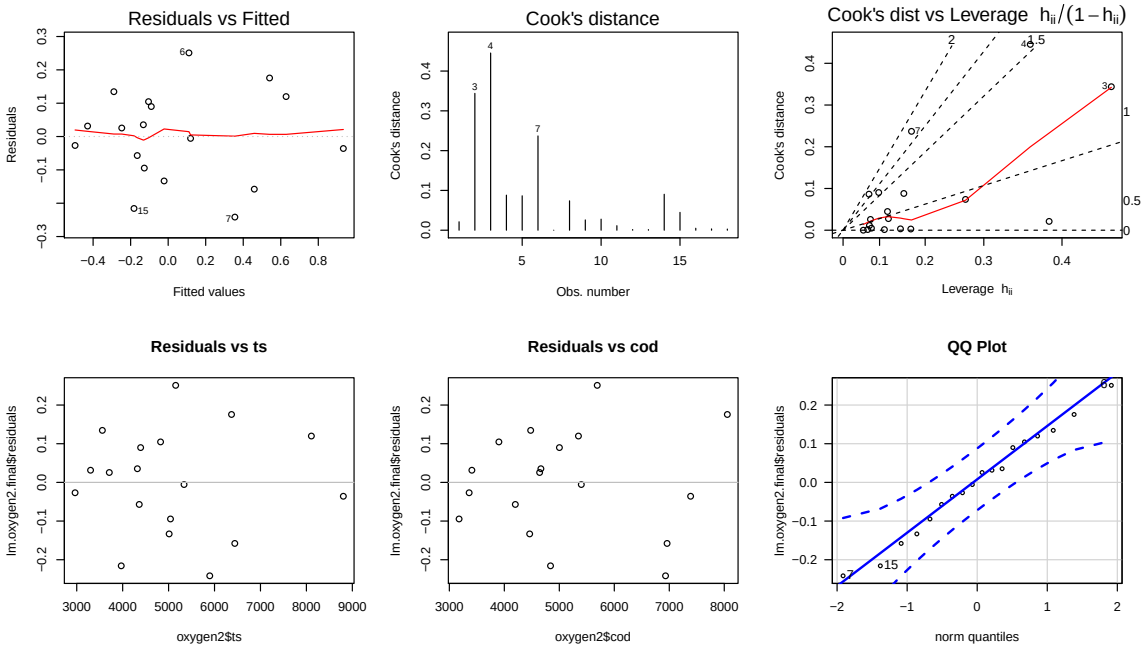
```
##
## Call:
## lm(formula = logup ~ ts + cod, data = oxygen2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.24157 -0.08517  0.01004  0.10102  0.25094
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.335e+00  1.338e-01  -9.976 5.16e-08 ***
## ts           1.852e-04  3.182e-05   5.820 3.38e-05 ***
## cod          8.638e-05  3.517e-05   2.456   0.0267 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1438 on 15 degrees of freedom
## Multiple R-squared:  0.8923,Adjusted R-squared:  0.878
## F-statistic: 62.15 on 2 and 15 DF,  p-value: 5.507e-08
```

```
# plot diagnostics
par(mfrow=c(2,3))
plot(lm.oxygen2.final, which = c(1,4,6))

plot(oxygen2$ts, lm.oxygen2.final$residuals, main="Residuals vs ts")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

plot(oxygen2$cod, lm.oxygen2.final$residuals, main="Residuals vs cod")
  # horizontal line at zero
  abline(h = 0, col = "gray75")

# Normality of Residuals
```

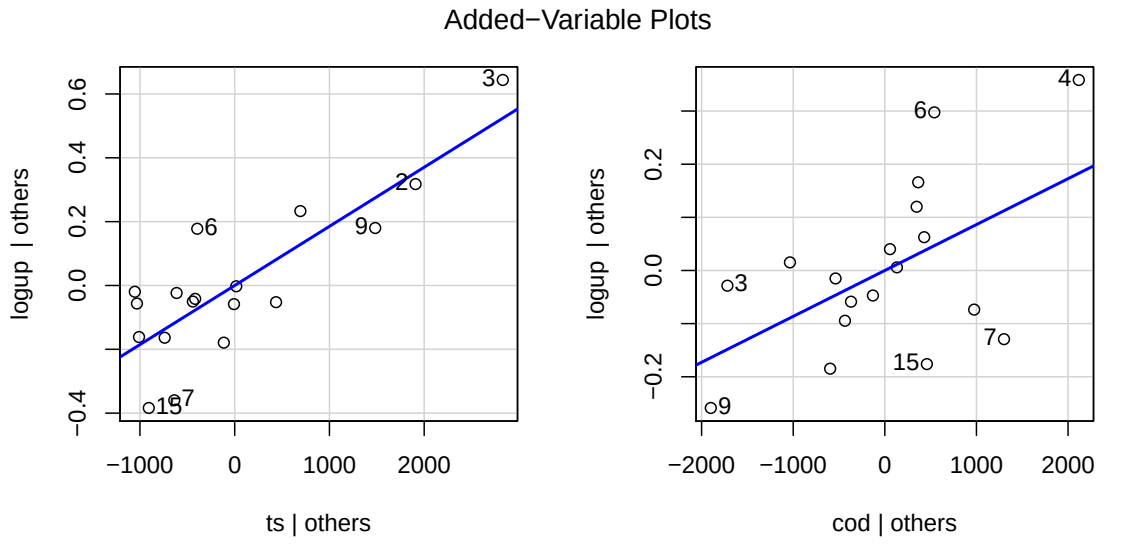
```
library(car)
qqPlot(lm.oxygen2.final$residuals, las = 1, id = list(n = 3), main="QQ Plot")
```

```
##  6  7 15
##  5  6 14
```

```
## residuals vs order of data
#plot(lm.oxygen2.final$residuals, main="Residuals vs Order of data")
#  # horizontal line at zero
#  abline(h = 0, col = "gray75")
```




```
library(car)
avPlots(lm.oxygen2.final, id = list(n = 3))
```



Let us recall that the researcher's primary goal was to identify important predictors of o2up. Regardless of whether we are inclined to include the end observations in the analysis or not, it is reasonable to conclude that ts and cod are useful for explaining the variation in $\log_{10}$(o2up). If these data were the final experiment, I might be inclined to eliminate the end observations and use the following equation to predict oxygen uptake:

$$\log_{10}(\text{o2up}) = -1.335302 + 0.000185\ \text{ts} + 0.000086\ \text{cod}.$$

Chapter 11

Logistic Regression

Logistic regression analysis is used for predicting the outcome of a categorical dependent variable based on one or more predictor variables. The probabilities describing the possible outcomes of a single trial are modeled, as a function of the explanatory (predictor) variables, using a logistic function. Logistic regression is frequently used to refer to the problem in which the dependent variable is binary — that is, the number of available categories is two — and problems with more than two categories are referred to as multinomial logistic regression or, if the multiple categories are ordered, as ordered logistic regression.

Logistic regression measures the relationship between a categorical dependent variable and usually (but not necessarily) one or more continuous independent variables, by converting the dependent variable to probability scores. As such it treats the same set of problems as does probit regression using similar techniques.

11.1 Generalized linear model variance and link families

The generalized linear model (GLM) is a flexible generalization of ordinary linear regression that allows for response variables that have other than a normal distribution. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value.

In R, the basic tool for fitting generalized linear models is the `glm()` function, which has the following general structure:

```
glm(formula, family, data, weights, subset, ...)
```

where "..." stands for additional options. The key parameter here is `family`, which is a simple way of specifying a choice of variance and link functions. Some choices

of family are listed in the table. As can be seen, each of the first five choices has an associated variance function (for binomial the binomial variance $\mu(1-\mu)$), and one or more choices of link functions (for binomial the logit, probit, or complementary log-log).

Family	Variance	Link
gaussian	gaussian	identity
binomial	binomial	logit, probit, or cloglog
poisson	poisson	log, identity, or sqrt
Gamma	Gamma	inverse, identity, or log
inverse.gaussian	inverse.gaussian	$1/\mu^2$
quasi	user-defined	user-defined

As long as you want the default link, all you have to specify is the family name. If you want an alternative link, you must add a link argument. For example to do probits you use:

```
glm(formula, family = binomial(link = probit))
```

The last family on the list, quasi, is there to allow fitting user-defined models by maximum quasi-likelihood.

The rest of this chapter concerns logistic regression with a binary response variable.

11.2 Example: Age of Menarche in Warsaw

The data[1] below are from a study conducted by Milicer and Szczotka on pre-teen and teenage girls in Warsaw, Poland in 1965. The subjects were classified into 25 age categories. The number of girls in each group (Total) and the number that reached menarche (Menarche) at the time of the study were recorded. The age for a group corresponds to the midpoint for the age interval.

```
#### Example: Menarche
# menarche dataset is available in MASS package
#   (remove previous instance if it exists, important if rerunning code)
rm(menarche)
## Warning in rm(menarche):  object 'menarche' not found
library(MASS)
# these frequencies look better in the table as integers
menarche$Total    <- as.integer(menarche$Total)
menarche$Menarche <- as.integer(menarche$Menarche)
str(menarche)
```

[1] Milicer, H. and Szczotka, F. (1966) Age at Menarche in Warsaw girls in 1965. Human Biology 38, 199–203.

```
## 'data.frame': 25 obs. of  3 variables:
##  $ Age     : num  9.21 10.21 10.58 10.83 11.08 ...
##  $ Total   : int  376 200 93 120 90 88 105 111 100 93 ...
##  $ Menarche: int  0 0 0 2 2 5 10 17 16 29 ...
# create estimated proportion of girls reaching menarche for each age group
menarche$p.hat <- menarche$Menarche / menarche$Total
```

	Age	Total	Menarche	p.hat
1	9.21	376	0	0.00
2	10.21	200	0	0.00
3	10.58	93	0	0.00
4	10.83	120	2	0.02
5	11.08	90	2	0.02
6	11.33	88	5	0.06
7	11.58	105	10	0.10
8	11.83	111	17	0.15
9	12.08	100	16	0.16
10	12.33	93	29	0.31
11	12.58	100	39	0.39
12	12.83	108	51	0.47
13	13.08	99	47	0.47
14	13.33	106	67	0.63
15	13.58	105	81	0.77
16	13.83	117	88	0.75
17	14.08	98	79	0.81
18	14.33	97	90	0.93
19	14.58	120	113	0.94
20	14.83	102	95	0.93
21	15.08	122	117	0.96
22	15.33	111	107	0.96
23	15.58	94	92	0.98
24	15.83	114	112	0.98
25	17.58	1049	1049	1.00

The researchers were curious about how the proportion of girls that reached menarche ($\hat{p}$ = Menarche/Total) varied with age. One could perform a test of homogeneity (Multinomial goodness-of-fit test) by arranging the data as a 2-by-25 contingency table with columns indexed by age and two rows: ROW1 = Menarche, and ROW2 = number that have not reached menarche = (Total − Menarche). A more powerful approach treats these as regression data, using the proportion of girls reaching menarche as the response and age as a predictor.

A plot of the observed proportion $\hat{p}$ of girls that have reached menarche shows that the proportion increases as age increases, but that the relationship is nonlinear. The observed proportions, which are bounded between zero and one, have a lazy S-shape (a **sigmoidal function**) when plotted against age. The change in the observed proportions for a given change in age is much smaller when the proportion is near 0 or 1 than when the proportion is near 1/2. This phenomenon is common with regression data where the response is a proportion.

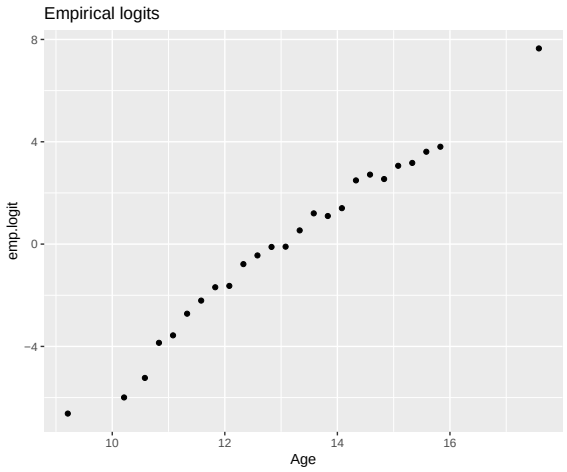
The trend is nonlinear so linear regression is inappropriate. A sensible alternative might be to transform the response or the predictor to achieve near linearity. A common transformation of response proportions following a sigmoidal curve is to the

logit scale $\hat{\mu} = \log_e\{\hat{p}/(1-\hat{p})\}$. This transformation is the basis for the **logistic regression model**. The natural logarithm (base e) is traditionally used in logistic regression.

The logit transformation is undefined when $\hat{p} = 0$ or $\hat{p} = 1$. To overcome this problem, researchers use the **empirical logits**, defined by $\log\{(\hat{p} + 0.5/n)/(1 - \hat{p} + 0.5/n)\}$, where n is the sample size or the number of observations on which $\hat{p}$ is based.

A plot of the empirical logits against age is roughly linear, which supports a logistic transformation for the response.

```
library(ggplot2)
p <- ggplot(menarche, aes(x = Age, y = p.hat))
p <- p + geom_point()
p <- p + labs(title = paste("Observed probability of girls reaching menarche,\n",
                            "Warsaw, Poland in 1965", sep=""))
print(p)

# emperical logits
menarche$emp.logit <- log((    menarche$p.hat + 0.5/menarche$Total) /
                          (1 - menarche$p.hat + 0.5/menarche$Total))
library(ggplot2)
p <- ggplot(menarche, aes(x = Age, y = emp.logit))
p <- p + geom_point()
p <- p + labs(title = "Empirical logits")
print(p)
```



11.3 Simple logistic regression model

The simple logistic regression model expresses the population proportion p of individuals with a given attribute (called the probability of success) as a function of a

single predictor variable X. The model assumes that p is related to X through

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X$$

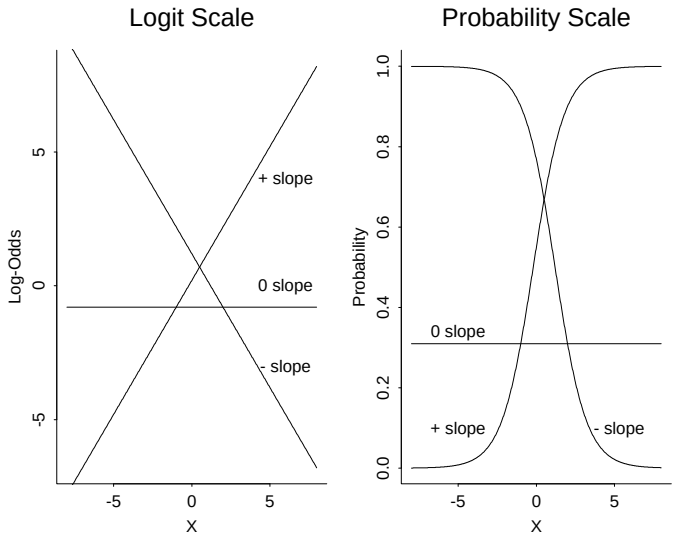
or, equivalently, as

$$p = \frac{\exp(\beta_0 + \beta_1 X)}{1 + \exp(\beta_0 + \beta_1 X)}.$$

The logistic regression model is a **binary response model**, where the response for each case falls into one of two exclusive and exhaustive categories, success (cases with the attribute of interest) and failure (cases without the attribute of interest).

The odds of success are $p/(1-p)$. For example, when $p = 1/2$ the odds of success are 1 (or 1 to 1). When $p = 0.9$ the odds of success are 9 (or 9 to 1). The logistic model assumes that the log-odds of success is linearly related to X. Graphs of the logistic model relating p to X are given below. The sign of the slope refers to the sign of β_1.

I should write $p = p(X)$ to emphasize that p is the proportion of all individuals with score X that have the attribute of interest. In the menarche data, $p = p(X)$ is the population proportion of girls at age X that have reached menarche.



The data in a logistic regression problem are often given in summarized or **aggregate** form:

X	n	y
X_1	n_1	y_1
X_2	n_2	y_2
$\vdots$	$\vdots$	$\vdots$
X_m	n_m	y_m

where y_i is the number of individuals with the attribute of interest among n_i randomly selected or representative individuals with predictor variable value X_i. For **raw data** on individual cases, $y_i = 1$ or 0, depending on whether the case at X_i is a success or failure, and the sample size column n is omitted with raw data.

For logistic regression, a plot of the sample proportions $\hat{p}_i = y_i/n_i$ against X_i should be roughly sigmoidal, and a plot of the empirical logits against X_i should be roughly linear. If not, then some other model is probably appropriate. I find the second plot easier to calibrate, but neither plot is very informative when the sample sizes are small, say 1 or 2. (Why?).

There are a variety of other binary response models that are used in practice. The **probit** regression model or the **complementary log-log** regression model might be appropriate when the logistic model does fit the data.

The following section describes the standard MLE strategy for estimating the logistic regression parameters.

11.3.1 Estimating Regression Parameters via LS of empirical logits

(This is a naive method; we will discuss a better way in the next section.)

There are two unknown population parameters in the logistic regression model

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X.$$

A simple way to estimate β_0 and β_1 is by least squares (LS), using the empirical logits as responses and the X_is as the predictor values.

Below we use standard regression to calculate the LS fit between the empirical logits and age.

```
lm.menarche.e.a <- lm(emp.logit ~ Age, data = menarche)
# LS coefficients
coef(lm.menarche.e.a)
```

```
## (Intercept)         Age
##  -22.027933    1.676395
```

The LS estimates for the menarche data are $b_0 = -22.03$ and $b_1 = 1.68$, which gives the fitted relationship

$$\log\left(\frac{\tilde{p}}{1-\tilde{p}}\right) = -22.03 + 1.68 \text{ Age}$$

or

$$\tilde{p} = \frac{\exp(-22.03 + 1.68 \text{ Age})}{1 + \exp(-22.03 + 1.68 \text{ Age})},$$

where $\tilde{p}$ is the predicted proportion (under the model) of girls having reached menarche at the given age. I used $\tilde{p}$ to identify a predicted probability, in contrast to $\hat{p}$ which is the observed proportion at a given age.

The power of the logistic model versus the contingency table analysis discussed earlier is that the model gives estimates for the population proportion reaching menarche at all ages within the observed age range. The observed proportions allow you to estimate only the population proportions at the observed ages.

11.3.2 Maximum Likelihood Estimation for Logistic Regression Model

There are better ways to the fit the logistic regression model than LS which assumes that the responses are normally distributed with constant variance. A deficiency of the LS fit to the logistic model is that the observed counts y_i have a **Binomial distribution** under random sampling. The Binomial distribution is a discrete probability model associated with counting the number of successes in a fixed size sample, and other equivalent experiments such as counting the number of heads in repeated flips of a coin. The distribution of the empirical logits depend on the y_is so they are not normal (but are approximately normal in large samples), and are extremely skewed when the sample sizes n_i are small. The response variability depends on the population proportions, and is not roughly constant when the observed proportions or the sample sizes vary appreciably across groups.

The differences in variability among the empirical logits can be accounted for using **weighted least squares** (WLS) when the sample sizes are large. An alternative approach called maximum likelihood uses the exact Binomial distribution of the responses y_i to generate optimal estimates of the regression coefficients. Software for maximum likelihood estimation is widely available, so LS and WLS methods are not really needed.

In **maximum likelihood estimation** (MLE), the regression coefficients are estimated iteratively by minimizing the **deviance** function (also called the likelihood ratio chi-squared statistic)

$$D = 2\sum_{i=1}^{m}\left\{y_i \log\left(\frac{y_i}{n_i p_i}\right) + (n_i - y_i)\log\left(\frac{n_i - y_i}{n_i - n_i p_i}\right)\right\}$$

over all possible values of β_0 and β_1, where the p_is satisfy the logistic model

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 X_i.$$

The ML method also gives standard errors and significance tests for the regression estimates.

The deviance is an analog of the residual sums of squares in linear regression. The choices for β_0 and β_1 that minimize the deviance are the parameter values that make the observed and fitted proportions as close together as possible in a "likelihood sense".

Suppose that b_0 and b_1 are the MLEs of β_0 and β_1. The deviance evaluated at the MLEs,

$$D = 2\sum_{i=1}^{m}\left\{y_i \log\left(\frac{y_i}{n_i\tilde{p}_i}\right) + (n_i - y_i)\log\left(\frac{n_i - y_i}{n_i - n_i\tilde{p}_i}\right)\right\},$$

where the fitted probabilities $\tilde{p}_i$ satisfy

$$\log\left(\frac{\tilde{p}_i}{1-\tilde{p}_i}\right) = b_0 + b_1 X_i,$$

is used to test the adequacy of the model. The deviance is small when the data fits the model, that is, when the observed and fitted proportions are close together. Large values of D occur when one or more of the observed and fitted proportions are far apart, which suggests that the model is inappropriate.

If the logistic model holds, then D **has a chi-squared distribution with** $m-r$ **degrees of freedom**, where m is the the number of groups and r (here 2) is the number of estimated regression parameters. A p-value for the deviance is given by the area under the chi-squared curve to the right of D. A small p-value indicates that the data does not fit the model.

Alternatively, the fit of the model can be evaluated using the chi-squared approximation to the Pearson X^2 statistic:

$$X^2 = \sum_{i=1}^{m}\left\{\frac{(y_i - n_i\tilde{p}_i)^2}{n_i\tilde{p}_i} + \frac{((n_i - y_i) - n_i(1-\tilde{p}_i))^2}{n_i(1-\tilde{p}_i)}\right\} = \sum_{i=1}^{m}\frac{(y_i - n_i\tilde{p}_i)^2}{n_i\tilde{p}_i(1-\tilde{p}_i)}.$$

11.3.3 Fitting the Logistic Model by Maximum Likelihood, Menarche

```
# For our summarized data (with frequencies and totals for each age)
# The left-hand side of our formula binds two columns together with cbind():
#   the columns are the number of "successes" and "failures".
# For logistic regression with logit link we specify family = binomial,
#   where logit is the default link function for the binomial family.
glm.m.a <- glm(cbind(Menarche, Total - Menarche) ~ Age, family = binomial, menarche)
```

The `glm()` statement creates an object which we can use to create the fitted probabilities and 95% CIs for the population proportions at the ages in **menarche**. The fitted probabilities and the limits are stored in columns labeled `fitted.values`, `fit.lower`, and `fit.upper`, respectively.

```
# put the fitted values in the data.frame
menarche$fitted.values <- glm.m.a$fitted.values
pred <- predict(glm.m.a, data.frame(Age = menarche$Age), type = "link", se.fit = TRUE)
menarche$fit <- pred$fit
menarche$se.fit <- pred$se.fit
# CI for fitted values
menarche <- within(menarche, {
  fit.lower = exp(fit - 1.96 * se.fit) / (1 + exp(fit - 1.96 * se.fit))
  fit.upper = exp(fit + 1.96 * se.fit) / (1 + exp(fit + 1.96 * se.fit))
  })
#round(menarche, 3)
```

This printed summary information is easily interpreted. For example, the estimated population proportion of girls aged 15.08 (more precisely, among girls in the age interval with midpoint 15.08) that have reached menarche is 0.967. You are 95% confident that the population proportion is between 0.958 and 0.975. A variety of other summaries and diagnostics can be produced.

	Age	Total	Menarche	p.hat	emp.logit	fitted.values	fit	se.fit	fit.upper	fit.lower
21	15.08	122.00	117.00	0.96	3.06	0.97	3.38	0.14	0.97	0.96

	Age	Total	Menarche	p.hat	emp.logit	fitted.values	fit	se.fit	fit.upper	fit.lower
1	9.21	376.00	0.00	0.00	−6.62	0.00	−6.20	0.23	0.00	0.00
2	10.21	200.00	0.00	0.00	−5.99	0.01	−4.56	0.18	0.01	0.01
3	10.58	93.00	0.00	0.00	−5.23	0.02	−3.96	0.16	0.02	0.01
4	10.83	120.00	2.00	0.02	−3.86	0.03	−3.55	0.14	0.04	0.02
5	11.08	90.00	2.00	0.02	−3.57	0.04	−3.14	0.13	0.05	0.03
6	11.33	88.00	5.00	0.06	−2.72	0.06	−2.74	0.12	0.08	0.05
7	11.58	105.00	10.00	0.10	−2.21	0.09	−2.33	0.11	0.11	0.07
8	11.83	111.00	17.00	0.15	−1.69	0.13	−1.92	0.10	0.15	0.11
9	12.08	100.00	16.00	0.16	−1.63	0.18	−1.51	0.08	0.21	0.16
10	12.33	93.00	29.00	0.31	−0.78	0.25	−1.10	0.07	0.28	0.22
11	12.58	100.00	39.00	0.39	−0.44	0.33	−0.70	0.07	0.36	0.30
12	12.83	108.00	51.00	0.47	−0.11	0.43	−0.29	0.06	0.46	0.40
13	13.08	99.00	47.00	0.47	−0.10	0.53	0.12	0.06	0.56	0.50
14	13.33	106.00	67.00	0.63	0.54	0.63	0.53	0.07	0.66	0.60
15	13.58	105.00	81.00	0.77	1.20	0.72	0.94	0.07	0.75	0.69
16	13.83	117.00	88.00	0.75	1.10	0.79	1.34	0.08	0.82	0.77
17	14.08	98.00	79.00	0.81	1.40	0.85	1.75	0.09	0.87	0.83
18	14.33	97.00	90.00	0.93	2.49	0.90	2.16	0.10	0.91	0.88
19	14.58	120.00	113.00	0.94	2.72	0.93	2.57	0.11	0.94	0.91
20	14.83	102.00	95.00	0.93	2.54	0.95	2.98	0.12	0.96	0.94
21	15.08	122.00	117.00	0.96	3.06	0.97	3.38	0.14	0.97	0.96
22	15.33	111.00	107.00	0.96	3.17	0.98	3.79	0.15	0.98	0.97
23	15.58	94.00	92.00	0.98	3.61	0.98	4.20	0.16	0.99	0.98
24	15.83	114.00	112.00	0.98	3.81	0.99	4.61	0.18	0.99	0.99
25	17.58	1049.00	1049.00	1.00	7.65	1.00	7.46	0.28	1.00	1.00

The summary table gives MLEs and standard errors for the regression parameters. The z-value column is the parameter estimate divided by its standard error. The p-values are used to test whether the corresponding parameters of the logistic model are zero.

```
summary(glm.m.a)
##
## Call:
## glm(formula = cbind(Menarche, Total - Menarche) ~ Age, family = binomial,
##     data = menarche)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0363  -0.9953  -0.4900   0.7780   1.3675
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -21.22639    0.77068  -27.54   <2e-16 ***
## Age           1.63197    0.05895   27.68   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3693.884  on 24  degrees of freedom
## Residual deviance:   26.703  on 23  degrees of freedom
## AIC: 114.76
##
## Number of Fisher Scoring iterations: 4
```

If the model is correct and when sample sizes are large, the residual deviance D has an approximate chi-square distribution,

$$\text{residual } D \quad = \quad \chi^2_{\text{residual df}}.$$

If D is too large, or the p-value is too small, then the model does not capture all the features in the data.

The deviance statistic is $D = 26.70$ on $25 - 2 = 23$ df. The large p-value for D suggests no gross deficiencies with the logistic model. The observed and fitted proportions (`p.hat` and `fitted.values` in the output table above are reasonably close at each observed age. Also, `emp.logit` and `fit` are close. This is consistent with D being fairly small. The data fits the logistic regression model reasonably well.

```
# Test residual deviance for lack-of-fit (if > 0.10, little-to-no lack-of-fit)
glm.m.a$deviance
## [1] 26.70345
glm.m.a$df.residual
## [1] 23
dev.p.val <- 1 - pchisq(glm.m.a$deviance, glm.m.a$df.residual)
dev.p.val
## [1] 0.2687953
```

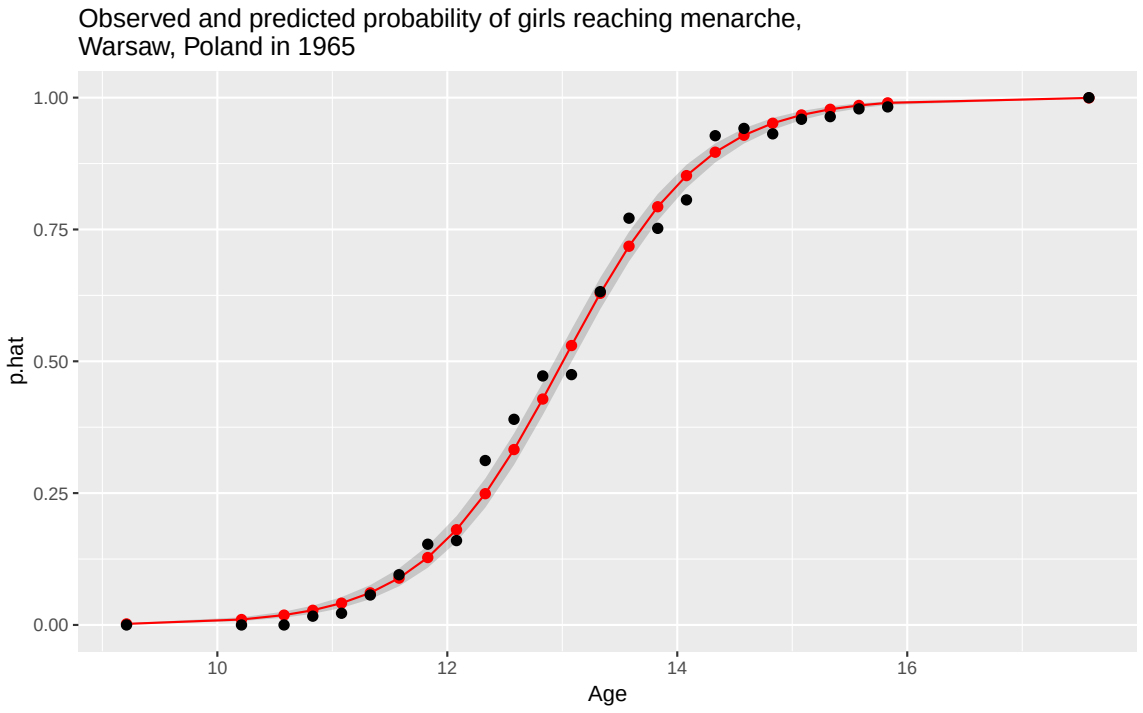
The MLEs $b_0 = -21.23$ and $b_1 = 1.63$ for the intercept and slope are close to the LS estimates of $b_{LS0} = -22.03$ and $b_{LS1} = 1.68$, respectively from page 680. The two estimation methods give similar predicted probabilities here. The MLE of the predicted probabilities satisfy

$$\log\left(\frac{\tilde{p}}{1-\tilde{p}}\right) = -21.23 + 1.63 \text{ Age}$$

or

$$\tilde{p} = \frac{\exp(-21.23 + 1.63 \text{ Age})}{1 + \exp(-21.23 + 1.63 \text{ Age})}.$$

```
library(ggplot2)
p <- ggplot(menarche, aes(x = Age, y = p.hat))
# predicted curve and point-wise 95% CI
p <- p + geom_ribbon(aes(x = Age, ymin = fit.lower, ymax = fit.upper), alpha = 0.2)
p <- p + geom_line(aes(x = Age, y = fitted.values), color = "red")
# fitted values
p <- p + geom_point(aes(y = fitted.values), color = "red", size=2)
# observed values
p <- p + geom_point(size=2)
p <- p + labs(title = paste("Observed and predicted probability of girls reaching menarche,\n",
                            "Warsaw, Poland in 1965", sep=""))
print(p)
```



If the model holds, then a slope of $\beta_1 = 0$ implies that p does not depend on AGE, i.e., the proportion of girls that have reached menarche is identical across age groups.

The Wald p-value for the slope is < 0.0001, which leads to rejecting $H_0 : \beta_1 = 0$ at any of the usual test levels. The proportion of girls that have reached menarche is not constant across age groups. Again, the power of the model is that it gives you a simple way to quantify the effect of age on the proportion reaching menarche. This is more appealing than testing homogeneity across age groups followed by multiple comparisons.

Wald tests can be performed to test the global null hypothesis, that all non-intercept βs are equal to zero. This is the logistic regression analog of the overall model F-test in ANOVA and regression. The only predictor is AGE, so the implied test is that the slope of the regression line is zero. The Wald test statistic and p-value reported here are identical to the Wald test and p-value for the AGE effect given in the parameter estimates table. The Wald test can also be used to test specific contrasts between parameters.

```
# Testing Global Null Hypothesis
library(aod)
coef(glm.m.a)
```

```
## (Intercept)         Age
##  -21.226395    1.631968
```

```
# specify which coefficients to test = 0  (Terms = 2:4 would be terms 2, 3, and 4)
wald.test(b = coef(glm.m.a), Sigma = vcov(glm.m.a), Terms = 2:2)
```

```
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 766.3, df = 1, P(> X2) = 0.0
```

11.4 Example: Leukemia white blood cell types

Feigl and Zelen[2] reported the survival time in weeks and the white cell blood count (WBC) at time of diagnosis for 33 patients who eventually died of acute leukemia. Each person was classified as AG+ or AG−, indicating the presence or absence of a certain morphological characteristic in the white cells. Four variables are given in the data set: WBC, a binary factor or **indicator variable** AG (1 for AG+, 0 for AG−), NTOTAL (the number of patients with the given combination of AG and WBC),

[2]Feigl, P., Zelen, M. (1965) Estimation of exponential survival probabilities with concomitant information. Biometrics 21, 826–838. Survival times are given for 33 patients who died from acute myelogenous leukaemia. Also measured was the patient's white blood cell count at the time of diagnosis. The patients were also factored into 2 groups according to the presence or absence of a morphologic characteristic of white blood cells. Patients termed AG positive were identified by the presence of Auer rods and/or significant granulation of the leukaemic cells in the bone marrow at the time of diagnosis.

and NRES (the number of NTOTAL that survived at least one year from the time of diagnosis).

The researchers are interested in modelling the probability p of surviving at least one year as a function of WBC and AG. They believe that WBC should be transformed to a log scale, given the skewness in the WBC values.

```
#### Example: Leukemia
## Leukemia white blood cell types example
# ntotal = number of patients with IAG and WBC combination
# nres   = number surviving at least one year
# ag     = 1 for AG+, 0 for AG-
# wbc    = white cell blood count
# lwbc   = log white cell blood count
# p.hat  = Emperical Probability
leuk <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch11_leuk.dat"
                   , header = TRUE)
leuk$ag    <- factor(leuk$ag)
leuk$lwbc  <- log(leuk$wbc)
leuk$p.hat <- leuk$nres / leuk$ntotal
str(leuk)
```

```
## 'data.frame': 30 obs. of  6 variables:
##  $ ntotal: int  1 1 1 1 1 1 1 1 3 1 ...
##  $ nres  : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ ag    : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 1 ...
##  $ wbc   : int  75 230 260 430 700 940 1000 1050 10000 300 ...
##  $ lwbc  : num  4.32 5.44 5.56 6.06 6.55 ...
##  $ p.hat : num  1 1 1 1 1 ...
```

	ntotal	nres	ag	wbc	lwbc	p.hat
1	1	1	1	75	4.32	1.00
2	1	1	1	230	5.44	1.00
3	1	1	1	260	5.56	1.00
4	1	1	1	430	6.06	1.00
5	1	1	1	700	6.55	1.00
6	1	1	1	940	6.85	1.00
7	1	1	1	1000	6.91	1.00
8	1	1	1	1050	6.96	1.00
9	3	1	1	10000	9.21	0.33
10	1	1	0	300	5.70	1.00
11	1	1	0	440	6.09	1.00
12	1	0	1	540	6.29	0.00
13	1	0	1	600	6.40	0.00
14	1	0	1	1700	7.44	0.00
15	1	0	1	3200	8.07	0.00
16	1	0	1	3500	8.16	0.00
17	1	0	1	5200	8.56	0.00
18	1	0	0	150	5.01	0.00
19	1	0	0	400	5.99	0.00
20	1	0	0	530	6.27	0.00
21	1	0	0	900	6.80	0.00
22	1	0	0	1000	6.91	0.00
23	1	0	0	1900	7.55	0.00
24	1	0	0	2100	7.65	0.00
25	1	0	0	2600	7.86	0.00
26	1	0	0	2700	7.90	0.00
27	1	0	0	2800	7.94	0.00
28	1	0	0	3100	8.04	0.00
29	1	0	0	7900	8.97	0.00
30	2	0	0	10000	9.21	0.00

As an initial step in the analysis, consider the following model:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 \text{ LWBC} + \beta_2 \text{ AG},$$

where LWBC = log(WBC). The model is best understood by separating the AG+ and AG− cases. For AG− individuals, AG=0 so the model reduces to

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 \text{ LWBC} + \beta_2 * 0 = \beta_0 + \beta_1 \text{ LWBC}.$$

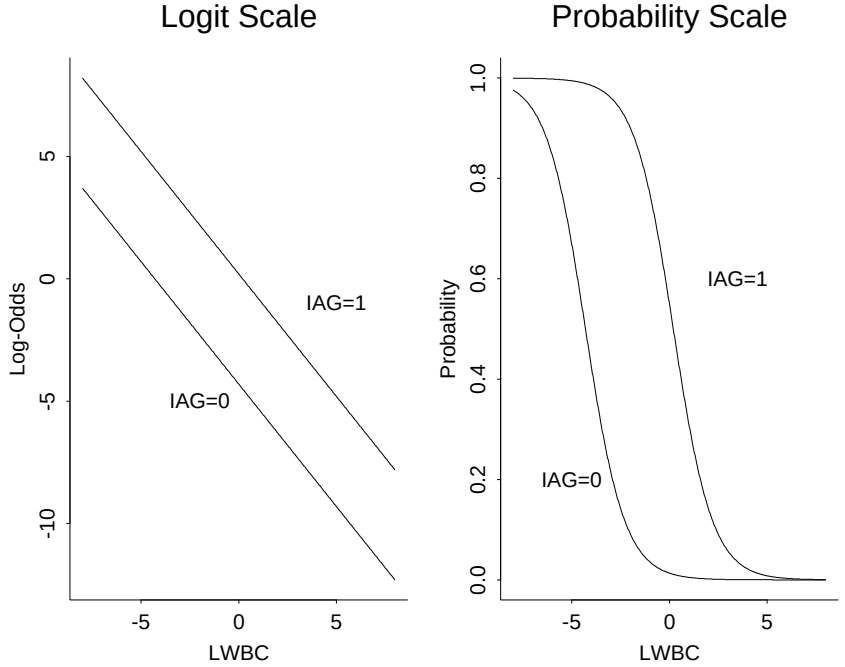
For AG+ individuals, AG=1 and the model implies

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 \text{ LWBC} + \beta_2 * 1 = (\beta_0 + \beta_2) + \beta_1 \text{ LWBC}.$$

The model without AG (i.e., $\beta_2 = 0$) is a simple logistic model where the log-odds of surviving one year is linearly related to LWBC, and is independent of AG. The reduced model with $\beta_2 = 0$ implies that there is no effect of the AG level on the survival probability once LWBC has been taken into account.

Including the **binary predictor** AG in the model implies that there is a linear relationship between the log-odds of surviving one year and LWBC, with a constant slope for the two AG levels. This model includes an effect for the AG morphological factor, but more general models are possible. A natural extension would be to include a product or interaction effect, a point that I will return to momentarily.

The parameters are easily interpreted: β_0 and $\beta_0 + \beta_2$ are intercepts for the population logistic regression lines for AG− and AG+, respectively. The lines have a common slope, β_1. The β_2 coefficient for the AG indicator is the difference between intercepts for the AG+ and AG− regression lines. A picture of the assumed relationship is given below for $\beta_1 < 0$. The population regression lines are parallel on the logit scale only, but the order between AG groups is preserved on the probability scale.



Before looking at output for the equal slopes model, note that the data set has 30 distinct AG and LWBC combinations, or 30 "groups" or samples. Only two samples

have more than 1 observation. The majority of the observed proportions surviving at least one year (number surviving $\geq$ 1 year/group sample size) are 0 (i.e., 0/1) or 1 (i.e., 1/1). This sparseness of the data makes it difficult to graphically assess the suitability of the logistic model (because the estimated proportions are almost all 0 or 1). Although significance tests on the regression coefficients do not require large group sizes, the chi-squared approximation to the deviance statistic is suspect in sparse data settings. With small group sizes as we have here, most researchers would not interpret the p-value for D literally. Instead, they would use the p-values to informally check the fit of the model. Diagnostics would be used to highlight problems with the model.

```
glm.i.l <- glm(cbind(nres, ntotal - nres) ~ ag + lwbc, family = binomial, leuk)

# Test residual deviance for lack-of-fit (if > 0.10, little-to-no lack-of-fit)
dev.p.val <- 1 - pchisq(glm.i.l$deviance, glm.i.l$df.residual)
dev.p.val
## [1] 0.6842804
```

The large p-value for D indicates that there are no gross deficiencies with the model. Recall that the Testing Global Null Hypothesis gives p-values for testing the hypothesis that the regression coefficients are zero for each predictor in the model. The two predictors are LWBC and AG, so the small p-values indicate that LWBC or AG, or both, are important predictors of survival. The p-values in the estimates table suggest that LWBC and AG are both important. If either predictor was insignificant, I would consider refitting the model omitting the least significant effect, as in regression.

```
# Testing Global Null Hypothesis
library(aod)
coef(glm.i.l)
## (Intercept)         ag1        lwbc
##    5.543349    2.519562   -1.108759
# specify which coefficients to test = 0  (Terms = 2:3 is for terms 2 and 3)
wald.test(b = coef(glm.i.l), Sigma = vcov(glm.i.l), Terms = 2:3)
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 8.2, df = 2, P(> X2) = 0.017
```

Given that the model fits reasonably well, a test of $H_0 : \beta_2 = 0$ might be a primary interest here. This checks whether the regression lines are identical for the two AG levels, which is a test for whether AG affects the survival probability, after taking LWBC into account. This test is rejected at any of the usual significance levels, suggesting that the AG level affects the survival probability (assuming a very specific model).

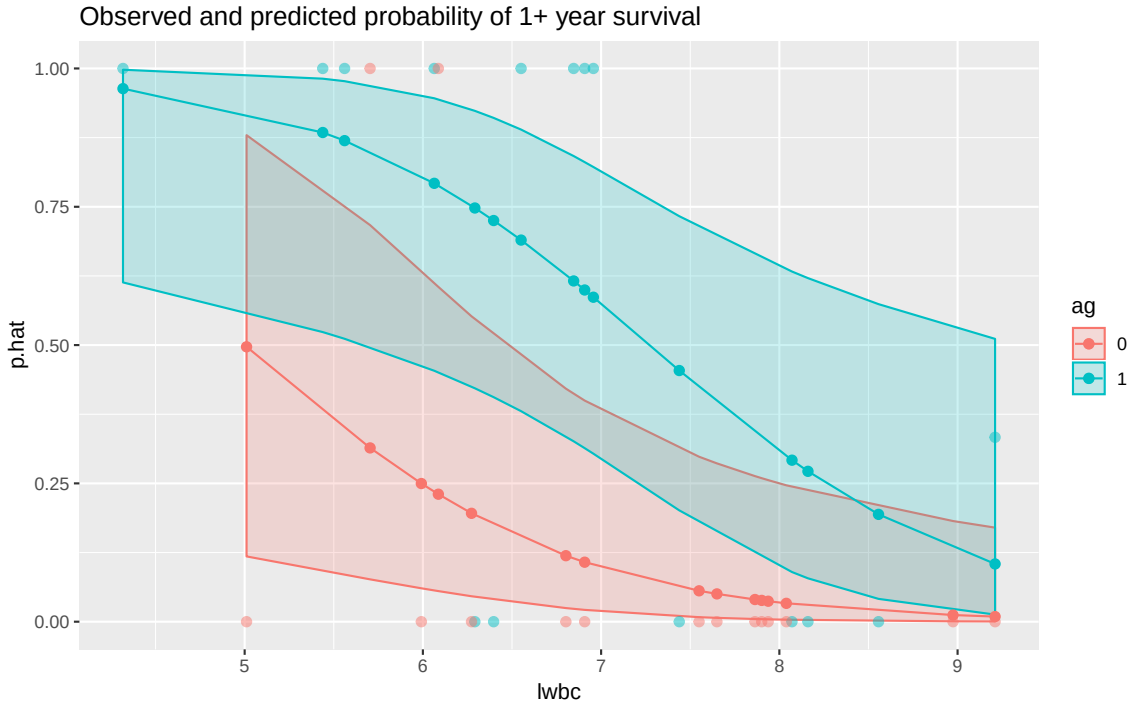
```
summary(glm.i.l)
```

```
##
## Call:
## glm(formula = cbind(nres, ntotal - nres) ~ ag + lwbc, family = binomial,
##     data = leuk)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6599  -0.6595  -0.2776   0.6438   1.7131
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   5.5433     3.0224   1.834   0.0666 .
## ag1           2.5196     1.0907   2.310   0.0209 *
## lwbc         -1.1088     0.4609  -2.405   0.0162 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 38.191  on 29  degrees of freedom
## Residual deviance: 23.014  on 27  degrees of freedom
## AIC: 30.635
##
## Number of Fisher Scoring iterations: 5
```

A plot of the predicted survival probabilities as a function of LWBC, using AG as the plotting symbol, indicates that the probability of surviving at least one year from the time of diagnosis is a decreasing function of LWBC. For a given LWBC the survival probability is greater for AG+ patients than for AG− patients. This tendency is consistent with the observed proportions, which show little information about the exact form of the trend.

```
# put the fitted values in the data.frame
leuk$fitted.values <- glm.i.l$fitted.values
pred <- predict(glm.i.l, data.frame(lwbc = leuk$lwbc, ag = leuk$ag), type = "link"
              , se.fit = TRUE)
leuk$fit <- pred$fit
leuk$se.fit <- pred$se.fit
# CI for fitted values
leuk <- within(leuk, {
  fit.lower = exp(fit - 1.96 * se.fit) / (1 + exp(fit - 1.96 * se.fit))
  fit.upper = exp(fit + 1.96 * se.fit) / (1 + exp(fit + 1.96 * se.fit))
  })
#round(leuk, 3)

library(ggplot2)
p <- ggplot(leuk, aes(x = lwbc, y = p.hat, colour = ag, fill = ag))
```

```
# predicted curve and point-wise 95% CI
p <- p + geom_ribbon(aes(x = lwbc, ymin = fit.lower, ymax = fit.upper), alpha = 0.2)
p <- p + geom_line(aes(x = lwbc, y = fitted.values))
# fitted values
p <- p + geom_point(aes(y = fitted.values), size=2)
# observed values
p <- p + geom_point(size = 2, alpha = 0.5)
p <- p + labs(title = "Observed and predicted probability of 1+ year survival")
print(p)
```



	ntotal	nres	ag	wbc	lwbc	p.hat	fitted.values	fit	se.fit	fit.upper	fit.lower
1	1	1	1	75	4.32	1.00	0.96	3.28	1.44	1.00	0.61
2	1	1	1	230	5.44	1.00	0.88	2.03	0.99	0.98	0.52
3	1	1	1	260	5.56	1.00	0.87	1.90	0.94	0.98	0.51
4	1	1	1	430	6.06	1.00	0.79	1.34	0.78	0.95	0.45
5	1	1	1	700	6.55	1.00	0.69	0.80	0.66	0.89	0.38
6	1	1	1	940	6.85	1.00	0.62	0.47	0.61	0.84	0.33
7	1	1	1	1000	6.91	1.00	0.60	0.40	0.61	0.83	0.31
8	1	1	1	1050	6.96	1.00	0.59	0.35	0.60	0.82	0.30
9	3	1	1	10000	9.21	0.33	0.10	−2.15	1.12	0.51	0.01
10	1	1	0	300	5.70	1.00	0.31	−0.78	0.87	0.72	0.08
11	1	1	0	440	6.09	1.00	0.23	−1.21	0.83	0.61	0.06
12	1	0	1	540	6.29	0.00	0.75	1.09	0.72	0.92	0.42
13	1	0	1	600	6.40	0.00	0.73	0.97	0.69	0.91	0.41
14	1	0	1	1700	7.44	0.00	0.45	−0.18	0.61	0.73	0.20
15	1	0	1	3200	8.07	0.00	0.29	−0.89	0.73	0.63	0.09
16	1	0	1	3500	8.16	0.00	0.27	−0.99	0.75	0.62	0.08
17	1	0	1	5200	8.56	0.00	0.19	−1.42	0.88	0.57	0.04
18	1	0	0	150	5.01	0.00	0.50	−0.01	1.02	0.88	0.12
19	1	0	0	400	5.99	0.00	0.25	−1.10	0.84	0.63	0.06
20	1	0	0	530	6.27	0.00	0.20	−1.41	0.83	0.55	0.05
21	1	0	0	900	6.80	0.00	0.12	−2.00	0.86	0.42	0.02
22	1	0	0	1000	6.91	0.00	0.11	−2.12	0.87	0.40	0.02
23	1	0	0	1900	7.55	0.00	0.06	−2.83	1.01	0.30	0.01
24	1	0	0	2100	7.65	0.00	0.05	−2.94	1.03	0.29	0.01
25	1	0	0	2600	7.86	0.00	0.04	−3.18	1.09	0.26	0.00
26	1	0	0	2700	7.90	0.00	0.04	−3.22	1.11	0.26	0.00
27	1	0	0	2800	7.94	0.00	0.04	−3.26	1.12	0.26	0.00
28	1	0	0	3100	8.04	0.00	0.03	−3.37	1.15	0.25	0.00
29	1	0	0	7900	8.97	0.00	0.01	−4.41	1.48	0.18	0.00
30	2	0	0	10000	9.21	0.00	0.01	−4.67	1.57	0.17	0.00

The estimated survival probabilities satisfy

$$\log\left(\frac{\tilde{p}}{1-\tilde{p}}\right) = 5.54 - 1.11 \text{ LWBC} + 2.52 \text{ AG}.$$

For AG− individuals with AG=0, this reduces to

$$\log\left(\frac{\tilde{p}}{1-\tilde{p}}\right) = 5.54 - 1.11 \text{ LWBC},$$

or equivalently,

$$\tilde{p} = \frac{\exp(5.54 - 1.11 \text{ LWBC})}{1 + \exp(5.54 - 1.11 \text{ LWBC})}.$$

For AG+ individuals with AG=1,

$$\log\left(\frac{\tilde{p}}{1-\tilde{p}}\right) = 5.54 - 1.11 \text{ LWBC} + 2.52(1) = 8.06 - 1.11 \text{ LWBC},$$

or

$$\tilde{p} = \frac{\exp(8.06 - 1.11 \text{ LWBC})}{1 + \exp(8.06 - 1.11 \text{ LWBC})}.$$

Using the **logit scale**, the difference between AG+ and AG− individuals in the estimated log-odds of surviving at least one year, at a fixed but arbitrary LWBC, is the estimated AG regression coefficient

$$(8.06 - 1.11 \text{ LWBC}) - (5.54 - 1.11 \text{ LWBC}) = 2.52.$$

Using properties of exponential functions, the odds that an AG+ patient lives at least one year is $\exp(2.52) = 12.42$ times larger than the odds that an AG− patient lives at least one year, regardless of LWBC.

This summary, and a CI for the AG odds ratio, is given in the **Odds Ratio** table. Similarly, the estimated odds ratio of 0.33 for LWBC implies that the odds of surviving at least one year is reduced by a factor of 3 for each unit increase of LWBC.

We can use the `confint()` function to obtain confidence intervals for the coefficient estimates. Note that for logistic models, confidence intervals are based on the profiled log-likelihood function.

```
## CIs using profiled log-likelihood
confint(glm.i.l)
```

```
## Waiting for profiling to be done...
##                  2.5 %     97.5 %
## (Intercept)  0.1596372 12.4524409
## ag1          0.5993391  5.0149271
## lwbc        -2.2072275 -0.3319512
```

We can also get CIs based on just the standard errors by using the default method.

```
## CIs using standard errors
confint.default(glm.i.l)
```

```
##                  2.5 %    97.5 %
## (Intercept) -0.3804137 11.467112
## ag1          0.3818885  4.657236
## lwbc        -2.0121879 -0.205330
```

You can also exponentiate the coefficients and confidence interval bounds and interpret them as odds-ratios.

```
## coefficients and 95% CI
cbind(OR = coef(glm.i.l), confint(glm.i.l))
```

```
## Waiting for profiling to be done...
##                    OR      2.5 %     97.5 %
## (Intercept)  5.543349  0.1596372 12.4524409
## ag1          2.519562  0.5993391  5.0149271
## lwbc        -1.108759 -2.2072275 -0.3319512
```

```
## odds ratios and 95% CI
exp(cbind(OR = coef(glm.i.l), confint(glm.i.l)))
```

```
## Waiting for profiling to be done...
##                        OR     2.5 %       97.5 %
## (Intercept) 255.5323676 1.1730851 2.558741e+05
```

```
## ag1          12.4231582 1.8209149 1.506452e+02
## lwbc          0.3299682 0.1100052 7.175224e-01
```

Although the equal slopes model appears to fit well, a more general model might fit better. A natural generalization here would be to add an **interaction**, or product term, AG × LWBC to the model. The logistic model with an AG effect and the AG × LWBC interaction is equivalent to fitting separate logistic regression lines to the two AG groups. This interaction model provides an easy way to test whether the slopes are equal across AG levels. I will note that the interaction term is not needed here.

