Math 145: Introduction to Biostatistics

R Pruim

Fall 2013
## Contents

### 7 Chi-Squared Tests

7.1 Goodness of Fit Testing for Simple Hypotheses ................................................. 97
7.2 A better way to test Rock-Paper-Scissors .......................................................... 102
7.3 Chi-squared for Two-Way Tables ..................................................................... 103

### 8 Comparing More Than Two Means Using ANOVA

8.1 The Basic ANOVA situation .............................................................................. 115
8.2 Confidence Intervals for One Mean At a Time ....................................................... 125
8.3 Pairwise Comparison ....................................................................................... 125

### 9 Regression

9.1 Simple Linear Regression .................................................................................. 129
9.2 Inference (Confidence Intervals and Hypothesis Tests) ......................................... 131
9.3 ANOVA for Regression ..................................................................................... 138
9.4 Making Predictions .......................................................................................... 139
9.5 Regression Cautions ......................................................................................... 141
7.1 Goodness of Fit Testing for Simple Hypotheses

Goodness of fit tests test how well a distribution fits some hypothesis.

7.1.1 Golfballs in the Yard Example

Allan Rossman used to live along a golf course. One summer he collected 500 golf balls in his back yard and tallied the number each golf ball to see if the numbers were equally likely to be 1, 2, 3, or 4.

Step 1: State the Null and Alternative Hypotheses.

- \( H_0 \): The four numbers are equally likely (in the population of golf balls driven 150–200 yards and sliced).
- \( H_a \): The four numbers are not equally likely (in the population).

If we let \( p_i \) be the proportion of golf balls with the number \( i \) on them, then we can restate the null hypothesis as

- \( H_0 : p_1 = p_2 = p_3 = p_4 = 0.25. \)
- \( H_a : \) The four numbers are not equally likely (in the population).

Step 2: Compute a test statistic. Here are Allan’s data:

<table>
<thead>
<tr>
<th>golfballs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>137</td>
<td>138</td>
<td>107</td>
<td>104</td>
</tr>
</tbody>
</table>

(Fourteen golf balls didn’t have any of these numbers and were removed from the data leaving 486 golf balls to test our null hypotheses.)

Although we made up several potential test statistics in class, the standard test statistic for this situation is...
The Chi-squared test statistic:

\[ \chi^2 = \sum \frac{(\text{observed} - \text{expected})^2}{\text{expected}} \]

There is one term in this sum for each cell in our data table, and
- observed = the tally in that cell (a count from our raw data)
- expected = the number we would “expect” if the percentages followed our null hypothesis exactly. (Note: the expected counts might not be whole numbers.)

In our particular example, we would expect 25% of the 486 (i.e., 121.5) golf balls in each category, so

\[ \chi^2 = \frac{(137 - 121.5)^2}{121.5} + \frac{(138 - 121.5)^2}{121.5} + \frac{(107 - 121.5)^2}{121.5} + \frac{(104 - 121.5)^2}{121.5} = 8.469 \]

Step 3: Compute a p-value
Our test statistic will be large when the observed counts and expected counts are quite different. It will be small when the observed counts and expected counts are quite close. So we will reject when the test statistic is large. To know how large is large enough, we need to know the sampling distribution.

If \( H_0 \) is true and the sample is large enough, then the sampling distribution for the Chi-squared test statistic will be approximately a Chi-squared distribution.
- The degrees of freedom for this type of goodness of fit test is one less than the number of cells.
- The approximation gets better and better as the sample size gets larger.

The mean of a Chi-squared distribution is equal to its degrees of freedom. This can help us get a rough idea about whether our test statistic is unusually large or not.

How unusual is it to get a test statistic at least as large as ours (8.47)? We compare to a Chi-squared distribution with 3 degrees of freedom. The mean value of such a statistic is 3, and our test statistic is almost 3 times as big, so we anticipate that our value is rather unusual, but not extremely unusual. This is the case:

\[ 1 - \text{pchisq}(8.469, \ df = 3) \]

[1] 0.0373

Step 4: Interpret the p-value and Draw a Conclusion. Based on this smallish p-value, we can reject our null hypothesis at the usual \( \alpha = 0.05 \) level. We have sufficient evidence to cast doubt on the hypothesis that all the numbers are equally common among the population of golfballs.

Automation. Of course, R can automate this whole process for us if we provide the data table and the null hypothesis. The default null hypothesis is that all the probabilities are equal, so in this case it is enough to give R the golf balls data.

`chisq.test(golfballs)`
### Chi-Squared Tests

**Chi-squared test for given probabilities**

Data: golfballs  
X-squared = 8.47, df = 3, p-value = 0.03725

---

#### 7.1.2 Plant Genetics Example

A biologist is conducting a plant breeding experiment in which plants can have one of four phenotypes. If these phenotypes are caused by a simple Mendelian model, the phenotypes should occur in a 9:3:3:1 ratio. She raises 41 plants with the following phenotypes.

<table>
<thead>
<tr>
<th>phenotype</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>20</td>
<td>10</td>
<td>7</td>
<td>4</td>
</tr>
</tbody>
</table>

Should she worry that the simple genetic model doesn’t work for her phenotypes?

```r
plants <- c(20, 10, 7, 4)  
chisq.test(plants, p = c(9/16, 3/16, 3/16, 1/16))
```

**Warning:** Chi-squared approximation may be incorrect

Chi-squared test for given probabilities

Data: plants  
X-squared = 1.97, df = 3, p-value = 0.5786

---

**Things to notice:**

- `plants <- c(20, 10, 7, 4)` is the way to enter this kind of data by hand.
- This time we need to tell R what the null hypothesis is. That’s what `p=c(9/16, 3/16, 3/16, 1/16)` is for.
- The Chi-squared distribution is only an approximation to the sampling distribution of our test statistic, and the approximation is not very good when the expected cell counts are too small. This is the reason for the warning.

---

#### 7.1.3 Small Cell Counts

The chi-squared approximation is not very good when expected cell counts are too small. Our rule of thumb will be this:

The chi-squared approximation is good enough when

- All the expected counts are at least 1.
- Most (at least 80%) of the expected counts are at least 5.

Notice that this depends on expected counts, not observed counts.
Our expected count for phenotype 4 is only \( \frac{1}{16} \cdot 41 = 2.562 \), which means that 25% of our expected counts are between 1 and 5, so our approximation will be a little crude. (The other expected counts are well in the safe range.) If you are too lazy to calculate all those expected counts yourself, you can use the `xchisq.test()` in the pkgmosaic package, which prints out some additional information about our test. If we are getting sick of typing those null probabilities over and over, we could save them, too:

```r
nullHypothesis <- c(9/16, 3/16, 3/16, 1/16)
xchisq.test(plants, p = nullHypothesis)
```

\textit{Warning: Chi-squared approximation may be incorrect}

Chi-squared test for given probabilities

data: plants
X-squared = 1.97, df = 3, p-value = 0.5786

\begin{verbatim}
   20.00 10.00  7.00  4.00
  (23.06) ( 7.69) ( 7.69) ( 2.56)
[0.407] [0.696] [0.061] [0.806]
<-0.64> < 0.83> <-0.25> < 0.90>
\end{verbatim}

key:
observed (expected)
[contribution to X-squared]
<residual>

When the expected cell counts are not large enough, we can ask R to use the empirical method (just like we did with `statTally`) instead.

```r
chisq.test(plants, p = nullHypothesis, simulate = TRUE)
```

Chi-squared test for given probabilities with simulated p-value (based on 2000 replicates)

data: plants
X-squared = 1.97, df = NA, p-value = 0.5802

This doesn’t change our overall conclusion in this case. These data are not inconsistent with the simple genetic model.

If you repeat the code above several times, you will see how the empirical p-value varies from one computation to the next. By default, R is using 2000 randomizations. This can be adjusted with the `B` argument to `chisq.test()`.

\subsection*{7.1.4 If we had more data ...}

Interestingly, if we had 10 times as much data in the same proportions, the conclusion would be very different.
morePlants <- c(200, 100, 70, 40)
chisq.test(morePlants, p = nullHypothesis)

Chi-squared test for given probabilities
data: morePlants
X-squared = 19.7, df = 3, p-value = 0.0001957

7.1.5 When there are only two categories

When there are only two categories, the Chi-squared goodness of fit test is equivalent to the 1-proportion test. So we can redo our rock-paper-scissors example using the Chi-squared test. Notice that `prop.test()` uses the count in one category and total but that `chisq.test()` uses cell counts.

```r
prop.test(66, 119, p = 1/3)
```

1-sample proportions test with continuity correction
data: x and n
X-squared = 25.2, df = 1, p-value = 5.072e-07
alternative hypothesis: true p is not equal to 0.333
95 percent confidence interval:
 0.461 0.645
sample estimates:
 p
0.555

```r
chisq.test(c(66, 53), p = c(1/3, 2/3))  # 66 + 53 = 119
```

Chi-squared test for given probabilities
data: c(66, 53)
X-squared = 26.2, df = 1, p-value = 3.042e-07

Both of these test the same hypotheses:

- $H_0: p_R = 1/3$
- $H_a: p_R \neq 1/3$

where $p_R$ is the proportion of people playing rock first.

We could also use `binom.test()`:

```r
binom.test(66, 119, p = 1/3)
```
Exact binomial test

data: x and n
number of successes = 66, number of trials = 119, p-value = 7.823e-07
alternative hypothesis: true probability of success is not equal to 0.333
95 percent confidence interval:
0.461 0.646
sample estimates:
probability of success
0.555

Although all three tests test the same hypotheses and give similar p-values (in this example), the binomial test is generally used because

- The binomial test is exact for all sample sizes while the Chi-squared test and 1-proportion test are only approximate, and the approximation is poor when sample sizes are small.
- The binomial test and 1-proportion test also produce confidence intervals.