Interpretting odds ratios in logistic regression 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) $= 0.25/4 = 0.0625$.

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, $p = 7/10 = 0.7$ 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.

Leukemia example In the example above, the OR of surviving at least one year increases 12.43 times for AG+ vs AG−, and increases 0.33 times (that's a decrease) for every unit increase in lwbc.

Example: Mortality of confused flour beetles This example illustrates a quadratic logistic model.

[3]Borrowed graciously from UCLA Academic Technology Services at `http://www.ats.ucla.edu/stat/sas/faq/oratio.htm`

The aim of an experiment originally reported by Strand (1930) and quoted by Bliss (1935) was to assess the response of the confused flour beetle, *Tribolium confusum*, to gaseous carbon disulphide (CS_2). In the experiment, prescribed volumes of liquid carbon disulphide were added to flasks in which a tubular cloth cage containing a batch of about thirty beetles was suspended. Duplicate batches of beetles were used for each concentration of CS_2. At the end of a five-hour period, the proportion killed was recorded and the actual concentration of gaseous CS_2 in the flask, measured in mg/l, was determined by a volumetric analysis. The mortality data are given in the table below.

```
#### Example: Beetles
## Beetles data set
# conc = CS2 concentration
# y    = number of beetles killed
# n    = number of beetles exposed
# rep  = Replicate number (1 or 2)
beetles <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch11_beetles.dat", header = TRUE)
beetles$rep   <- factor(beetles$rep)
```

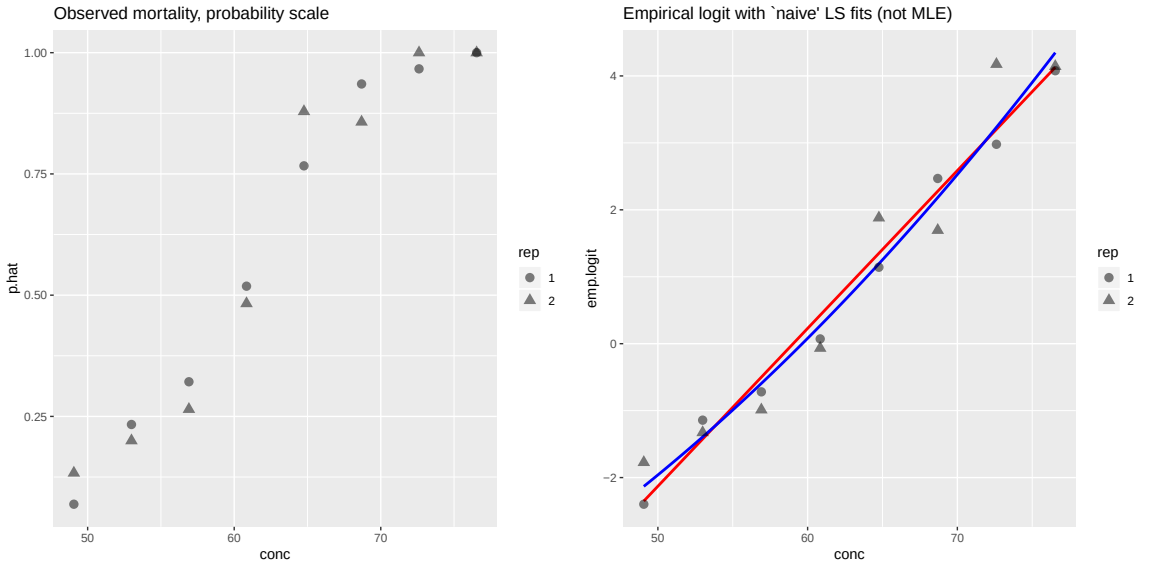
	conc	y	n	rep		conc	y	n	rep
1	49.06	2	29	1	9	49.06	4	30	2
2	52.99	7	30	1	10	52.99	6	30	2
3	56.91	9	28	1	11	56.91	9	34	2
4	60.84	14	27	1	12	60.84	14	29	2
5	64.76	23	30	1	13	64.76	29	33	2
6	68.69	29	31	1	14	68.69	24	28	2
7	72.61	29	30	1	15	72.61	32	32	2
8	76.54	29	29	1	16	76.54	31	31	2

```
beetles$conc2 <- beetles$conc^2 # for quadratic term (making coding a little easier)
beetles$p.hat <- beetles$y / beetles$n # observed proportion of successes
# emperical logits
beetles$emp.logit <- log((    beetles$p.hat + 0.5/beetles$n) /
                         (1 - beetles$p.hat + 0.5/beetles$n))
#str(beetles)
```

Plot the observed probability of mortality and the empirical logits with linear and quadratic LS fits (which are not the same as the logistic MLE fits).

```
library(ggplot2)
p <- ggplot(beetles, aes(x = conc, y = p.hat, shape = rep))
# observed values
p <- p + geom_point(color = "black", size = 3, alpha = 0.5)
p <- p + labs(title = "Observed mortality, probability scale")
print(p)

library(ggplot2)
p <- ggplot(beetles, aes(x = conc, y = emp.logit))
p <- p + geom_smooth(method = "lm", colour = "red", se = FALSE)
```

```
p <- p + geom_smooth(method = "lm", formula = y ~ poly(x, 2), colour = "blue", se = FALSE)
# observed values
p <- p + geom_point(aes(shape = rep), color = "black", size = 3, alpha = 0.5)
p <- p + labs(title = "Empirical logit with `naive' LS fits (not MLE)")
print(p)
```



In a number of articles that refer to these data, the responses from the first two concentrations are omitted because of apparent non-linearity. Bliss himself remarks that

> . . . in comparison with the remaining observations, the two lowest concentrations gave an exceptionally high kill. Over the remaining concentrations, the plotted values seemed to form a moderately straight line, so that the data were handled as two separate sets, only the results at 56.91 mg of CS_2 per litre being included in both sets.

However, there does not appear to be any biological motivation for this and so here they are retained in the data set.

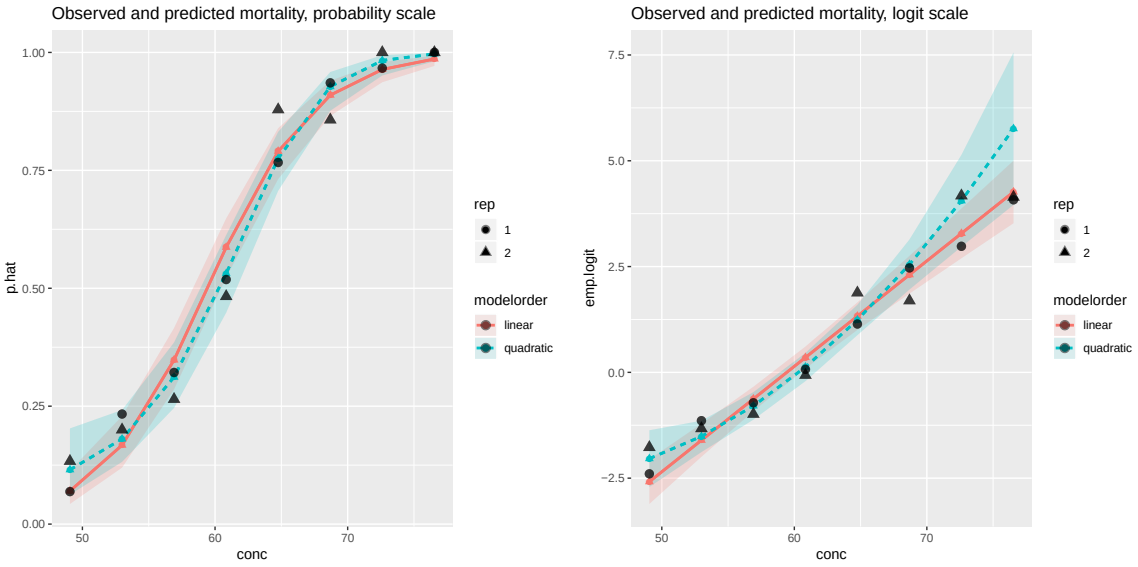
Combining the data from the two replicates and plotting the empirical logit of the observed proportions against concentration gives a relationship that is better fit by a quadratic than a linear relationship,

$$\log\left(\frac{p}{1-p}\right) \quad = \quad \beta_0 + \beta_1 X + \beta_2 X^2.$$

The right plot below shows the linear and quadratic model fits to the observed values with point-wise 95% confidence bands on the logit scale, and on the left is the same on the proportion scale.

```r
# fit logistic regression to create lines on plots below
# linear
glm.beetles1 <- glm(cbind(y, n - y) ~ conc, family = binomial, beetles)
# quadratic
glm.beetles2 <- glm(cbind(y, n - y) ~ conc + conc2, family = binomial, beetles)

## put model fits for two models together
beetles1 <- beetles
# put the fitted values in the data.frame
beetles1$fitted.values <- glm.beetles1$fitted.values
pred <- predict(glm.beetles1, data.frame(conc = beetles1$conc), type = "link", se.fit = TRUE) #£
beetles1$fit     <- pred$fit
beetles1$se.fit  <- pred$se.fit
# CI for fitted values
beetles1 <- within(beetles1, {
  fit.lower = exp(fit - 1.96 * se.fit) / (1 + exp(fit - 1.96 * se.fit))
  fit.upper = exp(fit + 1.96 * se.fit) / (1 + exp(fit + 1.96 * se.fit))
  })
beetles1$modelorder <- "linear"

beetles2 <- beetles
# put the fitted values in the data.frame
beetles2$fitted.values <- glm.beetles2$fitted.values
pred <- predict(glm.beetles2, data.frame(conc = beetles2$conc, conc2 = beetles2$conc2), type = "link", s
beetles2$fit     <- pred$fit
beetles2$se.fit  <- pred$se.fit
# CI for fitted values
beetles2 <- within(beetles2, {
  fit.lower = exp(fit - 1.96 * se.fit) / (1 + exp(fit - 1.96 * se.fit))
  fit.upper = exp(fit + 1.96 * se.fit) / (1 + exp(fit + 1.96 * se.fit))
  })
beetles2$modelorder <- "quadratic"

beetles.all <- rbind(beetles1, beetles2)
beetles.all$modelorder <- factor(beetles.all$modelorder)

# plot on logit and probability scales

library(ggplot2)
p <- ggplot(beetles.all, aes(x = conc, y = p.hat, shape = rep, colour = modelorder, fill = modelorder))
# predicted curve and point-wise 95% CI
p <- p + geom_ribbon(aes(x = conc, ymin = fit.lower, ymax = fit.upper), linetype = 0, alpha = 0.1)
p <- p + geom_line(aes(x = conc, y = fitted.values, linetype = modelorder), size = 1)
# fitted values
p <- p + geom_point(aes(y = fitted.values), size=2)
# observed values
p <- p + geom_point(color = "black", size = 3, alpha = 0.5)
p <- p + labs(title = "Observed and predicted mortality, probability scale")
```

```
print(p)

library(ggplot2)
p <- ggplot(beetles.all, aes(x = conc, y = emp.logit, shape = rep, colour = modelorder, fill = mo
# predicted curve and point-wise 95% CI
p <- p + geom_ribbon(aes(x = conc, ymin = fit - 1.96 * se.fit, ymax = fit + 1.96 * se.fit), linet
p <- p + geom_line(aes(x = conc, y = fit, linetype = modelorder), size = 1)
# fitted values
p <- p + geom_point(aes(y = fit), size=2)
# observed values
p <- p + geom_point(color = "black", size = 3, alpha = 0.5)
p <- p + labs(title = "Observed and predicted mortality, logit scale")
print(p)
```



11.5 Example: The UNM Trauma Data

The data to be analyzed here were collected on 3132 patients admitted to The University of New Mexico Trauma Center between the years 1991 and 1994. For each patient, the attending physician recorded their age, their revised trauma score (RTS), their injury severity score (ISS), whether their injuries were blunt (i.e., the result of a car crash: BP=0) or penetrating (i.e., gunshot/knife wounds: BP=1), and whether they eventually survived their injuries (SURV=0 if not, SURV=1 if survived). Approximately 10% of patients admitted to the UNM Trauma Center eventually die from their injuries.

The ISS is an overall index of a patient's injuries, based on the approximately 1300 injuries cataloged in the Abbreviated Injury Scale. The ISS can take on values

from 0 for a patient with no injuries to 75 for a patient with 3 or more life threatening injuries. The ISS is the standard injury index used by trauma centers throughout the U.S. The RTS is an index of physiologic injury, and is constructed as a weighted average of an incoming patient's systolic blood pressure, respiratory rate, and Glasgow Coma Scale. The RTS can take on values from 0 for a patient with no vital signs to 7.84 for a patient with normal vital signs.

Champion et al. (1981) proposed a logistic regression model to estimate the probability of a patient's survival as a function of RTS, the injury severity score ISS, and the patient's age, which is used as a surrogate for physiologic reserve. Subsequent survival models included the binary effect BP as a means to differentiate between blunt and penetrating injuries.

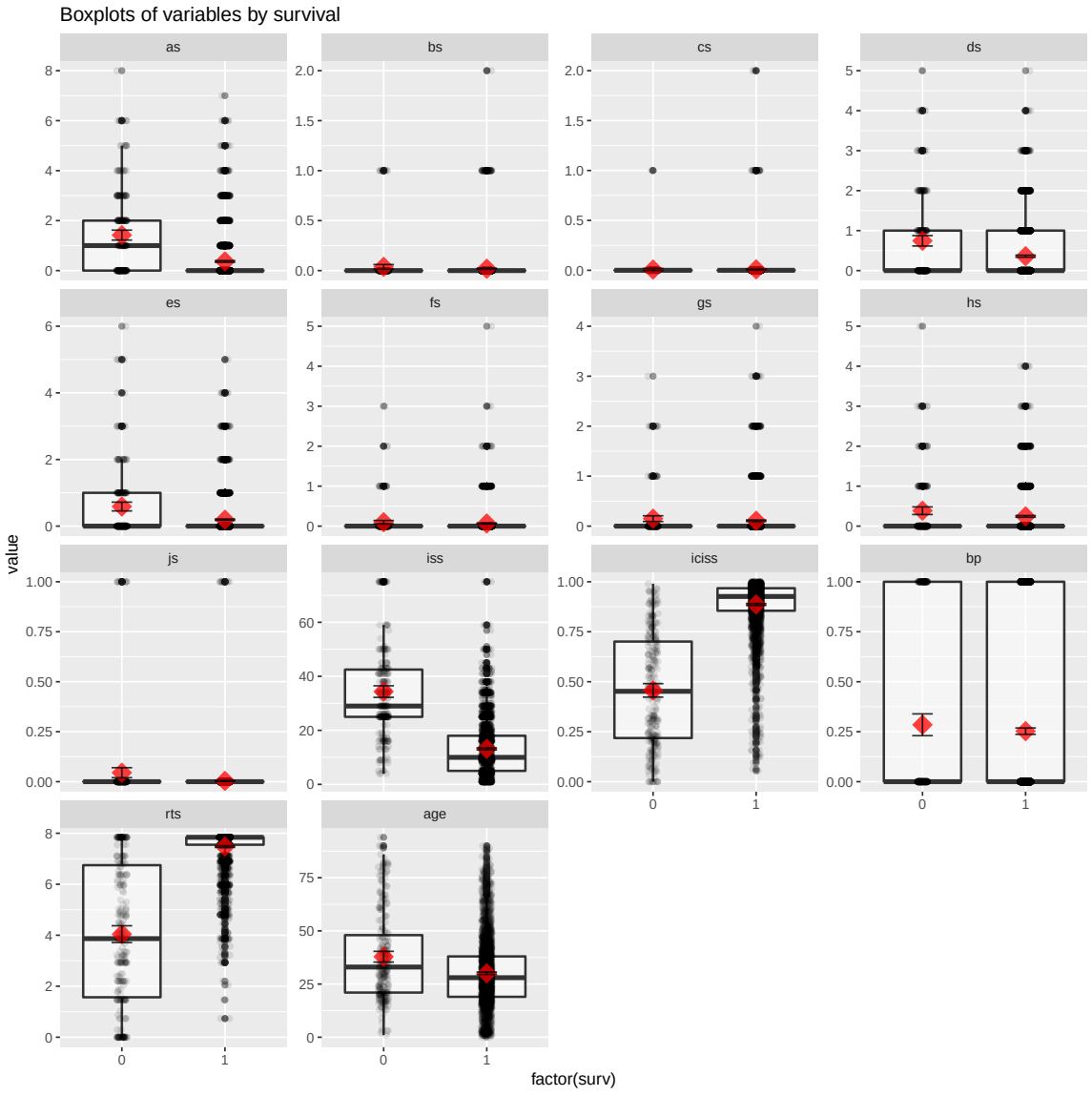
We will develop a logistic model for predicting survival from ISS, AGE, BP, and RTS, and nine body regions. Data on the number of severe injuries in each of the nine body regions is also included in the database, so we will also assess whether these features have any predictive power. The following labels were used to identify the number of severe injuries in the nine regions: AS = head, BS = face, CS = neck, DS = thorax, ES = abdomen, FS = spine, GS = upper extremities, HS = lower extremities, and JS = skin.

```
#### Example: UNM Trauma Data
trauma <- read.table("http://statacumen.com/teach/ADA2/ADA2_notes_Ch11_trauma.dat"
                     , header = TRUE)

## Variables
# surv = survival (1 if survived, 0 if died)
# rts  = revised trauma score (range: 0 no vital signs to 7.84 normal vital signs)
# iss  = injury severity score (0 no injuries to 75 for 3 or more life threatening injuries)
# bp   = blunt or penetrating injuries (e.g., car crash BP=0 vs gunshot/knife wounds BP=1)
# Severe injuries: add the severe injuries 3--6 to make summary variables
trauma <- within(trauma, {
  as = a3 + a4 + a5 + a6    # as = head
  bs = b3 + b4 + b5 + b6    # bs = face
  cs = c3 + c4 + c5 + c6    # cs = neck
  ds = d3 + d4 + d5 + d6    # ds = thorax
  es = e3 + e4 + e5 + e6    # es = abdomen
  fs = f3 + f4 + f5 + f6    # fs = spine
  gs = g3 + g4 + g5 + g6    # gs = upper extremities
  hs = h3 + h4 + h5 + h6    # hs = lower extremities
  js = j3 + j4 + j5 + j6    # js = skin
  })
# keep only columns of interest
names(trauma)
##  [1] "id"    "surv"  "a1"    "a2"    "a3"    "a4"    "a5"    "a6"    "b1"    "b2"
## [11] "b3"    "b4"    "b5"    "b6"    "c1"    "c2"    "c3"    "c4"    "c5"    "c6"
## [21] "d1"    "d2"    "d3"    "d4"    "d5"    "d6"    "e1"    "e2"    "e3"    "e4"
```

```
## [31] "e5"    "e6"    "f1"    "f2"    "f3"    "f4"    "f5"    "f6"    "g1"    "g2"
## [41] "g3"    "g4"    "g5"    "g6"    "h1"    "h2"    "h3"    "h4"    "h5"    "h6"
## [51] "j1"    "j2"    "j3"    "j4"    "j5"    "j6"    "iss"   "iciss" "bp"    "rts"
## [61] "age"   "prob"  "js"    "hs"    "gs"    "fs"    "es"    "ds"    "cs"    "bs"
## [71] "as"
trauma <- subset(trauma, select = c(id, surv, as:js, iss:prob))
head(trauma)
##        id surv as bs cs ds es fs gs hs js iss     iciss bp    rts age      prob
## 1 1238385    1  0  0  0  1  0  0  0  0  0  13 0.8612883  0 7.8408  13 0.9909890
## 2 1238393    1  0  0  0  0  0  0  0  0  0   5 0.9421876  0 7.8408  23 0.9947165
## 3 1238898    1  0  0  0  0  0  0  2  0  0  13 0.7251130  0 7.8408  43 0.9947165
## 4 1239516    1  1  0  0  0  0  0  0  0  0  16 1.0000000  0 5.9672  17 0.9615540
## 5 1239961    1  1  0  0  0  0  0  0  0  1   9 0.9346634  0 4.8040  20 0.9338096
## 6 1240266    1  0  0  0  0  0  0  0  1  0  13 0.9004691  0 7.8408  32 0.9947165
#str(trauma)
```

I made side-by-side boxplots of the distributions of ISS, AGE, RTS, and AS through JS for the survivors and non-survivors. In several body regions the number of injuries is limited, so these boxplots are not very enlightening. Survivors tend to have lower ISS scores, tend to be slightly younger, tend to have higher RTS scores, and tend to have fewer severe head (AS) and abdomen injuries (ES) than non-survivors. The importance of the effects individually towards predicting survival is directly related to the separation between the survivors and non-survivors scores.

```
# Create boxplots for each variable by survival
library(reshape2)
trauma.long <- melt(trauma, id.vars = c("id", "surv", "prob"))

# Plot the data using ggplot
library(ggplot2)
p <- ggplot(trauma.long, aes(x = factor(surv), y = value))
# 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 = 0.1)
# diamond at mean for each group
p <- p + stat_summary(fun.y = mean, geom = "point", shape = 18, size = 6,
                      alpha = 0.75, colour = "red")
# confidence limits based on normal distribution
p <- p + stat_summary(fun.data = "mean_cl_normal", geom = "errorbar",
                      width = .2, alpha = 0.8)
p <- p + facet_wrap( ~ variable, scales = "free_y", ncol = 4)
p <- p + labs(title = "Boxplots of variables by survival")
print(p)
```



11.5.1 Selecting Predictors in the Trauma Data

The same automated methods for model building are available for the `glm()` procedure, including backward elimination, forward selection, and stepwise methods, among others. Below we perform a stepwise selection using AIC starting at the full model, starting with a full model having 13 effects: ISS, BP, RTS, AGE, and AS–JS. Revisit Chapter 10 for more information.

In our previous logistic regression analyses, the cases in the data set were pre-aggregated into groups of observations having identical levels of the predictor vari-

ables. The numbers of cases in the success category and the group sample sizes were specified in the model statement, along with the names of the predictors. The trauma data set, which is not reproduced here, is **raw data** consisting of one record per patient (i.e., 3132 lines). The logistic model is fitted to data on individual cases by specifying the binary response variable (SURV) with successes and 1 − SURV failures with the predictors on the right-hand side of the formula. Keep in mind that we are defining the logistic model to model the **success** category, so we are modeling the probability of surviving.

As an aside, there are two easy ways to model the probability of dying (which we don't do below). The first is to swap the order the response is specified in the formula: `cbind(1 - surv, surv)`. The second is to convert a model for the log-odds of surviving to a model for the log-odds of dying by simply changing the sign of each regression coefficient in the model.

I only included the summary table from the backward elimination, and information on the fit of the selected model.

```
glm.tr <- glm(cbind(surv, 1 - surv) ~ as + bs + cs + ds + es + fs + gs + hs + js
                                      + iss + rts + age + bp
              , family = binomial, trauma)

# option: trace = 0 doesn't show each step of the automated selection
glm.tr.red.AIC <- step(glm.tr, direction="both", trace = 0)

# the anova object provides a summary of the selection steps in order
glm.tr.red.AIC$anova
```

```
##   Step Df   Deviance Resid. Df Resid. Dev      AIC
## 1      NA         NA      3118   869.4309 897.4309
## 2 - as  1 0.04911674      3119   869.4800 895.4800
## 3 - bs  1 0.12242766      3120   869.6024 893.6024
## 4 - fs  1 0.15243093      3121   869.7549 891.7549
## 5 - cs  1 0.78703127      3122   870.5419 890.5419
## 6 - ds  1 0.81690443      3123   871.3588 889.3588
## 7 - gs  1 1.18272567      3124   872.5415 888.5415
## 8 - hs  1 1.17941462      3125   873.7209 887.7209
## 9 - js  1 1.71204029      3126   875.4330 887.4330
```

The final model includes effects for ES (number of severe abdominal injuries), ISS, RTS, AGE, and BP. All of the effects in the selected model are significant at the 5% level.

```
summary(glm.tr.red.AIC)
```

```
##
## Call:
## glm(formula = cbind(surv, 1 - surv) ~ es + iss + rts + age +
##     bp, family = binomial, data = trauma)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -3.1546   0.0992   0.1432   0.2316   3.4454
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.355845   0.442943   0.803   0.4218
## es          -0.461317   0.110098  -4.190 2.79e-05 ***
## iss         -0.056920   0.007411  -7.680 1.59e-14 ***
## rts          0.843143   0.055339  15.236  < 2e-16 ***
## age         -0.049706   0.005291  -9.394  < 2e-16 ***
## bp          -0.635137   0.249597  -2.545   0.0109 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1825.37  on 3131  degrees of freedom
## Residual deviance:  875.43  on 3126  degrees of freedom
## AIC: 887.43
##
## Number of Fisher Scoring iterations: 7
```

The p-value for D is large, indicating no gross deficiencies with the selected model.

```
# Test residual deviance for lack-of-fit (if > 0.10, little-to-no lack-of-fit)
dev.p.val <- 1 - pchisq(glm.tr.red.AIC$deviance, glm.tr.red.AIC$df.residual)
dev.p.val
## [1] 1
```

Letting p be the probability of survival, the estimated survival probability is given by

$$\begin{aligned}\log\left(\frac{\tilde{p}}{1-\tilde{p}}\right) &= 0.3558 - 0.4613\ \text{ES} - 0.6351\ \text{BP} - 0.0569\ \text{ISS} \\ &\quad +0.8431\ \text{RTS} - 0.0497\ \text{AGE}.\end{aligned}$$

Let us interpret the sign of the coefficients, and the odds ratios, in terms of the impact that individual predictors have on the survival probability.

```
## coefficients and 95% CI
cbind(OR = coef(glm.tr.red.AIC), confint(glm.tr.red.AIC))
## Waiting for profiling to be done...
##                        OR       2.5 %      97.5 %
## (Intercept)  0.35584499 -0.51977300  1.21869015
## es          -0.46131679 -0.67603693 -0.24307991
## iss         -0.05691973 -0.07159539 -0.04249502
## rts          0.84314317  0.73817886  0.95531089
## age         -0.04970641 -0.06020882 -0.03943822
```

```
## bp          -0.63513735 -1.12051508 -0.14005288
## odds ratios and 95% CI
exp(cbind(OR = coef(glm.tr.red.AIC), confint(glm.tr.red.AIC)))
## Waiting for profiling to be done...
##                     OR     2.5 %    97.5 %
## (Intercept) 1.4273863 0.5946555 3.3827539
## es          0.6304529 0.5086287 0.7842088
## iss         0.9446699 0.9309075 0.9583952
## rts         2.3236592 2.0921220 2.5994786
## age         0.9515087 0.9415679 0.9613293
## bp          0.5298627 0.3261118 0.8693123
```

11.5.2 Checking Predictions for the Trauma Model

To assess the ability of the selected model to accurately predict survival, assume that the two types of errors (a false positive prediction of survival, and a false negative prediction) have equal costs. Under this assumption, the optimal classification rule is to predict survival for an individual with an estimated survival probability of 0.50 or larger.

In the below script, a table is generated that gives classifications for cutoff probabilities `thresh`$= 0.0, 0.1, \ldots, 1.0$ based on the selected model. While I do not do this below, it is common to use the **jackknife** method for assessing classification accuracy. To implement the jackknife method, each observation is temporarily held out and the selected model is fitted to the remaining (3131) cases. This leads to an estimated survival probability for the case, which is compared to the actual result. The results are summarized in terms of total number of correct and incorrect classifications for each possible outcome. In the table below, the columns labeled **Event** (a survival) and **Non-Event** (a death) refer to the classification of observations, and not to the actual outcomes. The columns labeled **Correct** and **Incorrect** identify whether the classifications are accurate.

```
# thresholds for classification given model proportion predictions for each observation
thresh  <- seq(0,1,by=0.1)

# predicted probabilities
Yhat <- fitted(glm.tr.red.AIC)

# Name: lower (0) = NonEvent, higher (1) = Event
YObs <- cut(trauma$surv, breaks = c(-Inf, mean(trauma$surv), Inf)
                         , labels = c("NonEvent", "Event"))

classify.table <- data.frame(Thresh     = rep(NA, length(thresh))
                             , Cor.Event  = rep(NA, length(thresh))
                             , Cor.NonEv  = rep(NA, length(thresh))
```

```
                          , Inc.Event  = rep(NA, length(thresh))
                          , Inc.NonEv  = rep(NA, length(thresh))
                          , Cor.All    = rep(NA, length(thresh))
                          , Sens       = rep(NA, length(thresh))
                          , Spec       = rep(NA, length(thresh))
                          , Fal.P      = rep(NA, length(thresh))
                          , Fal.N      = rep(NA, length(thresh)))

for (i.thresh in 1:length(thresh)) {
  # choose a threshold for dichotomizing according to predicted probability
  YhatPred <- cut(Yhat, breaks = c(-Inf, thresh[i.thresh], Inf)
                        , labels = c("NonEvent", "Event"))

  # contingency table and marginal sums
  cTab <- table(YhatPred, YObs)
  addmargins(cTab)

  # Classification Table
  classify.table$Thresh   [i.thresh] <- thresh[i.thresh]                 # Prob.Level
  classify.table$Cor.Event[i.thresh] <- cTab[2,2]                        # Correct.Event
  classify.table$Cor.NonEv[i.thresh] <- cTab[1,1]                        # Correct.NonEvent
  classify.table$Inc.Event[i.thresh] <- cTab[2,1]                        # Incorrect.Event
  classify.table$Inc.NonEv[i.thresh] <- cTab[1,2]                        # Incorrect.NonEvent
  classify.table$Cor.All  [i.thresh] <- 100 * sum(diag(cTab)) / sum(cTab) # Correct.Overall
  classify.table$Sens     [i.thresh] <- 100 * cTab[2,2] / sum(cTab[,2])  # Sensitivity
  classify.table$Spec     [i.thresh] <- 100 * cTab[1,1] / sum(cTab[,1])  # Specificity
  classify.table$Fal.P    [i.thresh] <- 100 * cTab[2,1] / sum(cTab[2,])  # False.Pos
  classify.table$Fal.N    [i.thresh] <- 100 * cTab[1,2] / sum(cTab[1,])  # False.Neg
}
round(classify.table, 1)
```

```
##    Thresh Cor.Event Cor.NonEv Inc.Event Inc.NonEv Cor.All  Sens  Spec Fal.P Fal.N
## 1     0.0      2865         0       267         0    91.5 100.0   0.0   8.5   NaN
## 2     0.1      2861        79       188         4    93.9  99.9  29.6   6.2   4.8
## 3     0.2      2856       105       162         9    94.5  99.7  39.3   5.4   7.9
## 4     0.3      2848       118       149        17    94.7  99.4  44.2   5.0  12.6
## 5     0.4      2837       125       142        28    94.6  99.0  46.8   4.8  18.3
## 6     0.5      2825       139       128        40    94.6  98.6  52.1   4.3  22.3
## 7     0.6      2805       157       110        60    94.6  97.9  58.8   3.8  27.6
## 8     0.7      2774       174        93        91    94.1  96.8  65.2   3.2  34.3
## 9     0.8      2727       196        71       138    93.3  95.2  73.4   2.5  41.3
## 10    0.9      2627       229        38       238    91.2  91.7  85.8   1.4  51.0
## 11    1.0         0       267         0      2865     8.5   0.0 100.0   NaN  91.5
```

The data set has 2865 survivors and 267 people that died. Using a 0.50 cutoff, 2825 of the survivors would be correctly identified, and 40 misclassified. Similarly, 139 of the patients that died would be correctly classified and 128 would not. The overall percentage of cases correctly classified is $(2825+138)/3132 = 94.6\%$. The **sensitivity** is the percentage of survivors that are correctly classified, $2825/(2825+40) = 98.6\%$.

The **specificity** is the percentage of patients that died that are correctly classified, $139/(139+128) = 52.1\%$. The **false positive rate**, which is the % of those predicted to survive that did not, is $128/(128+2825) = 4.3\%$. The **false negative rate**, which is the % of those predicted to die that did not is $40/(40+139) = 22.5\%$.

```
# Thresh = 0.5 classification table
YhatPred <- cut(Yhat, breaks=c(-Inf, 0.5, Inf), labels=c("NonEvent", "Event"))
# contingency table and marginal sums
cTab <- table(YhatPred, YObs)
addmargins(cTab)
```

```
##           YObs
## YhatPred   NonEvent Event  Sum
##   NonEvent      139    40  179
##   Event         128  2825 2953
##   Sum           267  2865 3132
```

```
round(subset(classify.table, Thresh == 0.5), 1)
```

```
##   Thresh Cor.Event Cor.NonEv Inc.Event Inc.NonEv Cor.All Sens Spec Fal.P Fal.N
## 6    0.5      2825       139       128        40    94.6 98.6 52.1   4.3  22.3
```

The misclassification rate seems small, but you should remember that approximately 10% of patients admitted to UNM eventually die from their injuries. Given this historical information only, you could achieve a 10% misclassification rate by completely ignoring the data and classifying each admitted patient as a survivor. Using the data reduces the misclassification rate by about 50% (from 10% down to 4.4%), which is an important reduction in this problem.

11.6 Historical Example: O-Ring Data

The table below presents data from Dalal, Fowlkes, and Hoadley (1989) on field O-ring failures in the 23 pre-*Challenger* space shuttle launches. Temperature at lift-off and O-ring joint pressure are predictors. The binary version of the data views each flight as an independent trial. The result of a trial (y) is a 1 if at least one field O-ring failure occurred on the flight and a 0 if all six O-rings functioned properly. A regression for the binary data would model the probability that **any** O-ring failed as a function of temperature and pressure. The binomial version of these data is also presented. It views the six field O-rings on each flight as independent trials. A regression for the binomial data would model the probability that **a particular** O-ring on a given flight failed as a function of temperature and pressure. The two regressions model different probabilities. The *Challenger* explosion occurred during a takeoff at 31 degrees Fahrenheit.

```
#### Example: Shuttle O-ring data
shuttle <- read.csv("http://statacumen.com/teach/ADA2/ADA2_notes_Ch11_shuttle.csv")
```

	case	flight	y	six	temp	pressure
1	1	14	1	2	53	50
2	2	9	1	1	57	50
3	3	23	1	1	58	200
4	4	10	1	1	63	50
5	5	1	0	0	66	200
6	6	5	0	0	67	50
7	7	13	0	0	67	200
8	8	15	0	0	67	50
9	9	4	0	0	68	200
10	10	3	0	0	69	200
11	11	8	0	0	70	50
12	12	17	0	0	70	200
13	13	2	1	1	70	200
14	14	11	1	1	70	200
15	15	6	0	0	72	200
16	16	7	0	0	73	200
17	17	16	0	0	75	100
18	18	21	1	2	75	200
19	19	19	0	0	76	200
20	20	22	0	0	76	200
21	21	12	0	0	78	200
22	22	20	0	0	79	200
23	23	18	0	0	81	200

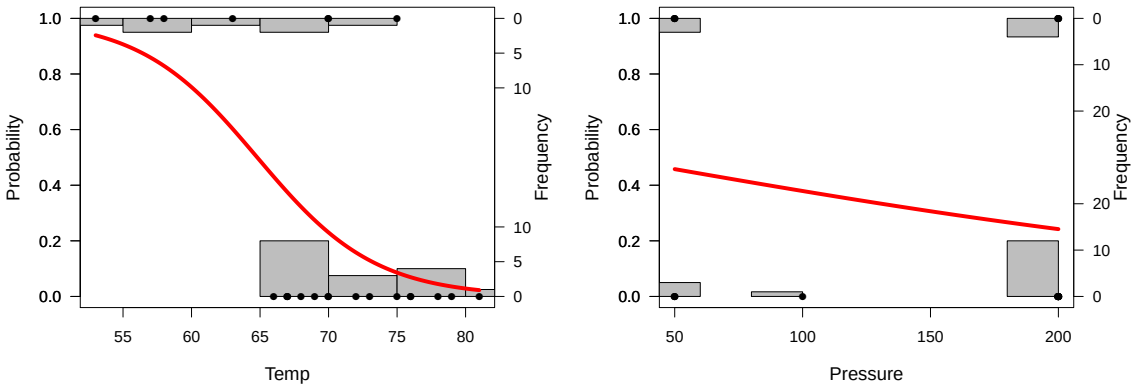
The regression model for the binomial data (i.e., six trials per launch) is suspect on substantive grounds. The binomial model makes no allowance for an effect due to having the six O-rings on the same flight. If these data were measurements rather than counts, they would almost certainly be treated as dependent repeated measures. If interest is focused on whether one or more O-rings failed, the simplest, most direct data are the binary data.

Consider fitting a logistic regression model using temperature and pressure as predictors. Let p_i be the probability that any O-ring fails in case i and model this as

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 \text{ Temp}_i + \beta_2 \text{ Pressure}_i.$$

Logistic histogram plots of the data show a clear marginal relationship of failure with temp but not with pressure. We still need to assess the model with both variables together.

```
# plot logistic plots of response to each predictor individually
library(popbio)
##
## Attaching package:  'popbio'
## The following object is masked from 'package:gdata':
##
##    resample
logi.hist.plot(shuttle$temp,     shuttle$y, boxp=FALSE, type="hist"
             , rug=TRUE, col="gray", ylabel = "Probability", xlabel = "Temp")
logi.hist.plot(shuttle$pressure, shuttle$y, boxp=FALSE, type="hist"
             , rug=TRUE, col="gray", ylabel = "Probability", xlabel = "Pressure")
```



We fit the logistic model below using $Y = 1$ if at least one O-ring failed, and 0 otherwise. We are modelling the chance of one or more O-ring failures as a function of temperature and pressure.

The D goodness-of-fit statistic suggest no gross deviations from the model. Furthermore, the test of $H_0 : \beta_1 = \beta_2 = 0$ (no regression effects) based on the Wald test has a p-value of 0.1, which suggests that neither temperature or pressure, or both, are useful predictors of the probability of O-ring failure. The z-test test p-values for testing $H_0 : \beta_1 = 0$ and $H_0 : \beta_2 = 0$ individually are 0.037 and 0.576, respectively, which indicates pressure is not important (when added last to the model), but that temperature is important. This conclusion might be anticipated by looking at data plots above.

```
glm.sh <- glm(cbind(y, 1 - y) ~ temp + pressure, family = binomial, shuttle)
# Test residual deviance for lack-of-fit (if > 0.10, little-to-no lack-of-fit)
dev.p.val <- 1 - pchisq(glm.sh$deviance, glm.sh$df.residual)
dev.p.val
## [1] 0.4589415
# Testing Global Null Hypothesis
library(aod)
coef(glm.sh)
##  (Intercept)         temp     pressure
## 16.385319489 -0.263404073  0.005177602
# specify which coefficients to test = 0  (Terms = 2:3 is for terms 2 and 3)
wald.test(b = coef(glm.sh), Sigma = vcov(glm.sh), Terms = 2:3)
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 4.6, df = 2, P(> X2) = 0.1
# Model summary
summary(glm.sh)
```

```
##
## Call:
## glm(formula = cbind(y, 1 - y) ~ temp + pressure, family = binomial,
##     data = shuttle)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1928  -0.7879  -0.3789   0.4172   2.2031
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 16.385319   8.027474   2.041   0.0412 *
## temp        -0.263404   0.126371  -2.084   0.0371 *
## pressure     0.005178   0.009257   0.559   0.5760
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 28.267  on 22  degrees of freedom
## Residual deviance: 19.984  on 20  degrees of freedom
## AIC: 25.984
##
## Number of Fisher Scoring iterations: 5
```

A reasonable next step would be to refit the model, after omitting pressure as a predictor. The target model is now

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 \text{ Temp}_i.$$

```
glm.sh <- glm(cbind(y, 1 - y) ~ temp, family = binomial, shuttle)
# Test residual deviance for lack-of-fit (if > 0.10, little-to-no lack-of-fit)
dev.p.val <- 1 - pchisq(glm.sh$deviance, glm.sh$df.residual)
dev.p.val
## [1] 0.5013827
# Model summary
summary(glm.sh)
##
## Call:
## glm(formula = cbind(y, 1 - y) ~ temp, family = binomial, data = shuttle)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0611  -0.7613  -0.3783   0.4524   2.2175
##
## Coefficients:
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  15.0429     7.3786   2.039   0.0415 *
## temp         -0.2322     0.1082  -2.145   0.0320 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 28.267  on 22  degrees of freedom
## Residual deviance: 20.315  on 21  degrees of freedom
## AIC: 24.315
##
## Number of Fisher Scoring iterations: 5
```

Our conclusions on the overall fit of the model and the significance of the effect of temperature on the probability of O-ring failure are consistent with the results for two predictor model. The model estimates the log-odds of (at least one) O-ring failure to be

$$\log\left(\frac{\tilde{p}}{1-\tilde{p}}\right) = 15.043 - 0.2322 \text{ Temp}.$$

The estimated probability of (at least one) O-ring failure is

$$\tilde{p} = \frac{\exp(15.043 - 0.2322 \text{ Temp})}{1 + \exp(15.043 - 0.2322 \text{ Temp})}.$$

This is an decreasing function of temperature.

The *Challenger* was set to launch on a morning where the predicted temperature at lift-off was 31 degrees. Given that temperature appears to affect the probability of O-ring failure (a point that NASA missed), what can/should we say about the potential for O-ring failure?

Clearly, the launch temperature is outside the region for which data were available. Thus, we really have no prior information about what is likely to occur. If we assume the logistic model holds, and we can extrapolate back to 31 degrees, what is the estimated probability of O-ring failure?

The following gives the answer this question. I augmented the original data set to obtain predicted probabilities for temperatures not observed in the data set. Note that the fitted model to data with missing observations gives the same model fit because `glm()` excludes observations with missing values. Predictions are then made for all temperatures in the dataset and the resulting table and plot are reported.

```
# append values to dataset for which we wish to make predictions
shuttle.pred <- data.frame( case     = rep(NA, 5)
                          , flight   = rep(NA, 5)
                          , y        = rep(NA, 5)
                          , six      = rep(NA, 5)
```

```
                          , temp     = c(31, 35, 40, 45, 50) # temp values to predict
                          , pressure = rep(NA, 5)
                          )
shuttle <- rbind(shuttle.pred, shuttle)
# fit model
glm.sh <- glm(cbind(y, 1 - y) ~ temp, family = binomial, shuttle)
# Note: same model fit as before since glm() does not use missing values
round(summary(glm.sh)$coefficients, 3)
```

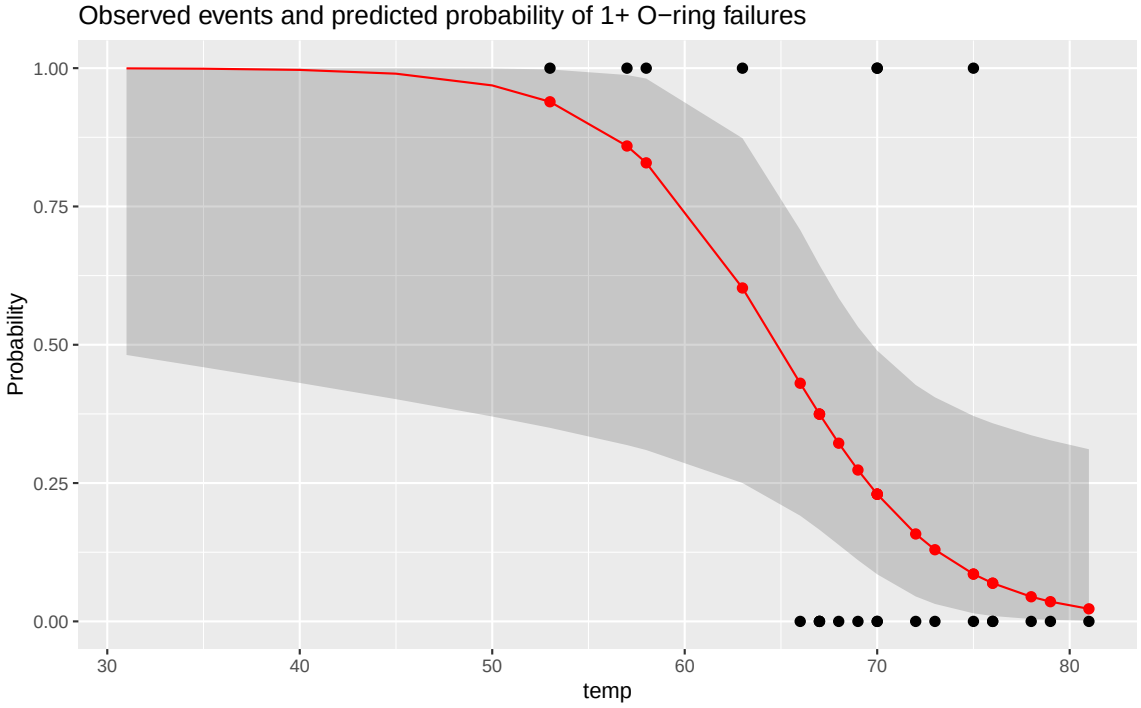
```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   15.043      7.379   2.039    0.041
## temp          -0.232      0.108  -2.145    0.032
```

```
# put the fitted values in the data.frame
shuttle$fitted.values <- c(rep(NA, 5), glm.sh$fitted.values)
# predict() uses all the temp values in dataset, including appended values
pred <- predict(glm.sh, data.frame(temp = shuttle$temp), type = "link", se.fit = TRUE)
shuttle$fit    <- pred$fit
shuttle$se.fit <- pred$se.fit
# CI for fitted values
shuttle <- within(shuttle, {
  # added "fitted" to make predictions at appended temp values
  fitted    = exp(fit) / (1 + exp(fit))
  fit.lower = exp(fit - 1.96 * se.fit) / (1 + exp(fit - 1.96 * se.fit))
  fit.upper = exp(fit + 1.96 * se.fit) / (1 + exp(fit + 1.96 * se.fit))
  })
```

	case	flight	y	six	temp	pressure	fitted.values	fit	se.fit	fit.upper	fit.lower	fitted
1					31.00			7.85	4.04	1.00	0.48	1.00
2					35.00			6.92	3.61	1.00	0.46	1.00
3					40.00			5.76	3.08	1.00	0.43	1.00
4					45.00			4.60	2.55	1.00	0.40	0.99
5					50.00			3.43	2.02	1.00	0.37	0.97
6	1.00	14.00	1.00	2.00	53.00	50.00	0.94	2.74	1.71	1.00	0.35	0.94
7	2.00	9.00	1.00	1.00	57.00	50.00	0.86	1.81	1.31	0.99	0.32	0.86
8	3.00	23.00	1.00	1.00	58.00	200.00	0.83	1.58	1.21	0.98	0.31	0.83
9	4.00	10.00	1.00	1.00	63.00	50.00	0.60	0.42	0.77	0.87	0.25	0.60
10	5.00	1.00	0.00	0.00	66.00	200.00	0.43	−0.28	0.59	0.71	0.19	0.43
11	6.00	5.00	0.00	0.00	67.00	50.00	0.38	−0.51	0.56	0.64	0.17	0.38
12	7.00	13.00	0.00	0.00	67.00	200.00	0.38	−0.51	0.56	0.64	0.17	0.38
13	8.00	15.00	0.00	0.00	67.00	50.00	0.38	−0.51	0.56	0.64	0.17	0.38
14	9.00	4.00	0.00	0.00	68.00	200.00	0.32	−0.74	0.55	0.58	0.14	0.32
15	10.00	3.00	0.00	0.00	69.00	200.00	0.27	−0.98	0.56	0.53	0.11	0.27
16	11.00	8.00	0.00	0.00	70.00	50.00	0.23	−1.21	0.59	0.49	0.09	0.23
17	12.00	17.00	0.00	0.00	70.00	200.00	0.23	−1.21	0.59	0.49	0.09	0.23
18	13.00	2.00	1.00	1.00	70.00	200.00	0.23	−1.21	0.59	0.49	0.09	0.23
19	14.00	11.00	1.00	1.00	70.00	200.00	0.23	−1.21	0.59	0.49	0.09	0.23
20	15.00	6.00	0.00	0.00	72.00	200.00	0.16	−1.67	0.70	0.43	0.05	0.16
21	16.00	7.00	0.00	0.00	73.00	200.00	0.13	−1.90	0.78	0.40	0.03	0.13
22	17.00	16.00	0.00	0.00	75.00	100.00	0.09	−2.37	0.94	0.37	0.01	0.09
23	18.00	21.00	1.00	2.00	75.00	200.00	0.09	−2.37	0.94	0.37	0.01	0.09
24	19.00	19.00	0.00	0.00	76.00	200.00	0.07	−2.60	1.03	0.36	0.01	0.07
25	20.00	22.00	0.00	0.00	76.00	200.00	0.07	−2.60	1.03	0.36	0.01	0.07
26	21.00	12.00	0.00	0.00	78.00	200.00	0.04	−3.07	1.22	0.34	0.00	0.04
27	22.00	20.00	0.00	0.00	79.00	200.00	0.04	−3.30	1.32	0.33	0.00	0.04
28	23.00	18.00	0.00	0.00	81.00	200.00	0.02	−3.76	1.51	0.31	0.00	0.02

```
library(ggplot2)
p <- ggplot(shuttle, aes(x = temp, y = y))
# predicted curve and point-wise 95% CI
p <- p + geom_ribbon(aes(x = temp, ymin = fit.lower, ymax = fit.upper), alpha = 0.2)
p <- p + geom_line(aes(x = temp, y = fitted), colour="red")
# fitted values
p <- p + geom_point(aes(y = fitted.values), size=2, colour="red")
# observed values
p <- p + geom_point(size = 2)
p <- p + ylab("Probability")
p <- p + labs(title = "Observed events and predicted probability of 1+ O-ring failures")
print(p)
```

```
## Warning:  Removed 5 rows containing missing values (geom_point).
## Warning:  Removed 5 rows containing missing values (geom_point).
```



Part IX

ADA2: Multivariate Methods

Chapter 12

An Introduction to Multivariate Methods

Multivariate statistical methods are used to display, analyze, and describe data on two or more features or variables simultaneously. I will discuss multivariate methods for measurement data. Methods for multi-dimensional count data, or mixtures of counts and measurements are available, but are beyond the scope of what I can do here. I will give a brief overview of the type of problems where multivariate methods are appropriate.

Example: Turtle shells Jolicouer and Mosimann provided data on the height, length, and width of the carapace (shell) for a sample of female painted turtles. **Cluster analysis** is used to identify which shells are similar on the three features. **Principal component analysis** is used to identify the linear combinations of the measurements that account for most of the variation in size and shape of the shells.

Cluster analysis and principal component analysis are primarily descriptive techniques.

Example: Fisher's Iris data Random samples of 50 flowers were selected from three iris species: Setosa, Virginica, and Versicolor. Four measurements were made on each flower: sepal length, sepal width, petal length, and petal width. Suppose the sample means on each feature are computed within the three species. Are the means on the four traits significantly different across species? This question can be answered using four separate one-way ANOVAs. A more powerful **MANOVA** (multivariate analysis of variance) method compares species on the four features simultaneously.

Discriminant analysis is a technique for comparing groups on multi-dimensional data. Discriminant analysis can be used with Fisher's Iris data to find the linear combinations of the flower features that best distinguish species. The linear combinations

are optimally selected, so insignificant differences on one or all features may be significant (or better yet, important) when the features are considered simultaneously! Furthermore, the discriminant analysis could be used to classify flowers into one of these three species when their species is unknown.

MANOVA, discriminant analysis, and **classification** are primarily inferential techniques.

12.1 Linear Combinations

Suppose data are collected on p measurements or features $X_1, X_2, \ldots, X_p$. Most multivariate methods use **linear combinations** of the features as the basis for analysis. A linear combination has the form

$$Y = a_1X_1 + a_2X_2 + \cdots + a_pX_p,$$

where the coefficients $a_1, a_2, \ldots, a_p$ are known constants. Y is evaluated for each observation in the data set, keeping the coefficients constant.

For example, three linear combinations of $X_1, X_2, \ldots, X_p$ are:

$$\begin{aligned} Y &= 1X_1 + 0X_2 + 0X_3 + \cdots + 0X_p = X_1, \\ Y &= \frac{1}{p}(X_1 + X_2 + \cdots + X_p), \text{ and} \\ Y &= 2X_1 - 4X_2 + 55X_3 - 1954X_4 + \cdots + 44X_p. \end{aligned}$$

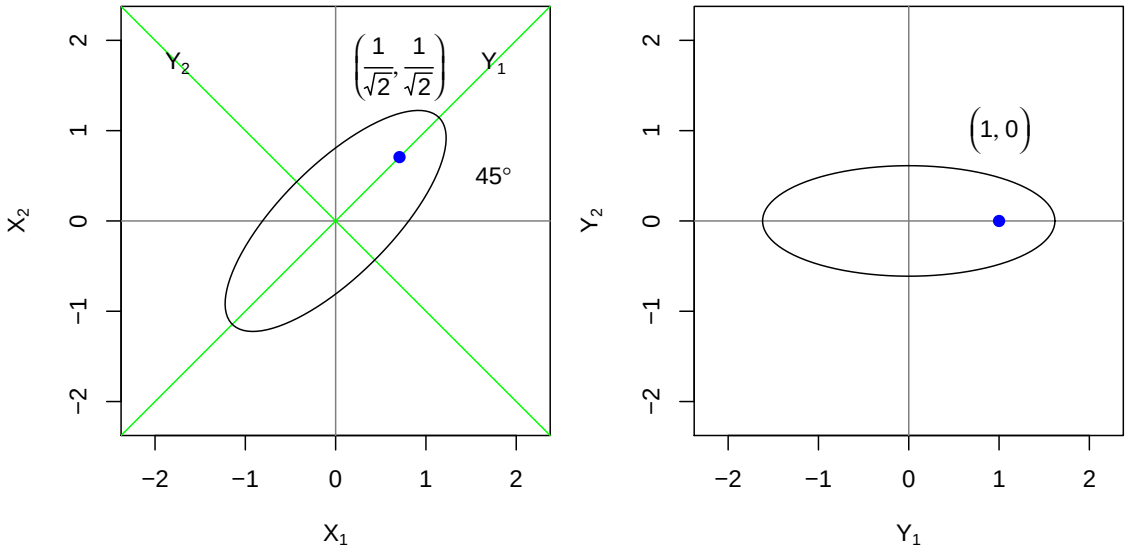
Vector and **matrix** notation are useful for representing and summarizing multivariate data. Before introducing this notation, let us try to understand linear combinations geometrically when $p = 2$.

Example: $-45°$ rotation A plot of data on two features X_1 and X_2 is given below. Also included is a plot for the two linear combinations

$$\begin{aligned} Y_1 &= \frac{1}{\sqrt{2}}(X_1 + X_2) \qquad \text{and} \\ Y_2 &= \frac{1}{\sqrt{2}}(X_2 - X_1). \end{aligned}$$

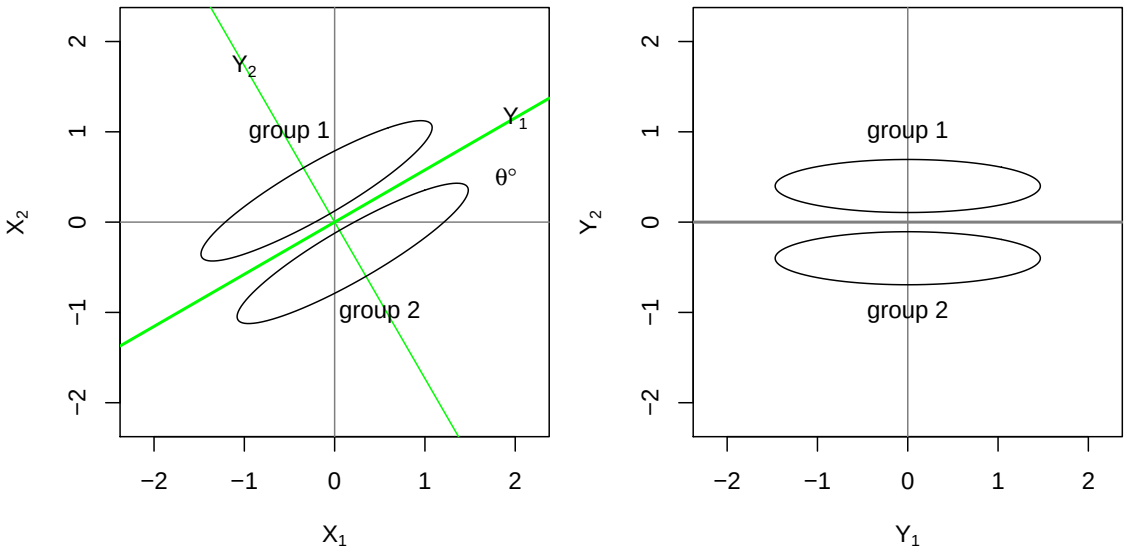
This transformation creates two (roughly) uncorrelated linear combinations Y_1 and Y_2 from two highly correlated features X_1 and X_2. The transformation corresponds to a **rotation** of the original coordinate axes by -45 degrees. Each data point is then expressed relative to the new axes. The new features are uncorrelated!

```
##
## Attaching package:  'ellipse'
## The following object is masked from 'package:car':
##
##    ellipse
## The following object is masked from 'package:graphics':
##
##    pairs
```



The $\sqrt{2}$ divisor in Y_1 and Y_2 does not alter the interpretation of these linear combinations: Y_1 is essentially the sum of X_1 and X_2, whereas Y_2 is essentially the difference between X_2 and X_1.

Example: Two groups The plot below shows data on two features X_1 and X_2 from two distinct groups.



If you compare the groups on X_1 and X_2 separately, you may find no significant differences because the groups overlap substantially on each feature. The plot on the right was obtained by rotating the coordinate axes $-\theta$ degrees, and then plotting the data relative to the new coordinate axes. The rotation corresponds to creating two linear combinations:

$$\begin{aligned} Y_1 &= \cos(\theta)X_1 + \sin(\theta)X_2 \\ Y_2 &= -\sin(\theta)X_1 + \cos(\theta)X_2. \end{aligned}$$

The two groups differ substantially on Y_2. This linear combination is used with discriminant analysis and MANOVA to distinguish between the groups.

The linear combinations used in certain multivariate methods do not correspond to a rotation of the original coordinate axes. However, the pictures given above should provide some insight into the motivation for the creating linear combinations of two features. The ideas extend to three or more features, but are more difficult to represent visually.

12.2 Vector and Matrix Notation

A **vector** is a string of numbers or variables that is stored in either a row or in a column. For example, the collection $X_1, X_2, \ldots, X_p$ of features can be represented as

a **column-vector** with p rows, using the notation

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \end{bmatrix}.$$

The entry in the j^{th} row is X_j. The transpose of X, represented by X', is a **row-vector** with p columns: $X' = (X_1, X_2, \ldots, X_p)$. The j^{th} column of X' contains X_j.

Suppose you collect data on p features $X_1, X_2, \ldots, X_p$ for a sample of n individuals. The data for the i^{th} individual can be represented as the column-vector:

$$X_i = \begin{bmatrix} X_{i1} \\ X_{i2} \\ \vdots \\ X_{ip} \end{bmatrix}.$$

or as the row-vector $X_i' = (X_{i1}, X_{i2}, \cdots, X_{ip})$. Here X_{ij} is the value on the j^{th} variable. Two subscripts are needed for the data values. One subscript identifies the individual and the other subscript identifies the feature.

A **matrix** is a rectangular array of numbers or variables. A data set can be viewed as a matrix with n rows and p columns, where n is the sample size. Each row contains data for a given individual:

$$\begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1p} \\ X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{np} \end{bmatrix}.$$

Vector and matrix notation are used for summarizing multivariate data. For example, the **sample mean vector** is

$$\bar{X} = \begin{bmatrix} \bar{X}_1 \\ \bar{X}_2 \\ \vdots \\ \bar{X}_p \end{bmatrix},$$

where $\bar{X}_j$ is the sample average on the j^{th} feature. Using matrix algebra, $\bar{X}$ is defined using a familiar formula:

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i.$$

This mathematical operation is well-defined because vectors are added elementwise.

The sample variances and covariances on the p variables can be grouped together in a $p \times p$ **sample variance-covariance matrix** S (i.e., p rows and p columns)

$$S = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1p} \\ s_{21} & s_{22} & \cdots & s_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ s_{p1} & s_{p2} & \cdots & s_{pp} \end{bmatrix},$$

where

$$s_{ii} = \frac{1}{n-1}\sum_{k=1}^{n}(X_{ki} - \bar{X}_i)^2$$

is the sample variance for the i^{th} feature, and

$$s_{ij} = \frac{1}{n-1}\sum_{k=1}^{n}(X_{ki} - \bar{X}_i)(X_{kj} - \bar{X}_j)$$

is the sample covariance between the i^{th} and j^{th} features. The subscripts on the elements in S identify where the element is found in the matrix: s_{ij} is stored in the i^{th} row and the j^{th} column. The variances are found on the **main diagonal** of the matrix. The covariances are **off-diagonal** elements. S is symmetric, meaning that the elements above the main diagonal are a reflection of the entries below the main diagonal. More formally, $s_{ij} = s_{ji}$.

Matrix algebra allows you to express S using a formula analogous to the sample variance for a single feature:

$$S = \frac{1}{n-1}\sum_{k=1}^{n}(X_k - \bar{X})(X_k - \bar{X})'.$$

Here $(X_k - \bar{X})(X_k - \bar{X})'$ is the matrix product of a column vector with p entries times a row vector with p entries. This matrix product is a $p \times p$ matrix with $(X_{ki} - \bar{X}_i)(X_{kj} - \bar{X}_j)$ in the i^{th} row and j^{th} column. The matrix products are added up over all n observations and then divided by $n - 1$.

The interpretation of covariances is enhanced by standardizing them to give correlations. The **sample correlation matrix** is denoted by the $p \times p$ symmetric matrix

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1p} \\ r_{21} & r_{22} & \cdots & r_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ r_{p1} & r_{p2} & \cdots & r_{pp} \end{bmatrix}.$$

The i^{th} row and j^{th} column element of R is the correlation between the i^{th} and j^{th} features. The diagonal elements are one: $r_{ii} = 1$. The off-diagonal elements satisfy

$$r_{ij} = r_{ji} = \frac{s_{ij}}{\sqrt{s_{ii}s_{jj}}}.$$

In many applications the data are standardized to have mean 0 and variance 1 on each feature. The data are standardized through the so-called **Z-score transformation**: $(X_{ki} - \bar{X}_i)/s_{ii}$ which, on each feature, subtracts the mean from each observation and divides by the corresponding standard deviation. The sample variance-covariance matrix for the standardized data is the correlation matrix R for the raw data.

Example: Let X_1, X_2, and X_3 be the reaction times for three visual stimuli named A, B and C, respectively. Suppose you are given the following summaries based on a sample of 30 individuals:

$$\bar{X} = \begin{bmatrix} 4 \\ 5 \\ 4.7 \end{bmatrix},$$

$$S = \begin{bmatrix} 2.26 & 2.18 & 1.63 \\ 2.18 & 2.66 & 1.82 \\ 1.63 & 1.82 & 2.47 \end{bmatrix},$$

$$R = \begin{bmatrix} 1.00 & 0.89 & 0.69 \\ 0.89 & 1.00 & 0.71 \\ 0.69 & 0.71 & 1.00 \end{bmatrix}.$$

The average response time on B is 5. The sample variance of response times on A is 2.26. The sample covariance between response times on A and C is 1.63. The sample correlation between response times on B and C is 0.71.

12.3 Matrix Notation to Summarize Linear Combinations

Matrix algebra is useful for computing sample summaries for linear combinations of the features $X' = (X_1, X_2, \ldots, X_p)$ from the sample summaries on these features. For example, suppose you define the linear combination

$$Y_1 = a_1X_1 + a_2X_2 + \cdots + a_pX_p.$$

Using matrix algebra, Y_1 is the matrix product $Y_1 = a'X$, where $a' = (a_1, a_2, \ldots, a_p)$. The sample mean and variance of Y_1 are

$$\bar{Y} = a_1\bar{X}_1 + a_2\bar{X}_2 + \cdots + a_p\bar{X}_p = a'\bar{X}$$

and

$$s_Y^2 = \sum_{ij} a_i a_j s_{ij} = a'Sa,$$

where $\bar{X}$ and S are the sample mean vector and sample variance-covariance matrix for $X' = (X_1, X_2, \ldots, X_p)$.

Similarly, the sample covariance between Y_1 and

$$Y_2 = b'X = b_1X_1 + b_2X_2 + \cdots + b_pX_p$$

is

$$s_{Y_1,Y_2} = \sum_{ij} a_i b_j s_{ij} = a'Sb = b'Sa.$$

Example: In the stimuli example, the total reaction time per individual is

$$Y \;=\; [1\ 1\ 1]\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} \;=\; X_1 + X_2 + X_3.$$

The mean reaction time is

$$\begin{aligned} \bar{Y} &= [1\ 1\ 1] \\ \bar{X} &= [1\ 1\ 1]\begin{bmatrix} 4 \\ 5 \\ 4.7 \end{bmatrix} \;=\; 4 + 5 + 4.7 \;=\; 13.7. \end{aligned}$$

The variance of Y is the sum of the elements in the variance-covariance matrix:

$$s_Y^2 \;=\; [1\ 1\ 1]\, S \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \;=\; \sum_{ij} s_{ij} \;=\; 2.26 + 2.18 + \cdots + 2.47 \;=\; 18.65.$$

Chapter 13

Principal Component Analysis

Principal component analysis (PCA) is a multivariate technique for understanding variation, and for summarizing measurement data possibly through **variable reduction**. Principal components (the variables created in PCA) are sometimes used in addition to, or in place of, the original variables in certain analyses. I will illustrate the use and misuse of principal components in a series of examples.

Given data on p variables or features X_1, X_2, ..., X_p, PCA uses a rotation of the original coordinate axes to produce a **new** set of p **uncorrelated** variables, called principal components, that are **unit-length linear combinations** of the original variables. A unit-length linear combination $a_1X_1 + a_2X_2 + \cdots + a_pX_p$ has $a_1^2 + a_2^2 + \cdots + a_p^2 = 1$.

The principal components have the following properties. The **first principal component**

$$\text{PRIN1} = a_{11}X_1 + a_{12}X_2 + \cdots + a_{1p}X_p$$

has the **largest variability** among all unit-length linear combinations of the original variables. The **second principal component**

$$\text{PRIN2} = a_{21}X_1 + a_{22}X_2 + \cdots + a_{2p}X_p$$

has the largest variability among all unit-length linear combinations of X_1, X_2, ..., X_p that are uncorrelated with PRIN1. In general, the j^{th} principal component PRINj for $j = 1, 2, \ldots, p$, has the largest variability among all unit-length linear combinations of the features that are uncorrelated with PRIN1, PRIN2, ... , PRIN$(j-1)$. The **last** or p^{th} **principal component** PRINp has the **smallest variability** among all unit-length linear combinations of the features.

I have described PCA on the raw or unstandardized data. This method is often called PCA on the sample covariance matrix, because the principal components are computed numerically using a **singular value decomposition** of the sample covariance matrix for the X_is. The variances of the PCs are **eigenvalues** of the sample

covariance matrix. The coefficients in the PCs are **eigenvectors** of the sample covariance matrix. The sum of the variances of the principal components is equal to the sum of the variances in the original features. An alternative method for PCA uses standardized data, which is often called PCA on the correlation matrix.

The ordered principal components are uncorrelated variables with progressively less variation. Principal components are often viewed as separate dimensions corresponding to the collection of features. The variability of each component divided by the total variability of the components is the proportion of the total variation in the data captured by each component. If data reduction is your goal, then you might need only the first few principal components to capture most of the variability in the data. This issue will be returned to later.

The unit-length constraint on the coefficients in PCA is needed to make the maximization well-defined. Without this constraint, there does not exist a linear combination with maximum variation. For example, the variability of an arbitrary linear combination $a_1X_1 + a_2X_2 + \cdots + a_pX_p$ is increased by 100 when each coefficient is multiplied by 10!

The principal components are unique only up to a change of the sign for each coefficient. For example,

$$\text{PRIN1} = 0.2X_1 - 0.4X_2 + 0.4X_3 + 0.8X_4$$

and

$$\text{PRIN1} = -0.2X_1 + 0.4X_2 - 0.4X_3 - 0.8X_4$$

have the same variability, so either could play the role of the first principal component. This non-uniqueness does not have an important impact on the analysis.

13.1 Example: Temperature Data

The following temperature example includes mean monthly temperatures in January and July for 64 U.S. cities.

```
#### Example: Temperature of cities
## The Temperature data file is in "fixed width format", an older data file format.
## Each field is specified by column ranges.
## Below I've provided numbers to help identify the column numbers
##   as well as the first three observations in the dataset.
## 12345678901234567890123456
## [  14 char   ][ 5 ][ 5 ]
#  mobile         51.2 81.6
#  phoenix        51.2 91.2
#  little rock    39.5 81.4

fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch13_temperature.dat"
```

```
temp <- read.fwf(fn.data, widths = c(14, 5, 5))
# the city names have trailing white space (we fix this below)
str(temp)
```

```
## 'data.frame': 64 obs. of  3 variables:
##  $ V1: Factor w/ 64 levels "albany        ",..: 39 48 33 56 21 27 64 62 31 36 ...
##  $ V2: num  51.2 51.2 39.5 45.1 29.9 24.8 32 35.6 54.6 67.2 ...
##  $ V3: num  81.6 91.2 81.4 75.2 73 72.7 75.8 78.7 81 82.3 ...
```

```
head(temp)
```

```
##               V1   V2   V3
## 1 mobile         51.2 81.6
## 2 phoenix        51.2 91.2
## 3 little rock    39.5 81.4
## 4 sacramento     45.1 75.2
## 5 denver         29.9 73.0
## 6 hartford       24.8 72.7
```

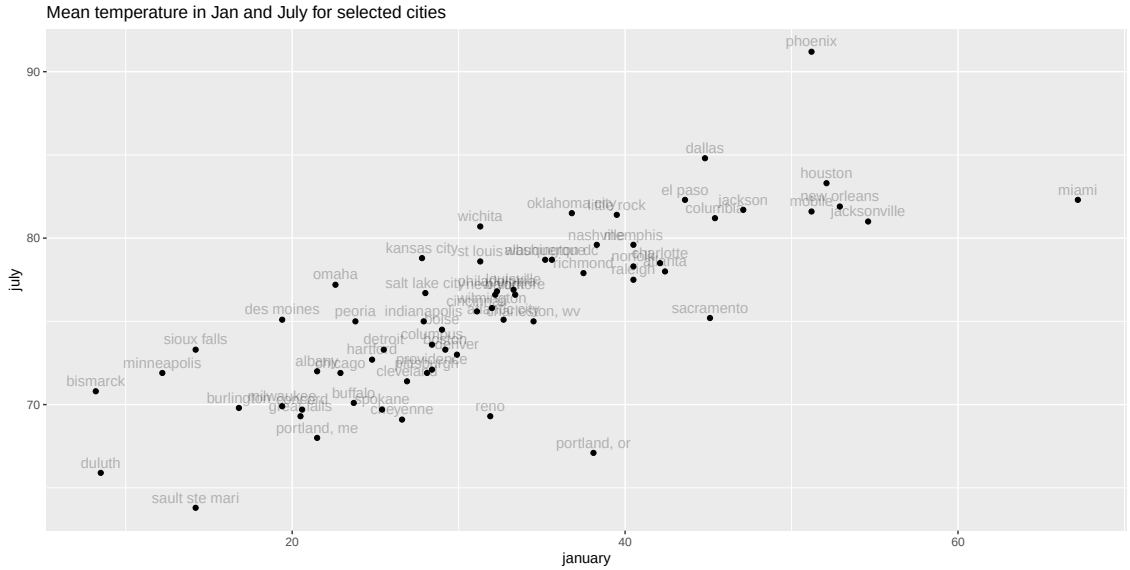
```
# remove that white space with strip.white=TRUE
temp <- read.fwf(fn.data, widths = c(14, 5, 5), strip.white = TRUE)
# name columns
colnames(temp) <- c("city", "january", "july")
temp$id <- 1:nrow(temp)
str(temp)
```

```
## 'data.frame': 64 obs. of  4 variables:
##  $ city   : Factor w/ 64 levels "albany","albuquerque",..: 39 48 33 56 21 27 64 62 31 36 ...
##  $ january: num  51.2 51.2 39.5 45.1 29.9 24.8 32 35.6 54.6 67.2 ...
##  $ july   : num  81.6 91.2 81.4 75.2 73 72.7 75.8 78.7 81 82.3 ...
##  $ id     : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
head(temp)
```

```
##          city january july id
## 1      mobile    51.2 81.6  1
## 2     phoenix    51.2 91.2  2
## 3 little rock    39.5 81.4  3
## 4  sacramento    45.1 75.2  4
## 5      denver    29.9 73.0  5
## 6    hartford    24.8 72.7  6
```

```
# plot original data
library(ggplot2)
p1 <- ggplot(temp, aes(x = january, y = july))
p1 <- p1 + geom_point()           # points
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                  # good idea since both are in the same units
p1 <- p1 + geom_text(aes(label = city), vjust = -0.5, alpha = 0.25) # city labels
p1 <- p1 + labs(title = "Mean temperature in Jan and July for selected cities")
print(p1)
```



The `princomp()` procedure is used for PCA. By default the principal components are computed based on the covariance matrix. The correlation matrix may also be used (it effectively z-scores the data first) with the `cor = TRUE` option. The **principal component scores** are the values of the principal components across cases. The principal component scores PRIN1, PRIN2, ..., PRINp are centered to have mean zero.

Output from a PCA on the covariance matrix is given. Two principal components are created because $p = 2$.

```
# perform PCA on covariance matrix
temp.pca <- princomp( ~ january + july, data = temp)
# standard deviation and proportion of variation for each component
summary(temp.pca)
## Importance of components:
##                            Comp.1    Comp.2
## Standard deviation     12.3217642 3.0004557
## Proportion of Variance  0.9440228 0.0559772
## Cumulative Proportion   0.9440228 1.0000000
# coefficients for PCs
loadings(temp.pca)
##
## Loadings:
##         Comp.1 Comp.2
## january  0.939  0.343
## july     0.343 -0.939
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
```

```
## Cumulative Var    0.5    1.0
```

```
# scores are coordinates of each observation on PC scale
head(temp.pca$scores)
```

```
##      Comp.1     Comp.2
## 1 20.000106  0.9239612
## 2 23.291460 -8.0941867
## 3  8.940669 -2.8994977
## 4 12.075589  4.8446790
## 5 -2.957414  1.7000283
## 6 -7.851160  0.2333138
```

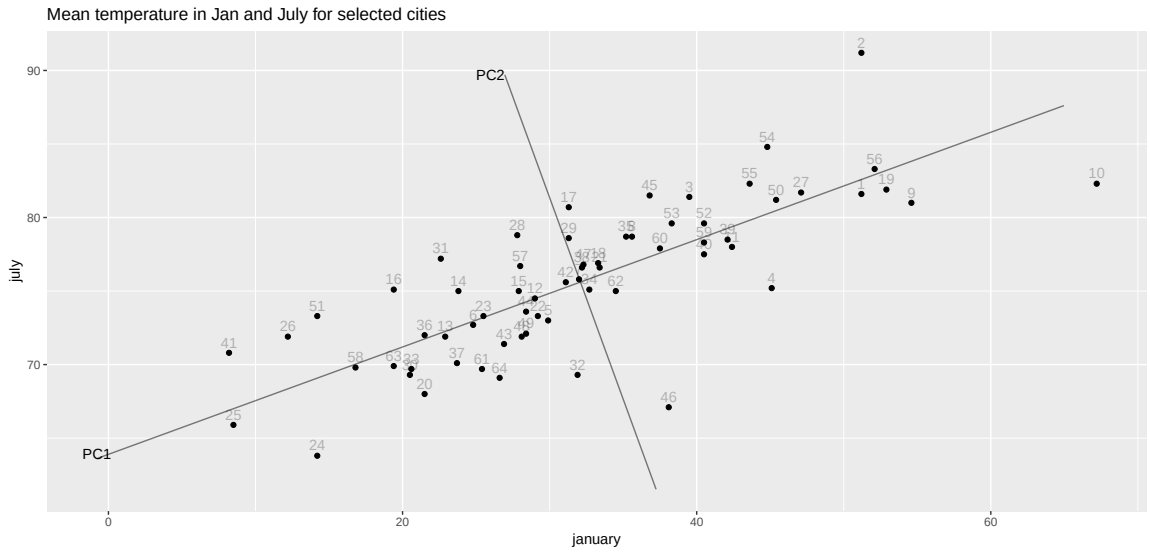
PCA is effectively doing a location shift (to the origin, zero) and a rotation of the data. When the correlation is used for PCA (instead of the covariance), it also performs a scaling so that the resulting PC scores have unit-variance in all directions.

```
# create small data.frame with endpoints of PC lines through data
line.scale <- c(35, 15) # length of PCA lines to draw
# endpoints of lines to draw
temp.pca.line.endpoints <-
  data.frame(PC = c(rep("PC1", 2), rep("PC2", 2))
             ,  x = c(temp.pca$center[1] - line.scale[1] * temp.pca$loadings[1, 1]
                    , temp.pca$center[1] + line.scale[1] * temp.pca$loadings[1, 1]
                    , temp.pca$center[1] - line.scale[2] * temp.pca$loadings[1, 2]
                    , temp.pca$center[1] + line.scale[2] * temp.pca$loadings[1, 2])
             ,  y = c(temp.pca$center[2] - line.scale[1] * temp.pca$loadings[2, 1]
                    , temp.pca$center[2] + line.scale[1] * temp.pca$loadings[2, 1]
                    , temp.pca$center[2] - line.scale[2] * temp.pca$loadings[2, 2]
                    , temp.pca$center[2] + line.scale[2] * temp.pca$loadings[2, 2])
              )
temp.pca.line.endpoints
```

```
##    PC          x        y
## 1 PC1 -0.7833519 63.61121
## 2 PC1 64.9739769 87.61066
## 3 PC2 26.9525727 89.70179
## 4 PC2 37.2380523 61.52008
```

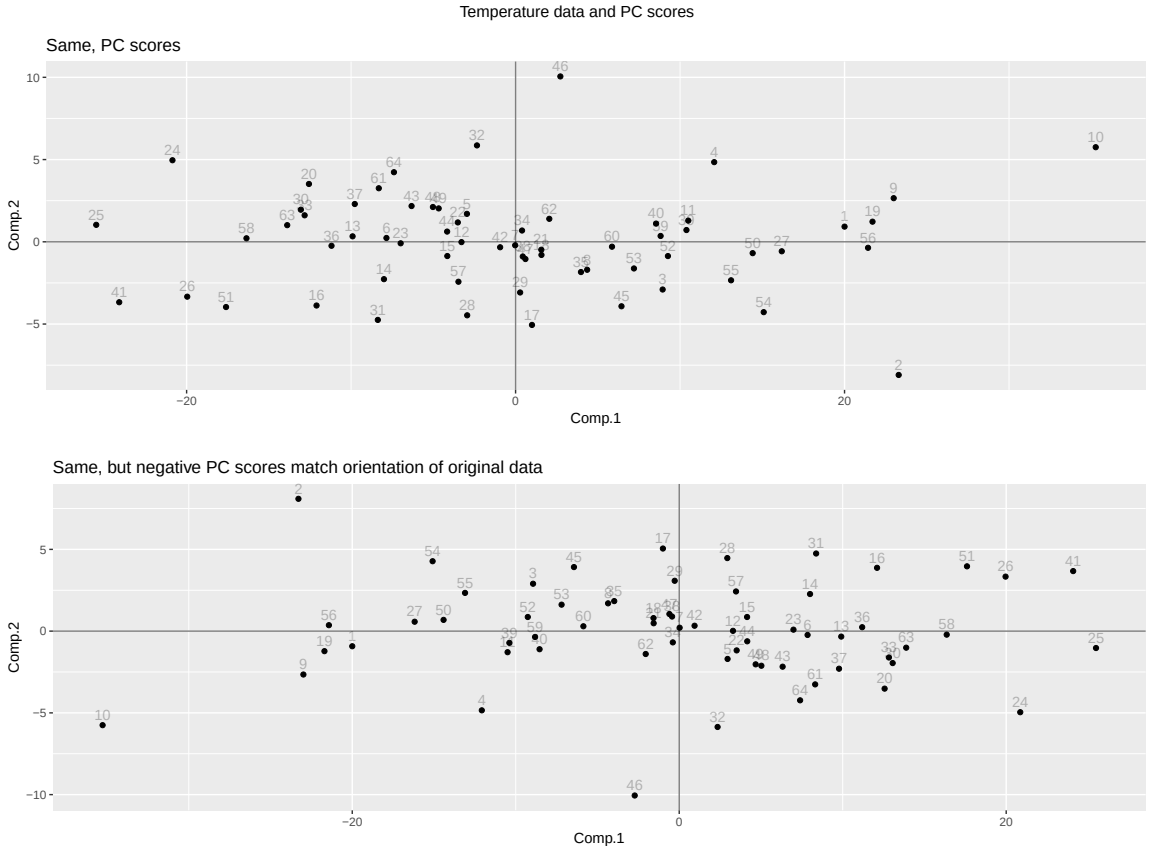
```
# plot original data with PCA vectors overlayed
library(ggplot2)
p1 <- ggplot(temp, aes(x = january, y = july))
p1 <- p1 + geom_point()           # points
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                  # good idea since both are in the same units
p1 <- p1 + geom_text(aes(label = id), vjust = -0.5, alpha = 0.25) # city labels
# plot PC lines
p1 <- p1 + geom_path(data = subset(temp.pca.line.endpoints, PC=="PC1"), aes(x=x, y=y)
                     , alpha=0.5)
p1 <- p1 + geom_path(data = subset(temp.pca.line.endpoints, PC=="PC2"), aes(x=x, y=y)
                     , alpha=0.5)
# label lines
```

```
p1 <- p1 + annotate("text"
                 , x     = temp.pca.line.endpoints$x[1]
                 , y     = temp.pca.line.endpoints$y[1]
                 , label = as.character(temp.pca.line.endpoints$PC[1])
                 , vjust = 0) #, size  = 10)
p1 <- p1 + annotate("text"
                 , x     = temp.pca.line.endpoints$x[3]
                 , y     = temp.pca.line.endpoints$y[3]
                 , label = as.character(temp.pca.line.endpoints$PC[3])
                 , hjust = 1) #, size  = 10)
p1 <- p1 + labs(title = "Mean temperature in Jan and July for selected cities")
print(p1)
```




```
# plot PCA scores (data on PC-scale centered at 0)
library(ggplot2)
p2 <- ggplot(as.data.frame(temp.pca$scores), aes(x = Comp.1, y = Comp.2))
p2 <- p2 + geom_point()           # points
p2 <- p2 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                  # good idea since both are in the same units
p2 <- p2 + geom_text(aes(label = rownames(temp.pca$scores)), vjust = -0.5, alpha = 0.25) # city labels
# plot PC lines
p2 <- p2 + geom_vline(xintercept = 0, alpha=0.5)
p2 <- p2 + geom_hline(yintercept = 0, alpha=0.5)
p2 <- p2 + labs(title = "Same, PC scores")
#print(p2)

# plot PCA scores (data on (negative) PC-scale centered at 0)
library(ggplot2)          # negative temp.pca$scores
p3 <- ggplot(as.data.frame(-temp.pca$scores), aes(x = Comp.1, y = Comp.2))
p3 <- p3 + geom_point()           # points
```

```
p3 <- p3 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                  # good idea since both are in the same units
p3 <- p3 + geom_text(aes(label = rownames(temp.pca$scores)), vjust = -0.5, alpha = 0.25) # city
# plot PC lines
p3 <- p3 + geom_vline(xintercept = 0, alpha=0.5)
p3 <- p3 + geom_hline(yintercept = 0, alpha=0.5)
p3 <- p3 + labs(title = "Same, but negative PC scores match orientation of original data")
#print(p3)

library(gridExtra)
grid.arrange(grobs = list(p2, p3), ncol=1, top="Temperature data and PC scores")
```



Some comments on the output:

1. You can visualize PCA when $p = 2$. In the temperature plot, the direction of maximal variability corresponds to the first PC axis. The PRIN1 score for each city is obtained by projecting the temperature pairs perpendicularly onto this axis. The direction of minimum variation corresponds to the second PC axis, which is perpendicular to the first PC axis. The PRIN2 score for each city is obtained by projecting the temperature pairs onto this axis.
2. The **total variance** is the sum of variances for the monthly temperatures:

$163.38 = 137.18 + 26.20$.

```
# variance of data (on diagonals, covariance of off-diags)
var(temp[,c("january","july")])
```

```
##          january     july
## january 137.1811 46.72910
## july     46.7291 26.20035
```

```
# sum of variance
sum(diag(var(temp[,c("january","july")])))
```

```
## [1] 163.3814
```

```
# variance of PC scores
var(temp.pca$scores)
```

```
##                Comp.1        Comp.2
## Comp.1  1.542358e+02 -1.831125e-15
## Comp.2 -1.831125e-15  9.145635e+00
```

```
# sum is same as original data
sum(diag(var(temp.pca$scores)))
```

```
## [1] 163.3814
```

3. The **eigenvalues of the covariance matrix** are variances for the PCs. The variability of

$$\text{PRIN1} = +0.939 \text{ JAN} + 0.343 \text{ JULY}$$

is 154.236. The variability of

$$\text{PRIN2} = -0.343 \text{ JAN} + 0.939 \text{ JULY}$$

is 9.146. The proportion of the total variability due to PRIN1 is $0.944 = 154.23/163.38$. The proportion of the total variability due to PRIN2 is $0.056 = 9.146/163.38$.

```
# eigenvalues and eigenvectors of covariance matrix give PC variance and loadings
eigen(var(temp[,c("january","july")]))
```

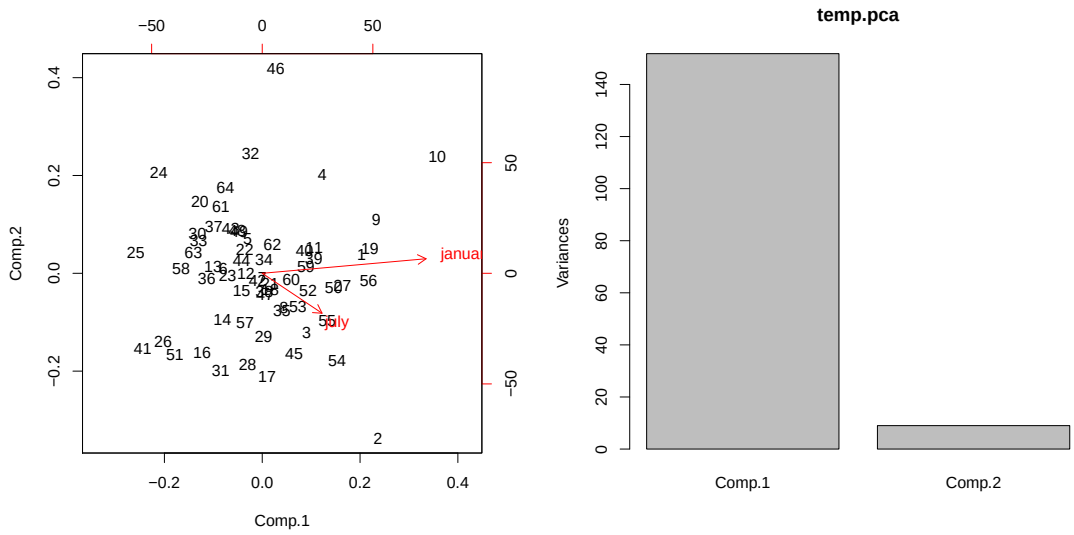
```
## eigen() decomposition
## $values
## [1] 154.235808   9.145635
##
## $vectors
##            [,1]       [,2]
## [1,] -0.9393904  0.3428493
## [2,] -0.3428493 -0.9393904
```

4. Almost all of the variability (94.4%) in the original temperatures is captured by the first PC. The second PC accounts for the remaining 5.6% of the total variability.

5. PRIN1 weights the January temperature about three times the July temperature. This is sensible because PRIN1 maximizes variation among linear combinations of the January and July temperatures. January temperatures are more variable, so they are weighted heavier in this linear combination.
6. The PCs PRIN1 and PRIN2 are standardized to have mean zero. This explains why some PRIN1 scores are negative, even though PRIN1 is a weighted average of the January and July temperatures, each of which is non-negative.

The built-in plots plot the scores and original data directions (biplot) and the screeplot shows the relative variance proportion of all components in decreasing order.

```
# a couple built-in plots
par(mfrow=c(1,2))
biplot(temp.pca)
screeplot(temp.pca)
```



13.2 PCA on Correlation Matrix

The coefficients in the first principal component reflect the relative sizes of the feature variances. The features with large variances have larger coefficients or loadings. This might be considered a problem, because variability is **scale dependent** and the principal component analysis on the raw data does not take scale into account. For example, if you measure height in meters but then change height to centimeters, the variability increases by a factor of 100*100 = 10,000. This might have a dramatic effect on the PCA.

You might prefer to standardize the features when they are measured on different

scales, or when the features have wildly different variances. The features are standardized to have mean zero and variance one by using the Z-score transformation: (Obs − Mean)/Std Dev. The PCA is then performed on the standardized data.

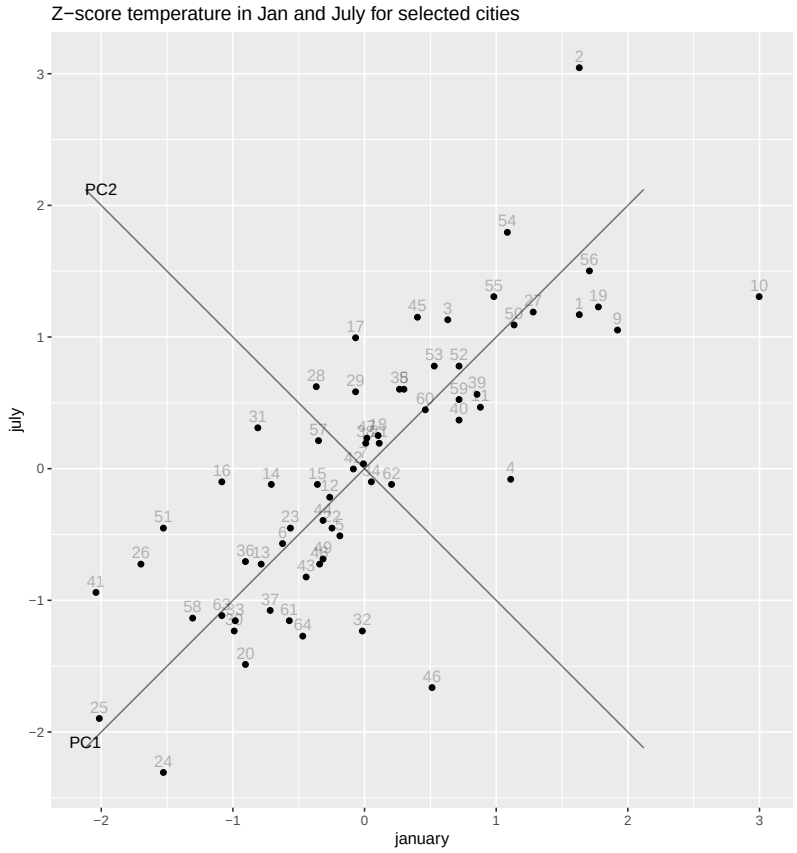
```
temp.z <- temp
# manual z-score
temp.z$january <- (temp.z$january - mean(temp.z$january)) / sd(temp.z$january)
# z-score using R function scale()
temp.z$july <- scale(temp.z$july)

# the manual z-score and scale() match
all.equal(temp.z$january, as.vector(scale(temp.z$january)))
## [1] TRUE
# scale() includes attributes for the mean() and sd() used for z-scoring
str(temp.z)
## 'data.frame': 64 obs. of  4 variables:
##  $ city   : Factor w/ 64 levels "albany","albuquerque",..: 39 48 33 56 21 27 64 62 31 36 ...
##  $ january: num  1.631 1.631 0.632 1.11 -0.187 ...
##  $ july   : num [1:64, 1] 1.1701 3.0456 1.131 -0.0803 -0.5101 ...
##   ..- attr(*, "scaled:center")= num 75.6
##   ..- attr(*, "scaled:scale")= num 5.12
##  $ id     : int  1 2 3 4 5 6 7 8 9 10 ...
head(temp.z)
##          city    january        july id
## 1      mobile  1.6311459  1.17005226  1
## 2     phoenix  1.6311459  3.04555476  2
## 3 little rock  0.6322075  1.13097929  3
## 4  sacramento  1.1103319 -0.08028274  4
## 5      denver -0.1874344 -0.51008540  5
## 6    hartford -0.6228691 -0.56869485  6
# z-scored data has mean 0 and variance 1
colMeans(temp.z[,c("january","july")])
##       january          july
##  1.228943e-16 -1.214842e-15
var(temp.z[,c("january","july")])
##           january      july
## january 1.0000000 0.7794472
## july    0.7794472 1.0000000
# the correlation is used to contruct the PCs
#   (same as covariance for z-scored data)
cor(temp.z[,c("january","july")])
##           january      july
## january 1.0000000 0.7794472
## july    0.7794472 1.0000000
```

```
## Plot z-scored data
temp.z.pca <- princomp( ~ january + july, data = temp.z)
```

```
# create small data.frame with endpoints of PC lines through data
line.scale <- c(3, 3) # length of PCA lines to draw
# endpoints of lines to draw
temp.z.pca.line.endpoints <-
  data.frame(PC = c(rep("PC1", 2), rep("PC2", 2))
           ,  x = c(temp.z.pca$center[1] - line.scale[1] * temp.z.pca$loadings[1, 1]
                  , temp.z.pca$center[1] + line.scale[1] * temp.z.pca$loadings[1, 1]
                  , temp.z.pca$center[1] - line.scale[2] * temp.z.pca$loadings[1, 2]
                  , temp.z.pca$center[1] + line.scale[2] * temp.z.pca$loadings[1, 2])
           ,  y = c(temp.z.pca$center[2] - line.scale[1] * temp.z.pca$loadings[2, 1]
                  , temp.z.pca$center[2] + line.scale[1] * temp.z.pca$loadings[2, 1]
                  , temp.z.pca$center[2] - line.scale[2] * temp.z.pca$loadings[2, 2]
                  , temp.z.pca$center[2] + line.scale[2] * temp.z.pca$loadings[2, 2])
           )
temp.z.pca.line.endpoints
```

```
##    PC        x        y
## 1 PC1 -2.12132 -2.12132
## 2 PC1  2.12132  2.12132
## 3 PC2 -2.12132  2.12132
## 4 PC2  2.12132 -2.12132
```

```
# plot original data with PCA vectors overlayed
library(ggplot2)
p1 <- ggplot(temp.z, aes(x = january, y = july))
p1 <- p1 + geom_point()           # points
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
                                  # good idea since both are in the same units
p1 <- p1 + geom_text(aes(label = id), vjust = -0.5, alpha = 0.25) # city labels
# plot PC lines
p1 <- p1 + geom_path(data = subset(temp.z.pca.line.endpoints, PC=="PC1"), aes(x=x, y=y), alpha=0.5)
p1 <- p1 + geom_path(data = subset(temp.z.pca.line.endpoints, PC=="PC2"), aes(x=x, y=y), alpha=0.5)
# label lines
p1 <- p1 + annotate("text"
                  , x     = temp.z.pca.line.endpoints$x[1]
                  , y     = temp.z.pca.line.endpoints$y[1]
                  , label = as.character(temp.z.pca.line.endpoints$PC[1])
                  , vjust = 0) #, size  = 10)
p1 <- p1 + annotate("text"
                  , x     = temp.z.pca.line.endpoints$x[3]
                  , y     = temp.z.pca.line.endpoints$y[3]
                  , label = as.character(temp.z.pca.line.endpoints$PC[3])
                  , hjust = 0) #, size  = 10)
p1 <- p1 + labs(title = "Z-score temperature in Jan and July for selected cities")
print(p1)
```



The covariance matrix computed from the standardized data is the correlation matrix. Thus, principal components based on the standardized data are computed from the correlation matrix. This is implemented by adding the `cor = TRUE` option on the `princomp()` procedure statement.

```
# perform PCA on correlation matrix
temp.pca2 <- princomp( ~ january + july, data = temp, cor = TRUE)
# standard deviation and proportion of variation for each component
summary(temp.pca2)
```

```
## Importance of components:
##                           Comp.1    Comp.2
## Standard deviation     1.3339592 0.4696305
## Proportion of Variance 0.8897236 0.1102764
## Cumulative Proportion  0.8897236 1.0000000
```

```
# coefficients for PCs
loadings(temp.pca2)
```

```
##
## Loadings:
##         Comp.1 Comp.2
## january  0.707  0.707
## july     0.707 -0.707
```

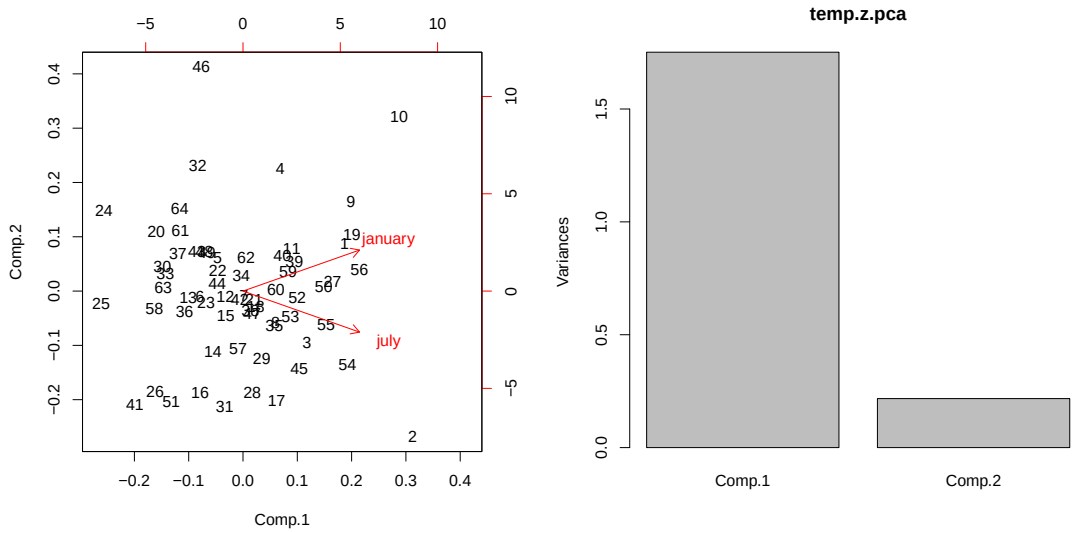
```
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
## Cumulative Var    0.5    1.0
```

```
# scores are coordinates of each observation on PC scale
head(temp.pca2$scores)
```

```
##       Comp.1     Comp.2
## 1  1.9964045  0.3286199
## 2  3.3330689 -1.0080444
## 3  1.2566173 -0.3554730
## 4  0.7341125  0.8485470
## 5 -0.4971200  0.2299523
## 6 -0.8492236 -0.0386098
```

This plot is the same except for the top/right scale around the biplot and the variance scale on the screeplot.

```
# a couple built-in plots
par(mfrow=c(1,2))
biplot(temp.z.pca)
screeplot(temp.z.pca)
```



The standardized features are dimensionless, so the PCs are not influenced by the original units of measure, nor are they affected by the variability in the features. The only important factor is the correlation between the features, which is not changed by standardization.

The PCs from the correlation matrix are

$$\text{PRIN1} = +0.707\ \text{JAN} + 0.707\ \text{JULY}$$

and

$$\text{PRIN2} = -0.707 \text{ JAN} + 0.707 \text{ JULY}.$$

PCA is an exploratory tool, so neither a PCA on the covariance matrix nor a PCA on the correlation matrix is always the "right" method. I often do both and see which analysis is more informative.

13.3 Interpreting Principal Components

You should try to interpret the linear combinations created by multivariate techniques. The interpretations are usually non-trivial, and some degree of creativity is involved.

The **coefficients** or **loadings** in a principal component reflect the **relative** contribution of the features to the linear combination. Most researchers focus more on the **signs** of the coefficents than on the **magnitude** of the coefficients. The principal components are then interpreted as weighted averages or comparisons of weighted averages.

A **weighted average** is a linear combination of features with non-negative loadings. The simplest case of a weighted average is an arithmetic average, where each feature has the same coefficient.

The difference $Z = X - Y$ is a **comparison** of X and Y. The sign and magnitude of Z indicates which of X and Y is larger, and by how much. To see this, note that $Z = 0$ if and only if $X = Y$, whereas $Z < 0$ when $X < Y$ and $Z > 0$ when $X > Y$.

In the temperature data, PRIN1 is a weighted average of January and July temperatures (signs of the loadings: JAN is + and JULY is +):

$$\text{PRIN1} = +0.94 \text{ JAN} + 0.34 \text{ JULY}.$$

PRIN2 is a comparison of January and July temperatures (signs of the loadings: JAN is − and JULY is +):

$$\text{PRIN2} = -0.34 \text{ JAN} + 0.94 \text{ JULY}.$$

Principal components often have positive and negative loadings when $p \geq 3$. To interpret the components, group the features with + and − signs together and then interpret the linear combination as a comparison of weighted averages.

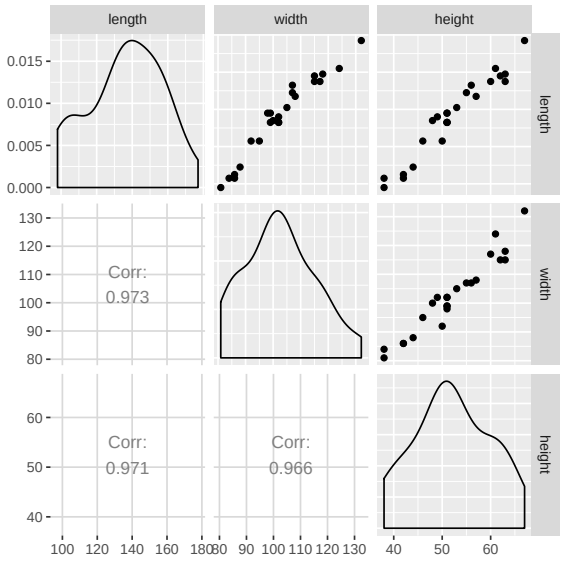
You can often simplify the interpretation of principal components by **mentally eliminating** features from the linear combination that have relatively small (in magnitude) loadings or coefficients. This strategy does not carry over to all multivariate analyses, so I will be careful about this issue when necessary.

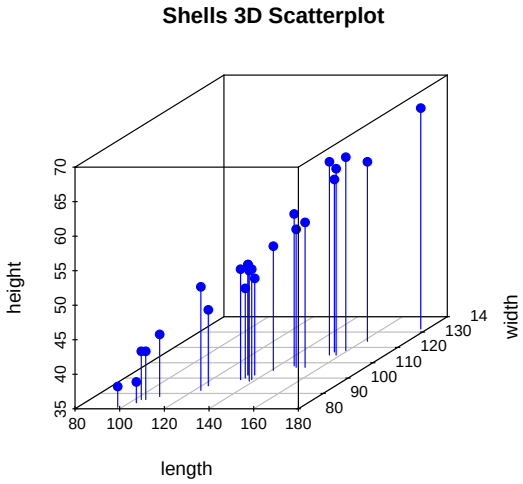
13.4 Example: Painted turtle shells

Jolicouer and Mosimann gave the length, width, and height in mm of the carapace (shell) for a sample of 24 female painted turtles. I perform a PCA on the original data and on the standardized data.

```
#### Example: Painted turtle shells
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch13_shells.dat"
shells <- read.table(fn.data, header = TRUE)
str(shells)
## 'data.frame': 24 obs. of  3 variables:
##  $ length: int  98 103 103 105 109 123 123 133 133 133 ...
##  $ width : int  81 84 86 86 88 92 95 99 102 102 ...
##  $ height: int  38 38 42 42 44 50 46 51 51 51 ...
head(shells)
##   length width height
## 1     98    81     38
## 2    103    84     38
## 3    103    86     42
## 4    105    86     42
## 5    109    88     44
## 6    123    92     50
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
# put scatterplots on top so y axis is vertical
p <- ggpairs(shells, upper = list(continuous = "points")
                   , lower = list(continuous = "cor")
                   , progress=FALSE
             )
print(p)

  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)

## 3D scatterplot
library(scatterplot3d)
par(mfrow=c(1,1))
with(shells, {
  scatterplot3d(x=length
              , y=width
              , z=height
              , main="Shells 3D Scatterplot"
              , type = "h" # lines to the horizontal xy-plane
              , color="blue", pch=19, # filled blue circles
              #, highlight.3d = TRUE  # makes color change with z-axis value
```

```
            )
  })

#### For a rotatable 3D plot, use plot3d() from the rgl library
# ##   This uses the R version of the OpenGL (Open Graphics Library)
# library(rgl)
# with(shells, { plot3d(x = length, y = width, z = height) })
```





13.4.1 PCA on shells covariance matrix

The plots show that the shell measurements are strongly positively correlated, which is not surprising. Let us perform a PCA on the covariance matrix and interpret the results.

```
# perform PCA on covariance matrix
shells.pca <- princomp( ~ length + width + height, data = shells)
# standard deviation and proportion of variation for each component
summary(shells.pca)
## Importance of components:
##                             Comp.1      Comp.2      Comp.3
## Standard deviation     25.4970668 2.547081962 1.653745717
## Proportion of Variance  0.9860122 0.009839832 0.004148005
## Cumulative Proportion   0.9860122 0.995851995 1.000000000
# coefficients for PCs
loadings(shells.pca)
##
## Loadings:
##        Comp.1 Comp.2 Comp.3
## length  0.814  0.555  0.172
```

```
## width   0.496 -0.818  0.291
## height  0.302 -0.151 -0.941
##
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

The three principal components from the raw data are given below. Length and width are grouped in PRIN3 because they have negative loadings.

$$\begin{aligned} \text{PRIN1} &= 0.81\ \text{Length} + 0.50\ \text{Width} + 0.30\ \text{Height} \\ \text{PRIN2} &= -0.55\ \text{Length} + (0.82\ \text{Width} + 0.15\ \text{Height}) \\ \text{PRIN3} &= -(0.17\ \text{Length} + 0.29\ \text{Width}) + 0.94\ \text{Height}. \end{aligned}$$

PRIN1 is a weighted average of the carapace measurements, and can be viewed as an overall measure of shell size. Jolicouer and Mosimann interpreted the second and third principal components as measures of **shape**, for they appear to be a comparison of length with an average of width and height, and a comparison of height with length and width, respectively.

Jolicouer and Mosimann argue that the size of female turtle shells can be characterized by PRIN1 with little loss of information because this linear combination accounts for 98.6% of the total variability in the measurements. The form of PRIN1 makes sense conceptually. The carapace measurements are positively correlated with each other, so larger lengths tend to occur with larger widths and heights. The primary way the shells vary is with regards to their overall size, as measured by a weighted average of length, width, and height.

Question: Can PRIN2 and PRIN3, which have relatively little variation, be used in any meaningful way? To think about this, suppose the variability in PRIN2 and PRIN3 was zero.

13.4.2 PCA on shells correlation matrix

For the analysis on the correlation matrix, add the `cor = TRUE` option.