7.2 A better way to test Rock-Paper-Scissors

But there is a better way to deal with the Rock-Paper-Scissors data. We should take advantage of the full data.

```r
chisq.test(c(66, 39, 14), p = c(1/3, 1/3, 1/3))
```

Chi-squared test for given probabilities

data: c(66, 39, 14)
X-squared = 34.1, df = 2, p-value = 3.936e-08

```r
xchisq.test(c(66, 39, 14), p = c(1/3, 1/3, 1/3))
```

Chi-squared test for given probabilities

data: c(66, 39, 14)
X-squared = 34.1, df = 2, p-value = 3.936e-08

66.00 39.00 14.00
(39.67) (39.67) (39.67)
[17.482] [ 0.011] [16.608]
< 4.18> <-0.11> <-4.08>

key:
observed (expected)
[contribution to X-squared] <residual>
7.3 Chi-squared for Two-Way Tables

7.3.1 Example: Trematodes

```r
require(abd)

Loading required package: abd
Loading required package: nlme

fish <- tally(~eaten + infection.status, Trematodes)
fish

   infection.status
  eaten   high light uninfected Total
   no     9    35     49      93
  yes    37    10     1      48
Total  46    45     50     141

chisq.test(fish)

Pearson's Chi-squared test

data:  fish
X-squared = 69.8, df = 6, p-value = 4.589e-13

Some details:

• $H_0$: The proportion of fish that are eaten is the same for each treatment group (high infection, low infection, or no infection).

• $H_A$: The proportion of fish differs among the treatment groups.

• expected count = \frac{(row total) (column total)}{grand total}

observed <- c(9, 35, 49, 37, 10, 1)
9 + 35 + 49  # row 1 total
[1] 93

37 + 10 + 1  # row 2 total
[1] 48

9 + 37  # col 1 total
[1] 46
```
\[ 35 + 10 \quad \# \text{col 2 total} \]

\[
\begin{bmatrix}
[1] 45
\end{bmatrix}
\]

\[ 49 + 1 \quad \# \text{col 3 total} \]

\[
\begin{bmatrix}
[1] 50
\end{bmatrix}
\]

\[ 9 + 35 + 49 + 37 + 10 + 1 \quad \# \text{grand total} \]

\[
\begin{bmatrix}
[1] 141
\end{bmatrix}
\]

\[
\begin{align*}
\text{expected} & \leftarrow c(93 \ast 46/141, 93 \ast 45/141, 93 \ast 50/141, 48 \ast 46/141, 48 \ast 45/141, 48 \ast 50/141) \\
\text{expected} \\
\begin{bmatrix}
[1] 30.3 & 29.7 & 33.0 & 15.7 & 15.3 & 17.0
\end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
\text{X.squared} & \leftarrow \text{sum}((\text{observed} - \text{expected})^2/\text{expected}) \\
\text{X.squared} \\
\begin{bmatrix}
[1] 69.8
\end{bmatrix}
\end{align*}
\]

- expected proportion = \( \frac{\text{row total}}{\text{grand total}} \times \frac{\text{column total}}{\text{grand total}} = \) (row proportion) (column proportion)

  - So \( H_0 \) can be stated in terms of independent rows and columns.

- This also explains the degrees of freedom. We need to estimate all but one row proportion (since they sum to 1, once we have all but one we know them all), and all but one column proportion. So

  \[
  \text{degrees of freedom} = (\text{number of rows})(\text{number of columns}) - 1 - (\text{number of rows} - 1) - (\text{number of columns} - 1) \\
  = (\text{number of rows} - 1)(\text{number of columns} - 1)
\]

- Again, \texttt{xchisq.test()} will show us some of the internal details.

\[
\text{xchisq.test(fish)}
\]

Pearson’s Chi-squared test

data:  fish
X-squared = 69.8, df = 6, p-value = 4.589e-13

\[
\begin{bmatrix}
9.00 & 35.00 & 49.00 & 93.00 \\
(30.34) & (29.68) & (32.98) & (93.00) \\
[15.01] & [0.95] & [7.78] & [0.00] \\
<-3.87> & <0.98> & <2.79> & <0.00>
\end{bmatrix}
\]
7.3.2 Example: Physicians Health Study

```r
phs <- cbind(c(104, 189), c(10933, 10845))
phs

[,1] [,2]
[1,] 104 10933
[2,] 189 10845

rownames(phs) <- c("aspirin", "placebo") # add row names
colnames(phs) <- c("heart attack", "no heart attack") # add column names
phs

heart attack no heart attack
aspirin 104 10933
placebo 189 10845

xchisq.test(phs)
```

Pearson's Chi-squared test with Yates' continuity correction

data:  phs
X-squared = 24.4, df = 1, p-value = 7.71e-07

```
  104.00 10933.00
  ( 146.52) (10890.48)
  [12.34] [ 0.17]
  <-3.51> < 0.41>

  189.00 10845.00
  ( 146.48) (10887.52)
  [12.34] [ 0.17]
```
7.3.3 Example: Fisher Twin Study

Here is an example that was treated by R. A. Fisher in an article [?] and again later in a book [?]. The data come from a study of same-sex twins where one twin had had a criminal conviction. Two pieces of information were recorded: whether the sibling had also had a criminal conviction and whether the twins were monozygotic (identical) twins and dizygotic (nonidentical) twins. Dizygotic twins are no more genetically related than any other siblings, but monozygotic twins have essentially identical DNA. Twin studies like this have often used to investigate the effects of “nature vs. nurture”.

Here are the data from the study Fisher analyzed:

<table>
<thead>
<tr>
<th></th>
<th>Convicted</th>
<th>Not convicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dizygotic</td>
<td>2</td>
<td>15</td>
</tr>
<tr>
<td>Monozygotic</td>
<td>10</td>
<td>3</td>
</tr>
</tbody>
</table>

The question is whether this gives evidence for a genetic influence on criminal convictions. If genetics had nothing to do with convictions, we would expect conviction rates to be the same for monozygotic and dizygotic twins since other factors (parents, schooling, living conditions, etc.) should be very similar for twins of either sort. So our null hypothesis is that the conviction rates are the same for monozygotic and dizygotic twins. That is,

- $H_0: p_M = p_D$.
- $H_a: p_M \neq p_D$.

```r
twins <- rbind(c(2, 15), c(10, 3))
twins

[,1] [,2]  
[1,]  2  15  
[2,] 10  3

rownames(twins) <- c("dizygotic", "monozygotic")
colnames(twins) <- c("convicted", "not convicted")
twins

convicted not convicted
dizygotic  2  15
monozygotic 10  3
```
Chi-Squared Tests

```r
chisq.test(twins)
```

Pearson's Chi-squared test with Yates' continuity correction

data:  twins
X-squared = 10.5, df = 1, p-value = 0.001221

We can get R to compute the expected counts for us, too.

```r
xchisq.test(twins)
```

Pearson's Chi-squared test with Yates' continuity correction

data:  twins
X-squared = 10.5, df = 1, p-value = 0.001221

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>2.00</td>
<td>15.00</td>
</tr>
<tr>
<td>(6.80)</td>
<td>(10.20)</td>
</tr>
<tr>
<td>[3.39]</td>
<td>[2.26]</td>
</tr>
</tbody>
</table>
| <-1.84>| < 1.50>|<br>
| 10.00  | 3.00   |
| (5.20) | (7.80) |
| [4.43] | [2.95] |
| < 2.10>| <-1.72>|

key:
observed
(expected)
[contribution to X-squared]
<residual>

This is an important check because the chi-squared approximation is not good when expected counts are too small. Recall our general rule of thumb:

- All expected counts should be at least 1.
- 80% of expected counts should be at least 5.
  
  For a 2×2 table, this means all four expected counts should be at least 5.

In our example, we are just on the good side of our rule of thumb. We could compare with the randomization p-value:

```r
chisq.test(twins, simulate = TRUE)
```

Pearson's Chi-squared test with simulated p-value (based on 2000 replicates)

data:  twins
X-squared = 13, df = NA, p-value = 0.0009995
There is also an exact test that works only in the case of a $2 \times 2$ table (much like the binomial test can be used instead of a goodness of fit test if there are only two categories). The test is called **Fisher’s Exact Test**.

```r
fisher.test(twins)
```

Fisher's Exact Test for Count Data
data: twins
p-value = 0.0005367
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
0.00333  0.36318
sample estimates:
odds ratio
  0.0469

In this case we see that the simulated p-value from the Chi-squared Test is nearly the same as the exact p-value from Fisher’s Exact Test. This is because Fisher’s test is using mathematical formulas to compute probabilities of all randomizations – it is essentially the same as doing infinitely many randomizations!

Regardless of the approach we use, it appears that the conviction rates is higher for monozygotic twins than for dizygotic twins.

Note: For a $2 \times 2$ table, we could also use the method of 2-proportions (`prop.test()`), manual resampling, or formula-based). The approximations based on the normal distribution will be poor in the same situations where the Chi-squared test gives a poor approximation.

### 7.3.4 Dealing with Low Expected Counts

What do we do if we have low expected counts?

1. We can use `simulate=TRUE` to tell R to use simulations instead of comparing the test statistic to the Chi-Squared (or normal) distribution.

2. If we have a $2 \times 2$ table, we can use Fisher’s Exact Test.

3. For goodness of fit testing with only two categories, we can use the binomial test.

4. For goodness of fit testing and larger 2-way tables, we can sometimes combine some of the cells to get larger expected counts.

5. Low expected counts often arise when analysing small data sets. Sometimes the only good solution is to design a larger study so there is more data to analyse.

### 7.3.5 Cows and Bats

In Costa Rica, the vampire bat *Desmodus rotundus* feeds on the blood of domestic cattle. If the bats respond to a hormonal signal, cows in estrous (in heat) may be bitten with a different probability than cows not in estrous. (The researcher could tell the difference by harnessing painted sponges to the undersides of bulls who would leave their mark during the night.)