```
# perform PCA on correlation matrix
shells.pca <- princomp( ~ length + width + height, data = shells, cor = TRUE)
# standard deviation and proportion of variation for each component
summary(shells.pca)
```

```
## Importance of components:
##                           Comp.1    Comp.2      Comp.3
## Standard deviation     1.714584 0.1853043 0.160820482
## Proportion of Variance 0.979933 0.0114459 0.008621076
```

```
## Cumulative Proportion  0.979933 0.9913789 1.000000000
# coefficients for PCs
loadings(shells.pca)
##
## Loadings:
##        Comp.1 Comp.2 Comp.3
## length  0.578  0.137  0.804
## width   0.577  0.628 -0.522
## height  0.577 -0.766 -0.284
##
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

The three principal components for the standardized data are

$$\begin{aligned} \text{PRIN1} &= 0.58\ \text{Length} + 0.58\ \text{Width} + 0.58\ \text{Height} \\ \text{PRIN2} &= -(0.14\ \text{Length} + 0.63\ \text{Width}) + 0.77\ \text{Height} \\ \text{PRIN3} &= -0.80\ \text{Length} + (0.52\ \text{Width} + 0.28\ \text{Height}). \end{aligned}$$

The first principal component accounts for 98% of the total variability in the standardized data. The total variability for correlation is always the number p of features because it is the sum of the variances. Here, $p = 3$. Little loss of information is obtained by summarizing the standardized data using PRIN1, which is essentially an average of length, width, and height. PRIN2 and PRIN3 are measures of shape.

The loadings in the first principal component are approximately equal because the correlations between pairs of features are almost identical. The standardized features are essentially interchangeable with regards to the construction of the first principal component, so they must be weighted similarly. True, but not obvious.

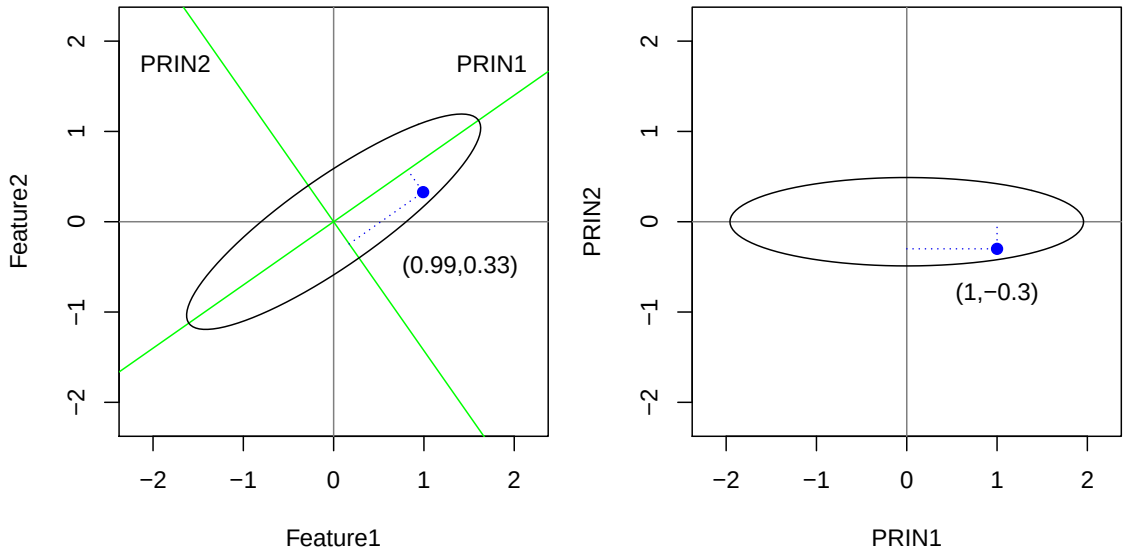
13.5 Why is PCA a Sensible Variable Reduction Technique?

My description of PCA provides little insight into why principal components are reasonable summaries for **variable reduction**. Specifically, why does it make sense for researchers to consider the linear combination of the original features with the **largest variance** as the best single variable summary of the data?

There is an alternative description of PCA that provides more insight into this issue. For simplicity, consider the following data plot of two features, and the implied principal components. The PC scores for an observation are obtained by projecting

the feature scores onto the axes of maximal and minimal variation, and then rotating the axes appropriately.

One can show mathematically that PRIN1 is the best (in some sense) linear combination of the two features to predict the original two features simultaneously. Intuitively, this is plausible. In a PCA, you know the direction for the axis of maximal variation. Given the value of PRIN1, you get a good prediction of the original feature scores by moving PRIN1 units along the axis of maximal variation in the feature space.



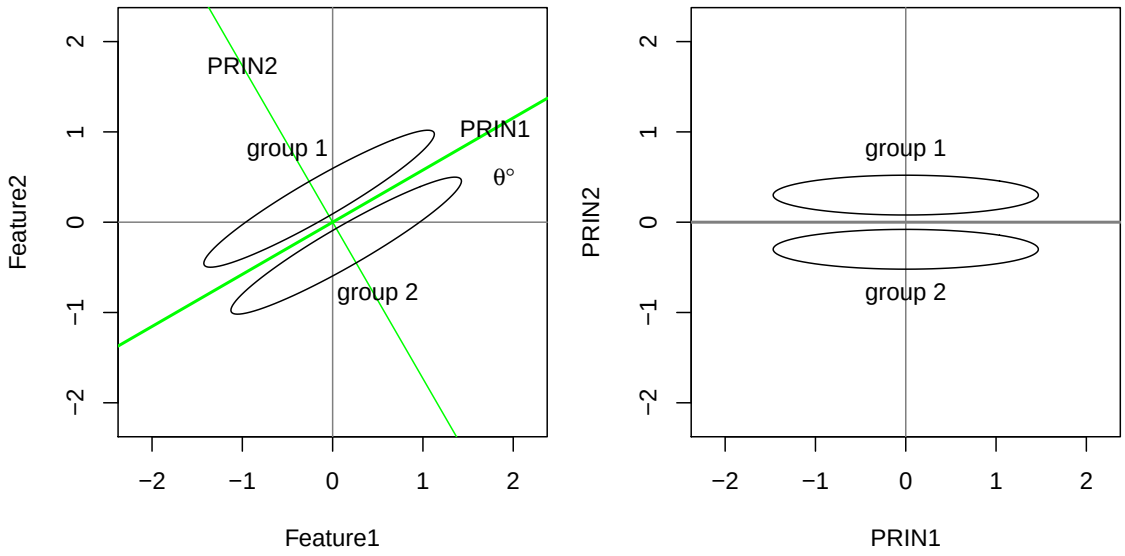
The LS line from regressing feature 2 on feature 1 gives the best prediction for feature 2 scores when the feature 1 score is known. Similarly, the LS line from regressing feature 1 on feature 2 gives the best prediction for feature 1 scores when the feature 2 score is known. PRIN1 is the best linear combination of features 1 and 2 to predict both features *simultaneously*. Note that feature 1 and feature 2 are linear combinations as well!

This idea generalizes. The first k principal components give the best simultaneous prediction of the original p features, among all possible choices of k uncorrelated unit-length linear combinations of the features. Prediction of the original features improves as additional components are added, but the improvement is slight when the added principal components have little variability. Thus, summarizing the data using the principal components with maximum variation is a sensible strategy for data reduction.

13.5.1 A Warning on Using PCA as a Variable Reduction Technique

Some researchers view PCA as a "catchall" technique for reducing the number of variables that need to be considered in an analysis. They will replace the original variables with a small number of principal components that explain most of the variation in the original data and proceed with an analysis on the principal components.

This strategy is not always sensible, especially if a primary interest in the analysis is a comparison of heterogeneous groups. If the group structure was ignored in the PCA analysis, then the linear combinations retained by the researcher may contain little information for distinguishing among groups. For example, consider the following data plot on two features and two groups, and the implied principal components.

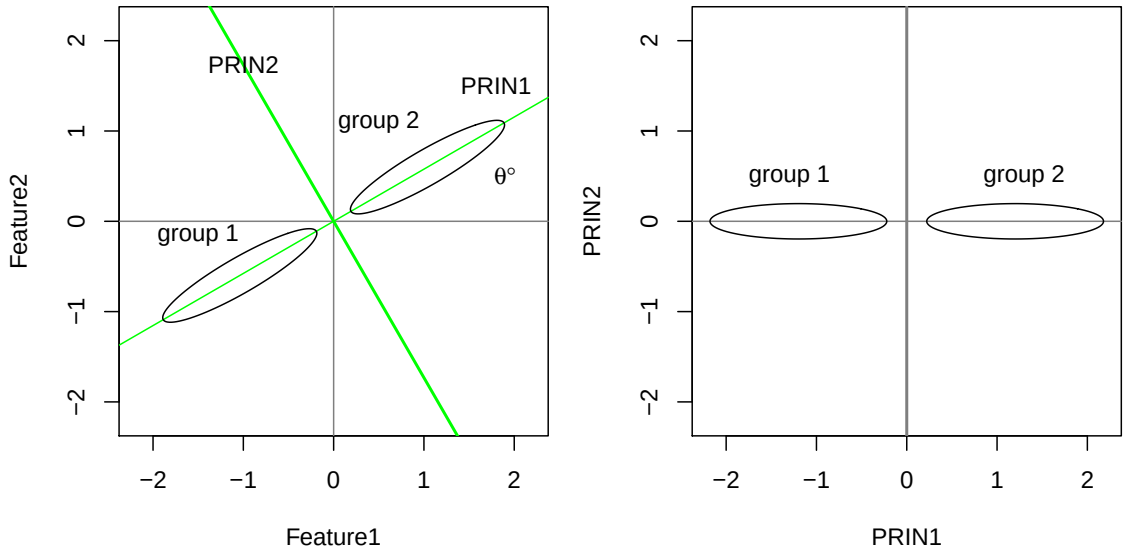


Although PRIN1 explains most of the variability in the two features (ignoring the groups), little of the total variation is due to group differences. If the researcher reduced the two features to the first principal component, he would be throwing away most of the information for distinguishing between the groups. PRIN2 accounts for little of the total variation in the features, but most of the variation in PRIN2 is due to group differences.

If a comparison of the two groups was the primary interest, then the researcher should use discriminant analysis instead. Although there is little gained by reducing two variables to one, this principle always applies in multivariate problems. In

discriminant analysis, a stepwise selection of variables can be implemented to eliminate features that have no information for distinguishing among groups. This is data reduction as it should be practiced — with a final goal in mind.

Variable reduction using PCA followed by group comparisons might be fruitful if you are fortunate enough to have the directions with large variation correspond to the directions of group differences. For example, in the plot below, the first principal component is a linear combination of the features that distinguishes between the groups. A comparison of the groups based on the first principal component will lead to similar conclusions as a discriminant analysis.

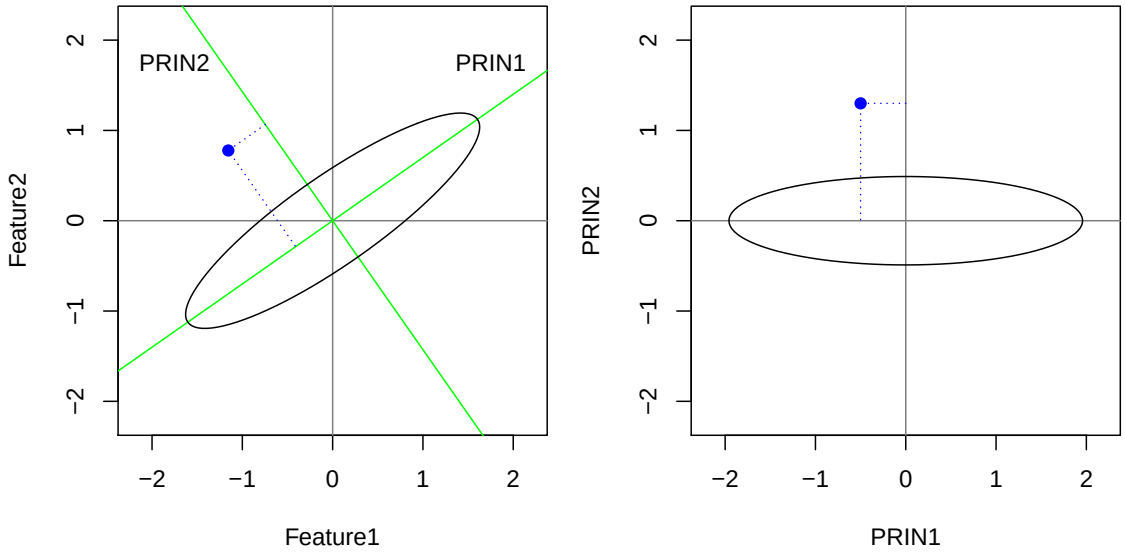


There is nothing unreasonable about using principal components in a comparison of groups, provided you recognize that the principal component scores with the largest variability need not be informative for group comparisons!

In summary, PCA should be used to summarize the variation **within a homogeneous group**, and should not, in general, be used as a data reduction tool prior to a comparison across groups. The same concern applies to using PCA for identifying groups of similar objects (use cluster analysis instead), or when **factor analysis** is used prior to a group comparison.

13.5.2 PCA is Used for Multivariate Outlier Detection

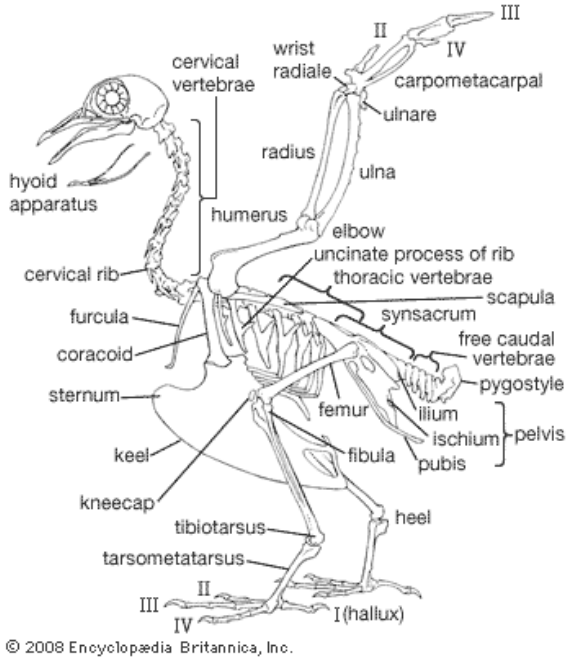
An outlier in a multidimensional data set has atypical readings on one or on several features *simultaneously*. These observations can often be found in univariate plots of the lead principal component scores, even when outliers are not extreme in any individual feature.



13.6 Example: Sparrows, for Class Discussion

After a severe storm in 1898, a number of sparrows were taken to the biological laboratory at the University of Rhode Island. H. Bumbus[1] measured several morphological characteristics on each bird. The data here correspond to five measurements on a sample of 49 females. The measurements are the total length, alar extent, beak-head length, humerus length, and length of keel of sternum.

[1]Bumpus, Hermon C. 1898. Eleventh lecture. The elimination of the unfit as illustrated by the introduced sparrow, Passer domesticus. (A fourth contribution to the study of variation.) Biol. Lectures: Woods Hole Marine Biological Laboratory, 209–225.

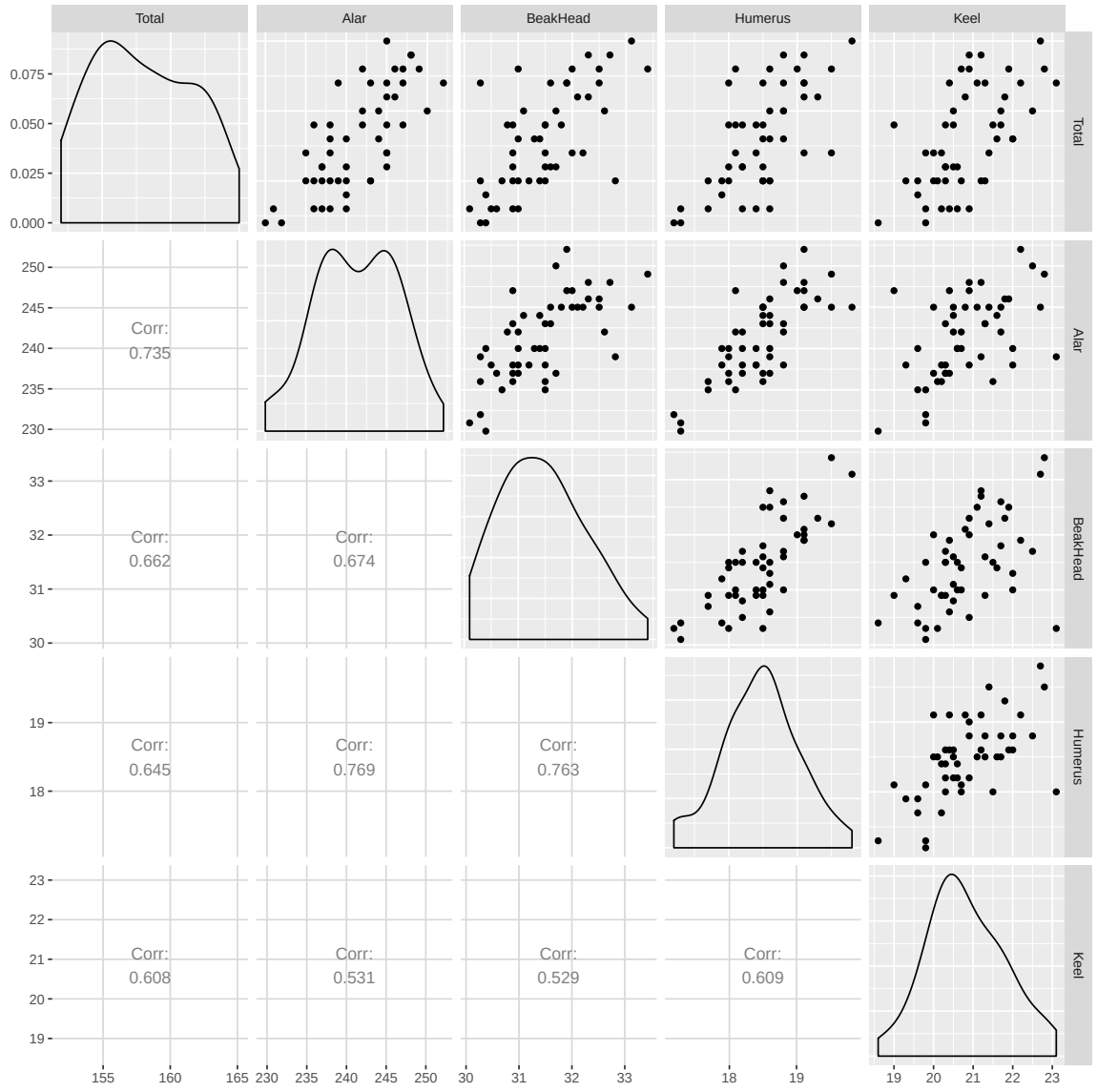


http://media-2.web.britannica.com/eb-media/46/51946-004-D003BC49.gif

Let us look at the output, paying careful attention to the interpretations of the principal components (zeroing out small loadings). How many components seem sufficient to capture the total variation in the morphological measurements?

```
#### Example: Sparrows
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch13_sparrows.dat"
sparrows <- read.table(fn.data, header = TRUE)
str(sparrows)
## 'data.frame': 49 obs. of  5 variables:
##  $ Total   : int  156 153 155 157 164 158 161 157 158 155 ...
##  $ Alar    : int  245 240 243 238 248 240 246 235 244 236 ...
##  $ BeakHead: num  31.6 31 31.5 30.9 32.7 31.3 32.3 31.5 31.4 30.3 ...
##  $ Humerus : num  18.5 18.4 18.6 18.4 19.1 18.6 19.3 18.1 18.5 18.5 ...
##  $ Keel    : num  20.5 20.6 20.3 20.2 21.2 22 21.8 19.8 21.6 20.1 ...
head(sparrows)
##   Total Alar BeakHead Humerus Keel
## 1   156  245     31.6    18.5 20.5
## 2   153  240     31.0    18.4 20.6
## 3   155  243     31.5    18.6 20.3
## 4   157  238     30.9    18.4 20.2
## 5   164  248     32.7    19.1 21.2
## 6   158  240     31.3    18.6 22.0
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
```

```
# put scatterplots on top so y axis is vertical
p <- ggpairs(sparrows, upper = list(continuous = "points")
                     , lower = list(continuous = "cor")
                     , progress=FALSE
             )
print(p)


  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```



```r
# perform PCA on covariance matrix
sparrows.pca <- princomp( ~ Total + Alar + BeakHead + Humerus + Keel
                        , data = sparrows)
# standard deviation and proportion of variation for each component
summary(sparrows.pca)
```

```
## Importance of components:
##                            Comp.1    Comp.2     Comp.3      Comp.4
## Standard deviation     5.8828991 2.1280701 0.78468836 0.552957190
## Proportion of Variance 0.8623099 0.1128372 0.01534175 0.007618397
## Cumulative Proportion  0.8623099 0.9751472 0.99048891 0.998107311
##                             Comp.5
## Standard deviation     0.275612798
## Proportion of Variance 0.001892689
## Cumulative Proportion  1.000000000
```

```r
# coefficients for PCs
# loadings(sparrows.pca)  # print method for loadings() uses cutoff = 0.1 by default
print(loadings(sparrows.pca), cutoff = 0) # to show all values
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## Total     0.536  0.828  0.157  0.039  0.018
## Alar      0.829 -0.551  0.058  0.069 -0.040
## BeakHead  0.096  0.034 -0.241 -0.897 -0.357
## Humerus   0.074 -0.015 -0.205 -0.306  0.927
## Keel      0.101  0.100 -0.934  0.310 -0.110
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0
```

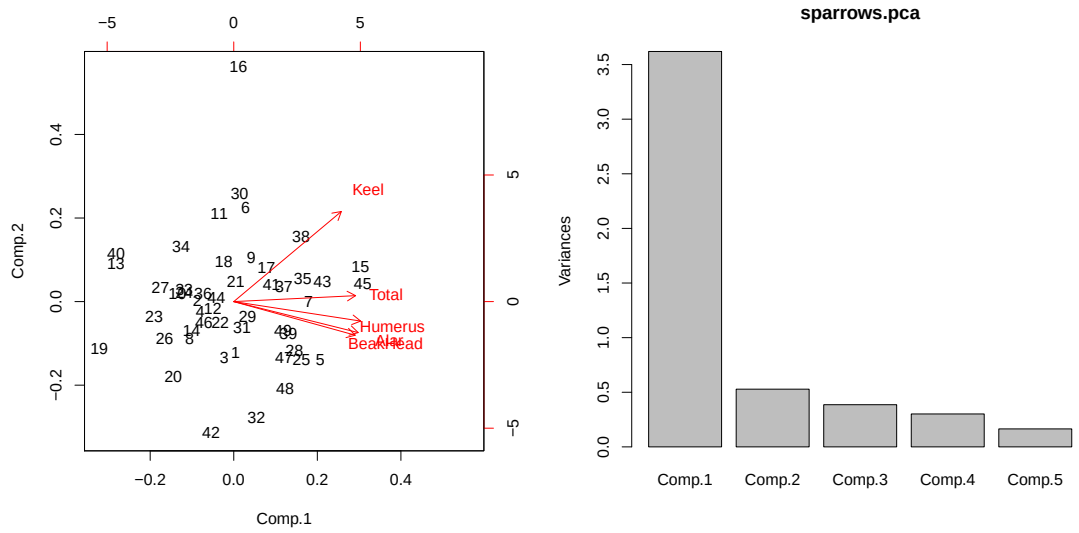
```r
# perform PCA on correlation matrix
sparrows.pca <- princomp( ~ Total + Alar + BeakHead + Humerus + Keel
                        , data = sparrows, cor = TRUE)
# standard deviation and proportion of variation for each component
summary(sparrows.pca)
```

```
## Importance of components:
##                            Comp.1    Comp.2     Comp.3     Comp.4
## Standard deviation     1.9025587 0.7267974 0.62139498 0.54902221
## Proportion of Variance 0.7239459 0.1056469 0.07722634 0.06028508
## Cumulative Proportion  0.7239459 0.8295928 0.90681917 0.96710425
##                             Comp.5
## Standard deviation     0.40555980
## Proportion of Variance 0.03289575
## Cumulative Proportion  1.00000000
```

```r
# coefficients for PCs
#loadings(sparrows.pca)
print(loadings(sparrows.pca), cutoff = 0) # to show all values
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## Total     0.452  0.058  0.689  0.422  0.375
## Alar      0.461 -0.301  0.345 -0.545 -0.530
## BeakHead  0.450 -0.326 -0.453  0.607 -0.342
## Humerus   0.470 -0.189 -0.409 -0.390  0.651
## Keel      0.399  0.874 -0.184 -0.073 -0.194
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0
```

```
# a couple built-in plots
par(mfrow=c(1,2))
biplot(sparrows.pca)
screeplot(sparrows.pca)
```



13.7 PCA for Variable Reduction in Regression

I will outline an analysis where PCA was used to create predictors in a regression model. The data were selected from the Berkeley Guidance Study, a longitudinal monitoring of children born in Berkeley, California, between 1928 and 1929. The variables selected from the study are

ID an identification number,

WT2 weight at age 2 in kg,

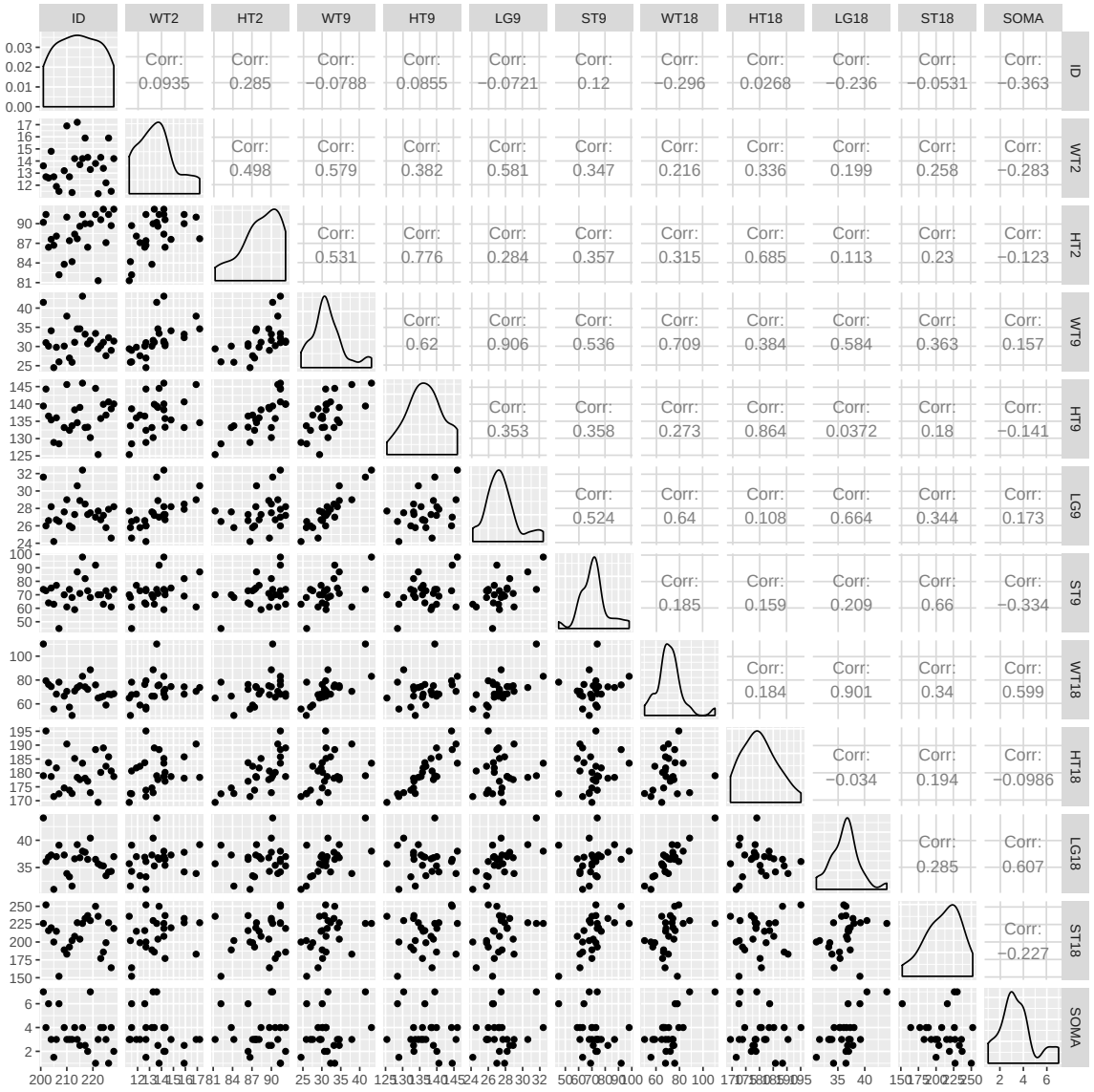
HT2 height at age 2 in cm,
WT9 weight at age 9,
HT9 height at age 9,
LG9 leg circumference at age 9 in cm,
ST9 a composite measure of strength at age 9 (higher is stronger),
WT18 weight at age 18,
HT18 height at age 18,
LG18 leg circumference at age 18,
ST18 a composite measure of strength at age 18, and
SOMA somatotype on a 7-point scale, as a measure of fatness (1=slender to 7=fat) determined using a photo taken at age 18.

Data on 26 boys are given below.

We are interested in building a regression model to predict somatotype (SOMA) from the other variables.

```
#### Example: BGS (Berkeley Guidance Study)
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch13_bgs.dat"
bgs <- read.table(fn.data, header = TRUE)
str(bgs)
## 'data.frame': 26 obs. of  12 variables:
##  $ ID  : int  201 202 203 204 205 206 207 209 210 211 ...
##  $ WT2 : num  13.6 12.7 12.6 14.8 12.7 11.9 11.5 13.2 16.9 12.7 ...
##  $ HT2 : num  90.2 91.4 86.4 87.6 86.7 88.1 82.2 83.8 91 87.4 ...
##  $ WT9 : num  41.5 31 30.1 34.1 24.5 29.8 26 30.1 37.9 27 ...
##  $ HT9 : num  139 144 136 135 129 ...
##  $ LG9 : num  31.6 26 26.6 28.2 24.2 26.7 26.5 27.6 29 26 ...
##  $ ST9 : int  74 73 64 75 63 77 45 70 61 74 ...
##  $ WT18: num  110.2 79.4 76.3 74.5 55.7 ...
##  $ HT18: num  179 195 184 179 172 ...
##  $ LG18: num  44.1 36.1 36.9 37.3 31 37 39.1 37.3 33.9 33.3 ...
##  $ ST18: int  226 252 216 220 200 215 152 189 183 193 ...
##  $ SOMA: num  7 4 6 3 1.5 3 6 4 3 3 ...
head(bgs)
##    ID  WT2  HT2  WT9   HT9  LG9 ST9  WT18  HT18 LG18 ST18 SOMA
## 1 201 13.6 90.2 41.5 139.4 31.6  74 110.2 179.0 44.1  226  7.0
## 2 202 12.7 91.4 31.0 144.3 26.0  73  79.4 195.1 36.1  252  4.0
## 3 203 12.6 86.4 30.1 136.5 26.6  64  76.3 183.7 36.9  216  6.0
## 4 204 14.8 87.6 34.1 135.4 28.2  75  74.5 178.7 37.3  220  3.0
## 5 205 12.7 86.7 24.5 128.9 24.2  63  55.7 171.5 31.0  200  1.5
## 6 206 11.9 88.1 29.8 136.0 26.7  77  68.2 181.8 37.0  215  3.0
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
p <- ggpairs(bgs, progress=FALSE)
print(p)
```

```
# detach package after use so reshape2 works (old reshape (v.1) conflicts)
#detach("package:GGally", unload=TRUE)
#detach("package:reshape", unload=TRUE)
```

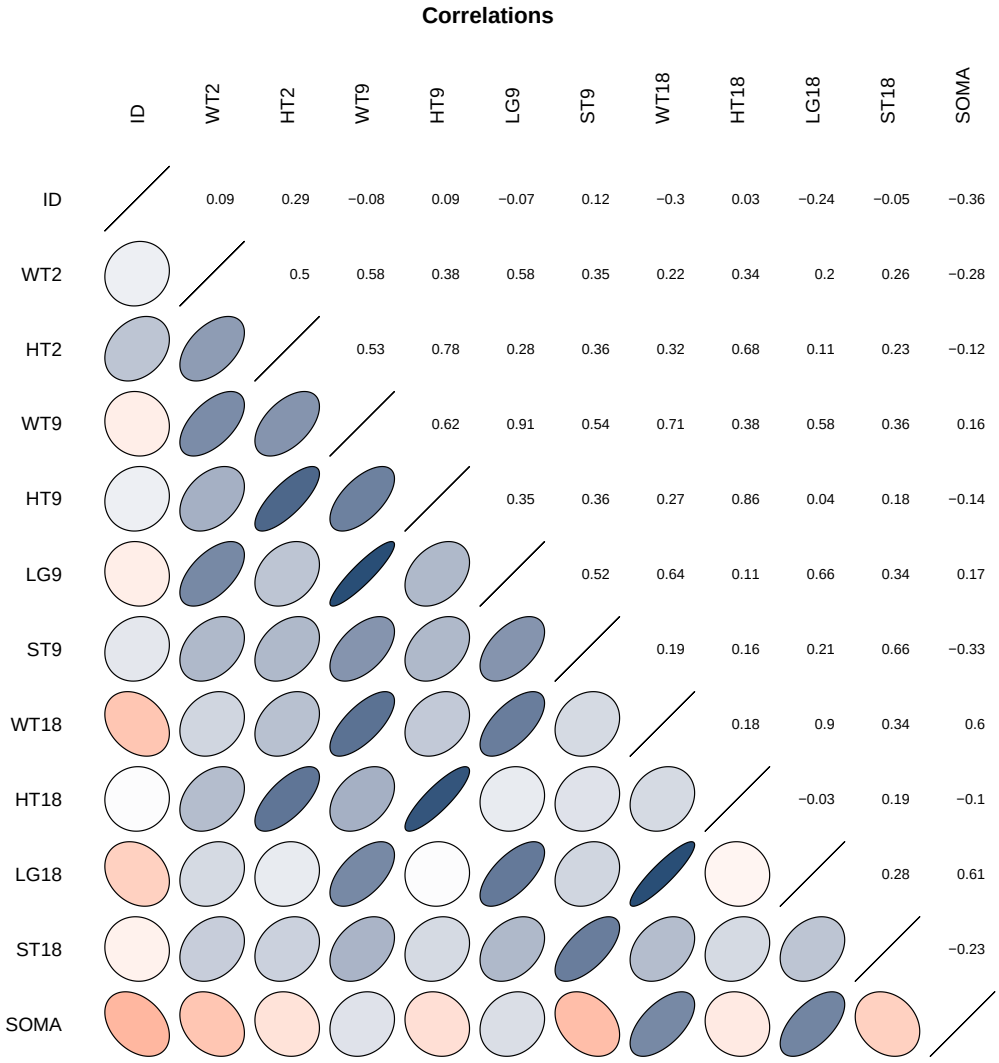


As an aside, there are other ways to visualize the linear relationships more quickly. The `ellipse` library has a function `plotcorr()`, though it's output is less than ideal. An improvement has been made with an updated version[2] of the `plotcorr()` function.

```
## my.plotcorr example, see associated R file for my.plotcorr() function code
```

[2] `http://hlplab.wordpress.com/2012/03/20/correlation-plot-matrices-using-the-ellipse-library/`

```
# calculate correlations to plot
corr.bgs <- cor(bgs)

# Change the column and row names for clarity
colnames(corr.bgs) = colnames(bgs)
rownames(corr.bgs) = colnames(corr.bgs)

# set colors to use (blue positive, red negative)
colsc=c(rgb(241, 54, 23, maxColorValue=255), 'white', rgb(0, 61, 104, maxColorValue=255))
colramp = colorRampPalette(colsc, space='Lab')
colors = colramp(100)

# plot correlations, colored ellipses on lower diagonal, numerical correlations on upper
my.plotcorr(corr.bgs, col=colors[((corr.bgs + 1)/2) * 100]
          , diag='ellipse', upper.panel="number", mar=c(0,2,0,0), main='Correlations')
```



It is reasonable to expect that the characteristics measured over time, for example HT2, HT9, and HT18 are strongly correlated. Evidence supporting this hypothesis is given in the following output, which summarizes correlations within subsets of the predictors. Two of the three subsets include measures over time on the same characteristic.

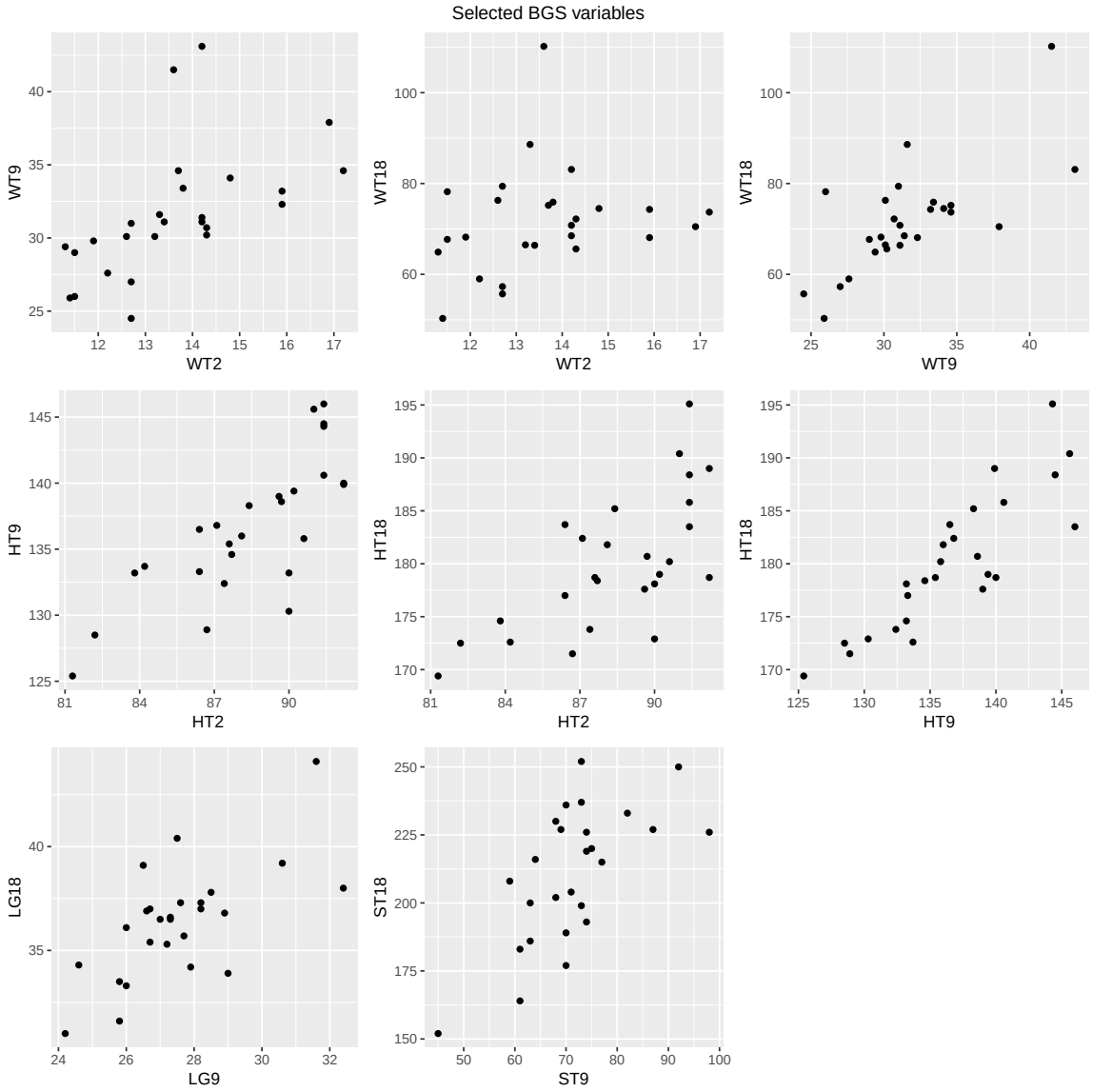
```
cor(bgs[,c("WT2", "WT9", "WT18")])
##            WT2       WT9      WT18
## WT2  1.0000000 0.5792217 0.2158735
## WT9  0.5792217 1.0000000 0.7089029
## WT18 0.2158735 0.7089029 1.0000000
cor(bgs[,c("HT2", "HT9", "HT18")])
```

```
##            HT2       HT9      HT18
## HT2  1.0000000 0.7758332 0.6847144
## HT9  0.7758332 1.0000000 0.8644624
## HT18 0.6847144 0.8644624 1.0000000
```

```
cor(bgs[,c("LG9", "LG18", "ST9", "ST18")])
```

```
##            LG9      LG18       ST9      ST18
## LG9  1.0000000 0.6644753 0.5239476 0.3440756
## LG18 0.6644753 1.0000000 0.2085289 0.2845414
## ST9  0.5239476 0.2085289 1.0000000 0.6595947
## ST18 0.3440756 0.2845414 0.6595947 1.0000000
```

```
library(ggplot2)
p1 <- ggplot(bgs, aes(x = WT2 , y = WT9 )) + geom_point()
p2 <- ggplot(bgs, aes(x = WT2 , y = WT18)) + geom_point()
p3 <- ggplot(bgs, aes(x = WT9 , y = WT18)) + geom_point()
p4 <- ggplot(bgs, aes(x = HT2 , y = HT9 )) + geom_point()
p5 <- ggplot(bgs, aes(x = HT2 , y = HT18)) + geom_point()
p6 <- ggplot(bgs, aes(x = HT9 , y = HT18)) + geom_point()
p7 <- ggplot(bgs, aes(x = LG9 , y = LG18)) + geom_point()
p8 <- ggplot(bgs, aes(x = ST9 , y = ST18)) + geom_point()

library(gridExtra)
grid.arrange(grobs = list(p1, p2, p3, p4, p5, p6, p7, p8), ncol=3, top="Selected BGS variables")
```



Strong correlation among predictors can cause collinearity problems in regression. The presence of collinearity makes the interpretation of regression effects more difficult, and can reek havoc with the numerical stability of certain algorithms used to compute least squares summaries. A natural way to avoid collinearity and improve interpretation of regression effects is to use uncorrelated linear combinations of the original predictors, such as principal components, to build a model.

The interpretation of regression effects changes when linear combinations of predictors are used in place of the original predictors. However, two models will be equivalent in the sense of giving identical fitted values when p linearly independent combinations of p predictors are used. The fitted model can be expressed in terms of

the original predictors, or in terms of the linear combinations. Similarly, standardizing the original predictors does not change the significance of individual regression effects nor does it change the interpretation of the model.

A reasonable strategy with the Berkeley data might be to find principal components for the three height variables separately, the three weights separately, and so on. It may not make sense to combine the four different types of measures (HT, WT, ST, and LG) together because the resulting linear combinations would likely be uninterpretable.

Output from a PCA on the four sets of standardized measures is given below. Two sets (ST9, ST18) and (LG9, LG18) have two predictors. The PCs on the strength and leg measures are essentially the sum and difference between the two standardized features. The loadings have magnitude $0.707 = (1/2)^{1/2}$ to satisfy the unit-length restriction.

Here are some comments on how I might use the PCA to build a regression model to predict somatotype. First, I would not necessarily use the given PCS, but would instead use interpretable linear combinations that were nearly identical to the PCs. For example, the first principal component of the heights is roughly an unweighted average of the weights at ages 2, 9, and 18. I would use (WT2+WT9+WT18)/3 instead. The overall sum of squared loadings is not important in the regression analysis. Only the relative sizes are important. Following this idea, what linear combinations of the heights are reasonable?

Second, you should not assume that principal components with low variance are unimportant for predicting somatotype. Recall our earlier discussion on potential problems with ignoring low variability components when group comparisons are the primary interest. The same problem is possible here.

Feel free to explore these ideas at your leisure.

```
# WT
bgsWT.pca <- princomp( ~ WT2 + WT9 + WT18, data = bgs, cor = TRUE)
summary(bgsWT.pca)
## Importance of components:
##                           Comp.1    Comp.2     Comp.3
## Standard deviation     1.4241276 0.8882456 0.42764499
## Proportion of Variance 0.6760465 0.2629934 0.06096008
## Cumulative Proportion  0.6760465 0.9390399 1.00000000
print(loadings(bgsWT.pca), cutoff = 0)
##
## Loadings:
##      Comp.1 Comp.2 Comp.3
## WT2   0.492  0.781  0.384
## WT9   0.665 -0.053 -0.745
## WT18  0.562 -0.622  0.545
##
```

```
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

```
# HT
bgsHT.pca <- princomp( ~ HT2 + HT9 + HT18, data = bgs, cor = TRUE)
summary(bgsHT.pca)
```

```
## Importance of components:
##                            Comp.1    Comp.2     Comp.3
## Standard deviation     1.5975991 0.5723653 0.34651866
## Proportion of Variance 0.8507743 0.1092007 0.04002506
## Cumulative Proportion  0.8507743 0.9599749 1.00000000
```

```
print(loadings(bgsHT.pca), cutoff = 0)
```

```
##
## Loadings:
##      Comp.1 Comp.2 Comp.3
## HT2   0.554  0.800  0.231
## HT9   0.599 -0.190 -0.778
## HT18  0.578 -0.570  0.584
##
##                Comp.1 Comp.2 Comp.3
## SS loadings     1.000  1.000  1.000
## Proportion Var  0.333  0.333  0.333
## Cumulative Var  0.333  0.667  1.000
```

```
# LG
bgsLG.pca <- princomp( ~ LG9 + LG18, data = bgs, cor = TRUE)
summary(bgsLG.pca)
```

```
## Importance of components:
##                            Comp.1    Comp.2
## Standard deviation     1.2901455 0.5792449
## Proportion of Variance 0.8322377 0.1677623
## Cumulative Proportion  0.8322377 1.0000000
```

```
print(loadings(bgsLG.pca), cutoff = 0)
```

```
##
## Loadings:
##      Comp.1 Comp.2
## LG9   0.707  0.707
## LG18  0.707 -0.707
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
## Cumulative Var    0.5    1.0
```

```
# ST
bgsST.pca <- princomp( ~ ST9 + ST18, data = bgs, cor = TRUE)
summary(bgsST.pca)
```

```
## Importance of components:
##                           Comp.1    Comp.2
## Standard deviation     1.2882526 0.5834426
## Proportion of Variance 0.8297974 0.1702026
## Cumulative Proportion  0.8297974 1.0000000
print(loadings(bgsST.pca), cutoff = 0)
##
## Loadings:
##      Comp.1 Comp.2
## ST9   0.707  0.707
## ST18  0.707 -0.707
##
##                Comp.1 Comp.2
## SS loadings       1.0    1.0
## Proportion Var    0.5    0.5
## Cumulative Var    0.5    1.0
```

Chapter 14

Cluster Analysis

14.1 Introduction

Cluster analysis is an exploratory tool for locating and grouping observations that are similar to each other across features. Cluster analysis can also be used to group variables that are similar across observations.

Clustering or grouping is distinct from discriminant analysis and classification. In discrimination problems there are a given number of known groups to compare or distinguish. The aim in cluster analysis is to define groups based on similarities. The clusters are then examined for underlying characteristics that might help explain the grouping.

There are a variety of clustering algorithms[1]. I will discuss a simple **(agglomerative) hierarchical clustering method** for grouping observations. The method begins with each observation as an individual cluster or group. The two most similar observations are then grouped, giving one cluster with two observations. The remaining clusters have one observation. The clusters are then joined sequentially until one cluster is left.

14.1.1 Illustration

To illustrate the steps, suppose eight observations are collected on two features X_1 and X_2. A plot of the data is given below.

Step 1. Each observation is a cluster.

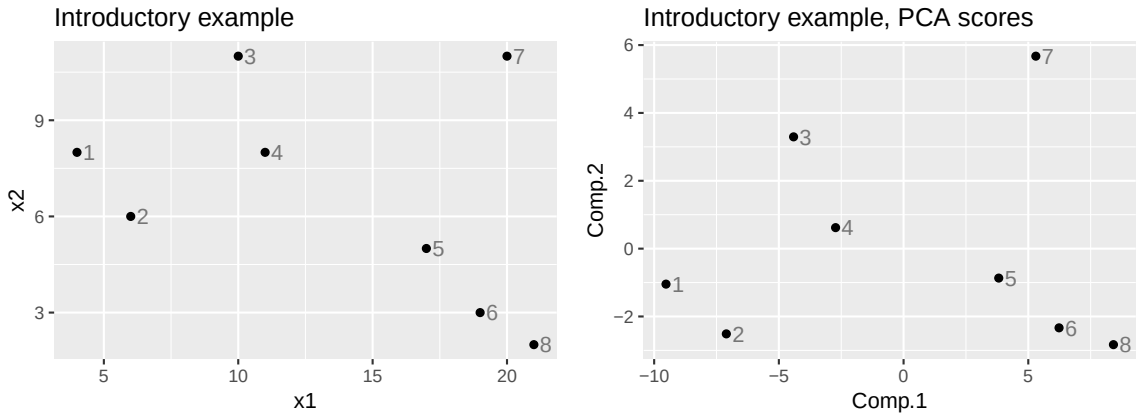
Step 2. Form a new cluster by grouping the two clusters that are most similar, or closest to each other. This leaves seven clusters.

Step 3. Form a new cluster by grouping the two clusters that are most similar, or closest to each other. This leaves six clusters.

[1] `http://cran.r-project.org/web/views/Cluster.html`

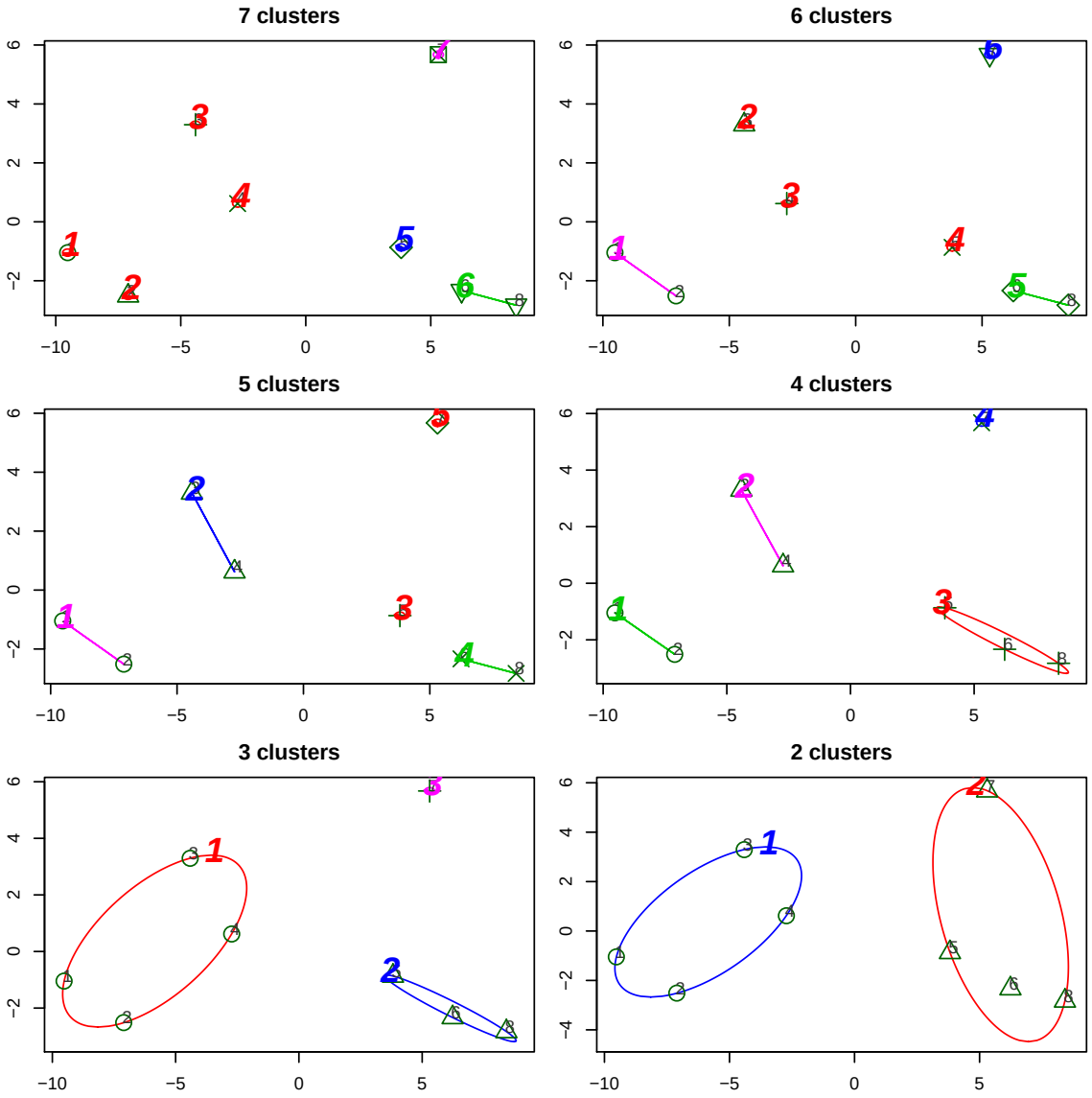
Step 4–7. Continue the process of merging clusters one at a time.
Step 8. Merge (fuse or combine) the remaining two clusters.
Finally Use a tree or dendrogram to summarize the steps in the cluster formation.

```
#### Example: Fake data cluster illustration
# convert to a data.frame by reading the text table
intro <- read.table(text = "
x1 x2
 4  8
 6  6
10 11
11  8
17  5
19  3
20 11
21  2
", header=TRUE)
str(intro)
## 'data.frame': 8 obs. of  2 variables:
##  $ x1: int  4 6 10 11 17 19 20 21
##  $ x2: int  8 6 11 8 5 3 11 2
# perform PCA on covariance matrix
intro.pca <- princomp( ~ x1 + x2, data = intro)


# plot original data
library(ggplot2)
p1 <- ggplot(intro, aes(x = x1, y = x2))
p1 <- p1 + geom_point()                                          # points
p1 <- p1 + geom_text(aes(label = 1:nrow(intro)), hjust = -0.5, alpha = 0.5)  # labels
p1 <- p1 + labs(title = "Introductory example")
print(p1)

# plot PCA scores (data on PC-scale centered at 0)
library(ggplot2)
p2 <- ggplot(as.data.frame(intro.pca$scores), aes(x = Comp.1, y = Comp.2))
p2 <- p2 + geom_point()           # points
p2 <- p2 + geom_text(aes(label = rownames(intro.pca$scores)), hjust = -0.5, alpha = 0.5) # labels
p2 <- p2 + labs(title = "Introductory example, PCA scores")
print(p2)
```



Here are the results of one distance measure, which will be discussed in more detail after the plots. The clustering algorithm order for average linkage is plotted here.

```
# create distance matrix between points
intro.dist <- dist(intro)
intro.hc.average  <- hclust(intro.dist, method = "average")

    op <- par(no.readonly = TRUE) # save original plot options
    par(mfrow = c(3,2), mar = c(2, 2, 2.5, 1))  # margins are c(bottom, left, top, right)

library(cluster)
for (i.clus in 7:2) {
  clusplot(intro, cutree(intro.hc.average, k = i.clus)
           , color = TRUE, labels = 2, lines = 0
           , cex = 2, cex.txt = 1, col.txt = "gray20"
           , main = paste(i.clus, "clusters"), sub = NULL)
}

    par(op) # reset plot options
```



The order of clustering is summarized in the average linkage dendrogram on the right reading the tree from the bottom upwards[2].