Here is a summary of the data from one study investigating this:
### Chi-Squared Tests

<table>
<thead>
<tr>
<th></th>
<th>cows in estrous</th>
<th>cows not in estrous</th>
<th>row totals</th>
</tr>
</thead>
<tbody>
<tr>
<td>bitten</td>
<td>15</td>
<td>6</td>
<td>21</td>
</tr>
<tr>
<td>not bitten</td>
<td>7</td>
<td>322</td>
<td>329</td>
</tr>
<tr>
<td>column totals</td>
<td>22</td>
<td>328</td>
<td>350</td>
</tr>
</tbody>
</table>

```r
cows <- cbind(c(15, 6), c(7, 322))
cows

[,1] [,2]
[1,] 15 7
[2,] 6 322

rownames(cows) <- c("bitten", "not bitten")  # add row names
colnames(cows) <- c("in estrous", "not in estrous")  # add column names
cows

       in estrous not in estrous
bitten   15         7
not bitten   6      322

xchisq.test(cows)
```

**Warning:** Chi-squared approximation may be incorrect

Pearson’s Chi-squared test with Yates’ continuity correction

data:  cows
X-squared = 149, df = 1, p-value < 2.2e-16

```
15.00 7.00
(1.32) (20.68)
[141.77] [9.05]
<11.91> <-3.01>
```

```
6.00 322.00
(19.68) (308.32)
[9.51] [0.61]
<-3.08> <0.78>
```

key:
observed
(expected)
[contribution to X-squared]
<residual>

This time one of our expected counts is too low for our rule of thumb. So let’s use Fisher’s Exact Test instead:
Fisher's Exact Test for Count Data

```
fisher.test(cows)
```

```
data:  cows
p-value < 2.2e-16
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
    29.9  457.3
sample estimates:
  odds ratio
      108
```

Alternatively, we could use the Chi-Squared test statistic and a randomization approach:

```
chisq.test(cows, simulate = TRUE)
```

```
Pearson's Chi-squared test with simulated p-value (based on 2000 replicates)
data:  cows
X-squared = 161, df = NA, p-value = 0.0004998
```

In this example the conclusion is the same: extremely strong evidence against the null hypothesis. The p-value from the randomization test is not as small because you cannot estimate such small p-values with so few randomizations. The information we have is probably sufficient for our purposes, but to get a more accurate estimate of very small p-values requires many more randomizations.

### 7.3.6 Odds and Odds Ratio

You may be wondering about the odds ratio that appears in Fisher's exact test. Here is a brief explanation of that number.

\[
\text{odds} = O = \frac{p}{1 - p} = \frac{pn}{(1 - p)n} = \frac{\text{number of successes}}{\text{number of failures}}
\]

\[
\text{odds ratio} = OR = \frac{\text{odds}_1}{\text{odds}_2} = \frac{\frac{p_1}{1 - p_1}}{\frac{p_2}{1 - p_2}} = \frac{\frac{p_1}{1 - p_1}}{\frac{1 - p_2}{p_2}}
\]
We can estimate these quantities using data. When we do, we put hats on everything

\[
\text{estimated odds} = \hat{O} = \frac{\hat{p}}{1 - \hat{p}}
\]

\[
\text{estimated odds ratio} = \hat{OR} = \frac{\hat{O}_1}{\hat{O}_2} = \frac{\frac{\hat{p}_1}{1 - \hat{p}_1}}{\frac{\hat{p}_2}{1 - \hat{p}_2}} = \frac{\hat{p}_1}{1 - \hat{p}_1} \cdot \frac{1 - \hat{p}_2}{\hat{p}_2}
\]

The null hypothesis of Fisher’s Exact Test is that the odds ratio is 1, that is, that the two proportions are equal. `fisher.test()` uses a somewhat more complicated method to estimate odds ratio, so the formulas above won’t exactly match the odds ratio displayed in the output of `fisher.test()`, but the value will be quite close.

\[
\frac{2/15}{10/3} \quad \# \text{simple estimate of odds ratio}
\]

```
[1] 0.04
```

\[
\frac{10/3}{2/15} \quad \# \text{simple estimate of odds ratio with roles reversed}
\]

```
[1] 25
```

Notice that when \(p_1\) and \(p_1\) are small, then \(1 - p_1\) and \(1 - p_2\) are both close to 1, so

\[
\text{odds ratio} \approx \frac{p_1}{p_2} = \text{relative risk}
\]

Relative risk is easier to understand, but odds ratio has nicer statistical and mathematical properties.

Fisher’s exact test is equivalent to a randomization test using the odds ratio as the test statistic.

### 7.3.7 Survey Says

Let’s take a look at some data from the survey we administered the first week of class.

```r
survey <- read.file("http://www.calvin.edu/~rpruim/data/survey/littleSurvey.csv")
head(survey, 2)
```

<table>
<thead>
<tr>
<th>time</th>
<th>code number</th>
<th>color</th>
<th>animal</th>
<th>pulse</th>
<th>tv</th>
<th>social</th>
<th>play</th>
<th>disease</th>
<th>homework</th>
<th>numberVer</th>
<th>colorVer</th>
<th>animalVer</th>
<th>pulseVer</th>
<th>tvVer</th>
<th>date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2/1/2010</td>
<td>10:35:48</td>
<td>122112222</td>
<td>blue giraffe</td>
<td>60</td>
<td>2-4</td>
<td>Not surprising</td>
<td>No</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>2/1/2010</td>
</tr>
<tr>
<td>2 2/1/2010</td>
<td>10:36:50</td>
<td>121132121</td>
<td>blue giraffe</td>
<td>90</td>
<td>1-2</td>
<td>Not surprising</td>
<td>No</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1/2/2010</td>
</tr>
</tbody>
</table>

This includes data from past semesters as well. Let’s make a data set that contains only things from 2013:
survey2 <- subset(survey, grepl("2013", date))  # only keep rows with 2013 in the date

tally(play ~ playVer, data = survey2, format = "count")

<table>
<thead>
<tr>
<th>playVer</th>
<th>play 1</th>
<th>play 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>8</td>
<td>21</td>
</tr>
<tr>
<td>Yes</td>
<td>21</td>
<td>17</td>
</tr>
<tr>
<td>Total</td>
<td>29</td>
<td>38</td>
</tr>
</tbody>
</table>

tally(play ~ playVer, data = survey2)

<table>
<thead>
<tr>
<th>playVer</th>
<th>play 1</th>
<th>play 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>0.276</td>
<td>0.553</td>
</tr>
<tr>
<td>Yes</td>
<td>0.724</td>
<td>0.447</td>
</tr>
<tr>
<td>Total</td>
<td>1.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>

playTable <- tally(play ~ playVer, data = survey2, format = "count", margin = FALSE)

<table>
<thead>
<tr>
<th>playVer</th>
<th>play 1</th>
<th>play 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>8</td>
<td>21</td>
</tr>
<tr>
<td>Yes</td>
<td>21</td>
<td>17</td>
</tr>
</tbody>
</table>

chisq.test(playTable)

Pearson's Chi-squared test with Yates' continuity correction
data: playTable
X-squared = 4.07, df = 1, p-value = 0.04373

chisq.test(playTable, simulate = TRUE)

Pearson's Chi-squared test with simulated p-value (based on 2000 replicates)
data: playTable
X-squared = 5.13, df = NA, p-value = 0.02449

fisher.test(playTable)

Fisher's Exact Test for Count Data
data: playTable
p-value = 0.02788  
alternative hypothesis: true odds ratio is not equal to 1  
95 percent confidence interval:  
0.0947 0.9659  
sample estimates:  
odds ratio  
0.314  

prop.test(play ~ playVer, data = survey2)  

2-sample test for equality of proportions with continuity correction  
data: x  
X-squared = 4.07, df = 1, p-value = 0.04373  
alternative hypothesis: two.sided  
95 percent confidence interval:  
-0.5340 -0.0195  
sample estimates:  
prop 1 prop 2  
0.276 0.553
8.1 The Basic ANOVA situation

• Two variables: categorical explanatory and quantitative response
  – Can be used in either experimental or observational designs.

• Main Question: Does the population mean response depend on the (treatment) group?
  – $H_0$: the population group means are all equal ($\mu_1 = \mu_2 = \cdots = \mu_k$)
  – $H_a$: the population group means are not all equal

• If categorical variable has only 2 values, we already have a method: 2-sample $t$-test
  – ANOVA allows for 3 or more groups (sub-populations)

• $F$ statistic compares within group variation (how different are individuals in the same group?) to between group variation (how different are the different group means?)

• ANOVA assumes that each group is normally distributed with the same (population) standard deviation.
  – Check normality with normal quantile plots (of residuals)
  – Check equal standard deviation using 2:1 ratio rule (largest standard deviation at most twice the smallest standard deviation).

8.1.1 An Example: Ants and Sandwiches

```
favstats(Ants ~ Filling, data = SandwichAnts)

<table>
<thead>
<tr>
<th></th>
<th>min</th>
<th>Q1</th>
<th>median</th>
<th>Q3</th>
<th>max</th>
<th>mean</th>
<th>sd</th>
<th>n</th>
<th>missing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ham &amp; Pickles</td>
<td>34.0</td>
<td>42.0</td>
<td>51.0</td>
<td>55.2</td>
<td>65</td>
<td>49.2</td>
<td>10.79</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Peanut Butter</td>
<td>19</td>
<td>21.8</td>
<td>30.5</td>
<td>44.0</td>
<td>59</td>
<td>34.0</td>
<td>14.63</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Vegemite</td>
<td>18</td>
<td>24.0</td>
<td>30.0</td>
<td>39.0</td>
<td>42</td>
<td>30.8</td>
<td>9.25</td>
<td>8</td>
<td>0</td>
</tr>
</tbody>
</table>
```

```
xyplot(Ants ~ Filling, SandwichAnts, type = c("p", "a"))
bwplot(Ants ~ Filling, SandwichAnts)
```
Question: Are these differences significant? Or would we expect sample differences this large by random chance even if (in the population) the mean amount of shift is equal for all three groups?

Whether differences between the groups are significant depends on three things:

1. the difference in the means
2. the amount of variation within each group
3. the sample sizes

\texttt{anova(lm(Ants ~ Filling, SandwichAnts))}

Analysis of Variance Table

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filling</td>
<td>2</td>
<td>1561</td>
<td>780</td>
<td>5.63</td>
<td>0.011</td>
</tr>
<tr>
<td>Residuals</td>
<td>21</td>
<td>2913</td>
<td>139</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The p-value listed in this output is the p-value for our null hypothesis that the mean population response is the same in each treatment group. In this case we would reject the null hypothesis at either the $\alpha = 0.05$ or $\alpha = 0.01$ levels.

In the next section we'll look at this test in more detail, but notice that if you know the assumptions of a test, the null hypothesis being tested, and the p-value, you can generally interpret the results even if you don't know all the details of how the test statistic is computed.
8.1.2 The ANOVA test statistic

The ingredients

The ANOVA test statistic (called $F$) is based on three ingredients:

1. how different the group means are (between group differences)
2. the amount of variability within each group (within group differences)
3. sample size

Each of these will be involved in the calculation of $F$.

The $F$ statistic is a bit complicated to compute. We’ll generally let the computer handle that for us. But is useful to see one small example to see how the ingredients are baked into a test statistic.

A Small Dataset

Suppose we have three groups with the following response values.

- Group A: 5.3, 6.0, 6.7 \[ \bar{y}_1 = 6.0 \]
- Group B: 5.5, 6.2, 6.4, 5.5 \[ \bar{y}_2 = 5.8 \]
- Group C: 7.5, 7.2, 7.8 \[ \bar{y}_3 = 7.5 \]

Computing $F$

First, let’s represent our small data set in the more usual way:

```r
response <- c(5.4, 6.1, 6.8, 5.4, 6.1, 6.3, 5.4, 7.5, 7.2, 7.8)
small <- data.frame(response = response, group = group)
```

<table>
<thead>
<tr>
<th>response</th>
<th>group</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.4</td>
<td>A</td>
</tr>
<tr>
<td>6.1</td>
<td>A</td>
</tr>
<tr>
<td>6.8</td>
<td>A</td>
</tr>
<tr>
<td>5.4</td>
<td>B</td>
</tr>
<tr>
<td>6.1</td>
<td>B</td>
</tr>
<tr>
<td>6.3</td>
<td>B</td>
</tr>
<tr>
<td>5.4</td>
<td>B</td>
</tr>
<tr>
<td>7.5</td>
<td>C</td>
</tr>
<tr>
<td>7.2</td>
<td>C</td>
</tr>
<tr>
<td>7.8</td>
<td>C</td>
</tr>
</tbody>
</table>

Now let’s compute the grand mean and group means.
Comparing More Than Two Means Using ANOVA

```r
mean(response, data = small)  # grand mean
[1] 6.4

mean(response ~ group, data = small)  # group means

A B C
6.1 5.8 7.5
```

And add those to our data frame

```r
small <- transform(small, groupMean = c(6.1, 6.1, 6.1, 5.8, 5.8, 5.8, 5.8, 7.5, 7.5, 7.5))
small <- transform(small, grandMean = rep(6.4, 10))
small

<table>
<thead>
<tr>
<th>response</th>
<th>group</th>
<th>groupMean</th>
<th>grandMean</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.4</td>
<td>A</td>
<td>6.1</td>
</tr>
<tr>
<td>2</td>
<td>6.1</td>
<td>A</td>
<td>6.1</td>
</tr>
<tr>
<td>3</td>
<td>6.8</td>
<td>A</td>
<td>6.1</td>
</tr>
<tr>
<td>4</td>
<td>5.4</td>
<td>B</td>
<td>5.8</td>
</tr>
<tr>
<td>5</td>
<td>6.1</td>
<td>B</td>
<td>5.8</td>
</tr>
<tr>
<td>6</td>
<td>6.3</td>
<td>B</td>
<td>5.8</td>
</tr>
<tr>
<td>7</td>
<td>5.4</td>
<td>B</td>
<td>5.8</td>
</tr>
<tr>
<td>8</td>
<td>7.5</td>
<td>C</td>
<td>7.5</td>
</tr>
<tr>
<td>9</td>
<td>7.2</td>
<td>C</td>
<td>7.5</td>
</tr>
<tr>
<td>10</td>
<td>7.8</td>
<td>C</td>
<td>7.5</td>
</tr>
</tbody>
</table>
```

Now we are ready to put the ingredients together:

- **Between group variability:** \( G = \text{groupMean} - \text{grandMean} \)
  
  This measures how different a group is from the overall average.

- **Within group variability:** \( E = \text{response} - \text{groupMean} \)
  
  This measures how different and individual is from its group average. \( E \) stands for “error”, but just as in “standard error” it is not a “mistake”. It is simply measure how different an individual response is from the model prediction (in this case, the group mean).

  The individual values of \( E \) are called **residuals**.

```r
small <- transform(small, M = groupMean - grandMean)
small <- transform(small, E = response - groupMean)
small

<table>
<thead>
<tr>
<th>response</th>
<th>group</th>
<th>groupMean</th>
<th>grandMean</th>
<th>M</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.4</td>
<td>A</td>
<td>6.1</td>
<td>6.4</td>
<td>-0.3</td>
</tr>
<tr>
<td>2</td>
<td>6.1</td>
<td>A</td>
<td>6.1</td>
<td>6.4</td>
<td>-0.3</td>
</tr>
<tr>
<td>3</td>
<td>6.8</td>
<td>A</td>
<td>6.1</td>
<td>6.4</td>
<td>-0.3</td>
</tr>
<tr>
<td>4</td>
<td>5.4</td>
<td>B</td>
<td>5.8</td>
<td>6.4</td>
<td>-0.6</td>
</tr>
<tr>
<td>5</td>
<td>6.1</td>
<td>B</td>
<td>5.8</td>
<td>6.4</td>
<td>-0.6</td>
</tr>
<tr>
<td>6</td>
<td>6.3</td>
<td>B</td>
<td>5.8</td>
<td>6.4</td>
<td>-0.6</td>
</tr>
</tbody>
</table>
```
As we did with variance, we will square these differences:

```r
table <- transform(small, M2 = (groupMean - grandMean)^2)
table <- transform(small, E2 = (response - groupMean)^2)
table
```

<table>
<thead>
<tr>
<th>response</th>
<th>group</th>
<th>groupMean</th>
<th>grandMean</th>
<th>M</th>
<th>E</th>
<th>M2</th>
<th>E2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.4</td>
<td>A</td>
<td>6.1</td>
<td>6.4</td>
<td>-0.3</td>
<td>-0.7</td>
<td>0.09</td>
<td>0.49</td>
</tr>
<tr>
<td>6.1</td>
<td>A</td>
<td>6.1</td>
<td>6.4</td>
<td>-0.3</td>
<td>0.0</td>
<td>0.09</td>
<td>0.00</td>
</tr>
<tr>
<td>6.8</td>
<td>A</td>
<td>6.1</td>
<td>6.4</td>
<td>-0.3</td>
<td>0.7</td>
<td>0.09</td>
<td>0.49</td>
</tr>
<tr>
<td>5.4</td>
<td>B</td>
<td>5.8</td>
<td>6.4</td>
<td>-0.6</td>
<td>-0.4</td>
<td>0.36</td>
<td>0.16</td>
</tr>
<tr>
<td>6.1</td>
<td>B</td>
<td>5.8</td>
<td>6.4</td>
<td>-0.6</td>
<td>0.3</td>
<td>0.36</td>
<td>0.09</td>
</tr>
<tr>
<td>6.3</td>
<td>B</td>
<td>5.8</td>
<td>6.4</td>
<td>-0.6</td>
<td>0.5</td>
<td>0.36</td>
<td>0.25</td>
</tr>
<tr>
<td>5.4</td>
<td>B</td>
<td>5.8</td>
<td>6.4</td>
<td>-0.6</td>
<td>-0.4</td>
<td>0.36</td>
<td>0.16</td>
</tr>
<tr>
<td>7.5</td>
<td>C</td>
<td>7.5</td>
<td>6.4</td>
<td>1.1</td>
<td>0.0</td>
<td>1.21</td>
<td>0.00</td>
</tr>
<tr>
<td>7.2</td>
<td>C</td>
<td>7.5</td>
<td>6.4</td>
<td>1.1</td>
<td>-0.3</td>
<td>1.21</td>
<td>0.09</td>
</tr>
<tr>
<td>7.8</td>
<td>C</td>
<td>7.5</td>
<td>6.4</td>
<td>1.1</td>
<td>0.3</td>
<td>1.21</td>
<td>0.09</td>
</tr>
</tbody>
</table>

And then add them up (SS stands for “sum of squares”)

```r
SSM <- sum( ~M2, data=table); SSM  # also called SSG

[1] 5.34

0.16 * 3 + 0.36 * 4 + 1.21 * 3 # alternate way to calculate

[1] 5.55

SSE <- sum( ~E2, data=table); SSE

[1] 1.82
```

Now we adjust for sample size using

- \( MSM = SSM/DFM = SSM/(\text{number of groups} - 1) \)
- \( MSE = SSE/DFE = SSE/(n - \text{number of groups}) \)

MS stands for “mean square”

```r
MSM <- SSM / ( 3-1); MSM

[1] 2.67
```
Finally, our test statistic is

\[ F = \frac{MSM}{MSE} \]

Finally, our test statistic is

\[ F = \frac{MSM}{MSE} \]

\[ F \leftarrow \frac{MSM}{MSE} \]

8.1.3 P-values from the randomization distribution

We can now compute a p-value by comparing our value of \( F \) (10.269) to a randomization distribution. If the null hypothesis is true, the three groups are really just one big group and the group labels are meaningless, so we can shuffle the group labels to get a randomization distribution:

```r
small.rand <- do(1000) * anova(lm(response ~ shuffle(group), data = small))
tally(~(F >= 10.27), data = small.rand)
```

<table>
<thead>
<tr>
<th></th>
<th>TRUE</th>
<th>FALSE</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>16</td>
<td>984</td>
<td>1000</td>
</tr>
</tbody>
</table>

```r
prop(~(F >= 10.27), data = small.rand)
```

<table>
<thead>
<tr>
<th>TRUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.016</td>
</tr>
</tbody>
</table>

```r
histogram(~F, data = small.rand, v = 10.27)
```

Since our estimated p-value is small, we have enough evidence in the data to reject the null hypothesis.
8.1.4 P-values without simulations

Under certain conditions, the $F$ statistic has a known distribution (called the $F$ distribution). Those conditions are

1. The null hypothesis is true (i.e., each group has the same mean)
2. Each group is sampled from a normal population
3. Each population group has the same standard deviation

When these conditions are met, we can use the $F$-distribution to compute the p-value without generating the randomization distribution.