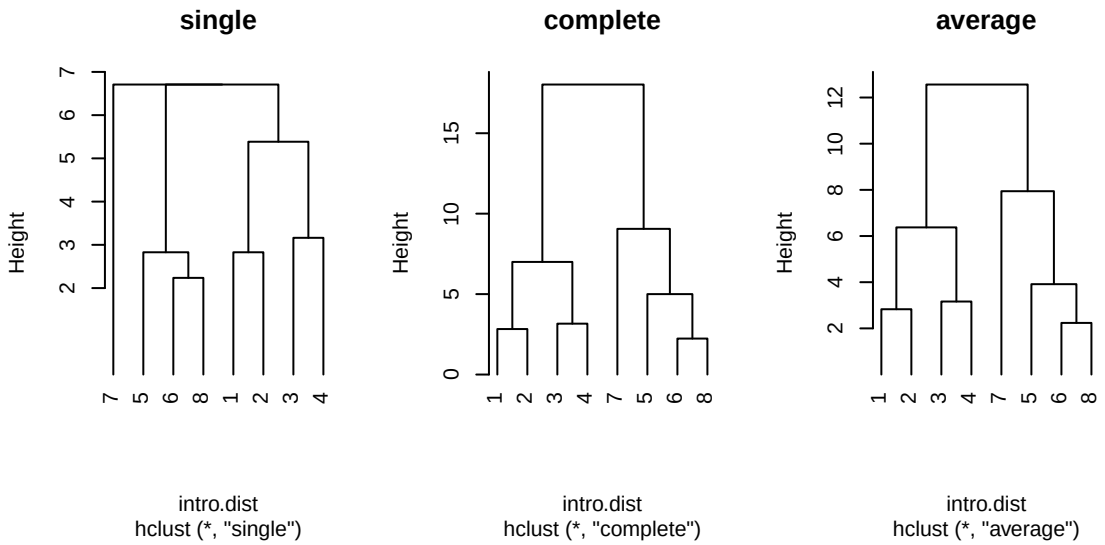
```
# create distance matrix between points
intro.dist <- dist(intro)
intro.dist
```

```
##          1        2        3        4        5        6
## 2 2.828427
## 3 6.708204 6.403124
## 4 7.000000 5.385165 3.162278
```

[2]There are many ways to create dendrograms in R, see `http://gastonsanchez.com/blog/how-to/2012/10/03/Dendrograms.html` for several examples.

```
## 5 13.341664 11.045361  9.219544  6.708204
## 6 15.811388 13.341664 12.041595  9.433981  2.828427
## 7 16.278821 14.866069 10.000000  9.486833  6.708204  8.062258
## 8 18.027756 15.524175 14.212670 11.661904  5.000000  2.236068
##           7
## 2
## 3
## 4
## 5
## 6
## 7
## 8  9.055385
```

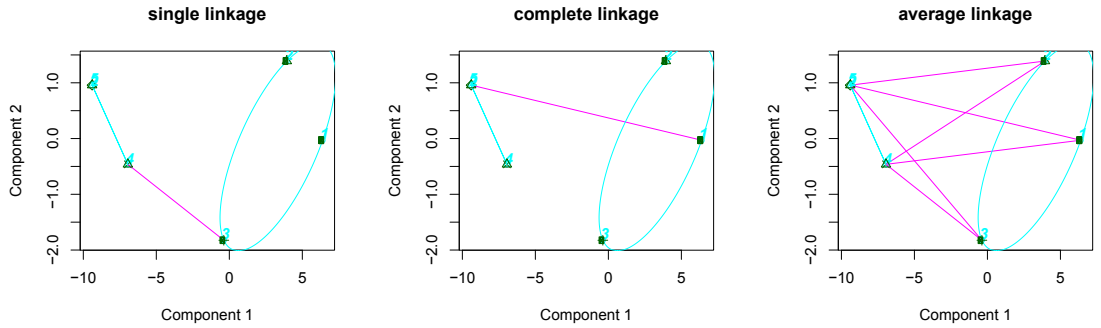
```
    op <- par(no.readonly = TRUE) # save original plot options
    par(mfrow = c(1,3))  # margins are c(bottom, left, top, right)

intro.hc.single   <- hclust(intro.dist, method = "single")
# note: plotting used to use plclust()
plot(intro.hc.single, hang = -1,       main = "single")

intro.hc.complete <- hclust(intro.dist, method = "complete")
plot(intro.hc.complete, hang = -1,     main = "complete")

intro.hc.average  <- hclust(intro.dist, method = "average")
plot(intro.hc.average, hang = -1,      main = "average")

    par(op) # reset plot options
```



14.1.2 Distance measures

There are several accepted measures of distance between clusters. The **single linkage** distance is the minimum distance between points across two clusters. The **complete linkage** distance is the maximum distance between points across two clusters. The **average linkage** distance is the average distance between points across two clusters. In these three cases the distance between points is the Euclidean or "ruler" distance. The pictures below illustrate the measures.

Given a distance measure, the distance between each pair of clusters is evaluated at each step. The two clusters that are closest to each other are merged. The observations are usually standardized prior to clustering to eliminate the effect of different variability on the distance measure.



Different distance measures can produce different **shape** clusters.

Single uses the length of the **shortest** line between points in clusters.
Single linkage has the ability to produce and detect elongated and irregular clusters.

Complete uses the length of the **longest** line between points in clusters.
Complete linkage is biased towards producing clusters with roughly equal diameters.

Average uses the **average** length of all line between points in clusters.
Average linkage tends to produce clusters with similar variability.

You should try different distances to decide the most sensible measure for your problem.

14.2 Example: Mammal teeth

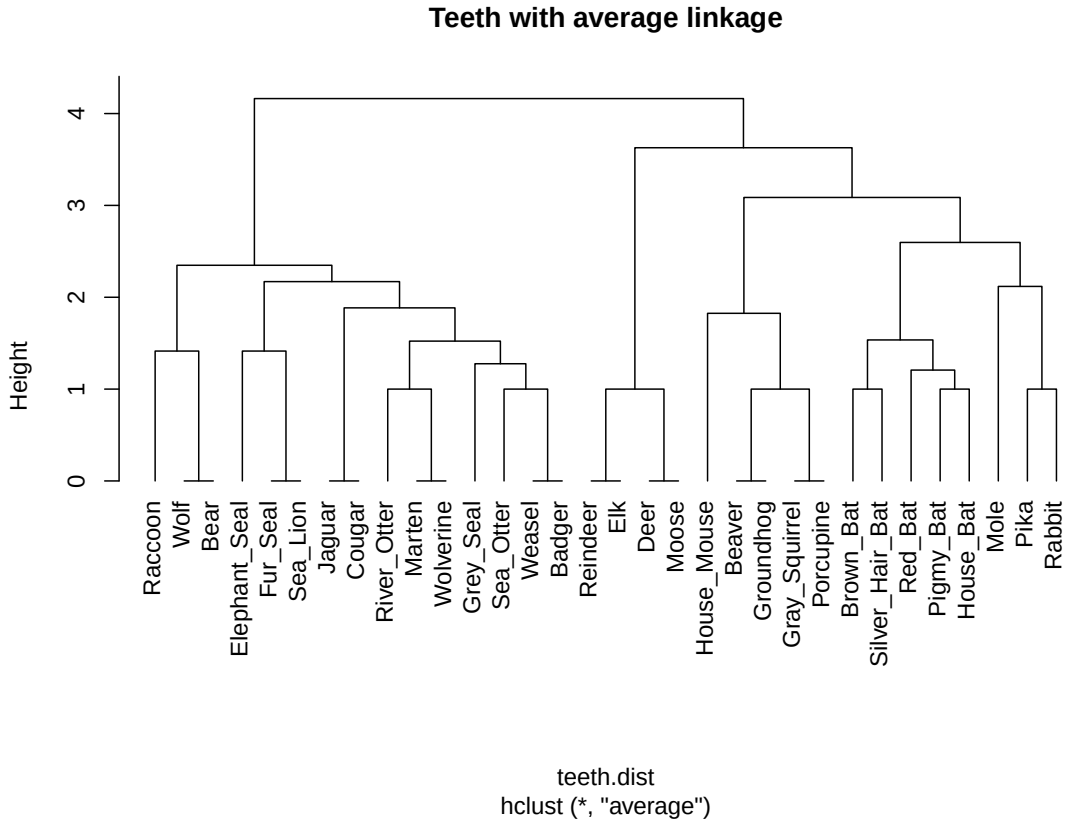
The table below gives the numbers of different types of teeth for 32 mammals. The columns, from left to right, give the numbers of (v1) top incisors, (v2) bottom incisors, (v3) top canines, (v4) bottom canines, (v5) top premolars, (v6) bottom premolars, (v7) top molars, (v8) bottom molars, respectively. A cluster analysis will be used to identify the mammals that have similar counts across the eight types of teeth.

```
#### Example: Mammal teeth
## Mammal teeth data
# mammal = name
#      number of teeth
# v1 = top incisors
# v2 = bottom incisors
# v3 = top canines
# v4 = bottom canines
# v5 = top premolars
# v6 = bottom premolars
# v7 = top molars
# v8 = bottom molars

fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch14_teeth.dat"
teeth <- read.table(fn.data, header = TRUE)
str(teeth)
## 'data.frame': 32 obs. of  9 variables:
##  $ mammal: Factor w/ 32 levels "Badger","Bear",..: 4 17 29 19 13 24 20 22 3 12 ...
##  $ v1    : int  2 3 2 2 2 1 2 2 1 1 ...
##  $ v2    : int  3 2 3 3 3 3 1 1 1 1 ...
##  $ v3    : int  1 1 1 1 1 1 0 0 0 0 ...
##  $ v4    : int  1 0 1 1 1 1 0 0 0 0 ...
##  $ v5    : int  3 3 2 2 1 2 2 3 2 2 ...
##  $ v6    : int  3 3 3 2 2 2 2 2 1 1 ...
##  $ v7    : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ v8    : int  3 3 3 3 3 3 3 3 3 3 ...
```

	mammal	v1	v2	v3	v4	v5	v6	v7	v8
1	Brown_Bat	2	3	1	1	3	3	3	3
2	Mole	3	2	1	0	3	3	3	3
3	Silver_Hair_Bat	2	3	1	1	2	3	3	3
4	Pigmy_Bat	2	3	1	1	2	2	3	3
5	House_Bat	2	3	1	1	1	2	3	3
6	Red_Bat	1	3	1	1	2	2	3	3
7	Pika	2	1	0	0	2	2	3	3
8	Rabbit	2	1	0	0	3	2	3	3
9	Beaver	1	1	0	0	2	1	3	3
10	Groundhog	1	1	0	0	2	1	3	3
11	Gray_Squirrel	1	1	0	0	1	1	3	3
12	House_Mouse	1	1	0	0	0	0	3	3
13	Porcupine	1	1	0	0	1	1	3	3
14	Wolf	3	3	1	1	4	4	2	3
15	Bear	3	3	1	1	4	4	2	3
16	Raccoon	3	3	1	1	4	4	3	2
17	Marten	3	3	1	1	4	4	1	2
18	Weasel	3	3	1	1	3	3	1	2
19	Wolverine	3	3	1	1	4	4	1	2
20	Badger	3	3	1	1	3	3	1	2
21	River_Otter	3	3	1	1	4	3	1	2
22	Sea_Otter	3	2	1	1	3	3	1	2
23	Jaguar	3	3	1	1	3	2	1	1
24	Cougar	3	3	1	1	3	2	1	1
25	Fur_Seal	3	2	1	1	4	4	1	1
26	Sea_Lion	3	2	1	1	4	4	1	1
27	Grey_Seal	3	2	1	1	3	3	2	2
28	Elephant_Seal	2	1	1	1	4	4	1	1
29	Reindeer	0	4	1	0	3	3	3	3
30	Elk	0	4	1	0	3	3	3	3
31	Deer	0	4	0	0	3	3	3	3
32	Moose	0	4	0	0	3	3	3	3

The program below produces cluster analysis summaries for the mammal teeth data.

```
# create distance matrix between points
teeth.dist <- dist(teeth[,-1])

# number of clusters to identify with red boxes and ellipses
# i.clus <- 8

# create dendrogram
teeth.hc.average  <- hclust(teeth.dist, method = "average")
plot(teeth.hc.average, hang = -1
      , main = paste("Teeth with average linkage") # and", i.clus, "clusters")
      , labels = teeth[,1])
# rect.hclust(teeth.hc.average, k = i.clus)

# # create PCA scores plot with ellipses
# clusplot(teeth, cutree(teeth.hc.average, k = i.clus)
#          , color = TRUE, labels = 2, lines = 0
#          , cex = 2, cex.txt = 1, col.txt = "gray20"
#          , main = paste("Teeth PCA with average linkage and", i.clus, "clusters")
#          , sub = NULL)
```



14.3 Identifying the Number of Clusters

Cluster analysis can be used to produce an "optimal" splitting of the data into a prespecified number of groups or clusters, with different algorithms[3] usually giving different clusters. However, the important issue in many analyses revolves around identifying the number of clusters in the data. A simple empirical method is to continue grouping until the clusters being fused are relatively dissimilar, as measured by the normalized RMS between clusters. Experience with your data is needed to provide a reasonable stopping rule.

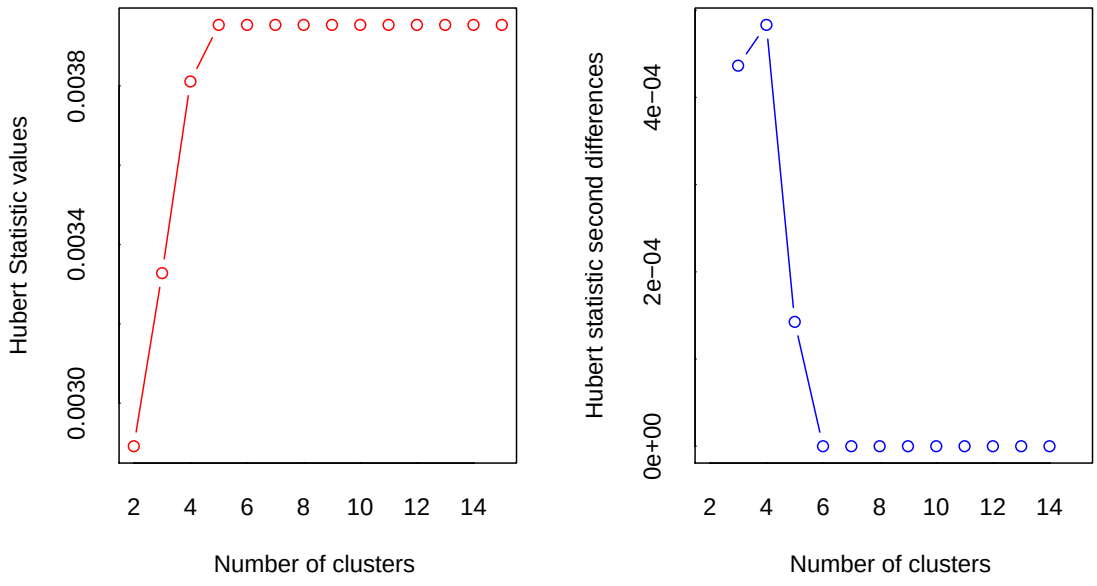
```
# NbClust provides methods for determining the number of clusters
library(NbClust)
str(teeth)
```

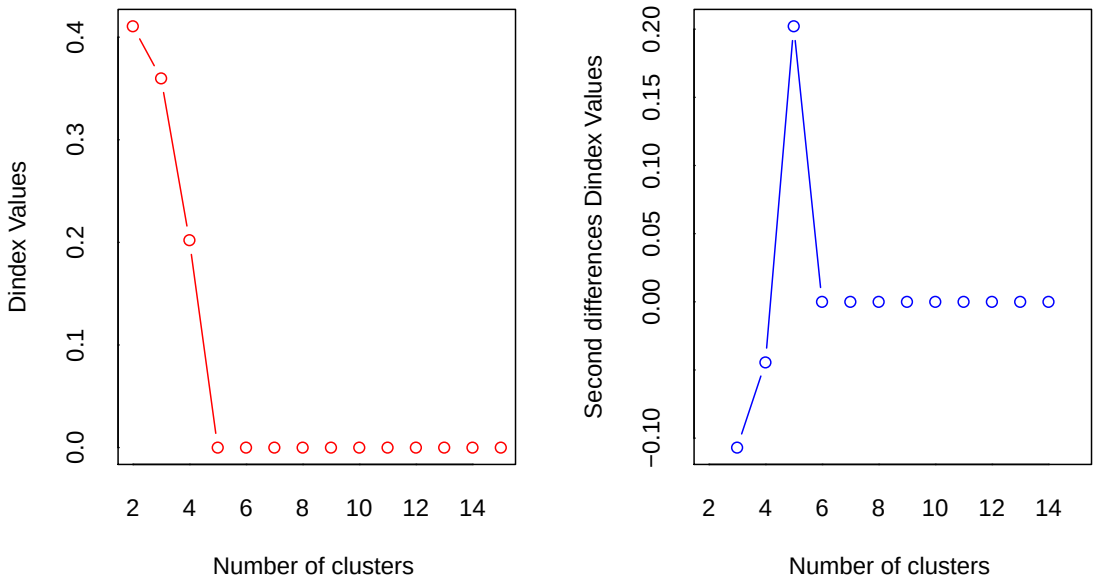
```
## 'data.frame': 32 obs. of  9 variables:
##  $ mammal: Factor w/ 32 levels "Badger","Bear",..: 4 17 29 19 13 24 20 22 3 12 ...
```

[3]There are thirty in this package: `http://cran.r-project.org/web/packages/NbClust/NbClust.pdf`

```
##  $ v1    : int  2 3 2 2 2 1 2 2 1 1 ...
##  $ v2    : int  3 2 3 3 3 3 1 1 1 1 ...
##  $ v3    : int  1 1 1 1 1 1 0 0 0 0 ...
##  $ v4    : int  1 0 1 1 1 1 0 0 0 0 ...
##  $ v5    : int  3 3 2 2 1 2 2 3 2 2 ...
##  $ v6    : int  3 3 3 2 2 2 2 2 1 1 ...
##  $ v7    : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ v8    : int  3 3 3 3 3 3 3 3 3 3 ...
# Because the data type is "int" for integer, the routine fails (error expected)
NbClust(teeth[,-1], method = "average", index = "all")
## Error in solve.default(W): system is computationally singular:  reciprocal condition
number = 1.51394e-16
# However, change the data type from integer to numeric and it works just fine!
teeth.num <- as.numeric(as.matrix(teeth[,-1]))
NC.out <- NbClust(teeth.num, method = "average", index = "all")
## Warning in max(DiffLev[, 5], na.rm = TRUE): no non-missing arguments to max; returning
-Inf
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## Warning in matrix(c(results), nrow = 2, ncol = 26):  data length [51] is not a sub-multiple
or multiple of the number of rows [2]
## Warning in matrix(c(results), nrow = 2, ncol = 26, dimnames = list(c("Number_clusters",
:  data length [51] is not a sub-multiple or multiple of the number of rows [2]
## *******************************************************************
## * Among all indices:
## * 1 proposed 4 as the best number of clusters
## * 5 proposed 5 as the best number of clusters
##
##                    ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  5
##
##
## *******************************************************************
# most of the methods suggest 4 or 5 clusters, as do the plots
NC.out$Best.nc
##                  KL  CH Hartigan      CCC    Scott Marriot TrCovW
## Number_clusters   5   5        4   5.0000    5.000       5   -Inf
## Value_Index     Inf Inf      Inf 369.1341 7787.404     414      5
```

```
##                 TraceW     Friedman         Rubin Cindex DB
## Number_clusters 25.875 8.720698e+14 -9.810785e+14      0  0
## Value_Index      5.000 5.000000e+00  6.000000e+00      5  5
##                 Silhouette   Duda PseudoT2  Beale Ratkowsky    Ball
## Number_clusters          1 0.4663 168.2472 0.3789    0.4737 61.8333
## Value_Index              2 2.0000   2.0000 3.0000    3.0000  3.0000
##                 PtBiserial Frey McClain Dunn Hubert SDindex Dindex
## Number_clusters     0.7713   NA       0  Inf      0     Inf      0
## Value_Index         1.0000    5       5    0      2       0      5
##                 SDbw
## Number_clusters    0
## Value_Index        5
```





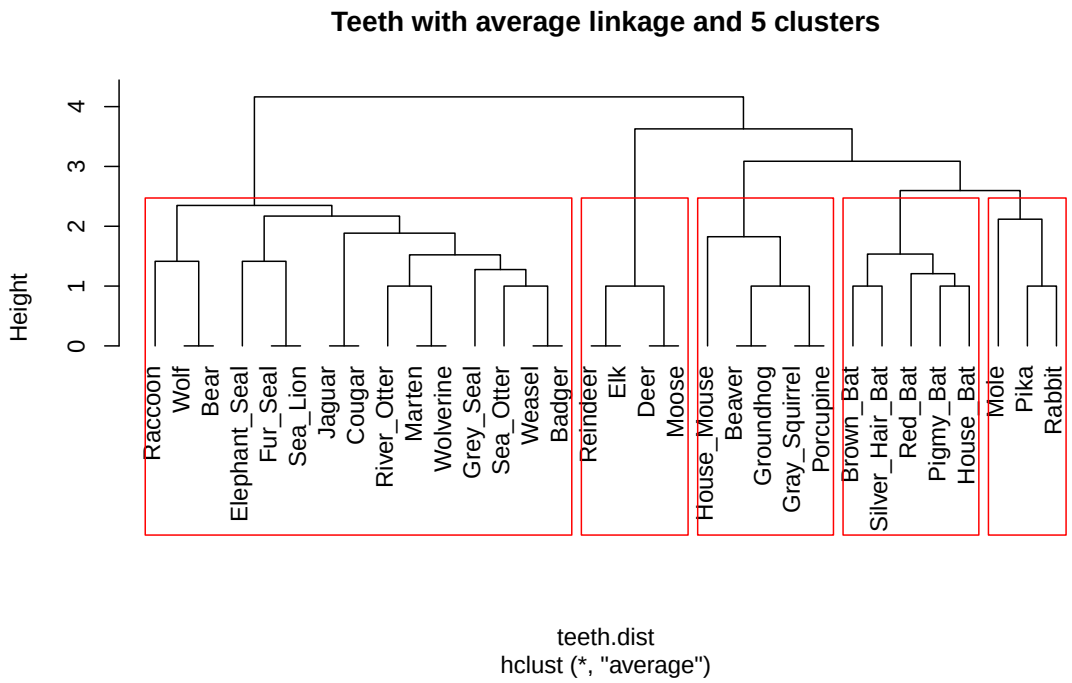
There are several statistical methods for selecting the number of clusters. No method is best. They suggest using the cubic clustering criteria (`ccc`), a pseudo-F statistic, and a pseudo-t statistic. At a given step, the pseudo-t statistic is the distance between the center of the two clusters to be merged, relative to the variability within these clusters. A large pseudo-t statistic implies that the clusters to be joined are relatively dissimilar (i.e., much more variability between the clusters to be merged than within these clusters). The pseudo-F statistic at a given step measures the variability among the centers of the current clusters relative to the variability within the clusters. A large pseudo-F value implies that the clusters merged consist of fairly similar observations. As clusters are joined, the pseudo-t statistic tends to increase, and the pseudo-F statistic tends to decrease. The ccc is more difficult to describe.

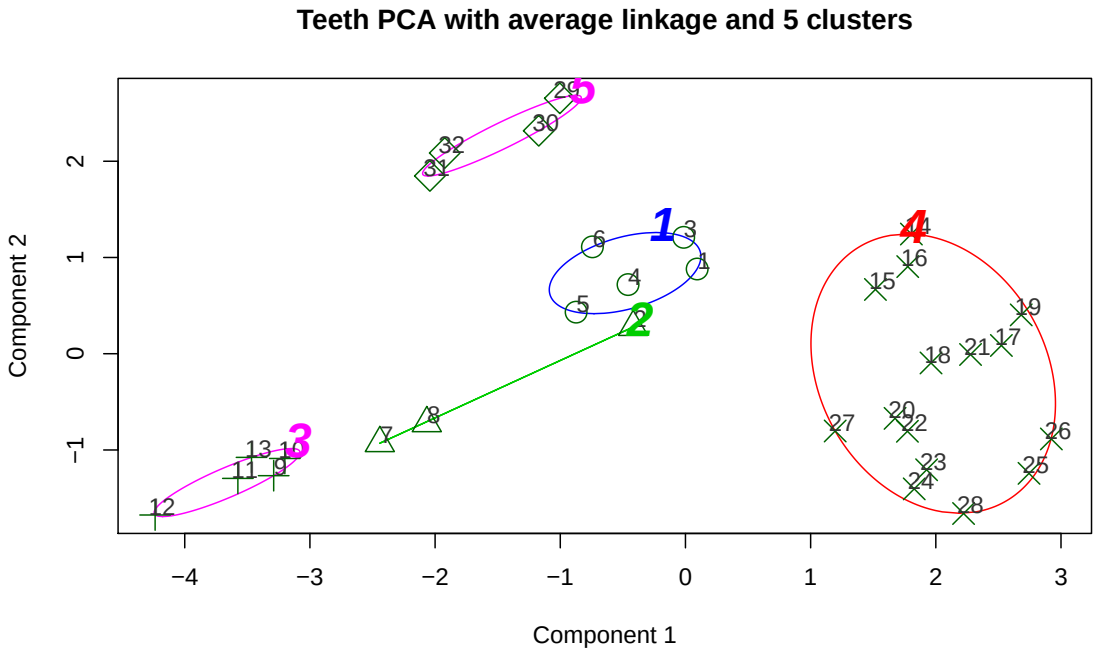
The RSQ summary is also useful for determining the number of clusters. RSQ is a pseudo-R^2 statistic that measures the proportion of the total variation explained by the differences among the existing clusters at a given step. RSQ will typically decrease as the pseudo-F statistic decreases.

A common recommendation on cluster selection is to choose a cluster size where the values of ccc and the pseudo-F statistic are relatively high (compared to what you observe with other numbers of clusters), and where the pseudo-t statistic is relatively low and increases substantially at the next proposed merger. For the mammal teeth data this corresponds to four clusters. Six clusters is a sensible second choice.

Let's look at the results of 5 clusters.

```
# create distance matrix between points
teeth.dist <- dist(teeth[,-1])

# number of clusters to identify with red boxes and ellipses
i.clus <- 5

# create dendrogram
teeth.hc.average  <- hclust(teeth.dist, method = "average")
plot(teeth.hc.average, hang = -1
      , main = paste("Teeth with average linkage and", i.clus, "clusters")
      , labels = teeth[,1])
rect.hclust(teeth.hc.average, k = i.clus)

# create PCA scores plot with ellipses
clusplot(teeth, cutree(teeth.hc.average, k = i.clus)
        , color = TRUE, labels = 2, lines = 0
        , cex = 2, cex.txt = 1, col.txt = "gray20"
        , main = paste("Teeth PCA with average linkage and", i.clus, "clusters")
        , sub = NULL)
```






```
# print the observations in each cluster
for (i.cut in 1:i.clus) {
  print(paste("Cluster", i.cut, " ----------------------------- "))
  print(teeth[(cutree(teeth.hc.average, k = i.clus) == i.cut),])
}
```

```
## [1] "Cluster 1  ----------------------------- "
##            mammal v1 v2 v3 v4 v5 v6 v7 v8
## 1       Brown_Bat  2  3  1  1  3  3  3  3
## 3 Silver_Hair_Bat  2  3  1  1  2  3  3  3
## 4       Pigmy_Bat  2  3  1  1  2  2  3  3
## 5       House_Bat  2  3  1  1  1  2  3  3
## 6         Red_Bat  1  3  1  1  2  2  3  3
## [1] "Cluster 2  ----------------------------- "
##   mammal v1 v2 v3 v4 v5 v6 v7 v8
## 2   Mole  3  2  1  0  3  3  3  3
## 7   Pika  2  1  0  0  2  2  3  3
## 8 Rabbit  2  1  0  0  3  2  3  3
## [1] "Cluster 3  ----------------------------- "
##           mammal v1 v2 v3 v4 v5 v6 v7 v8
## 9          Beaver  1  1  0  0  2  1  3  3
## 10     Groundhog  1  1  0  0  2  1  3  3
## 11 Gray_Squirrel  1  1  0  0  1  1  3  3
## 12   House_Mouse  1  1  0  0  0  0  3  3
## 13     Porcupine  1  1  0  0  1  1  3  3
## [1] "Cluster 4  ----------------------------- "
```

```
##           mammal v1 v2 v3 v4 v5 v6 v7 v8
## 14          Wolf  3  3  1  1  4  4  2  3
## 15          Bear  3  3  1  1  4  4  2  3
## 16       Raccoon  3  3  1  1  4  4  3  2
## 17        Marten  3  3  1  1  4  4  1  2
## 18        Weasel  3  3  1  1  3  3  1  2
## 19     Wolverine  3  3  1  1  4  4  1  2
## 20        Badger  3  3  1  1  3  3  1  2
## 21   River_Otter  3  3  1  1  4  3  1  2
## 22     Sea_Otter  3  2  1  1  3  3  1  2
## 23        Jaguar  3  3  1  1  3  2  1  1
## 24        Cougar  3  3  1  1  3  2  1  1
## 25      Fur_Seal  3  2  1  1  4  4  1  1
## 26      Sea_Lion  3  2  1  1  4  4  1  1
## 27     Grey_Seal  3  2  1  1  3  3  2  2
## 28 Elephant_Seal  2  1  1  1  4  4  1  1
## [1] "Cluster 5  ---------------------------- "
##      mammal v1 v2 v3 v4 v5 v6 v7 v8
## 29 Reindeer  0  4  1  0  3  3  3  3
## 30      Elk  0  4  1  0  3  3  3  3
## 31     Deer  0  4  0  0  3  3  3  3
## 32    Moose  0  4  0  0  3  3  3  3
```

14.4 Example: 1976 birth and death rates

Below are the 1976 crude birth and death rates in 74 countries. A data plot and output from a complete and single linkage cluster analyses are given.

```
#### Example: Birth and death rates
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch14_birthdeath.dat"
bd <- read.table(fn.data, header = TRUE)
str(bd)
```

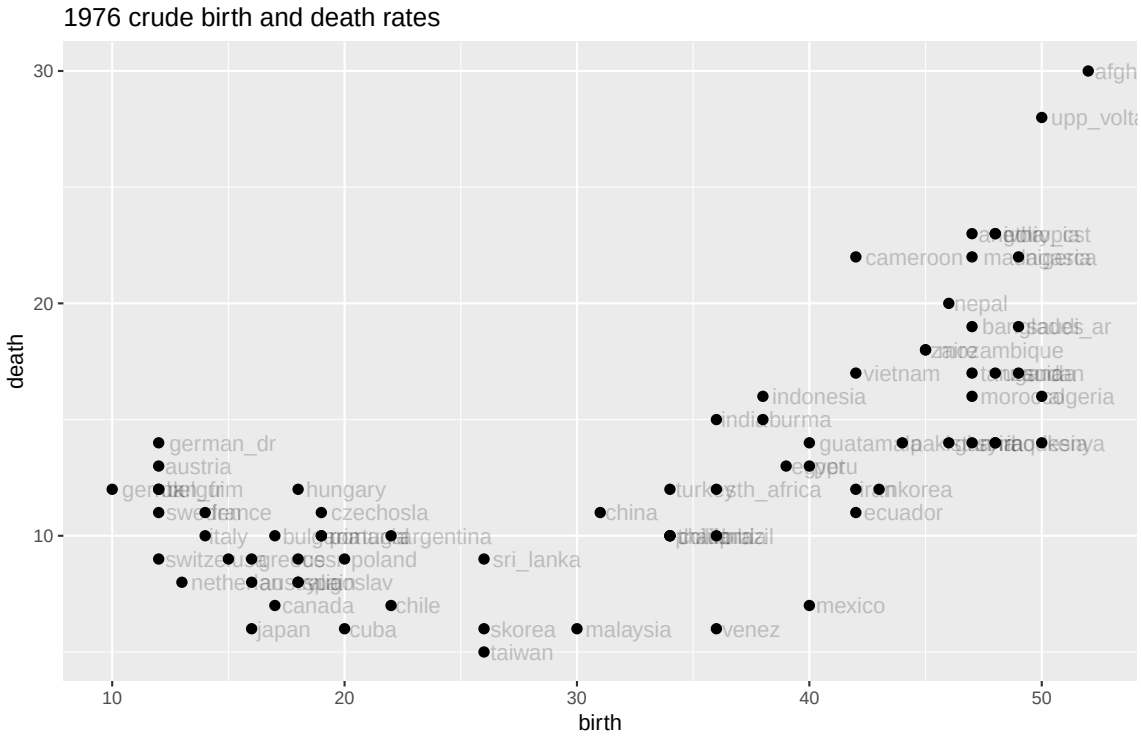
```
## 'data.frame': 74 obs. of  3 variables:
##  $ country: Factor w/ 74 levels "afghan","algeria",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ birth  : int  52 50 47 22 16 12 47 12 36 17 ...
##  $ death  : int  30 16 23 10 8 13 19 12 10 10 ...
```

	country	birth	death
1	afghan	52	30
2	algeria	50	16
3	angola	47	23
4	argentina	22	10
5	australia	16	8
6	austria	12	13
7	banglades	47	19
8	belguim	12	12
9	brazil	36	10
10	bulgaria	17	10
11	burma	38	15
12	cameroon	42	22
13	canada	17	7
14	chile	22	7
15	china	31	11
16	taiwan	26	5
17	columbia	34	10
18	cuba	20	6
19	czechosla	19	11
20	ecuador	42	11
21	egypt	39	13
22	ethiopia	48	23
23	france	14	11
24	german_dr	12	14
25	german_fr	10	12

	country	birth	death
26	ghana	46	14
27	greece	16	9
28	guatamala	40	14
29	hungary	18	12
30	india	36	15
31	indonesia	38	16
32	iran	42	12
33	iraq	48	14
34	italy	14	10
35	ivory_cst	48	23
36	japan	16	6
37	kenya	50	14
38	nkorea	43	12
39	skorea	26	6
40	madagasca	47	22
41	malaysia	30	6
42	mexico	40	7
43	morocco	47	16
44	mozambique	45	18
45	nepal	46	20
46	netherlan	13	8
47	nigeria	49	22
48	pakistan	44	14
49	peru	40	13
50	phillip	34	10

	country	birth	death
51	poland	20	9
52	portugal	19	10
53	rhodesia	48	14
54	romania	19	10
55	saudi_ar	49	19
56	sth_africa	36	12
57	spain	18	8
58	sri_lanka	26	9
59	sudan	49	17
60	sweden	12	11
61	switzer	12	9
62	syria	47	14
63	tanzania	47	17
64	thailand	34	10
65	turkey	34	12
66	ussr	18	9
67	uganda	48	17
68	uk	12	12
69	usa	15	9
70	upp_volta	50	28
71	venez	36	6
72	vietnam	42	17
73	yugoslav	18	8
74	zaire	45	18

```
# plot original data
library(ggplot2)
p1 <- ggplot(bd, aes(x = birth, y = death))
p1 <- p1 + geom_point(size = 2)                                    # points
p1 <- p1 + geom_text(aes(label = country), hjust = -0.1, alpha = 0.2)   # labels
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
p1 <- p1 + labs(title = "1976 crude birth and death rates")
print(p1)
```

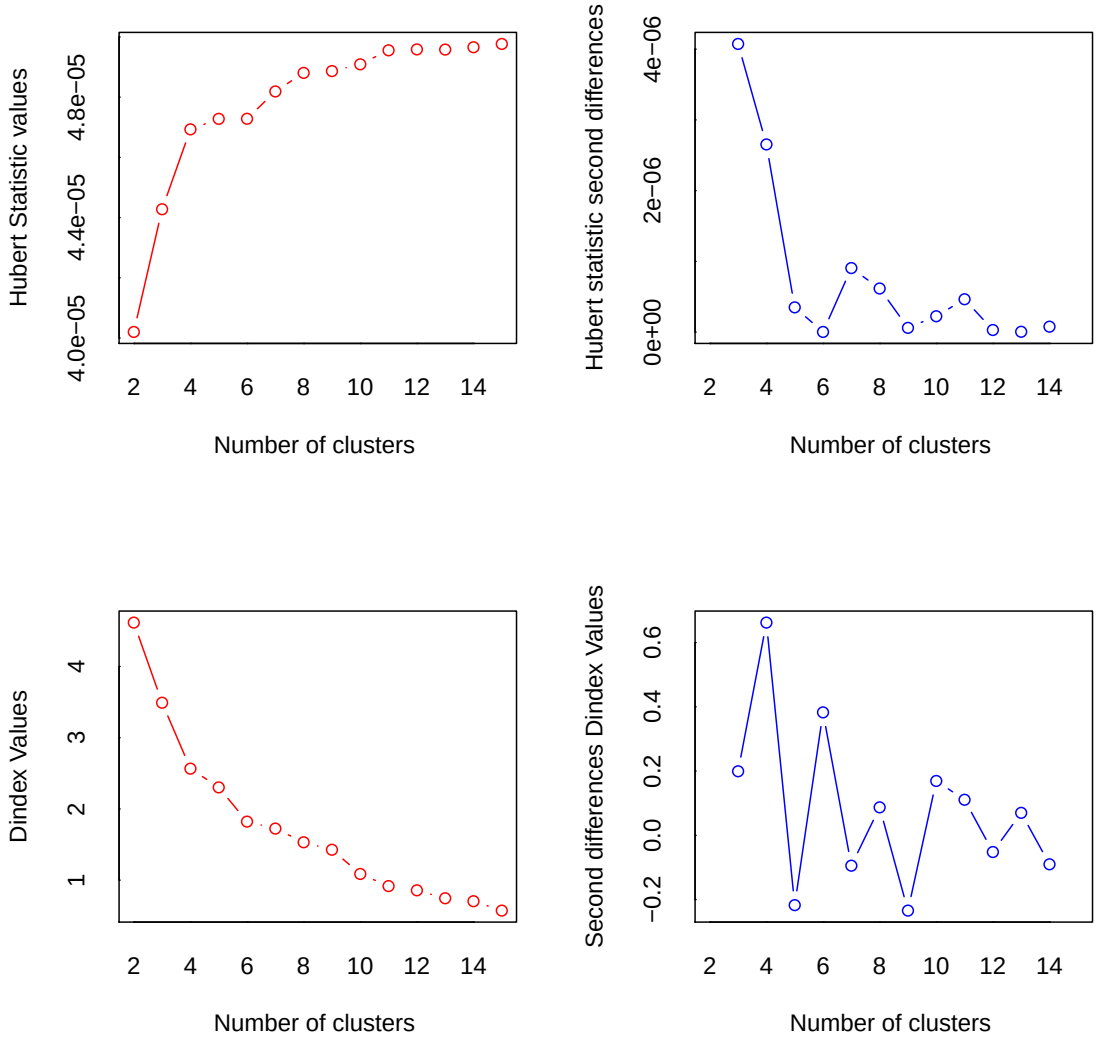


14.4.1 Complete linkage

```
library(NbClust)
# Change integer data type to numeric
bd.num <- as.numeric(as.matrix(bd[,-1]))
NC.out <- NbClust(bd.num, method = "complete", index = "all")
## Warning in max(DiffLev[, 5], na.rm = TRUE): no non-missing arguments to max; returning
-Inf
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in H
##                 index second differences plot.
##
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Di
##                 second differences plot) that corresponds to a significant increase of the val
##                 the measure.
##
## Warning in matrix(c(results), nrow = 2, ncol = 26):  data length [51] is not a sub-multiple
```

```
or multiple of the number of rows [2]
## Warning in matrix(c(results), nrow = 2, ncol = 26, dimnames = list(c("Number_clusters",
:  data length [51] is not a sub-multiple or multiple of the number of rows [2]
```

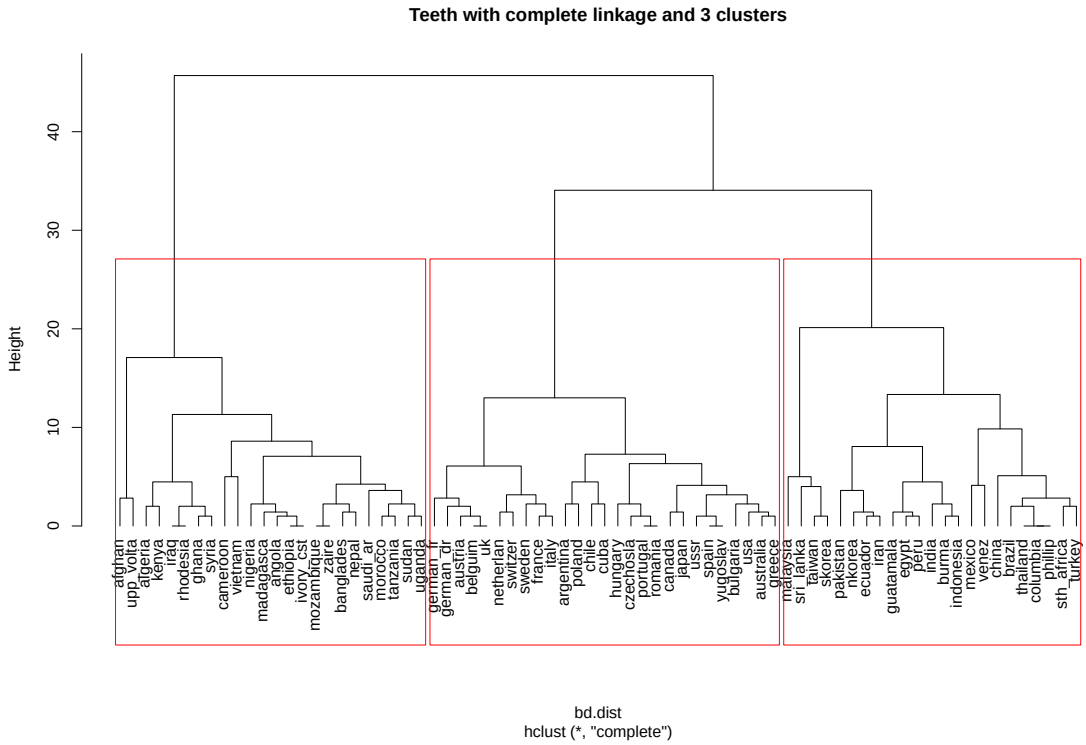
```
# most of the methods suggest 2 to 6 clusters, as do the plots
NC.out$Best.nc
```

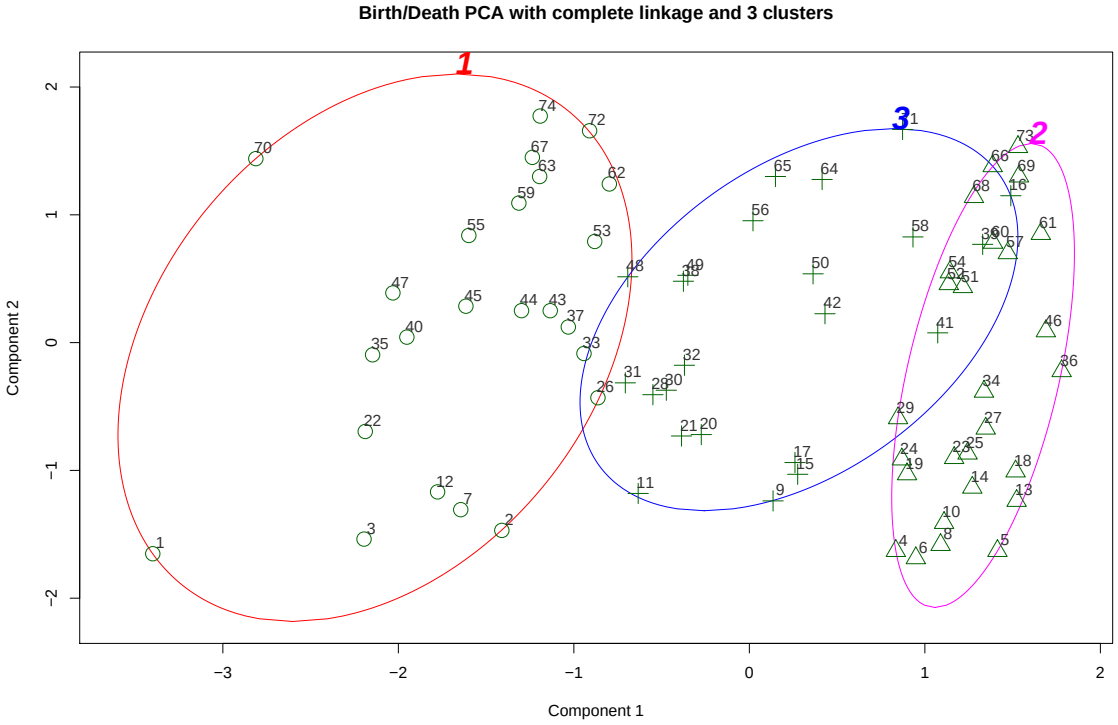


Let's try 3 clusters based on the dendrogram plots below. First we'll use complete linkage.

```
# create distance matrix between points
bd.dist <- dist(bd[,-1])

# number of clusters to identify with red boxes and ellipses
i.clus <- 3

# create dendrogram
bd.hc.complete  <- hclust(bd.dist, method = "complete")
plot(bd.hc.complete, hang = -1
```

```
      , main = paste("Teeth with complete linkage and", i.clus, "clusters")
      , labels = bd[,1])
rect.hclust(bd.hc.complete, k = i.clus)

# create PCA scores plot with ellipses
clusplot(bd, cutree(bd.hc.complete, k = i.clus)
        , color = TRUE, labels = 2, lines = 0
        , cex = 2, cex.txt = 1, col.txt = "gray20"
        , main = paste("Birth/Death PCA with complete linkage and", i.clus, "clusters"), sub = NULL)

# create a column with group membership
bd$cut.comp <- factor(cutree(bd.hc.complete, k = i.clus))
```



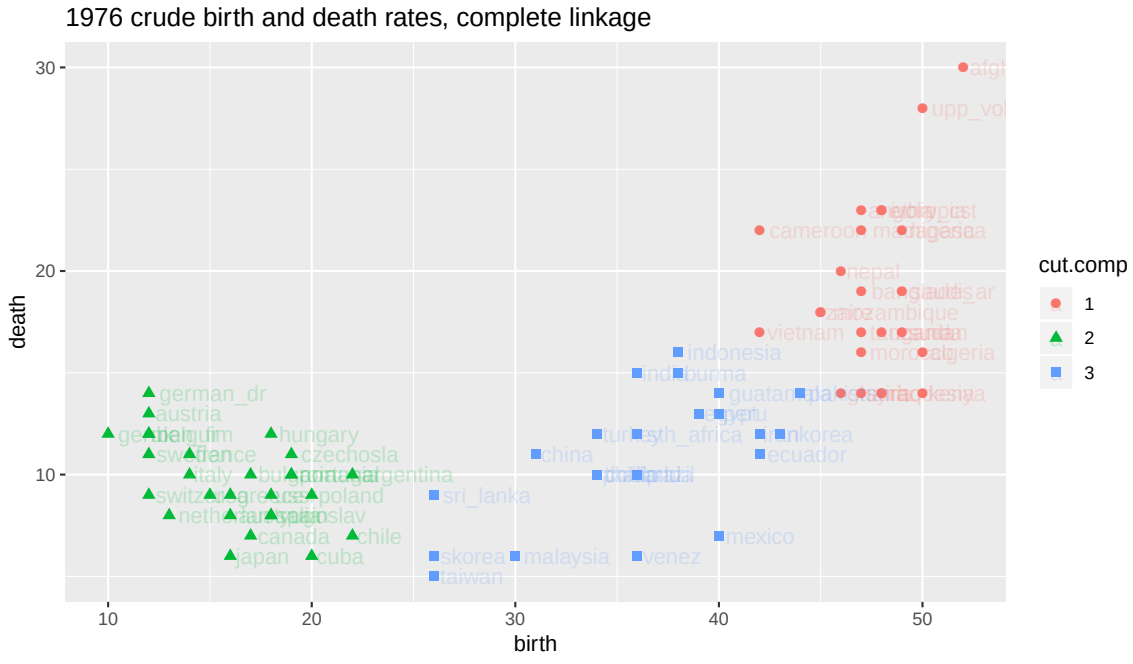



```
# print the observations in each cluster
for (i.cut in 1:i.clus) {
  print(paste("Cluster", i.cut, " ------------------------------ "))
  print(bd[(cutree(bd.hc.complete, k = i.clus) == i.cut),])
}
## [1] "Cluster 1  ------------------------------ "
##       country birth death cut.comp
## 1      afghan    52    30        1
## 2     algeria    50    16        1
## 3      angola    47    23        1
## 7   banglades    47    19        1
## 12   cameroon    42    22        1
## 22   ethiopia    48    23        1
## 26      ghana    46    14        1
## 33       iraq    48    14        1
## 35  ivory_cst    48    23        1
## 37      kenya    50    14        1
## 40  madagasca    47    22        1
## 43    morocco    47    16        1
## 44 mozambique    45    18        1
## 45      nepal    46    20        1
## 47    nigeria    49    22        1
## 53   rhodesia    48    14        1
## 55   saudi_ar    49    19        1
```

```
## 59      sudan    49    17        1
## 62      syria    47    14        1
## 63   tanzania    47    17        1
## 67     uganda    48    17        1
## 70  upp_volta    50    28        1
## 72    vietnam    42    17        1
## 74      zaire    45    18        1
## [1] "Cluster 2  ----------------------------- "
##      country birth death cut.comp
## 4  argentina    22    10        2
## 5  australia    16     8        2
## 6    austria    12    13        2
## 8    belguim    12    12        2
## 10  bulgaria    17    10        2
## 13    canada    17     7        2
## 14     chile    22     7        2
## 18      cuba    20     6        2
## 19 czechosla    19    11        2
## 23    france    14    11        2
## 24 german_dr    12    14        2
## 25 german_fr    10    12        2
## 27    greece    16     9        2
## 29   hungary    18    12        2
## 34     italy    14    10        2
## 36     japan    16     6        2
## 46 netherlan    13     8        2
## 51    poland    20     9        2
## 52  portugal    19    10        2
## 54   romania    19    10        2
## 57     spain    18     8        2
## 60    sweden    12    11        2
## 61   switzer    12     9        2
## 66      ussr    18     9        2
## 68        uk    12    12        2
## 69       usa    15     9        2
## 73  yugoslav    18     8        2
## [1] "Cluster 3  ----------------------------- "
##       country birth death cut.comp
## 9      brazil    36    10        3
## 11      burma    38    15        3
## 15      china    31    11        3
## 16     taiwan    26     5        3
## 17   columbia    34    10        3
## 20    ecuador    42    11        3
## 21      egypt    39    13        3
## 28  guatamala    40    14        3
## 30      india    36    15        3
## 31  indonesia    38    16        3
```

```
## 32       iran    42    12         3
## 38     nkorea    43    12         3
## 39     skorea    26     6         3
## 41   malaysia    30     6         3
## 42     mexico    40     7         3
## 48   pakistan    44    14         3
## 49       peru    40    13         3
## 50    phillip    34    10         3
## 56 sth_africa    36    12         3
## 58  sri_lanka    26     9         3
## 64   thailand    34    10         3
## 65     turkey    34    12         3
## 71      venez    36     6         3
```

```
# plot original data
library(ggplot2)
p1 <- ggplot(bd, aes(x = birth, y = death, colour = cut.comp, shape = cut.comp))
p1 <- p1 + geom_point(size = 2)                                    # points
p1 <- p1 + geom_text(aes(label = country), hjust = -0.1, alpha = 0.2)  # labels
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
p1 <- p1 + labs(title = "1976 crude birth and death rates, complete linkage")
print(p1)
```



In very general/loose terms[4], it appears that at least some members of the "Four

[4]Thanks to Drew Enigk from Spring 2013 who provided this interpretation.