- $F$ distributions have two parameters – the degrees of freedom for the numerator and for the denominator. In our example, this is 2 for the numerator and 7 for the denominator.
- When $H_0$ is true, the numerator and denominator both have a mean of 1, so $F$ will tend to be close to 1.
- When $H_0$ is false, there is more difference between the groups, so the numerator tends to be larger. This means we will reject the null hypothesis when $F$ gets large enough.
- The p-value is computed using `pf()`.

```
1 - pf(F, 2, 7)
```

```
[1] 0.00828
```

8.1.5 Getting R to do the work

Of course, R can do all of this work for us. We saw this earlier. Here it is again in a slightly different way:

```R
small.model <- lm(response ~ group, small)
anova(small.model)
```

```
Analysis of Variance Table

Response: response
   Df  Sum Sq Mean Sq F value Pr(>F)
group   2   5.34  2.670   10.3 0.0083
Residuals 7   1.82  0.263
```

`lm()` stands for “linear model” and can be used to fit a wide variety of situations. It knows to do 1-way ANOVA by looking at the types of variables involved.

The `anova()` prints the ANOVA table. Notice how DFM, SSM, MSM, DFE, SSE, and MSE show up in this table as well as $F$ and the p-value.

Here is the ANOVA table for the SandwichAnts example again:
Comparing More Than Two Means Using ANOVA

```r
anova(lm(Ants ~ Filling, data = SandwichAnts))
```

**Analysis of Variance Table**

Response: Ants

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filling</td>
<td>2</td>
<td>1561</td>
<td>780</td>
<td>5.63</td>
<td>0.011</td>
</tr>
<tr>
<td>Residuals</td>
<td>21</td>
<td>2913</td>
<td>139</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Proportion of Variation Explained**

The `summary()` function can be used to provide a different summary of the ANOVA model:

```r
summary(small.model)
```

**Call:**

`lm(formula = response ~ group, data = small)`

**Residuals:**

Min 1Q Median 3Q Max
-0.700 -0.375 0.000 0.300 0.700

**Coefficients:**

|            | Estimate | Std. Error | t value | Pr(>|t|) |
|------------|----------|------------|---------|----------|
| (Intercept)| 6.100    | 0.294      | 20.72   | 1.5e-07  |
| groupB     | -0.300   | 0.389      | -0.77   | 0.466    |
| groupC     | 1.400    | 0.416      | 3.36    | 0.012    |

Residual standard error: 0.51 on 7 degrees of freedom
Multiple R-squared: 0.746, Adjusted R-squared: 0.673
F-statistic: 10.3 on 2 and 7 DF, p-value: 0.00828

The ratio

\[ R^2 = \frac{SSM}{SSM + SSE} = \frac{SSM}{SST} \]

measures the proportion of the total variation that is explained by the grouping variable (treatment).

### 8.1.6 An Example: Jet Lag

```r
require(abd)
favstats(shift ~ treatment, data = JetLagKnees)
```

<table>
<thead>
<tr>
<th></th>
<th>min</th>
<th>Q1</th>
<th>median</th>
<th>Q3</th>
<th>max</th>
<th>mean</th>
<th>sd</th>
<th>n</th>
<th>missing</th>
</tr>
</thead>
<tbody>
<tr>
<td>control</td>
<td>-1.27</td>
<td>-0.65</td>
<td>-0.485</td>
<td>0.24</td>
<td>0.53</td>
<td>-0.309</td>
<td>0.618</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>eyes</td>
<td>-2.83</td>
<td>-1.78</td>
<td>-1.480</td>
<td>-1.10</td>
<td>-0.78</td>
<td>-1.551</td>
<td>0.706</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>knee</td>
<td>-1.61</td>
<td>-0.76</td>
<td>-0.290</td>
<td>0.17</td>
<td>0.73</td>
<td>-0.336</td>
<td>0.791</td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>
Comparing More Than Two Means Using ANOVA

```r
xyplot(shift ~ treatment, data = JetLagKnees, type = c("p", "a"))
bwplot(shift ~ treatment, data = JetLagKnees)
```

![Graph](image)

```r
jetlag.model <- lm(shift ~ treatment, data = JetLagKnees)
anova(jetlag.model)
```

Analysis of Variance Table

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>treatment</td>
<td>2</td>
<td>7.22</td>
<td>3.61</td>
<td>7.29</td>
<td>0.0045</td>
</tr>
<tr>
<td>Residuals</td>
<td>19</td>
<td>9.42</td>
<td>0.50</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagnostic Plots

We can use `plot()` to create some diagnostic plots. The first two are the most important for our purposes. They provide a residual plot and a normal-quantile plot of residuals.

```r
plot(small.model, w = 1:2)
```
Comparing More Than Two Means Using ANOVA

The residual plot shows the residual broken down by groups (actually by group means). Ideally we should see similar patterns of variation in each cluster. The second plot is a normal-quantile plot of the residuals. Since the differences from the group means should be normally distributed with the same standard deviation, we can combine all the residuals into a single normal-quantile plot.

8.1.7 Back to Jet Lag

Here is all the code needed to analyze the jet lag experiment

```r
jetlag.model <- lm(shift ~ treatment, JetLagKnees)
anova(jetlag.model)
summary(jetlag.model)
```

Analysis of Variance Table

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>treatment</td>
<td>2</td>
<td>7.22</td>
<td>3.61</td>
<td>7.29</td>
<td>0.0045</td>
</tr>
<tr>
<td>Residuals</td>
<td>19</td>
<td>9.42</td>
<td>0.50</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

summary(jetlag.model)

Call:
`lm(formula = shift ~ treatment, data = JetLagKnees)`

Residuals:

```
    Min   1Q Median   3Q   Max
-1.2786 -0.3613  0.0386  0.6115  1.0657
```

Coefficients:

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.309     0.249   -1.24  0.2299
```

Last Modified: November 25, 2013
8.2 Confidence Intervals for One Mean At a Time

We can construct a confidence interval for any of the means by just taking a subset of the data and using \texttt{t.test()}, but there are some problems with this approach. Most importantly,

We were primarily interested in comparing the means across the groups. Often people will display confidence intervals for each group and look for “overlapping” intervals. But this is not the best way to look for differences.

Nevertheless, you will sometimes see graphs showing multiple confidence intervals and labeling them to indicate which means appear to be different from which. (See the solution to problem 15.3 for an example.)

8.3 Pairwise Comparison

We really want to compare groups in pairs, and we have a method for this: 2-sample \textit{t}. But we need to make a couple adjustments to the two-sample \textit{t}.

1. We will use a new formula for standard error that makes use of all the data (even from groups not involved in the pair).
2. We also need to adjust the critical value to take into account the fact that we are (usually) making multiple comparisons.

### 8.3.1 The Standard Error

\[
SE = \sqrt{MSE \left( \frac{1}{n_i} + \frac{1}{n_j} \right)} = \sqrt{MSE \left( \frac{1}{n_i} + \frac{1}{n_j} \right)}
\]

where \( n_i \) and \( n_j \) are the sample sizes for the two groups being compared. Basically, \( \sqrt{MSE} \) is taking the place of \( s \) in our usual formula. The degrees of freedom for this estimate is

\[
DFE = \text{total sample size} - \text{number of groups}.
\]

Ignoring the multiple comparisons issue, we can now compute confidence intervals or hypothesis tests just as before.

- **confidence interval:**
  \[\bar{y}_i - \bar{y}_j \pm t \cdot SE\]

- **test statistic** (for \( H_0: \mu_1 - \mu_2 = 0 \)):
  \[t = \frac{\bar{y}_i - \bar{y}_j}{SE} .\]

### 8.3.2 The Multiple Comparisons Problem

Suppose we have 5 groups in our study and we want to make comparisons between each pair of groups. That’s \( 4 + 3 + 2 + 1 = 10 \) pairs. If we made 10 independent 95% confidence intervals, the probability that all of the cover the appropriate parameter is 0.599:

\[0.95^{10}\]

\[\text{[1]} \ 0.599\]

So we have **family-wide error rate** of nearly 40%.

We can correct for this by adjusting our critical value. Let’s take a simple example: just two 95% confidence intervals. The probability that both cover (assuming independence) is

\[0.95^2\]

\[\text{[1]} \ 0.902\]

Now suppose we want both intervals to cover 95% instead of 90.2% of the time. We could get this by forming two 97.5% confidence intervals.

\[\sqrt{0.95}\]

\[\text{[1]} \ 0.975\]
This means we need a larger value for $t^*$ for each interval.

The ANOVA situation is a little bit more complicated because

- There are more than two comparisons.
- The different comparisons are not independent (because they all come from the same data set.)

We will briefly describe two ways to make an adjustment for multiple comparisons.

### 8.3.3 Bonferroni Corrections – An Easy Over-adjustment

Bonferroni’s idea is simple: Simple divide the desired family-wise error rate by the number of tests or intervals. This is an over-correction, but it is easy to do, and is used in many situations where a better method is not known or a quick estimate is desired.

Here is a table showing a few Bonferroni corrections for looking at all pairwise comparisons.

<table>
<thead>
<tr>
<th>number of groups</th>
<th>number of pairs of groups</th>
<th>family-wise error rate</th>
<th>individual error rate</th>
<th>confidence level for determining $t^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>.05</td>
<td>0.017</td>
<td>0.983</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>.05</td>
<td>0.008</td>
<td>0.992</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>.05</td>
<td>0.005</td>
<td>0.995</td>
</tr>
</tbody>
</table>

Similar adjustments could be made for looking at only a special subset of the pairwise comparisons.

### 8.3.4 Tukey’s Honest Significant Differences

Tukey’s Honest Significant Differences is a better adjustment method specifically designed for making all pairwise comparisons in an ANOVA situation. (It takes into account the fact that the tests are not independent.) R can compute Tukey’s Honest Significant Differences easily.

```r
TukeyHSD(lm(shift ~ treatment, JetLagKnees))
```

Tukey multiple comparisons of means  
95% family-wise confidence level

Fit: aov(formula = x)

<table>
<thead>
<tr>
<th>treatment</th>
<th>diff</th>
<th>lwr</th>
<th>upr</th>
<th>p adj</th>
</tr>
</thead>
<tbody>
<tr>
<td>eyes-control</td>
<td>-1.243</td>
<td>-2.168</td>
<td>-0.317</td>
<td>0.008</td>
</tr>
<tr>
<td>knee-control</td>
<td>-0.027</td>
<td>-0.953</td>
<td>0.899</td>
<td>0.997</td>
</tr>
<tr>
<td>knee-eyes</td>
<td>1.216</td>
<td>0.260</td>
<td>2.172</td>
<td>0.012</td>
</tr>
</tbody>
</table>
Tukey’s method adjusts the confidence intervals, making them a bit wider, to give them the desired family-wide error rate. Tukey’s method also adjusts p-values (making them larger), so that when the means are all the same, there is only a 5% chance that a sample will produce any p-values below 0.05.

In this example we see that the eye group differs significantly from control group and also from the knee group, but that the knee and control groups are not significantly different. (We can tell this by seeing which confidence intervals contain 0 or by checking which adjusted p-values are less than 0.05.)

8.3.5 Other Adjustments

There are similar methods for testing other sets of multiple comparisons. Testing “one against all the others” goes by the name of Dunnet’s method, for example. This is useful when one group represents a control against which various treatments are being compared.
9.1 Simple Linear Regression

9.1.1 The Simple Linear Regression Model

\[ Y = \beta_0 + \beta_1 x + \epsilon \quad \text{where } \epsilon \sim \text{Norm}(0, \sigma). \]

In other words:

- The mean response for a given predictor value \( x \) is given by a linear formula
  \[ \text{mean response} = \beta_0 + \beta_1 x \]
- The distribution of all responses for a given predictor value \( x \) is normal.
- The standard deviation of the responses is the same for each predictor value.

One of the goals in simple linear regression is to estimate this linear relationship – that is to estimate the intercept and the slope.

9.1.2 The Least Squares Method

Of course, there are lots of lines. We want to determine the line that fits the data best. But what does that mean?

The usual method is called the **method of least squares** and chooses the line that has the *smallest possible sum of squares of residuals*, where residuals are defined by

\[ \text{residual} = \text{observed response} - \text{predicted response} \]

For a line with equation \( y = b_0 + b_1 x \), this would be

\[ e_i = y_i - (b_0 + b_1 x) \]

Simple calculus (that you don’t need to know) allows us to compute the best \( b_0 \) and \( b_1 \) possible. These best values define the least squares regression line. Fortunately, statistical software packages do all this work for us. In R, the command that does this is \texttt{lm(.)}. 
9.1.3 Getting R to Compute Regression Lines

Florida Lakes

```r
xyplot(AvgMercury ~ pH, data = FloridaLakes, type = c("p", "r"))
lm(AvgMercury ~ pH, data = FloridaLakes)
```

Call:
`lm(formula = AvgMercury ~ pH, data = FloridaLakes)`

Coefficients:
```
          (Intercept) pH
1.531  -0.152
```

You can get terser output with

```r
coef(lm(AvgMercury ~ pH, data = FloridaLakes)) # just show me the coefficients
```

```
          (Intercept) pH
1.531  -0.152
```

From these coefficients, we see that our regression equation is

\[
\text{AvgMercury} = 1.531 - 0.152 \cdot \text{pH}
\]

So for example, this suggests that the average average mercury level (yes, that’s two averages\(^1\)) for lake with a pH of 6 is approximately

\[
\text{AvgMercury} = 1.531 - 0.152 \cdot 6.0 = 0.617
\]

**Inkjet Printers**

Here’s another example in which we want to predict the price of an inkjet printer from the number of pages it prints per minute (ppm).

\(^1\)For each lake, the average mercury level is calculated. Different lakes will have different average mercury levels. Our regression line is estimating the average of these averages for lakes with a certain pH.
You can get terser output with

```r
coef(lm(Price ~ PPM, data = InkjetPrinters))
```

So our regression equation is

\[
\text{Price} = -94.222 + 90.878 \cdot \text{PPM}
\]

For example, this suggests that the average price for inkjet printers that print 3 pages per minute is

\[
\text{Price} = -94.222 + 90.878 \cdot 3.0 = 178.412
\]

9.2 Inference (Confidence Intervals and Hypothesis Tests)

9.2.1 Bootstrap

So how good are these estimates? We would like have interval estimates rather than just point estimates. One way to get interval estimates for the coefficients is to use the bootstrap.

Florida Lakes
boot.lakes <- do(1000) * lm(AvgMercury ~ pH, data = resample(FloridaLakes))
head(boot.lakes, 2)

    Intercept   pH sigma r.squared
1  1.32  -0.119  0.268   0.187
2  1.45  -0.156  0.240   0.472

dotPlot(~pH, data = boot.lakes, width = 0.003)
dotPlot(~Intercept, data = boot.lakes, width = 0.02)
histogram(~pH, data = boot.lakes, width = 0.01)
histogram(~Intercept, data = boot.lakes, width = 0.1)

cdata(0.95, pH, boot.lakes)

    low   hi
95% -0.21  -0.105

cdata(0.95, Intercept, boot.lakes)

    low   hi
95%  1.21   1.94

Inkjet Printers
Regression

\[\text{boot.printers <- do(1000) * lm(Price ~ PPM, data = resample(InkjetPrinters))}\]
\[\text{head(boot.printers, 2)}\]

<table>
<thead>
<tr>
<th>Intercept</th>
<th>PPM</th>
<th>sigma</th>
<th>r.squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>-125</td>
<td>104</td>
<td>54.2</td>
<td>0.622</td>
</tr>
<tr>
<td>-153</td>
<td>113</td>
<td>54.2</td>
<td>0.714</td>
</tr>
</tbody>
</table>

\[\text{histogram(˜PPM, data = boot.printers)}\]
\[\text{histogram(˜Intercept, data = boot.printers)}\]
\[\text{cdata(0.95, PPM, boot.printers)}\]

\[
\begin{array}{cc}
\text{low} & \text{hi} \\
95\% & 48 & 131 \\
\end{array}
\]
\[\text{cdata(0.95, Intercept, boot.printers)}\]

\[
\begin{array}{cc}
\text{low} & \text{hi} \\
95\% & -214 & 16.6 \\
\end{array}
\]

9.2.2 Using Standard Errors

We can also compute confidence intervals using

\[
\text{estimate \pm t, SE}
\]

The formulas for SE are quite a bit more complicated in this case, but R will calculate them for us. For \(t\), we use \(n - 2\) degrees of freedom. (The other two degrees of freedom go for estimating the intercept and the slope).

Florida Lakes

\[\text{summary(lm(AvgMercury ~ pH, data = resample(FloridaLakes)))}\]

Call:
\[\text{lm(formula = AvgMercury ~ pH, data = resample(FloridaLakes))}\]
Residuals:
Min 1Q Median 3Q Max
-0.4768 -0.2030 -0.0665 0.1818 0.6746

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.4082 0.2110 6.67 1.8e-08
pH -0.1310 0.0325 -4.04 0.00018

Residual standard error: 0.315 on 51 degrees of freedom
Multiple R-squared: 0.242, Adjusted R-squared: 0.227
F-statistic: 16.3 on 1 and 51 DF, p-value: 0.000182

The $t$ value is based on $DFE$, the degrees of freedom for the errors (residuals). For simple linear regression, the error degrees of freedom is $n - 2 = 51$. For a 95% confidence interval, we first compute $t_*$:

```r
> t.star <- qt(0.975, df = 51)
> t.star
[1] 2.01
```

So we get the following confidence intervals for intercept

\[
1.63 \pm t_* SE
\]
\[
1.63 \pm 2.018 \cdot 0.2118
\]
\[
1.63 \pm 0.425
\]

and the slope

\[
-0.153 \pm t_* SE
\]
\[
-0.1532.008 \cdot 0.00319
\]
\[
-0.153 \pm 0.064
\]

Of course, R can compute the confidence intervals for us:

```r
> confint(lm(AvgMercury ~ pH, data = resample(FloridaLakes))) # 95% CI
   2.5 % 97.5 %
(Intercept) 0.966  1.865
pH          -0.206 -0.071