Asian Tigers[5]" are toward the bottom of the swoop, while the countries with more Euro-centric wealth are mostly clustered on the left side of the swoop, and many developing countries make up the steeper right side of the swoop. Perhaps the birth and death rates of a given country are influenced in part by the primary means by which the country has obtained wealth[6] (if it is considered a wealthy country). For example, the Four Asian Tigers have supposedly developed wealth in more recent years through export-driven economies, and the Tiger Cub Economies[7] are currently developing in a similar fashion[8].

14.4.2 Single linkage

Now we'll use single linkage to compare.

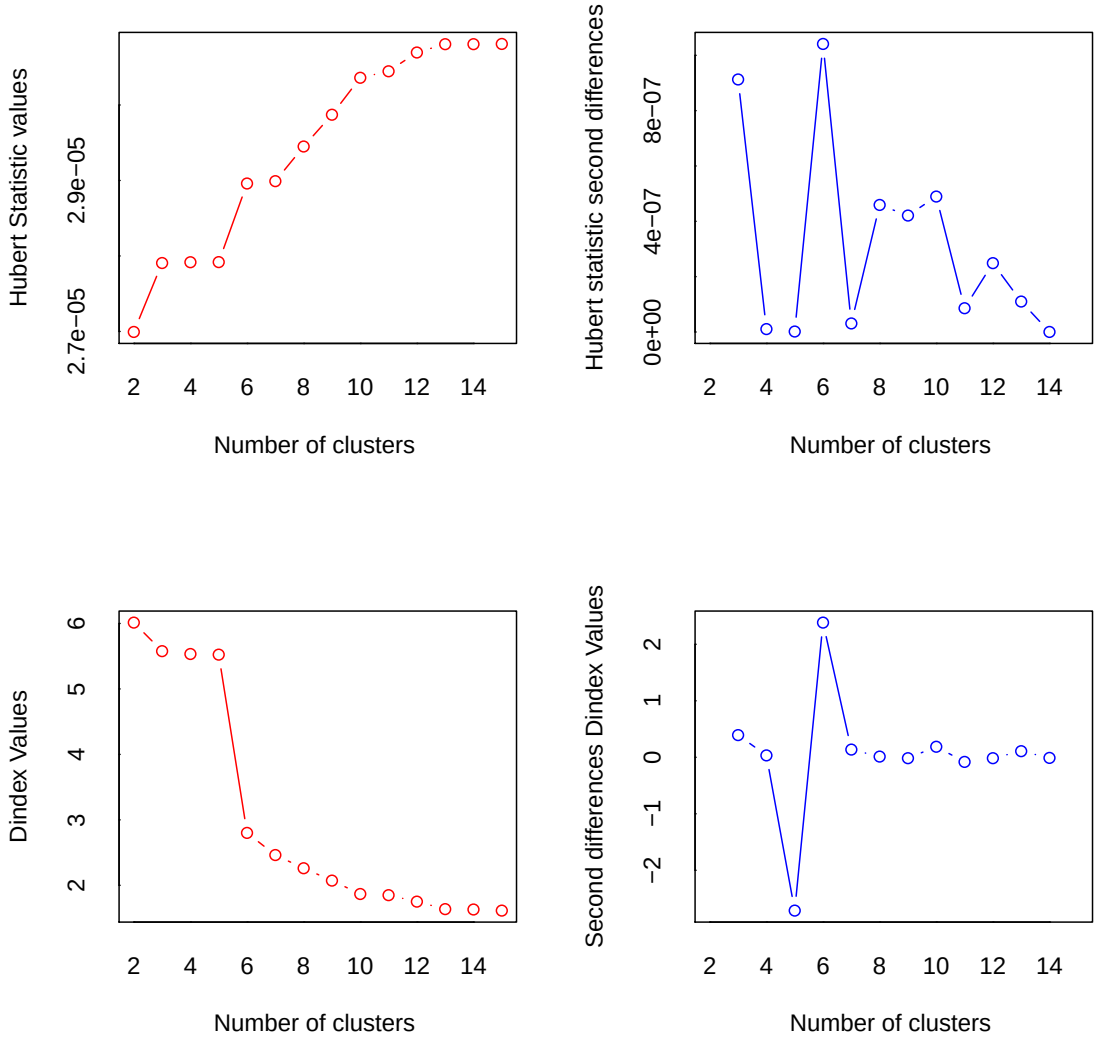
```
library(NbClust)
# Change integer data type to numeric
bd.num <- as.numeric(as.matrix(bd[,-1]))
NC.out <- NbClust(bd.num, method = "single", index = "all")
```

```
## Warning in max(DiffLev[, 5], na.rm = TRUE): no non-missing arguments to max; returning -Inf
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## Warning in matrix(c(results), nrow = 2, ncol = 26):  data length [51] is not a sub-multiple or multiple of the number of rows [2]
## Warning in matrix(c(results), nrow = 2, ncol = 26, dimnames = list(c("Number_clusters", :  data length [51] is not a sub-multiple or multiple
of the number of rows [2]
## *******************************************************************
## * Among all indices:
## * 1 proposed 2 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 2 proposed 6 as the best number of clusters
## * 1 proposed 7 as the best number of clusters
## * 1 proposed 11 as the best number of clusters
##
##                    ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  6
##
##
## *******************************************************************
```

```
# most of the methods suggest 4 to 11 clusters, as do the plots
NC.out$Best.nc
```

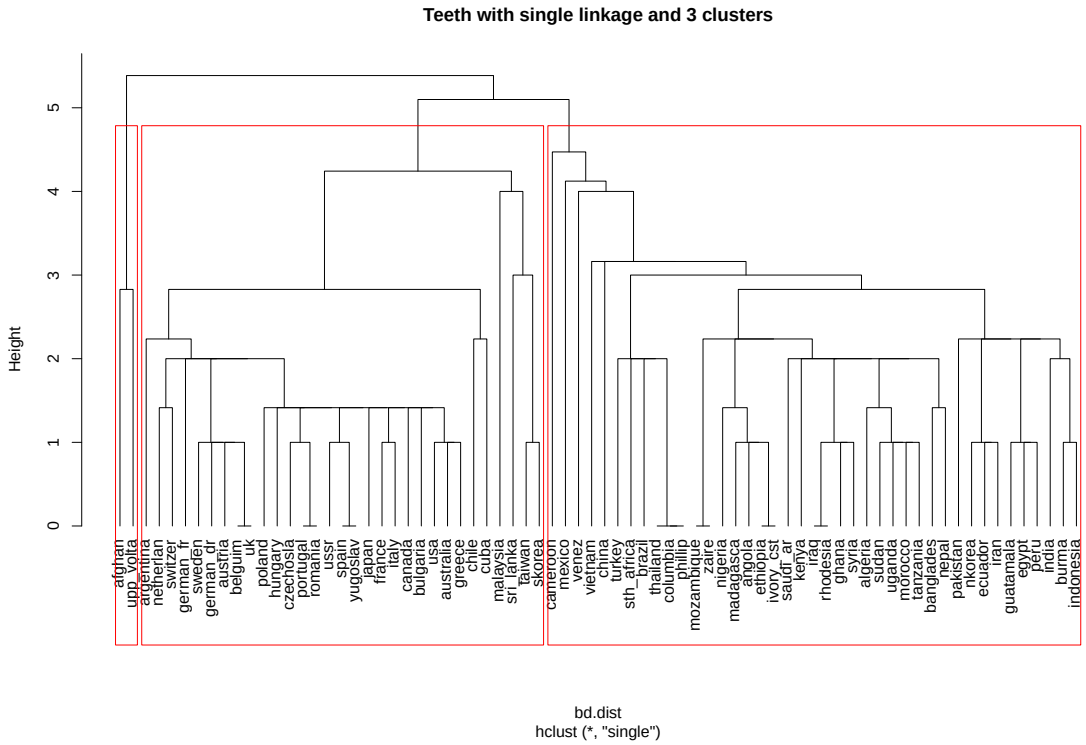
```
##                     KL       CH Hartigan     CCC    Scott  Marriot
## Number_clusters 7.0000  11.0000   5.0000  2.0000   6.0000      6.0
## Value_Index     8.5944 342.3491 473.8986 13.7816 221.8442 115129.2
##                 TrCovW   TraceW Friedman    Rubin Cindex    DB
## Number_clusters   -Inf 4990.876  20.3754 -12.0601  0.189 0.341
## Value_Index          6    6.000   6.0000   7.0000 15.000 2.000
##                 Silhouette   Duda PseudoT2 Beale Ratkowsky     Ball
## Number_clusters     0.7364 0.4667  53.7092 0.373    0.3521 9161.833
## Value_Index         2.0000 2.0000   2.0000 7.000    3.0000    2.000
##                 PtBiserial   Frey McClain   Dunn Hubert SDindex
## Number_clusters     0.8462 4.0846  0.1235 0.1364      0  0.5134
## Value_Index         2.0000 2.0000  3.0000 0.0000      3  0.0000
##                 Dindex   SDbw
## Number_clusters      0 0.0442
## Value_Index         15 7.0000
```

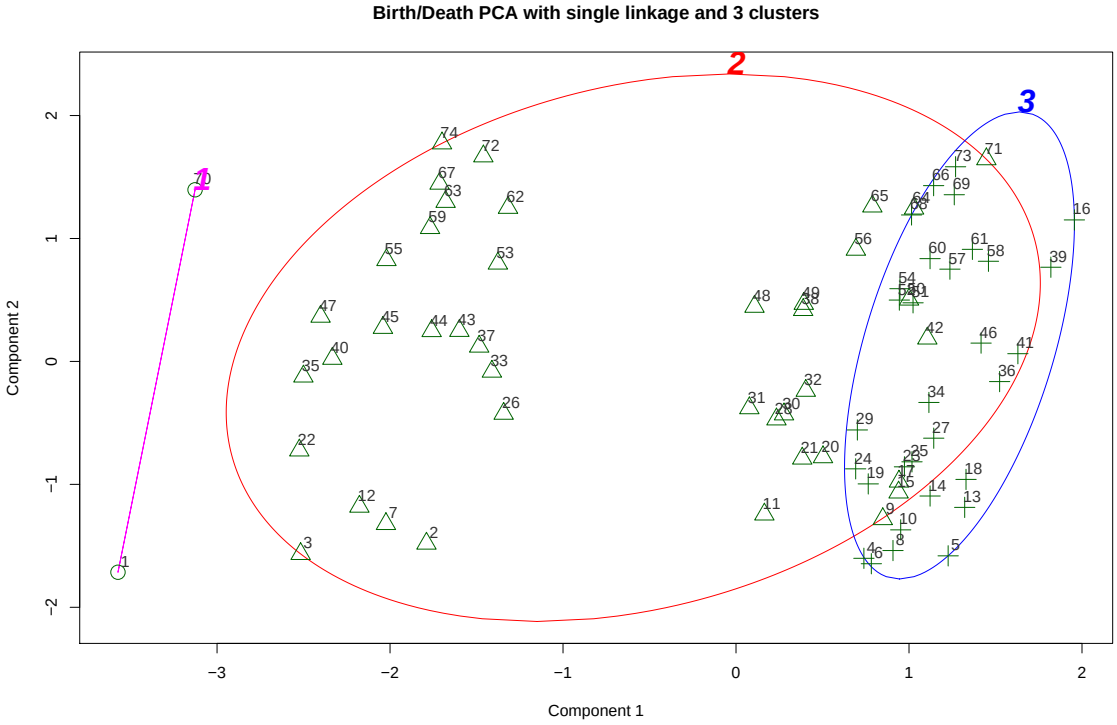
[5] http://en.wikipedia.org/wiki/Four_Asian_Tigers
[6] http://www.povertyeducation.org/the-rise-of-asia.html
[7] http://en.wikipedia.org/wiki/Tiger_Cub_Economies
[8] http://www.investopedia.com/terms/t/tiger-cub-economies.asp




```r
# create distance matrix between points
bd.dist <- dist(bd[,-1])

# number of clusters to identify with red boxes and ellipses
i.clus <- 3

# create dendrogram
bd.hc.single  <- hclust(bd.dist, method = "single")
plot(bd.hc.single, hang = -1
      , main = paste("Teeth with single linkage and", i.clus, "clusters")
      , labels = bd[,1])
rect.hclust(bd.hc.single, k = i.clus)

# create PCA scores plot with ellipses
clusplot(bd, cutree(bd.hc.single, k = i.clus)
        , color = TRUE, labels = 2, lines = 0
        , cex = 2, cex.txt = 1, col.txt = "gray20"
        , main = paste("Birth/Death PCA with single linkage and", i.clus, "clusters")
        , sub = NULL)

# create a column with group membership
```

```
bd$cut.sing <- factor(cutree(bd.hc.single, k = i.clus))
```



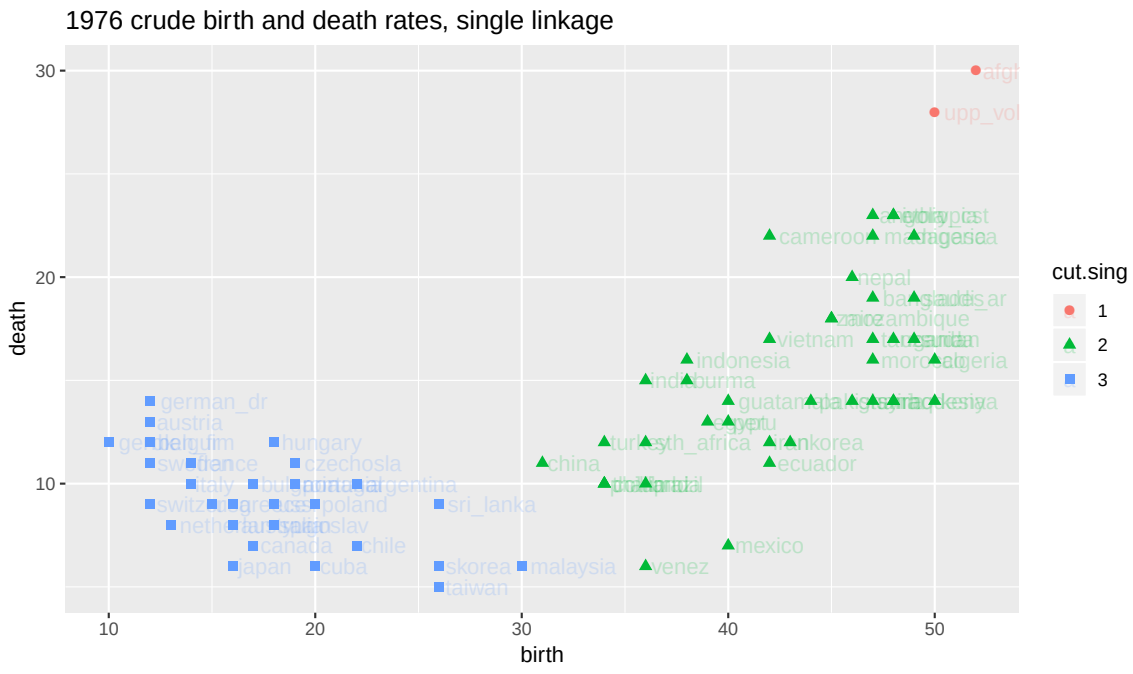



```
# print the observations in each cluster
for (i.cut in 1:i.clus) {
  print(paste("Cluster", i.cut, " ------------------------------ "))
  print(bd[(cutree(bd.hc.single, k = i.clus) == i.cut),])
}
## [1] "Cluster 1  ------------------------------ "
##      country birth death cut.comp cut.sing
## 1     afghan    52    30        1        1
## 70 upp_volta    50    28        1        1
## [1] "Cluster 2  ------------------------------ "
##       country birth death cut.comp cut.sing
## 2     algeria    50    16        1        2
## 3      angola    47    23        1        2
## 7   banglades    47    19        1        2
## 9      brazil    36    10        3        2
## 11      burma    38    15        3        2
## 12   cameroon    42    22        1        2
## 15      china    31    11        3        2
## 17   columbia    34    10        3        2
## 20    ecuador    42    11        3        2
## 21      egypt    39    13        3        2
## 22   ethiopia    48    23        1        2
## 26      ghana    46    14        1        2
## 28  guatamala    40    14        3        2
```

```
## 30      india    36    15        3        2
## 31  indonesia    38    16        3        2
## 32       iran    42    12        3        2
## 33       iraq    48    14        1        2
## 35  ivory_cst    48    23        1        2
## 37      kenya    50    14        1        2
## 38     nkorea    43    12        3        2
## 40  madagasca    47    22        1        2
## 42     mexico    40     7        3        2
## 43    morocco    47    16        1        2
## 44 mozambique    45    18        1        2
## 45      nepal    46    20        1        2
## 47    nigeria    49    22        1        2
## 48   pakistan    44    14        3        2
## 49       peru    40    13        3        2
## 50    phillip    34    10        3        2
## 53   rhodesia    48    14        1        2
## 55   saudi_ar    49    19        1        2
## 56 sth_africa    36    12        3        2
## 59      sudan    49    17        1        2
## 62      syria    47    14        1        2
## 63   tanzania    47    17        1        2
## 64   thailand    34    10        3        2
## 65     turkey    34    12        3        2
## 67     uganda    48    17        1        2
## 71      venez    36     6        3        2
## 72    vietnam    42    17        1        2
## 74      zaire    45    18        1        2
## [1] "Cluster 3  ------------------------------ "
##      country birth death cut.comp cut.sing
## 4  argentina    22    10        2        3
## 5  australia    16     8        2        3
## 6    austria    12    13        2        3
## 8    belguim    12    12        2        3
## 10  bulgaria    17    10        2        3
## 13    canada    17     7        2        3
## 14     chile    22     7        2        3
## 16    taiwan    26     5        3        3
## 18      cuba    20     6        2        3
## 19 czechosla    19    11        2        3
## 23    france    14    11        2        3
## 24 german_dr    12    14        2        3
## 25 german_fr    10    12        2        3
## 27    greece    16     9        2        3
## 29   hungary    18    12        2        3
## 34     italy    14    10        2        3
## 36     japan    16     6        2        3
## 39    skorea    26     6        3        3
```

```
## 41  malaysia    30     6        3        3
## 46 netherlan    13     8        2        3
## 51    poland    20     9        2        3
## 52  portugal    19    10        2        3
## 54   romania    19    10        2        3
## 57     spain    18     8        2        3
## 58 sri_lanka    26     9        3        3
## 60    sweden    12    11        2        3
## 61   switzer    12     9        2        3
## 66      ussr    18     9        2        3
## 68        uk    12    12        2        3
## 69       usa    15     9        2        3
## 73  yugoslav    18     8        2        3
```

```
# plot original data
library(ggplot2)
p1 <- ggplot(bd, aes(x = birth, y = death, colour = cut.sing, shape = cut.sing))
p1 <- p1 + geom_point(size = 2)                                    # points
p1 <- p1 + geom_text(aes(label = country), hjust = -0.1, alpha = 0.2)   # labels
p1 <- p1 + coord_fixed(ratio = 1) # makes 1 unit equal length on x- and y-axis
p1 <- p1 + labs(title = "1976 crude birth and death rates, single linkage")
print(p1)
```



The two methods suggest three clusters. Complete linkage also suggests 14 clusters, but the clusters were unappealing so this analysis will not be presented here.

The three clusters generated by the two methods are very different. The same tendency was observed using average linkage and Ward's method.

An important point to recognize is that different clustering algorithms may agree on the number of clusters, but they may not agree on the composition of the clusters.

Chapter 15

Multivariate Analysis of Variance

Jolicouer and Mosimann studied the relationship between the size and shape of painted turtles. The table below gives the length, width, and height (all in mm) for 24 males and 24 females.

```
#### Example: Painted turtle shells
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch15_shells_mf.dat"
shells <- read.table(fn.data, header = TRUE)
str(shells)
```

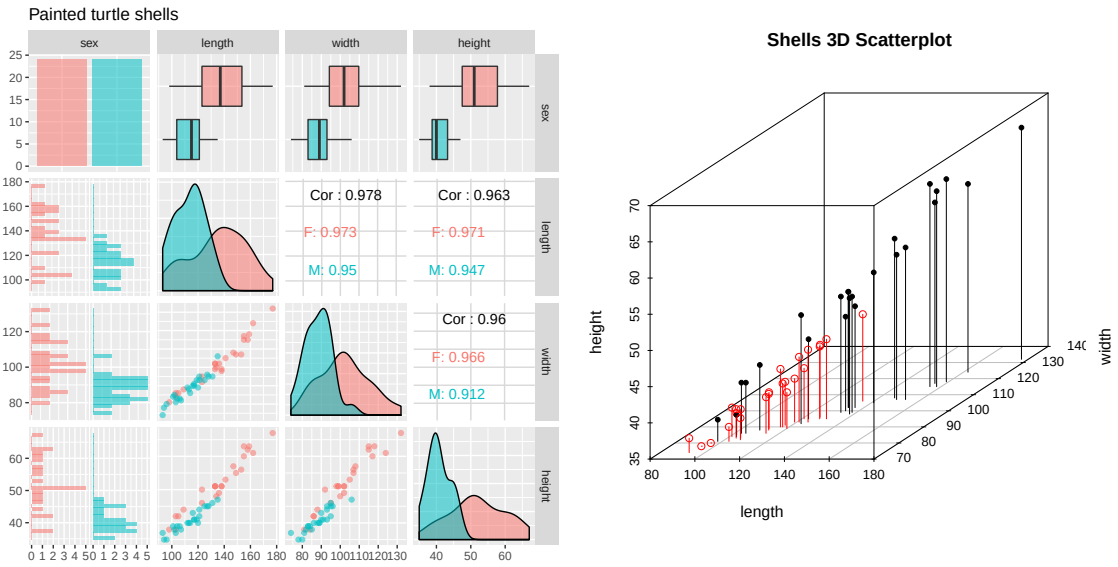
```
## 'data.frame': 48 obs. of  4 variables:
##  $ sex   : Factor w/ 2 levels "F","M": 1 1 1 1 1 1 1 1 1 1 ...
##  $ length: int  98 103 103 105 109 123 123 133 133 133 ...
##  $ width : int  81 84 86 86 88 92 95 99 102 102 ...
##  $ height: int  38 38 42 42 44 50 46 51 51 51 ...
```

```
#head(shells)
```

	sex	length	width	height		sex	length	width	height
1	F	98	81	38	25	M	93	74	37
2	F	103	84	38	26	M	94	78	35
3	F	103	86	42	27	M	96	80	35
4	F	105	86	42	28	M	101	84	39
5	F	109	88	44	29	M	102	85	38
6	F	123	92	50	30	M	103	81	37
7	F	123	95	46	31	M	104	83	39
8	F	133	99	51	32	M	106	83	39
9	F	133	102	51	33	M	107	82	38
10	F	133	102	51	34	M	112	89	40
11	F	134	100	48	35	M	113	88	40
12	F	136	102	49	36	M	114	86	40
13	F	138	98	51	37	M	116	90	43
14	F	138	99	51	38	M	117	90	41
15	F	141	105	53	39	M	117	91	41
16	F	147	108	57	40	M	119	93	41
17	F	149	107	55	41	M	120	89	40
18	F	153	107	56	42	M	121	93	44
19	F	155	115	63	43	M	121	95	42
20	F	155	117	60	44	M	125	93	45
21	F	158	115	62	45	M	127	96	45
22	F	159	118	63	46	M	128	95	45
23	F	162	124	61	47	M	131	95	46
24	F	177	132	67	48	M	135	106	47

```
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
# color by sex
p <- ggpairs(shells
             , mapping = ggplot2::aes(colour = sex, alpha = 0.5)
             , title = "Painted turtle shells"
             , progress=FALSE
            )
print(p)
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)

## 3D scatterplot
library(scatterplot3d)
with(shells, {
  scatterplot3d(x = length
                , y = width
                , z = height
                , main = "Shells 3D Scatterplot"
                , type = "h"                    # lines to the horizontal xy-plane
                , color = as.integer(sex)       # color by group
                , pch = as.integer(sex)+19      # plotting character by group
```

```
                #, highlight.3d = TRUE          # makes color change with z-axis value
                , angle = 40                    # viewing angle (seems hard to control)
                )
  })

#### Try this!
#### For a rotatable 3D plot, use plot3d() from the rgl library
# ##   This uses the R version of the OpenGL (Open Graphics Library)
# library(rgl)
# with(shells, { plot3d(x = length, y = width, z = height, col = sex) })
```



MANOVA considers the following two questions:

- Are the population mean length, width, and height the same for males and females?
- If not, then what combination of features is most responsible for the differences?

To describe MANOVA, suppose you measure p features on independent random samples from k strata, groups, or populations. Let

$$\mu_i' = \begin{bmatrix} \mu_{i1} & \mu_{i2} & \cdots & \mu_{ip} \end{bmatrix}'$$

be the vector of population means for the i^{th} population, where μ_{ij} is the i^{th} population mean on the j^{th} feature. For the turtles, $p = 3$ features and $k = 2$ strata (sexes).

A one-way MANOVA tests the hypothesis that the population **mean vectors** are identical: $H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$ against H_A : not H_0. For the carapace data, you are simultaneously testing that the sexes have equal population mean lengths, equal population mean widths, and equal population mean heights.

Assume that the sample sizes from the different groups are $n_1, n_2, \ldots, n_k$. The

total sample size is $n = n_1 + n_2 + \cdots + n_k$. Let

$$X_{ij}' = \begin{bmatrix} X_{ij1} & X_{ij2} & \cdots & X_{ijp} \end{bmatrix}'$$

be the vector of responses for the j^{th} individual from the i^{th} sample. Let

$$\bar{X}_i' = \begin{bmatrix} \bar{X}_{i1} & \bar{X}_{i2} & \cdots & \bar{X}_{ip} \end{bmatrix}'$$

and S_i be the mean vector and variance-covariance matrix for the i^{th} sample. Finally, let

$$\bar{X}' = \begin{bmatrix} \bar{X}_1 & \bar{X}_2 & \cdots & \bar{X}_p \end{bmatrix}'$$

be the vector of means ignoring samples (combine all the data across samples and compute the average on each feature), and let

$$S = \frac{\sum_i (n_i - 1) S_i}{n - k}$$

be the pooled variance-covariance matrix. The pooled variance-covariance matrix is a weighted average of the variance-covariance matrices from each group.

To test H_0, construct the following MANOVA table, which is the multivariate analog of the ANOVA table:

Source	*df*	SS	MS
Between	$k-1$	$\sum_i n_i(\bar{X}_i - \bar{X})(\bar{X}_i - \bar{X})'$	
Within	$n-k$	$\sum_i (n_i - 1) S_i$	
Total	$n-1$	$\sum_{ij}(X_{ij} - \bar{X})(X_{ij} - \bar{X})'$	

where all the MSs are SS/df.

The expressions for the SS have the same form as SS in univariate analysis of variance, except that each SS is a $p \times p$ symmetric matrix. The diagonal elements of the **SS matrices** are the SS for one-way ANOVAs on the individual features. The off-diagonal elements are SS between features. The Error MS matrix is the pooled variance-covariance matrix S.

The standard MANOVA assumes that you have independent samples from multivariate normal populations with identical variance-covariance matrices. This implies that each feature is normally distributed in each population, that a feature has the same variability across populations, and that the correlation (or covariance) between two features is identical across populations. The Error MS matrix estimates the common population variance-covariance matrix when the population variance-covariance matrices are identical.

The H_0 of equal population mean vectors should be rejected when the difference among mean vectors, as measured by the Between MS matrix, is large relative to the

variability within groups, as measured by the Error MS matrix. Equivalently, H_0 is implausible if a significant portion of the total variation in the data, as measured by the Total SS matrix, is due to differences among the groups. The same idea is used in a one-way ANOVA to motivate the F-test of no differences in population means. However, some care is needed to quantify these ideas in a MANOVA because there are several natural matrix definitions for comparing the Between MS matrix to the Error MS matrix. As a result, several MANOVA tests of H_0 have been proposed.

Graphical summaries for the carapace data are given above, with numerical summaries below.

```
# summary statistics for each sex
by(shells, shells$sex, summary)
## shells$sex: F
##  sex        length          width           height
##  F:24   Min.   : 98.0   Min.   : 81.00   Min.   :38.00
##  M: 0   1st Qu.:123.0   1st Qu.: 94.25   1st Qu.:47.50
##         Median :137.0   Median :102.00   Median :51.00
##         Mean   :136.0   Mean   :102.58   Mean   :52.04
##         3rd Qu.:153.5   3rd Qu.:109.75   3rd Qu.:57.75
##         Max.   :177.0   Max.   :132.00   Max.   :67.00
## ------------------------------------------------------
## shells$sex: M
##  sex        length          width           height
##  F: 0   Min.   : 93.0   Min.   : 74.00   Min.   :35.00
##  M:24   1st Qu.:103.8   1st Qu.: 83.00   1st Qu.:38.75
##         Median :115.0   Median : 89.00   Median :40.00
##         Mean   :113.4   Mean   : 88.29   Mean   :40.71
##         3rd Qu.:121.0   3rd Qu.: 93.00   3rd Qu.:43.25
##         Max.   :135.0   Max.   :106.00   Max.   :47.00
# standard deviations
by(shells[, 2:4], shells$sex, apply, 2, sd)
## shells$sex: F
##   length    width   height
## 21.24900 13.10465  8.04595
## ------------------------------------------------------
## shells$sex: M
##    length     width    height
## 11.806103  7.074013  3.355452
# correlation matrix (excluding associated p-values testing "H0: rho == 0")
library(Hmisc)
rcorr(as.matrix(shells[, 2:4]))[[1]]                    # all
##           length     width    height
## length 1.0000000 0.9778962 0.9628010
## width  0.9778962 1.0000000 0.9599055
## height 0.9628010 0.9599055 1.0000000
rcorr(as.matrix(shells[shells$sex == "F", 2:4]))[[1]]   # females
```

```
##           length     width    height
## length 1.0000000 0.9731162 0.9706748
## width  0.9731162 1.0000000 0.9659029
## height 0.9706748 0.9659029 1.0000000
```

```
rcorr(as.matrix(shells[shells$sex == "M", 2:4]))[[1]]  # males
```

```
##           length     width    height
## length 1.0000000 0.9501287 0.9470730
## width  0.9501287 1.0000000 0.9122648
## height 0.9470730 0.9122648 1.0000000
```

The features are positively correlated within each sex. The correlations between pairs of features are similar for males and females. Females tend to be larger on each feature. The distributions for length, width, and height are fairly symmetric within sexes. No outliers are present. Although females are more variable on each feature than males, the MANOVA assumptions do not appear to be grossly violated here. (Additionally, you could consider transforming the each dimension of the data in hopes to make the covariances between sexes more similar, though it may not be easy to find a good transformation to use.)

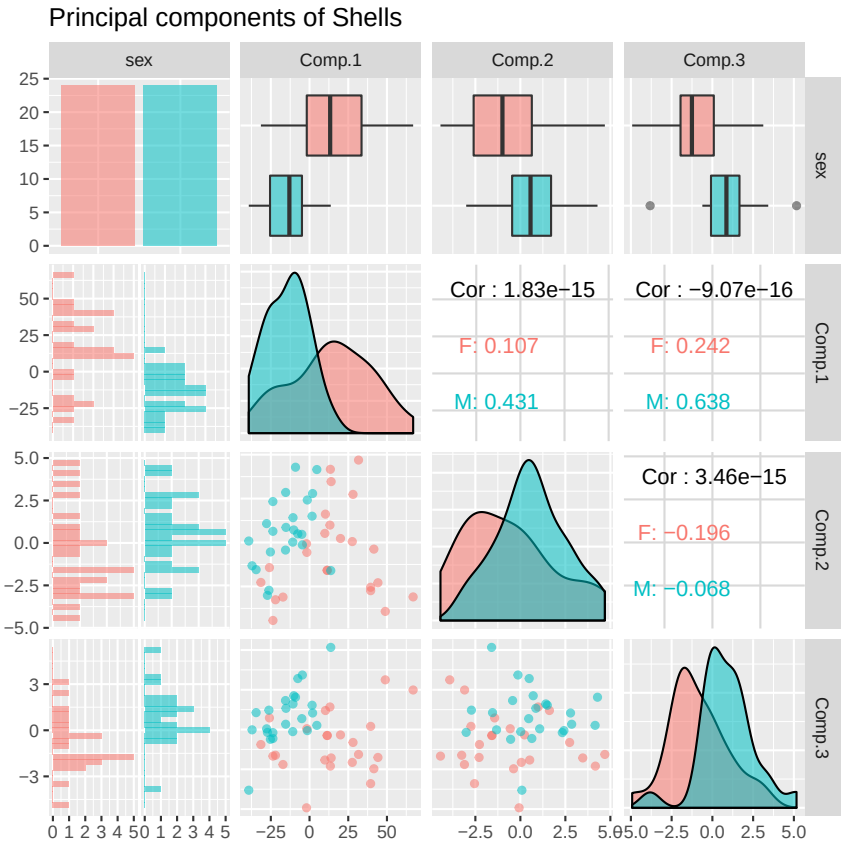
```
pca.sh <- princomp(shells[, 2:4])
df.pca.sh <- data.frame(sex = shells$sex, pca.sh$scores)
str(df.pca.sh)
```

```
## 'data.frame': 48 obs. of  4 variables:
##  $ sex   : Factor w/ 2 levels "F","M": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Comp.1: num  -31.4 -25.9 -23.6 -22 -17.1 ...
##  $ Comp.2: num  -2.27 -1.43 -4.44 -3.26 -3.11 ...
##  $ Comp.3: num  -0.943 0.73 -1.671 -1.618 -2.2 ...
```

```
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
# put scatterplots on top so y axis is vertical
p <- ggpairs(df.pca.sh
             , mapping = ggplot2::aes(colour = sex, alpha = 0.5)
             , title = "Principal components of Shells"
             , progress=FALSE
             )
print(p)
```

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

```
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```



For comparison with the MANOVA below, here are the univariate ANOVAs for each feature. For the carapace data, the univariate ANOVAs indicate significant differences between sexes on length, width, and height. Females are larger on average than males on each feature.

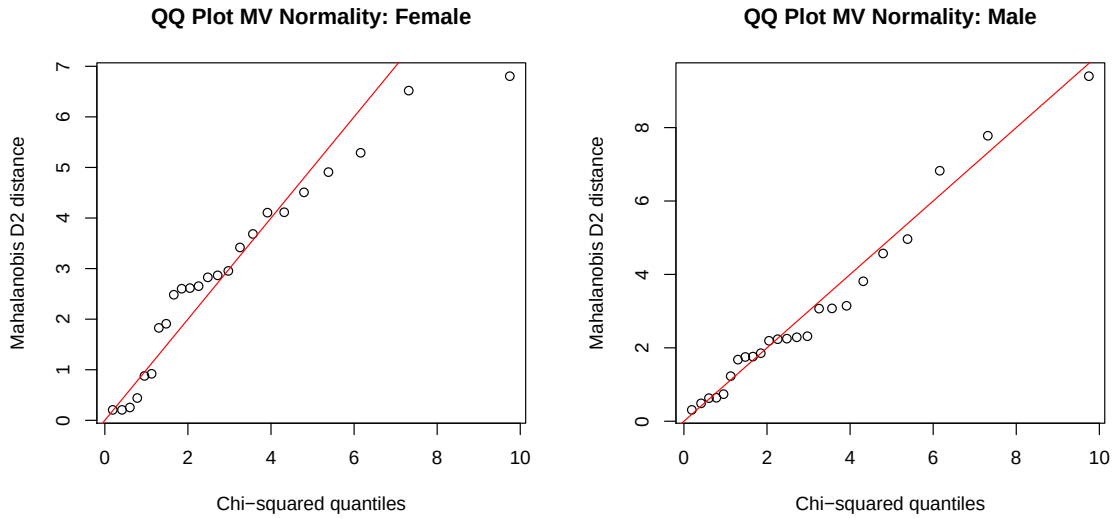
```
# Univariate ANOVA tests, by each response variable
lm.sh <- lm(cbind(length, width, height) ~ sex, data = shells)
summary(lm.sh)
## Response length :
##
## Call:
## lm(formula = length ~ sex, data = shells)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.042 -10.667   1.271  11.927  40.958
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  136.042      3.509   38.77  < 2e-16 ***
## sexM         -22.625      4.962   -4.56 3.79e-05 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.19 on 46 degrees of freedom
## Multiple R-squared:  0.3113,Adjusted R-squared:  0.2963
## F-statistic: 20.79 on 1 and 46 DF,  p-value: 3.788e-05
##
##
## Response width :
##
## Call:
## lm(formula = width ~ sex, data = shells)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -21.5833  -5.5417  -0.4375   4.8854  29.4167
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  102.583      2.149  47.725  < 2e-16 ***
## sexM         -14.292      3.040  -4.701 2.38e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.53 on 46 degrees of freedom
## Multiple R-squared:  0.3246,Adjusted R-squared:  0.3099
## F-statistic:  22.1 on 1 and 46 DF,  p-value: 2.376e-05
##
##
## Response height :
##
## Call:
## lm(formula = height ~ sex, data = shells)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.0417  -2.7917  -0.7083   4.0417  14.9583
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   52.042      1.258  41.360  < 2e-16 ***
## sexM         -11.333      1.779  -6.369 8.09e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.164 on 46 degrees of freedom
## Multiple R-squared:  0.4686,Adjusted R-squared:  0.457
## F-statistic: 40.56 on 1 and 46 DF,  p-value: 8.087e-08
```

```
# Alternatively, for many ANOVAs at once, it may be easier
#   to select by column number, but you won't get the column names in the output.
#   Also, the left-hand side needs to be a matrix data type.
# lm.sh <- lm(as.matrix(shells[, 2:4]) ~ shells[, 1])
```

A few procedures can be used for one-way MANOVA; two are `manova()` and the `car` package's `Manova()`. First we check the assumption of multivariate normality.

```
# Test multivariate normality using the Shapiro-Wilk test for multivariate normality
library(mvnormtest)
# The data needs to be transposed t() so each variable is a row
#   with observations as columns.

mshapiro.test(t(shells[shells$sex == "F", 2:4]))
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.89324, p-value = 0.01551
mshapiro.test(t(shells[shells$sex == "M", 2:4]))
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.93602, p-value = 0.1329
# Graphical Assessment of Multivariate Normality
f.mnv.norm.qqplot <- function(x, name = "") {
  # creates a QQ-plot for assessing multivariate normality

  x <- as.matrix(x)         # n x p numeric matrix
  center <- colMeans(x)     # centroid
  n <- nrow(x);
  p <- ncol(x);
  cov <- cov(x);
  d <- mahalanobis(x, center, cov) # distances
  qqplot(qchisq(ppoints(n), df=p), d
    , main=paste("QQ Plot MV Normality:", name)
    , ylab="Mahalanobis D2 distance"
    , xlab="Chi-squared quantiles")
  abline(a = 0, b = 1, col = "red")
}

f.mnv.norm.qqplot(shells[shells$sex == "F", 2:4], "Female")
f.mnv.norm.qqplot(shells[shells$sex == "M", 2:4], "Male")
```



The curvature in the Famale sample cause us to reject normality, while the males do not deviate from normality. We'll proceed anyway since this deviation from normality in the female sample will largely increase the variability of the sample and not displace the mean greatly, and the sample sizes are somewhat large.

Multivariate test statistics These four multivariate test statistics are among the most common to assess differences across the levels of the categorical variables for a linear combination of responses. In general Wilks' lambda is recommended unless there are problems with small total sample size, unequal sample sizes between groups, violations of assumptions, etc., in which case Pillai's trace is more robust.

Wilks' lambda, (λ)

- Most commonly used statistic for overall significance
- Considers differences over all the characteristic roots
- The smaller the value of Wilks' lambda, the larger the between-groups dispersion

Pillai's trace

- Considers differences over all the characteristic roots
- More robust than Wilks'; should be used when sample size decreases, unequal cell sizes or homogeneity of covariances is violated

Hotelling's trace

- Considers differences over all the characteristic roots

Roy's greatest characteristic root

- Tests for differences on only the first discriminant function (Chapter 16)
- Most appropriate when responses are strongly interrelated on a single dimension

- Highly sensitive to violation of assumptions, but most powerful when all assumptions are met.

```
# Multivariate MANOVA test
#   the specific test is specified in summary()
#           test = c("Pillai", "Wilks", "Hotelling-Lawley", "Roy")
man.sh <- manova(cbind(length, width, height) ~ sex, data = shells)
summary(man.sh, test="Wilks")
##           Df   Wilks approx F num Df den Df    Pr(>F)
## sex        1 0.38695   23.237      3     44 3.622e-09 ***
## Residuals 46
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# I prefer the output from the car package
library(car)
lm.man <- lm(cbind(length, width, height) ~ sex, data = shells)
man.sh <- Manova(lm.man)
summary(man.sh)
##
## Type II MANOVA Tests:
##
## Sum of squares and products for error:
##           length    width   height
## length 13590.792 8057.500 4679.875
## width   8057.500 5100.792 2840.458
## height  4679.875 2840.458 1747.917
##
## ------------------------------------------
##
## Term: sex
##
## Sum of squares and products for the hypothesis:
##          length    width   height
## length 6142.688 3880.188 3077.000
## width  3880.188 2451.021 1943.667
## height 3077.000 1943.667 1541.333
##
## Multivariate Tests: sex
##                   Df test stat approx F num Df den Df    Pr(>F)
## Pillai             1 0.6130506 23.23665      3     44 3.622e-09 ***
## Wilks              1 0.3869494 23.23665      3     44 3.622e-09 ***
## Hotelling-Lawley   1 1.5843173 23.23665      3     44 3.622e-09 ***
## Roy                1 1.5843173 23.23665      3     44 3.622e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The four MANOVA tests of no differences between sexes are all highly significant. These tests reinforce the univariate analyses. Of the four tests, I prefer Roy's test

because it has an intuitive interpretation. I will mostly ignore the other three tests for discussion.

Roy's test locates the linear combination of the features that produces the most significant one-way ANOVA test for no differences among groups. If the groups are not significantly different on the linear combination that best separates the groups in a one-way ANOVA sense, then there is no evidence that the population mean vectors are different. The critical value for Roy's test accounts for the linear combination being suggested by the data. That is, the critical value for Roy's test is not the same critical value that is used in a one-way ANOVA. The idea is similar to a Bonferroni-type correction with multiple comparisons.

Roy's method has the ability to locate linear combinations of the features on which the groups differ, even when the differences across groups are not significant on any feature. This is a reason for treating multivariate problems using multivariate methods rather than through individual univariate analyses on each feature.

```
## For Roy's characteristic Root and vector
#str(man.sh)
H <- man.sh$SSP$sex  # H = hypothesis matrix
#    man.sh£df       #     hypothesis df
E <- man.sh$SSPE     # E = error matrix
#    man.sh£error.df #     error df

# characteristic roots of (E inverse * H)
EinvH <- solve(E) %*% H  # solve() computes the matrix inverse
ev <- eigen(EinvH)       # eigenvalue/eigenvectors
ev
## eigen() decomposition
## $values
## [1]  1.584317e+00  1.401030e-15 -1.315838e-15
##
## $vectors
##            [,1]       [,2]        [,3]
## [1,]  0.2151877  0.3563618  0.02975353
## [2,]  0.1014317  0.1651253  0.59173018
## [3,] -0.9712908 -0.9196412 -0.80558682
```

The **characteristic vector** (eigenvector) output gives one linear combination of the features for each variable in the data set. By construction, the linear combinations are uncorrelated (adjusting for groups). In general, the first $a = \text{minimum}(p, k-1)$ linear combinations contain information in decreasing amounts for distinguishing among the groups. The first linear combination is used by Roy's test.

The three linear combinations for the carapace data are (reading down the columns

in the matrix of eigenvectors)

$$\begin{aligned} \text{D1} &= 0.2152\ \text{Length} + 0.1014\ \text{Width} + -0.9713\ \text{Height} \\ \text{D2} &= 0.3564\ \text{Length} + 0.1651\ \text{Width} + -0.9196\ \text{Height} \\ \text{D3} &= 0.02975\ \text{Length} + 0.5917\ \text{Width} + -0.8056\ \text{Height}. \end{aligned}$$

Here $p = 3$ and $k = 2$ gives $a = \min(p, k-1) = \min(3, 2-1) = 1$ so only D1 from (D1, D2, and D3) contains information for distinguishing between male and female painted turtles.

As in PCA, the linear combinations should be interpreted. However, do not discount the contribution of a feature with a small loading (though, in the shells example, D2 and D3 have 0 for loadings). In particular, the most important feature for distinguishing among the groups might have a small loading because of the measurement scale.

The following output shows the differences between sexes on D1. The separation between sexes on D1 is greater than on any single feature. Females typically have much larger D1 scores than males.

Since the matrix of eigenvectors are a rotation matrix[1] we can create the D linear combinations by matrix multiplication of the eigenvector (rotation) matrix with the original data (being careful about dimensions).

```
# linear combinations of features
D <- as.matrix(shells[,2:4]) %*% ev$vectors
colnames(D) <- c("D1", "D2", "D3")
df.D <- data.frame(sex = shells$sex, D)
str(df.D)
```

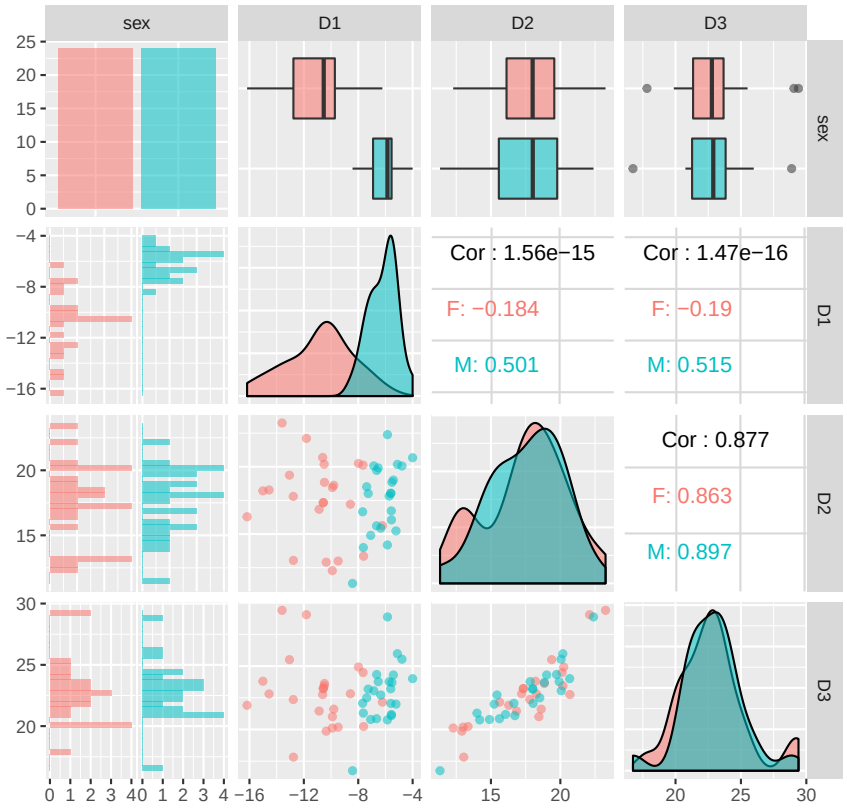
```
## 'data.frame': 48 obs. of  4 variables:
##  $ sex: Factor w/ 2 levels "F","M": 1 1 1 1 1 1 1 1 1 1 ...
##  $ D1 : num  -7.6 -6.22 -9.91 -9.48 -10.36 ...
##  $ D2 : num  13.4 15.6 12.3 13 12.9 ...
##  $ D3 : num  20.2 22.2 20.1 20.2 19.9 ...
```

```
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
# put scatterplots on top so y axis is vertical
p <- ggpairs(df.D
             , mapping = ggplot2::aes(colour = sex, alpha = 0.5)
             , title = "D1 is the linear combination that best distinguishes the sexes"
             , progress=FALSE
              )
print(p)
```

[1] http://en.wikipedia.org/wiki/Rotation_matrix

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```

D1 is the linear combination that best distinguishes the sexes




```
# Univariate ANOVA tests, by D1 linear combination variable
lm.D.sh <- lm(D1 ~ sex, data = df.D)
summary(lm.D.sh)
```

```
##
## Call:
## lm(formula = D1 ~ sex, data = df.D)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.3047 -0.9791  0.3260  0.9570  4.6434
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -10.8679     0.3884 -27.978  < 2e-16 ***
```

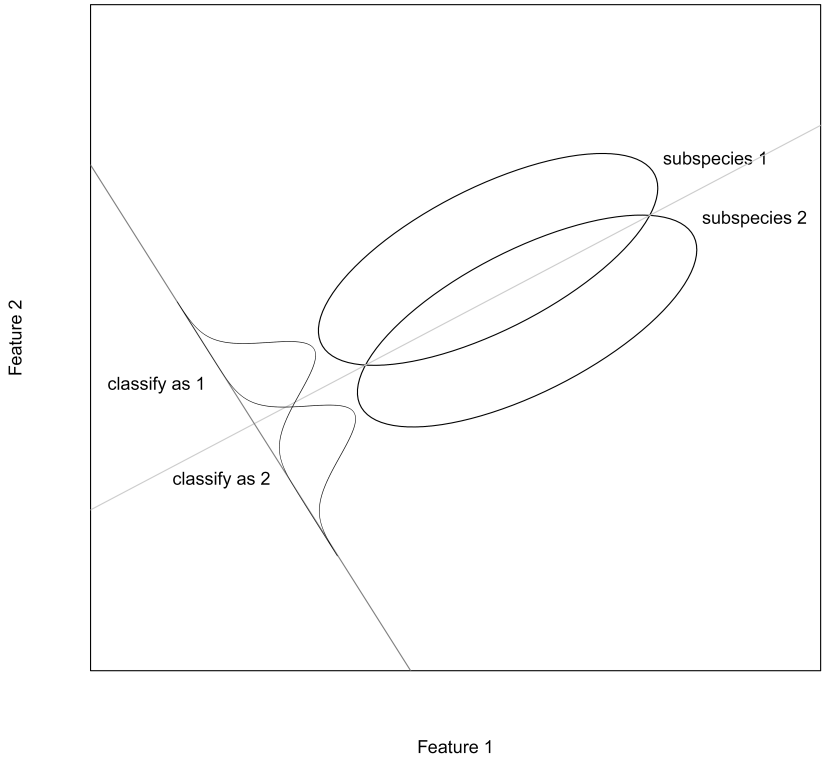
```
## sexM          4.6897     0.5493   8.537 4.84e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.903 on 46 degrees of freedom
## Multiple R-squared:  0.6131,Adjusted R-squared:  0.6046
## F-statistic: 72.88 on 1 and 46 DF,  p-value: 4.841e-11
```

Chapter 16

Discriminant Analysis

A researcher collected data on two external features for two (known) sub-species of an insect. She can use **discriminant analysis** to find linear combinations of the features that best distinguish the sub-species. The analysis can then be used to classify insects with unknown sub-species origin into one of the two sub-species based on their external features.

To see how this might be done, consider the following data plot. Can1 is the linear combination of the two features that best distinguishes or discriminates the two sub-species. The value of Can1 could be used to classify insects into one of the two groups, as illustrated.



The method generalizes to more than two features and sub-species.

16.1 Canonical Discriminant Analysis

While there's a connection between **canonical discriminant analysis** and **canonical correlation**, I prefer to emphasize the connection between canonical discriminant analysis and MANOVA because these techniques are essentially identical.

Assume that you have representative samples from k groups, strata, or subpopulations. Each selected individual is measured on p features (measurements) $X_1, X_2, \ldots, X_p$. As in MANOVA, canonical discriminant analysis assumes you have independent samples from multivariate normal populations with identical variance-covariance matrices.