> confint(lm(AvgMercury ~ pH, data = resample(FloridaLakes)), level = 0.99) # 99% CI
   0.5 % 99.5 %
(Intercept) 1.290  2.336
pH          -0.267 -0.106
```

In fact, the `confint()` function can be applied to data sets containing bootstrap distributions, too. This makes it even easier to calculate bootstrap confidence intervals. We even have a choice between (a) using the standard error as estimated by taking the standard deviation of the bootstrap distribution or (b) using the percentile method:
Regression

```r
confint(boot.lakes) # 95% CIs for each parameter
```

<table>
<thead>
<tr>
<th>name</th>
<th>lower</th>
<th>upper</th>
<th>level</th>
<th>method</th>
<th>estimate</th>
<th>margin.of.error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1.182</td>
<td>1.911</td>
<td>0.95</td>
<td>stderr</td>
<td>1.547</td>
<td>0.3645</td>
</tr>
<tr>
<td>pH</td>
<td>-0.205</td>
<td>-0.103</td>
<td>0.95</td>
<td>stderr</td>
<td>-0.154</td>
<td>0.0514</td>
</tr>
<tr>
<td>sigma</td>
<td>0.223</td>
<td>0.330</td>
<td>0.95</td>
<td>stderr</td>
<td>0.277</td>
<td>0.0535</td>
</tr>
<tr>
<td>r.squared</td>
<td>0.155</td>
<td>0.524</td>
<td>0.95</td>
<td>stderr</td>
<td>0.339</td>
<td>0.1844</td>
</tr>
</tbody>
</table>

```r
cconfint(boot.lakes, method = "perc") # 95% CIs for each parameter; percentile method
```

<table>
<thead>
<tr>
<th>name</th>
<th>lower</th>
<th>upper</th>
<th>level</th>
<th>method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1.207</td>
<td>1.944</td>
<td>0.95</td>
<td>quantile</td>
</tr>
<tr>
<td>pH</td>
<td>-0.210</td>
<td>-0.105</td>
<td>0.95</td>
<td>quantile</td>
</tr>
<tr>
<td>sigma</td>
<td>0.222</td>
<td>0.325</td>
<td>0.95</td>
<td>quantile</td>
</tr>
<tr>
<td>r.squared</td>
<td>0.164</td>
<td>0.522</td>
<td>0.95</td>
<td>quantile</td>
</tr>
</tbody>
</table>

```r
cconfint(boot.lakes, "pH", level = 0.98, method = c("stderr", "perc")) # 98% CI just for pH, both methods
```

<table>
<thead>
<tr>
<th>name</th>
<th>lower</th>
<th>upper</th>
<th>level</th>
<th>method</th>
<th>estimate</th>
<th>margin.of.error</th>
</tr>
</thead>
<tbody>
<tr>
<td>pH</td>
<td>-0.215</td>
<td>-0.0930</td>
<td>0.98</td>
<td>stderr</td>
<td>-0.154</td>
<td>0.061</td>
</tr>
<tr>
<td>pH</td>
<td>-0.225</td>
<td>-0.0962</td>
<td>0.98</td>
<td>quantile</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

Inkjet Printers

```r
summary(lm(Price ~ PPM, data = resample(InkjetPrinters)))
```

Call: lm(formula = Price ~ PPM, data = resample(InkjetPrinters))

Residuals:
  Min 1Q Median 3Q Max
-62.1 -44.9 -26.4 44.7 91.8

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -54.4   83.5    -0.65   0.523
PPM          76.9   28.8     2.67   0.016

Residual standard error: 59.9 on 18 degrees of freedom
Multiple R-squared: 0.284, Adjusted R-squared: 0.244
F-statistic: 7.14 on 1 and 18 DF, p-value: 0.0156

```r
cconfint(lm(Price ~ PPM, data = resample(InkjetPrinters)), "PPM")
```

<table>
<thead>
<tr>
<th>2.5 %</th>
<th>97.5 %</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPM</td>
<td>59.7</td>
</tr>
</tbody>
</table>
### 9.2.3 Hypothesis Tests

The summary of linear models includes the results of some hypothesis tests:

```r
summary(lm(AvgMercury ~ pH, data = FloridaLakes))
```

Call:
```
lm(formula = AvgMercury ~ pH, data = FloridaLakes)
```

Residuals:
```
  Min 1Q Median 3Q Max
-0.4890 -0.1919 -0.0577 0.0946 0.7113
```

Coefficients:
```
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.5309     0.2035   7.52  8.1e-10
pH         -0.1523     0.0303  -5.02  6.6e-06
```

Residual standard error: 0.282 on 51 degrees of freedom
Multiple R-squared: 0.331, Adjusted R-squared: 0.318
F-statistic: 25.2 on 1 and 51 DF, p-value: 6.57e-06

Of these the most interesting is the one in the row labeled `pH`. This is a test of

- $H_0 : \beta_1 = 0$
- $H_a : \beta_1 \neq 0$

The test statistic $t = \frac{\hat{\beta}_1 - 0}{SE}$ is converted to a p-value using a t-distribution with $DFE = n - 2$ degrees of freedom.

```r
t <- -0.1523 / 0.0303; t

[1] -5.03

2 * pt( t, df = 51 ) # p-value

[1] 6.52e-06
```

We could also estimate this p-value using randomization. If $\beta_1 = 0$, then the model equation becomes

$$\text{response} = \beta_0 + \epsilon$$
so the explanatory variable doesn’t matter for determining the response. This means we can simulate a world in which the null hypothesis is true by shuffling the explanatory variable:

```r
rand.lakes <- do(1000) * lm(AvgMercury ~ shuffle(pH), data = FloridaLakes)
histogram(~pH, data = rand.lakes, v = 0)
2 * prop(~(pH <= -0.1523), data = rand.lakes) # p-value from randomization distribution
```

In this case, none of our 1000 resamples produced such a small value for \( \hat{\beta}_1 \). This is consistent with the small p-value computed previously.

Explanatory and Response Variables Matter

It is important that the explanatory variable be the “x” variable and the response variable be the “y” variable when doing regression.

### 9.2.4 What Does the Slope Tell Us?

The slope is usually more interesting because it tells useful things about the relationship between the explanatory and response variable.

1. If the slope is 0, then our model becomes

   \[
   Y = b_0 + \epsilon \quad \epsilon \sim \text{Norm}(0, \sigma)
   \]

   That is, \( Y \sim \text{Norm}(b_0, \sigma) \) and knowing the explanatory variable does not help us predict the response.

   The test of whether \( \beta_1 = 0 \) is therefore called the model utility test: it tells us whether the model is useful.

2. If the slope is not 0, then the slope tells us how much the average response changes for each unit increase in the predictor.

### 9.2.5 What Does the Intercept Tell Us?

The intercept on the other hand, tells us what happens when the predictor is 0, which may be very uninteresting, especially if the predictor can’t be 0. (Suppose your predictor is your pulse, for example.)
9.3 ANOVA for Regression

We can also think about regression as a way to analyze the variability in the response.

```r
anova(lm(AvgMercury ~ pH, data = FloridaLakes))
```

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pH</td>
<td>1</td>
<td>2.00</td>
<td>2.002</td>
<td>25.2</td>
<td>6.6e-06</td>
</tr>
<tr>
<td>Residuals</td>
<td>51</td>
<td>4.05</td>
<td>0.079</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This is a lot like the ANOVA tables we have seen before. This time:

\[
SST = \sum (y - \bar{y})^2 \\
SSE = \sum (y - \hat{y})^2 \\
SSM = \sum (\hat{y} - \bar{y})^2 \\
SST = SSM + SSE
\]

As before, when SSM is large and SSE is small, then the model (\(\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x\)) explains a lot of the variability and little is left unexplained (SSE). On the other hand, if SSM is small and SSE is large, then the model explains only a little of the variability and most of it is due to things not explained by the model.

The percentage of explained variability is denoted \(r^2\) or \(R^2\):

\[
R^2 = \frac{SSM}{SST} = \frac{SSM}{SSM + SSE}
\]

For our the Florida lakes study, we see that

- \(SSM = 2.00\)
- \(SSE = 4.05\)
- \(SST = 2.00 + 4.05 = 6.05\)
- \(R^2 = \frac{SSM}{SST} = \frac{2.00}{6.05} = 0.331\)

This number is listed as “Multiple R-squared” on the summary output.

So pH explains roughly 1/3 of the variability in mercury levels. The other two thirds of the variability in mercury levels is due to other things. (We can think of many things that might matter: size of the lake, depth of the lake, types of fish in the lake, types of plants in the lake, proximity to industrialization – highways, streets, manufacturing plants, etc.) More complex studies might investigate the effects of several such factors simultaneously.

The correlation coefficient

The square root of this (with a sign to indicate whether the association between explanatory and response variables is positive or negative) is called the correlation coefficient, \(R\) (or \(r\)). Here are some important facts about \(R\):
1. $R$ is always between -1 and 1

2. $R$ is 1 or -1 only if all the dots fall exactly on a line.

3. If the relationship between the explanatory and response variables is not roughly linear, then $R$ is not a very useful number. (And simple linear regression is not very useful either).

4. For linear relationships, $R$ is a measure of the strength of the relationship. If $R$ is close to 1 or -1, the linear association is strong. If it is closer to 0, the linear association is weak (with lots of scatter about the best fit line).

### 9.3.1 Two facts about the regression line

We always compute these values using software, but it is good to note that the least squares line satisfies two very nice properties.

1. The point $(\bar{x}, \bar{y})$ is on the line.
   
   This means that $\bar{y} = b_0 + b_1 \bar{x}$ (and $b_0 = \bar{y} - b_1 \bar{x}$)

2. The slope of the line is $b_1 = r \frac{s_y}{s_x}$.

Since we have a point and the slope, it is easy to compute the equation for the line if we know $\bar{x}$, $s_x$, $\bar{y}$, $s_y$, and $r$.

### 9.4 Making Predictions

#### 9.4.1 Point Estimates for Response

It may be very interesting to make predictions when the explanatory variable has some other value, however. There are two ways to do this in $R$. One uses the `predict()` function. It is simpler, however, to use the `makeFun()` function in the `mosaic` package, so that’s the approach we will use here.

First, let’s build our linear model and store it.

```r
lakes.model <- lm(AvgMercury ~ pH, data = FloridaLakes)
coef(lakes.model)
```

```
(Intercept) pH
1.531 -0.152
```

Now let’s create a function that will estimate values of `AvgMercury` for a given value of `pH`:

```r
mercury <- makeFun(lakes.model)
```

We can now input a pH value and see what our least squares regression line predicts for the average mercury level in the fish:

```r
mercury(pH = 5) # estimate AvgMercury when pH is 5
```
9.4.2 Interval Estimates for the Mean and Individual Response

R can compute two kinds of confidence intervals for the response for a given value

1. A confidence interval for the mean response for a given explanatory value can be computed by adding `interval='confidence'`.

   ```
   mercury(pH = 5, interval = "confidence")
   fit lwr upr
   1 0.769 0.645 0.894
   ```

2. An interval for an individual response (called a prediction interval to avoid confusion with the confidence interval above) can be computed by adding `interval='prediction'` instead.

   ```
   mercury(pH = 5, interval = "prediction")
   fit lwr upr
   1 0.769 0.191 1.35
   ```

Prediction intervals

(a) are much wider than confidence intervals
(b) are very sensitive to the assumption that the population normal for each value of the predictor.
(c) are (for a 95% confidence level) a little bit wider than

\[ \hat{y} \pm 2SE \]

where SE is the “residual standard error” reported in the summary output.

The prediction interval is a little wider because it takes into account the uncertainty in our estimated slope and intercept as well as the variability of responses around the true regression line.

The figure below shows the confidence (dotted) and prediction (dashed) intervals as bands around the regression line.

```
As the graph illustrates, the intervals are narrow near the center of the data and wider near the edges of the data. It is not safe to extrapolate beyond the data (without additional information), since there is no data to let us know whether the pattern of the data extends.

9.5 Regression Cautions

9.5.1 Don’t Fit a Line If a Line Doesn’t Fit

When doing regression you should always look at the data to see if a line is a good fit. If it is not, it may be that a suitable transformation of one or both of the variables may improve things. Or perhaps some other method is required.

Anscombe’s Data

Anscombe illustrated the importance of looking at the data by concocting an interesting data set.

Notice how similar the numerical summaries are for these for pairs of variables

```
summary(lm(y1 ~ x1, anscombe))
```

Call:
`lm(formula = y1 ~ x1, data = anscombe)`

Residuals:
```
          Min 1Q Median 3Q Max
-1.9213 -0.4558 -0.0414 0.7094 1.8388
```

Coefficients:
```
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                  3.000      1.125   2.67  0.0257
x1                           0.500      0.118   4.24  0.0022
```

Residual standard error: 1.24 on 9 degrees of freedom
Multiple R-squared: 0.667, Adjusted R-squared: 0.629
F-statistic: 18 on 1 and 9 DF, p-value: 0.00217

Math 145 : Fall 2013 : Pruim
```r
summary(lm(y2 ~ x2, anscombe))

Call:
  lm(formula = y2 ~ x2, data = anscombe)

Residuals:
     Min      1Q  Median      3Q     Max
-1.9010 -0.7611  0.1289  0.9487  1.2691

Coefficients:
                     Estimate  Std. Error    t value  Pr(>|t|)
(Intercept)        3.001000     1.12490   2.669492   0.025071
x2                  0.500000     0.11825   4.238286   0.002180

Residual standard error: 1.24 on 9 degrees of freedom
Multiple R-squared: 0.6656, Adjusted R-squared: 0.6291
F-statistic: 18 on 1 and 9 DF, p-value: 0.00218

summary(lm(y3 ~ x3, anscombe))

Call:
  lm(formula = y3 ~ x3, data = anscombe)

Residuals:
     Min      1Q  Median      3Q     Max
-1.1590 -0.6152 -0.2300  0.1548  3.2409

Coefficients:
                     Estimate  Std. Error    t value  Pr(>|t|)
(Intercept)        3.001667     1.12409   2.668252   0.025080
x3                  0.500000     0.11825   4.238286   0.002180

Residual standard error: 1.24 on 9 degrees of freedom
Multiple R-squared: 0.6656, Adjusted R-squared: 0.6291
F-statistic: 18 on 1 and 9 DF, p-value: 0.00218

summary(lm(y4 ~ x4, anscombe))

Call:
  lm(formula = y4 ~ x4, data = anscombe)

Residuals:
     Min      1Q  Median      3Q     Max
-1.7510 -0.8310  0.0000  0.8090  1.8390

Coefficients:
                     Estimate  Std. Error    t value  Pr(>|t|)
(Intercept)        3.002000     1.12409   2.668252   0.025080
x4                  0.500000     0.11825   4.238286   0.002180
```

Last Modified: November 25, 2013

Math 145: Fall 2013: Pruim
Regression

9.5.2 Outliers in Regression

Outliers can be very influential in regression, especially in small data sets, and especially if they occur for extreme values of the explanatory variable. Outliers cannot be removed just because we don’t like them, but they should be explored to see what is going on (data entry error? special case? etc.)

Some researchers will do “leave-one-out” analysis, or “leave some out” analysis where they refit the regression with each data point left out once. If the regression summary changes very little when we do this, this means that the regression line is summarizing information that is shared among all the points relatively equally. But if removing one or a small number of values makes a dramatic change, then we know that that point is exerting a lot of influence over the resulting analysis (a cause for caution).

9.5.3 Residual Plots

In addition to scatter plots of the response vs. the explanatory variable, we can also create plots of the residuals of the model vs either the explanatory variable or the fitted values ($\hat{y}$). The latter works in a wider variety of settings (including multiple regression and two-way ANOVA).

```r
model1 <- lm(y1 ~ x1, data = anscombe)
model2 <- lm(y2 ~ x2, data = anscombe)
model3 <- lm(y3 ~ x3, data = anscombe)
model4 <- lm(y4 ~ x4, data = anscombe)

xyplot(resid(model1) ~ x1, data = anscombe)
xyplot(resid(model1) ~ fitted(model1), data = anscombe)
```
You can make similar plots for models 3 and 4. The main advantage of these plots is that they use the vertical space in the plot more efficiently. This is especially important when the size of the residuals is small relative to the range of the response variable.

Returning to our Florida lakes, we see that things look reasonable for the model we have been fitting (but stay tuned for the next section).

```r
lake.model <- lm(AvgMercury ~ pH, data = FloridaLakes)
xyplot(AvgMercury ~ pH, data = FloridaLakes, type = c("p", "r"))
xyplot(resid(lake.model) ~ fitted(lake.model), data = FloridaLakes)
```

We are hoping not to see any strong patterns in these residual plots.
9.5.4 Checking the Distribution of the Residuals for Normality

Residuals should be checked to see that the distribution looks approximately normal and that that standard deviation remains consistent across the range of our data (and across time).

```r
histogram(~resid(lakes.model))
xqmath(~resid(lakes.model))
```

The normal-quantile plot shown above is designed so that the points will fall along a straight line when the underlying distribution is exactly normal. As the distribution becomes less and less normal, the normal-quantile will look less and less like a straight line.

Similar plots (and some others as well) can also be made with

```r
plot(lakes.model)
```

In this case things don’t look quite as good as we would like on the normality front. The residuals are a bit too skewed (too many large positive residuals). Using a log transformation on the response (see below) might improve things.

9.5.5 Transformations

Transformations of one or both variables can change the shape of the relationship (from non-linear to linear, we hope) and also the distribution of the residuals. In biological applications, a logarithmic transformation is often useful.

```r
lakes.model2 <- lm(log(AvgMercury) ~ pH, data = FloridaLakes)
xyplot(log(AvgMercury) ~ pH, data = FloridaLakes, type = c("p", "r"))
summary(lakes.model2)
```

Call:
`lm(formula = log(AvgMercury) ~ pH, data = FloridaLakes)`

Residuals:
```
       Min      10     Median     30      Max
-1.6794 -0.4315   0.0994  0.4422  1.3715
```

Coefficients:

Math 145 : Fall 2013 : Pruim

Last Modified: November 25, 2013
Regression

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | 1.7400     | 0.4819  | 3.61     | 7e-04    |
| pH       | -0.4022    | 0.0718  | -5.60    | 8.5e-07  |

Residual standard error: 0.667 on 51 degrees of freedom
Multiple R-squared: 0.381, Adjusted R-squared: 0.369
F-statistic: 31.4 on 1 and 51 DF, p-value: 8.54e-07

If we like, we can show the new model fit overlaid on the original data:

```r
xyplot(AvgMercury ˜ pH, data = FloridaLakes, main = "untransformed model", type = c("p", "r"))
xyplot(AvgMercury ˜ pH, data = FloridaLakes, main = "log transformed model")
Hg <- makeFun(lakes.model2)  # turn model into a function
plotFun(exp(Hg(pH)) ˜ pH, add = TRUE)  # add this function to the plot
```

A logarithmic transformation of \texttt{AvgMercury} improves the normality of the residuals.

```r
histogram(~ resid(lakes.model2))
qqmath(~ resid(lakes.model2))
xyplot(resid(lakes.model2) ˜ pH, data = FloridaLakes)
xyplot(resid(lakes.model2) ˜ fitted(lakes.model2))
```
The absolute values of the residuals are perhaps a bit larger when the pH is higher (and fits are smaller), although this is exaggerated somewhat in the plots because there is so little data with very small pH values. If we look at square roots of standardized residuals this effect is not as pronounced:

```
plot(lakes.model2, w = 3)
```

On balance, the log transformation seems to improve the situation and is to be preferred over the original model.