Canonical discriminant analysis computes $r = \min(p, k-1)$ linear combinations of the features with the following properties. The first linear combination, called the **first linear discriminant function**

$$\text{Can1} = a_{11}X_1 + a_{12}X_2 + \cdots + a_{1p}X_p$$

gives the most significant F-test for a null hypothesis of no group differences in a one-way ANOVA, among all linear combinations of the features. The second linear combination or the **second linear discriminant function**:

$$\text{Can2} = a_{21}X_1 + a_{22}X_2 + \cdots + a_{2p}X_p$$

gives the most significant F-test for no group differences in a one-way ANOVA, among all linear combinations of the features that are uncorrelated (adjusting for groups) with Can1. In general, the j^{th} linear combination Can$j$ $(j = 1, 2, \ldots, r)$ gives the most significant F-test for no group differences in a one-way ANOVA, among all linear combinations of the features that are uncorrelated with $\text{Can1}, \text{Can2}, \ldots, \text{Can}(j-1)$.

The coefficients in the canonical discriminant functions can be multiplied by a constant, or all the signs can be changed (that is, multiplied by the constant -1), without changing their properties or interpretations.

16.2 Example: Owners of riding mowers

The manufacturer of a riding lawn mower wishes to identify the best prospects for buying their product using data on the incomes (X_1) and lot sizes (X_2) of homeowners (Johnson and Wichern, 1988). The data below are the incomes and lot sizes from independent random samples of 12 current owners and 12 non-owners of the mowers.

```
#### Example: Riding mowers
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch16_mower.dat"
mower <- read.table(fn.data, header = TRUE)
# income  = income in £1000
# lotsize = lot size in 1000 sq ft
# owner   = nonowners or owners
str(mower)
```

```
## 'data.frame': 24 obs. of  3 variables:
##  $ income : num  20 28.5 21.6 20.5 29 36.7 36 27.6 23 31 ...
##  $ lotsize: num  9.2 8.4 10.8 10.4 11.8 9.6 8.8 11.2 10 10.4 ...
##  $ owner  : Factor w/ 2 levels "nonowner","owner": 2 2 2 2 2 2 2 2 2 2 ...
```

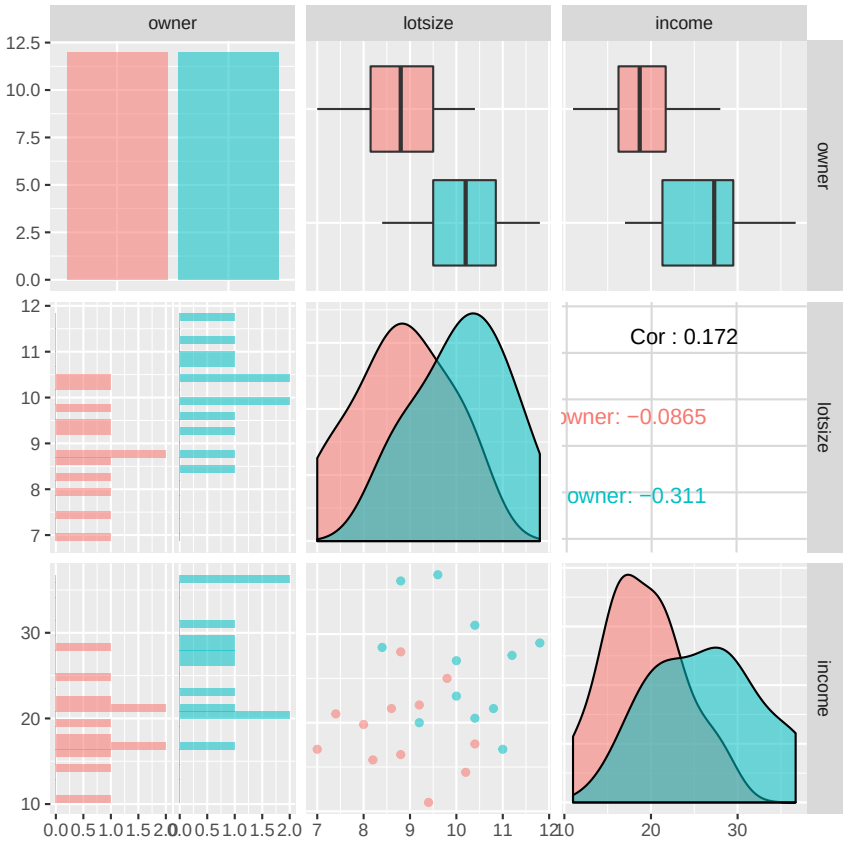
	income	lotsize	owner		income	lotsize	owner
1	20.00	9.20	owner	13	25.00	9.80	nonowner
2	28.50	8.40	owner	14	17.60	10.40	nonowner
3	21.60	10.80	owner	15	21.60	8.60	nonowner
4	20.50	10.40	owner	16	14.40	10.20	nonowner
5	29.00	11.80	owner	17	28.00	8.80	nonowner
6	36.70	9.60	owner	18	19.80	8.00	nonowner
7	36.00	8.80	owner	19	22.00	9.20	nonowner
8	27.60	11.20	owner	20	15.80	8.20	nonowner
9	23.00	10.00	owner	21	11.00	9.40	nonowner
10	31.00	10.40	owner	22	17.00	7.00	nonowner
11	17.00	11.00	owner	23	16.40	8.80	nonowner
12	27.00	10.00	owner	24	21.00	7.40	nonowner

```
library(ggplot2)

p <- ggplot(mower, aes(x = income, y = lotsize, shape = owner, colour = owner))
p <- p + geom_point(size = 3)
p <- p + scale_y_continuous(limits = c(0, 15))
p <- p + scale_x_continuous(limits = c(0, 40))
p <- p + coord_fixed(ratio = 1)                # square axes (for perp lines)
p <- p + xlab("income in $1000")
p <- p + ylab("lot size in 1000 sq ft")
print(p)
```


```
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
p <- ggpairs(rev(mower)
             , mapping = ggplot2::aes(colour = owner, alpha = 0.5)
             , progress=FALSE
              )
print(p)
```

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

```
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```



Although the two groups overlap, the owners tend to have higher incomes and larger lots than the non-owners. Income seems to distinguish owners and non-owners better than lot size, but both variables seem to be useful for discriminating between groups.

Qualitatively, one might classify prospects based on their location relative to a roughly vertical line on the scatter plot. A discriminant analysis gives similar results to this heuristic approach because the Can1 scores will roughly correspond to the projection of the two features onto a line perpendicular to the hypothetical vertical line. `candisc()` computes one discriminant function here because $p = 2$ and $k = 2$ gives $r = \min(p, k-1) = \min(2, 1) = 1$.

Below we first fit a `lm()` and use that object to compare populations. First we compare using univariate ANOVAs. The p-values are for one-way ANOVA comparing owners to non-owners and both income and lotsize features are important individually for distinguishing between the groups.

```
# first fit lm() with formula = continuous variables ~ factor variables
lm.mower <- lm(cbind(income, lotsize) ~ owner, data = mower)

# univariate ANOVA tests
```

```
summary(lm.mower)
## Response income :
##
## Call:
## lm(formula = income ~ owner, data = mower)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.4917 -3.8021  0.5875  2.5979 10.2083
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   19.133      1.601  11.954 4.28e-11 ***
## ownerowner     7.358      2.264   3.251  0.00367 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.545 on 22 degrees of freedom
## Multiple R-squared:  0.3245,Adjusted R-squared:  0.2938
## F-statistic: 10.57 on 1 and 22 DF,  p-value: 0.003665
##
##
## Response lotsize :
##
## Call:
## lm(formula = lotsize ~ owner, data = mower)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.81667 -0.66667 -0.01667  0.71667  1.66667
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.8167     0.2984   29.55  < 2e-16 ***
## ownerowner    1.3167     0.4220    3.12  0.00498 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.034 on 22 degrees of freedom
## Multiple R-squared:  0.3068,Adjusted R-squared:  0.2753
## F-statistic: 9.736 on 1 and 22 DF,  p-value: 0.004983
```

Second, the MANOVA indicates the multivariate means are different indicating both income and lotsize features taken together are important for distinguishing between the groups.

```
# test whether the multivariate means of the two populations are different
library(car)
```

```
man.mo <- Manova(lm.mower)
summary(man.mo)
##
## Type II MANOVA Tests:
##
## Sum of squares and products for error:
##            income   lotsize
## income  676.31583 -26.41333
## lotsize -26.41333  23.50333
##
## ----------------------------------------
##
## Term: owner
##
## Sum of squares and products for the hypothesis:
##            income  lotsize
## income  324.87042 58.13083
## lotsize  58.13083 10.40167
##
## Multivariate Tests: owner
##                  Df test stat approx F num Df den Df     Pr(>F)
## Pillai            1 0.5386044 12.25704      2     21 0.00029701 ***
## Wilks             1 0.4613956 12.25704      2     21 0.00029701 ***
## Hotelling-Lawley  1 1.1673374 12.25704      2     21 0.00029701 ***
## Roy               1 1.1673374 12.25704      2     21 0.00029701 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Finally, we fit the canonical discriminant function with `candisc()`. The LR (likelihood ratio) p-values below correspond to tests of no differences between groups on the canonical discriminant functions. There is only one canonical discriminant function here. The tests of no differences based on the first canonical discriminant function is equivalent to Roy's MANOVA test.

```
# perform canonical discriminant analysis
library(candisc)
## Loading required package:  heplots
##
## Attaching package:  'candisc'
## The following object is masked from 'package:stats':
##
##    cancor
can.mower <- candisc(lm.mower)
can.mower
##
## Canonical Discriminant Analysis for owner:
##
##   CanRsq Eigenvalue Difference Percent Cumulative
```

```
## 1 0.5386     1.1673                100        100
##
## Test of H0: The canonical correlations in the
## current row and all that follow are zero
##
##   LR test stat approx F numDF denDF Pr(> F)
## 1       0.4614              2
```

The objects available from the `candisc()` object are named below, and we'll soon use a few. There are also a few plots available, but I'll be creating other plots shortly.

```
names(can.mower) # list of objects in can.mower
```

```
##  [1] "dfh"         "dfe"         "eigenvalues" "canrsq"
##  [5] "pct"         "rank"        "ndim"        "means"
##  [9] "factors"     "term"        "terms"       "coeffs.raw"
## [13] "coeffs.std"  "structure"   "scores"
```

```
# plot(can.mower) # this plot causes Rnw compile errors
                  # it would show box plots
                  # with proportional contribution of each variable to Can1

### can also plot 2D plots when have more than two groups (will use later)
## library(heplots)
#heplot(can.mower, scale=6, fill=TRUE)
#heplot3d(can.mower, scale=6, fill=TRUE)
```

The raw canonical coefficients define the canonical discriminant variables and are identical to the feature loadings in a one-way MANOVA, except for an unimportant multiplicative factor. Only Can1 is generated here.

```
can.mower$coeffs.raw
```

```
##               Can1
## income  -0.1453404
## lotsize -0.7590457
```

The means output gives the mean score on the canonical discriminant variables by group, after centering the scores to have mean zero over all groups. These are in order of the owner factor levels (nonowner, owner).

```
can.mower$means
```

```
## [1]  1.034437 -1.034437
```

The linear combination of income and lotsize that best distinguishes owners from non-owners

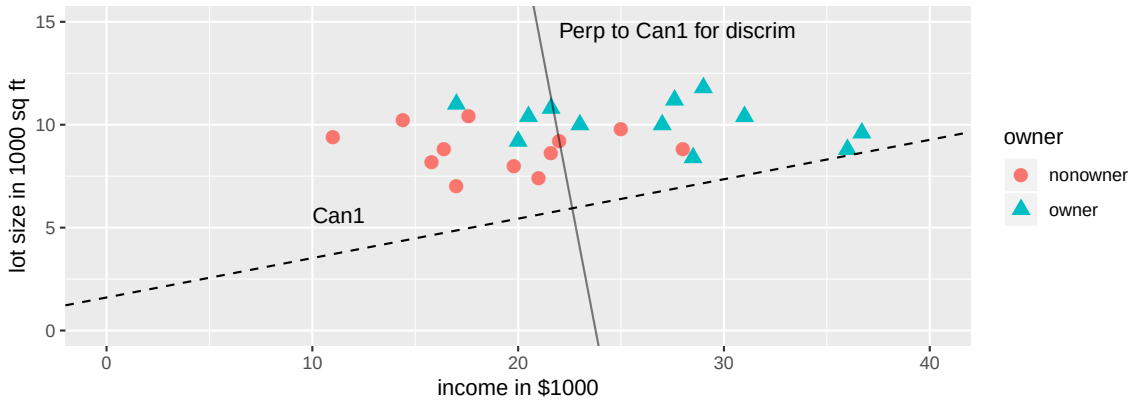
$$\text{Can1} = -0.1453 \text{ INCOME} + -0.759 \text{ LOTSIZE}$$

is a weighted average of income and lotsize.

In the scatterplot below, Can1 is the direction indicated by the dashed line.

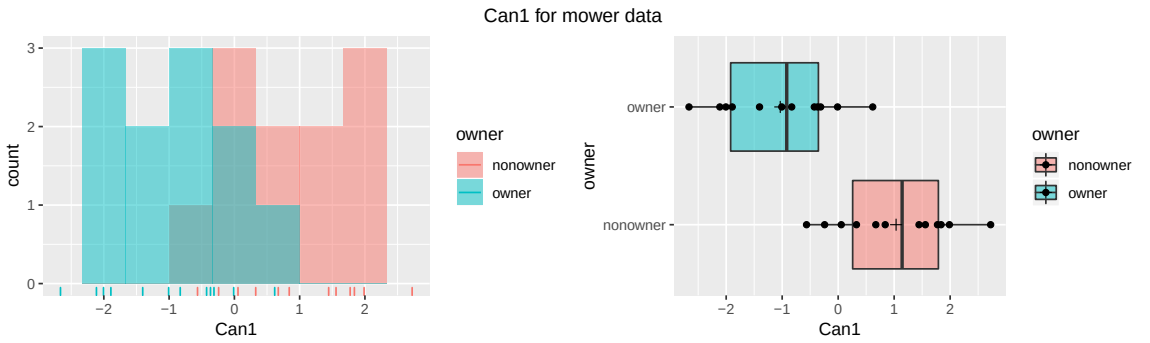
```
library(ggplot2)
```

```
# Scatterplots with Can1 line overlayed
p <- ggplot(mower, aes(x = income, y = lotsize, shape = owner, colour = owner))
p <- p + geom_point(size = 3)

  # use a little algebra to determine the intercept and slopes of the
  #   Can1 line and a line perpendicular to it.

  # dashed line of Can1
  b1 <- can.mower$coeffs.raw[1]/can.mower$coeffs.raw[2]     # slope
  a1 <- mean(mower$lotsize) - b1 * mean(mower$income) - 3.5 # intercept
p <- p + geom_abline(intercept = a1, slope = b1, linetype = 2)
p <- p + annotate("text", x = 10, y = 6, label = "Can1"
                , hjust = 0, vjust = 1, size = 4)

  # solid line to separate groups (perpendicular to Can1)
  b2 <- -can.mower$coeffs.raw[2]/can.mower$coeffs.raw[1]  # slope
  a2 <- mean(mower$lotsize) - b2 * mean(mower$income) - 4.5    # intercept
p <- p + geom_abline(intercept = a2, slope = b2, linetype = 1, alpha = 0.5)
p <- p + annotate("text", x = 22, y = 15, label = "Perp to Can1 for discrim"
                , hjust = 0, vjust = 1, size = 4)

p <- p + scale_y_continuous(limits = c(0, 15))
p <- p + scale_x_continuous(limits = c(0, 40))
p <- p + coord_fixed(ratio = 1)                # square axes (for perp lines)
p <- p + xlab("income in $1000")
p <- p + ylab("lot size in 1000 sq ft")
print(p)
```




```
# Plots of Can1
p1 <- ggplot(can.mower$scores, aes(x = Can1, fill = owner))
p1 <- p1 + geom_histogram(binwidth = 2/3, alpha = 0.5, position="identity")
p1 <- p1 + scale_x_continuous(limits = c(min(can.mower$scores$Can1), max(can.mower$scores$Can1)))
p1 <- p1 + geom_rug(aes(colour = owner))
#p1 <- p1 + labs(title = "Can1 for mower data")
#print(p1)

p2 <- ggplot(can.mower$scores, aes(y = Can1, x = owner, fill = owner))
p2 <- p2 + geom_boxplot(alpha = 0.5)
          # add a "+" at the mean
p2 <- p2 + stat_summary(fun.y = mean, geom = "point", shape = 3, size = 2)
```

```
p2 <- p2 + geom_point()
p2 <- p2 + coord_flip()
p2 <- p2 + scale_y_continuous(limits = c(min(can.mower$scores$Can1), max(can.mower$scores$Can1)))
#p2 <- p2 + labs(title = "Can1 for mower data")
#print(p2)

library(gridExtra)
grid.arrange(grobs = list(p1, p2), ncol=2, top = "Can1 for mower data")
## Warning:  Removed 2 rows containing missing values (geom_bar).
```



The standardized coefficients (use the pooled within-class coefficients) indicate the relative contributions of the features to the discrimination. The standardized coefficients are roughly equal, which suggests that income and lotsize contribute similarly to distinguishing the owners from non-owners.

```
can.mower$coeffs.std
##               Can1
## income   -0.8058419
## lotsize -0.7845512
```

The p-value of 0.0004 on the likelihood ratio test indicates that Can1 strongly distinguishes between owners and non-owners. This is consistent with the separation between owners and non-owners in the boxplot of Can1 scores.

I noted above that Can1 is essentially the same linear combination given in a MANOVA comparison of owners to non-owners. Here is some `Manova()` output to support this claim. The MANOVA test p-values agree with the `candisc` output (as we saw earlier). The first characteristic vector from the MANOVA is given here.

```
## For Roy's characteristic Root and vector
H <- man.mo$SSP$owner # H = hypothesis matrix
E <- man.mo$SSPE      # E = error matrix
# characteristic roots of (E inverse * H)
EinvH <- solve(E) %*% H  # solve() computes the matrix inverse
ev <- eigen(EinvH)       # eigenvalue/eigenvectors
ev
## eigen() decomposition
## $values
## [1] 1.167337 0.000000
##
## $vectors
##           [,1]       [,2]
```

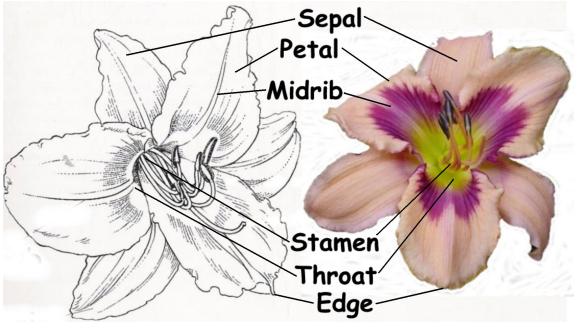
```
## [1,] 0.1880613 -0.1761379
## [2,] 0.9821573  0.9843655
mult.char.can.disc <- can.mower$coeffs.raw[1] / ev$vectors[1,1]
mult.char.can.disc
## [1] -0.7728352
```

The first canonical discriminant function is obtained by multiplying the first characteristic vector given in MANOVA by -0.7728:

$$\begin{aligned} \text{Can1} &= -0.1453\ \text{INCOME} + -0.759\ \text{LOTSIZE} \\ &= -0.7728\,(0.1881\ \text{INCOME} + 0.9822\ \text{LOTSIZE}) \end{aligned}$$

16.3 Discriminant Analysis on Fisher's Iris Data

Fisher's iris data consists of samples of 50 flowers from each of three species of iris: Setosa, Versicolor, and Virginica. Four measurements (in mm) were taken on each flower: sepal length, sepal width, petal length, and petal width.



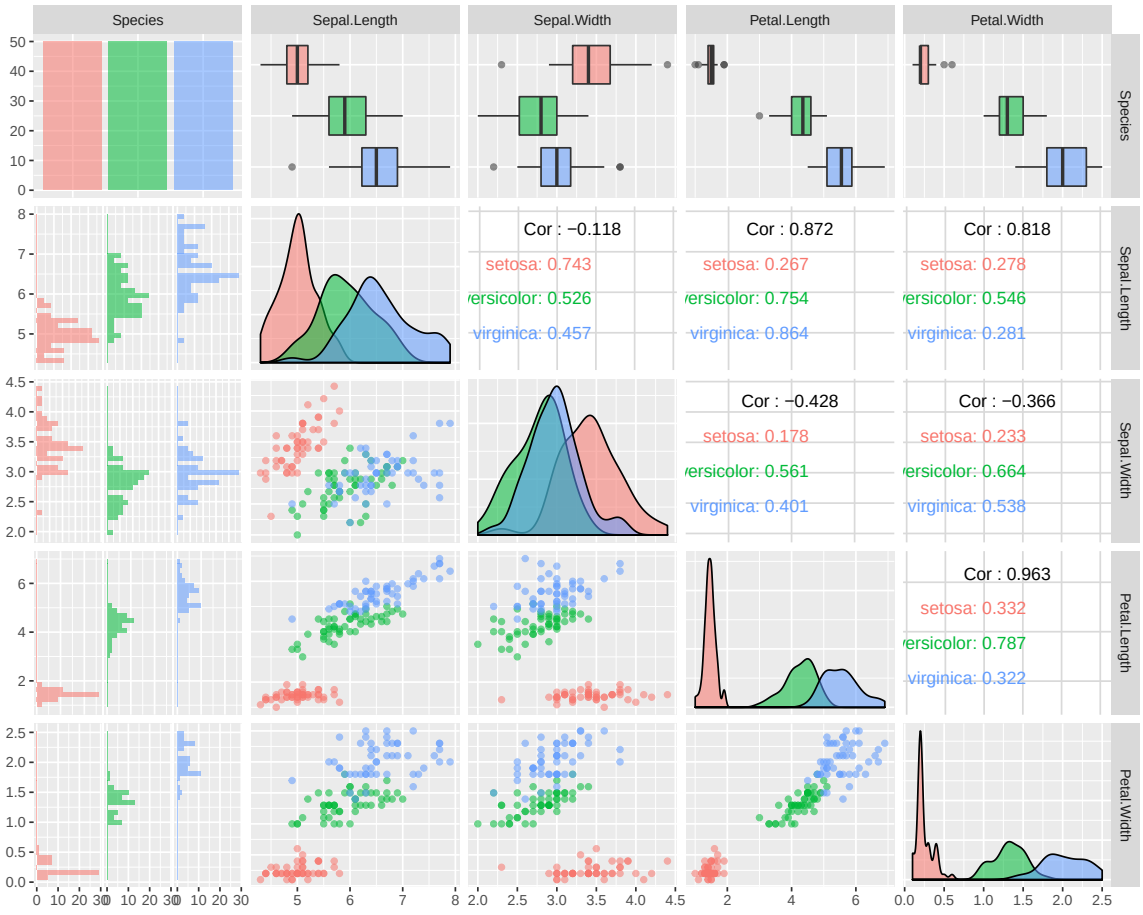
The plots show big differences between Setosa and the other two species. The differences between Versicolor and Virginica are smaller, and appear to be mostly due to differences in the petal widths and lengths.

```
#### Example: Fisher's iris data
# The "iris" dataset is included with R in the library(datasets)
data(iris)
str(iris)
## 'data.frame': 150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
```

```
library(GGally)
p <- ggpairs(iris[,c(5,1,2,3,4)]
             , mapping = ggplot2::aes(colour = Species, alpha = 0.5)
             , progress=FALSE
              )
print(p)
```

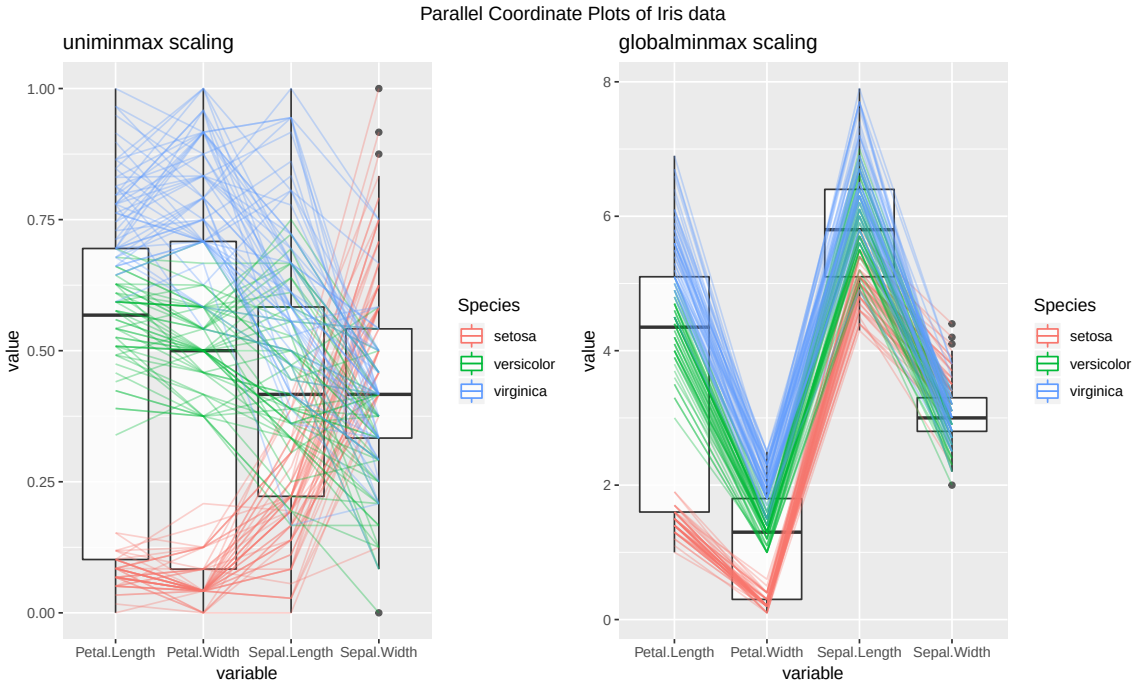
```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```

```
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```




```
## parallel coordinate plot
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
```

```
# univariate min/max scaling
p1 <- ggparcoord(
  data = iris
  , columns = 1:4
  , groupColumn = 5
  , order = "anyClass"
  , scale = "uniminmax" # "uniminmax". "globalminmax"
  , showPoints = FALSE
  , title = "uniminmax scaling"
  , alphaLines = 1/3
  #, shadeBox = "white"
  , boxplot = TRUE
  ) #+ theme_bw()

# global min/max scaling
p2 <- ggparcoord(
  data = iris
  , columns = 1:4
  , groupColumn = 5
  , order = "anyClass"
  , scale = "globalminmax" # "uniminmax". "globalminmax"
  , showPoints = FALSE
  , title = "globalminmax scaling"
  , alphaLines = 1/3
  #, shadeBox = "white"
  , boxplot = TRUE
  ) #+ theme_bw()

library(gridExtra)
grid.arrange(grobs = list(p1, p2), ncol=2, top = "Parallel Coordinate Plots of Iris data")

  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```



`candisc` was used to discriminate among species. There are $k = 3$ species and $p = 4$ features, so the number of discriminant functions is 2 (the minimum of 4 and $3 - 1$).

```
# first fit lm() with formula = continuous variables ~ factor variables
lm.iris <- lm(cbind(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) ~ Species
              , data = iris)

## univariate ANOVA tests
#summary(lm.iris)
## test whether the multivariate means of the two populations are different
#library(car)
#man.mo <- Manova(lm.iris)
#summary(man.mo)
# perform canonical discriminant analysis

library(candisc)
can.iris <- candisc(lm.iris)
can.iris$coeffs.raw
```

```
##                    Can1        Can2
## Sepal.Length -0.8293776  0.02410215
## Sepal.Width  -1.5344731  2.16452123
## Petal.Length  2.2012117 -0.93192121
## Petal.Width   2.8104603  2.83918785
```

Can1 is a comparison of petal and sepal measurements (from Raw Canonical

Coefficients):

$$\text{Can1} = -0.8294\ \text{sepalL} + -1.534\ \text{sepalW} + 2.201\ \text{petalL} + 2.81\ \text{petalW}.$$

Can2 is not easily interpreted, though perhaps a comperison of lengths and widths ignoring sepalL:

$$\text{Can2} = 0.0241\ \text{sepalL} + 2.165\ \text{sepalW} + -0.9319\ \text{petalL} + 2.839\ \text{petalW}.$$

The canonical directions provide a maximal separation the species. Two lines across Can1 will provide a classification rule.

```
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
p <- ggpairs(can.iris$scores
             , mapping = ggplot2::aes(colour = Species, alpha = 0.5)
             , progress=FALSE
              )
print(p)
```

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

```
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```

There are significant differences among species on both discriminant functions; see the p-values under the likelihood ratio tests. Of course, Can1 produces the largest differences — the overlap among species on Can1 is small. Setosa has the lowest Can1 scores because this species has the smallest petal measurements relative to its sepal measurements. Virginica has the highest Can1 scores.

```
can.iris
##
## Canonical Discriminant Analysis for Species:
##
##    CanRsq Eigenvalue Difference  Percent Cumulative
## 1 0.96987   32.19193     31.907 99.12126     99.121
## 2 0.22203    0.28539     31.907  0.87874    100.000
##
## Test of H0: The canonical correlations in the
## current row and all that follow are zero
##
##   LR test stat approx F numDF denDF   Pr(> F)
## 1      0.02344  199.145     8   288 < 2.2e-16 ***
```

```
## 2      0.77797   13.794     3   145 5.794e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

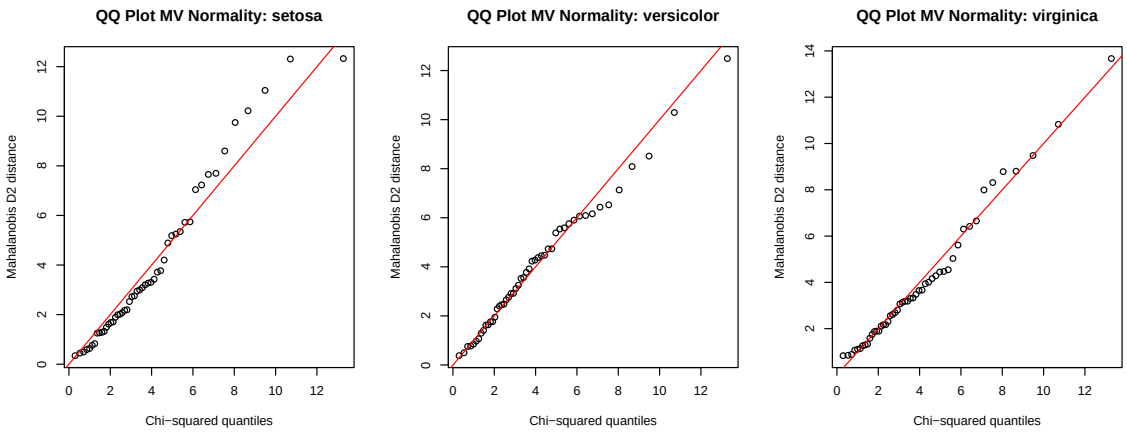
Questions:

1. What is the most striking feature of the plot of the Can1 scores?
2. Does the assumption of equal population covariance matrices across species seem plausible?
3. How about multivariate normality?

```
# Covariance matrices by species
by(iris[,1:4], iris$Species, cov)
## iris$Species: setosa
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length   0.12424898 0.099216327  0.016355102 0.010330612
## Sepal.Width    0.09921633 0.143689796  0.011697959 0.009297959
## Petal.Length   0.01635510 0.011697959  0.030159184 0.006069388
## Petal.Width    0.01033061 0.009297959  0.006069388 0.011106122
## ------------------------------------------------------------
## iris$Species: versicolor
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length   0.26643265  0.08518367   0.18289796  0.05577959
## Sepal.Width    0.08518367  0.09846939   0.08265306  0.04120408
## Petal.Length   0.18289796  0.08265306   0.22081633  0.07310204
## Petal.Width    0.05577959  0.04120408   0.07310204  0.03910612
## ------------------------------------------------------------
## iris$Species: virginica
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length   0.40434286  0.09376327   0.30328980  0.04909388
## Sepal.Width    0.09376327  0.10400408   0.07137959  0.04762857
## Petal.Length   0.30328980  0.07137959   0.30458776  0.04882449
## Petal.Width    0.04909388  0.04762857   0.04882449  0.07543265
```

```
# Test multivariate normality using the Shapiro-Wilk test for multivariate normality
library(mvnormtest)
# The data needs to be transposed t() so each variable is a row
#   with observations as columns.

mshapiro.test(t(iris[iris$Species == "setosa"     , 1:4]))
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.95878, p-value = 0.07906
mshapiro.test(t(iris[iris$Species == "versicolor", 1:4]))
##
##  Shapiro-Wilk normality test
```

```
##
## data:  Z
## W = 0.93043, p-value = 0.005739
mshapiro.test(t(iris[iris$Species == "virginica" , 1:4]))
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.93414, p-value = 0.007955
# Graphical Assessment of Multivariate Normality
f.mnv.norm.qqplot <- function(x, name = "") {
  # creates a QQ-plot for assessing multivariate normality

  x <- as.matrix(x)          # n x p numeric matrix
  center <- colMeans(x)      # centroid
  n <- nrow(x);
  p <- ncol(x);
  cov <- cov(x);
  d <- mahalanobis(x, center, cov) # distances
  qqplot(qchisq(ppoints(n), df=p), d
    , main=paste("QQ Plot MV Normality:", name)
    , ylab="Mahalanobis D2 distance"
    , xlab="Chi-squared quantiles")
  abline(a = 0, b = 1, col = "red")
}

par(mfrow=c(1,3))
f.mnv.norm.qqplot(iris[iris$Species == "setosa"    , 1:4], "setosa"    )
f.mnv.norm.qqplot(iris[iris$Species == "versicolor", 1:4], "versicolor")
f.mnv.norm.qqplot(iris[iris$Species == "virginica" , 1:4], "virginica" )
par(mfrow=c(1,1))
```

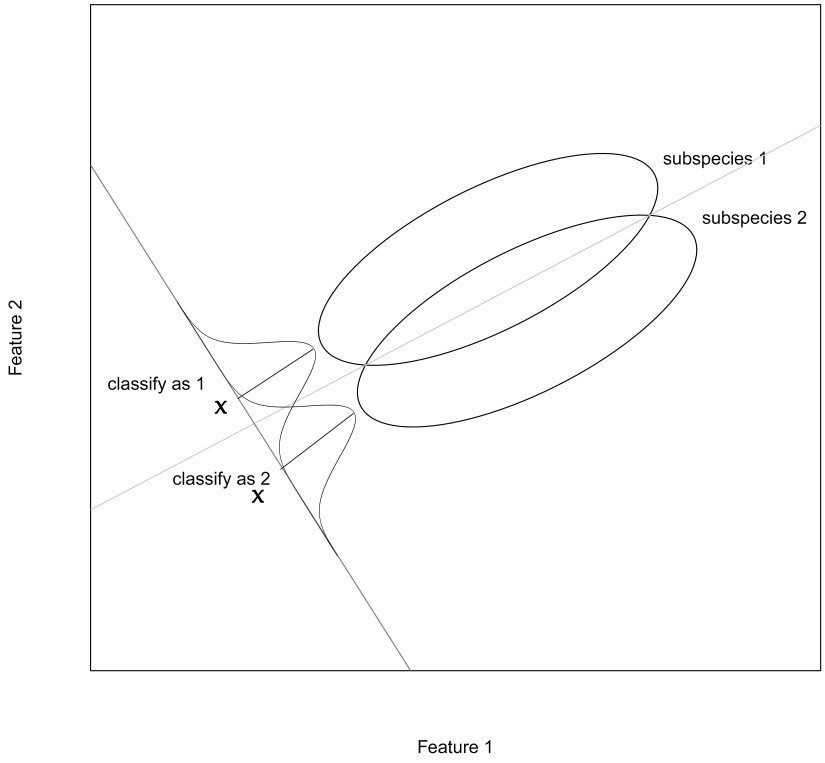


Chapter 17

Classification

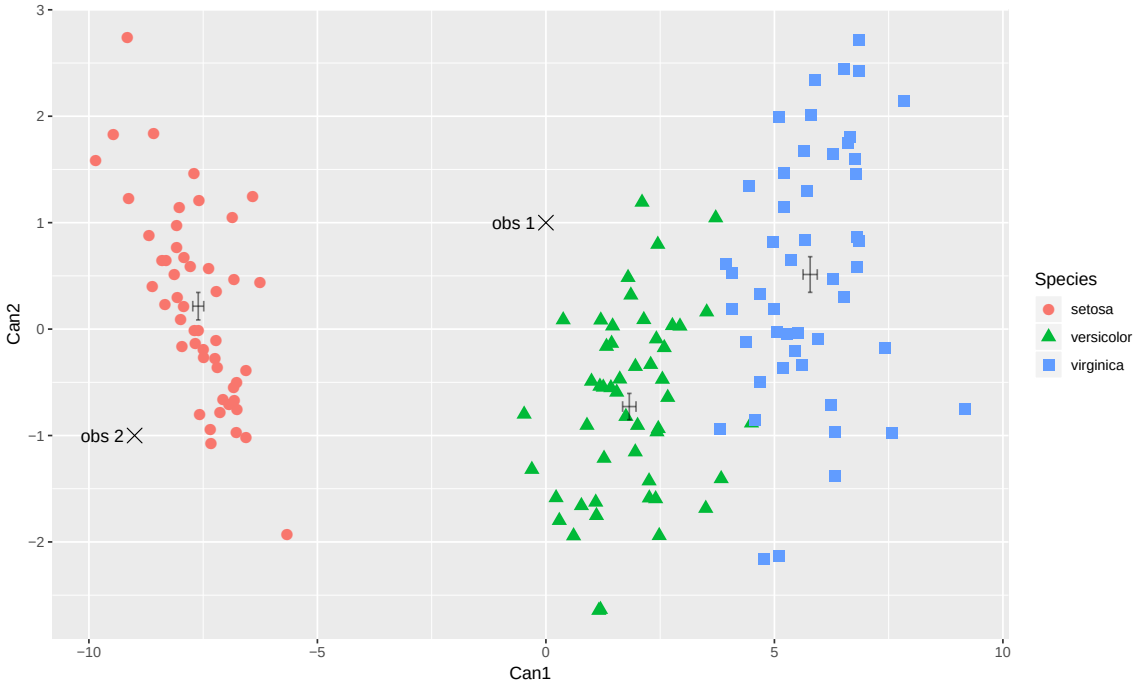
A goal with discriminant analysis might be to classify individuals of unknown origin into one of several known groups. How should this be done? Recall our example where two subspecies of an insect are compared on two external features. The discriminant analysis gives one discriminant function (CAN1) for distinguishing the subspecies. It makes sense to use CAN1 to classify insects because CAN1 is the best (linear) combination of the features to distinguish between subspecies.

Given the score on CAN1 for each insect to be classified, assign insects to the subspecies that they most resemble. Similarity is measured by the distance on CAN1 to the average CAN1 scores for the two subspecies, identified by X's on the plot.



To classify using r discriminant functions, compute the average response on CAN1, ..., CANr in each sample. Then compute the canonical discriminant function scores for each individual to be classified. Each observation is classified into the group it is closest to, as measured by the distance from the observation in r-space to the sample mean vector on the canonical variables. How do you measure distance in r-space?

The plot below illustrates the idea with the $r = 2$ discriminant functions in Fisher's iris data: Obs 1 is classified as Versicolor and Obs 2 is classified as Setosa.



17.1 Classification using Mahalanobis distance

Classification using discrimination is based on the **original features** and not the canonical discriminant function scores. Although a linear classification rule is identical to the method I just outlined, the equivalence is not obvious. I will discuss the method without justifying the equivalence.

Suppose p features $X = (X_1, X_2, \ldots, X_p)'$ are used to discriminate among the k groups. Let $\bar{X}_i = (\bar{X}_{i1}, \bar{X}_{i2}, \ldots, \bar{X}_{ip})'$ be the vector of mean responses for the i^{th} sample, and let S_i be the p-by-p variance-covariance matrix for the i^{th} sample. The pooled variance-covariance matrix is given by

$$S = \frac{(n_1 - 1)S_1 + (n_2 - 1)S_2 + \cdots + (n_k - 1)S_k}{n - k},$$

where the n_is are the group sample sizes and $n = n_1 + n_2 + \cdots + n_k$ is the total sample size.

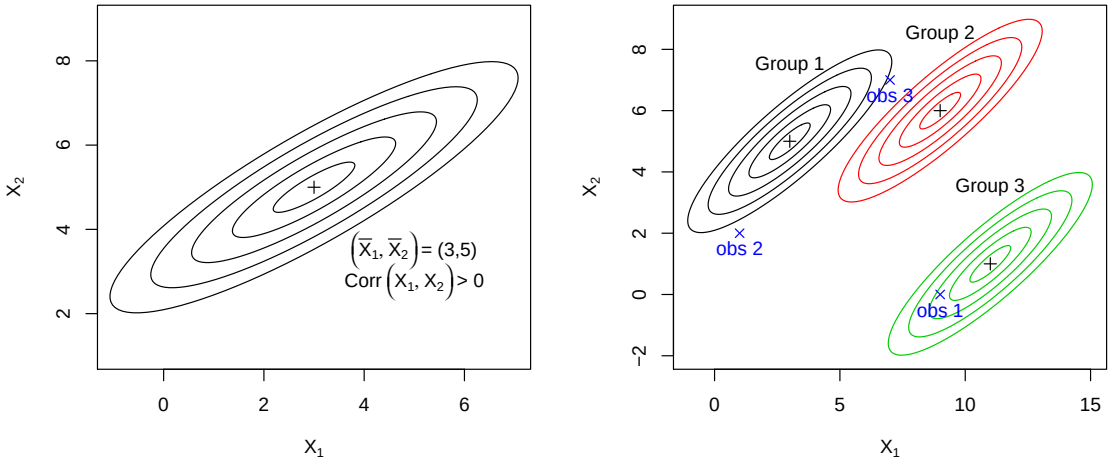
The classification rules below can be defined using either the **Mahalanobis generalized squared distance** or in terms of a probability model. I will describe each, starting with the Mahalanobis or M-distance.

The M-distance from an observation X to (the center of) the i^{th} sample is

$$D_i^2(X) = (X - \bar{X}_i)' S^{-1} (X - \bar{X}_i),$$

where $(X - \bar{X}_i)'$ is the transpose of the column vector $(X - \bar{X}_i)$, and S^{-1} is the matrix inverse of S. Note that if S is the identity matrix (a matrix with 1s on the diagonal and 0s on the off-diagonals), then this is the Euclidean distance. Given the M-distance from X to each sample, classify X into the group which has the minimum M-distance.

The M-distance is an elliptical distance measure that accounts for correlation between features, and adjusts for different scales by standardizing the features to have unit variance. The picture below (left) highlights the idea when $p = 2$. All of the points on a given ellipse are the same M-distance to the center $(\bar{X}_1, \bar{X}_2)'$. As the ellipse expands, the M-distance to the center increases.



To see how classification works, suppose you have three groups and two features, as in the plot above (right). Observations 1 is closest in M-distance to the center of group 3. Observation 2 is closest to group 1. Thus, classify observations 1 and 2 into groups 3 and 1, respectively. Observation 3 is closest to the center of group 2 in terms of the standard Euclidean (walking) distance. However, observation 3 is more similar to data in group 1 than it is to either of the other groups. The M-distance from observation 3 to group 1 is substantially smaller than the M-distances to either group 2 or 3. The M-distance accounts for the elliptical cloud of data within each group, which reflects the correlation between the two features. Thus, you would classify observation 3 into group 1.

The M-distance from the i^{th} group to the j^{th} group is the M-distance between the centers of the groups:

$$D^2(i,j) = D^2(j,i) = (\bar{X}_i - \bar{X}_j)' S^{-1} (\bar{X}_i - \bar{X}_j).$$

Larger values suggest relatively better potential for discrimination between groups.

In the plot above, $D^2(1,2) < D^2(1,3)$ which implies that it should be easier to distinguish between groups 1 and 3 than groups 1 and 2.

M-distance classification is equivalent to classification based on a probability model that assumes the samples are independently selected from multivariate normal populations with identical covariance matrices. This assumption is consistent with the plot above where the data points form elliptical clouds with similar orientations and spreads across samples. Suppose you can assume *a priori* (without looking at the data for the individual that you wish to classify) that a randomly selected individual from the **combined population** (i.e., merge all sub-populations) is equally likely to be from any group:

$$\text{PRIOR}_j \equiv \Pr(\text{observation is from group } j) = \frac{1}{k},$$

where k is the number of groups. Then, given the observed features X for an individual

$$\begin{aligned} \Pr(j|X) &\equiv \Pr(\text{observation is from group } j \text{ given } X) \\ &= \frac{\exp\{-0.5D_j^2(X)\}}{\sum_k \exp\{-0.5D_k^2(X)\}}. \end{aligned}$$

To be precise, I will note that $\Pr(j|X)$ is unknown, and the expression for $\Pr(j|X)$ is an estimate based on the data.

The group with the largest **posterior** probability $\Pr(j|X)$ is the group into which X is classified. Maximizing $\Pr(j|X)$ across groups is equivalent to minimizing the M-distance $D_j^2(X)$ across groups, so the two classification rules are equivalent.

17.2 Evaluating the Accuracy of a Classification Rule

The **misclassification rate**, or the expected proportion of misclassified observations, is a good yardstick to gauge a classification rule. Better rules have smaller misclassification rates, but there is no universal cutoff for what is considered good in a given problem. You should judge a classification rule relative to the current standards in your field for "good classification".

Resubstitution evaluates the misclassification rate using the data from which the classification rule is constructed. The resubstitution estimate of the error rate is optimistic (too small). A greater percentage of misclassifications is expected when the rule is used on new data, or on data from which the rule is not constructed.

Cross-validation is a better way to estimate the misclassification rate. In many statistical packages, you can implement cross-validation by randomly splitting the

data into a **training** or **calibration set** from which the classification rule is constructed. The remaining data, called the **test data set**, is used with the classification rule to estimate the error rate. In particular, the proportion of test cases misclassified estimates the misclassification rate. This process is often repeated, say 10 times, and the error rate estimated to be the average of the error rates from the individual splits. With repeated random splitting, it is common to use 10% of each split as the test data set (a 10-fold cross-validation).

Repeated random splitting can be coded. As an alternative, you might consider using one random 50-50 split (a 2-fold) to estimate the misclassification rate, provided you have a reasonably large data base.

Another form of cross-validation uses a **jackknife** method where single cases are held out of the data (an n-fold), then classified after constructing the classification rule from the remaining data. The process is repeated for each case, giving an estimated misclassification rate as the proportion of cases misclassified.

The `lda()` function allows for jackknife cross-validation (`CV`) and cross-validation using a single test data set (`predict()`). The jackknife method is necessary with small sized data sets so single observations don't greatly bias the classification. You can also classify observations with unknown group membership, by treating the observations to be classified as a test data set.

17.3 Example: Carapace classification and error

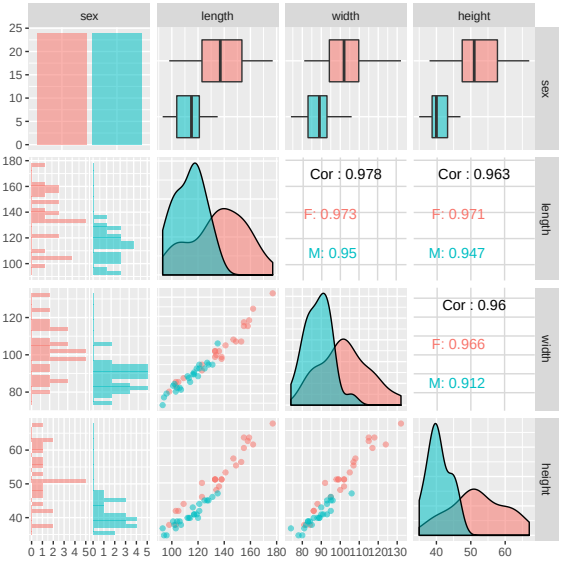
```
#### Example: Painted turtle shells
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch15_shells_mf.dat"
shells <- read.table(fn.data, header = TRUE)
```

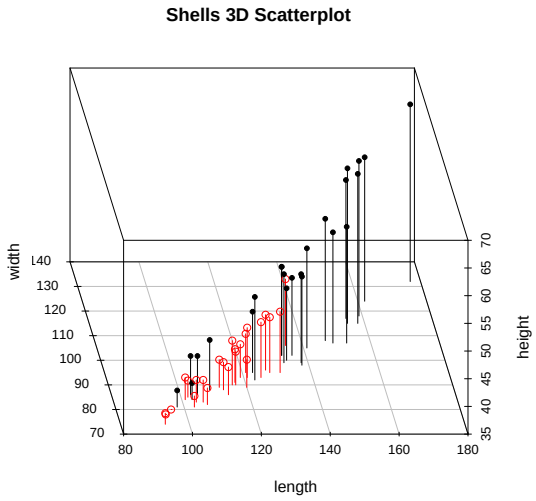
```
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
# put scatterplots on top so y axis is vertical
p <- ggpairs(shells
             , mapping = ggplot2::aes(colour = sex, alpha = 0.5)
             , progress=FALSE
             )
print(p)
```

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

```
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```

```
## 3D scatterplot
library(scatterplot3d)
with(shells, {
  scatterplot3d(x = length
                , y = width
                , z = height
                , main = "Shells 3D Scatterplot"
                , type = "h"                    # lines to the horizontal xy-plane
                , color = as.integer(sex)       # color by group
                , pch = as.integer(sex)+19      # plotting character by group
                #, highlight.3d = TRUE          # makes color change with z-axis value
                , angle = 100                    # viewing angle (seems hard to control)
                )
  })

#### Try this!
#### For a rotatable 3D plot, use plot3d() from the rgl library
# ##   This uses the R version of the OpenGL (Open Graphics Library)
# library(rgl)
# with(shells, { plot3d(x = length, y = width, z = height, col = sex) })
```

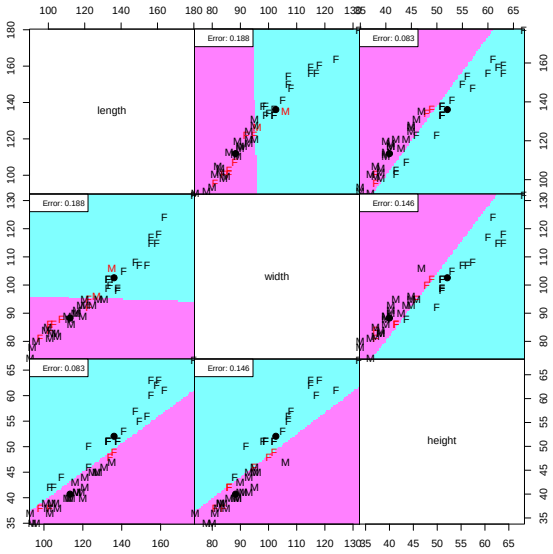




As suggested in the `partimat()` plot below (and by an earlier analysis), classification based on the length and height of the carapace appears best (considering only pairs). For this example, we'll only consider those two features.

```
# classification of observations based on classification methods
#   (e.g. lda, qda) for every combination of two variables.
library(klaR)
##
## Attaching package:  'klaR'
## The following object is masked from 'package:TeachingDemos':
```

```
##
##    triplot
partimat(sex ~ length + width + height, data = shells
        , plot.matrix = TRUE)
```



The default linear discriminant analysis assumes equal prior probabilities for males and females.

```
library(MASS)
lda.sh <- lda(sex ~ length + height, data = shells)
lda.sh
```

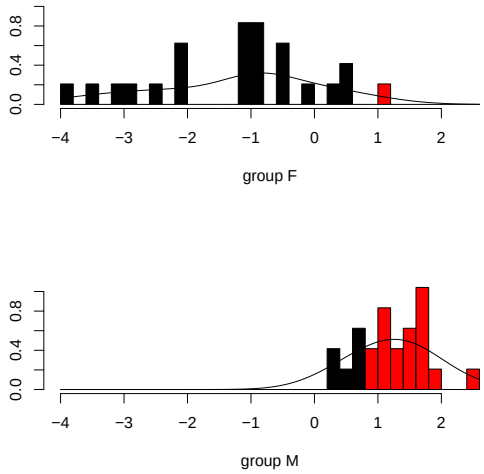
```
## Call:
## lda(sex ~ length + height, data = shells)
##
## Prior probabilities of groups:
##   F   M
## 0.5 0.5
##
## Group means:
##     length   height
## F 136.0417 52.04167
## M 113.4167 40.70833
##
## Coefficients of linear discriminants:
##                LD1
## length  0.1370519
## height -0.4890769
```

The linear discrimant function is in the direction that best separates the sexes,

$$\text{LD1} \quad = \quad 0.1371 \text{ length} + -0.4891 \text{ height}.$$

The plot of the `lda` object shows the groups across the linear discriminant function. From the `klaR` package we can get color-coded classification areas based on a perpendicular line across the LD function.

```
plot(lda.sh, dimen = 1, type = "both", col = as.numeric(shells$sex))
```



The constructed table gives the jackknife-based classification and posterior probabilities of being male or female for each observation in the data set. The misclassification rate follows.

```
# CV = TRUE does jackknife (leave-one-out) crossvalidation
lda.sh.cv <- lda(sex ~ length + height, data = shells, CV = TRUE)

# Create a table of classification and posterior probabilities for each observation
classify.sh <- data.frame(sex = shells$sex
                  , class = lda.sh.cv$class
                  , error = ""
                  , round(lda.sh.cv$posterior,3))
colnames(classify.sh) <- c("sex", "class", "error"
                         , paste("post", colnames(lda.sh.cv$posterior), sep=""))
                               # "postF" and "postM" column names

# error column
classify.sh$error <- as.character(classify.sh$error)
classify.agree <- as.character(as.numeric(shells$sex) - as.numeric(lda.sh.cv$class))
classify.sh$error[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]
```

The **classification summary table** is constructed from the canonical discriminant functions by comparing the predicted group membership for each observation to its actual group label. To be precise, if you assume that the sex for the 24 males are unknown then you would classify each of them correctly. Similarly, 20 of the

24 females are classified correctly, with the other four classified as males. The **Total Error** of 0.0833 is the estimated miscassification rate, computed as the sum of Rates×Prior over sexes: $0.0833 = 0.1667 \times 0.5 + 0 \times 0.5$. Are the misclassification results sensible, given the data plots that you saw earlier?

The listing of the posterior probabilities for each sex, by case, gives you an idea of the **clarity of classification**, with larger differences between the male and female posteriors corresponding to more definitive (but not necessarily correct!) classifications.

```
# print table
classify.sh
```

```
##    sex class error postF postM
## 1    F     M    -1 0.166 0.834
## 2    F     M    -1 0.031 0.969
## 3    F     F       0.847 0.153
## 4    F     F       0.748 0.252
## 5    F     F       0.900 0.100
## 6    F     F       0.992 0.008
## 7    F     F       0.517 0.483
## 8    F     F       0.937 0.063
## 9    F     F       0.937 0.063
## 10   F     F       0.937 0.063
## 11   F     M    -1 0.184 0.816
## 12   F     M    -1 0.294 0.706
## 13   F     F       0.733 0.267
## 14   F     F       0.733 0.267
## 15   F     F       0.917 0.083
## 16   F     F       0.994 0.006
## 17   F     F       0.886 0.114
## 18   F     F       0.864 0.136
## 19   F     F       1.000 0.000
## 20   F     F       0.998 0.002
## 21   F     F       1.000 0.000
## 22   F     F       1.000 0.000
## 23   F     F       0.993 0.007
## 24   F     F       0.999 0.001
## 25   M     M       0.481 0.519
## 26   M     M       0.040 0.960
## 27   M     M       0.020 0.980
## 28   M     M       0.346 0.654
## 29   M     M       0.092 0.908
## 30   M     M       0.021 0.979
## 31   M     M       0.146 0.854
## 32   M     M       0.078 0.922
## 33   M     M       0.018 0.982
## 34   M     M       0.036 0.964
## 35   M     M       0.026 0.974
```

```
## 36   M     M       0.019 0.981
## 37   M     M       0.256 0.744
## 38   M     M       0.023 0.977
## 39   M     M       0.023 0.977
## 40   M     M       0.012 0.988
## 41   M     M       0.002 0.998
## 42   M     M       0.175 0.825
## 43   M     M       0.020 0.980
## 44   M     M       0.157 0.843
## 45   M     M       0.090 0.910
## 46   M     M       0.067 0.933
## 47   M     M       0.081 0.919
## 48   M     M       0.074 0.926
# Assess the accuracy of the prediction
pred.freq <- table(shells$sex, lda.sh.cv$class) # row = true sex, col = classified sex
pred.freq
##
##      F  M
##   F 20  4
##   M  0 24
prop.table(pred.freq, 1) # proportions by row
##
##             F         M
##   F 0.8333333 0.1666667
##   M 0.0000000 1.0000000
# proportion correct for each category
diag(prop.table(pred.freq, 1))
##         F         M
## 0.8333333 1.0000000
# total proportion correct
sum(diag(prop.table(pred.freq)))
## [1] 0.9166667
# total error rate
1 - sum(diag(prop.table(pred.freq)))
## [1] 0.08333333
```

17.4 Example: Fisher's Iris Data cross-validation

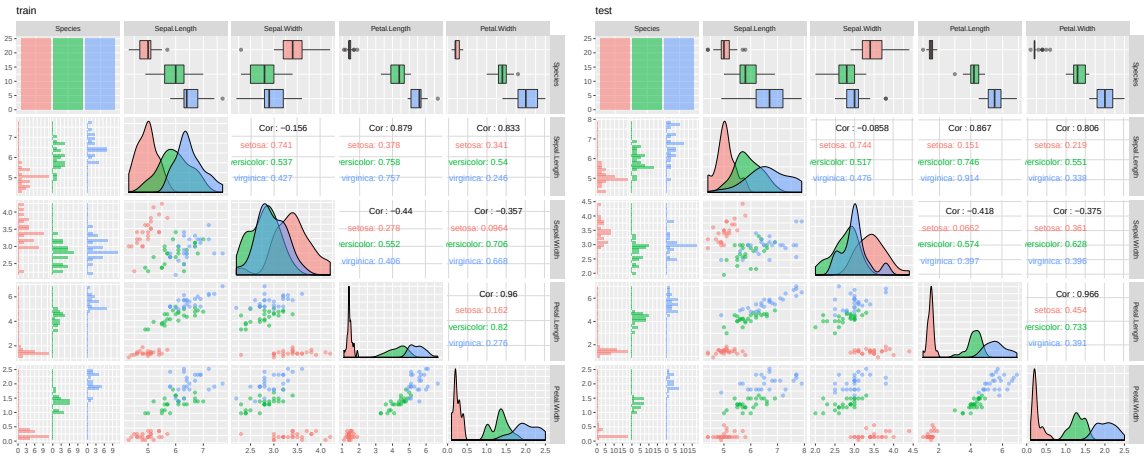
I will illustrate cross-validation on Fisher's iris data first using a test data set, and then using the jackknife method. The 150 observations were randomly rearranged and separated into two batches of 75. The 75 observations in the calibration set were used to develop a classification rule. This rule was applied to the remaining 75 flowers, which form the test data set. There is no general rule about the relative sizes

of the test data and the training data. Many researchers use a 50-50 split. Regardless of the split, you should combine the two data sets at the end of the cross-validation to create the actual rule for classifying future data.

Below, the half of the indices of the `iris` data set are randomly selected, and assigned a label "test", whereas the rest are "train". A plot indicates the two subsamples are similar.

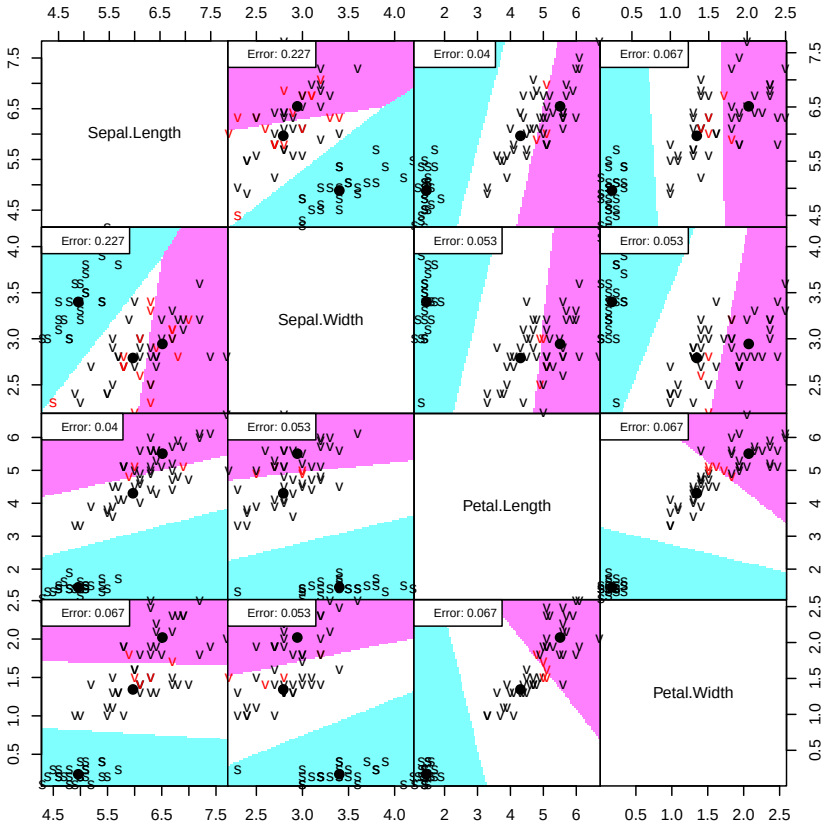
```
#### Example: Fisher's iris data
# The "iris" dataset is included with R in the library(datasets)
data(iris)
str(iris)
## 'data.frame': 150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
# Randomly assign equal train/test by Species strata
library(plyr)
iris <- ddply(iris, .(Species), function(X) {
      ind <- sample.int(nrow(X), size = round(nrow(X)/2))
      sort(ind)
      X$test      <- "train"
      X$test[ind] <- "test"
      X$test <- factor(X$test)
      X$test
      return(X)
      })
summary(iris$test)
##  test train
##    75    75
table(iris$Species, iris$test)
##
##              test train
##   setosa       25    25
##   versicolor   25    25
##   virginica    25    25
## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
p <- ggpairs(subset(iris, test == "train")[,c(5,1,2,3,4)]
           , mapping = ggplot2::aes(colour = Species, alpha = 0.5)
           , title = "train"
           , progress=FALSE
            )
print(p)
```

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
p <- ggpairs(subset(iris, test == "test")[,c(5,1,2,3,4)]
             , mapping = ggplot2::aes(colour = Species, alpha = 0.5)
             , title = "test"
             , progress=FALSE
             )
print(p)
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)
```



As suggested in the `partimat()` plot below, we should expect `Sepal.Length` to potentially not contribute much to the classification, since (pairwise) more errors are introduced with that variable than between other pairs. (In fact, we'll see below that the coefficients for `Sepal.Length` is smallest.)

```
# classification of observations based on classification methods
#   (e.g. lda, qda) for every combination of two variables.
library(klaR)
partimat(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
         , data = subset(iris, test == "train")
         , plot.matrix = TRUE)
```




```
library(MASS)
lda.iris <- lda(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
           , data = subset(iris, test == "train"))
lda.iris
```

```
## Call:
## lda(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
##     data = subset(iris, test == "train"))
##
## Prior probabilities of groups:
##     setosa versicolor  virginica
##  0.3333333  0.3333333  0.3333333
##
## Group means:
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa            4.960       3.400        1.448       0.240
## versicolor        5.968       2.792        4.304       1.344
## virginica         6.512       2.944        5.508       2.020
##
## Coefficients of linear discriminants:
##                    LD1        LD2
## Sepal.Length  0.598122 -0.3320011
```

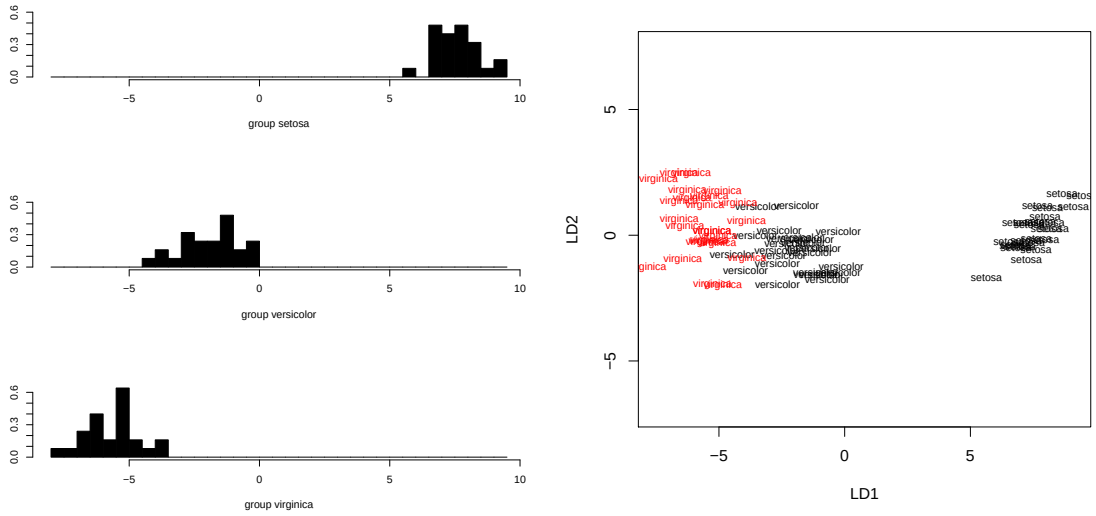
```
## Sepal.Width   1.684216  2.0882107
## Petal.Length -2.287078 -0.8035116
## Petal.Width  -2.297160  2.8095820
##
## Proportion of trace:
##    LD1    LD2
## 0.9934 0.0066
```

The linear discrimant functions that best classify the Species in the training set are

$$\begin{aligned} \text{LD1} &= 0.5981\ \text{sepalL} + 1.684\ \text{sepalW} + -2.287\ \text{petalL} + -2.297\ \text{petalW} \\ \text{LD2} &= -0.332\ \text{sepalL} + 2.088\ \text{sepalW} + -0.8035\ \text{petalL} + 2.81\ \text{petalW}. \end{aligned}$$

The plots of the `lda` object shows the data on the LD scale.

```
plot(lda.iris, dimen = 1, col = as.numeric(iris$Species))
plot(lda.iris, dimen = 2, col = as.numeric(iris$Species))
#pairs(lda.iris, col = as.numeric(iris£Species))
```




```
# CV = TRUE does jackknife (leave-one-out) crossvalidation
lda.iris.cv <- lda(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
                   , data = subset(iris, test == "train"), CV = TRUE)

# Create a table of classification and posterior probabilities for each observation
classify.iris <- data.frame(Species = subset(iris, test == "train")$Species
                    , class = lda.iris.cv$class
                    , error = ""
                    , round(lda.iris.cv$posterior,3))
colnames(classify.iris) <- c("Species", "class", "error"
                           , paste("post", colnames(lda.iris.cv$posterior), sep=""))
```

```
# error column
classify.iris$error <- as.character(classify.iris$error)
classify.agree <- as.character(as.numeric(subset(iris, test == "train")$Species)
                               - as.numeric(lda.iris.cv$class))
classify.iris$error[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]
```

The misclassification error is low within the training set.

```
# print table
#classify.iris

# Assess the accuracy of the prediction
#     row = true Species, col = classified Species
pred.freq <- table(subset(iris, test == "train")$Species, lda.iris.cv$class)
pred.freq
```

```
##
##              setosa versicolor virginica
##   setosa         25          0         0
##   versicolor      0         23         2
##   virginica       0          1        24
```

```
prop.table(pred.freq, 1) # proportions by row
```

```
##
##              setosa versicolor virginica
##   setosa       1.00       0.00      0.00
##   versicolor   0.00       0.92      0.08
##   virginica    0.00       0.04      0.96
```

```
# proportion correct for each category
diag(prop.table(pred.freq, 1))
```

```
##     setosa versicolor  virginica
##       1.00       0.92       0.96
```

```
# total proportion correct
sum(diag(prop.table(pred.freq)))
```

```
## [1] 0.96
```

```
# total error rate
1 - sum(diag(prop.table(pred.freq)))
```

```
## [1] 0.04
```

How well does the LD functions constructed on the training data predict the Species in the independent test data?

```
# predict the test data from the training data LDFs
pred.iris <- predict(lda.iris, newdata = subset(iris, test == "test"))

# Create a table of classification and posterior probabilities for each observation
classify.iris <- data.frame(Species = subset(iris, test == "test")$Species
                  , class = pred.iris$class
                  , error = ""
```

```
                    , round(pred.iris$posterior,3))
colnames(classify.iris) <- c("Species", "class", "error"
                           , paste("P", colnames(lda.iris.cv$posterior), sep=""))

# error column
classify.iris$error <- as.character(classify.iris$error)
classify.agree <- as.character(as.numeric(subset(iris, test == "test")$Species)
                               - as.numeric(pred.iris$class))
classify.iris$error[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]

# print table
classify.iris
```

```
##        Species      class error Psetosa Pversicolor Pvirginica
## 2       setosa     setosa             1       0.000      0.000
## 3       setosa     setosa             1       0.000      0.000
## 6       setosa     setosa             1       0.000      0.000
## 8       setosa     setosa             1       0.000      0.000
## 9       setosa     setosa             1       0.000      0.000
## 10      setosa     setosa             1       0.000      0.000
## 11      setosa     setosa             1       0.000      0.000
## 12      setosa     setosa             1       0.000      0.000
## 15      setosa     setosa             1       0.000      0.000
## 16      setosa     setosa             1       0.000      0.000
## 23      setosa     setosa             1       0.000      0.000
## 24      setosa     setosa             1       0.000      0.000
## 26      setosa     setosa             1       0.000      0.000
## 28      setosa     setosa             1       0.000      0.000
## 29      setosa     setosa             1       0.000      0.000
## 31      setosa     setosa             1       0.000      0.000
## 35      setosa     setosa             1       0.000      0.000
## 37      setosa     setosa             1       0.000      0.000
## 40      setosa     setosa             1       0.000      0.000
## 41      setosa     setosa             1       0.000      0.000
## 43      setosa     setosa             1       0.000      0.000
## 44      setosa     setosa             1       0.000      0.000
## 45      setosa     setosa             1       0.000      0.000
## 47      setosa     setosa             1       0.000      0.000
## 49      setosa     setosa             1       0.000      0.000
## 52  versicolor versicolor             0       0.999      0.001
## 53  versicolor versicolor             0       0.992      0.008
## 54  versicolor versicolor             0       0.999      0.001
## 55  versicolor versicolor             0       0.992      0.008
## 57  versicolor versicolor             0       0.988      0.012
## 59  versicolor versicolor             0       1.000      0.000
## 61  versicolor versicolor             0       1.000      0.000
## 62  versicolor versicolor             0       0.999      0.001
## 63  versicolor versicolor             0       1.000      0.000
## 69  versicolor versicolor             0       0.918      0.082
```

```
## 74  versicolor versicolor             0       0.999      0.001
## 76  versicolor versicolor             0       1.000      0.000
## 79  versicolor versicolor             0       0.991      0.009
## 80  versicolor versicolor             0       1.000      0.000
## 83  versicolor versicolor             0       1.000      0.000
## 85  versicolor versicolor             0       0.973      0.027
## 89  versicolor versicolor             0       1.000      0.000
## 90  versicolor versicolor             0       1.000      0.000
## 91  versicolor versicolor             0       0.998      0.002
## 93  versicolor versicolor             0       1.000      0.000
## 95  versicolor versicolor             0       0.999      0.001
## 96  versicolor versicolor             0       1.000      0.000
## 97  versicolor versicolor             0       1.000      0.000
## 98  versicolor versicolor             0       1.000      0.000
## 99  versicolor versicolor             0       1.000      0.000
## 103  virginica  virginica             0       0.000      1.000
## 105  virginica  virginica             0       0.000      1.000
## 106  virginica  virginica             0       0.000      1.000
## 107  virginica  virginica             0       0.114      0.886
## 108  virginica  virginica             0       0.000      1.000
## 109  virginica  virginica             0       0.000      1.000
## 113  virginica  virginica             0       0.001      0.999
## 114  virginica  virginica             0       0.001      0.999
## 116  virginica  virginica             0       0.000      1.000
## 118  virginica  virginica             0       0.000      1.000
## 119  virginica  virginica             0       0.000      1.000
## 122  virginica  virginica             0       0.004      0.996
## 124  virginica  virginica             0       0.137      0.863
## 127  virginica  virginica             0       0.283      0.717
## 130  virginica  virginica             0       0.053      0.947
## 132  virginica  virginica             0       0.001      0.999
## 136  virginica  virginica             0       0.000      1.000
## 138  virginica  virginica             0       0.010      0.990
## 139  virginica  virginica             0       0.346      0.654
## 140  virginica  virginica             0       0.003      0.997
## 141  virginica  virginica             0       0.000      1.000
## 145  virginica  virginica             0       0.000      1.000
## 148  virginica  virginica             0       0.009      0.991
## 149  virginica  virginica             0       0.000      1.000
## 150  virginica  virginica             0       0.039      0.961
```

```
# Assess the accuracy of the prediction
#     row = true Species, col = classified Species
pred.freq <- table(subset(iris, test == "test")$Species, pred.iris$class)
pred.freq
```

```
##
##              setosa versicolor virginica
##   setosa         25          0         0
##   versicolor      0         25         0
```

```
##   virginica       0          0        25
prop.table(pred.freq, 1) # proportions by row
##
##              setosa versicolor virginica
##   setosa          1          0         0
##   versicolor      0          1         0
##   virginica       0          0         1
# proportion correct for each category
diag(prop.table(pred.freq, 1))
##     setosa versicolor  virginica
##          1          1          1
# total proportion correct
sum(diag(prop.table(pred.freq)))
## [1] 1
# total error rate
1 - sum(diag(prop.table(pred.freq)))
## [1] 0
```

The classification rule based on the training set works well with the test data. Do not expect such nice results on all classification problems! Usually the error rate is slightly higher on the test data than on the training data.

It is important to recognize that statistically significant differences (MANOVA) among groups on linear discriminant function scores do not necessarily translate into accurate classification rules! (WHY?)

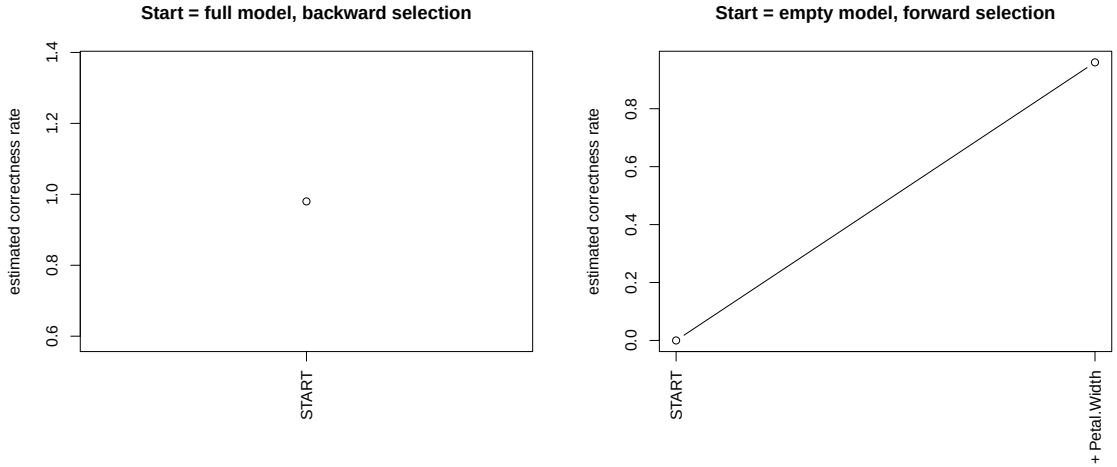
17.4.1 Stepwise variable selection for classification

Stepwise variable selection for classification can be performed using package `klaR` function `stepclass()` using any specified classification function. Classification performance is estimated by selected from one of Uschi's classification performance measures.

The resulting model can be very sensitive to the starting model. Below, the first model starts full and ends full. The second model starts empty and ends after one variable is added. Note that running this repeatedly could result in slightly different models because the k-fold crossvalidation partitions the data at random. The `formula` object gives the selected model.

```
library(klaR)
# start with full model and do stepwise (direction = "backward")
step.iris.b <- stepclass(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
                     , data = iris
                     , method = "lda"
                     , improvement = 0.01 # stop criterion: improvement less than 1%
                                          # default of 5% is too coarse
```

```
                    , direction = "backward")
##  'stepwise classification', using 10-fold cross-validated correctness rate of method
lda'.
## 150 observations of 4 variables in 3 classes; direction:  backward
## stop criterion:  improvement less than 1%.
## correctness rate: 0.98;  starting variables (4): Sepal.Length, Sepal.Width, Petal.Length, Petal.Widt
##
##  hr.elapsed min.elapsed sec.elapsed
##        0.00        0.00        0.19
plot(step.iris.b, main = "Start = full model, backward selection")
step.iris.b$formula
## Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
## <environment: 0x000000002045a3f0>
# start with empty model and do stepwise (direction = "both")
step.iris.f <- stepclass(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
                    , data = iris
                    , method = "lda"
                    , improvement = 0.01 # stop criterion: improvement less than 1%
                                         # default of 5% is too coarse
                    , direction = "forward")
##  'stepwise classification', using 10-fold cross-validated correctness rate of method
lda'.
## 150 observations of 4 variables in 3 classes; direction:  forward
## stop criterion:  improvement less than 1%.
## correctness rate: 0.96;  in: "Petal.Width";  variables (1): Petal.Width
##
##  hr.elapsed min.elapsed sec.elapsed
##         0.0         0.0         0.2
plot(step.iris.f, main = "Start = empty model, forward selection")
step.iris.f$formula
## Species ~ Petal.Width
## <environment: 0x000000003222b898>
```



Given your selected model, you can then go on to fit your classification model by using the formula from the `stepclass()` object.

```
library(MASS)
lda.iris.step <- lda(step.iris.b$formula
                     , data = iris)
lda.iris.step
```

```
## Call:
## lda(step.iris.b$formula, data = iris)
##
## Prior probabilities of groups:
##     setosa versicolor  virginica
##  0.3333333  0.3333333  0.3333333
##
## Group means:
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa            5.006       3.428        1.462       0.246
## versicolor        5.936       2.770        4.260       1.326
## virginica         6.588       2.974        5.552       2.026
##
## Coefficients of linear discriminants:
##                     LD1         LD2
## Sepal.Length  0.8293776  0.02410215
## Sepal.Width   1.5344731  2.16452123
## Petal.Length -2.2012117 -0.93192121
## Petal.Width  -2.8104603  2.83918785
##
## Proportion of trace:
##    LD1    LD2
## 0.9912 0.0088
```

Note that if you have many variables, you may wish to use the alternate syntax below to specify your formula (see the help `?stepclass` for this example).

```
iris.d <- iris[,1:4]  # the data
iris.c <- iris[,5]    # the classes
sc_obj <- stepclass(iris.d, iris.c, "lda", start.vars = "Sepal.Width")
```

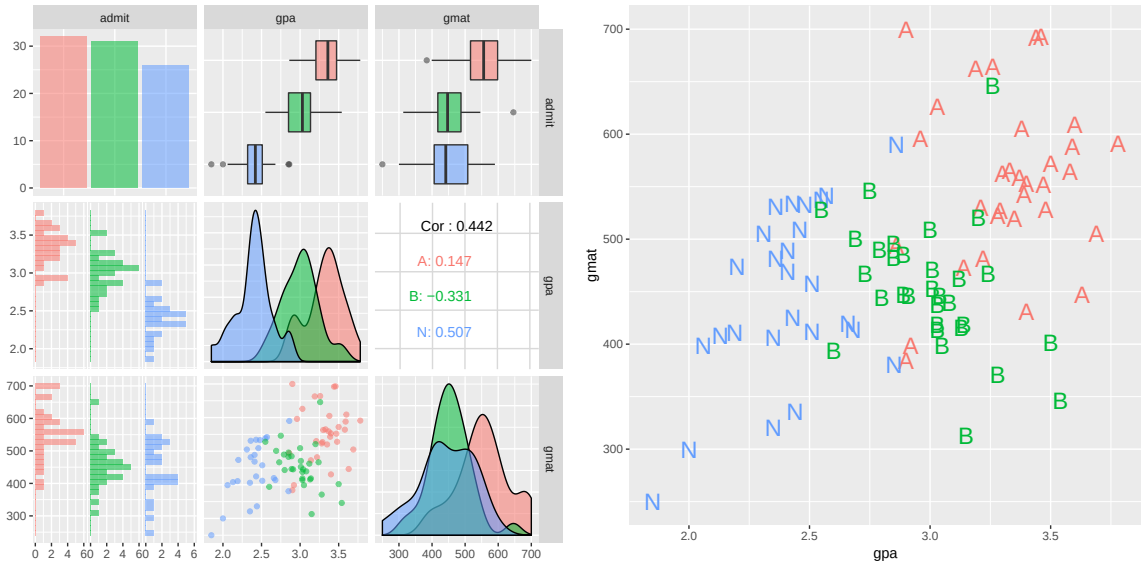
17.5 Example: Analysis of Admissions Data

The admissions officer of a business school has used an index of undergraduate GPA and management aptitude test scores (GMAT) to help decide which applicants should be admitted to graduate school. The data below gives the GPA and GMAT scores for recent applicants who are classified as admit (A), borderline (B), or not admit (N). An equal number of A, B, and N's (roughly) were selected from their corresponding populations (Johnson and Wichern, 1988).

```
#### Example: Business school admissions data
fn.data <- "http://statacumen.com/teach/ADA2/ADA2_notes_Ch17_business.dat"
business <- read.table(fn.data, header = TRUE)

## Scatterplot matrix
library(ggplot2)
#suppressMessages(suppressWarnings(library(GGally)))
library(GGally)
p <- ggpairs(business
             , mapping = ggplot2::aes(colour = admit, alpha = 0.5)
             , progress=FALSE
             )
print(p)
```

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

```
  # detach package after use so reshape2 works (old reshape (v.1) conflicts)
  #detach("package:GGally", unload=TRUE)
  #detach("package:reshape", unload=TRUE)

library(ggplot2)
p <- ggplot(business, aes(x = gpa, y = gmat, shape = admit, colour = admit))
p <- p + geom_point(size = 6)
  library(R.oo) # for ascii code lookup
  p <- p + scale_shape_manual(values=charToInt(sort(unique(business$admit))))
p <- p + theme(legend.position="none") # remove legend with fill colours
print(p)
```



The officer wishes to use these data to develop a more quantitative (i.e., less subjective) approach to classify prospective students. Historically, about 20% of all applicants have been admitted initially, 10% are classified as borderline, and the remaining 70% are not admitted. The officer would like to keep these percentages roughly the same in the future.

This is a natural place to use discriminant analysis. Let us do a more careful analysis here, paying attention to underlying assumptions of normality and equal covariance matrices.

The GPA and GMAT distributions are reasonably symmetric. Although a few outliers are present, it does not appear that any transformation will eliminate the outliers and preserve the symmetry. Given that the outliers are not very extreme, I would analyze the data on this scale. Except for the outliers, the spreads (IQRs) are roughly equal across groups within GPA and GMAT. I will look carefully at the variance-covariance matrices later.

There is a fair amount of overlap between the borderline and other groups, but this should not be too surprising. Otherwise these applicants would not be borderline!

17.5.1 Further Analysis of the Admissions Data

The assumption of constant variance-covariance matrices is suspect. The GPA and GMAT variances are roughly constant across groups, but the correlation between GPA and GMAT varies greatly over groups.

```
# Covariance matrices by admit
by(business[,2:3], business$admit, cov)
## business$admit: A
##             gpa        gmat
```

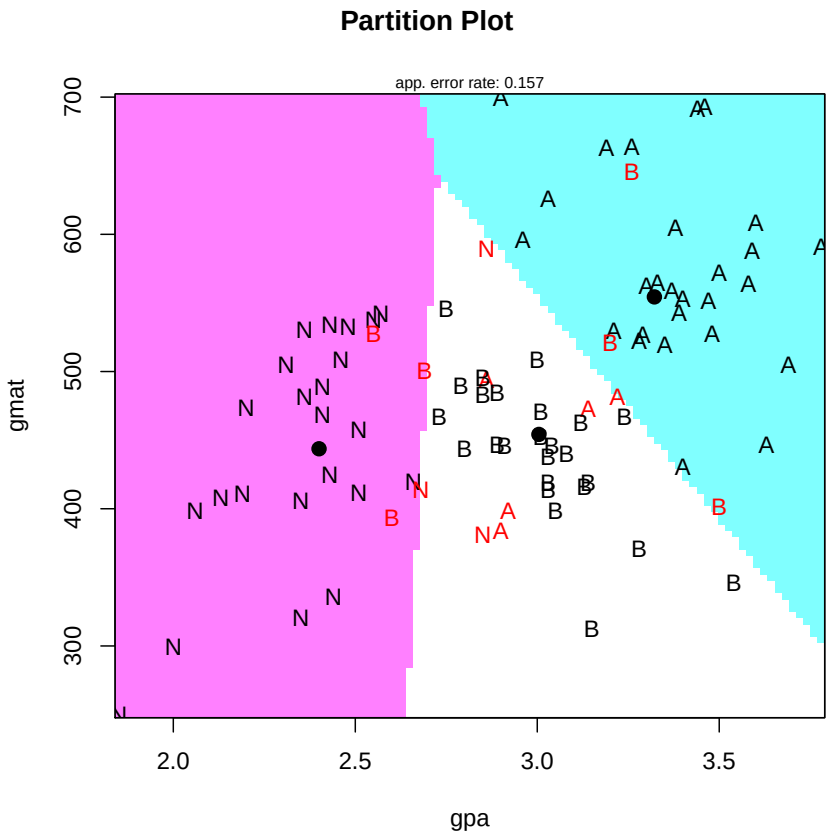
```
## gpa  0.05866734    2.857601
## gmat 2.85760081 6479.990927
## ------------------------------------------------------
## business$admit: B
##              gpa       gmat
## gpa   0.05422559   -4.87757
## gmat -4.87756989 4002.76129
## ------------------------------------------------------
## business$admit: N
##              gpa       gmat
## gpa   0.05602785   10.01171
## gmat 10.01170769 6973.16462
```

```
# Correlation matrices by admit
by(business[,2:3], business$admit, FUN=stats::cor)
```

```
## business$admit: A
##            gpa      gmat
## gpa  1.0000000 0.1465602
## gmat 0.1465602 1.0000000
## ------------------------------------------------------
## business$admit: B
##             gpa       gmat
## gpa   1.0000000 -0.3310713
## gmat -0.3310713  1.0000000
## ------------------------------------------------------
## business$admit: N
##            gpa      gmat
## gpa  1.0000000 0.5065137
## gmat 0.5065137 1.0000000
```

Assuming equal variance-covariance matrices, both GPA and GMAT are important for discriminating among entrance groups. This is consistent with the original data plots.

```
# classification of observations based on classification methods
#   (e.g. lda, qda) for every combination of two variables.
library(klaR)
partimat(admit ~ gmat + gpa
        , data = business
        , plot.matrix = FALSE)
```




```
library(MASS)
lda.business <- lda(admit ~ gpa + gmat
            , data = business)
lda.business
```

```
## Call:
## lda(admit ~ gpa + gmat, data = business)
##
## Prior probabilities of groups:
##         A         B         N
## 0.3595506 0.3483146 0.2921348
##
## Group means:
##        gpa     gmat
## A 3.321875 554.4062
## B 3.004516 454.1935
## N 2.400385 443.7308
##
## Coefficients of linear discriminants:
##                LD1         LD2
## gpa  -3.977912929 -1.48346456
## gmat -0.003057846  0.01292319
```

```
##
## Proportion of trace:
##    LD1    LD2
## 0.9473 0.0527
```

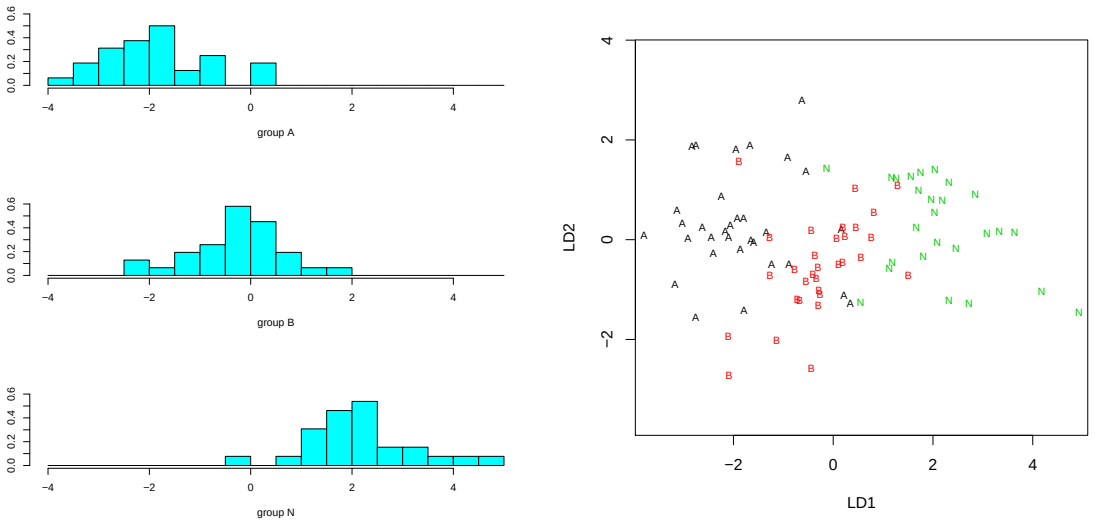
The linear discrimant functions that best classify the admit are

$$\begin{aligned} \text{LD1} &= -3.978\ \text{gpa} + -0.003058\ \text{gmat} \\ \text{LD2} &= -1.483\ \text{gpa} + 0.01292\ \text{gmat}, \end{aligned}$$

interpretted as a weighted average of the scores and a contrast of the scores.

The plots of the `lda()` object shows the data on the LD scale.

```
plot(lda.business, dimen = 1)
plot(lda.business, dimen = 2, col = as.numeric(business$admit))
#pairs(lda.business, col = as.numeric(business$admit))
```




```
# CV = TRUE does jackknife (leave-one-out) crossvalidation
lda.business.cv <- lda(admit ~ gpa + gmat
                  , data = business, CV = TRUE)

# Create a table of classification and posterior probabilities for each observation
classify.business <- data.frame(admit = business$admit
                    , class = lda.business.cv$class
                    , error = ""
                    , round(lda.business.cv$posterior,3))
colnames(classify.business) <- c("admit", "class", "error"
                         , paste("post", colnames(lda.business.cv$posterior), sep=""))

# error column
classify.business$error <- as.character(classify.business$error)
```

```
classify.agree <- as.character(as.numeric(business$admit)
                               - as.numeric(lda.business.cv$class))
classify.business$error[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]
```

The misclassification error within the training set is reasonably low, given the overlap between the B group and the others, and never a misclassification between A and N.

```
# print table
#classify.business

# Assess the accuracy of the prediction
#     row = true admit, col = classified admit
pred.freq <- table(business$admit, lda.business.cv$class)
pred.freq
##
##      A  B  N
##   A 26  6  0
##   B  3 25  3
##   N  0  3 23
prop.table(pred.freq, 1) # proportions by row
##
##               A          B          N
##   A 0.81250000 0.18750000 0.00000000
##   B 0.09677419 0.80645161 0.09677419
##   N 0.00000000 0.11538462 0.88461538
# proportion correct for each category
diag(prop.table(pred.freq, 1))
##         A         B         N
## 0.8125000 0.8064516 0.8846154
# total proportion correct
sum(diag(prop.table(pred.freq)))
## [1] 0.8314607
# total error rate
1 - sum(diag(prop.table(pred.freq)))
## [1] 0.1685393
```

17.5.2 Classification Using Unequal Prior Probabilities

A new twist to this problem is that we have prior information on the relative sizes of the populations. Historically 20% of applicants are admitted, 10% are borderline, and 70% are not admitted. This prior information is incorporated into the classification rule by using a `prior` option with `lda()`. The prior probabilities are assumed to be equal when this statement is omitted. The classification rule for unequal prior probabilities uses both the M-distance and the prior probability for making decisions,

as outlined below.

When the prior probabilities are unequal, classification is based on the **generalized distance** to group j:

$$D_j^2(X) = (X - \bar{X}_j)' S^{-1} (X - \bar{X}_j) - 2\log(\text{PRIOR}_j),$$

or on the estimated posterior probability of membership in group j:

$$\Pr(j|X) = \frac{\exp\{-0.5 D_j^2(X)\}}{\sum_k \exp\{-0.5 D_k^2(X)\}}.$$

Here S is the pooled covariance matrix, and $\log(\text{PRIOR}_j)$ is the (natural) log of the prior probability of being in group j. As before, you classify observation X into the group that it is closest to in terms of generalized distance, or equivalently, into the group with the maximum posterior probability.

Note that $-2\log(\text{PRIOR}_j)$ exceeds zero, and is extremely large when PRIOR_j is near zero. The generalized distance is the M-distance plus a penalty term that is large when the prior probability for a given group is small. If the prior probabilities are equal, the penalty terms are equal so the classification rule depends only on the M-distance.

The penalty makes it harder (relative to equal probabilities) to classify into a low probability group, and easier to classify into high probability groups. In the admissions data, an observation has to be very close to the B or A groups to not be classified as N.

Note that in the analysis below, we make the tenuous assumption that the population covariance matrices are equal. We also have 6 new observations that we wish to classify. These observations are entered as a test data set with missing class levels.

```
# new observations to classify
business.test <- read.table(text = "
admit gpa gmat
   NA 2.7  630
   NA 3.3  450
   NA 3.4  540
   NA 2.8  420
   NA 3.5  340
   NA 3.0  500
", header = TRUE)
```

With priors, the LDs are different.

```
library(MASS)
lda.business <- lda(admit ~ gpa + gmat
            , prior = c(0.2, 0.1, 0.7)
            , data = business)
lda.business
```

```
## Call:
## lda(admit ~ gpa + gmat, data = business, prior = c(0.2, 0.1,
##     0.7))
##
## Prior probabilities of groups:
##   A   B   N
## 0.2 0.1 0.7
##
## Group means:
##        gpa     gmat
## A 3.321875 554.4062
## B 3.004516 454.1935
## N 2.400385 443.7308
##
## Coefficients of linear discriminants:
##               LD1         LD2
## gpa  -4.014778092 -1.38058511
## gmat -0.002724201  0.01299761
##
## Proportion of trace:
##    LD1    LD2
## 0.9808 0.0192
```

About 1/2 of the borderlines in the calibration set are misclassified. This is due to the overlap of the B group with the other 2 groups, but also reflects the low prior probability for the borderline group. The classification rule requires strong evidence that an observation is borderline before it can be classified as such.

```
# CV = TRUE does jackknife (leave-one-out) crossvalidation
lda.business.cv <- lda(admit ~ gpa + gmat
                       , prior = c(0.2, 0.1, 0.7)
                       , data = business, CV = TRUE)

# Create a table of classification and posterior probabilities for each observation
classify.business <- data.frame(admit = business$admit
                     , class = lda.business.cv$class
                     , error = ""
                     , round(lda.business.cv$posterior,3))
colnames(classify.business) <- c("admit", "class", "error"
                           , paste("post", colnames(lda.business.cv$posterior), sep=""))

# error column
classify.business$error <- as.character(classify.business$error)
classify.agree <- as.character(as.numeric(business$admit)
                               - as.numeric(lda.business.cv$class))
classify.business$error[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]

# print table, errors only
classify.business[!(classify.business$error == ""), ]
```

```
##    admit class error postA postB postN
## 29     A     B    -1 0.027 0.557 0.416
## 30     A     N    -2 0.114 0.383 0.503
## 31     A     B    -1 0.045 0.595 0.360
## 35     N     A     2 0.442 0.273 0.285
## 36     N     B     1 0.033 0.516 0.451
## 59     B     N    -1 0.001 0.044 0.955
## 60     B     N    -1 0.002 0.016 0.982
## 61     B     N    -1 0.126 0.329 0.545
## 66     B     A     1 0.747 0.253 0.000
## 67     B     N    -1 0.161 0.412 0.428
## 68     B     N    -1 0.037 0.277 0.686
## 71     B     N    -1 0.059 0.227 0.714
## 72     B     N    -1 0.017 0.089 0.894
## 74     B     N    -1 0.070 0.129 0.801
## 75     B     N    -1 0.020 0.146 0.834
## 82     B     N    -1 0.107 0.344 0.549
## 85     B     A     1 0.758 0.233 0.009
## 86     B     A     1 0.979 0.021 0.001
## 87     B     A     1 0.627 0.365 0.008
## 88     B     A     1 0.633 0.367 0.000
```

```
# Assess the accuracy of the prediction
#     row = true admit, col = classified admit
pred.freq <- table(business$admit, lda.business.cv$class)
pred.freq
```

```
##
##      A  B  N
##   A 29  2  1
##   B  5 16 10
##   N  1  1 24
```

```
prop.table(pred.freq, 1) # proportions by row
```

```
##
##              A          B          N
##   A 0.90625000 0.06250000 0.03125000
##   B 0.16129032 0.51612903 0.32258065
##   N 0.03846154 0.03846154 0.92307692
```

```
# proportion correct for each category
diag(prop.table(pred.freq, 1))
```

```
##         A         B         N
## 0.9062500 0.5161290 0.9230769
```

```
# total proportion correct
sum(diag(prop.table(pred.freq)))
```

```
## [1] 0.7752809
```

```
# total error rate
1 - sum(diag(prop.table(pred.freq)))
```

```
## [1] 0.2247191
```

The test data cases were entered with missing group IDs. The classification table compares the group IDs, which are unknown, to the ID for the group into which an observation is classified. These two labels differ, so all the test data cases are identified as misclassified. Do not be confused by this! Just focus on the classification for each case, and ignore the other summaries.

```
# predict the test data from the training data LDFs
pred.business <- predict(lda.business, newdata = business.test)

# Create a table of classification and posterior probabilities for each observation
classify.business.test <- data.frame(admit = business.test$admit
                    , class = pred.business$class
                    #, error = ""
                    , round(pred.business$posterior,3))
colnames(classify.business.test) <- c("admit", "class"#, "error"
                          , paste("post", colnames(pred.business$posterior), sep=""))

## error column
#classify.business.test$error <- as.character(classify.business.test$error)
#classify.agree <- as.character(as.numeric(business.test$admit)
#                               - as.numeric(pred.business$class))
#classify.business.test$error[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]

# print table
classify.business.test
##   admit class postA postB postN
## 1    NA     N 0.102 0.074 0.824
## 2    NA     A 0.629 0.367 0.004
## 3    NA     A 0.919 0.081 0.000
## 4    NA     N 0.026 0.297 0.676
## 5    NA     B 0.461 0.538 0.001
## 6    NA     B 0.385 0.467 0.148
```

Except for observation 5, the posterior probabilities for the test cases give strong evidence in favor of classification into a specific group.

17.5.3 Classification With Unequal Covariance Matrices, QDA

The assumption of multivariate normal populations with equal covariance matrices is often unrealistic. Although there are no widely available procedures for MANOVA or stepwise variable selection in discriminant analysis that relax these assumptions, there are a variety of classification rules that weaken one or both of these assumptions. The `qda()` function is a **quadratic discriminant classification rule** that assumes normality but **allows unequal covariance matrices**.

The quadratic discriminant classification rule is based on the generalized distance

to group j:

$$D_j^2(X) = (X - \bar{X}_j)' S_j^{-1} (X - \bar{X}_j) - 2\log(\text{PRIOR}_j) + \log|S_j|,$$

or equivalently, the posterior probability of membership in group j:

$$\Pr(j|X) = \frac{\exp\{-0.5 D_j^2(X)\}}{\sum_k \exp\{-0.5 D_k^2(X)\}}.$$

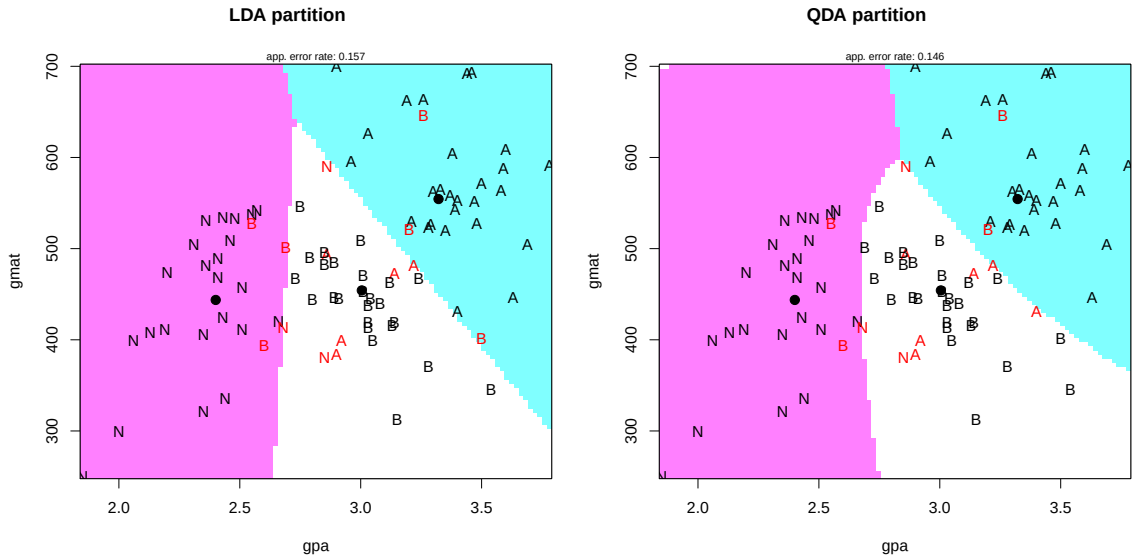
Here S_j is the sample covariance matrix from group j and $\log|S_j|$ is the log of the determinant of this covariance matrix. The determinant penalty term is large for groups having large variability. The rule is not directly tied to linear discriminant function variables, so interpretation and insight into this method is less straightforward.

There is evidence that quadratic discrimination does not improve misclassification rates in many problems with small to modest sample sizes, in part, because the quadratic rule requires an estimate of the covariance matrix for each population. A modest to large number of observations is needed to accurately estimate variances and correlations. I often compute the linear and quadratic rules, but use the linear discriminant analysis unless the quadratic rule noticeably reduces the misclassification rate.

Recall that the GPA and GMAT sample variances are roughly constant across admission groups, but the correlation between GPA and GMAT varies widely across groups.

The quadratic rule does not classify the training data noticeably better than the linear discriminant analysis. The individuals in the test data have the same classifications under both approaches. Assuming that the optimistic error rates for the two rules were "equally optimistic", I would be satisfied with the standard linear discriminant analysis, and would summarize my analysis based on this approach. Additional data is needed to decide whether the quadratic rule might help reduce the misclassification rates.

```
# classification of observations based on classification methods
#   (e.g. qda, qda) for every combination of two variables.
library(klaR)
partimat(admit ~ gmat + gpa
        , data = business
        , plot.matrix = FALSE
        , method = "lda", main = "LDA partition")
partimat(admit ~ gmat + gpa
        , data = business
        , plot.matrix = FALSE
        , method = "qda", main = "QDA partition")
```




```
library(MASS)
qda.business <- qda(admit ~ gpa + gmat
            , prior = c(0.2, 0.1, 0.7)
            , data = business)
qda.business
```

```
## Call:
## qda(admit ~ gpa + gmat, data = business, prior = c(0.2, 0.1,
##     0.7))
##
## Prior probabilities of groups:
##   A   B   N
## 0.2 0.1 0.7
##
## Group means:
##        gpa     gmat
## A 3.321875 554.4062
## B 3.004516 454.1935
## N 2.400385 443.7308
```

```
# CV = TRUE does jackknife (leave-one-out) crossvalidation
qda.business.cv <- qda(admit ~ gpa + gmat
                , prior = c(0.2, 0.1, 0.7)
                , data = business, CV = TRUE)

# Create a table of classification and posterior probabilities for each observation
classify.business <- data.frame(admit = business$admit
                  , class = qda.business.cv$class
                  , error = ""
                  , round(qda.business.cv$posterior,3))
colnames(classify.business) <- c("admit", "class", "error"
```

```
                        , paste("post", colnames(qda.business.cv$posterior), sep=""))

# error column
classify.business$error <- as.character(classify.business$error)
classify.agree <- as.character(as.numeric(business$admit)
                               - as.numeric(qda.business.cv$class))
classify.business$error[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]
```

```
# print table
#classify.business

# Assess the accuracy of the prediction
#     row = true admit, col = classified admit
pred.freq <- table(business$admit, qda.business.cv$class)
pred.freq
```

```
##
##      A  B  N
##   A 27  3  2
##   B  5 17  9
##   N  0  1 25
```

```
prop.table(pred.freq, 1) # proportions by row
```

```
##
##              A          B          N
##   A 0.84375000 0.09375000 0.06250000
##   B 0.16129032 0.54838710 0.29032258
##   N 0.00000000 0.03846154 0.96153846
```

```
# proportion correct for each category
diag(prop.table(pred.freq, 1))
```

```
##         A         B         N
## 0.8437500 0.5483871 0.9615385
```

```
# total proportion correct
sum(diag(prop.table(pred.freq)))
```

```
## [1] 0.7752809
```

```
# total error rate
1 - sum(diag(prop.table(pred.freq)))
```

```
## [1] 0.2247191
```

```
# predict the test data from the training data LDFs
pred.business <- predict(qda.business, newdata = business.test)

# Create a table of classification and posterior probabilities for each observation
classify.business.test <- data.frame(admit = business.test$admit
                   , class = pred.business$class
                   #, error = ""
                   , round(pred.business$posterior,3))
colnames(classify.business.test) <- c("admit", "class"#, "error"
```

```
                          , paste("post", colnames(pred.business$posterior), sep=""))

## error column
#classify.business.testferror <- as.character(classify.business.testferror)
#classify.agree <- as.character(as.numeric(business.testfadmit)
#                               - as.numeric(pred.businessfclass))
#classify.business.testferror[!(classify.agree == 0)] <- classify.agree[!(classify.agree == 0)]

# print table
classify.business.test
##   admit class postA postB postN
## 1    NA     N 0.043 0.038 0.919
## 2    NA     A 0.597 0.402 0.000
## 3    NA     A 0.978 0.022 0.000
## 4    NA     N 0.051 0.423 0.526
## 5    NA     B 0.292 0.708 0.000
## 6    NA     B 0.363 0.513 0.123
```